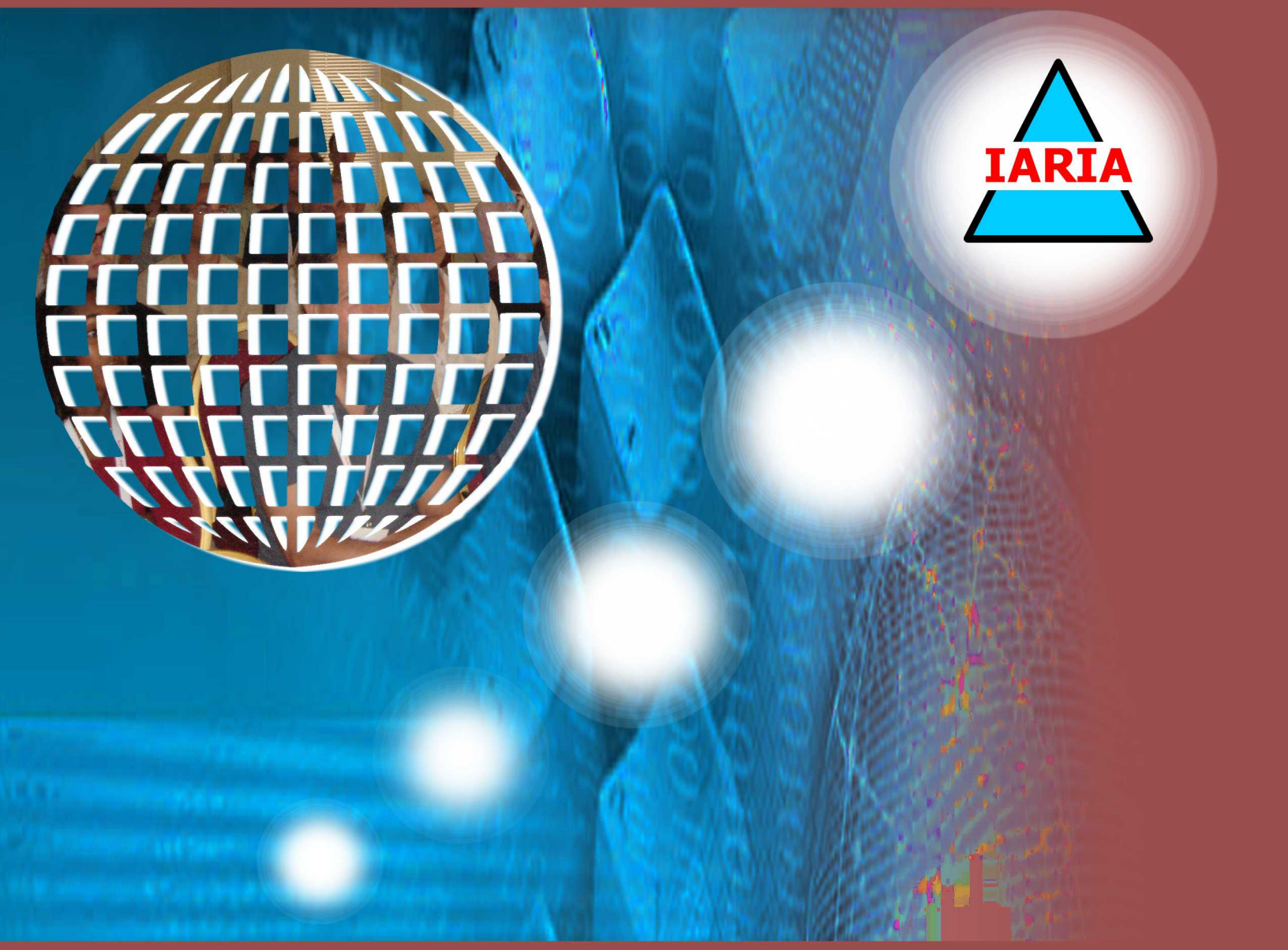
International Journal on Advances in Security



2018 vol. 11 nr. 1&2

The *International Journal on Advances in Security* is published by IARIA.
ISSN: 1942-2636
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Security
Volume 11, Number 1 & 2, 2018

Catalin V. Birjoveanu, "Al.I.Cuza" University of Iasi, Romania
Wolfgang Boehmer, Technische Universitaet Darmstadt, Germany
Alexis Bonnecaze, Université d'Aix-Marseille, France
Carlos T. Calafate, Universitat Politècnica de València, Spain
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Zhixiong Chen, Mercy College, USA
Clelia Colombo Vilarrasa, Autonomous University of Barcelona, Spain
Peter Cruickshank, Edinburgh Napier University Edinburgh, UK
Nora Cuppens, Institut Telecom / Telecom Bretagne, France
Glenn S. Dardick, Longwood University, USA
Vincenzo De Florio, University of Antwerp & IBBT, Belgium
Paul De Hert, Vrije Universiteit Brussels (LSTS) - Tilburg University (TILT), Belgium
Pierre de Leusse, AGH-UST, Poland
William Dougherty, Secern Consulting - Charlotte, USA
Raimund K. Ege, Northern Illinois University, USA
Laila El Aimani, Technicolor, Security & Content Protection Labs., Germany
El-Sayed M. El-Alfy, King Fahd University of Petroleum and Minerals, Saudi Arabia
Rainer Falk, Siemens AG - Corporate Technology, Germany
Shao-Ming Fei, Capital Normal University, Beijing, China
Eduardo B. Fernandez, Florida Atlantic University, USA
Anders Fongen, Norwegian Defense Research Establishment, Norway
Somchart Fugkeaw, Thai Digital ID Co., Ltd., Thailand
Steven Furnell, University of Plymouth, UK
Clemente Galdi, Universita' di Napoli "Federico II", Italy
Emiliano Garcia-Palacios, ECIT Institute at Queens University Belfast - Belfast, UK
Birgit Gersbeck-Schierholz, Leibniz Universität Hannover, Germany
Manuel Gil Pérez, University of Murcia, Spain
Karl M. Goeschka, Vienna University of Technology, Austria
Stefanos Gritzalis, University of the Aegean, Greece
Michael Grottke, University of Erlangen-Nuremberg, Germany
Ehud Gudes, Ben-Gurion University - Beer-Sheva, Israel
Indira R. Guzman, Trident University International, USA
Huong Ha, University of Newcastle, Singapore
Petr Hanáček, Brno University of Technology, Czech Republic
Gerhard Hancke, Royal Holloway / University of London, UK
Sami Harari, Institut des Sciences de l'Ingénieur de Toulon et du Var / Université du Sud Toulon Var, France
Dan Harkins, Aruba Networks, Inc., USA
Ragib Hasan, University of Alabama at Birmingham, USA
Masahito Hayashi, Nagoya University, Japan
Michael Hobbs, Deakin University, Australia
Hans-Joachim Hof, Munich University of Applied Sciences, Germany
Neminath Hubballi, Infosys Labs Bangalore, India
Mariusz Jakubowski, Microsoft Research, USA
Ángel Jesús Varela Vaca, University of Seville, Spain
Ravi Jhawar, Università degli Studi di Milano, Italy
Dan Jiang, Philips Research Asia Shanghai, China

Georgios Kambourakis, University of the Aegean, Greece
Florian Kammueller, Middlesex University - London, UK
Sokratis K. Katsikas, University of Piraeus, Greece
Seah Boon Keong, MIMOS Berhad, Malaysia
Sylvia Kierkegaard, IAITL-International Association of IT Lawyers, Denmark
Hyunsung Kim, Kyungil University, Korea
Geir M. Køien, University of Agder, Norway
Ah-Lian Kor, Leeds Metropolitan University, UK
Evangelos Kranakis, Carleton University - Ottawa, Canada
Lam-for Kwok, City University of Hong Kong, Hong Kong
Jean-Francois Lalande, ENSI de Bourges, France
Gyungho Lee, Korea University, South Korea
Clement Leung, Hong Kong Baptist University, Kowloon, Hong Kong
Diego Liberati, Italian National Research Council, Italy
Giovanni Livraga, Università degli Studi di Milano, Italy
Gui Lu Long, Tsinghua University, China
Jia-Ning Luo, Ming Chuan University, Taiwan
Thomas Margoni, University of Western Ontario, Canada
Rivalino Matias Jr ., Federal University of Uberlandia, Brazil
Manuel Mazzara, UNU-IIST, Macau / Newcastle University, UK
Catherine Meadows, Naval Research Laboratory - Washington DC, USA
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Ajaz H. Mir, National Institute of Technology, Srinagar, India
Jose Manuel Moya, Technical University of Madrid, Spain
Leonardo Mostarda, Middlesex University, UK
Jogesh K. Muppala, The Hong Kong University of Science and Technology, Hong Kong
Syed Naqvi, CETIC (Centre d'Excellence en Technologies de l'Information et de la Communication),Belgium
Sarmistha Neogy, Jadavpur University, India
Mats Neovius, Åbo Akademi University, Finland
Jason R.C. Nurse, University of Oxford, UK
Peter Parycek, Donau-Universität Krems, Austria
Konstantinos Patsakis, Rovira i Virgili University, Spain
João Paulo Barraca, University of Aveiro, Portugal
Sergio Pozo Hidalgo, University of Seville, Spain
Yong Man Ro, KAIST (Korea advanced Institute of Science and Technology), Korea
Rodrigo Roman Castro, University of Malaga, Spain
Heiko Roßnagel, Fraunhofer Institute for Industrial Engineering IAO, Germany
Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance, Germany
Antonio Ruiz Martinez, University of Murcia, Spain
Paul Sant, University of Bedfordshire, UK
Peter Schartner, University of Klagenfurt, Austria
Alireza Shameli Sendi, Ecole Polytechnique de Montreal, Canada
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Pedro Sousa, University of Minho, Portugal
George Spanoudakis, City University London, UK

International Journal on Advances in Security
Volume 11, Numbers 1 & 2, 2018

CONTENTS

Seven Steps to a Forever-Safe Cipher

(An Introduction to Poly-Substitution Encryption)

Julián Murguía Hughes
Independent Researcher
Montevideo, Uruguay
email: jmurguia@montevideo.com.uy

Abstract—**All cryptography currently in use is vulnerable to an attacker with enough computational power and most of them will become obsolete once quantum computing becomes widely available. Continuing the current path seeking for more and more complex algorithms cannot guarantee neither secrecy nor unbreakability. Increasing the complexity while it keeps being vulnerable does not seem to be the right approach. Thinking outside the box is not enough. We need to start looking from a different perspective for a different path to ensure data privacy and secrecy. In this paper, we introduce Poly-Substitution encryption and share advances in searching for unconditional security instead of complexity and we try to light a path to a whole different cryptography based on simplicity and resistant not only to quantum attacks but also to what may come later, including attackers with infinite computational power.**

Keywords-cipher; poly-substitution; unconditional security; perfect secrecy; inifinite computational power; quantum-resistant; cryptography; secrecy; unbreakability; privacy; encryption; quantum; computing; resistant; data.

I. INTRODUCTION

In this work in progress, we show current achievements in the field of cryptography and present some future ideas in this area and their potential. No final results or final data is available at this time.

This work updates, continues and expands our paper "Seven Steps to a Quantum-Resistant Cipher" presented at SECURWARE 2016, The Tenth International Conference on Emerging Security Information, Systems and Technologies; held in Nice, France – July 24-28, 2016 [1].

Since the beginning, cryptography has worked the same way; you take the original source of information (the plaintext), a key and a fixed substitution and you apply the substitution using the plaintext and the key as input to generate the cryptogram or ciphertext as its output. Modern cryptography keeps working in the exact same way.

A. Definitions

To set a common ground and avoid confusions and misunderstandings, a minimum set of definitions is required and listed here:

Symbol. A symbol is a representation of something. From a single character in any given language like English or an ideogram in Chinese to an abstract concept like π representing the relation between a circumference and its diameter, which is a numeric value with infinite decimal values never repeating.

Alphabet. An alphabet is a finite set of symbols listed in a given order.

Shifted alphabet. It is an alphabet where the symbols are shifted place by a given number of positions from the original order and the alphabet is considered circular for the shifting process, where the first symbol follows the last one and the last symbol precedes the first one.

Mixed or Permuted alphabet. It is an alphabet where the order of the symbols is arbitrarily mixed or permuted from the original order.

Word. A word is a finite sequence of symbols in an arbitrary order where not all the symbols from the alphabet need to be present and any symbol may appear more than once. The meaning of a word does not depend on the order of the symbols within the alphabet.

Phrase. A phrase is a finite sequence of separated words.

Text. A text is a finite sequence of separated phrases.

Dictionary. A dictionary is a text listing all valid words and using phrases to define the meaning of each one.

Plaintext. The original unencrypted text or message.

Ciphertext. The result of encrypting the plaintext.

Unconditionally Secure System. We will use the definition given by Whitfield Diffie and Martin E. Hellman [2] as they stated that "a system that can resist any cryptanalytic attack, no matter how much computation is allowed, is called unconditionally secure".

B. Caesar Cipher

Although the first known evidence of some form of cryptography is almost four millennia old [3], one of the oldest known forms of encryption is the Caesar's cipher. It was a substitution cipher where each character was replaced for the one located three places later in alphabetic order and considered the alphabet as a round circle as it is shown in Figure 1, where 'A' follows 'Z' and so, 'X' would be replaced by 'A', 'Y' would be replaced by 'B', 'Z' would be replaced by 'C', 'A' would be replaced by 'D' and so on.

To encrypt or cipher a plaintext using Caesar's cipher, each character from the plaintext is replaced by the one placed three positions moving clockwise. To decrypt or decipher a ciphertext, each character from the ciphertext is replaced by the one located three positions moving counter clockwise.

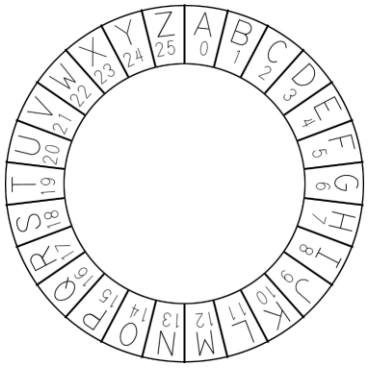


Figure 1. Circular positional alphabet and position values.

The Caesar's algorithm was just a shift by places process and the key used was just three, indicating the algorithm that each character in the plaintext needed to be shifted by three to generate the cryptogram. All shift by places encryption algorithms are generically referred as Caesar ciphers. As in this type of cipher each letter is replaced always by the same letter, it is called a mono-alphabetic substitution cipher.

The Caesar's cipher can be represented using modular arithmetic. Modular arithmetic is a system of arithmetic for integer numbers where values wrap around upon reaching a maximum value.

To represent Caesar's cipher using modular arithmetic, we start by assigning a numeric value to each letter from the alphabet according to their position within such alphabet. In the classic English alphabet and its standard alphabetic order, to the letter "A" corresponds the value 0 (zero), to the letter "B" corresponds the value 1, and so up to the letter "Z" with a value of 25. Figure 2 shows a traditional positional alphabet and the numeric value associated to each letter of such alphabet.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Figure 2. Positional Alphabet and Position Values.

As the maximum value is 25 and the value 26 wraps around to zero, the modulus for the Caesar's cipher will be 26.

The substitution of any letter (x) for the one located *n* places to the right can be represented through the mathematical formula:

$$E_n(x) = (x + n) \bmod 26 \quad (1)$$

For the original Caesar's cipher, the formula to cipher would be:

$$E_3(x) = (x + 3) \bmod 26 \quad (2)$$

The reverse deciphering process can be represented through the mathematical formula:

$$D_n(x) = (x - n) \bmod 26 \quad (3)$$

For the original Caesar's cipher, the formula would be:

$$D_3(x) = (x - 3) \bmod 26 \quad (4)$$

Although it is considered obsolete and today it can be broken without the need of a computer, just with pencil, paper and some spare time, it lasted for centuries.

C. Vigenère Cipher

In 1553, Italian cryptologyst Giovan Battista Belasso described in his book [4] a new cipher later attributed to Blaise de Vigenère and which is still known as the Vigenère cipher.

This cipher, instead of using a single key value for the substitution, uses a sequence of letters so that instead of performing a mono-alphabetic substitution, performs what is called a variable or poly alphabetic substitution, where each letter may produce a different result.

Vigenère's encryption is functionally based on the use of the tabula recta, invented by German monk Johannes Trithemius in 1508, which is a square table of alphabets where each row is made by shifting the row above one position to the left.

To cipher, the plaintext character to be encrypted is looked into the table's first row, the character from the key to be used is looked into the table's first column and the ciphertext character will be the one located at the intersection of the column corresponding to the plaintext character with the row corresponding to the key character. To decipher, the key character is looked into the first column and then that row is looked for the position of the ciphertext character. Once found, the character at the top of this column will be the plaintext character.

Figure 3 shows the Tabula Recta created by Johannes Trithemius, which is also called the Vigenère table.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 3. Vigenère Table.

The Vigenère cipher can also be represented using modular arithmetic, assigning a value to each letter, the same way we did with the Caesar's cipher in Figure 1 and also in Figure 2.

To cipher, for each letter in the plaintext message and the corresponding key letter, the alphabet position values are added using modular addition module 26 and the resulting value will indicate the position of the ciphertext letter corresponding to the result.

Being *x* the letter in position *n* within the plaintext message we want to cipher and *k* the corresponding letter from the key to be used, the cipher process can be represented using the following formula:

$$E(x_n) = (x_n + k_n) \bmod 26 \quad (5)$$

To decipher, the formula would be:

$$D(x_n) = (x_n - k_n) \bmod 26 \quad (6)$$

The Vigenère cipher is known for being easy to be understood and implemented and hard to break. It lasted for almost four centuries, as we will see when we address its vulnerability in Section II.

D. Vernam Cipher

About a century ago, Gilbert Vernam invented an encryption technique [5] (Patent US 1310719 [6]) that thirty-something years later Claude Shannon proved [7] it offered Perfect Secrecy and properly used will remain impervious to any attack no matter how powerful the attacker may be, including quantum computing and even an attacker with infinite computational power. It is not used because it requires the key to have the same length as the plaintext, to be truly random and not to be reused. Those constraints were considered and are still considered strong enough to prevent its usage.

As today's information is always measured in bytes or multiples of byte sizes (Kilobytes, Megabytes, Gigabytes, Terabytes, etc.) for all the explanations and examples here, the byte as the basic unit of information will be used. Considering the byte as just a group of eight bits, being a bit a binary digit that can either have a value of zero (0) or one (1).

A single byte can represent 256 different values, from 0 to 255 in decimal notation, from 00 to FF in hexadecimal notation and from 00000000 to 11111111 in binary format representation.

For a byte, the Vernam cipher will perform exactly the same way as for a single bit, it will use the XOR function between the plaintext byte and the key byte. The behavior of the function is simple, it will compare each bit within the byte from the plaintext to the bit in the same position in the byte from the key and will generate a bit with a value of zero if both bits have the same value and one if they are different. This XOR function will return the cryptogram or ciphertext byte as its result. For a specific plaintext byte value, each of the 256 possible values of the key byte will produce a different ciphertext byte value.

If you get the cryptogram or ciphertext byte and do not know the value of the key byte, every single possible value of the key byte has the exact same probability of being the right one and you have no way to decide which one of them is the right one and thus, which of the 256 possible values of the plaintext byte is the right one.

There is no possible cryptanalysis of this process and a brute force attack will end up with the plaintext mixed with a huge number of false positives (apparently valid results that are not the original plaintext) with no way to tell which one is the original one.

Shannon proved that even knowing that the plaintext is just text, any possible text with the same length has the exact same probability of being the original plaintext [8].

Since then, algorithms have grown in complexity looking to enhance the security of the process and to make harder to recover the plaintext without knowing the key.

But what has not changed is the logic, i.e., the way it is done. Cryptography is still using an algorithm with a fixed set of instructions that will use the plaintext and the key as input to produce the ciphertext. The same plaintext and the same key will always produce the same cryptogram.

There are two main attacks to try to get the plaintext without knowing the key: Cryptanalysis (analyze the process trying to find weaknesses or shortcuts that may allow to retrieve the original information without having the key) and Brute Force (try all possible keys).

Modern cryptography is not unbreakable and bases its security on two premises:

1) Cryptanalysis is not possible or too complex to be achieved.
2) Brute Force attacks require too much time and computational power.

In this paper, we will prove that Caesar and Vernam ciphers are just reduced or limited versions of the Vigenère cipher; we will introduce our proposed poly-substitution encryption technique and the seven steps to build cryptographic algorithms based on it and also prove that the Vigenère cipher is a mono substitution cipher and a reduced or limited version of our proposed encryption.

The rest of this paper is organized as follows. Section II will analyze the Vigenère cipher and prove Caesar and Vernam ciphers are just limited or reduced versions of it and will also explain why we consider the Vigenère cipher as a poly alphabetic mono substitution cipher. Section III describes data persistence, cryptography state of the art and their vulnerability to quantum attacks. Section IV introduces poly-substitution encryption, its theory, basis, definitions and how it works. Section V describes each of the seven steps we defined to build a poly-substitution cipher based on our proposed encryption technique. Section VI analyzes a cipher constructed using our proposed encryption and presents an example of it and its results. Section VII compares this sample

cipher against Vigenère, Vernam and other current standards. Section VIII analyzes all possible attacks to our proposed encryption. Section IX describes how partial data universes are handled by symmetric and public key cryptography and their limitations to offer Format Preserving Encryption (FPE); it will also show how our proposed technique handles them better. Section X describes how our proposed poly-substitution encryption can offer and provide unconditional security. Section XI addresses practical applicability. In Section XII we address key and message distribution taking advantage of the use of internet. Section XIII describes the conclusions and Section XIV describes the future work and goals.

II. THEY ARE ALL VIGENÈRE

A. Caesar is Vigenère

It is trivial to prove that any generic Caesar cipher is a reduced or limited version of the Vigenère cipher, where the key is just one symbol or character long.

B. Vernam is Vigenère

At the level of one bit, modular addition module 2, modular subtraction module 2 and XOR, all behave the same way and are in fact the exact same operation as it is shown in Figure 4.

(p + k) mod 2		k	
		0	1
p	0	0	1
	1	1	0

(p - k) mod 2		k	
		0	1
p	0	0	1
	1	1	0

p XOR k		k	
		0	1
p	0	0	1
	1	1	0

Figure 4. One-Bit Binary Operations.

Based on that, it is trivial to prove that the Vernam cipher is a reduced or limited version of the Vigenère cipher, where the alphabet has only two different symbols or characters.

C. Vulnerability of the Vigenère Cipher

In 1863, Friedrich Wilhelm Kasiski published a book about cryptography [9], where he described a method for cryptanalysis or cryptographic attack based on the existence of repeated sequences within the ciphertext. He assumed those repetitions were caused by a key shorter than the message and that they represented repeated sequences within the plaintext encrypted using the same portion of the key.

The Kasiski method argued that the distance between repeated sequences was a multiple of the length of the key. Starting from that, searched for multiple repeated sequences, measured the distances between repetitions and calculated the greatest common divisor to find a value that will be the length of the key or a multiple of it. Once the length of the key is obtained, the ciphertext is divided into blocks of that size and sub-cryptograms are formed by taking the first character of each block, then the second one and so on. Each one of those sub-cryptograms will have been encrypted using the same symbol or letter and so each one of them will be a mono-alphabetic substitution and so we would be able to perform a simple statistical frequency analysis attack.

The Kasiski statistical frequency analysis attack is based on two pillars:

- The known statistical distribution of the letters in a regular text.
- The known distance between the letters from the alphabet.

The most common letters in English are letter “E”, letter “T” and letter “A”, in that order; and is known that letter “E” is the fifth letter of the alphabet and its value is 4, the letter “T” is in position 19, 15 positions to the right of letter “E”, and letter “A” is in position 0, 7 positions to the right of letter “T”.

The Kasiski statistical frequency analysis attack will search in each sub-cryptogram the frequency distribution of the encrypted letters, focusing on those with highest frequencies (those who should correspond to the letters “E”, “T” and “A”) and that also comply with the alphabet structure and the distance between the most common letters within the alphabet. As letter “E” is in position 4, the following formula is true:

$$\text{Key} = \text{cipher letter} - \text{E} = \text{cipher letter} - 4 \quad (7)$$

So, the relative position of letter “E” on each sub-cryptogram will form the key ciphered through a substitution cipher like Caesar’s with a displacement of 4 positions to the right. From it, the key could be retrieved by a simple Caesar decryption using a displacement value of 4.

Statistical analysis is also used to search for repeated n-grams (known sequences of letters, “e.g.”, bigrams “TH”, “HE”, “IN”; trigrams “THE”, “AND”, and so on).

The use of a mixed or permuted alphabet only adds some extra work but it is still vulnerable.

Christopher Swenson, on his book [10], explains the Index of Coincidence (IC) defined by William F. Friedman as a measure of how evenly distributed the character sequences are within the frequency distribution table.

He considered "The Complete Works of William Shakespeare" as an adequate representation of the English language and calculated its IC to be approximately 0.0639.

He defines theoretically perfect IC as if all characters occurred the exact same number of times so that none was more likely than any other to be repeated, so, for an alphabet of 26 characters, he calculated it to be 1/26, which is approximately 0.03846.

IC can also be used with bigrams (sets of two characters) and trigrams (sets of three characters) to measure how evenly distributed they are within their corresponding distribution tables.

The theoretically perfect IC for bigrams is approximately 0.0015 (1/26*26) and for trigrams is approximately 0.00006 (1/26*26*26).

In English, a maximum of 676 bigrams and 17,576 trigrams can exist, although not every one of them may be valid.

The three most common bigrams HE, TH and IN have an IC of about 0.035, 0.034 and 0.0189 respectively. The most common trigram THE, has an IC of about 0.022.

William F. Friedman [11] presents a practical example using IC to break a poly alphabetic mono substitution encryption when the key has been reused.

The conditions set by Shannon to the Vernam cipher for it to offer unconditional security (The key should be as long as the plaintext, it should be random and it should not be used again) makes it a One-Time-Pad and the same applies to the Vigenère cipher. A plaintext encrypted using the Vigenère cipher using a random key with the same length as the plaintext that is not used again offers the same unconditional security defined by Shannon.

D. Mono Substitution

Although being the Vigenère cipher a poly alphabetic one and considering it can be used in reverse and use the decryption process to encrypt and the encryption process to retrieve the original plaintext, the substitution used along the encryption or decryption processes is always the same on each instance.

Each symbol or character is processed using the exact same substitution, modular addition for the encryption and modular subtraction for the decryption.

That is the reason why we call the Vigenère cipher a poly alphabetic mono substitution cipher using mono substitution encryption.

III. STATE OF THE ART

A. Information and Data Persistance

Something that is not directly related to cryptography but needs to be considered together because it has a direct impact on the information life cycle is the persistence of any data or information digitally transmitted or stored. Any information digitally transmitted or stored, persists.

Transmitted data leaves traces and copies between the source and the destination. Even encrypted information, transmitted through secure connections travels from router to router, from server to server from the point of origin to the destination point, and it can be copied in travel without being noticed.

Stored data also leaves copies behind. To totally delete specific data is very but very hard and cannot be assured nor guaranteed. Computer forensic tools are capable of retrieving information believed to have been deleted.

Automatic backups, storage cache, redundant storage and the cloud also help to the persistence of the information.

Two simple and clear examples of information persistence are:

- A picture uploaded into a social network remains there even after the uploaders believe they deleted them.
- Data from no longer available internet servers or storages can still be found in web search engines' caches.

B. Cryptography

According to the European Telecommunications Standards Institute (ETSI), "Without quantum-safe encryption, everything that has been transmitted, or will ever be transmitted, over a network is vulnerable to eavesdropping and public disclosure" [12].

But the privacy concerns go beyond that, once a hacker breaches the security of a system or organization, the information stored there is usually not encrypted. Wikileaks, the Panama Papers, the NSA breach and the World Anti-Doping Agency (WADA) medical records disclosure are clear examples of that.

Encrypting sensitive information within a database is not an easy or low cost task and once a hacker has gone beyond the system security, everything there is at hand and readable. We will come back this topic later when we address Format Preserving Encryption related to partial data universes in Section IX.

Discussion and comparison between symmetric and public key cryptography currently in use becomes irrelevant once one understands that none of them is unbreakable and that anything encrypted with any of them can be read if the attacker has enough computational power. Something that will happen sooner than later.

Public key algorithms such as RSA (Rivest, Shamir and Adleman), ECC (Elliptic Curve Cryptography), Diffie-Hellman and DSA (Digital Signature Algorithm) will be easily broken by quantum computers using Shor's algorithms [13] and so, they are deemed to be insecure to quantum computing.

Symmetric algorithms as AES (Advanced Encryption Standard) [14] or Blowfish [15] are believed (but not proven) to be resilient against quantum attacks by doubling the key length.

Cecilia Boschini, from IBM's Zurich Research Laboratory, was overwhelming during her presentation in IBM's annual conference Think 2018, when she emphatically affirmed that "The security our current cryptography is based, are solvable with a quantum computer".

During his talk at RSA 2018 Conference held in San Francisco, CA, USA, Konstantinos Karagiannis, CTO of Security Consulting, BT Americas, estimated that symmetric algorithms (DES, AES) with 512-bit key lengths will fall first, when the number of qubits surpasses 100.

According to Sergey Lurye from Kaspersky's Lab blog [16], "We may forecast that symmetric encryption with 512-bit keys might finally get breached by a hypothetical 144-qubit Bristlecone (Google's latest quantum processor) descendant sometime in late 2019."

Even theoretical Quantum Key Distribution (QKD) has been proved vulnerable to eavesdropping.

Any cipher that bases its strength on its complexity and in the assumption of the unavailability of the computational

power required for an attack, will eventually be broken and persisting on this way will only provide a false sense of security that will last briefly.

C. Post-Quantum Cryptography

Post-Quantum Cryptography is still theoretical and far from being available.

Johannes A. Buchmann, Denis Butin, Florian Göpfert and Albrecht Petzoldt from the Technische Universität Darmstadt in their paper "Post-Quantum Cryptography: State of the Art" [17] ask and answer the question. How far is post-quantum cryptography? Their answer; "There are many promising proposals some of which are rather close to becoming practical".

Some theoretical and practical advances in Quantum Cryptography had already been proved to be vulnerable even to current non-quantum computers.

D. Vulnerability

By definition, all the cryptography in use nowadays is vulnerable to an attacker with enough computational power.

The matter is not if they can be broken but when will this happen.

Cryptography and cryptographers have been racing the Red Queen's race for a very long time. Like Alice in Lewis Carrol's "Through the Looking-Glass" [18], cryptographers have been taking all the running they can do, just to keep in the same place.

All used encryption algorithms are just temporary solutions that will eventually be rendered obsolete. A clear example of this is the Data Encryption Standard (DES) [19], it became a standard in 1977 and was broken by brute force in 1999. The Advanced Encryption Standard (AES) became a standard in 2001 and is already 17 years old.

All this because, with the only exception of the Vigenère and Vernam ciphers properly used, none of the currently used encryption solutions can answer yes to the simple question: Can this cipher resist an attacker with infinite computational power?

It is not sure whether any of the new ciphers and algorithms being developed, including those considered to be post-quantum ones, can answer yes to that same question or not.

Cryptographers will continue running the Red Queen's race as far as they continue to design complex but breakable algorithms offering only conditional and temporary security that will eventually be rendered obsolete.

E. Quantum Computingy

Quantum Computing is computing using quantum mechanics and is a field that was initiated by the work of Paul Benioff [20] and Yuri Manin [21] in 1980, Richard Feynmann [22] in 1982 and David Deutsch [23] in 1985. Current digital computers use data encoded into binary digits or bits, which can have only one value or state (0 or 1). A Quantum Bit or Qubit can have a value of 0, or 1 or 0 and 1, all at the same time.

In May 2016, International Business Machines (IBM) publicly announced they will grant access through their cloud to one of their 5 qubit quantum computers for everyone to run programs or just play with it, as a way to motivate, encourage and accelerate innovation.

In October 2017, Edwin Pednault, John A. Gunnels, Giacomo Nannicini, Lior Horesh, Thomas Magerlein, Edgar Solomonik, and Robert Wisnieff presented their paper "Breaking the 49-Qubit Barrier in the Simulation of Quantum Circuits" [24]. There they present calculations that were previously thought to be impossible due to impracticable memory requirements.

In November 2017, the MIT Technology Review informed and commented IBM's announcement of a 50-qubit commercial quantum computer [25].

In March 2018, Google introduced their new Bristlecone quantum processor with 72 qubits.

They are not the only ones on the field. Most governments and cutting edge technological companies and universities around the world, are dedicating time and effort in researching and investing in the development, design and manufacturing of quantum computers.

On May 2016, the European Commission announced €1 billion quantum technologies flagship project for the next ten years with the objective to reinforce European scientific leadership in quantum research and in quantum technologies.

Canadian company D-Wave [26] is already manufacturing quantum computers with two thousand qubit processors (the D-Wave 2000Q™ System) and they continue improving, growing and expanding their processors.

According to CBC News, big names in the worlds of big brains and cutting edge technology like Google, NASA, Lockheed Martin and Los Alamos National Laboratory, among others, are investing big money into this company.

The Los Alamos National Laboratory's magazine 1663, on its July 2016 edition [27], published a very interesting article titled "Not Magic… Quantum", telling about a nascent commercial quantum computer that arrived to their facilities and may solve certain problems with such astonishing speed that it would be like pulling answers out of a hat.

Bjoern Lekitsch, Sebastian Weidt, Austin G. Fowler, Klaus Mølmer, Simon J. Devitt, Christof Wunderlich and Winfried K. Hensinger published the blueprint for a microwave trapped ion quantum computer in Science Advances magazine in February 2017 [28].

What we hope to achieve is to provide a technique to create ciphers offering perfect unconditional security against eavesdroppers no matter how arbitrarily powerful they may be or become in the future and without the constraints the Vigenère and Vernam ciphers have.

We want to provide a technique to create ciphers with perfect unconditional security against arbitrarily powerful eavesdroppers even if they have infinite computational power.

Something none of the currently in use standards and solutions can offer.

IV. POLY-SUBSTITUTION ENCRYPTION

A. Multiple Substitutions as Part of the Encryption

In mono substitution encryption, the ciphertext is usually referred as the result of applying the key to the plaintext and this is not exact, which leads us to our first definition:

Definition 1. In substitution encryption, each ciphertext character is the result of applying the defined substitution and the corresponding key character's positional value to the plaintext character.

Based on that, we define the ciphertext as follows:

Definition 2. The ciphertext is the result of applying a sequence of pairs formed by the substitution and the key value to each symbol or character from the plaintext until the plaintext is exhausted.

It is crucial to understand that the ciphertext is not just the result of applying the key to the plaintext but the result of applying the sequence of pairs formed by each substitution and each key value used.

When the key is shorter than the plaintext, it wraps up at the end starts to repeat. In fact, in mono substitution encryption what starts to repeat is the same sequence of pairs formed by the substitution and the key value.

B. Multiple substitutions

Vigenère used modular addition as the substitution for encrypting and modular subtraction as the substitution for decrypting and those two substitutions are different, as the following example shows:

$$(17 + 23) \bmod 26 \neq (17 - 23) \bmod 26 \quad (8)$$

$$(17 + 23) \bmod 26 = 14 \quad (9)$$

$$(17 - 23) \bmod 26 = 20 \quad (10)$$

We used module 26 because Vigenère used an alphabet with 26 different symbols or characters, but an alphabet may include any number of symbols or characters with a minimum of two.

From now on, we will consider a generic alphabet called A that contains a number equal to a of different symbols or characters, being $a \geq 2$.

Formulas (5) and (6) will be generically expressed as:

$$E(x_n) = (x_n + k_n) \bmod a \quad (11)$$

$$D(x_n) = (x_n - k_n) \bmod a \quad (12)$$

Being v and v' two integer variables with values from 0 to a–1, the following formulas are always true:

$$(x_n + k_n + v) \bmod a = (x_n + k_n + v') \bmod a, \quad \text{for } v = v' \quad (13)$$

$$(x_n + k_n + v) \bmod a \neq (x_n + k_n + v') \bmod a, \quad \text{for } v \neq v' \quad (14)$$

We will represent Vigenère's encryption through the mathematical formula:

$$E(x_n) = (x_n + k_n + v) \bmod a \quad (15)$$

Vigenère's decryption through the mathematical formula:

$$D(x_n) = (x_n - k_n - v) \bmod a \quad (16)$$

So far, we have as many different encryption and decryption substitutions as symbols or characters are present in the alphabet, and as any decryption substitution can be used for encryption, we have in fact twice as many substitutions as symbols or characters in the alphabet.

If we take into consideration another mono substitution cipher as it is the Beaufort cipher, created by Sir Francis Beaufort, we can get another set of substitutions.

The Beaufort cipher, created by Sir Francis Beaufort, is a variation of the Vigenère cipher where, plaintext is subtracted from the key in order to obtain the ciphertext.

The logic is similar, only a different substitution is used, as it is shown in the following formula:

$$E(x_n) = (k_n - x_n) \bmod a \quad (17)$$

Applying what we have seen, such formula will become the following one:

$$E(x_n) = (k_n - x_n + v) \bmod a \quad (18)$$

The main difference here is that while in the Beaufort cipher the encryption and decryption substitution is the same (the plaintext is subtracted from the key to obtain the ciphertext and the ciphertext is subtracted from the key to recover the original plaintext), for $v \neq 0$, the formula for the decryption substitution will be:

$$D(x_n) = (k_n - x_n - v) \bmod a \quad (19)$$

For $v = 0$, the encryption and decryption substitutions are both the same.

This proves the existence of far more available encryption substitutions than symbols or characters in any given alphabet.

C. Multiple Alphabets

The use of more than one alphabet is possible, each with its own set of substitutions and the substitution to be used selected according to the alphabet the plaintext belongs to.

Suppose there are two alphabets, one with the letters A…Z and another one with the numbers 0…9, that way the encryption process may allow to encrypt letters into letters and numbers into numbers in a single pass.

We will come back to this later in Section IX.

D. Fixed Substitution Sequences

Arbitrary Sequences of substitutions can be built up with any given length, using any available substitution and placing them in any order. Each substitution may be used more than once and not all of them require to be used.

As with the key, the substitution sequence wraps up at the end when the plaintext is longer than the sequence.

If the key and the substitution sequence have the same length, then the same sequence of pairs formed by the substitution and the key value will repeat and that will make the whole encryption process vulnerable to a statistical analysis attack.

But, if the key and the substitution sequence have different lengths, when the key starts to repeat, the substitution sequence will not be the same and so there will be no sequence pair of substitution and key value at least until a position is reached within the plaintext equal to the less common multiple of the lengths of the sequence and the key.

If the less common multiple of the lengths of the substitution sequence and the key is larger than the length of the plaintext, unconditional security will be achieved, but only on such case.

E. Variable Substitution Sequences

The best and simplest way to get a variable substitution sequence is to get a fixed one and make it variable.

When hardware is constructed or software is written, the substitutions to be used are listed and located in a given order.

A list of n items can be ordered into n! (n! = 1x2x…xn) different orders and two different orders will produce two different ciphers.

Once the list is built up, the order they are listed in will not change, so, to make it variable we need an additional parameter.

There are two types of parameters we may use for that.

- External Substitution Sequence
  Instead of hardcoding the substitution sequence within the encryption process, it can be an external parameter. Doing that will allow the encryption process to use a different substitution sequence on each run. Each item on this substitution sequence will indicate which substitution will be used on each instance.
- Order Changing Parameter
  Suppose there are n different substitutions used and listed in a given order and they are numbered from 0 to n-1. That means there are n! different possible orders of the numbers from 0 to n-1. One of those permutations is loaded into an array and used as an external parameter. Each element of such array will point to a specific substitution from the list.

Using the same external substitution sequence with a different order changing parameter will produce a different substitution sequence to be used.

If every time the key and/or the substitution sequence is exhausted a new order changing parameter is used, it may be guaranteed that there will be no sequence pairs of substitution and key value repetition no matter how long the plaintext may be. We will come back to this later.

F. Variable Processing Blocks

With the substitutions we have seen so far, what any attacker will know for sure is that the first character in the ciphertext corresponds to the first character in the plaintext and so on up to the last character.

This can be avoided in a simple and elegant way. Another external parameter is used to specify a block size used to process the plaintext. As a mode of example, the block size parameter is used to define how many symbols or characters will be read at once from the plaintext and then processed in reverse order, from the last symbol or character to the first within the defined block. If the remainder of the plaintext is shorter than the last block, the block size is adjusted accordingly.

This external parameter can be a single block size or a list of different block sizes to be used along the encryption process.

Even if an attacker gets the encryption process, it provides no information about the external parameters used and so there is no way to match the plaintext symbol or character order with the ciphertext symbol or character order.

G. Poly Substitution Encryption

Now we can define what we understand for poly substitution encryption and decryption:

Definition 3. Poly Substitution Encryption is encrypting in such a way two or more different substitutions are used in sequence among the key to produce a ciphertext from the plaintext.

Definition 4. Poly Substitution Decryption is decrypting in such a way two or more different substitutions are used in sequence among the key to retrieve the original plaintext form the ciphertext.

V. THE SEVEN STEPS

We defined the process to build up ciphers based on our technique as a step by step process comprised of 7 steps.

A. Step One (Use Multiple Encryption Substitutions)

While Vigenère used modular addition module 26 as the substitution and Vernam used a single function (XOR) as the substitution. Our approach will use many of them. Each substitution will take a plaintext character and a key character and will return a ciphertext character and for each of the possible key character values will return a different ciphertext character.

Using multiple substitutions provides additional security because, if an attacker has a cryptogram or ciphertext character or symbol, not only the key character used is unknown, but also the substitution used.

For a given plaintext byte character, any valid substitution should return different results for each possible value of the key character.

When the plaintext's length is larger than one character we can use one substitution to process the first byte, the same or a different one to process the second character and so on. That leads us to the following step.

B. Step Two (Use a Third Parameter)

The first two parameters will be the plaintext and the key.

A third parameter will be used to indicate which substitution to use on each instance.

A value from this third parameter will indicate which one of the many available substitutions will be used to process a character or symbol from the plaintext and one from the key.

Let us say we decide to use the same number of different substitutions from all that can be created as the number of symbols or characters in the used alphabet. In such case, we will only need one character from this third parameter to indicate which of those substitutions will be used for these specific plaintext and key characters. So far, the third parameter character value x will trigger substitution z. How do we know which of the available substitutions is substitution z, is explained in the next step.

C. Step Three (Order of the Substitutions)

When we have many different substitutions, we need to identify them somehow and make a list of them.

This list is what will be used to decide which substitution will be triggered by which value from the third parameter.

As previously indicated, this list is not unique and a different substitution order will produce a different ciphertext for the same plaintext and key.

Now, an attacker not only needs to try every possible key, also needs to try every possible third parameter and guess which substitution is triggered by each possible value of the third parameter, assuming the selected substitution order is hardcoded within the process.

So far, parameter byte value x will always trigger substitution z, unless we can make parameter value x trigger substitution w in a different run.

The order of the substitutions can be changed, as explained in the next step.

D. Step Four (Changing the Order of the Substitutions)

How do we make third parameter byte value x to trigger a substitution different from substitution z?

The solution is both simple and elegant.

We add a fourth parameter. One of those n! possible orders of the numbers from 0 to n-1 is loaded into a n element array, and value x is used to point to the array's element whose value will be used to trigger the substitution.

A different fourth parameter will provide a different substitution order.

Now, third parameter character value x will trigger a substitution depending on the x^{th} element of the fourth parameter.

So far, any attacker would know that the first byte from the ciphertext corresponds to the first byte of the plaintext, the second byte from the ciphertext corresponds to the second byte of the plaintext, and so on.

The next step will show how to change that.

E. Step Five (Block Processing)

Let us take a block of characters of a given length from the plaintext and process it in reverse order, starting from the last symbol or character in the block, processing it and saving it as the first character in the ciphertext block. Then the previous to the last to be the second character in the ciphertext block and so on, until we end processing the block by processing its first character and then continue with the next block.

The last block may be shorter but it is equally processed from last character to first one as any other block without any need of any padding or additional dummy information to be added.

Now, unless the attacker knows the exact length of the block used, there is no way to know from where to start to retrieve from the ciphertext to obtain the original plaintext in the original order.

F. Step Six (Key Length and Key Repetitions)

So far, no mention has been made of the key length.

When encrypting, the process uses two items with the plaintext: the key and the encryption substitution. So, for each portion of the plaintext, a key-substitution pair is used. This is usually ignored due to the encryption substitution being always the same.

Vigenère's and Vernam's ciphers require the key to have at least the same length as the plaintext for them to offer unconditional security. If the key is shorter, the process starts to repeat the same key-substitution pair sequence and this weakens its security and makes a statistical distribution analysis attack feasible.

If we use a key shorter than the plaintext it will wrap up at the end, but unless the key and the third parameter both have the exact same length, there will be no same key-substitution pair sequence repetitions until we reach a position within the plaintext equal to the least common multiple of the lengths of both the key and the third parameter. As it may eventually happen the whole process would be vulnerable unless we find a way to avoid repetitions.

The solution is, once again, simple and elegant. When the end of the key is reached (or the end of the third parameter or the less common multiple of both lengths, or at any point between them), before starting to repeat it, the process changes the substitution order by modifying the elements in the array explained in step four.

One way to do it is to use the last used substitution and last used plaintext or ciphertext symbol or character and apply that transformation to each element of the array using the element's content and the plaintext character as input and replacing the content of the element with the result, thus obtaining a different permutation of the array elements.

Each time this happens, the change process behaves differently and a different permutation is obtained. Now, even if the key and the third parameter have the exact same length and they start to repeat in the exact same order, the sequence of key-substitution pairs triggered will not be the same and so no repetitions will occur. The same third parameter value will point to the same array element but a different substitution will then be triggered because the content of the array element will have changed and so the second parameter-substitution pair sequence will be totally different.

G. Step Seven (Make Lengths Variable)

Current encryption standards use fixed length blocks and fixed length keys (they may offer different key sizes but with very limited pre-defined fixed sizes).

Our solution allows for user selected lengths for the key, the third parameter and the processing block (or blocks). Two successive encryptions may use not only different keys but also the lengths of both keys may be different. The same applies to the third parameter and also the processing block size may be different. The key length may go from a single character to any length, even the same length of the plaintext or longer. The third parameter may go from a single character to any length, even the same length of the plaintext or longer. The processing block size may go from a single character to any length up to the length of the plaintext and is limited only by the maximum size allowed by the system where the encryption is implemented. It is also possible to process consecutive blocks of different sizes by using a sequence of values instead of a fixed one, indicating the individual size for each individual block to be processed. When the last processing size list element is exhausted, it wraps up and starts over from the beginning. When building up an application, different groups and number of substitutions may be used to create personalized non-standard versions.

VI. Building Up A Poly Substitution Cipher

A. A cipher complying with these seven steps

In order to be able to make comparisons with known standards, we decided to use a standard alphabet of 26 letters (A…Z).

We built up a cipher accordingly and complying with these seven steps and it uses five parameters:

- The plaintext to encrypt
  The plaintext is just a sequence of characters of any length.
- The key to be used.
  This key is just a sequence of characters of any length and can be longer, equal in length or shorter than the plaintext.
- A third parameter defining which substitution to use on each instance.
  This third parameter is a sequence of characters of any length and there is no required relation between its length and the lengths of the plaintext or the key.
- An initial substitution order.
  This is a sequence of values that will be used to define an initial order for the encryption substitutions to be used.
- A processing block size.
  This will define the number of characters to be read at once from the plaintext and processed in reverse order (from the last character to the first one) to generate the ciphertext. A value of 1 (one) will make the plaintext to be processed straight from the first character to the last one.
  This can be a fixed value to process same size blocks (all but maybe the last one) or a sequence of values to indicate the size of each individual block to be processed.

Depending on how the cipher is programmed and implemented, it can allow the user to manually type every parameter or to select or chose them.

The encryption process will work as follows:

1. The user may select or enter the plaintext to process, the key, the third parameter, the initial substitution order and the processing block size or sizes.
2. The process loads the initial substitution order into an array with the same number of elements as substitutions to be used.
3. If the remaining of the plaintext is shorter than the processing block, the processing block size is adjusted accordingly.
4. The process reads a processing block from the plaintext. If the plaintext has been exhausted, the process ends.
5. The process takes the last character from the processing block.
6. The process takes a character from the key.
   If the key has been exhausted, reorder the initial substitution order array elements and read the first key byte again.
7. The process takes a character from the third parameter.
   If the third parameter has been exhausted, start over from its first character.
8. The process uses the character from the third parameter to point to an element from the substitution order array and uses its value to trigger an encryption substitution passing the plaintext and key characters as parameters.
9. The substitution triggered returns a ciphertext character that is written to the ciphertext output.
10. The process takes the previous character from the processing block.
    If the processing block has been exhausted, jump to step 3.
11. Jump to step 5.

The decryption process will work the exact same way, using the ciphertext instead of the plaintext and reversing the encryption process, using the same key, and same remaining parameters, using the reverse substitution on each instance.

B. Typical Distribution of Letters in English Language

There is not an unique and generic distribution of letters in the English language and the use of field-related jargons may impact the results. (i.e., a statistical analysis of medical books may provide a different result from financial or sports books). Despite those small differences, the results are mostly similar on which are the most common letters. Figure 5 shows the relative frequencies of letters in the English language based

on "Letter Frequencies in English", published by The Oxford Math Center [29] from Oxford College of Emory University.

Letter	Frequency (in %)
A	8.167%
B	1.492%
C	2.782%
D	4.253%
E	12.702%
F	2.228%
G	2.015%
H	6.094%
I	6.966%
J	0.153%
K	0.772%
L	4.025%
M	2.406%
N	6.749%
O	7.507%
P	1.929%
Q	0.095%
R	5.987%
S	6.327%
T	9.056%
U	2.758%
V	0.978%
W	2.360%
X	0.150%
Y	1.974%
Z	0.074%

Figure 5. Relative Frequencies of Letters in English.

These letter frequency values can also be graphically represented as it is shown in Figure 6, which is very similar to the one published by Wikipedia [30].

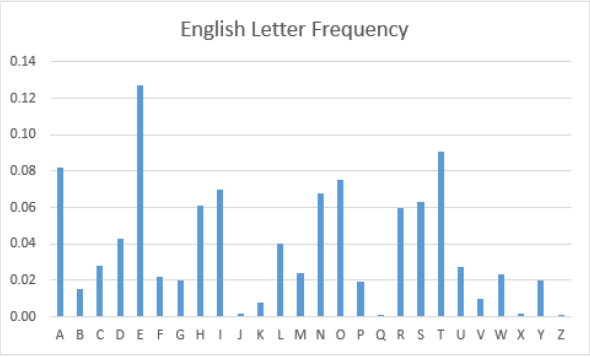


Figure 6. English Letters Frequency Graph.

C. Our Test

We took a text file from Project Gutemberg containing the Complete Works from Winston Churchill (the American Winston Churchill, not the British one) [31]. A simple text file 9,540,229 characters long from which we removed all non-alphabetic characters and obtained a 7,221,951 character long alphabetic only text.

We selected the 13 character long non-random key "WHENIWASYOUNG" and the 15 character long non-random value "WEONLYJUSTBEGUN" as the third parameter and used a fixed block size of 1 byte to keep the character sequence between the plaintext and the ciphertext.

From all 26 that form the English alphabet, the key uses only 11 different letters (A, E, G, H, I, N, O, S, U, W and Y) while the third parameters uses only 12 different letters (B, E, G, J, L, N, O, S, T, U, W, and Y). Both parameters share 8 letters in common (E, G, N, O, S, U, W and Y).

The less common multiple of the lengths of the key and the third parameter is 195. The same sequence pair of third parameter character and key character repeats over 37,035 times to match the plaintext file length.

We performed a frequency distribution analysis of the source file and the results are listed in Figure 7.

Letter	Frequency (in %)	Frequency
A	8.0788%	583,448
B	1.4146%	102,161
C	2.4972%	180,350
D	4.5871%	331,279
E	12.6098%	910,672
F	2.1128%	152,585
G	2.0429%	147,538
H	6.6493%	480,210
I	6.9210%	499,830
J	0.1486%	10,732
K	0.8086%	58,399
L	3.8873%	280,738
M	2.6948%	194,620
N	6.9766%	503,850
O	7.5462%	544,985
P	1.6051%	115,918
Q	0.0815%	5,884
R	5.8154%	419,982
S	6.1690%	445,519
T	9.0004%	650,006
U	2.7707%	200,100
V	0.9488%	68,521
W	2.4342%	175,795
X	0.1404%	10,140
Y	1.9878%	143,558
Z	0.0710%	5,131

Figure 7. Plaintext File English Letter Frequency in numbers.

The result is not an exact match but the differences are minimal and the order of the four most common letters (E, T, A and O), is the same. Based on those results, we generated a graphical representation, which is presented in Figure 8 and matches the standard graphical distribution from Figure 6.

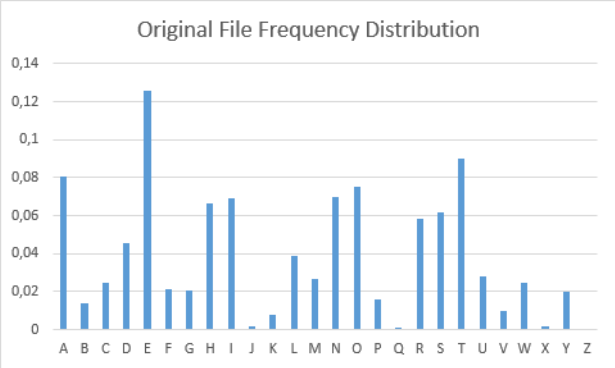


Figure 8. Plaintext File Statistical Letter Frequency Graph.

There are 7,221,950 bigrams formed by two consecutive letters in the plaintext and 7,221,949 trigrams formed by three consecutive letters. Figure 9 shows the results for the 30 most common bigrams found in the source file (those that are repeated more times along the plaintext).

Order	Bigram	Frequency in %	Frequency
1	TH	3.0676%	221,539
2	HE	3.0159%	217,807
3	IN	1.7960%	129,709
4	ER	1.7902%	129,286
5	AN	1.6384%	118,326
6	RE	1.3856%	100,070
7	ED	1.3314%	96,154
8	HA	1.2584%	90,882
9	ES	1.2079%	87,233
10	TO	1.2042%	86,970
11	ND	1.1985%	86,552
12	EN	1.1590%	83,704
13	OU	1.1378%	82,169
14	NT	1.1341%	81,903
15	ON	1.1195%	80,850
16	AT	1.0931%	78,943
17	ST	1.0649%	76,909
18	EA	0.9474%	68,424
19	HI	0.9456%	68,293
20	IT	0.9145%	66,047
21	AS	0.8947%	64,612
22	ET	0.8520%	61,530
23	OR	0.8452%	61,039
24	NG	0.8423%	60,833
25	IS	0.8241%	59,517
26	TE	0.7945%	57,378
27	AR	0.7714%	55,710
28	TI	0.7699%	55,600
29	OF	0.7433%	53,678
30	SE	0.7316%	52,837

Figure 9. Source File Bigram Frequency.

On the other extreme of the listing, from all 676 possible bigrams, there are 97 that are repeated 10 times or less within the plaintext. From them, 49 of them are not present at all, 11 are present only once, 9 appear twice, 4 appear three times, 4 appear four times and 4 appear five times. The results from the plaintext will be compared with the results from the ciphertext.

Figure 10 shows the results for the 30 most common trigrams found in the source file.

Order	Trigram	Frequency in %	Frequency
1	THE	1.8690%	134,980
2	AND	0.8596%	62,080
3	ING	0.6603%	47,687
4	HER	0.6143%	44,362
5	THA	0.4724%	34,118
6	ERE	0.4350%	31,419
7	HAT	0.4128%	29,811
8	YOU	0.3814%	27,542
9	NTH	0.3670%	26,504
10	ENT	0.3650%	26,363
11	WAS	0.3440%	24,845
12	SHE	0.3425%	24,733
13	HIS	0.3362%	24,278
14	ETH	0.3251%	23,478
15	HES	0.3168%	22,876
16	DTH	0.3046%	21,999
17	THI	0.2988%	21,577
18	INT	0.2934%	21,190
19	FOR	0.2912%	21,030
20	ITH	0.2707%	19,548
21	HAD	0.2693%	19,450
22	TTH	0.2476%	17,880
23	TER	0.2392%	17,272
24	ION	0.2372%	17,128
25	OFT	0.2360%	17,047
26	FTH	0.2351%	16,981
27	EST	0.2327%	16,807
28	OTH	0.2315%	16,716
29	EDT	0.2286%	16,507
30	WIT	0.2232%	16,119

Figure 10. Source File Trigram Frequency.

On the other extreme of the listing, from all 17,576 possible trigrams, 8642 are not present at all, meaning that almost half of all possible trigrams (49.17%) are not present in the plaintext.

After encrypting the file, we performed the same frequency distribution analysis on the encrypted file as we did on the original source text file. Figure 11 shows the encrypted file letter frequencies graphical distribution (showing frequency distribution as a percentage of the total number of characters).

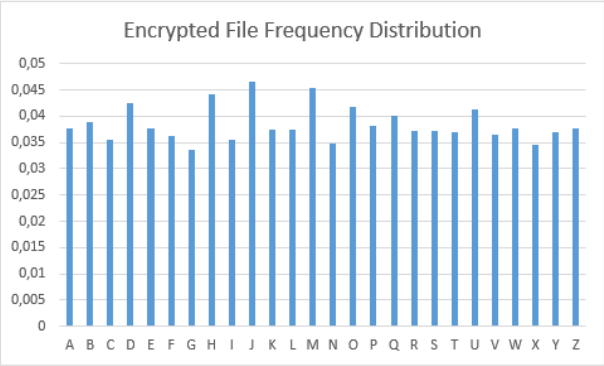


Figure 11. Ciphertext File Statistical Letter Frequency Graph.

The letter frequency of the encrypted file is homogeneous and almost flat making impossible a statistical distribution analysis attack based on the statistical distribution of the letters in the English language. There is no way to match the letters from the encrypted file with those from the source file. Figure 12 shows the letter frequency from the ciphertext.

Letter	Frequency (in %)	Frequency
A	3.7686%	272,167
B	3.8798%	280,198
C	3.5510%	256,452
D	4.2553%	307,314
E	3.7649%	271,896
F	3.6380%	262,733
G	3.3720%	243,523
H	4.4154%	318,879
I	3.5593%	257,049
J	4.6710%	337,340
K	3.7397%	270,079
L	3.7402%	270,112
M	4.5380%	327,730
N	3.4808%	251,384
O	4.1863%	302,336
P	3.8112%	275,244
Q	4.0089%	289,520
R	3.7138%	268,209
S	3.7167%	268,416
T	3.7002%	267,229
U	4.1338%	298,538
V	3.6556%	264,003
W	3.7597%	271,523
X	3.4586%	249,776
Y	3.7073%	267,740
Z	3.7741%	272,561

Figure 12. Ciphertext File Letter Frequency.

When we order the letters from the plaintext and the ciphertext sorting them down in decreasing order of the statistical distribution of the letters within them, we get the graphical representation displayed in Figure 13.

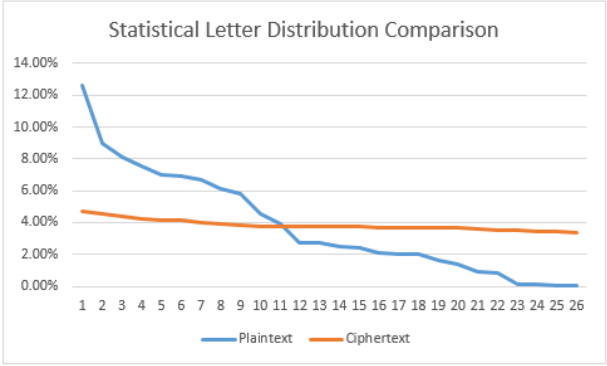


Figure 13. Statistical Letter Distribution Comparison.

We analyzed the frequency distribution of bigrams and trigrams within our encrypted text file and Figure 14 shows the results for the 30 most common bigrams in the ciphertext.

Order	Bigram	Frequency in %	Frequency
1	JJ	0.2181%	15,750
2	JM	0.2118%	15,299
3	MJ	0.2096%	15,136
4	JH	0.2055%	14,843
5	HJ	0.2040%	14,734
6	MM	0.2022%	14,603
7	MH	0.2021%	14,596
8	JD	0.2007%	14,492
9	JO	0.1997%	14,422
10	HM	0.1995%	14,411
11	MD	0.1968%	14,211
12	DJ	0.1962%	14,168
13	OJ	0.1957%	14,130
14	HH	0.1955%	14,116
15	JU	0.1944%	14,036
16	UJ	0.1940%	14,008
17	DM	0.1938%	13,999
18	MU	0.1904%	13,753
19	QJ	0.1897%	13,701
20	JQ	0.1888%	13,632
21	MO	0.1883%	13,602
22	OM	0.1878%	13,565
23	UM	0.1868%	13,494
24	DH	0.1864%	13,461
25	OH	0.1857%	13,412
26	HU	0.1853%	13,384
27	HO	0.1851%	13,369
28	QM	0.1843%	13,310
29	HD	0.1836%	13,257
30	MQ	0.1826%	13,185

Figure 14. Encrypted File Bigram Frequency.

The first finding analyzing bigrams was that all possible 676 bigrams are present in the ciphertext, the most repeated one appears 15,750 times and the least repeated one appears

8,223 times in it. The bigram distribution for the encrypted file is almost flat and homogeneous and there is no way to match the bigrams from the encrypted file with those from the source file as Figure 15 shows.

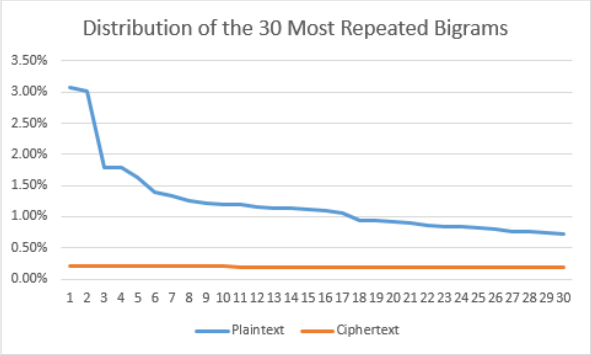


Figure 15. Bigram Statistical Frequency Comparison.

Figure 16 shows the results for the 30 most common trigrams found in the encrypted file.

Order	Trigram	Frequency in %	Frequency
1	MMJ	0.0106%	763
2	HJJ	0.0105%	759
3	MJJ	0.0105%	755
4	JJM	0.0102%	739
5	JMJ	0.0101%	733
6	MJH	0.0101%	727
7	JJH	0.0101%	726
8	MDM	0.0100%	722
9	JJJ	0.0099%	718
10	JJD	0.0098%	710
11	MHJ	0.0098%	710
12	HJM	0.0098%	709
13	JHM	0.0098%	707
14	MMM	0.0097%	704
15	JMM	0.0097%	701
16	JMD	0.0097%	698
17	JDM	0.0097%	697
18	MHH	0.0096%	695
19	JHH	0.0096%	694
20	JJU	0.0096%	691
21	MJO	0.0096%	691
22	JEJ	0.0095%	688
23	HMJ	0.0095%	687
24	UJO	0.0095%	687
25	JOJ	0.0095%	684
26	JOH	0.0095%	683
27	DJJ	0.0094%	681
28	MPJ	0.0094%	681
29	YJJ	0.0093%	675
30	HHJ	0.0093%	673

Figure 16. Encrypted File Trigram Frequency.

The first finding analyzing trigrams was that all possible 17,576 trigrams are present in the ciphertext, the most repeated once appears 763 times and 260 times the least repeated one. The most repeated bigram (MMJ) repeats less than three times the least repeated one (GGV) and the flatness of the distribution makes unfeasible any statistical analysis attack to retrieve the original plaintext from the ciphertext based on the distribution of trigrams.

Comparing the statistical distribution of the 30 most common trigrams from the plaintext and the ciphertext, Figure 17 shows the differences that make unfeasible a statistical analysis attack based on the distribution of trigrams.

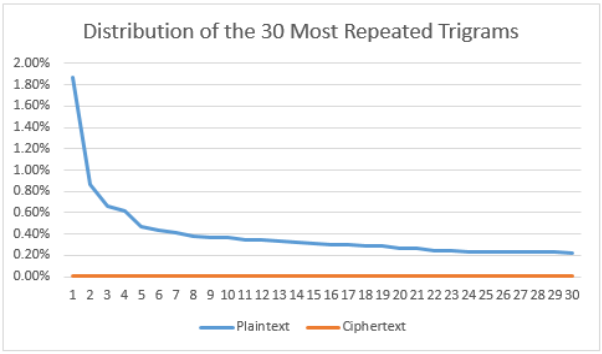


Figure 17. Trigram Statistical Frequency Comparison.

Going one step further and considering tetragrams (sets of four consecutive letters), there are 456,976 possible tetragrams and from them 378,431 are not present in the plaintext. Only 17.188% from all possible tetragrams appear in the plaintext being THAT with a 0.3171% (22,898 repetitions) and THER with 0.3089% (22,305 repetitions) the two most common of them. From all 456,976 possible tetragrams, only two are not present within the ciphertext (HXYA and NGXC), being HJMJ with a 0.00071% (51 repetitions) and HHJJ with 0.00068% (49 repetitions) the two most common of them.

Figure 18 compares the statistical distribution of the 26 most common tetragrams from the plaintext and the ciphertext.

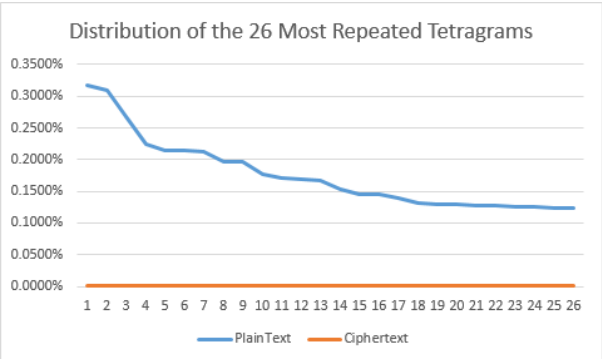


Figure 18. Tetragram Statistical Frequency Comparison.

D. Results of Our Test

Although the key used was very short (only 13 characters long) and not random, the third parameter was also very short (only 15 characters long) and also not random, and the source file was pretty big (over 7 Megabytes long), the use of poly substitution encryption provided a level of confusion and diffusion that makes any cryptanalysis or statistical distribution analysis attack totally unfeasible, as will be proved in Section VIII.

When the gross frequency (the number of times each item is repeated as a number, not as a percentage) of letters, bigrams, trigrams and tetragrams are compared between the plaintext and the ciphertext, the flatness of the results gets visually clear.

In the distribution of letters within the plaintext (alphabetically ordered from A to Z), as shown in Figure 19, the most common letters E, T, A and O are clearly distinguishable.

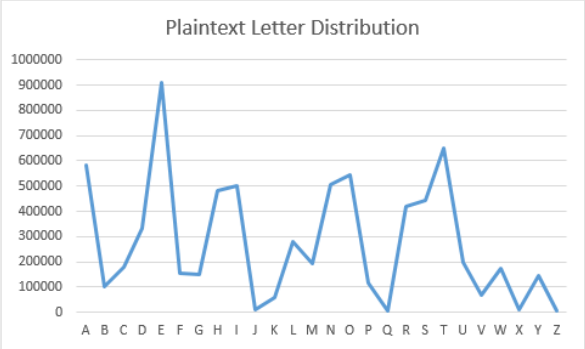


Figure 19. Plaintext Letter Distribution.

A ciphertext obtained from a Vigenère encryption of the plaintext using the same key, would have produced the exact same graph. The graphic of the letter distribution within the ciphertext displayed in Figure 20, shows not only the flatness of the distribution obtained through the use of a cipher based on our proposed encryption, but the small variation among the letters.

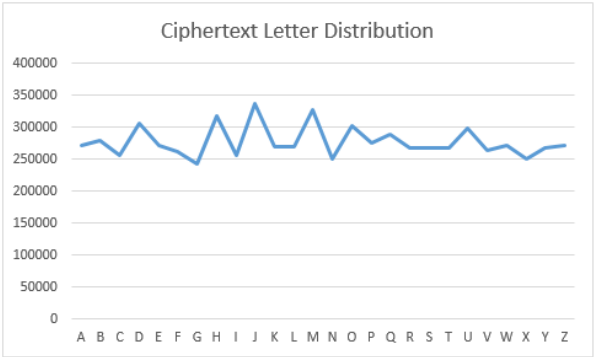


Figure 20. Ciphertext Letter Distribution.

In the distribution of bigrams within the plaintext (ordered from AA to ZZ), as shown in Figure 21, the most common bigrams TH and HE are clearly distinguishable.

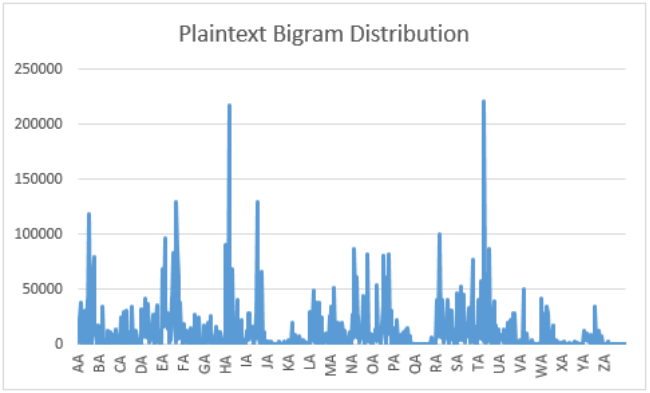


Figure 21. Plaintext Bigram Distribution.

The distribution of bigrams within the ciphertext also shows the flatness of the result and the low variation between the different bigrams, as shown in Figure 22.

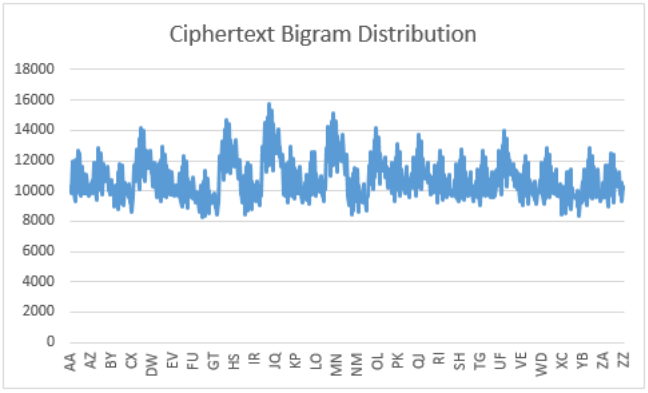


Figure 22. Ciphertext Bigram Distribution.

Although some bigrams are not present in the plaintext, all possible bigrams are present in the ciphertext.

In the distribution of trigrams within the plaintext (from AAA to ZZZ), as it is shown in Figure 23, the most common values THE, AND, ING and HER, are clearly distinguishable.

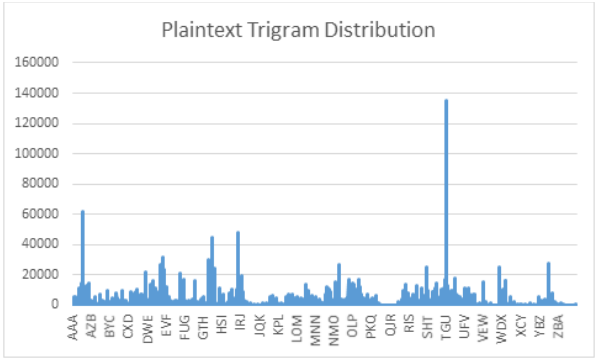


Figure 23. Plaintext Trigram Distribution.

But the distribution of trigrams within the ciphertext (from AAA to ZZZ), as it is shown in Figure 24, tells a totally different story. Once again, the graphic of the trigram

distribution within the ciphertext displayed in Figure 24, shows not only the flatness of the distribution obtained through the use of a cipher based on our proposed encryption, but the small variation among the different trigrams.

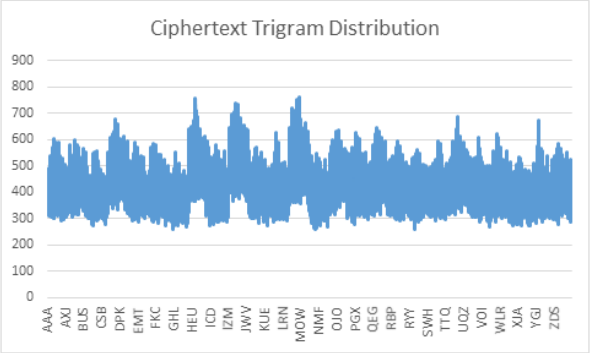


Figure 24. Ciphertext Trigram Distribution.

The graphical representation of the ciphertext trigram distribution resembles white noise. White noise is basically pure random noise that has all the frequencies in the audio spectrum. It is a random signal having equal intensity at different frequencies, giving it a constant power spectral density. Different sounds and musical notes are produced by a combination of different frequencies, the same way as words and phrases are formed by a combination of different letters from an alphabet. Some random number generators are based on white noise like the one used by Random.org, whom uses a system of atmospheric antennae to generate random digit patterns from white noise. White noise sounds pretty close to static from your old television set or radio when no station was tuned or a whooshing sound. Additive white Gaussian noise (AWGN) is a basic noise model used in information theory (originally proposed by Claude E. Shannon in 1948 in his landmark paper [7]) to mimic the effect of many random processes that occur in nature.

This shows how the distribution of trigrams within the ciphertext cannot be distinguished from being totally random. Considering that the same occurs with the distribution of bigrams and single letters within the ciphertext, that proves that any statistical analysis attack on the ciphertext will not succeed as we will prove in Section VIII.

VII. ANALYSIS

A. *Comparing this cipher with Vigenère's and Vernam's*

It is trivial to prove that the Vigenère cipher is a limited or restricted version of our proposed encryption where the processing block size is one character and only one substitution is used (modular addition with v = 0), which means the third and fourth parameters need to be built in a way that ensure the same substitution will always be triggered.

It has already been proved that the Vernam cipher is a restricted or limited version of the Vigenère cipher, where the alphabet used has only two characters or symbols).

A text message properly ciphered through the Vigenère or Vernam Ciphers (using a random one-time key as long as the plaintext) gives absolutely no clue on the key used or the original plaintext and a brute force attack will end up with a huge number of false positives.

A brute force attack will return some invalid or unreadable results but will also return any possible message with the exact same length and there is no way to decide which one is the true original one.

Vigenère and Vernam ciphers are not used because they have the same three requirements that need to be fulfilled to comply with Shannon's definition for Perfect Secrecy:

1) The key needs to have the same length as the plaintext.
2) The key must be random.
3) The key must not be reused.

These three requirements are mandatory because Vigenère and Vernam used a single encryption substitution (Modular Addition and XOR) in the process.

With the Vernam cipher, for any given ciphertext byte, the attacker needs to try every possible key byte value and will end up with 256 different results, each one with the exact same probability of being the plaintext byte value.

With the Vigenère cipher, for any given ciphertext symbol, the attacker needs to try every possible key symbol and will end up with every possible symbol in the alphabet, each one with the exact same probability of being the plaintext symbol.

With our proposed encryption technique and even assuming the attacker knows the exact processing block size used for this specific ciphe text, all the encryption substitutions used and can match each ciphertext byte with the corresponding byte position in the plaintext; the attacker will still need to try each of the 256 possible key byte values with each of the encryption substitutions involved. So, if we used 256 different encryption substitutions, the attacker will end up with 65,536 possible values for the plaintext byte, each one repeated many times and no way to decide which value is the original one.

If the attacker does not know the processing block size, it multiplies the effort required as the first byte from the plaintext may correspond to any of the bytes in the ciphertext, the second one to any of the bytes except the last and so on, doing the math it means there are n! possible orders for the ciphertext to match the byte order of the plaintext, being n the length in bytes of both the plaintext and the ciphertext.

Our proposed cipher does not have any constraints as Vigenère and Vernam ones do, and we will prove that our encryption overcomes those constraints that come from mono-substitution encryption limitations.

As we can assure the same key value-encryption substitution sequence will not be repeated, the length of the key becomes irrelevant, it may have any length and it does not matter if it is shorter than the plaintext.

So far, we have been able to overcome the first of Vigenère's and Vernam's constraints and now the key can be shorter than the plaintext without impacting the safety of the process.

As we use some additional parameters, does the key truly need to be random?

Leaving aside any discussion about what is truly random and what is not, anything can be used as a key; a text, a web page, a file from the Internet. As far as the key is kept secret, it really does not matter whether it is truly random or not.

In the example we provided the key used was extremely short compared to the length of the plaintext (the plaintext length was about half a million times the length of the key) and it was not random at all and despite that, no statistical analysis attack will succeed in retrieving the original plaintext having only the ciphertext.

As the process does not use an unique encryption substitution but a pool of them, the randomness of the key has no impact on the outcome because the substitution sequence order cannot be predicted.

Our test showed how using a short non-random key had no impact on the security of the process.

So far, we have been able to overcome the second of Vigenère's and Vernam's constraints and now the key does not need to be random.

What if the key is reused?

As we consider the encryption process as the result from applying a sequence of key value-encryption substitution pairs, what we need to avoid is reusing that exact same sequence.

It is clear that if any of the third parameter, the initial substitution order or the block size (or sizes) is different, the key value-encryption substitution pair sequence will be different for the same key. If the same substitution order array rearrangement used when the key is exhausted is used every time a processing block is exhausted, it can be guaranteed that the same key value-encryption substitution pair sequence will not be repeated for a different plaintext even if the exact same parameters are used and if the first processing block is just one byte long, then no other key value-encryption substitution pair sequence will be repeated beyond the first symbol even if the same exact parameters are reused. This happens because the probability that two different plaintexts or ciphertexts may have matching characters in the exact same position every time the key, the processing block or whatever trigger may be used is exhausted, is just zero.

This overcomes the third of Vigenère's and Vernam's constraints and now the key may be reused without compromising the unconditional security.

As you see, an encryption algorithm based on our proposed technique and complying with its seven steps can guarantee unconditional security using non-random keys that can be shorter than the plaintext and can be reused.

Also, two successive encryptions may exchange the key with the third parameter using the third parameter as the new key and the old key as a new third parameter keeping the unconditional security.

As far as the key does not need to be random, selecting a new different key is quite easy. There is no need of a random or pseudo-random key generator as any possible file may be used as a key (or third parameter). A web page from any site, a file of any type or even portions of them can be used.

Been Vernam the only cipher mathematically proved to be unbreakable if properly used, let us do a comparison between a cipher based on our proposed poly-substitution encryption and the Vernam cipher. Figure 25 shows a comparison between Vernam's cipher and our proposed one.

	VERNAM	Cipher based on our Proposed Technique
Sample plaintext length	140	140
Processing block size	1	Variable
Key size	140	Variable
Key and plaintext length must match	Yes	No
Key must be true random	Yes	No
Key must not be reused	Yes	No
No. of substitutions	1	256
No. of possible results per substitution	256	256
No. of possible results per Byte	256	65,536
Ciphertext to plaintext match	1	140!
No. of possible results per Byte from brute force attack	256	65,536 x 140!
No. of possible results per Byte from brute force attack as power of 2	2^(8)	2^(809)
Probability of being the plaintext byte	0,39%	0,39%
Rounds	1	1
False Positives	Yes	Yes

Figure 25. Comparing Vernam's cipher to our proposal.

An encryption algorithm based on our technique and complying with the seven steps will offer the same unconditional security offered by the Vernam cipher without the constraints it has.

A plaintext encrypted with such algorithm will remain impervious to any attack, no matter how powerful the attacker may be or may become, even if the attacker has infinite computational power.

B. Comparing this cipher with currently used ciphers

Due to their extreme complexity, none of the current encryption standards will produce a false positive when an incorrect or wrong key is used.

All currently used encryption base their privacy and security on the unavailability of enough computational power required to try all possible keys in a short time and that is why they will all fall under a quantum attack capable of trying every possible key in very little time.

There is an old saying: "How do you hide an elephant on a beach? By filling the beach with elephants".

The strength of our proposed encryption technique relies not on the computational power required to try every possible key, third parameter, initial substitution order, substitution set or block size or sizes; its strength relies on the fact that we assume it can be done but the real original plaintext will be hidden at plain sight within an immense sea of false positives

with absolutely no indication on which one is the right one. It is not that the attacker will not be able to get the original plaintext, it is that the attacker will not be able to distinguish the original plaintext from the false positives.

We assume the attacker will reach the beach and see all that is there. There can also be palms and monkeys, and seagulls and turtles, and crabs and people and of course sand and water on that beach, but even if the attacker knows that what is hidden there is only an elephant, still will have no clue and no way to know which one of all the elephants there is the right one. Even if the attacker is capable of knowing that what is hidden there is an elephant and ignores all the palms and monkeys and seagulls and turtles and crabs and people and any other element that is not what is hidden there, even then, there is absolutely no way and no clue to know which elephant is the right one.

Symmetric key cryptography relies on very complex processes where a brute force attack may need too much computational power to succeed and no such power is yet available. They do not say their ciphers cannot be broken with enough computational power, they claim and hope for the required computational power not to be available anytime soon.

Besides the existence of some known partial attacks that may succeed with more computational power like quantum computers promise to deliver, the use of a fixed size processing block and fixed size keys (they may offer different key sizes but limited to very few pre-defined options. They do not allow the user to freely choose any key size, less to select the block size), make them vulnerable to brute force attacks given the attacker has enough computational power.

While currently in use symmetric encryption standards use block sizes of 64 or 128 bits and keys of 128 to 512 bits, our encryption handles blocks of any size including successive blocks of different sizes and keys of any length. Figure 26 shows a comparison between a cipher based on our proposed technique and other symmetric ciphers.

	Block Size	Key Size	Rounds	False Positives
DES	64 bit	56 bit	16	No
3DES [33]	64 bit	128 bit	48	No
AES	128 bit	128, 192, 256 bit	9, 11, 13	No
BLOWFISH [34]	64 bit	32-448 bit	16	No
Cipher based on our Proposed Technique	Variable	Variable	1	Yes

Figure 26. Comparison between a cipher based on this technique and current symmetric standards.

Public-key cryptography currently in use (including RSA and ECC) relies on the assumption that some problems cannot be solved or would will require an extremely long time to be solved, and therefore, that it would take a very long time for their secured data to be decrypted. But as quantum algorithms can solve some of these problems with ease, that assumption is fatally challenged. It is known that a quantum computer running Shor's algorithms can easily solve complex problems like long integer factorization and discrete logarithms, which are the foundation of public key cryptography.

VIII. Attacking a Cipher Based on This Technique

Trying to retrieve the plaintext from a ciphertext created through an implementation of this technique without having any additional information will be at least as difficult as trying to retrieve the plaintext from a one-time pad created ciphertext or one created through a proper use of Vigenère's or Vernam's ciphers having only the ciphertext.

Any attack must take into consideration that all the parameters are external to the process and they all may be different from one encryption to the next and also the fact that the process may be used in reverse order. Decryption can be used to protect the plaintext and encryption with the same parameters used to retrieve the original plaintext.

As cryptanalysis of our encryption is just not possible, any possible attacker will need to face the following difficulties when attempting a brute force attack to break an encryption created with a cipher based on this technique and complying with its seven steps:

- Which ciphertext byte corresponds to each plaintext byte.
- Which encryption substitutions exist and which of them were used.
- Which substitution was used on each instance.
- Which was the key used.
- Which processing block size or sizes were used.

Let us give the attacker the advantage of knowing all the encryption substitutions involved, the specific set used to create the ciphertext and the processing block size or sizes used. In such situation, for each byte in the ciphertext, the attacker needs to try every possible substitution for every possible key byte value and so, instead of getting 256 possible values as with Vigenère's or Vernam's ciphers, the result will be 65,536 possible values having every single one the exact same probability of being the plaintext byte value despite the repetitions.

That is the best case scenario for the attacker.

If the processing block size or the sequence of block sizes is not known, the attacker will need to try any possible fixed or variable block size from a single byte to the length of the ciphertext. While this adds time and difficulty to the attack, every possible outcome still has the exact same probability of being the original plaintext despite the repetitions.

The original plaintext will still be lost in a sea of false positives with no way to decide which one is the right one. The beach will remain full of elephants with no clue on which one is the right one.

As Ronald Linn Rivest, Adi Shamir and Leonard Adleman stated on their seminal paper [35], "Since no techniques exist to prove that an encryption scheme is secure, the only test available is to see whether anyone can think of a way to break it." Based on that, we will give a try to classical and modern cryptanalysis attacks and try to show how and why they will not succeed in breaking the encryption.

A. Statistical Letter Distribution Attack

We have seen that single character distribution within the ciphertext is practically flat, making impossible to identify the original corresponding characters from the plaintext through a statistical distribution attack.

B. Kasiski Statistical Analysis Attack

A Kasiski Statistical Analysis attack uses repeated sequences from the ciphertext to try to narrow down the length of the key.

First we analyzed the 10 most common bigrams (JJ, JM, MJ, JH, HJ, MM, JD, JO and HM) and measured the distances between consecutive occurrences of the same bigram across the ciphertext and found out that for all of them, the minimum distance between two occurrences of the same bigram is zero, meaning that each bigram shows two times consecutively and together, and this happens many times for each of them. Figure 27 shows the ten most common bigrams and the number of times they appear twice together within the ciphertext.

Order	Bigram	Appears as	Occurrences
1	JJ	JJJJ	37
2	JM	JMJM	36
3	MJ	MJMJ	30
4	JH	JHJH	23
5	HJ	HJHJ	35
6	MM	MMMM	26
7	MH	MHMH	32
8	JD	JDJD	23
9	JO	JOJO	28
10	HM	HMHM	33

Figure 27. Trigram distances between repetitions.

Being zero the minimum distance between occurrences of the same repeated bigram and having the same happening for all 10 most common bigrams across the ciphertext, allows us to affirm that the distances between repeated occurrences of bigrams gives absolutely no information about the length of the key, making totally useless a Kasiski statistical analysis attack based on the repetition of bigrams.

Second, we took the three most repeated trigrams, MMJ, which repeats 763 times within the ciphertext, HJJ, which repeats 759 times and MJJ, which repeats 755 times.

We built up a table listing the distances from each appearance of each trigram to the next in one column and the list of distances sorted out from the smallest distance to the biggest and we did the same for each of the three trigrams.

For the first trigram (MMJ) only 58 out of the 763 repetitions are at a distance that is a multiple of the key length of 13 characters and only 5 of them were a multiple of the less common multiple of the lengths of the key and the second parameter.

For the second trigram (HJJ) only 48 out of the 759 repetitions are at a distance that is a multiple of the key length of 13 characters and only 2 of them were a multiple of the less common multiple of the lengths of the key and the second parameter.

For the third trigram (MJJ) only 49 out of the 755 repetitions are at a distance that is a multiple of the key length of 13 characters and only 7 of them were a multiple of the less common multiple of the lengths of the key and the second parameter.

It is clear that the encryption generates a diffusion of the results even using short keys, which makes any statistical analysis attack unfeasible. No matter how you compare the results of analyzing the most repeated trigrams, no information about the key length can be obtained.

Figure 28 shows the first 24 results for each of the trigrams.

	MMJ		HJJ		MJJ	
	U/O	O	U/O	O	U/O	O
1	11,339	***5***	771	4	5,020	***7***
2	6,817	***11***	***487***	15	11,054	***11***
3	8,828	12	19,573	38	***4,003***	***13***
4	1,944	15	8,517	55	***6,997***	20
5	6,598	22	1,248	58	5,062	32
6	11,538	45	854	69	4,700	44
7	9,032	46	***13,103***	80	19,637	***53***
8	2,079	66	***2,111***	82	7,679	58
9	33,580	75	2,507	91	6,878	66
10	638	106	692	***109***	1,784	66
11	1,164	121	20,869	***113***	36,239	74
12	11,664	124	5,634	133	8,294	77
13	12,180	130	1,390	***139***	6,008	99
14	1,687	156	6,226	144	6,341	152
15	13,358	171	12,450	160	7,704	***163***
16	4,925	183	287	***227***	2,888	190
17	***2,659***	187	1,323	228	6,840	216
18	2,018	192	5,049	***239***	22,798	***233***
19	1,394	***199***	26,270	***241***	26,708	242
20	26,814	209	42,915	243	2,959	266
21	***4,457***	216	11,013	246	7,174	268
22	3,385	242	***797***	260	***5,527***	***269***
23	10,621	259	***5,021***	267	10,530	270
24	***2,411***	265	5,061	267	9,970	284

Figure 28. Trigram distances between repetitions.

U/O stands for unordered, listing the distance from an occurrence of the trigram to the next one, while O stands for ordered, which is an ordered list of the distances between two occurrences of the same trigram ordered in ascending order from the shortest distance between two occurrences of the same trigram to the longest one.

This table shows that it is not possible to find any relation between the distances of the different trigram repetitions and the key length. We grayed out and styled in bold and italic those distances between two consecutive repetitions of the same trigram that are in fact a prime number, meaning either the key cannot be shorter or they are random repetitions.

It is clear that a Kasiski statistical analysis attack will not succeed even considering that the key is not random and extremely short when compared to the length of the original plaintext.

C. Friedman's Index of Coincidence Attack

The formula to calculate the Index of Coincidence (IC) for any given text is as follow:

$$Ic = \sum_{c \in alphabet} \left(\frac{count(c) \times (count(c) - 1)}{Length \times (length - 1)} \right) \qquad (20)$$

Theoretically perfect IC is defined as if all characters occurred the exact same number of times so that none was more likely than any other to be repeated, so, for an alphabet of 26 characters, it was calculated to be 1/26, which is approximately 0.03846. The closest the IC of a given ciphertext is to the perfect IC, the more difficult it will be to try to obtain the key length or the original plaintext.

The calculated IC of the original plaintext is 0.0659 and the calculated IC for the ciphertext produced by our encryption is 0.03874, a difference of 0.00028 from the perfect IC.

Poly-substitution encryption generates a level of diffusion that makes obtaining the key length through the IC an impossible task.

D. Linear Cryptanalysis.

The discovery of linear cryptanalysis is credited to Mitsuru Matsui [36], who first applied the technique to the FEAL cipher in 1992. Later, he published an attack on the Data Encryption Standard (DES) [37].

Linear cryptanalysis focuses on finding a linear relationship between a subset of plaintext bits and a subset of data bits that behaves in a non-random fashion. It is a known-plaintext attack, meaning the attacker will have some sets of plaintexts and associated ciphertexts, all encrypted with the same key. It was first intended as a cryptanalysis attack to DES, but proved to be useful for other multi-round fixed-block ciphers. It requires to have some pairs of known-plaintext/ciphertext pairs encrypted with the same key. The first difficulty linear cryptanalysis will find is that the key does not have a fixed length and also the processing block size or sizes are also of not fixed length. The second difficulty is that there are no s-boxes and the encryption is made in a single round. Even if the attacker gets a huge number of known-plaintext/ciphertext pairs all encrypted using the exact same parameters, the length of the key and the length or lengths of the processing block remain unknown and those pairs of known-plaintext/ciphertext give no clue about them, making unfeasible the use of linear cryptanalysis to attack poly substitution encryption as it is defined in this paper.

E. Differential Cryptanalysis.

Differential cryptanalysis discovery is usually credited to Adi Shamir and Eli Biham, who published a number of attacks against several block ciphers in the late 1980s. Don Coppersmith, a member of the original IBM DES team, published a paper stating that differential cryptanalysis was known to IBM as early as 1974 [38].

Linear cryptanalysis is based on exploiting linear relationships between bits in the cipher, while differential cryptanalysis uses differential relationships between various bits in the cipher. Differential cryptanalysis is a chosen-plaintext attack where the attacker is able to make a cryptosystem encrypt data he chooses using the target key which is unknown and remains secret. Analyzing the ciphertext obtained (which is known), the attacker can obtain the key used.

The standard differential cryptanalysis method is a probabilistic chosen-plaintext attack. It is also oriented to multi-round ciphers like AES with a fixed length key and a fixed length processing block. There is no way the cryptosystem may encrypt without providing all the parameters, including the key and as we have seen in the linear cryptanalysis attack, even having pairs of known-plaintext/ciphertext gives no information about the lengths of the key and the processing block or blocks used, making unfeasible the use of differential linear cryptanalysis to attack poly substitution encryption as it is defined in this paper.

F. Side-channel attacks..

A side-channel attack is an attack based on knowledge gained from the implementation of a computer system instead of weaknesses from the encryption algorithm itself. As the encryption we propose is based in simple arithmetic operations like addition and subtraction of byte values, timing and power-analysis attacks will fail as well as other side-channel attacks like Power-monitoring attacks, electromagnetic attacks or differential fault analysis just because none of them may distinguish an addition from a subtraction or two different additions being performed.

G. Other Modern Cryptanalysis

Time-Space Trade-Offs like Diffie-Hellman's meet-in-the-middle attack, Hellman's Time-Space Trade-off, or Rivest's Distinguished Endpoints just will not work, and we will explain why.

Diffie-Hellman's meet-in-the-middle attack is oriented to break multiple-encryption algorithms repeating the same encryption using different keys as in Double-DES and Triple-DES and it is not the case with our single round poly substitution encryption.

Hellman's Time-Space Trade-off is based on pre-computing sample plaintext/ciphertext pairs using random keys and considering the mapping of key k to ciphertext c as a random permutation function f over an N point space, being N the total number of possible keys and also assumed to be the total number of possible plaintexts and ciphertexts. The first difficulty such approach will face is that there is not an unique random permutation function f. For the same plaintext, it will exist more than one key-substitution pair sequence that will produce the exact same ciphertext. If we consider that we are using an alphabet A with a characters or symbols and we are using a third parameter with the same alphabet and using a number a of different substitutions, for any given plaintext symbol-ciphertext symbol pair and for each different

substitution, there will be a key value that will produce the ciphertext symbol from the plaintext one using that substitution and key values. The crucial issue Hellman's Time-Space Trade-off will find with poly-substitution encryption is that if the number of possible plaintexts and ciphertexts is N, and an alphabet with a symbols is used, the number of possible keys will be in the order of a^N, which for $a \geq 2$ is a lot bigger than N, making unfeasible to use this attack.

Rivest's Distinguished Endpoints is an enhancement of Hellman's Time-Space Trade-off but will face the same difficulties with poly-substitution encryption and will not work for the same reasons.

IX. Partial or Limited Data Universes

In certain situations, some byte values or sequences may be considered restricted, invalid or not acceptable. As an example, when transmitting data, the end-of-message value cannot be part of the transmitted message and an encrypted message cannot include the end-of-message within it. Also, some structured messages require the information stored within the message to comply with some structural lengths and data types for specific parts of the message and to encrypt such messages, the encryption must respect data structures and formats, something none of the available encryption solutions, either symmetric or public-key can provide.

A. Format Preserving Encryption

Format Preserving Encryption (FPE) refers to encrypting in such a way that the output ciphertext is in the same format as the input plaintext, including having the exact same length. If the plaintext is just numbers, you get numbers, if it is alphabetic characters you get alphabetic characters, etc.

For example: To encrypt a sixteen-digit credit card number so that the ciphertext is another sixteen-digit number, or, to encrypt a nine-digit social security number so that the ciphertext is another nine-digit number, or to encrypt a person's name so the ciphertext is another alphabetic string with the same length.

One reason to use FPE comes from the difficulty to integrate encryption into existing applications with well-defined data models like banking, industry, financial technologies or medical records databases among others.

Adding encryption to such applications comes with high costs associated to the field length limits, data type changes and computational power required.

Recent scandals like de Panama papers, the NSA secrets offered in auction or the World Anti-Doping Agency (WADA) medical records disclosure are clear examples of the need for format preserving encryption to protect sensitive information within databases. In the Panama papers and the WADA cases, if the fields containing sensitive information would have been encrypted, the information stolen would have been useless for the hackers despite the security breach they achieved, precisely because the names of the persons involved and other personally identifiable information (PII) would have been encrypted.

B. Symmetric Key Format Preserving Encryption

Block ciphers cannot preserve the plaintext length without additional work unless it exactly matches the block size used or an exact multiple of it.

If you are trying to encrypt a nine-digit social security number stored in a nine byte plaintext, by default a block cipher like AES will return a 16 byte (128 bits) ciphertext that cannot be guaranteed to be numeric.

John Black and Philip Rogaway [39] described three ways to implement Format Preserving Encryption they proved as secure as the block cipher used to construct each of them:

- FPE from a prefix cipher.
  Assign a pseudorandom weight to each integer in the range {0,…,N-1} and then sort by weight. The weights are defined by applying a block cipher to each integer and then sorting by the result ciphertext value. A different key will result in a different weight order. The size and number of entries required for the lookup table and the number of encryptions that need to be performed to initialize the table make this technique impractical for large values of N.
- FPE from cycle walking.
  If there is a limited set of valid values within the block cipher permutation domain, a Format Preserving Encryption algorithm can be created by repeatedly applying the block cipher until the result is within the valid ones. As the domain is finite and the permutation is one-to-one, the cycle walking is guaranteed to terminate, but it may end up with the same original value.
  The advantage of this technique is that the valid values do not need to be mapped as a consecutive sequence. The disadvantage is that too many cycles may be required for each operation and the encryption process stops being deterministic as it is impossible to know in advance how much time will the encryption process need.
- FPE from a Feistel network [40][41].
  The output of the block cipher can be used as the source of pseudo-random values for the sub-keys for each round of the Feistel network; the resulting construction is good if enough rounds are used.
  It cannot be guaranteed that the Feistel network will preserve the format, but it is possible to iterate it in the same way as the cycle-walking technique to ensure the format can be preserved.

The United States' National Institute of Standards and Technology (NIST) Special Publication 800-38G [42] defines methods for Format-Preserving Encryption for Block Ciphers that can be used for partial or limited data universes and not limited to numeric values only. The core of the proposed

solutions is derived from an approved block cipher with 128-bit blocks, mainly, the Advanced Encryption Standard (AES) algorithm. Classic and currently used Symmetric key FPE requires additional processing which implies more time and computational power is required.

C. Public-Key Format Preserving Encryption

There are no developments in public key FPE because based on the math used for the encryption, no secure Public-Key encryption can preserve the length. Besides that, being the key public and known to everybody, an attacker can just try encrypting every possible plaintext to see if the result matches the encrypted text.

If what was encrypted is a Social Security Number (SSN), the attacker only has to try every possible nine-digit numeric value, a mere billion tries that can be accomplished in short time with currently available computational power.

To prevent this, the public key encryption algorithm must not be deterministic and must include some randomness so that a large set of possible ciphertexts may result from a given plaintext using a given public key.

Using RSA as defined by PKCS#1 [43] pads the plaintext with random bytes causing the ciphertext to be necessarily larger than the plaintext.

We can conclude that Public key encryption is not suitable for FPE.

D. Our Proposed Format Preserving Encryption

Continuing with our character or byte level data usage, what we need is just an array containing all valid byte values corresponding to the characters or symbols from the defined alphabet, and this array can have at most 256 elements when all possible byte values are valid. The number of elements in the array will be the Module for all module based substitutions.

What each of the valid values represents is irrelevant to the process:

- If what we want to encrypt, preserving the format, is a 16-digit credit card number or a 9-digit social security number, the process needs to have an array with 10 elements with different values, one for each of the decimal values from 0 to 9.
- If what we want to encrypt is a database field corresponding to a person's name and we want the valid values to be all capital letters {A…Z}, all lower case letters {a…z}, the apostrophe and the space, we will need to have an array with 54 elements with different values.
- If what we want to encrypt is a database field corresponding to an address and we want the valid values to be all capital letters {A…Z}, all lower case letters {a…z}, all ten decimal numbers {0…9}, the apostrophe, the comma, the period and the space, we will need to have an array with 66 elements with different values.

Which are those values and how they are ordered is totally irrelevant to the process because we are not using the byte value for the character itself, but its position within the defined alphabet.

E. A practical example

Let us suppose an organization wants to encrypt a specific field within a database and the group of valid values for each character in that field, expressed in decimal values are:

{32,39,44,48…57,65…90,97…122}

We can see they are 65 different values, and the group as a whole is non-continuous.

The values can be stored into an array with 65 elements and there can be 65! (8.24765 * 10^90) possible orders for those values.

Please notice there is no mention to the field length. This is because the field id and length are values that can be obtained and used during the process and has no impact over the encryption because everything is encrypted at a byte level.

In this particular example, the module M will be 65 and an encryption algorithm using encryption substitutions like those presented here and complying with the seven steps of our technique will guarantee a very fast single-pass format preserving encryption without the need of any cycle walking or additional processing to obtain a valid result.

It preserves the unconditional security and also offers the advantage that the encryption process timing can be accurately estimated based on the size of the information to be encrypted (or decrypted) independently of its content.

Due to their intrinsic simplicity, all these substitutions are really fast and an algorithm using them and complying with the technique we presented here, can be easily implemented within existing systems without the need of massive investments in computational power or data structures modification.

X. Unconditional Security

The plaintext is finite and it is never random, which is a fact. Whatever needs to be encrypted has a measurable length and it is not random. It does not matter whether it is a text, an image, an audio, a video, a blueprint, a spreadsheet or whatever it may be; it is something that can be comprehended. Because if it cannot be comprehended, there is no need to encrypt it.

Our encryption technique does not make any computational assumption and so it does not depend for its effectiveness on any computational hardness.

Vigenère's and Vernam's cipher constraints and requirements for perfect secrecy are due to the use of a single encryption substitution (Modular Addition and XOR, respectively). Given the ciphertext, the only thing that is not

known is the key. The encryption substitution order of the Vigenère and the Vernam ciphers are publicly known, it is Modular Addition after Modular Addition and XOR after XOR respectively, repeated as many times as the length of the plaintext.

What happens with the Vigenère and the Vernam ciphers if the key is not random and shorter than the plaintext?

If the key is a sequence of equal characters, it will convert both the Vigenère and the Vernam ciphers into a Caesar cipher that can be easily broken.

If the key is a sequence of alphabetic characters, it will turn Vernam cipher into a running key or Vigenère cipher, which can also be easily broken.

How can we guarantee a cipher based on our technique and complying with the seven steps will offer perfect secrecy?

A sequence of substitutions to be applied to the plaintext and the key can be generated from the third parameter, the substitution pool, the initial substitution order and the length of the key, and this sequence can be generated as long required to match the plaintext length and having no repetitions. That sequence of substitutions will also depend on the plaintext, so a different plaintext will generate a different substitution sequence for the exact same remaining parameters.

So far, we can say that for any single byte, the ciphertext value will depend on the plaintext value, the key value and the specific substitution used.

Being:

- p the plaintext value
- k the key value
- a() the initial substitution order array
- s the third parameter value
- a(s) the value stored in the s^{th} element of the array
- $f_{a(s)}$ the encryption substitution triggered by a(s)
- c the ciphertext value

Equation (21) represents the encryption as:

$$c = f_{a(s)}(p,k) \quad (21)$$

Equation (22) represents the decryption as:

$$p = f'_{a(s)}(c,k) \quad (22)$$

Considering substitution $f'_{a(s)}$ to be the reverse of substitution $f_{a(s)}$.

What happens if the key is shorter than the plaintext and it is not random?

It does not affect the unconditional security because the key does not have a direct impact on the ciphertext that can be inferred in any way. Even if the attacker manages to know the substitution pool, it offers no information about its usage.

The unconditional security is guaranteed based on that for the attacker:

- The key is unknown.
- The key length is unknown.
- The third parameter is unknown.
- The third parameter length is unknown.
- The initial substitution order is unknown.
- The processing block size or sizes is unknown.
- Which ciphertext byte corresponds to which plaintext byte is also unknown.

Even if the attacker knows the substitution pool and has infinite computational power and is able to try all possible keys and all possible substitutions for each byte and every permutation of the results and can purge all the invalid results in just a fraction of a second, the plaintext will still remain hidden at plain sight in a sea of false positives and the attacker will still be unable to decide which elephant on the beach is the right one because every possibly valid plaintext with the same length or shorter has the exact same probability of being the original plaintext without any indication of which one is the right one. Any original text may be padded adding spaces at the end without affecting the text but generating a longer plaintext. Once decrypted, those extra spaces at the end has no impact at all in the text content and meaning.

XI. Practical Applicability

Modern complexity-based cryptography requires to have a large amount of resources available, specifically computational power, processing speed and memory and as the complexity increases resource requirements also increase.

While symmetric ciphers like AES plans to use 512 bit keys and public key ciphers like RSA plan to use 4,096 bit keys in an attempt to resist a quantum attack, as the number of qubits keeps growing so will do their key length requirements. Soon Megabit-long keys will be used and expanded to Gigabit-long keys to later be expanded to Terabit-long keys, and so on, and the same will happen with the computational power and processing speed requirements.

On the other hand, our proposed encryption does not need neither long keys nor high processing speed and will not need to expand key lengths or processing speed requirements as the number of qubits in quantum processors keeps growing. It can be used even with pen and paper and some spare time.

A. Performance Requirements

Although more complex substitutions can be built up, increasing the total number of substitutions available, our example has shown how using basic mathematical operations like modular addition and modular subtraction suffice to provide unconditional security. It is not even required to use

slightly more elaborated mathematical operations like multiplication or division, less to use more complex or advanced math.

Less processing power required means simpler, smaller processors emitting less heat and requiring less electrical power consumption. Most hardware controllers from the simplest to the most complex ones already have built-in basic math operations like addition and subtraction embedded making it very simple to add encryption to them without increasing their power requirements.

Considering the computational power required by AES or RSA, what is required for elliptic curve and what may be required for lattices encryption, it becomes obvious that our proposed encryption have much less computational power requirements.

The use of such simple math guarantees true and absolute cross platform encryption/decryption.

These low power requirements allow for this encryption to be easily added to any device either through a hardware, software or mixed implementation at a very low cost without jeopardizing its security.

B. Memory Requirements

In our examples, we have shown how the plaintext, the key, the third parameter and the ciphertext can be processed just one byte at a time, when we use a single processing block size of one byte, only the fourth parameter requires a maximum of 256 bytes (2,048 bits) of memory to store the substitution order array when the alphabet used contains all possible byte values.

The minimum memory requirements for a cipher based on our technique will depend on how it is implemented, the substitutions used, the maximum processing block size allowed and the reading and writing buffer sizes.

We have seen that the plaintext can be anything, as far it is a finite sequence of bytes and the same applies to the key and the third parameters, and so the ciphertext will also be a finite sequence of bytes.

C. Applications

The list of possible applications is endless, so we will provide just a few of them we are currently working on.

- Encrypt Data at Rest
  We tried our encryption test software in Microsoft Windows encrypting files of different sizes having encrypted and decrypted files up to 1 TB (one Terabyte) without any kind of hassle and with zero errors.
- Encrypt Data in Transit
  Our encryption test software is capable of reading a local file and remotely writing the encrypted file without sending any unencrypted data.
- Encrypted Remote Control
  We are currently testing remote controlling a drone using encrypted control packets and making the drone to identify and ignore any invalid packet making its control hacking-proof.

This is a work in progress and there is still a lot of work ahead before it could be considered complete.

D. Pros and Cons

The pros can be resumed in the fact that our proposed encryption is light, fast and unconditionally secure. Other pros are that it also allows for multiple different ciphers to be built up based on it. Its low memory and processing power requirements makes it an ideal solution to add encryption to the Internet of Things (IoT) and all the smart devices it is bringing up. Its Format Preserving Encryption capabilities makes it an ideal tool to develop database encryption solutions that would not require to modify the existing data structures. The best pro is that anything encrypted using a cipher based on it will remain impervious to any quantum attack, no matter how many qubits the quantum processor may have, or whatever may come later.

Due to its simplicity, low requirements and implementation ease, it has no cons, no drawbacks and no special requirements of any kind.

XII. Key and Message Distribution

Safe message and key distribution have been an issue since the very origin of cryptography and have played a major role in the development of the field, but the advent of internet has changed everything. A message of any size can be accurately sent from one point to another through cyberspace, the availability of cloud storage and file repositories has made internet the home of trillions of files of every possible type.

Now, there is no need to even send the encrypted file or the key to the addressee, the address from where they can be downloaded will suffice. The internet address of a file can be much shorter than the file itself. With all the cloud and data files storage providers available around the world, how could a single file be located without knowing its exact name and location?

From cloud storage providers to file storage providers, one single file hidden between trillions of files can be accessed only knowing how to reach it.

Some storage providers and file address link shrinking services allow to shorten the full file name and address to just 7 or 8 alphanumeric characters, each capable of identifying about 3 trillion different files.

The sender and the addressee only need to agree on the storage or storages and exchange only those very short codes

already encrypted. A 10 GB file can be shared using an encrypted message containing only 7 or 8 characters that can be anywhere. As part of a comment on a news or blog page, as part of a tweet and an incredibly huge etcetera.

XIII. CONCLUSIONS

All cryptography in use is vulnerable to an attacker with enough computational power. Everything that has been digitally stored or transmitted or will ever be digitally stored or transmitted by any means (network, wireless, internet, etc.) may be publicly disclosed sooner than later. Unencrypted databases may be hacked and its content made public like what happened in August 2016 when the World Anti-Doping Agency (WADA) was hacked by Russian hackers and private health records from famous athletes were made public to distress and discredit them.

The simplicity and ease of implementation of poly-substitution encryption sheds a light for the development of true cross-platform encryption solutions and add-ons fully compatible across hardware and software platforms and operating systems.

Poly-substitution encryption may be easily added to existing hardware, software and mixed solutions. Poly-substitution encryption cannot prevent hacking or system intrusions but may make the effort fruitless and useless.

A low cost and unconditionally secure format preserving encryption is urgently needed to preserve the privacy of sensitive but personal information. We want to help protect the privacy of personal information around the world.

Assuming there is currently enough available computational power to try in a very short time every single key length and value, with every single processing block size and every single possible encryption substitution there will still not be possible to decide which one of the apparently valid results is the true original plaintext.

Even knowing that the plaintext is just plain text, any possible text with the same length or shorter (just filled with spaces at the end, at the beginning, or within, in order to reach the same length) has the exact same possibility of being the original plaintext. That is the essence of unconditional security, something none of the currently in use encryption standards or solutions can offer.

We have seen here that this poly-substitution encryption technique offers the same level of unconditional security guaranteed by the Vernam cipher without its constraints.

With billions and billions of files available through the internet and the capability of using any of them as a key, as a third parameter and even as the original substitution order, nobody needs to remember long keys, just needs to remember which files were used and how to reach them.

If one has enough computational power like quantum computing promises to offer when it becomes widely available, one may be able to break and read any file encrypted with any of the current standards, techniques and tools with two exceptions:

- Anything protected through a One-Time-Pad (or a proper use of Vernam's or Vigenère's cipher) will remain secret.
- Anything protected through the use of a cipher based on our proposed technique and complying with its seven steps will remain secret.

XIV. FUTURE WORK

As we stated before, this is a work in progress and there is still a lot of work ahead before it could be considered complete.

We have already implemented an encryption solution complying with the seven steps defined here and used it for our tests. It is a Windows app programmed in Visual Basic 6.0 that uses 256 different encryption substitutions and is capable of encrypting and decrypting any kind of files up to about 900 TB (900,000,000,000,000 bytes long) and fast enough to cipher/decipher an 80 MB file in less than five seconds. There is plenty of room to enhance and improve the encryption speed by optimizing the code and using programming languages that may run faster.

We do not have the resources to test up the maximum possible file size but already tested it on a 1 TB (one Terabyte) text file, encrypting and decrypting it without any issue or error. We will continue testing encrypting and decrypting larger files of different types, including databases, audio and video files, images and compressed files, etc.

Future work will aim to validate the ideas presented in this paper by means of additional practical results, simulations, statistical analysis and practical performance comparisons with other ciphers.

We are open to share our development with the cryptographic community to be fully analyzed, tested, improved and enhanced.

Future work will also aim to develop and test practical solutions for low cost Format Preserving Encryption algorithms based on the technique presented here.

REFERENCES

[1] J. Murguia Hughes, "Seven Steps to a Quantum-Resistant Cipher", SECURWARE 2016, The Tenth International Conference on Emerging Security Information, Systems and Technologies, pp. 247-253, ISSN 2162-2116, ISBN 978-1-61208-493-0.

[2] W. Whitfield and M. E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory. 22 (6): 644–654.

[3] H. Sidhpurwala, "A Brief History of Cryptography" redhat Security Blog, August 14th, 2013.

[4] G. B. Belasso, "La cifra del Sig. Giova Battista Belasso", 1553.

[5] G. S. Vernam, "Cipher Printing Telegraph Systems for Secret Wire and Radio Communications", Journal of the IEEE 55: 109-115.

[6] G. S. Vernam, Patent 1,310,719. "Secret Signaling System", Patented July 22, 1919. United States Patent and Trademark Office.

[7] C. E. Shannon, “A Mathematical Theory of Communication”, The Bell System Technical Journal, Vol. XXVII, No. 3, July 1948, pp. 379-423 and October 1948, pp. 623-656.

[8] C. E. Shannon, “Communication Theory of Secrecy Systems”, The Bell System Technical Journal, Vol. XXVIII, No. 4, pp. 656-715.

[9] F. W. Kasiski, “Die Geheimschriften und die Dechiffrierkunst”, 1863.

[10] Swenson, C: “Modern Cryptanalysis - Techniques for Advanced Code Breaking”, John Wiley & Sons Inc. 2008.

[11] Friedman, W. F.: “The index of coincidence and its applications in Cryptanalysis”, Cryptographic Series, 1922.

[12] ETSI, “Quantum Safe Cryptography and Security”, ETSI Whitepaper No. 8, June 2015, ISBN No. 979-10-92620-03-0.

[13] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", SIAM J. Comput. 26 (5): 1484–1509.

[14] “Announcing the ADVANCED ENCRYPTION STANDARD (AES)”, FIPS PUB 197, United States National Institute of Standards and Technology (NIST), November 26, 2001.

[15] B. Schneier, “Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)”, Fast Software Encryption, Cambridge Security Workshop Proceedings (Springer-Verlag): 191-204, 1993.

[16] S. Lurye, “An uncertain path to quantum supremacy: Notes from RSA”, Kaspersky Lab Daily, May 8, 2018, https://www.kaspersky.com/blog/quantum-supremacy-rsa/22339/. 2018.05.31

[17] J. A. Buchmann, D. Butin, F. Göpfert and A. Petzoldt, “Post-Quantum Cryptography: State of the Art”, Springer LNCS, volume 9100: 88–108.

[18] L. Carrol, “Through the Looking-Glass”, Macmillan, 1871.

[19] “Data Encryption Standard (DES)”, FIPS PUB 46, United States National Institute of Standards and Technology (NIST), January 15, 1977.

[20] P. Benioff, “The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines". Journal of statistical physics. 22 (5): 563–591. 1980.

[21] Y. I. Manin, “Vychislimoe i nevychislimoe [Computable and Noncomputable]” (in Russian). Sov.Radio. pp. 13–15. 1980.

[22] R. P. Feynman, “Simulating physics with computers". International Journal of Theoretical Physics. 21 (6): 467–488. 1982.

[23] D. Deutsch, “Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer". Proceedings of the Royal Society of London A. 400 (1818): 97–117. 1985.

[24] E. Pednault, J. A. Gunnels, G. Nannicini, L. Horesh, T. Magerlein, E. Solomonik, and R. Wisnieff, “Breaking the 49-Qubit Barrier in the Simulation of Quantum Circuits”, arXiv:1710.05867 [quant-ph].

[25] W. Knight, “IBM Raises the Bar with a 50-Qubit Quantum Computer”, MIT Technology Review, November 10, 2017, https://www.technologyreview.com/s/609451/ibm-raises-the-bar-with-a-50-qubit-quantum-computer/. 2018.05.16

[26] D-Wave Systems, The Quantum Computing Company. http://www.dwavesys.com/. 2018.05.16.

[27] “Not Magic, Quantum”, 1663 - The Los Alamos Science and Technology Magazine, July 2016, pp. 14-19.

[28] B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich and W. K. Hensinger, "Blueprint for a microwave trapped ion quantum computer", Science Advances, 01 Feb 2017,Vol. 3, no. 2, e1601540.

[29] The Oxford Math Center from Oxford College of Emory University, ”Letter Frequencies in English”, http://www.oxfordmathcenter.com/drupal7/node/353. 2018.06.07

[30] Wikipedia, “Frequency Analysis”, https://en.wikipedia.org/wiki/Frequency_analysis. 2018.05.16.

[31] Project Gutemberg, “Project Gutenberg Complete Works of Winston Churchill by Winston Churchill”, https://www.gutenberg.org/ebooks/5400. 2018.05.16.

[32] CrypTool-Online, https://www.cryptool.org/en/, 2018.05.16.

[33] E. Barker and N. Mouha, "NIST Special Publication 800-67 Revision 2: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher", National Institute of Standards and Technology (NIST).

[34] B. Schneier, “Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)”, Fast Software Encryption, Cambridge Security Workshop Proceedings (Springer-Verlag): 191-204, 1993.

[35] Rl L. Rivest, A. Shamir and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems". Communications of the ACM. 21 (2): 120–126. 1978. ISSN 0001-0782.

[36] M. Matsui and A. Yamagishi, “A new method for known plaintext attack on FEAL cipher”, Advances in Cryptology – EUROCRYPT 1992.

[37] M. Matsui, “The first experimental cryptanalysis of the Data Encryption Standard”, Advances in Cryptology – CRYPTO 1994.

[38] D. Coppersmith, “The Data Encryption Standard (DES) and its strength against attacks”, IBM Journal of Research and Development. Vol38 No. 3 May 1994, pp. 243–250.

[39] J. Black and P. Rogaway, “Ciphers with Arbitrary Domains”, Proceedings RSA-CT, 2002, pp. 114–130.

[40] H. Feistel, “Cryptography and Computer Privacy”, Scientific American, Vol. 228, No. 5, 1973.

[41] A. J. Menezes and P. C. Oorschot and S. A. Vanstone, “Handbook of Applied Cryptography (Fifth ed.)”, 2001, p. 251.

[42] M. Dworkin, “Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption”, NIST Special Publication 800-38G, United States National Institute of Standards and Technology (NIST), March 2016.

[43] RSA Laboratories, “PKCS#1; RSA Cryptography Standard”, RSA Laboratories.

An Elaborated Framework for Protecting Privacy in the IoT

George O. M. Yee
Computer Research Lab, Aptusinnova Inc., Ottawa, Canada
Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
email: george@aptusinnova.com, gmyee@sce.carleton.ca

***Abstract*—The Internet of Things (IoT) is attracting great interest within the research community. Yet, there is little research on how data generated by the "things" can be shared while respecting the privacy wishes of the data's owners. Consider a smart refrigerator as one of the "things". It keeps track of which food items are consumed, in order that the consumer can know when and what foods need to be replenished. Suppose the smart refrigerator sends this consumption information to online grocers that can automatically schedule deliveries to replenish the food. The consumption information may contain personal information (e.g., foods identifying a particular medical condition) leading to privacy concerns. This paper extends the CYBER 2016 paper "An Approach for Protecting Privacy in the IoT". The original version proposed an approach that utilizes personal privacy policies and policy compliance checking to protect privacy in the IoT, using the smart refrigerator as an example to illustrate the approach. This paper adds additional explanations and diagrams, a health monitoring example, and more discussion on related work.**

Keywords-privacy protection; IoT; privacy policy; compliance; controller.

I. INTRODUCTION

The objective of this paper is to present an elaborated framework that makes use of privacy policies and policy compliance checking to protect privacy in the IoT. Privacy protection is in the context of smart devices (defined below) that supply data to e-services (defined below). The smart devices themselves may also be providing e-services. The objective of this paper is achieved by focusing on a smart device as sending data that needs privacy protection.

This work extends Yee [1] by expanding all sections with additional details. In particular, an additional example using health monitoring has been added, and the section on related works has been enlarged.

A "smart" device is any physical device endowed with computing and communication capabilities. Some smart devices may have more computing and communication capabilities (e.g., smartphones) than others (e.g., sensors). An e-service is a grouping of computation that optionally takes input and produces output (the service). For example, the connected smart refrigerator would access the food replenish e-service from the online grocer and transmit its food consumption information (the input) to the food replenish e-service. In response, the food replenish e-service would schedule food deliveries (the output). As another example, a sensor would provide an e-service of transmitting data to another e-service that requested the data. In this case, the sensor e-service would not require any input (except for the request to transmit data).

This work addresses an Internet of things environment (see Fig. 1) with the following characteristics:

- Smart devices (e.g., laptops, smartphones, workstations, smart sensors, smart appliances, smart home switches and cameras, smart speakers) are optionally locally networked (e.g., Ethernet, Wi-Fi, IrDA, Bluetooth) or standalone (i.e., not locally networked). The locally networked or standalone smart devices are connected to the Internet via an Internet Service Provider (ISP).
- The locally networked or standalone smart devices are owned by a human or an organization.
- Human users employ these devices to make use of e-services, to offer e-services, or both. A user who makes use of an e-service sends information to that e-service and is called a *data sender*. One who offers an e-service receives information needed by that e-service and is called a *data receiver*. A user who both makes use of e-services and offers e-services is both a data sender and a data receiver.

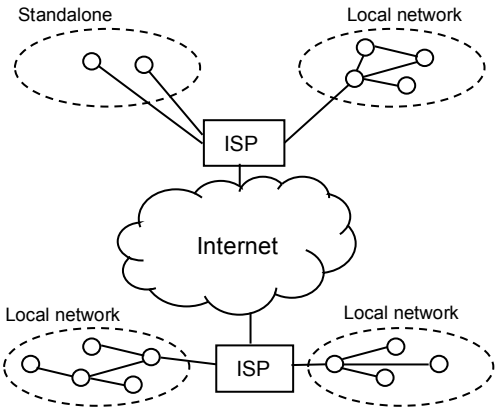


Figure 1. IoT network environment (ISP = Internet Service Provider, circles are smart devices)

The remainder of this paper is organized as follows. Section II looks at privacy and the use of privacy policies. Section III presents the proposed framework. Section IV

gives two examples of applying the framework. Section V discusses some strengths and weaknesses of the framework. Section VI examines related work. Section VII concludes the paper and lists some ideas for future research.

II. PRIVACY POLICIES

A. Privacy

As defined by Goldberg et al. in 1997 [2], privacy refers to the ability of individuals to *control* the collection, retention, and distribution of information about themselves. This is the definition of privacy used for this work. Protecting an individual's privacy then involves endowing the individual with the ability to control the collection, retention, and distribution of her personal information.

B. Use of Privacy Policies

In this work, a data sender is given control over her private information as follows. The data sender specifies in her sender privacy policy how she wants her personal information handled by the data receiver; the data receiver, on the other hand, specifies in her receiver privacy policy what personal information her service requires from the data sender and how she plans to handle the data sender's information. The data sender's policy has to be compatible or match the data receiver's policy before information sending can begin. If the policies do not match, the data sender can either negotiate with the data receiver to try to resolve the disagreement or choose a different data receiver. Once the information is sent, the data receiver has to comply with the sender's privacy policy (which is compatible with her own receiver privacy policy). Foolproof mechanisms must be in place to ensure compliance. The detailed mechanics of privacy policy matching [3] and negotiation [4] are outside the scope of this work, although we do explain below the meaning of matching.

Fig. 2 shows example sender and receiver privacy policies for a smart refrigerator. We have not expressed these policies in any specific policy language, preferring to keep our meaning clear and unencumbered with language details (see Section III D and Section VI). Referring to Fig. 2, a privacy policy for sending personal information consists of a header section (shaded) followed by one or more privacy rules, where there is one rule for each item of personal information. The fields within the header have the following meaning: *Policy Use* identifies the e-service (e.g., replenish food), *Data Sender / Data Receiver* gives the name of the party that owns the policy, and *Valid* indicates the period of time during which the policy is valid. The fields in each privacy rule have the following meaning: *Data Receiver* identifies the party that receives the information, *What* describes the nature of the information, *Purpose* identifies the purpose for which the information is being sent or received, *Retention Time* specifies the amount of time the data receiver can keep the information, and *Disclose-To* identifies any parties who will receive the information from the data receiver. Fig. 3 shows example sender and receiver privacy policies for a smart watch, which is able to monitor the wearer's heart rate, skin temperature, sleep pattern, and exercise pattern. In this case, the e-service identified in *Policy Use* is Health Monitor, which is an online service that continuously monitors a person's health by gathering and processing health indicators such as heart rate and skin temperature. The other fields in the privacy policies have the same meaning as described above for Fig. 2.

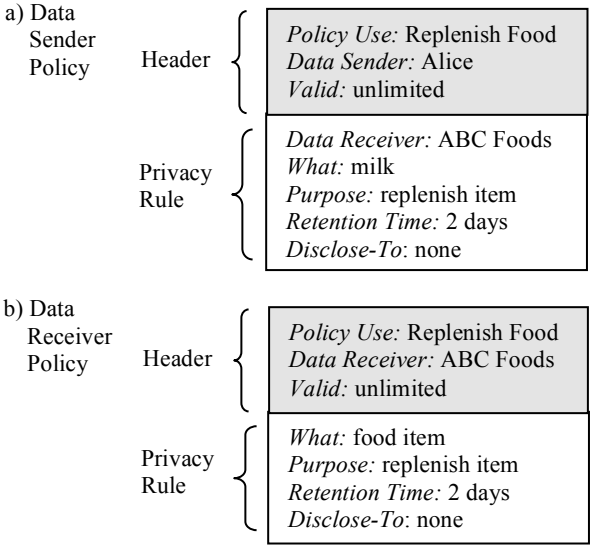


Figure 2. Example data sender / data receiver privacy policies for a smart refrigerator. Each policy can have as many privacy rules as are needed.

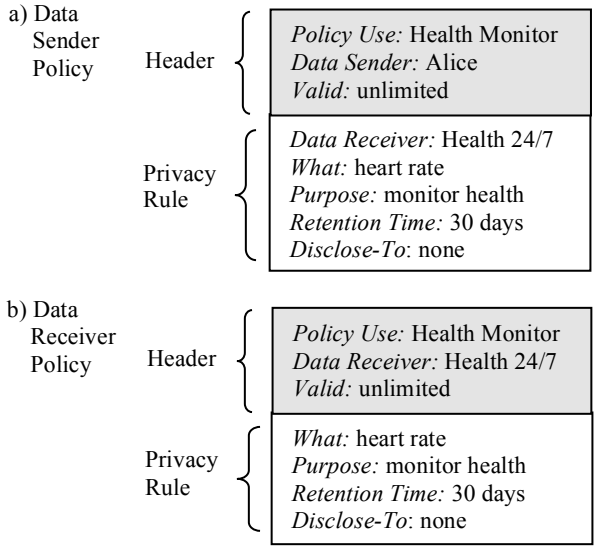


Figure 3. Example data sender / data receiver privacy policies for a smart watch. Each policy can have as many privacy rules as are needed.

It was mentioned above that the sender and receiver privacy policies have to "match". Matching means that the

values of the fields *What*, *Purpose*, and *Disclose-To* are the same in both sender and receiver policies for the same data item. It further means that the *Retention Time* of the receiver policy is not longer than the *Retention Time* of the sender policy for any data item. Finally, both sender and receiver policies must be valid during the utilization period of the e-service. For example, the policies in Fig. 3 would not match if the receiver policy were to have the following values: *Retention Time:* 40 days, *Disclose-To:* John, where John is Alice's husband (Alice does not want John to be concerned if she has an abnormal heart rate).

The above privacy rules and fields conform to Canadian privacy legislation, which is representative of privacy legislation in many parts of the world, including the European Union and the United States. The fields *What*, *Purpose*, *Retention Time*, and *Disclose-T*o correspond to fair information principles 4, 2, 5, and 5, respectively, as shown in Table I. Policy matching corresponds to principle 3 (consent). The fair information principles form the foundation of the Canadian Personal Information Protection and Electronic Documents Act (PIPEDA) [5]. Thus, a data receiver who complies with a data sender's privacy policy also complies with the sender's legislated privacy rights. Furthermore, the framework proposed here would apply in the European Union, the United States, and elsewhere in the world where privacy legislation similar to PIPEDA exist, with only minor changes to the content of the privacy policies.

TABLE I. PIPEDA FAIR INFORMATION PRINCIPLES

Principle	***Description***
1. Accountability	An organization is responsible for personal information under its control. It must appoint someone to be accountable for its compliance with these fair information principles.
2. Identifying Purposes	The purposes for which personal information is collected must be identified by the organization before or at the time of collection.
3. Consent	The knowledge and consent of the individual are required for the collection, use, or disclosure of personal information, except when inappropriate.
4. Limiting Collection	The collection of personal information must be limited to that which is needed for the purposes identified by the organization. Information must be collected by fair and lawful means.
5. Limiting Use, Disclosure, and Retention	Unless the individual consents otherwise or it is required by law, personal information can only be used or disclosed for the purposes for which it was collected. Personal information must only be kept as long as required to serve those purposes.
6. Accuracy	Personal information must be as accurate, complete, and up-to-date as possible in order to properly satisfy the purposes for which it is to be used.
7. Safeguards	Personal information must be protected by appropriate security relative to the sensitivity of the information.
8. Openness	An organization must make detailed information about its policies and practices relating to the management of personal information publicly and readily available.
9. Individual Access	Upon request, an individual must be informed of the existence, use and disclosure of their personal information and be given access to that information. An individual shall be able to challenge the accuracy and completeness of the information and have it amended as appropriate.
10. Challenging Compliance	An individual shall be able to challenge an organization's compliance with the above principles. Their challenge should be addressed to the person accountable for the organization's compliance with PIPEDA, usually their Chief Privacy Officer.

III. FRAMEWORK

For each smart device, the framework consists of two phases: a privacy policy agreement (PPA) phase and a privacy policy compliance (PPC) phase. These phases apply to both data senders and data receivers.

A. PPA Phase and Design of Policy Controller

The PPA phase consists of the composition and exchange of privacy policies between data sender and data receiver, using a Policy Controller (PC), which runs on a desktop, laptop, a mobile device such as a smart phone or tablet, or on the IoT node itself if it has sufficient computing power. The components and functionality of the PC are given in Table II.

TABLE II. POLICY CONTROLLER (PC)

PC Component	**Functionality**
Policy Module (PM) - Data Sender	Partially composes the data sender policy; searches for e-services (data receivers) and obtains their receiver policies; determines if data receiver policies match the sender policy; selects a data receiver with a matching policy and completes the data sender policy by filling in the name of the data receiver; sends the sender privacy policy to the selected data receiver; sends the sender policy to the smart device; optionally sets up a privacy policy negotiation between the data sender and a data receiver for a particular policy pair that does not match, in order to try to arrive at a match (where possible)
PM - Data Receiver	Composes the data receiver privacy policy; sends the data receiver privacy policy to the PM of the data sender when requested; receives the data sender privacy policy and verifies that the sender policy matches its own policy; optionally cooperates to set up a privacy policy negotiation with the owner of a data sender
Policy Store (PS) – Data Sender	Holds the data sender privacy policy; holds the privacy policies received from data receivers
PS – Data Receiver	Holds the data receiver privacy policy; holds the privacy policies received from data senders

Fig. 4 presents a message sequence chart showing the interactions between the PMs of a data sender and a data receiver (only one receiver shown and policy composition excluded for simplicity). A first time successful privacy policies match is assumed.

Fig. 5 shows the same scenario as Fig. 4 except that the first time policy match is unsuccessful, resulting in the need for policy negotiation, assumed to be successful. If the

negotiation was unsuccessful, the sender would not be able to proceed any further with the receiver and would have to select a new receiver or find some way to satisfy the receiver's policy.

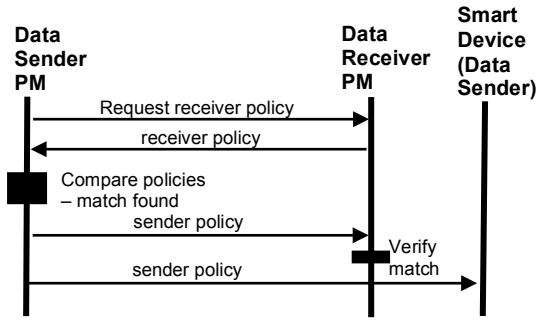


Figure 4. Message sequence chart showing the interactions for a first time successful policy match.

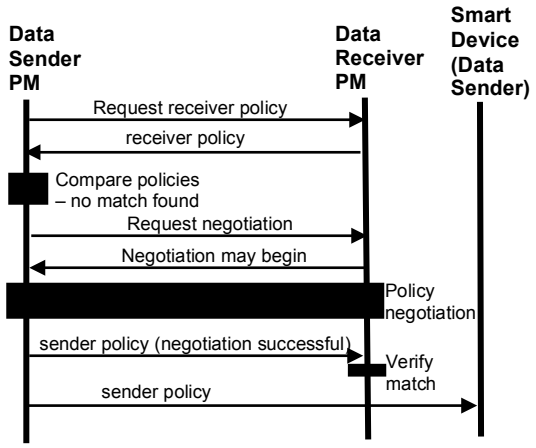


Figure 5. Message sequence chart showing the interactions for a first time unsuccessful policy match and the ensuing negotiation (assumed successful).

In the case where the data sender's PM finds a data receiver that has received data from the data sender in the past, it is very likely that the associated privacy policies already match. To account for this case, the PM always verifies if the data receiver found is one that has received data from the data sender in the past, prior to determining if the policies match. If this verification is positive, the PM further verifies if the sender and receiver policies are the same as in the previous interaction, and if yes, bypasses determining if the policies match. These checks may improve the performance of the data sender's PM. These checks are incorporated in the task "determines if data receiver policies match the sender policy" for the data sender's PM in Table II. They are also part of the "Compare policies" module in Fig. 4 and Fig. 5. Similarly, provided that no negotiation occurred, the data receiver's PM verifies if the receiver has received data from the data sender in the past, prior to verifying if the policies match. If so, the receiver's PM will further verify if the policies are the same as during the last interaction, and if they are the same, does not verify if the policies match. This may improve the performance of the data receiver's PM if no negotiation occurred. These verifications are incorporated in the task "verifies that the sender policy matches its own policy" for the data receiver's PM in Table II. They are also part of the "Verify match" module (Data Receiver PM) in Fig. 4 and Fig. 5.

B. PPC Phase and Design of Compliance Controller

In the PPC phase, the data sender sends its data to the data receiver, while ensuring that both sender and receiver privacy policies are respected. This phase is carried out using software called a Compliance Controller (CC), which runs on the smart device or on a computing platform (e.g., tablet) that is "linked" to the device. The components and functionality of the CC are given in Table III. The data sender's CC makes a single connection to the data receiver's CC per data sending session. In Table III, for a particular smart device, Compliance Module (CM) functionality depends on whether the device sends data, receives data, or both sends and receives data. In the latter case, each component would have the functionalities prescribed for a data sender and data receiver combined.

TABLE III. COMPLIANCE CONTROLLER (CC)

CC Component	Functionality
Compliance Module (CM) – Data Sender	Requests the Link Module (described below) to set up a connection with the data receiver; periodically requests the secure log (SL) from the data receiver to verify policy compliance; automatically verifies compliance and warns the user if the verification fails
CM – Data Receiver	Ensures that a data receiver complies with the privacy policy of a data sender; maintains a SL of all transactions involving the sender's private data; sends the SL to the sender when requested
Link Module (LM) – Data Sender	Sets up a connection for sending data to the selected data receiver with a matching privacy policy; tears down the connection once the associated data sending session is finished
LM – Data Receiver	Cooperates with the LM of the data sender to set up the connection for data reception, e.g., provides the port number to use in case there is a need to bypass a firewall
Data Store (DS) – Data Sender	Holds the sender's private information that is to be sent to the data receiver; holds the sender privacy policy received from the sender's PC
DS – Data Receiver	Holds the private information received from the data sender; holds the data receiver privacy policy

The CM uses the secure log to verify that the data receiver complies with the data sender's privacy policy, in terms of the policy fields *Purpose*, *Retention Time*, and *Disclose-To* (see Section II B). Note that it is not necessary to verify *Data Receiver* and *What*, since the data sender sends only the data items mentioned in the matched sender and receiver privacy policies to the data receiver in her policy. The data receiver is responsible for generating the

secure log, with the following requirements: a) there is a header containing the data sharing session identifier, the name of the data sender, and a time-stamp of when the log was started, b) has entries, where each entry is of the form *[time-stamp, operation, (personal data item)]* where *personal data item* may or may not be present, c) an entry is made every time an operation occurs, and d) the log is secured in that an entry once written cannot be altered. Fig. 6 shows an example of a secure log for heart rate (pulse) data, corresponding to the sender privacy policy in Fig. 3a. The secure log entries in Fig. 6 show two types of operations: a "Verify Pulse" operation and an "Erase-29.04.2018" operation. Each entry starts with a time-stamp in the format *dd.mm.yy:hour.minute.second*. Alice's heart rate (pulse) arrives at the data receiver's computer system approximately every 2 minutes. With each arrival, the Verify Pulse operation checks to see if the received pulse (at the end of the entry) is within expected bounds. The Erase-29.04.2018 operation erases all the heart rate data collected on 29.04.2018 (April 29, 2018), which is 31 days from the current date of 30.05.2018 (May 30, 2018). Using this secure log to check the data receiver's compliance with Alice's sender privacy policy (fig. 3a), the CM is able to verify compliance as follows: *Purpose* – the Verify Pulse operation in the secure log is compatible with the purpose of monitoring health and there are no operations that suggest a different purpose, *Retention Time* – the Erase-29.04.2018 operation has deleted pulse data that is 31 days old (the CM keeps track of past Erase operations and knows that the data receiver has always erased data older than 30 days), and *Disclose-To* – there are no operations that suggest that the data has been disclosed to any other party.

Session ID: 21673 Data Sender: Alice Log Started: 30.05.2018:09.29.11
30.05.2018:09.31.10-Verify Pulse-70
30.05.2018:09.33.09-Verify Pulse-72
30.05.2018:09.35.10-Verify Pulse-75
30.05.2018:09.37.11-Verify Pulse-71
30.05.2018:09.39.10-Verify Pulse-69
30.05.2018:09.41.09-Verify Pulse-68
30.05.2018:09.43.10-Verify Pulse-70
30.05.2018:09.45.10-Verify Pulse-70
30.05.2018:09.47.11-Erase-29.04.2018
30.05.2018:09.49.10-Verify Pulse-73
30.05.2018:09.51.09-Verify Pulse-72
30.05.2018:09.53.10-Verify Pulse-71
30.05.2018:09.55.10-Verify Pulse-70
30.05.2018:09.57.11-Verify Pulse-69

Figure 6. Example secure log for heart rate data corresponding to the data sender privacy policy of Fig. 3a.

In addition to the CC itself, the following are also required: a) local and global networking as shown in Fig. 1, and b) interfaces to connect the CC to the smart device. Local and global networking are assumed to be what is most commonly available, i.e., Ethernet, Wi-Fi, IrDA, or Bluetooth for local, and the Internet for global. Smart devices need to have appropriate interfaces that inter-work with the Compliance Controller to carry out policy compliance management (e.g., checking a secure log to verify compliance), connection setup for sending data, and the storage and retrieval of private data.

Fig. 7 presents a message sequence chart showing the interactions between the LMs and CMs of a data sender and a data receiver (only one receiver is shown for simplicity) for a data sending session. As shown, the CC makes a single connection

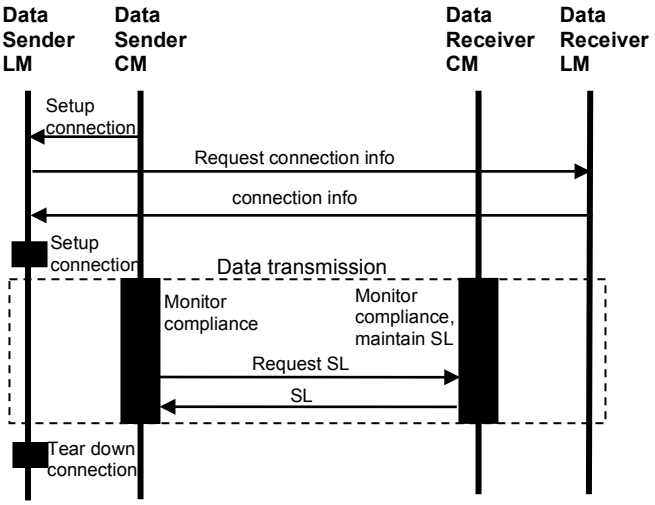


Figure 7. Message sequence chart showing the interactions for a connection setup, data transmission, policy compliance monitoring, and connection teardown.

C. System Configuration

This section considers the system configuration or where the different modules run. If the IoT node that is to send and receive data has sufficient computing capability, the PC and CC may both reside in and run in the node. Examples of such nodes are laptops and smartphones. If this node is less capable computationally but is more capable than the least capable node, one may experiment with having the PC reside in and run in a desktop, laptop or smartphone, while the CC resides in and runs in the node. An example of such a node is a smart refrigerator. Finally, if the IoT node that is to send and receive data is very limited in terms of computational power, both the PC and CC may reside in and run on a desktop, laptop or smartphone, using basic control signals to trigger the node to send data, receive data, or both. In this case, the data would be sent or received through the desktop, laptop, or smartphone. An example of such a node is a simple temperature sensor. Fig. 8 illustrates these three configurations for low, medium, and high computing power. Further, for the medium and low power cases in Fig. 8, each desktop, laptop, or smartphone may run

multiple instances of PC or multiple instances of a PC-CC pair, as required, corresponding to multiple IoT nodes.

The non-privacy preserving IoT network of Fig. 1 is converted to a privacy-preserving IoT network by adding a PC and CC to each smart device or node (Fig. 9) using one of the configurations shown in Fig. 8. In Fig. 9, the double arrows in the PC and CC blow-ups represent expected communication directions based on the functionalities described in Tables II and III. However, the actual communications will depend on how the PC and CC are implemented. Note that as mentioned above, more than one PC or more than one PC-CC pair may run in a desktop, laptop, or smart phone, so the relationship between desktop, laptop, or smart phone and IoT node may be one-to-many. However, Fig. 8 shows this relationship as one-to-one to keep the complexity manageable.

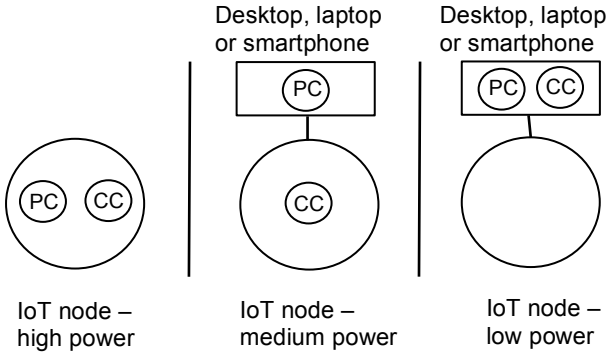


Figure 8. System configurations for the deployment of PC and CC depending on the computational power of the IoT node.

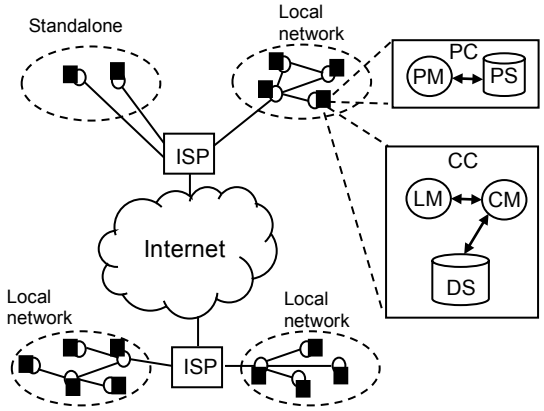


Figure 9. Proposed privacy preserving IoT network; each smart device (small circle) has a PC and CC (solid black rectangle) using one of the configurations in Fig. 7; blow-ups of a PC and CC are also shown (all acronyms defined above).

Although the functionalities of the PC and CC are different between data sender and data receiver, as stated in Tables II and III, the configurations in Fig. 8 apply to both data sender and data receiver. However, in the case where the data receiver is a commercial enterprise, its IoT nodes may consist of desktops or laptops.

Prior to using a smart device to send or receive data, the user accesses the device (possibly through a laptop or smartphone) using some secure form of authentication, such as 2-factor authentication requiring a password and a fingerprint scan. This is needed to protect the user's personal data that is stored in the device and can be satisfied by authentication software within the user's device or within the laptop or smartphone (e.g., part of operating system). As well, any additional security needed to secure the data sender's personal information and privacy policies from attack must be in place. This is satisfied by additional security measures such as certificates and encryption (discussed in Section III D below).

D. Implementation Notes

Some implementation aspects of the framework are considered here.

How does the owner of a data sender come up with her sender privacy policy? It is proposed that data receivers (e-services) routinely advertise their data requirements on the Internet. Note that this is in a way being done today by service websites (e.g., when the user is asked to fill out an online form) and would appear to be a natural way for the receiver to share requirements. Data sender owners can then use the PM to compose the sender policy based on these data requirements. The owners of data receivers also use the PM to compose receiver privacy policies based on how they would like to handle the private information that they receive. Further, data senders and data receivers may only need to create new privacy policies infrequently as they can re-use previous policies from past interactions with the same data receiver or data sender.

The heterogeneous nature of today's smart devices may present some implementation problems for the proposed framework. Some devices may not have sufficient computing power even to be considered as a low power device per Fig. 8. In this case, we concede that such a device would need to be excluded from participation in the proposed framework. A low power device must at least have sufficient computing power to receive signals telling it to send or receive data via the desktop, laptop, or smartphone, as shown in Fig. 8. However, such functionality would require very little computing power, and we expect that the majority of smart devices would possess the power needed.

Further to the need for interfaces to connect the CC to the smart device, mentioned in Section III B, the interfaces and communication links between the desktop, laptop, or smartphone and the smart device would need to be

implemented for medium and low power devices, as shown in Fig. 8. These interfaces and links may be overlaid on top of the ones used for the devices to communicate with one another and the Internet (networked devices, see Section I) and to communicate with the Internet (stand alone devices).

The search for data receivers in the PM may return a reputation value for each receiver. This would help the owner of a data sender to choose which receiver to include in her sender privacy policy. The reputation value may be calculated based on the receiver's history of past transactions, as is done on eBay.com for buyers and sellers. Gupta et al. [6] investigate the design of a reputation system for P2P networks like Gnutella. These authors believe that having reliable reputation information about peers can guide decision making such as whom to trust for a file, similar to this work. However, the choice of a data receiver such as an online grocery store, may depend on other factors such as availability of product, and even personal relationships, e.g. a friend of the data sender works at the grocery store. Nevertheless, a reputation value would be a good place to start.

What does matching of policies mean between data sender and data receiver? This has already been discussed in Section II B. However, an alternative way of comparing two privacy policies is to use a measure of compatibility such as levels of privacy [3]. For this work, matching policies has the meaning explained in Section II B.

Privacy policies need to be amenable to machine processing. Policy languages such as APPEL [7] that are XML-based are good choices. Section VI gives some references for choosing a suitable policy language for implementation.

Any additional security needed to secure the data sender owner's private information and her privacy policies from attack must be installed. Suitable authentication mechanisms, such as the use of certificates, will be needed for data sender / data receiver authentication. Other security mechanisms such as the use of encryption to encrypt the private information will need to be applied or developed and applied. Table IV suggests some security mechanisms that may be employed.

TABLE IV. ADDITIONAL SECURITY MECHANISMS

System Component Requiring Protection	Security Protection Mechanism
data sender / data receiver authentication	SSL with 2-way authentication
Internet communication channels	SSL with 2-way authentication
privacy policies stored in PS and DS	encryption (e.g., 3DES)
personal information stored in DS	encryption (e.g., 3DES)
Software for smart device, PC, and CC	anti-malware tools (e.g., Kaspersky)

In addition, the CC and in particular, the CM, need to be protected from malicious tampering. Since the CM plays the important role of checking for compliance, critical elements of the CM may be implemented in hardware to resist tampering (e.g., by using the Trusted Platform Module [8]). In fact, to further resist tampering, the entire CC may be implemented as a stand-alone hardware module that plugs into the smart device to operate (e.g., via a USB port). It can then be standardized and certified by a trusted authority such as a privacy commissioner to increase user trust.

IV. APPLICATION EXAMPLES

This section provides 2 application examples of the framework described in Section III, a smart refrigerator and a smart watch.

A. Smart Refrigerator

Suppose Alice has a smart refrigerator, which is running low on a number of food items. Alice's refrigerator is connected to the Internet through WI-FI as a node in the privacy-preserving IoT network proposed in this work (see Fig. 10). Before ordering these food items replenished, Alice's refrigerator compares their prices at three online grocers and orders the items from the grocers with the lowest price for each item. The following steps are performed:

1) Alice accesses her laptop (after entering her password), gets on the Internet, and launches her PC. Using network software that was packaged with her PC, she requests to see all grocers located within 10 kilometers of her home who are online. Alice receives a listing of online grocers located within 10 kilometers of her home. (Note: The details of grocer lookup and online messaging are assumed to go on in the background).

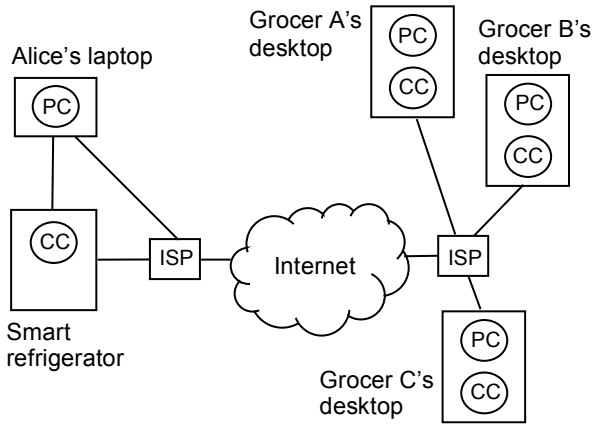


Figure 10. System configuration for smart refrigerator example; the connections from Alice's laptop and from the refrigerator may be Wi-Fi.

2) Alice uses her PC to retrieve her pre-specified privacy policy from her laptop's local storage (PS) and completes it by choosing and including three online grocers (e-services), based on her comfort level with their brand names (e.g., Loblaws, Metro, but shown generically in Fig. 10). Alice actually completes a policy for each grocer, each policy differing only in the *Data Receiver* field. We refer to these policies as Alice's privacy policy.

3) Alice's PC requests the privacy policy of each online grocer that Alice specified in her privacy policy after mutual authentication with each grocer. With the arrival of each grocer's policy, Alice's PC compares Alice's policy with the grocer's policy to see if the policies match up. All grocers' policies match except for one. Alice is asked if she wants to negotiate with the non-matching grocer to try to resolve the non-match. Alice agrees to negotiate and is able to negotiate to a successful conclusion. Now all policies match. Alice's PC sends her sender policy to the PC of each grocer whose policy matches Alice's policy. For added safety, the PC of each grocer receiving Alice's policy does a quick verification of the policy match. If a non-match is found here (unlikely since already checked by Alice's PC) the grocer's PC could terminate the interaction with Alice. Alice's PC sends the sender policy to the CC of the smart refrigerator.

4) The CC in Alice's refrigerator sets up connections between Alice's refrigerator and the three online grocers with the cooperation of the grocers' CCs. Alice's refrigerator then starts sending data to the grocers.

Alice's refrigerator sends personal consumption information to the grocers, such as Alice's favorite brand of food item, her consumption rate for each food item, and the prices that she expects to pay. In return, the online grocers provide Alice's refrigerator with the food items' prices. Alice's refrigerator completes the data transmission, ordering food items from the grocers with the lowest prices. In addition, during and after the transmission, the CM modules of the grocers' respective CCs, continuously checks the grocers' handling of Alice's personal information to ensure compliance with Alice's sender privacy policy. These CM modules log all private data activities to secure logs and sends them to Alice's CC when requested. Alice's CC verifies these secure logs for policy compliance and notifies Alice upon detection of any discrepancy, so that Alice can challenge the grocers' handling of her data when warranted.

B. Smart Watch

Suppose Alice has a smart watch which keeps track of her heart rate, skin temperature and how many steps she's walked during the day. Alice's smart watch is connected to the Internet through cellular LTE as a node in the privacy-preserving IoT network proposed in this work. Alice subscribes to an online health monitoring service called Top Health that encourages her to take remedial action whenever she has not taken a sufficient number of steps in a day, or is about to come down with an illness. The smart watch sends the data mentioned above to the service for use in its diagnoses of Alice's health (see Fig. 11).

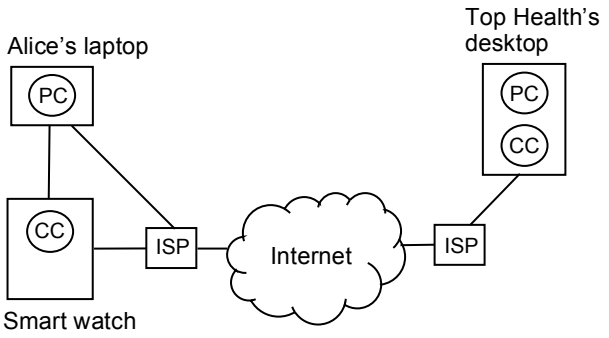


Figure 11. System configuration for smart watch example; the connections are as follows: Bluetooth between the laptop and the smart watch, Wi-Fi between the laptop and the ISP, and cellular LTE between the smart watch and the ISP (provider of both Internet and cellular services).

Alice performs the following steps when she first subscribes to Top Health:

1) Alice accesses her laptop (after entering her password), gets on the Internet, and launches her PC. Using network software that was packaged with her PC, she requests to see all online health monitoring services located in her province. Alice receives a listing of online health monitoring services located in her province. (Note: The details of service lookup and online messaging are assumed to go on in the background).

2) Alice uses her PC to retrieve her pre-specified privacy policy from her laptop's local storage (PS) and completes it with the name of the Top Health health monitoring service, based on her comfort level with Top Health after reading the reviews and recommendations from satisfied customers.

3) Alice's PC requests the privacy policy of Top Health after mutual authentication with it. With the arrival of Top Health's policy, Alice's PC compares Alice's policy with Top Health's policy to see if the policies match up. The policies match. Alice's PC sends her sender policy to the PC of Top Health, which does a quick verification of the policy match for added safety. If a non-match is found here (unlikely since already

checked by Alice's PC), Top Health's PC could terminate the interaction with Alice. Alice's PC sends the sender policy to the CC of the smart watch.

4) The CC in Alice's smart watch sets up a connection between Alice's smart watch and Top Health with the cooperation of Top Health's CC. Alice's smart watch then starts sending data to Top Health.

Alice's smart watch continuously sends the personal health information mentioned above to Top Health, which provides Alice with encouragements to take remedial action when needed. In addition, during the data transmission, the CM module of Top Health's CC, continuously checks Top Health's handling of Alice's personal information to ensure compliance with Alice's sender privacy policy. The CM module logs all private data activities to secure logs and sends them to Alice's CC when requested. Alice's CC verifies these secure logs for policy compliance and notifies Alice upon detection of any discrepancy, so that Alice can challenge Top Health's handling of her data when warranted. Of course, Alice would need to start a new data sending session with Top Health whenever the connection between Alice's smart watch and Top Health is unexpectedly broken, e.g., the smart watch runs out of power. Setting up a new session would simply require a repeat of step 4 above, assuming that the respective privacy policies have not changed since the previous data sharing session, and it is most likely that they have not, since the loss of connection was unintended. If either Alice or Top Health decides to change her/its privacy policy so that the policies no longer match up, then either party would notify the other party and Alice's use of Top Health's service would be ended. Alice may then use her PC to negotiate with Top Health in order to arrive once again at a policy match up. If this negotiation is unsuccessful, Alice may start again at Step 1 in order to choose a different health monitoring service.

V. Strengths and Weaknesses

Some strengths of the proposed framework are: a) upholds personally specified privacy preferences, b) can theoretically be used for all smart devices and all types of receivers or e-services, c) highly scalable due to the use of PCs and CCs (explained below), and d) easy to retrofit an existing non-privacy preserving IoT into a privacy preserving one. One weakness may be that the CM is not trusted to enforce privacy policy compliance. These points are elaborated below.

In terms of the strengths, the proposed framework allows each user to specify her privacy preferences in a privacy policy and for this policy to be upheld. Further, disagreements in privacy policies may be negotiated. Next, the framework allows a privacy preserving "session" to be set-up in which a data sender sends data to a data receiver. It leaves open what computing can be done in the session. Therefore, the session can be an e-commerce session where the data sender is a buyer and the data receiver is a seller, as in the above smart refrigerator example, or a health monitoring session where the data sender is a smart body worn sensor and the data receiver is a medical monitoring service as in the above smart watch example, or any other type of data transmission session that requires privacy protection. Another strength is the fact that the proposed framework is highly scalable. The privacy preserving IoT can be easily expanded by adding or linking PCs and CCs to devices that do not yet possess them (per Fig. 8), where if needed, multiple PC sessions may run in a desktop, laptop, or smartphone. Each additional IoT node so equipped may require a separate privacy policy exchange session. However, the increased cost per additional device is linear. The addition of PCs and CCs does increase network traffic, e.g., requests for the receiver's SL. However, the increased traffic can be accommodated by increasing network capacity, which is consistent with network growth and is not a limiting factor on scalability.

In terms of the weakness of trusting the CM, it must be made clear that malicious attacks on the CC and CM are always possible and could result in violation of privacy. One defense is to make it as hard as possible for those attacks to succeed, by protecting the CM. Ways to protect the CM and build trust for it have already been suggested above.

Reviewers have pointed out the following additional weaknesses: a) enforcement using SLs is not foolproof, i.e., the receiver can still leak personal information using channels not captured by SLs, b) people would need help in defining privacy policies, c) the framework may not apply to less powerful IoT devices, d) the CC may have performance issues in all that it is asked to do, and e) continuous checking of the vendor's handling of private information (Section IV above) could violate the vendor's privacy. These weaknesses are acknowledged, attenuated, or removed as follows. While enforcement using SLs is not foolproof, there is probably no method that is foolproof. As well, there would be tradeoffs to consider between using a more complex enforcement scheme, which is potentially more effective, and the complexity involved in the enforcement. For example, Mont and Thyne [13] (see Section VI) propose a potentially more effective enforcement scheme but which is more complex and thereby more error prone. Nevertheless, replacing SLs with a potentially more effective enforcement method is part of future work. People do need help defining privacy policies, usually through automation. Yee and Korba [12] address this issue (see Section VI) by proposing two semi-automated methods of privacy policy derivation. The framework can be applied to less powerful devices by implementing the PC and CC as software modules running on a desktop, laptop, or smartphone which is connected to the device, as mentioned above. In this scenario, the smart device merely has to be signaled to forward/receive its data

to/from the desktop, laptop, or smartphone running the CC, a change that should be implementable on even the least compute capable smart device. In terms of the CC potentially having performance issues, this is a possibility, especially if the smart device is not very powerful. This potential problem would be mitigated to some extent if the CC were to run on a desktop, laptop, or smartphone. In any case, this potential issue will be addressed through prototyping the CC, a part of future work. Finally, with regard to the possible violation of the vendor's privacy by the continuous checking of the vendor's handling of private information, note that this continuous checking is performed by the vendor's CC running on the vendor's platform for the benefit of the vendor so that the vendor can be assured that it is complying with the sender's privacy policy. Since there is no data associated with this checking that is forwarded back to the sender (only the SL is forwarded back to the sender – see Table III) there can be no violation of the vendor's privacy. It should also be noted here that the SL does not violate the vendor's privacy either, as it only refers to the sender's private information and how the receiver processed it in terms of the sender's privacy policy. In other words, the SL does not contain any vendor private information.

VI. Related Work

This work shares the notion of using controllers to monitor privacy policy compliance with an earlier work [9] in which we applied "privacy controllers" to protect privacy in web services. In this work, we have updated and re-designed the components in [9] to apply to the IoT.

Works that are related in terms of the application of personal privacy policies to implement privacy preferences are as follows. Yee [10] proposed a hybrid centralized / P2P architecture for ubiquitous computing that also protects privacy using privacy policies. Yee and Korba [11] examine privacy policy compliance for web services, and Yee and Korba [12] discuss privacy policy derivation. Another related work in this area is Mont and Thyne [13], which gives an approach for automatic privacy policy enforcement within an enterprise, by making data access control privacy-aware. Their approach incorporates a "Privacy Policy Decision Point" which makes decisions for allowing access based on privacy policies, and a "Data Enforcer" which intercepts attempts to access personal data and enforces the decisions made at the Privacy Policy Decision Point. Thus, their work is an example of enforcement other than checking a secure log as done in this work. However, their approach does not cover other privacy requirements such as purpose, retention time, and data disclosure. In terms of implementation languages for privacy policies, Kasem-Madani and Meier [14] overview 27 security and privacy policy languages and categorize them using a categorization framework. They also identify areas not covered by the languages. Kumaraguru et al. [15] summarize the literature available (at the time of their paper's publication) on privacy policy languages. They describe the features, characteristics, and requirements of the languages. They also provide a comprehensive framework for analysis and expect their work to aid the implementer in choosing a suitable language.

In the privacy literature for IoT, the following authors identify or analyze the security and privacy issues of the IoT. Loi et al. [16] develop a systematic method for identifying the security and privacy shortcomings of various IoT devices in order to alert consumers, manufacturers, and regulators to the associated risks. They apply their method to evaluate twenty market-ready consumer IoT devices and present their findings. Liu et al. [17] examine solutions for establishing smart devices in a smart home and demonstrate that such solutions may be error prone in terms of security and privacy. They argue that if the security and privacy issues are not considered, devices using a solution are inevitably vulnerable, seriously threatening the security and privacy of the smart home. Siby et al. [18] propose IoTScanner, a system that allows for passive, real-time identification and monitoring of an existing wireless infrastructure used for connecting IoT devices. Using this system, they identify privacy threats and investigate metrics for classifying devices. Kumar et al. [19] discuss the privacy and security concerns in IoT focusing on common IoT vulnerabilities such as distributed denial of service and data modification attacks. Their goal is to present the security and privacy concerns of IoT environments and the existing protection mechanisms.

In the privacy literature for IoT, the following authors propose partial or entire privacy protection solutions for the IoT. Kanuparthi et al. [20] describe privacy protection through the use of security measures such as encryption, which is the traditional approach to protection rather than checking for compliance to privacy policy as in our work. Alqassem [21] presents a privacy and security requirements framework for developing IoT, taking account of these requirements from the beginning of development, which is a software engineering approach distinct from our approach. Zhang et al. [22] consider privacy preservation in the application layer of the IoT and construct application scenarios to identify privacy preservation challenges. They look at privacy preservation in terms of maintaining confidentiality and examine various authentication schemes as means for providing confidentiality whereas we look at privacy preservation in terms of legislated privacy rights. Davies et al. [23] look at privacy protection for raw data that is streamed directly from IoT sensors to the cloud. They propose the use of a privacy mediator on every raw sensor stream. Each mediator is part of the same administrative domain as the sensors whose data is being streamed, and

dynamically enforces the privacy policies of the sensor owners. The use of privacy mediators in [23] is similar to the use of privacy controllers in our work. However, the application area is restricted to raw data streamed by sensors to the cloud and there is no mention of privacy policy matching or negotiation. Savola et al. [24] consider e-health applications in the IoT, such as biomedical sensor networks, as holding great promise but security and privacy are major concerns. They propose a high-level adaptive security management mechanism based on security metrics to cope with these concerns. Thus their approach is quite different than ours in their use of metrics to drive security management. Joy et al. [25] present a scheme to ensure granular location privacy for GPS enabled IoT devices. They accomplish this by designing and implementing a privacy module within the GPSD daemon, a low level GPS interface that runs on GPS enabled devices, thus giving data owners granular control over the release of their GPS location. Their proposal differs from ours in that they are only concerned with location privacy for GPS enabled devices, and of course, their method for protecting this privacy is different from ours. Pacheco et al. [26] study privacy preserving architectures for integrating the IoT with cloud computing. Their main concern for these architectures is to investigate the feasibility of implementing security and privacy mechanisms in IoT devices that are severely constrained in terms of computing resources. Their intention is to show that if such mechanisms can work in these constrained devices, they will work in almost all other devices. Thus, their approach is to safeguard privacy using traditional security and privacy mechanisms installed within the IoT devices (e.g., encryption), which is different than our approach of using privacy policies and verifying compliance to the policies. Appavoo et al. [27] address privacy-preserving access to sensor data for IoT based services such as health monitoring services. They observed that a large class of applications can function based on simple threshold detection, e.g., blood pressure above a pre-determined threshold. They propose a privacy-preserving approach based on this observation, their goal being to minimize privacy loss in the presence of untrusted service providers. The main algorithm in their proposed approach is an anonymization scheme that uses a combination of sensor aliases to hide the identity of the sensor data source, together with initialization vectors (or filters) to reveal information only to relevant service providers. Appavoo et al.'s work differs from this work in at least two ways. First, their work addresses a particular segment of services (monitoring services) whereas this work is applicable to all types of services. Second, they protect privacy through anonymizing the source of private information and restricting the private information to service providers that need to know. This work protects privacy through privacy policies and ensuring that the service provider complies with the policies.

VII. Conclusions and Future Work

This work has proposed a straightforward elaborated framework to protect privacy in the IoT, making use of compliance controllers together with sender and receiver privacy policies. In this framework, privacy is protected through compliance with sender privacy preferences, expressed as sender privacy policies. This work has greatly expanded the original CYBER 2016 paper by providing additional details for all sections.

Once privacy is protected, the smart devices in the IoT can engage in many applications, such as e-commerce (smart refrigerator using replenish food services) and e-health (smart body worn sensors using a health monitoring service).

The framework presented here is only theoretical. The effectiveness of the framework remains to be proven through prototyping and experimentation. However, much like a blueprint for a building, some security, performance and scalability aspects can already be predicted.

Future work includes the construction of a prototype to fine-tune the proposed framework, determine its effectiveness, and investigate some of the ideas discussed in the implementation notes, such as the use of reputation and other factors to help data senders decide which data receivers to select. We also plan to investigate other means of enforcing compliance with privacy policy that do not involve verifying SLs. Lastly, we plan to take a closer look at applying the framework to the transmission of private data from e-health smart devices in the wearable world (e.g., fitness trackers, smart watches, elders' call-for-help pendants).

References

[1] G. Yee, "An Approach for Protecting Privacy in the IoT," Proc. The First International Conference on Advances in Cyber-Technologies and Cyber-Systems (CYBER 2016), pp. 60-66, Oct. 9-13, 2016.

[2] I. Goldberg, D. Wagner, and E. Brewer, "Privacy-Enhancing Technologies for the Internet," Proc. IEEE COMPCON'97, pp. 103-109, Feb. 23-26, 1997.

[3] G. Yee and L. Korba, "Comparing and Matching Privacy Policies Using Community Consensus," Proc. 16th IRMA International Conference, in Managing Modern Organizations with Information Technology, edited by Mehdi Khosrow-Pour, pp. 208-211, 2005. Available Aug. 23, 2016: http://www.irma-international.org/proceeding-paper/comparing-matching-privacy-policies-using/32576/

[4] G. Yee and L. Korba, "The Negotiation of Privacy Policies in Distance Education," Proc. 14th IRMA International Conference, pp. 702-705, May 18-21, 2003. Available Aug. 23, 2016: http://www.irma-international.org/proceeding-paper/negotiation-privacy-policies-distance-education/32116/

[5] Office of the Privacy Commissioner of Canada, "PIPEDA fair information principles," available as of January 15, 2018 at:

https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-information-protection-and-electronic-documents-act-pipeda/p_principle/

[6] M. Gupta, P. Judge, and M. Ammar, "A Reputation System for Peer-to-Peer Networks," Proc. 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '03), pp. 144-152, June 2003.

[7] W3C, "A P3P Preference Exchange Language 1.0 (APPEL1.0)," available as of May 14, 2016 at: http://www.w3.org/TR/P3P-preferences/

[8] Trusted Computing Group, "Trusted Platform Module (TPM)," available as of May 14, 2016 at: http://www.trustedcomputinggroup.org/work-groups/trusted-platform-module/

[9] G. Yee, "A Privacy Controller Approach for Privacy Protection in Web Services," Proc. ACM Workshop on Secure Web Services (SWS '07), pp. 44-51, Oct. 29 – Nov. 2, 2007.

[10] G. Yee, "A Privacy-Preserving UBICOMP Architecture," Proc. Privacy, Security, Trust 2006 (PST 2006), pp. 224-232, 2006.

[11] G. Yee and L. Korba, "Privacy Policy Compliance for Web Services," Proc. 2004 IEEE International Conference on Web Services (ICWS 2004), pp. 158-165, July 6-9, 2004.

[12] G. Yee and L. Korba, "Semi-Automatic Derivation and Use of Personal Privacy Policies in E-Business," International Journal of E-Business Research, Vol. 1, Issue 1, pp. 54-69, 2005.

[13] M. C. Mont and R. Thyne, "A Systemic Approach to Automate Privacy Policy Enforcement in Enterprises," Proc. 6th Annual International Workshop on Privacy Enhancing Technologies (PET 2006), pp. 118-134, June 28-30, 2006.

[14] S. Kasem-Madani and M. Meier, "Security and Privacy Policy Languages: A Survey, Categorization and Gap Identification," in Cornell University Library, Dec. 2015. Available May 30, 2018: https://arxiv.org/abs/1512.00201

[15] P. Kumaraguru, L. F. Cranor, J. Lobo, and S. B. Calo, "A Survey of Privacy Policy Languages," Workshop on Usable IT Security Management (USM 07), in Proc. 3rd ACM Symposium on Usable Privacy and Security, pp. 1-4, July 2007. Available May 30, 2018: https://cups.cs.cmu.edu/soups/2007/workshop/Privacy_Policy_Languages.pdf

[16] F. Loi, A. Sivanathan, H. Gharakheili, A. Radford, and V. Sivaraman, "Systematically Evaluating Security and Privacy for Consumer IoT Devices," Proc. 2017 Workshop on Internet of Things Security and Privacy (IoTS&P '17), pp. 1-6, 2017.

[17] H. Liu, C. Li, X. Jin, J. Li, Y. Zhang, and D. Gu, "Smart Solution, Poor Protection: An Empirical Study of Security and Privacy Issues in Developing and Deploying Smart Home Devices," Proc. 2017 Workshop on Internet of Things Security and Privacy (IoTS&P '17), pp. 13-18, 2017.

[18] S. Siby, R. Maiti, and N. Tippenhauer, "IoTScanner: Detecting Privacy Threats in IoT Neighborhoods," Proc. 3rd ACM International Workshop on IoT Privacy, Trust, and Security (IoTPTS '17), pp. 23-30, 2017.

[19] N. Kumar, J. Madhuri, and M. ChanneGowda, "Review on Security and Privacy Concerns in Internet of Things," Proc. 2017 International Conference on IoT and Application (ICIOT), pp. 1-5, 2017.

[20] A. Kanuparthi, R. Karri, and S. Addepalli, "Hardware and Embedded Security in the Context of Internet of Things," Proc. 2013 ACM Workshop on Security, Privacy & Dependability for Cyber Vehicles (CyCAR'13), pp. 61-65, Nov. 4, 2013.

[21] I. Alqassem, "Privacy and Security Requirements Framework for the Internet of Things (IoT)," Proc. ICSE Companion'14, pp. 739-741, May 31-June 7, 2014.

[22] Z.-K. Zhang, M. Cho, and S. Shieh, "Emerging Security Threats and Countermeasures in IoT," Proc. 10th ACM Symposium on Information, Computer and Communications Security (Asia CCS '15), pp. 1-6, 2015.

[23] N. Davies, N. Taft, M. Satyanarayanan, S. Clinch, and B. Amos, "Privacy Mediators: Helping IoT Cross the Chasm," Proc. 17th International Workshop on Mobile Computing Systems and Applications (HotMobile '16), pp. 39-44, 2016.

[24] R. Savola, H. Abie, and M. Sihvonen, "Towards Metrics-Driven Adaptive Security Management in e-health IoT Applications," Proc. 7th International Conference on Body Area Networks (BodyNets '12), pp. 276-281, 2012.

[25] J. Joy, M. Le, and M. Gerla, "LocationSafe: Granular Location Privacy for IoT Devices," Proc. Eighth Wireless of the Students, by the Students, and for the Students Workshop (S3), pp. 39-41, 2016.

[26] L. Pacheco, E. Alchieri, and P. Barreto, "Enhancing and Evaluating an Architecture for Privacy in the Integration of Internet of Things and Cloud Computing," Proc. 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), pp. 1-8, 2017.

[27] P. Appavoo, M. C. Chan, A. Bhojan, and E.-C. Chang, "Efficient and Privacy-Preserving Access to Sensor Data for Internet of Things (IoT) Based Services," Proc. 8th International Conference on Communication Systems and Networks (COMSNETS), pp. 1-8, 2016.

Security and Privacy under a Unified Framework: A Review

Argyri Pattakou and Christos Kalloniatis

Privacy Engineering and Social
Informatics Laboratory,
Department of Cultural
Technology and Communication,
University of the Aegean
University Hill
GR 81100 Mytilene, Greece
Email: a.pattakou@aegean.gr, chkallon@aegean.gr

Stefanos Gritzalis

Information and Communication Systems
Security Laboratory,
Department of Information
and Communications Systems Engineering,
University of the Aegean
GR 83200 Samos
Greece
Email: sgritz@aegean.gr

***Abstract*—As the software industry experiences a rapid growth in developing information systems, many methodologies, technologies and tools are continuously developing in order to support the system implementation process. However, as security and privacy have been considered important aspects of an information system, many researchers presented methods that, through a number of specific steps, enable system designers to integrate security and privacy requirements at the early stage of system design. Different security and privacy engineering methods have been presented in order to be applied in traditional or cloud architectures. This paper reviews a number of security and privacy requirements engineering methods in both areas and presents a comparative study between these methods. Additionally, as at the recent years, security and privacy tend to be considered as two different but interdependent concepts, we present a conceptual model that considers both security and privacy under the same unified framework.**

Keywords–security; privacy; requirements engineering methods; traditional architecture; cloud computing; unified framework.

I. INTRODUCTION

For many decades, as the software industry has been constantly growing, the main interest of software engineers was to deliver new software releases rapidly, with no bugs and with the appropriate functionality. Under these circumstances, new tools, methodologies and technologies have been introduced in order to support system analysis and design, as well as software implementation. However, in the last decade, the software engineers community has realized that security and privacy are very important aspects in software engineering and, as a result, all the development software systems have to ensure security and privacy of the stored data [1-7].

As the interest of software engineers was mainly in developing new software, security and privacy was considered during implementation stage more as an ad-hoc process rather than an integrated process at the system design level. However, each late detection of possible security or privacy vulnerabilities has been proven to be extremely costly and time-consuming. Indeed, many researchers argue that security and privacy requirements have to be considered at the system analysis and design stage as security and privacy constraints might affect software functional requirements [8]. In this direction, we need mechanisms in order to elicit and analyse security and privacy requirements through a number of well-defined steps.

However, as the software industry was faced with a lack of integrated security and privacy requirements engineering methods, many researchers focused on introducing methods that support the elicitation of security and privacy requirements during the system design process. A requirement engineering method in the area of security and privacy can support engineers to define critical assets and the threats against them, to identify with accuracy security or/and privacy goals and to examine any kind of conflicts between them in order to come up with a clear and resistant set of security or/and privacy requirements.

Security and privacy requirements engineering methods have been built based on different approaches because, for each method, security and privacy requirements can be derived from different processes. For instance, some methods were introduced as goal-oriented methodologies as security and privacy goals might affect functional goals while other methods put as central issue potential risks and threats in order for security and privacy requirements to be derived. Different approaches can cover possible limitations or gaps among methods, as well as provide a variety of options to system analysts and designers in order to select the method that best fits the system into consideration.

During the last decade, literature has presented a number of security and privacy requirements engineering methods that support system designers and developers to implement secure and privacy-aware information systems hosted in traditional architectures. Some methods consider security or privacy requirements separately, but some other methods consider privacy as a subset of security. Recent literature efforts [7],[9-10] emphasize the need for parallel examination of security and privacy requirements under the same unified framework, as a possible security breach might affect users privacy and vice versa. However, few steps have been taken in this direction [11].

On the other hand, as cloud computing architecture introduces special characteristics, security and privacy requirements methods have to be developed in order to cover these special needs [12-14]. However, as the cloud computing area still suffers from a lack of integrated requirements engineering methods, methods that were initially introduced for traditional

architecture systems were extended in order to be applied in cloud systems as well [15]. But, at the moment, as far as we know, a method for cloud architecture that supports the parallel examination of security and privacy concepts has not been introduced.

Generally, literature presents a large number of papers that review security or privacy requirements engineering methodologies in traditional or cloud architectures. However, non of these reviews conduct a comparative study among them in order to support designers to select the most appropriate method for their system into consideration. Additionally, none of these studies justifies and analyses the need for a unified approach between security and privacy in the requirements engineering area.

In this paper, we present a number of security and privacy requirements methods that have been introduced in the last decades in order to support system design and analysis in traditional or cloud architectures. Also, we present a comparative study among methods that demonstrates the need for designing a framework that will consider security and privacy together under a holistic unified approach. At the end, a conceptual model for cloud-based systems that considers in parallel security and privacy requirements is presented.

Section II presents a set of security and privacy requirements engineering methods for traditional architectures and a comparative study among them. Section III is referring to security and privacy requirements engineering methods that can be applied in a cloud environment. Section IV presents the proposed conceptual model applied for cloud-based systems and Section V concludes the paper.

II. Security and Privacy Requirements Engineering Methods in Traditional Architectures

In this section, we present a number of security and privacy requirements engineering methods for traditional architecture that was introduced in the literature, in order to map the area and to conduct a comparative study between them.

A. Security and Privacy Requirements Engineering methods

1) Security Quality Requirements Engineering (SQUARE) Methodology: SQUARE methodology [16] was introduced because the software industry was missing an integrated model for eliciting and analyzing security requirements. The proposed methodology is a risk-driven method that supports the elicitation, categorization, prioritization and inspection of the security requirements through a number of specific steps. SQUARE also supports the performance of risk assessment in order to verify the tolerance of the system against possible threats. The final output of this method is a document that includes all the necessary security requirements that are essential in order for the security goals of the system to be satisfied. The methodology introduces the terms of security goal, threat and risk but does not take into consideration the assets and the vulnerabilities of the system. The application of SQUARE methodology requires the participation and the cooperation between stakeholders and the requirements engineering team in order to identify with accuracy all the necessary security requirements at the early stage of the development process. SQUARE does not refer to the elicitation of privacy requirements.

2) Model Oriented Security Requirements Engineering (MOSRE): As many research efforts conclude that considering non-functional requirements after system design can be proved very costly, Salini and Kanmani introduced a security requirements engineering framework (MOSRE) [17] for Web applications that considers security requirements at the early stages of the development process. The framework covers all phases of requirements engineering and suggests the specification of the security requirements alongside with the specification of system requirements. The authors suggest the identification of the objectives, stakeholders and assets of the Web application during the inception phase. The elicitation phase includes the identification of non-security goals and requirements in parallel with security goals, the identification-categorization-prioritization of threats and system vulnerabilities and a risk assessment process in order to elicit the final security requirements. Next phases include the analysis and modeling, the categorization-prioritization and the validation of the final security requirements. The framework does not support the elicitation of privacy requirements.

3) Security Requirements Engineering Framework (SREF): Haley et al. [18] introduced a problem based approach in order to elicit and analyze security requirements. The authors describe an iterative process of four steps. During these steps, security goals can be identified after the identification of functional (business) requirements. The identification of security goals includes the identification of system assets and a threat analysis. Risk assessment is also supported during the identification of security goals. However, in order to elicit security requirements from these security goals, the authors of Security Requirements Engineering Framework (SREF) take security requirements as constraints for functional requirements of the system under consideration and these constraints satisfy one or more security goals. The authors also encourage the use of problem diagrams to capture functional requirements with such constraints. The framework includes the notion of trust assumptions and the construction of satisfaction arguments by system analysts in order to validate security requirements. Privacy requirements are not considered by this framework.

4) Eliciting Security Requirements from the Business Process Models : Ahmed and Matulevicius introduced an asset based approach in order to elicit security goals from business process models and translate them into security requirements [19]. The method consists of two stages. At the first stage, an early analysis is performed in order to determine business assets that must be protected against security risks and security goals. At the second stage, the elicitation of security requirements is performed during examination of the security risk of business assets in five contextual areas: access control, communication channel, input interfaces, business services and data store. The final result is the elicitation of security requirements and the generation of business rules that satisfy security goals of the system under consideration. This framework does not support categorization, prioritization and validation of security requirements.

5) Security Requirements Engineering process (SREP): Mellado et al. presented SREP method [20] in order to provide a unified framework that considers concepts from requirements engineering and security engineering as well. Security Requirements Engineering Process (SREP) is an iterative and incremental security requirements engineering process and is aiming to integrate security requirements at the early stages

of software development life cycle [21]. SREP is an asset-based method, as well as a threat and risk driven method and it is based on the integration of Common Criteria [22] into the software life cycle in order to specify security requirements and validate that products meet security goals. The main idea of the proposed framework is that the unified process is divided into four phases: Inception, Elaboration, Construction and Transition. Each phase might include many iterations of nine activities (definitions, identification of assets, security objectives and threats, risk assessment, elicitation of security requirements, categorization-prioritization, inspection and repository improvement) but with different emphasis depending on what phase of the lifecycle the iteration is in [20]. Also, the authors propose the use of Security Resources Repositories to store sets of requirements that can be reused in different domains. Privacy requirements have not been considered by the authors.

6) Secure Tropos: Tropos methodology [23] was introduced by Castro et al. in order to cover system requirements during the whole software development process. However, Tropos methodology gives a strong focus on the early stage of system analysis. The framework includes five development phases: early requirements, late requirements, architectural design, detailed design and implementation. However, security concepts have not been considered in any of theses phases. Thus, Mouratidis et al. extended Tropos methodology in order to accommodate security concepts during the requirements analysis. The extension is called Secure Tropos [24] and utilizes only the early and late requirements phases of Tropos framework. Secure Tropos introduces the concept of security constraints. According to the authors, security constraints are a set of conditions, rules and restrictions that are imposed on a system and the system must operate in such way that none of them will be violated [24]. In the early requirements phase, a security diagram is constructed in order to represent the connection between security features, threats and mechanisms that help the satisfaction of security goals. The security diagram is taken into consideration at the late requirements phase in order for the designers to impose security constraints to the system-to-be. The enforcement of security constraints in different parts of the system can facilitate the disclosure of possible conflicts between requirements.

7) KAOS: In 2000, KAOS [25] was first introduced as a goal-oriented requirements engineering method in order to elaborate requirements from high level goals. According to the authors, the fulfillment of goals might be blocked by some exceptional agent behaviors that are called obstacles. In KAOS method, these obstacles have to be identified and resolved, through the elaboration of scenarios between software and agents, in order to produce a reliable system [26]. However, due to the fact that KAOS methodology considers only functional requirements, authors extended KAOS [27] in order to elaborate security requirements as well. The main idea of the extended framework is to build two models. A model of the system-to-be, that will describe the software and the relations between goals, agents, objects, operations and requirements and an anti-model that will capture possible attackers, their goals and system vulnerabilities in order to elicit all possible threats and security requirements to prevent such treats. Security requirements that derived by the anti-model as countermeasures have to be integrated in the original model.

8) PresSure: In 2014, Fabender et al. introduced a problem-based methodology, which is called presSure [28-29] in order to identify security needs during requirements analysis of software systems. The identification of security requirements is based on functional requirements of a system-to-be and on the early identification of possible threats. The methodology supports the modeling of functional requirements through problem diagrams. At next stage and after identifying the critical assets of the system and the rights of the authorized entities, possible attackers and their abilities have to be determined. Based on that information, a set of graphs is generated in order to visualize flows of possible threats related to the attackers access to critical assets. Security requirements derived from the analysis of these graphs. For each identified asset, every functional requirement is related with possible threats and security requirements.

9) LINDDUN: LINDDUN [30] was first introduced in 2010 by Deng et al. as a privacy threat analysis framework in order to support the elicitation and fulfillment of privacy requirements in software-based systems. According to the LINDDUN methodology, after designing a data flow diagram (DFD) of the system, privacy threats are related to the listed elements of the DFD. Threats in LINDDUN are categorized in seven types: Linkability, Identifiability, Non-repudiation, Detectability, Information Disclosure, Content Unawareness, Policy and consent Non-compliance. The method uses privacy threats trees and misuse cases in order to collect the threat scenarios of the system. Trough these misuse cases, privacy requirements can be extracted. Also, LINDDUN supports the prioritization and validation of privacy threat through the process of risk-assessment, before eliciting the final privacy requirements and before selecting the appropriate privacy-enhancing technologies. The authors of LINDDUN also map privacy-enhancing technologies to each privacy requirement in order to support system designers to select the appropriate techniques that satisfy privacy requirements.

10) SQUARE for privacy: As privacy plays an important role in software engineering, the authors of SQUARE methodology [16] adapted their approach in order to support the elicitation of privacy requirements at the early stages of software development process [31]. The extended framework includes the same steps as the original SQUARE method in conjunction with the Privacy Requirements Elicitation Technique (PRET) [32], a technique that supports the elicitation and prioritization of privacy requirements. This technique uses a database of privacy requirements based on privacy laws and regulations. However, the authors note that the database needs to be updated as the laws change and conclude that a new integrated tool is needed in order to support the elicitation of security and privacy requirements in parallel.

11) PriS: PriS [33] has been introduced by Kalloniatis et al. as a goal-oriented approach in order to integrate privacy requirements into the system design process. The main idea of this methodology is that privacy requirements are considered as organizational goals and adopts the use of privacy-process patterns in order to describe the impact of privacy goals to the affected organizational processes, to model the privacy-related organizational processes and to support the selection of the system architecture that best satisfies these processes. Thus, the authors of PriS cover the gap between system design and implementation phase. According to PriS, the identification of privacy goals is based on eight privacy concepts namely

authentication, authorization, identification, data protection, anonymity, pseudonymity, unlinkability and unobservability.

12) Secure Tropos and PriS metamodel: According to the above methodologies, most of the approaches in requirements engineering tend to consider security and privacy separately or consider privacy as a subset of security. However, a number of research efforts [7], [9] support that security and privacy are two different concepts that have to be examined separately but under the same unified framework. Under these circumstances, Islam et al. [11] introduced a model-based process that considers security and privacy concepts in parallel at the early stage of system analysis. This process integrates two different engineering methods. Secure Tropos is used as the main method in order to identify and analyse security requirements of the system under consideration. However, as privacy concepts are not considered through this method, Secure Tropos is extended integrating the PriS solution. Thus, security and privacy requirements can be identified and analysed in order to meet the goals but also the appropriate architecture and implementation technique can be selected in order for privacy goals to be satisfied.

13) Goal-based requirements analysis method (GBRAM): Anton and Earp introduced the Goal-based requirements method (GBRAM) [38] in order to support system designers to design secure e-commerce systems via identifying system and enterprise goals and requirements. The ultimate goal of this approach is to ensure security and privacy requirements coverage during the early stage of system design level by supporting the specification of security and privacy requirements and policies and checking the compliance among them. Risk assessment is considered critical in order to ensure that security and privacy policies reflect the actual security and privacy system requirements. In this direction, GBRAM describes four activities in order to support the identification of security and privacy goals and their conversion into security requirements and security/privacy policies. These activities include: Goals identification, Goals elaboration, Goals refinement and goals operationalization. In the GBRAM, each goal is assessed for risks and potential impacts. During risk assessment a new goal or a sub-goal might be added or the existing goal can be adjusted in order to mitigate the risk. In the GBRAM, goals are, also, categorized in five classes: user, system, communication, knowledge and quality goals. Finally, an assess compliance activity is introduced in order to be ensured that system requirements that have been elicited are aligned with the enterprise's security and privacy policy.

14)Abuse frames: L. Lin et al. presented the Abuse Frames approach [39] in order to analyse and represent security threats and to derive security requirements at the early phases of system requirements level. The authors support that abuse frames can delimit security problems so that system analysts and designers can focus on the characteristics of problem domains in order to uncover more easily security vulnerabilities and threats and to select the most appropriate security measures. According to the authors, abuse frames provide an abstract model of threats imposed by a potential malicious user within a defined system boundary. The approach uses Jackson's Problem Frames [40] in order to define boundaries in the problem areas and to focus on early security threat analysis. Also, the authors introduce the meaning of anti-requirements that define a set of undesirable requirements imposed by malicious users in order to subvert the existing system requirements. Privacy requirements are not considered by this approach.

15) Misuse Cases: Use Cases [41] have been proven very helpful for the elicitation and documentation of functional requirements in system analysis phase. Use cases methodology uses UML diagrams and various templates for textual description, in order to capture the appropriate functional requirements. UML diagrams contain actors, relations and processes in order to capture functions related to user's needs. However, Use Cases focus mostly on the representation of what the system should do rather than on the representation of what the system should not do. Thus, as security requirements was not possible to be derived by this method, Use Cases methodology extended in order to capture the behavior that should be avoided by the system. This extended version of Use Cases is named ”Misuse Cases”.

Misuse Cases [42] are an inversion of Use Cases that uses mis-actors instead of actors in order to represent possible threats by misuse behaviors. However, due to the fact that analysts need to indicate functions that prevent or detect misuse, the authors of Misuse Cases suggest the representation of a Use Case and the relevant Misuse Case in the same diagram. As in Use Cases, the textual representation of Misuse Cases is also important. The authors of Misuse Cases encourage the use of templates for textual description of use cases, but with adaptions in some fields in order to fulfill the representation of misuse behaviors. Security requirements derive from the analysis of threats that come up from Misuse Cases. Privacy requirements have not been considered in Misuse Cases.

16) The RBAC method: He and Anton [43] presented an agent-oriented framework for modeling privacy requirements and user privacy preferences in the role engineering process. The RBAC framework maps with a systematic way roles and permissions, while considers privacy requirements as constraints on permissions and roles in order to define access control policies. The framework consists of a data model and a goal-driven role engineering process.

As the authors of the framework refer, a typical access control rule in an RBAC policy is expressed as: $<$u, r, p $>$. That means that a user u can only access an object (o), if he/she is assigned a role r, and if the role is assigned certain permission p, which is allowed to access the object (o). Thus, the authors of the RBAC method consider Roles (r), Permissions(p) and Objects (o) as the basic elements of an RBAC system. Roles, permissions and objects are called contexts. On the other hand, purposes, conditions and obligations are identified as privacy elements in an RBAC system. The authors consider privacy elements as attributes of contexts. The data model that is included in the proposed framework represents the way that these three privacy elements can be modeled in the RBAC system.

Additionally, the framework includes a goal-driven role engineering process in order to support the elicitation and modeling of the three privacy elements. This process is divided in two phases: Role-Permission Analysis (RPA) and Role-Permission Refinement (RPR). During RPA phase, business processes and business task are analyzing via goal and scenario-oriented requirements analysis techniques in order to identify the corresponding permissions, permission constraints and roles for each task. However, as the set of roles and permissions generated in this stage are probably ambiguous, they can be refined in the RPR phase according to organization

structure, policy statement e.t.c.

With this framework, the authors aim to bridge the gap between high-level privacy requirements and low-level-access control policies in the early stages of system analysis and design.

17) The M-N (Moffett - Nuseibeh) framework: As the authors noted a non satisfactory integration of security requirements into requirements engineering, they presented a framework for analyzing and eliciting security requirements from the early stage of the system analysis and design process [44]. The proposed framework combines concepts both from software requirements engineering (functional goals, functional requirements and constraints) and from security engineering (assets and threats) as well.

According to the proposed framework, the elicitation of security requirements takes place in two steps. Firstly, the authors propose the application of risk analysis and management techniques in order to identify the threats against the valuable assets and to decide the appropriate security measures. Next step includes the definition of high-level security goals. Security goals arise from the inversion of the threats identified in the previous step. Generally, security goals aim to protect the valuable assets by possible threats and are operationalized into security requirements. Security requirements are considered as a set of constraints in functional requirements.

18)The STRAP method: The STRAP [45] method is a goal-oriented approach that promotes the design of privacy-aware systems as, following this approach, system designers are able to analyze and elicit privacy requirements form the early stage of system design level. The method is based on a structured analysis of privacy vulnerabilities in design and on an iterative process for the adaption of preferences. Thus, in STRAP, the derived vulnerabilities are considered as privacy requirements and these vulnerabilities are presented as obstacles in the satisfaction of a system 's functional requirements.

The STRAP method includes four (4) steps:

1)Analysis: The system 's analysis step includes a goal-oriented analysis. During this analysis, different actors and their privacy expectations, privacy goals and sub-goals and all the major system components and their relevant limitations are identified. In parallel, STRAP uses a number of questions for each goal and sub-goal that has been identified earlier. The result of the questions leads to the identification of a privacy vulnerabilities set. Thereafter, the derived vulnerabilities are evaluated for possible duplicates and are categorized in order to proceed to the Refinement step.

2)Refinement: During this step, system designers have to check the existence set of vulnerabilities and to identify those that can be eliminated or mitigated in order to eliminate the provided set of vulnerabilities.

3)Evaluation: As several design scenarios are generated by different designers, the purpose of this step is the selection of this design that decreases mostly the risk by the relevant privacy vulnerabilities.

4)Iteration: Finally, in the iteration phase, all the previous steps are repeated in order the new detected vulnerabilities to be integrated in the system design. Thus, the goal-model is re-designed in order the relevant alterations to be included. When no alterations are needed, the iteration phase ends.

19)The NFR (Non-Functional Requirements) framework: The authors proposed a process-oriented approach in order to support system designers to design secure Information Systems via a systematic, integrated and automated process. Although the NFR framework [46] can be applied in every phase of a system development life cycle, the authors proposed the use of NFR framework at the early stage of system design level. The NFR framework considers the non-functional requirements, such as security, accuracy, performance and cost, as softgoals that have to be achieved by the development system. Softgoals are considered special types of goals that need to be clarified, disambiguated, prioritized, elaborated upon, etc.

The main idea of this framework is to identify security goals, to represent them in a goal graph structure, to examine any possible interactions between security goals and finally to assess the degree of a goal achievement. More specific, the NFR approach includes the incremental and interactive construction, elaboration and revision of a softgoal interdependency graph (SIG). The graph consists of nodes that represent the softgoals (security goals) as well as the interdependencies among softgoals. Also, the interdependencies represent the relation between general softgoals with more specific low level soft-goals. In parallel, the NFR approach includes an evaluation procedure that considers interdependencies in order to verify that the identified softgoals have been achieved. Finally, the derived set of softgoals is linked to the system functional requirements. Softgoals operate as constraints in the implementation of the system's functional requirements.

Specific NFR catalogues have been constructed for security requirements also considering privacy (as part of the confidentiality security requirement), which makes NFR a useful tool for defining privacy requirement and identifying possible design alternatives. Also, the authors of NFR approach provide an automated tool, namely RE-Tools toolkit, in order to assist designers to build their NFR models.

20)The i method:* The i* method [47] was introduced as an agent-oriented method, as it focuses on systems agents and their social interdependencies. The main interest is to map the organization's logic and context at the early stages of system design. In this direction, the i* method was first designed as a tool for modeling, analyzing and redesigning organization processes. However, recently, the method is used in order to model security and privacy requirements of a system into consideration.As with the NFR method, the i* method is based on the notion of softgoals. However, the i* method focuses on the individuals goals of the systems'actors and not on the overall system goals. The actors are considered interdependent as the achievement of their goals depends on other actors and their tasks.

Security analysis takes place by using several analysis techniques and is aiming at the construction of a domain model that will capture the involved actors and their dependencies. In particular, attacker analysis helps identify potential system abusers and their malicious intents (threats). Dependency vulnerability analysis helps detect vulnerabilities in terms of organizational relationships among stakeholders. Countermeasure analysis supports the dynamic decision-making process of addressing vulnerabilities and threats. The results of this accurate analysis can be used for further refinement of actor softgoals. The i* method includes an evaluation procedure in order system designers to decide whether the impact of threats and vulnerabilities has been eliminated to an acceptable level. Finally, depending on actor s tasks, a role-based access control analysis can be performed in order the appropriate actor's roles to be defined.

B. A Comparison of Security and Privacy Requirements Engineering Methods

Many different approaches in the area of security and privacy requirements engineering have been presented in previous section. In this section, we present a comparative study between these methodologies according to specific criteria. A first criterion that is used in our study is the "Requirements" criterion and it is referred to the requirements that each method expects to meet. Some methodologies deal exclusively with the elicitation of security or privacy requirements but some others with both of them in parallel. Additionally, as each method bases the elicitation of security/privacy requirements on different concepts (i.e. goals, risks, threats etc) and processes, the approach that follows each method in order security/privacy requirements to be derived is used as a criterion in this comparative study. Another critical issue is the system development life cycle level where each method can be applied. Apart from these criteria, another set of criteria is examined. More specific, it is examined if:

- the assets of the system that have to be secured are considered by the method,

- any possible threats are considered by the method,

- a risk assessment is performed during the execution of the method,

- a categorization/prioritization of the derived security/privacy requirements is performed during the methodology,

- a requirements inspection step is included in the method,

- the method identifies and resolves any possible conflicts between the derived security/privacy requirements.

Table I summarizes and compares the aforementioned methodologies. A table entry that is labeled with Y or N means that the relevant criterion is considered or not by the relevant method.

A first remark is that most methods consider explicitly security or privacy requirements in order to design secure systems. On the other hand, the extension of KAOS, NFR and GBRAM method consider privacy as a subset of security. However, as privacy has separate aspects than security and a security incident might have a serious impact in user's privacy and vice versa, security and privacy requirements have to be examined in parallel under the same framework in order to design secure systems [7],[9-10]. The meta-model presented by Islam et al. [11] is able to support security and privacy requirements as it combines concepts from Secure Tropos and PriS methodologies that deal with security and privacy issues separately. Also, the i* method can support the elicitation of security and privacy requirements as well.

It is worth noting that all the aforementioned methodologies can be applied at the early stage of system analysis and design as a late reconsideration of security and privacy requirements can be extremely costly and time-consuming. LINDDUN, PriS methodology and therefore Secure Tropos and PriS metamodel include steps in order to fill the gap between system design and implementation stage and to support developers to select the most appropriate implementation technique. On the other hand, NFR method can be applied at all system development stages.

Each methodology has been build by using a different approach. MOSRE, Secure Tropos, KAOS, PriS, the Secure Tropos and PriS meta-model, GBRAM, M-N framework, STRAP and NFR method have been introduced as goal-oriented methodologies as security and privacy requirements are considered as organizational goals that have to be satisfied by the system into consideration. On the other hand, SQUARE methodology and SQUARE extension for integrating privacy requirements have been based on risk analysis results. It is worth noting that even if SQUARE method supports the identification of system threats and the corresponding vulnerabilities, the assets of the system that have to be secured are not considered by the method. On the contrary, the proposed methods by Ahmed et al. [19], MOSRE, SREF, SREP, M-N framework and i* method support risk analysis on business assets in order to elicit security/privacy requirements. Additionally, as many methodologies have integrated steps in order to support threat identification, SREP, LINDDUN, Misuse Cases and i* method put threat analysis in the center of their attention in order to elicit security or privacy requirements. SREF and presSure have been introduced as problem-based methods as the analysis and the elicitation of security requirements comes from the analysis of problem diagrams. Additionally, the RBAC and i* method have been characterized as agent-oriented methods. Both of these methods does not examine the overall security/privacy goals of the organization but the agent goals according to the roles that have been assigned to each agent.

Regarding the categorization/prioritization criterion, it could be noticed that for many methods this step is a logical extension of a risk analysis process. A categorization and prioritization of security or privacy requirements is an important aspect of many approaches, as, during this process, system designers have to decide if the implementation cost of a requirement is comparable with the value of the secured asset. SQUARE, MOSRE, SREP, LINDDUN, PriS, GBRAM, Misuse Cases and STRAP method support categorization/prioritization of requirements. Additionally, most of the approaches, SQUARE, MOSRE, SREF, SREP, PriS, the Secure Tropos with PriS metamodel, GBRAM,RBAC, M-N framework, SRAP and NFR method include steps for requirements inspection. Finally, MOSRE, Secure Tropos, PriS, the Secure Tropos with PriS meta-model, GBRAM and the NFR method examine the existence of any conflicts between requirements and security or privacy goals.

Table II presents the security and privacy requirements that each method aspires to cover. Where $\sim$ is labeled, that means that the author of the method does not specify the requirements that takes into consideration.

III. Security and Privacy Requirements Engineering Methods in Cloud Computing Environment

As presented in previous section, all the above methodologies were designed to contribute to system analysis and design in traditional architecture environments. As it is shown in Table II, the concepts of confidentiality, integrity and availability are the most frequent requirements that a method designed for traditional architecture systems examines. However, as a cloud computing structure is a more demanding environment, a methodology that is aiming to help system analysts and

TABLE I. COMPARISON OF SECURITY AND PRIVACY ENGINEERING METHODS

Method	Requirements	Approach	Stage	Assets	Risk Assessment	Categorization/Prioritization	Threats	Req. Inspection	Conflicts Identification
SQUARE	Security	Risk driven	Early Design	N	Y	Y	Y	Y	N
MOSRE	Security	Goal oriented	Early Design	Y	Y	Y	Y	Y	Y
SREF	Security	Problem based	Early Design	Y	Y	N	Y	Y	N
N. Ahmed et al.	Security	Asset based	Early Design	Y	Y	N	N	N	N
SREP	Security	Threat based	Early Design	Y	Y	Y	Y	Y	N
Secure Tropos	Security	Goal oriented	Early/Late Design	Y	N	N	Y	N	Y
KAOS	Security	Goal oriented	Early Design	N	Y	N	Y	N	N
PresSure	Security	Problem based	Early Design	Y	N	N	Y	N	N
LINDDUN	Privacy	Threat driven	Early/Late Design	N	Y	Y	Y	N	N
SQUARE for privacy	Privacy	Risk driven	Early Design	N	Y	Y	Y	Y	N
PriS	Privacy	Goal oriented	Early/Late Design - Implementation	N	N	Y	N	Y	Y
Secure Tropos with PriS	Security/Privacy	Goal oriented	Early/Late Design - Implementation	Y	N	N	Y	Y	Y
GBRAM	Security (Privacy is a subset)	Goal-oriented	Early Design	N	Y	Y	N	Y	Y
Abuse Frames	Security	Problem based	Early Design	Y	N	N	Y	N	N
Misuse Cases	Security	Threat Driven - UML Based	Early Design	N	N	Y	Y	N	N
RBAC	Privacy	Role based/Agent oriented	Early Design	N	N	N	N	Y	N
M-N framework	Security	Goal oriented	Early Design	Y	Y	N	Y	Y	N
STRAP	Privacy	Goal oriented	Early Design	N	Y	Y	Y	Y	N
NFR	Security (Privacy is a subset)	Goal oriented	All system development stages	N	N	N	N	Y	Y
i*	Security/Privacy	Agent oriented	Early Design	Y	Y	N	Y	N	N

*Y=Yes, N=No

designers to design a secure and privacy oriented information system in a cloud structure has to examine more specific requirements like user 's isolation, data portability, Cloud Service Providers transparency etc. Nevertheless, most of the methods presented in Section II, with the exception of Secure Tropos methodology, do not consider cloud security and privacy requirements and therefore could not be used during designing cloud systems. On the other hand, Secure Tropos methodology was extended in order to model special cloud security requirements [15]. A brief description of security and privacy requirements that are unique in a cloud structure is presented in Section IV.

In the recent years, as cloud computing has rapidly grown, many research efforts have been presented that consider security and privacy into the development process. Almorsy et al. [12] introduced a Model-Driven Security Engineering at Runtime (MDSE@R) approach for multi-tenant cloud-based applications. MDSE@R supports different tenants and service providers security requirements at runtime instead of design time by externalizing security from the application. More specific, service providers may impose some security controls as mandatory but multi tenants can also add extra security requirements at runtime at their own instance of the application. Fernandez et al. [13] presented a method on how to build a cloud Security Reference Architecture (SRE). An SRE is an abstract architecture that describes functionality without implementation details and includes security mechanisms to the appropriate places in order to provide a degree of security. This approach includes threat identification and uses misuse patterns in order to describe how an attack can be performed. Through this process, it can be verified that security patterns have been selected correctly and have been placed properly in the cloud architecture. In 2015, Perez et al. [14] presented a data-centric authorization solution, namely SecRBAC, in order to secure data in the cloud. SecRBC is a rule-based approach that provides data managing authorization to CSP through roles and object hierarchies. The authorization model uses advanced cryptographic techniques in order to protect data from CSP misbehavior also. In 2016, Mouratidis et al. [15] extended Secure Tropos requirements engineering approach for traditional software systems in order to enable modeling of security requirements that are unique in cloud computing environment and to support the selection of the appropriate cloud deployment model as well as the cloud service provider that best satisfies security requirements of the system under consideration. In 2013, Tancock et al. [34] presented the architecture of a Privacy Impact Assessment (PIA) tool in order to identify and evaluate possible future security and privacy risks on data stored in a cloud infrastructure. The risk summary that derives from PIA tool takes into consideration aspects like who the cloud provider is, what is the trust rating and what security and privacy mechanisms are used. As threat modeling is an important aspect for developing secure systems, Cloud Privacy Threat Modeling (CPTM) methodology [35] was proposed in order to support the identification of possible attacks and to propose the corresponding countermeasures for a cloud system through a number of specific steps. However, CPTM was designed in order to support only EU data protection directives and as a result the methodology presented a number of weaknesses in threat identification. Thus, A. Gholami and E. Laure [36] extended CPTM methodology in order to be complied with various legal frameworks. As it is hard for an organization to choose the appropriate cloud deployment

TABLE II. SECURITY AND PRIVACY REQUIREMENTS PER METHOD

Method	*Requirements*
SQUARE	CIA
MOSRE	CIA, Authentication, Authorization, Auditing
SREF	CIA, Accountability
N. Ahmed et al.	CIA, Authentication, Authorization
SREP	∼
Secure Tropos	CIA, Access control, Non-repudiation, Authentication, Accountability
KAOS	CIA, Privacy, Authentication, Non-repudiation
PresSure	CIA
LINDDUN	Unlinkability, Anonymity, Pseudonimity, Plausible deniability, Undetectability, Unobservability, Confidentiality, Content awareness, Policy & consent compliance
SQUARE for privacy	∼
PriS	Identification, Authentication, Authorization, Data protection, Anonymity, Pseudonimity, Unlinkability, Unobservability
Secure Tropos with PriS	All SecureTropos and PriS requirements
GBRAM	∼
Abuse frames	∼
Misuse Cases	∼
RBAC	∼
M-N framework	CIA
STRAP	∼
NFR framework	CIA
i*	∼

**CIA=Confidentiality, Integrity, Availability

type (public, private, hybrid or community), K. Beckers et al. presented a method that can support requirements engineers to decide which cloud deployment model best fits the privacy requirements of the system under consideration [37]. This approach is based on a threat analysis in parallel with the privacy requirements that the system shall satisfy and some other facts and assumptions about the environment like the number of stakeholders on each deployment scenario and the domains that have to be outsourced into a cloud.

Despite the fact that all these contributions develop different kind of mechanisms or processes that consider security and privacy issues in the context of cloud computing, most of them present a number of limitations. Some of them are related to specific cloud service models. MDSE@R is referred to a Software as a Service service (SaaS) model while the method for building a Security Reference Architecture is referred to an Infrastructure as a Service (IaaS) service model. On the other hand, most of the proposed frameworks, methods or processes in the context of cloud computing deal exclusively with security or privacy issues or in some cases privacy is considered as a subset of security. For instance, MDSE@R, secRBAC and SecureTropos consider only security issues while the Privacy Assessment Impact Tool (PIA), CPMT and the method for selecting the appropriate cloud deployment model focus explicitly on privacy issues. In our previous work [10], we presented the reasons why security and privacy have to be considered as two different concepts but have to be examined under the same unified framework. Nevertheless, one of the most important issues is that most of the proposed frameworks that are based on the idea of cloud computing integrate security and privacy controls during implementation phase and not earlier in requirements phase. But, such practices might create late corrections in security and privacy requirements, which means additional cost and severe delays in project delivery.

As cloud computing is a new and continuously developing environment, many research efforts have been presented over the last decade that highlight the need of adopting security and privacy mechanisms from the early stage of development life cycle. Nevertheless, until today security and privacy in the context of cloud computing is still performed as an ad-hoc process rather than an integrated process in the development life cycle. As it is mentioned above, Mouratidis et al. [15] presented a requirements engineering method in order to model cloud security requirements at the design level but no privacy requirements have been considered. Under these circumstances, literature presents a lack of integrated methods that through a number of specific steps could be able to support the parallel elicitation and analysis of cloud security and privacy requirements from the early stage of system design. It is worth noting that a security and privacy requirements engineering method at the design level should include steps in order to fill the gap between analysis and implementation phase in order to support system developers to select the appropriate technologies that best satisfy security and privacy requirements.

IV. CLOUD SECURITY AND PRIVACY UNDER THE SAME UNIFIED FRAMEWORK

The specific review aims on identifying the main security and privacy concepts that are proposed from the respective literature in the area of security and privacy requirements engineering from the relevant methodologies. Since most of the requirements engineering methods proposed in the literature were applied for the design and modelling of information

systems in traditional environments, the paper aims to identify the existence of requirements engineering methods proposed explicitly for the modelling of Information Systems in cloud environments and the changes that these methods bring in terms of the set of concepts that need to be considered when designing cloud-based services. Thus, the proposed conceptual model is a beginning towards the direction of proposing a framework that will be used by software engineers for the design of Information Systems in traditional and cloud-based systems.

As it is mentioned above, a cloud infrastructure is a continuously developing and a very demanding architecture as many additional parameters have to be considered during designing and developing a cloud infrastructure. As a result, system designers and developers have to take into consideration all the special characteristics of a cloud infrastructure (on-demanding services, resources sharing, remote access etc) in order to provide a secure and privacy-aware environment. An accurate determination of security and privacy goals of the system into consideration can prove to be crucial for achieving this goal. In this section, we present a set of security and privacy requirements that have to be provided by a trustworthy cloud infrastructure.

A. Security Requirements:

- Integrity: The integrity of data refers to the preservation of data from a possible malicious, intentional or unintentional modification during storage or transmission.
- Confidentiality: The confidentiality concept is referred to the assurance that user 's data and information will not be disclosed to unauthorized persons. Data should remain confidential not only to other users but to Cloud Service providers and system administrators as well.
- Availability: As the idea of cloud computing is based on the idea of on-demand services, data availability is referred to the ability of a Cloud Service Provider to provide continuous service delivery. Users have to be able to access their stored data any time by any device.
- Non-repudiation: This property is aiming to ensure that user 's actions will not be repudiated later.
- Authentication: Authentication requirements is referred to the implementation of authentication mechanisms in order to prevent access to data from non-legitimate users.
- Authorization: Authorization follows authentication and is aiming to the accurate determination of the resources and services that an authenticated user can access.

B. Privacy Requirements:

- Data portability: A cloud infrastructure have to ensure that user 's data could be transferred anytime to another cloud service provider. This requires that data will follow a standard format during their storage in the cloud infrastructure.
- Interoperability: Interoperability is referred to the ability of a cloud service provider to cooperate and interoperate with different cloud systems.
- Anonymity: anonymity is defined as the ability of a customer to use cloud resources and services without being obliged to reveal his/her identity and without being tracked [48].
- Pseudonimity: The concept of pseudonymity is very closed to the concept of anonymity. The difference lies in the fact that with pseudonimity a user can access cloud resources and services without being obliged to reveal his identity but by acting under one or more pseudonyms.
- Unlinkability: In order a cloud service provider to provide privacy to customers must prevent linkage between data and the customer that processes the specific data. In parallel, the provider must protect the privacy of a communication between a sender and a recipient. Than means that a possible attacker, another user or cloud administrators should not be able to identify two entities that communicate.
- Undetectability: Cloud users should be able to access cloud resources and services without being detectable by potential attackers.
- Unobservability: The concept of unobservability in the cloud is aiming to keep cloud users not only undetectable but anonymous as well while interaction with cloud resources or other cloud users.
- Provenancability: The requirement of provenancability is referred to the need for a mechanism that collects data in a structure way in order to record the history of every piece of data that exists in a cloud infrastructure. However, as provenance data might reveal sensitive data, the cloud service provider should be able to keep them secure inside the cloud infrastructure.
- Transparency: In order users to trust a cloud vendor, they should be aware for the procedures and policies that the cloud vendor follows. As Gartner [49] supports, cloud providers have the obligation to provide customers with clear details about architectures, risk controls policies, data location, recovery mechanisms etc.
- Isolation: As a cloud infrastructure allows the sharing of resources between multi tenants, the cloud provider should guarantee a certain level of isolation in order to achieve the complete seal of user's data [48].
- Accountability: An accountable cloud service provider must provide to cloud users a full control on their data and to function with transparency about how their data are used. That includes the clear identification of data policies, the compliance with the identified policies, the ability of data recovery in case of violation and the monitoring of data. Auditing user 's data and maintaining log records are common practices in this direction.
- Intervenability: Any cloud user should have the right to intervene in data processing where he considers that the cloud provider violates the policies. The meaning of intervenability includes the rights to data access without limitations, rectification and erasure of data, objection to data processing when processing does not comply with rules as well as the right to withdraw consent [50].
- Traceability: Traceability is referred to the ability of a cloud vendor to register in log files every human activity during processing data in the cloud infrastructure.

Below, we present a conceptual model that considers cloud security and privacy concepts, in parallel, during the system design process. The proposed conceptual model has been based on PriS method that was first introduced as a privacy requirements engineering method [33] in systems with traditional architecture only. In our previous work [10], the PriS framework was extended in order to consider security and privacy concepts, in parallel, in a cloud environment too. Indeed, the conceptual model represents a modeling language of security and privacy organization goals and requirements.

Also, the proposed conceptual model could be the base for developing a new requirements engineering method in a cloud environment that will consider system 's security and privacy requirements under the same unified framework. Such a method could contribute to the effective identification of security and privacy goals and requirements as well as to the effective evaluation of cloud providers.

In Figure 1, the central concept of the extended conceptual model is ”goal”. Goals are referred to any intentional objectives that an organization needs to achieve. Goals in a cloud environment are generated by the issues raised by stakeholders. Thus, goals can be derived by anyone involved in the cloud infrastructure (Cloud Service Provider, cloud users, system designers and administrators, external CSP, etc). For instance, a CSP must operate and provide services within a specific legal framework and must protect user 's privacy and data from any malicious attack. All these restrictions generate issues that in turn can generate new goals. Also, a SWOT (strength, weakness, opportunity, threats) analysis in a cloud based system might generate new issues and goals as well. Thus, an accurate identification of these issues at the early stage of system design level can contribute to the accurate determination of system 's objectives.

Processes can realise goals. However, the achievement of a goal might presuppose the achievement of one or more goals. Thus the origin goal has to be broken down to simpler goals by system designers in order each process to be applied in the relevant sub-goal and not directly to the origin goal. Also, a sub-goal might be related to the achievement of more than one goal, thus forming a structure of goals/sub-goals and their relationships. During this process, new goals can be identified and some others can be rejected or replaced in the hierarchy of goals. In Figure 1, the satisfaction relationships between goals and sub-goals is illustrated with the AND/OR decomposition entity.

The proposed conceptual model includes the examination of possible relations between two or more different goals. In this direction, two influence types are introduced in order system analysts to identify the relation between them and to examine whether two different goals are conflicting or not. The first influence type is referred as a Support relationship where the achievement of one goal assists in the achievement of another. The second one is illustrated as a Conflict relationship where the achievement of one goal prevents the achievement of another. In case of a conflict relationship, the involved stakeholders have to negotiate in order to resolve these conflicts.

As it is shown in Figure 1, goals are classified into three types: Organizational goals, Privacy goals and Security goals. Organizational goals are referred in the main objectives that an organization needs to achieve through the system into consideration. On the other side, privacy and security goals are introduced due to the special privacy and security concepts of a cloud based system. Anonymity, pseudonimity, undetectability, unlinkability, portability, interoperability and data protection have been identified as privacy-related concepts. Data protection includes the concepts of isolation, provenancability, traceability, intervenability, accountability and transparency as these concepts aim at protecting system or user's data in a cloud infrastructure. Unobservability is referred to the coexistence of undetectability of assets and anonymity of users. On the other side, integrity, confidentiality, availability, non-repudiation and access control have been indicated as security concepts. Additionally, authentication and authorization have been included in access controls concept as both aim at defining user's access level to the cloud infrastructure. However, privacy and security goals may have an impact on organizational goals as the identification of privacy and security requirements during system design might trigger new organization goals or reject others. Kalloniatis et al. in [48] described in details all the aforementioned security and privacy concepts.

As goals are realized by processes, it is proposed that system designers and developers use patterns in order to build processes with specific properties. Process patterns are general process models that deal with a specific issue through specific steps. In the security and privacy area, process patterns can help system designers to map the effect of security/privacy requirements on system processes and facilitate developers to select the technology (IDS, Digital Signature, PET's, etc) that best supports security and privacy goals. Thus, a system designer/developer should be able to select from a repository of patterns those that best fit in the process into consideration. Depending on the goal that a process is aiming to implement, the related pattern has to be selected. A privacy process pattern can be selected in case the relevant process aims to realize a privacy goal or a security process pattern can be used in order a security goal to be achieved. In general, security and privacy process patterns can standardize security and privacy procedures in order to support system designers to generate the appropriate processes that best satisfy security and privacy goals. In parallel, when processes have been generated based on specific process patterns, developers can select easily the appropriate security or privacy technologies that satisfy the specific process and therefore security or privacy goals.

V. Conclusion and Future work

In this paper, we presented a set of security and privacy requirements engineering methods that have been introduced by several researchers. Our research has focused on two areas: on those methods that aim to support software engineers to design and develop information systems hosted in traditional architectures and on those methods that can be applied in cloud systems. Thus, a narrow analysis for the security and privacy requirements engineering methods was performed, in order to map the area of security and privacy requirements methodologies as well as to record the gaps in this area and to justify the need for a holistic approach in the field of security and privacy. However, we will be able to provide extended information in this area in our future work.

As already mentioned, different security and privacy requirements engineering methods have been introduced in the past as software engineers community agree that security and privacy is still an integral part of the information systems design process. Referring to traditional architectures, there are different approaches that each method has been based on. For instance, security or privacy requirements can be derived from the determination of security or privacy goals, from the results of a risk analysis or from problem diagrams. Additionally, as it is clear from the above analysis, most researchers deal with security or privacy issues separately, a fact that can cause possible conflicts and late reconsiderations in functional requirements.

On the other hand, cloud computing is a more demanding structure as it introduces special characteristics like multi-

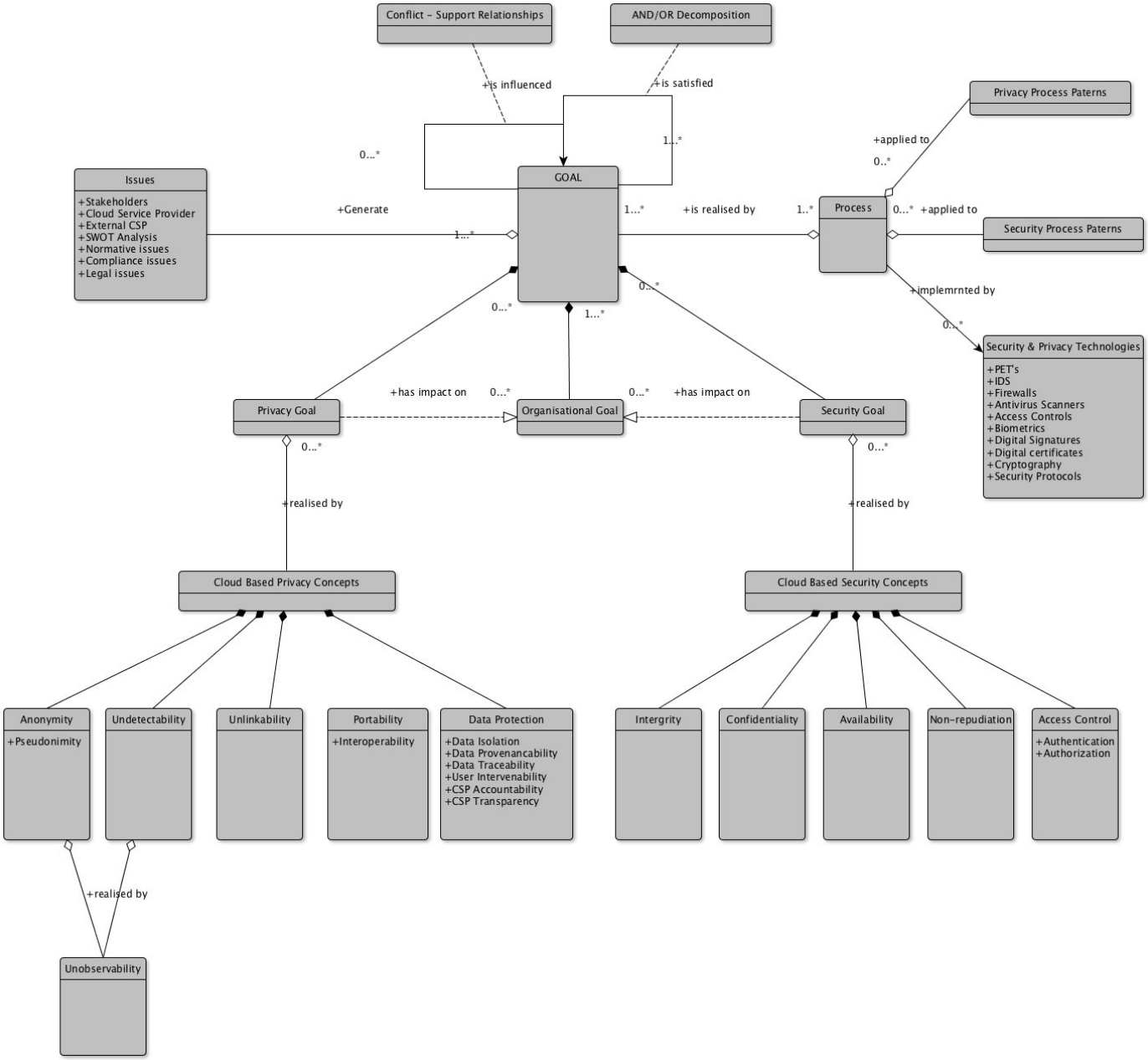


Figure 1. Conceptual model

tenancy and on-demand services. Special characteristics introduce new security and privacy concepts that software engineers have to take into account during system designing and developing. However, even though cloud computing presents a rapid growth last decade, all methods that have been presented by researchers present limitations while it is noting the lack of integrated methods that support the elicitation and analysis of security and privacy requirements in parallel.

The purpose of this research is to demonstrate that in cloud computing area there is a lack of integrated requirements engineering methods that consider security and privacy as two different concepts that have to be examined in parallel under the same unified framework. Thus, the aim of our analysis is to map and compare the existing methodologies in order to produce a unified framework that considers security and privacy concepts in parallel. This study constitutes the base for developing a new methodology in the cloud computing area that will consider security and privacy under the same unified framework.

REFERENCES

[1] A. Pattakou, C. Kalloniatis, and S. Gritzalis, "Security and Privacy Requirements Engineering Methods for Traditional and Cloud-Based Systems: A Review", CLOUD COMPUTING 2017 8th International Conference on Cloud Computing, GRIDs, and Virtualization, P. Dini, (ed), February 2017, Athens, Greece, IARIA

[2] I. M. Alharbi, S. Zyngier, and C. Hodkinson, "Privacy by design and customers perceived privacy and security concerns in the success of e-commerce," Journal of Enterprise Information Management, vol. 26, no. 6, 2013, pp. 702-718

[3] R. Cullen, "Culture, identity and information privacy in the age of digital government", Online Information Review, vol. 33, no. 3, 2009, pp. 405-421

[4] Z. Karake Shalhoub, "Trust, privacy, and security in electronic business: the case of the GCC countries", Information Management Computer Security, vol. 14, no. 3, 2006, pp. 270-283

[5] M. Meingast, T. Roosta, and S. Sastry, "Security and privacy issues with health care information technology", Engineering in Medicine and Biology Society, 2006. EMBS'06, 28th Annual International Conference of the IEEE, 2006, pp. 5453-5458

[6] S. E. Sarma, S. A. Weis, and D. W. Engels, "RFID systems and security and privacy implications", International Workshop on Cryptographic Hardware and Embedded Systems, Springer Berlin Heidelberg, 2002, pp. 454-469

[7] S. Gritzalis, "Enhancing Web privacy and anonymity in the digital era", Information Management and Computer Security, vol. 12, no. 3, 2004, Emerald Group Publishing Limited, pp. 255-288

[8] H. Mouratidis, P. Giorgini, G. Manson, and I. Philp, " A Natural Extension of Tropos Methodology for Modelling Security", Proceedings Agent Oriented Methodologies Workshop, Annual ACM Conference on Object Oriented Programming, Systems, Languages (OOPSLA), Seattle, USA, 2002

[9] C. Kalloniatis, E. Kavakli, and S. Gritzalis, "Addressing privacy requirements in system design: The PriS method", Requirements Engineering Journal, vol. 13, no. 3, 2008, pp. 241- 255

[10] A. Pattakou, C. Kalloniatis, and S. Gritzalis, "Reasoning About Security and Privacy in Cloud Computing under a Unified Meta-Model", In Proceedings of the Tenth International Symposium on Human Aspects of Information Security Assurance, HAISA 2016, pp. 56

[11] S. Islam, H. Mouratidis, C. Kalloniatis, A. Hudic, and L. Zechner, "Model based process to support security and privacy requirements engineering", International Journal of Secure Software Engineering (IJSSE), 2012, vol. 3, no. 3, pp. 1-22

[12] M. Almorsy, J. Grundy, and A. S. Ibrahim, "Adaptable, model-driven security engineering for SaaS cloud-based applications", Automated software engineering, vol. 21, no. 2, 2014, pp. 187-224

[13] E. B. Fernandez, R. Monge, and K. Hashizume, "Building a security reference architecture for cloud systems", Requirements Engineering, 2015, pp. 1-25

[14] J.M.M. Perez, G. M. Perez, and A. F. Gomez-Skarmeta, "SecRBAC: Secure data in the Clouds", IEEE Transactions on Services Computing, 2016

[15] H. Mouratidis, N. Argyropoulos, and S. Shei, "Security Requirements Engineering for Cloud Computing: The Secure Tropos Approach", Domain-Specific Conceptual Modeling, Springer International Publishing, 2016, pp. 357-380

[16] N. R. Mead and T. Stehney, "Security quality requirements engineering (SQUARE) methodology", ACM, 2005, vol. 30, no. 4, pp. 1-7

[17] P. Salini and S. Kanmani, "Model oriented security requirements engineering (MOSRE) framework for Web applications", Advances in Computing and Information Technology, Springer Berlin Heidelberg, 2013, pp. 341-353

[18] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis", IEEE Transactions on Software Engineering, 2008, vol. 34, no. 1, pp. 133-153

[19] N. Ahmed and R. Matulevicius, "A Method for Eliciting Security Requirements from the Business Process Models", In CAiSE (Forum/Doctoral Consortium), 2014, pp. 57-64

[20] D. Mellado, E. Fernandez-Medina, and M. Piattini, "A common criteria based security requirements engineering process for the development of secure information systems", Computer standards interfaces, vol. 29, no. 2, 2007, pp. 244-253

[21] B. Fabian, S. Grses, M. Heisel, T. Santen, and H. Schmidt, "A comparison of security requirements engineering methods", Requirements engineering, 2010, vol. 15, no. 1, pp. 7-40

[22] Infrastructure, Public Key, and Token Protection Profile, "Common criteria for information technology security evaluation." National Security Agency, 2002

[23] J. Castro, M. Kolp, and J. Mylopoulos, "Towards requirements-driven information systems engineering: the Tropos project", Information systems, vol. 27, no. 6, 2002, pp. 365-389

[24] H. Mouratidis, P. Giorgini, G. Manson, and I. Philp, "A Natural Extension of Tropos Methodology for Modelling Security", In Workshop on Agent-oriented methodologies, Annual ACM Conference on Object Oriented Programming, Systems, Languages (OOPSLA), 2002

[25] A. V. Lamsweerde and E. Letier, "Handling obstacles in goal-oriented requirements engineering", IEEE Transactions on Software Engineering, vol. 26, no. 10, 2000, pp. 978-1005

[26] S. Pachidi, "Goal-Oriented Requirements Engineering with KAOS", 2009, Method Description for "Method Engineering" course Master in Business Informatics, Utrecht University

[27] A. V. Lamsweerde, "Elaborating security requirements by construction of intentional anti-models", Proceedings of the 26th International Conference on Software Engineering, IEEE Computer Society, 2004

[28] S. Fassbender, M. Heisel, and R. Meis, "Functional requirements under security pressure", Software Paradigm Trends (ICSOFT-PT), 9th International Conference on IEEE, 2014

[29] S. Fabender, M. Heisel, and R. Meis, "Problem-Based Security Requirements Elicitation and Refinement with PresSuRE", International Conference on Software Technologies, Springer International Publishing, 2014, pp. 311-330

[30] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements", Requirements Engineering, 2011, vol. 16, no. 1, pp. 3-32

[31] A. Bijwe and N. R. Mead, "Adapting the square process for privacy requirements engineering", 2010

[32] S. Miyazaki, N. Mead, and J. Zhan, "Computer-aided privacy requirements elicitation technique", Asia-Pacific Services Computing Conference, APSCC'08, IEEE, 2008

[33] C. Kalloniatis, E. Kavakli, and S. Gritzalis, "Addressing privacy requirements in system design: the PriS method", Requirements Engineering, vol. 13, no. 3, 2008, pp. 241-255

[34] D. Tancock, S. Pearson, and A. Charlesworth, "A privacy impact

assessment tool for cloud computing", Privacy and security for Cloud computing, Springer London, 2013, pp. 73-123

[35] A. Gholami, A. S. Lind, J. Reichel, J.E. Litton, A. Edlund, and E. Laure, "Privacy threat modeling for emerging biobankclouds", Procedia Computer Science, 2014, vol. 37, pp. 489-496

[36] A. Gholami and E. Laure, "Advanced cloud privacy threat modeling", arXiv preprint arXiv:1601.01500, 2016

[37] K. Beckers, S. Fabender, S. Gritzalis, M. Heisel, C. Kalloniatis, and R. Meis, "Privacy-Aware Cloud Deployment Scenario Selection", In International Conference on Trust, Privacy and Security in Digital Business, 2014, September, pp. 94-105, Springer International Publishing

[38] A. I. Anton and J.B. Earp, "Strategies for developing policies and requirements for secure electronic commerce systems", E-commerce security and privacy, vol. 2, 2000, pp. 29-46

[39] L. Lin, B. Nuseibeh, D. Ince, M. Jackson, and J. Moffett, "Introducing abuse frames for analysing security requirements", In Requirements Engineering Conference Proceedings, 11th IEEE International, 2003, pp. 371-372

[40] M. Jackson, "Problem Frames: Analyzing and structuring software development problems", Addison-Wesley, 2001

[41] I. Jacobson, "Object-oriented software engineering: a use case driven approach", Pearson Education India, 1993

[42] G. Sindre and A. L.Opdahl, "Eliciting security requirements with misuse cases", Requirements engineering 10.1, 2005, pp. 34-44

[43] Q. He and A.I. Anton, "A framework for modeling privacy requirements in role engineering", In Proceedings of REFSQ, 2003, vol. 3, pp. 137-146

[44] J. D. Moffett and B. A. Nuseibeh, "A framework for security requirements engineering", REPORT-UNIVERSITY OF YORK, DEPARTMENT OF COMPUTER SCIENCE YCS, 2003

[45] C. Jensen, J. Tullio, C. Potts, and E.D. Mynatt, "STRAP: a structured analysis framework for privacy", Georgia Institute of Technology, 2005.

[46] L. Chung, B. Nixon, E. Yu., and J. Mylopoulos, " Non-Functional requirements in software engineering", Kluwer Academic Publishers, Massachusetts, USA, 2000

[47] L. Liu, E. Yu, and J. Mylopoulos, " Security and privacy requirements analysis within a social setting", In Proceedings of Requirements Engineering Conference, 2003, 11th IEEE International, IEEE, pp. 151-161

[48] C. Kalloniatis, H. Mouratidis, M. Vassilis, S. Islam, S. Gritzalis, and E. Kavakli, " Towards the design of secure and privacy-oriented information systems in the cloud: Identifying the major concepts", Computer Standards Interfaces, 2014, vol. 36, no. 4, pp. 759-775

[49] J. Brodkin, "Gartner: Seven cloud-computing security risks", Network World, 2008, pp. 1-3: https://www.networkworld.com/article/2281535/data-center/gartner–seven-cloud-computing-security-risks.html

[50] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data

Security Hardening of Automotive Networks Through the Implementation of Attribute-Based Plausibility Checks

Marcel Rumez, Jürgen Dürrwang, Johannes Braun and Reiner Kriesten

Institute of Energy Efficient Mobility
University of Applied Sciences Karlsruhe
Germany, International University Campus 3, 76646 Bruchsal
Email: {ruma0003, duju0001, brjo0002, krre0001}@hs-karlsruhe.de

***Abstract*—Future vehicles will be more and more part of the Internet of Things (IoT), providing enhanced functionalities such as autonomous driving, cloud-based functions or car-sharing features to their customers. However, this change has fundamental consequences for automotive networks and their safeguarding against unauthorized access. Based on our own research results regarding vulnerabilities in a Pyrotechnic Control Unit (PCU) and upcoming changes in automotive network architecture, we combined plausibility checks with an access control mechanism to restrict network requests in different vehicle states to prevent the exploitation of safety-critical functions. In this publication, we present our enhanced plausibility checks, which are based on vehicle attributes and trustworthy sensors. To do so, we propose moving the checks to powerful domain controllers in future automotive network architectures. Moreover, we adapt a vulnerability scoring metric from traditional Information Technology (IT) to determine the originality of the sensor values. As a result, we are hardening the security against unauthorized access.**

Keywords–Automotive Safety and Security; Vehicular Attacks; Plausibility Checks; Vehicle Networks

I. INTRODUCTION

Modern automobiles consist of more than 50 Electronic Control Units (ECUs), which contain a total of up to 100 million lines of code to control safety-critical functionality. This fact combined with the close interconnectivity of automotive ECUs and an increasing number of interfaces to the vehicle's surroundings, broadens the attack surface of modern vehicles. The feasibility of such attacks has been investigated and already demonstrated by several groups of researchers [2] [3]. Additionally, attacks via access to the internal vehicle network that can cause life-threatening injuries have also been demonstrated in the past [4] [5].

Furthermore, car manufacturers tend to equip their cars with more entertainment and comfort features using wireless connectivity. One example is the detection of traffic obstructions by using Car-2-X communication to process traffic or general environmental information provided by an ad-hoc network. In the same way, providers of car-sharing, car-rental and other fleet based services use cellular networks for the communication with their backbone [6]. Additionally, manufacturers implement the ability to execute software updates outside of car workshops, in order to fix problems within a short time [7]. These interfaces potentially provide means to remotely exploit vulnerabilities, obtain access to the in-vehicle network and control critical systems from a distance [8] [9].

Especially, with the remote exploitation of the Jeep Cherokee [8], Miller and Valasek showed that physical access through an On-Board Diagnostics (OBD)-Connector is not mandatory any more. One year after the remote exploitation of the Jeep they provided an update on what is possible in car hacking. Having already proven the remote exploitability of a vehicle, they used a direct connection to the internal car network via the OBD-connector. The fundamental approach was to stop an ECU, which is connected to the Controller Area Network (CAN), from broadcasting its own messages on the bus. This was done to enable them to send their own spoofed messages to another in-vehicular subscriber. As a result, they were able to execute different functions, e.g., deceleration of the vehicle or activating the parking assistant in an inappropriate driving condition. To prevent such misuses, ECUs typically use plausibility checks to validate the requested function with the state of the vehicle. For this purpose ECUs mostly use bus messages to derive the current state of the vehicle. Unfortunately, these messages are typically not protected from malicious modifications.

Our research has discovered a weakness in a safety critical component due to the fact that this component provides diagnostic functions for a special use case. The safety critical unit is a Pyrotechnic Control Unit (PCU), which offers the functionality to deploy attached airbags via vehicle diagnostics. This special use case scenario arises from the necessity of deploying airbags before a car can be crushed during its End of Life (EOL) recycling process. Unfortunately, these functions are available during the regular operation of the vehicle, potentially leading to life-threatening injuries. The discovered weakness is based on a requirement inside a standard [10], suggesting a weak algorithm to ensure authentication. Furthermore, no fundamental plausibility checks with available hard-wired sensors have been used, which we recommended in an earlier paper [1] and expand upon in this paper so that they can be used for future automotive architectures. Thus, we consider it as reasonable that this weakness scales over several manufacturers. This vulnerability has since been submitted to the Common Vulnerabilities and Exposures database and can now be accessed under its identifier CVE-2017-14937 [11]. To determine the existence of the vulnerability in vehicles a Metasploit Hardware Bridge module was created [12]. The module can check the availability of the functionality combined with the weak algorithm in a PCU.

To prevent such issues, authenticity and integrity of bus

messages have to be ensured and therefore cryptographic methods can be applied. The AUTomotive Open System ARchitecture (AUTOSAR) members have already recognized the necessity of the mentioned security goals for future on-board communication. For this reason, they have standardized the Secure Onboard Communication (SecOC) module [13], which includes authentication mechanisms on the level of Protocol Data Units (PDUs). The specification does not recommend a specific method for creating a Message Authentication Code (MAC), but rather defines the payload of a secured PDU with a freshness value and an authenticator for protecting against replay attacks and unauthorized manipulation of the message.

A typical approach for this is the application of a Keyed-Hash Message Authentication Code (HMAC) on salted messages. This type of cryptographic measure ensures the desired protection goals, with an acceptable need of computational performance, which is a fundamental constraint in the automotive domain. Nevertheless, there are existing drawbacks when using HMACs. In particular, the increasing bus load when attaching an HMAC on each message. Furthermore, it requires an extensive key management. According to the constraints in the automotive domain like restricted bandwidth and power, a trade-off between protection level and required resources is necessary. Unfortunately, this often leads to a non-implementation of necessary security measures. In this paper, we propose an approach of using local ECU signals, in addition to the information which the ECU receives from bus systems, to perform plausibility checks. In detail, the contributions of this paper are the following:

Problem: Spoofing and tampering of bus messages in vehicular networks can lead to safety critical situations. To prevent these threat scenarios, the message authenticity and integrity have to be ensured. However, channel protection alone is not sufficient if an ECU has been compromised. In this case, it is conceivable that an attacker would be able to transmit malicious payload with a valid message authenticity and integrity. Without additional checks, the receiver wouldn't be able to identify the tampered signal values of the message payload. **Solution:** Apply plausibility checks with trustworthy sensor signals as an additional security measure for cryptographic approaches to identify manipulations of received messages on ECU or domain controller level. Our **Contribution:** We present an enhanced network-based approach of attribute-based plausibility checks for future automotive networks based on our local approach for plausibility checks [1] and provide application examples to prevent two known attacks.

The paper is structured as follows: Section II summarizes the related work in the area of automotive security measures, followed by our approach in Section III, which is divided in methodology and its applicability. Furthermore, we propose a way to locate suitable signal sources inside vehicles that are necessary for our approach. This is followed by an application example that should be able to prevent the published exploitation of a passenger vehicle. In Section IV we give a short summary of our work and present an outlook for our future work in Section V.

II. Related Work

Automotive manufacturers, suppliers and other organizations have already recognized the necessity for security mechanisms in the automotive domain. For this reason, a cyber security alliance was founded in the USA. The major objective of the Automotive Information Sharing and Analysis Center (AUTO-ISAC) [14] is to enhance cyber security awareness and the coordination for the automotive domain. Moreover, the alliance is providing best practices for organizational and technical security issues to support the developing process of their members. An additional effort was initialized by the Society of Automotive Engineers (SAE) with the J3061 guidebook [15], summarizing recommended security practices that can be applied in the automotive domain. Unfortunately, the guidebook gives no concrete reference implementations for possible measures.

A more comprehensive approach for security in vehicles is presented by Gerlach et al. [16]. They propose a multi-layer security architecture for vehicular communication, which implements different measures. In particular, they propose digital signatures with certificates as methods for providing authentication, integrity, and non-repudiation of the received messages. Due to the underlying asymmetric cryptography, high-performance ECUs or ECUs with additional Hardware Security Modules (HSMs) are needed. They further consider an application of cross-layer plausibility checks [16] as meaningful. Therefore, they establish a single instance in the vehicle which collects information from any existent source in the vehicle. The instance is called plausibility checking module and creates its own independent view of the current vehicle state. If deviations from normal operation are detected, the instance reacts by triggering a warning. Unfortunately, the proposed instance is not implemented in each ECU, hence triggered counteractions or warnings have to be transferred over the unsecured bus again.

An additional approach is presented by Dhurandher et al. [17]. They propose an application of reputation and plausibility checks for Vehicular Ad Hoc Networks (VANETs). In particular, their proposed algorithm is able to detect and isolate malicious nodes by the use of sensors. Although they present an efficient and effective algorithm, the approach is designed for wireless nodes and their unique characteristics. Unfortunately, a concept for adaptation to in-vehicle networks is not given.

III. Approach

We consider an application of plausibility checks as additional protection mechanism as meaningful, if the relevant functions are able to change the physical state of the vehicle. This is partly explained by the fact that for these type of functions sensor values already exist. As a result, our approach is applicable for a great set of functions and in particular for safety-related functions. To decide if a function can be protected by our approach, some requirements have to be met. We define these requirements in the following and we further present an application example. Therefore, we divide our approach into two logical steps: First, it has to be determined if the selected function can be protected by a plausibility check (see Figure 1). This is followed by a method for implementing plausibility checks depending on the vehicle's network architecture. Finally, we give two application examples, which are explained in Section III-D).

A. Applicability of Plausibility Checks

To validate if plausibility checks are applicable, a few requirements have to be checked beforehand. For this purpose,

we define and highlight them as selection steps in Figure 1.

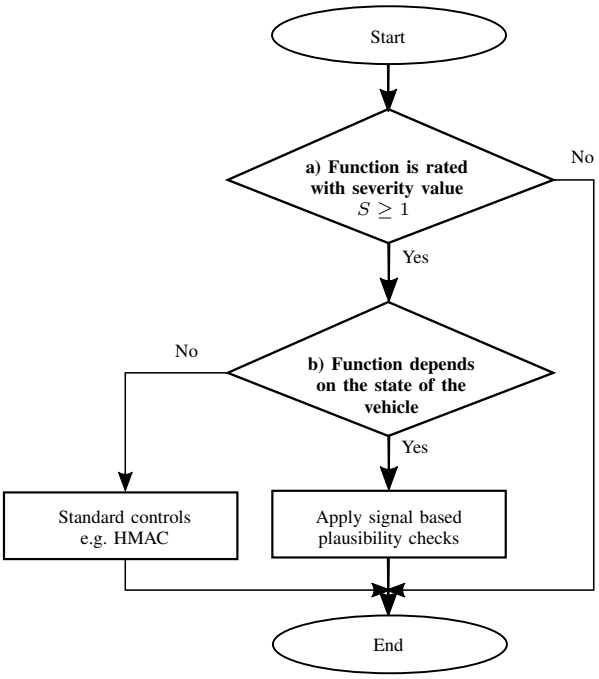


Figure 1. Methodology for applying signal based plausibility checks.

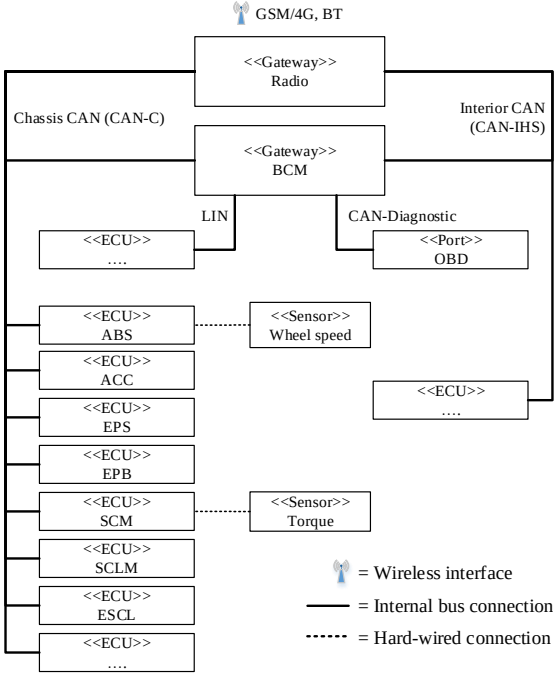


Figure 2. Part of the electrical architecture of a Jeep Cherokee 2014, based on the work of Valasek et al. [8]. As diagram notation we use the UML4PF profile extension [19].

Figure 1 shows the required steps to identify functions that are applicable for plausibility checks. Before we can validate *Step a)* a hazard and risk analysis must be performed. This is a demand of the functional safety standard ISO 26262 [18]. The aim of the analysis is to identify potential hazards of a function. Furthermore, a so-called Automotive Safety Integrity Level (ASIL) is calculated for each hazard based on three values. One of these values is defined as severity, describing the possible impact of the malfunction related to the selected function. Thus, we consider a selection of functions able to cause hazards with a severity value S greater or equal to 1 as meaningful. In particular, a severity value of $S \geq 1$ implies injuries of vehicle occupants [18] and must be prevented. If the function is rated with $S \geq 1$, the next step is to check, if the selected function has dependencies on the vehicle state (moving or standing still, etc.) as shown by *Step b)* in Figure 1. If plausibility checks are not applicable, but the function is rated with $S \geq 1$, we deem an application of standard security controls to be mandatory.

B. Plausibility Checks with Local ECU Signals

To guarantee that signals used for plausibility checks can not be maliciously modified or sent, we have to implement protection mechanisms. In particular, we have to ensure the authenticity and integrity of the used signals. Therefore, we could apply the already mentioned cryptographic methods with all their drawbacks, e.g., computing power, higher memory consumption, additional bus load, key management and testing of the implemented algorithms. Instead, we chose another way to check the originality of the signals indirectly without the afore mentioned drawbacks. To explain the approach, we take a closer look into automotive architectures like the one presented in Figure 2.

Figure 2 represents a part of the E/E architecture of a Jeep Cherokee 2014, which was the attack target of the researchers [8] [20] mentioned in the beginning. The architecture shows different ECUs and gateways interconnected by three CAN-Bus systems (CAN-C, CAN-IHS, CAN-Diagnostic) as well as one LIN-Bus. Furthermore, each wheel has a sensor measuring the wheel speed, which is hard-wired to the Antilock Braking System (ABS), respectively the Electronic Stability Control (ESC). This information can be used to derive local ECU signals for plausibility checks without the need for cryptographic algorithms. In particular, these sensor values can indirectly describe the state of the vehicle. With the wheel speed sensor shown in Figure 2, we can derive whether the vehicle is moving or not. If the vehicle is at a standstill, all sensor values of the wheels have to be zero or vary significantly due to a spinning wheel. This hard-wired sensor type is only an example. Additionally, we can combine two or more sensor values to derive more precise information about the state of the vehicle. The important point in our approach is that an ECU with hard-wired sensors can operate as a guardian against spoofed or tampered signals on the bus. In general, it is important that a safety critical function can be additionally protected by one or more hard-wired sensor values. By adding this requirement, an attacker would no longer be able to spoof sensor values over bus messages, because ECUs could verify the plausibility of the received values.

To be precise, authenticity and integrity are only ensured, if the attacker is not capable of getting access to the sensors themselves, which would require him to be in the vicinity of the vehicle. We assume that the possibility of an attacker accessing sensors is unlikely in comparison to his ability to send spoofed messages via CAN [8]. This is reasonable due to the fact that an attacker would have to overcome several

physical barriers, e.g., opening the hood, ECU housing or removing the wire insulation.

C. Plausibility Checks for Future Architectures

The next generation of E/E architectures (see Figure 3) in passenger vehicles will be modified in their structure. In the future, the ECUs will be divided in different domains like powertrain, chassis or driver assistance systems. This change provides more flexibility and scalability for the manufacturers. Moreover this opens up new ways for increasing the security level. The new domain controllers are powerful with regard to their clock-rate and memory, so that Original Equipment Manufacturers (OEMs) move computing-intensive applications from legacy ECUs to the enhanced domain controllers.

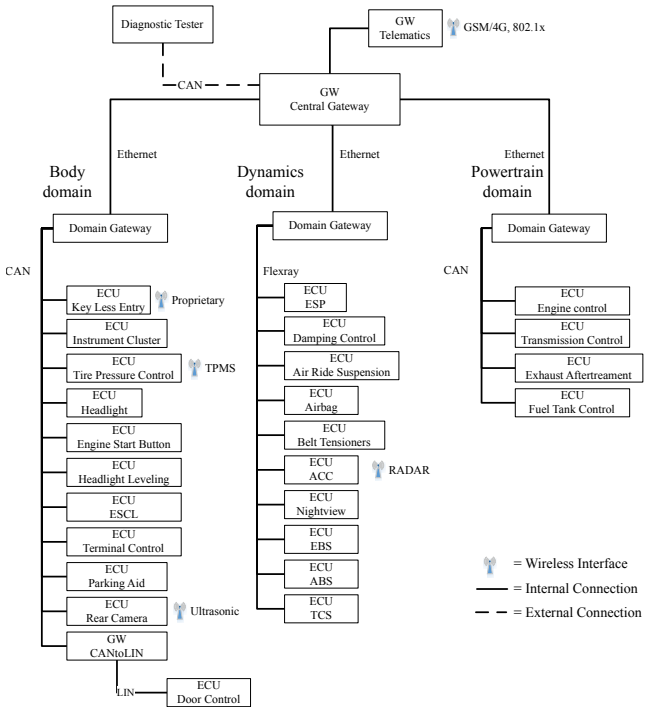


Figure 3. The next generation of E/E architectures (exemplary).

In this section we want to introduce our enhanced plausibility checks adapted to the new domain structure. Furthermore we have analyzed the latest Jeep hack [20] again and assigned the different attacks into two categories, whether the exploited function is based on diagnostic functionalities or not. The investigation has shown that five out of the seven performed attacks are based on the same approach, which sets the target ECU in Bootrom mode. This mode represents an extended diagnostic session for flash applications and requires authentication via the Security Access (SA) service. However, various research publications [21] have shown that implemented SA algorithms are often insecure. If an ECU is in Bootrom mode, it will exit the normal operation mode, e.g., stop sending CAN messages until the new firmware is successfully flashed. They exploited this functionality by setting the ECU in Bootrom mode without supplying a suitable firmware file. Therefore, the ECU is stuck in a loop and remains in this state even when the vehicle speeds up. The researcher used this attack technique for avoiding message confliction on the bus. Usually, additional injected messages typically lead to message confliction because the receiving ECU detects this anomaly and acts differently depending on the type of ECU.

Due to the fact that manufacturers move functions from single ECUs to the more powerful domain controller, we also follow this approach with our proposed plausibility checks. Furthermore, we are able to assess the trustworthiness of the request by comparing the current vehicles attributes to our security policy. In traditional IT exists a similar approach for authorization, which is called Attribute-Based-Access-Control (ABAC) [22] based on security policies in combination with different types of attributes (subject, object and environment). Generally in ABAC, the access control engine makes a decision to grant or deny any access request of a subject (entities that can perform actions on the system) to an object (function, file, variable, method) by interpreting the predefined security policy. Additionally, this model supports checks with boolean logic, e.g., "IF, THEN" statements and allows dynamic filtering due to the combination of subject attributes with environmental conditions (physical location, time, etc.).

We deem that ABAC can be adapted for an automotive Network Access Control (NAC) in combination with plausibility checks based on specific attributes, e.g., *message type, signals, device type, timing specifications*, because a lot of functions are only safety-critical when they are triggered during critical driving situations. Therefore, due to domain separation in the vehicle, we can use plausibility checks as an environmental attribute for granting access to network requests. As a result of domain controllers being connected to different networks by design, they are a prime candidate for implementing plausibility checks. For this reason, our approach is based on leveraging the most trustworthy sensor values to achieve a secure and reliable vehicle state information as an environmental attribute. To find suitable sensor values, we first have to identify all available sources, i.e., sensors in the vehicle. However, sensor sources vary wildly regarding the trustworthiness of their supplied signals.

Hence, we have decided to classify the sensor sources based on the CVSS, because this open industry standard is widely used for assessing the severity of information systems security vulnerabilities. The severity scores are based on criteria of three different metric groups [23]. We used the *Base Metrics* because their characteristics matched to the sensor sources the best. The base metric group is subdivided in *Exploitability metrics* and *Impact metrics*. For a better comprehension regarding to our sensor classification approach, we want to explain the different sub metric characteristics consecutively. The *Attack Vector (AV)* includes values how an attacker can exploit a vulnerability, e.g., is the attack target reachable via network or if physical access is needed. Furthermore, the *Attack Complexity (AC)* represents preconditions, e.g., the attacker must be man in the middle, which have to be fulfilled before an successful attack can be performed. The metric also includes the level of *Privileges Required (PR)* and whether a *User Interaction (UI)* besides an attacker is mandatory to exploit the vulnerability. The last item in this subgroup is the *Scope (S)*, which represents the ability of a vulnerability to impact other resources. The other subgroup includes the *Impact Metrics* whether an exploited vulnerability of the target component has an impact on the information assets, which are

TABLE I. Sensor Classification based on CVSS Base Metrics specification

Sensor	AV	AC	PR	UI	S	C	I	A	Score	Severity
Wheel speed	Physical	Low	High	None	Changed	None	High	High	6.5	medium
Acceleration	Adjacent	Low	High	None	Changed	None	High	High	8.1	high
Seat occupancy	Physical	Low	High	None	Changed	None	High	High	6.8	medium
ACC Radar	Network	Low	None	None	Changed	None	High	High	10	critical

specified with *Confidentiality Impact (C)*, *Integrity Impact (I)* and *Availability Impact (A)*.

We have applied the CVSS metric to some sensors of modern vehicles (see Table I) to identify the best suitable signal source for determining the current vehicle state. For our classification, we have set the scope of our investigation to determine the overall difficulty to manipulate the raw data of relevant sensors. However, as the CVSS originally comes from traditional IT some metrics have to be seen from another point of view for the automotive domain. That means, the metric PR is mapped to the required accessibility for tampering with raw sensor data from an attacker's point of view. The easiest way to manipulate raw sensor data is without any physical access or connections. In that case no specific privileges are required and the PR would be assigned with the value *None*. Furthermore, a higher degree of privileges would be, if an attacker needs any access to an physical interface, e.g., the OBD connector with partly standardized protocols, the associated PR value would be *Low*. The most difficult case from the point of view of the attacker is to manipulate raw data of in-vehicle sensors, because the protocols are mostly proprietary and the mounting position is often difficult to access. Hence, we are rating this case with the highest value (*High*). An example that some sensors can be easily manipulated from the outside has been shown in recent research with a camera mounted behind the windscreen of a car by displaying a specific graphic pattern in front of it [24]. However, to manipulate a sensor in the engine compartment an attacker would have to illegally unlock the car in order to unlock the hood latch.

To clarify the adaptation of the aforementioned metrics, the following section contains an example rating for the radar sensor of an Adaptive Cruise Control (ACC) system. We have assigned the value *Network* for the attack vector, because the sensor can be manipulated from the outside without any physical access. The attack complexity is low, because with no additional security measures, a manipulation of the sensor values can be performed very easily. Furthermore, no specific privileges or additional user interactions are required so that both metrics are rated with the value *None*. Due to the fact that an attack would have an impact on all distributed functions in the network, which are using these sensor values, we classified the scope with *Changed*. Looking now at the information assets, which are also included in the classification scheme, we deem that a successfully performed attack leads to a complete loss of the integrity as well as the availability. However, the confidentiality remains unaffected, because current in-vehicle communication is not encrypted.

The final score of the CVSS serves the purpose of comparing different signal sources that provide the same information. Furthermore, the metric provides a textual rating of the numerical score (see Table II). Based on this we have decided to set a rating limit for selecting a suitable sensor to a maximum value of 8.9 (High). For example, the score for the ACC radar sensor with the highest CVSS rating of 10.0 would be out of range to be used for sensor-based plausibility checks. Furthermore, it is recommended to select sensors with the lowest rating score, if more than one sensor for determining the same physical vehicle state is available.

In addition to finding suitable sensors for plausibility checks in domain controllers the CVSS supports another important feature in terms of rating trustworthiness in raw sensor data. A lot of research activities are focused on transmission security, e.g., protection of CAN messages. But what happens when an attacker manipulates the raw data of the corresponding sensors? Applying cryptographic measures afterwards, e.g., for the on-board communication are unable to detect this type of modified values discretely. At this time, the data is already manipulated. Before implementing complex protection mechanisms for network data, we should verify the raw sensor data first by applying plausibility checks in the first step of sensor data processing, e.g., through sensor fusion. To find which sensor should be additionally secured, the performed rating of the CVSS can be reused.

TABLE II. Qualitative Severity Rating Scale [23]

Rating	CVSS Score
None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

D. Application Example

1) Local-based Plausibility Checks: As an example, we want to discuss the latest Jeep hack [20], as well as the attack on the steering system which have been performed. Generally, the vulnerabilities in diagnostic mode, which the researchers used for disabling the Jeep's brakes among other things, are only working if the car is in reverse and slower than 5 mph. How can we make sure that the values received for plausibility checks are valid and not tampered with? We want to answer this question by the following examples, which explain how our approach would prevent these hacks in the future.

In the first example, the researchers set the real ECU in Bootrom Mode, causing it to stop sending messages on the bus. This step enabled them to send their own messages in the name of the jammed ECU. Electric Power Steering (EPS), which can be integrated in modern vehicles, e.g., the hacked Jeep series, requires various input parameters for calculating the electric steering support. One of these control values is

TABLE III. Extract of a Security Policy and corresponding Filtering rules for Domain Gateways

No.	Security Policy	Filtering rule
1	No driving operation, e.g., if an ECU is in Bootrom-Mode	Block all driving relevant requests
2	No extended diagnostic session while vehicle is moving	Block all diagnostic requests with SID 10 Sub-Function 02, if vehicle speed >= 6 mph
3	No diagnostic requests regarding End-of-life activation of pyrotechnic devices, while vehicle is moving	Block all diagnostic requests with SID 10 Sub-Function 04, if seat occupancy ! = 0 or vehicle speed >= 6 mph or seat buckle ! = 0

the velocity of the vehicle. Depending on the current speed and other parameters, the Steering Control Module (SCM) calculates the necessary steering torque. Basically, the steering torque support is decreasing by the SCM, when the velocity is increasing. Applied to the example of the Jeep hack, we want to show the determination of the steering torque threshold, which was one of the conditions the Jeep had to meet, in order to execute the steering angle change. A request for a high torque support in vehicle speeds of 30 mph or higher is not legitimate. However, we have to ensure that the integrity of the velocity value is given, for example by a hard-wired connection of the wheel speed sensors to the SCM. For instance, by implementing our approach, we deem that the execution of the function as done in the hack would have been refused during the plausibility check.

Another attack presented by Valasek and Miller [20] was the application of the car's brakes. The exploited function is normally used to activate the electronic parking brake for emergency braking by pressing the parking switch for a longer amount of time. Thereupon the pump for the ABS and ESC system gets activated and provides the necessary pressure to engage the brakes of the car. In this case, our approach is not applicable because of the missing hard-wired signals. In particular, an implemented plausibility check would not be possible, because of the lack of hard-wired signals. Therefore, it can not be differentiated between unintended or intended emergency braking, because we only have the information from the bus. In a case like this, where no hard-wired signal sources are available, we propose to check the feasibility of adding a hard-wired connection. The feasibility is given if the implementation effort of additional hard-wired connections is less than the implementation effort of a comparable cryptographic measure. Considering the mass-production of sensors in contrast to the effort of the selection, implementation, and testing of cryptographic measures, we consider additional hard-wired connections as less costly.

Our own attempts have shown that the related safety relevant ECU mentioned in the introduction has already connected hard-wired signals. However, the existent checks do not analyze the use-case correctly. Thus, it would have been possible to increase the security level simply by using enhanced software prompts, e.g., logical *and/or* conjunctions.

2) Network-based Plausibility Checks: Due to the change in future automotive architectures, we want to present an enhanced approach, how local plausibility checks can be adapted in future domain controllers to harden the security at the network level. In the redesigned methodology (see Figure 4), we have created several steps to achieve sensor-based plausibility checks that could be used in this new architecture. First we have to define insecure and prohibited vehicle states for different use cases and define a security policy based on those. The policy is generally written from the point of view of the object, which conditions have to be fulfilled for granting or denying access to a subject. Table III shows such an exemplary security policy, which includes several rules depending on different vehicle state attributes. After defining the policy, it is necessary to analyze the vehicle architecture to identify possible sensors for subsequent plausibility checks. The following step of sensor classification by applying the CVSS metric is mandatory to ensure the highest resilience against tampering of the sensor values. As we have already mentioned before, we recommend to select sensors based on the rating results. Moreover, it is recommended to select sensors of different domains to increase the trustworthiness even more. The next step should also be done carefully, because the transmitted sensor data within the network, e.g., via CAN, constitutes the trust anchor for the plausibility checks and therefore must not be manipulated during the transmission between sensor source and domain controllers. In detail, the authenticity and integrity of the selected sensor values must be ensured, e.g., by using the SecOC Module of the AUTOSAR standard. By securing only the specific information that is used in the plausibility checks later on, the approach tries to be as lightweight as possible.

After these steps, the preconditions for the sensor-based plausibility checks are complete. They can be used for specific message filtering in the domain controllers by deriving fine-grained filtering rules with boolean logic from the defined security policy in combination with the selected sensor attribute values. The rules should include more than one sensor value for determining a precise vehicle state. The best-case scenario would be the integration of the two of three principle referred to the sensor sources. However, it will not always be possible to find more than one sensor source for each defined vehicle state.

At this point, we want to depict an example, how these enhanced plausibility checks can be used to complicate specific attack techniques. In the mentioned Jeep hack, Miller and Valasek have often applied the same trick by setting a specific ECU in Bootrom Mode. After that, they were able to send their own spoofed CAN messages to alter the vehicle movement. By applying our approach this method would not have been possible, because the domain controllers are able to check the actual vehicle state via a state table, e.g., if an ECU is currently in a diagnostic session, before they route inter-domain messages. As a consequence, the domain controllers block all messages for triggering driving relevant functions. Furthermore, the domain controllers are also able to verify, if specific sensor values match to the current vehicle state. Besides blocking the relevant functions we suggest to record this event for a forensic process and we further suggest to place the vehicle in fail-safe mode if the requested function is rated with a severity value $\geq$ 2. This is reasonable due

to the fact that an attack could be started with the intent to injure the passengers. The fail-safe mode allows the driver to continue using the vehicle, but encourages him to visit the workshop. The workshop is then able to search for the source of the malicious request, e.g., attached OBD devices.

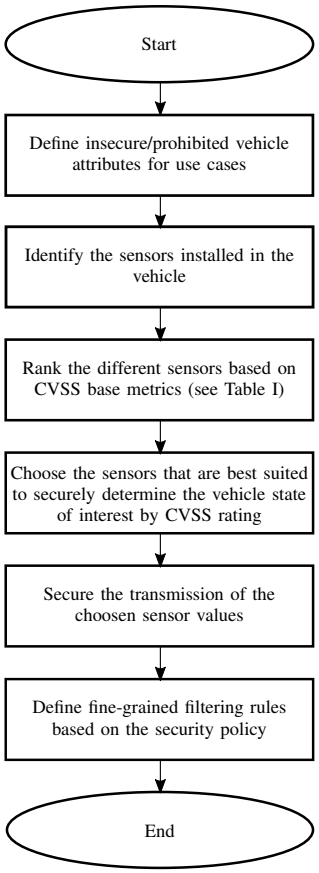


Figure 4. Process for deriving filtering rules based on trustworthy sensors and the defined security policy.

IV. Conclusion

In this publication, we proposed a new way to implement plausibility checks for automotive ECUs as well as a new approach to increase the security level in future architectures by enhanced network-based plausibility checks. Both approaches are capable to ensure that signals used for plausibility checks are resilient against replay and tampering. Based on the two approaches, we also want to divide the conclusion into two sections:

1) Local-based Plausibility Checks: The local approach uses already available information, like sensor signals, to verify function requests with the actual state of the vehicle. Due to the fact that our local approach uses no cryptography and existent information is reused, our approach tries to be as lightweight as possible to keep additional busload to a minimum. However, the approach is not suitable to secure all functions on ECUs, because at least one hard-wired sensor source should be available. Furthermore, we showed an example implementation of our plausibility approach, which is able to prevent a known attack. For this case we used hard-wired sensor signals like wheel speed sensors of the ABS to ensure the integrity of the velocity signal. The other example was focused on the electric power steering ECU.

2) Network-based Plausibility Checks: In our enhanced network-based approach for future architectures, we moved local plausibility checks to the more powerful domain controller for filtering inter-domain network traffic based on the actual vehicle state. In order to determine the actual vehicle state, we need trustworthy sensor information. For this reason, we presented a methodology to select suitable sensors by adapting the CVSS metric for an automotive severity assessment. Furthermore, we adapted a NAC approach from traditional IT, which allows dynamic and context-aware access control with security policies based on specific attributes, e.g., vehicle speed or diagnostic session. By analyzing the mentioned Jeep hack again, we examined and explained the Bootrom vulnerability, which enabled the involved researchers to exploit several functions. With the recommended state table and proposed enhanced plausibility approach, this kind of attack would not have been possible. Moreover, the vulnerability found during our own research activity, present in different ECUs that can lead to the detonation of pyrotechnic charges, would be blocked by checking the security policy in the domain controller. Due to the fact that in future architectures a variety of sensor signals will be protected by default, this leads to the assumption that the overhead for the network remains unchanged as well as that no additional wiring is required. We are currently working on evaluating the whole network traffic with respect to latencies and memory requirements to address this open gap.

After doing our own research we can confirm that replay attacks can be performed with minimal effort, if bus systems like CAN are used. In combination with our findings based on a safety critical function in a PCU, which is rated with a severity value of 3, we recommend that such functions should only be executable by bus messages as long as the plausibility of the request can be verified. Therefore, our approach recommends using at least two values received from different sources. In the best case scenario, one source is a hard-wired connection.

V. Future Work

The mentioned vulnerabilities show us the necessity of additional safeguards for upcoming vehicles. In the future the amount of interconnected services will continue to increase and the vehicle can be seen as a part of the IoT. That means current network design paradigms will also change from static signal-oriented approaches to service-oriented communication for achieving more flexibility regarding to software updates and upgrades during the whole vehicle life-cycle. This will also allow to swap out functions in the cloud for reducing the computing power requirements in vehicle ECUs or for providing more customer functionalities and creates new business cases for OEMs as well as third-party providers due to the introduction of vehicle Application Programming Interfaces (APIs). This creates new challenges for the whole automotive domain with focus on communication security. Due to this fact we are working on dynamic, distributed and scalable firewall techniques to address authorization regarding service-oriented architectures. In detail we want to enhance the presented sensor-based approach with focus on an ABAC automotive

policy framework and their evaluation in respect to timing and safety constraints.

ACKNOWLEDGMENT

This work has been developed in the projects SAFE ME ASAP (reference number: 03FH011IX5) and AUTO-SIMA (reference number: 13FH006IX6) which are partly funded by the German ministry of education and research (BMBF) within the research programme ICT 2020.

REFERENCES

[1] J. Dürrwang, M. Rumez, J. Braun, and R. Kriesten, "Security Hardening with Plausibility Checks for Automotive ECUs," in VEHICULAR 2017, 2017, vol. 6, pp. 38–41. [Online]. Available: http://www.thinkmind.org/download.php?articleid=vehicular_2017_2_40_30053

[2] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno et al., "Comprehensive experimental analyses of automotive attack surfaces," 2011.

[3] C. Miller and C. Valasek, "A survey of remote automotive attack surfaces," black hat USA, 2014, last checked on 09.05.2017. [Online]. Available: https://sm.asisonline.org/ASIS%20SM%20Documents/remote%20attack%20surfaces.pdf

[4] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham et al., "Experimental security analysis of a modern automobile," Symposium on Security and Privacy, 2010.

[5] C. Miller and C. Valasek, "Adventures in automotive networks and control units," DEF CON, vol. 21, 2013, pp. 260–264.

[6] J. C. Norte, "Hacking industrial vehicles from the internet," last checked on 12.02.2018. [Online]. Available: http://jcarlosnorte.com/security/2016/03/06/hacking-tachographs-from-the-internets.html

[7] M. S. Idrees, H. Schweppe, Y. Roudier, M. Wolf, D. Scheuermann, and O. Henninger, "Secure automotive on-board protocols: A case of over-the-air firmware updates," Nets4Cars/Nets4Trains 2011, 2011, pp. 224–238.

[8] C. Valasek and C. Miller, "Remote exploitation of an unaltered passenger vehicle," last checked on 09.05.2017. [Online]. Available: http://illmatics.com/Remote%20Car%20Hacking.pdf

[9] A. Greenberg, "Tesla Responds to Chinese Hack With a Major Security Upgrade," last checked on 23.03.2017. [Online]. Available: https://www.wired.com/2016/09/tesla-responds-chinese-hack-major-security-upgrade/

[10] ISO, "ISO 26021 Road vehicles – End-of-life activation of on-board pyrotechnic devices," 2009.

[11] "CVE-2017-14937," last checked on 10.01.2018. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-14937

[12] Rapid7, "Tesla Responds to Chinese Hack With a Major Security Upgrade," last checked on 07.02.2018. [Online]. Available: https://www.rapid7.com/db/modules/post/hardware/automotive/pdt

[13] AUTOSAR, "AUTOSAR 4.3.0 – Specification of Module Secure Onboard Communication," 2016.

[14] AUTO-ISAC, "Automotive information sharing and analysis center," https://www.automotiveisac.com/index.php, last checked on 09.05.2017.

[15] SAE, "Cybersecurity Guidebook for Cyber-Physical Vehicle Systems," 01.2016, last checked on 12.04.2016. [Online]. Available: http://standards.sae.org/wip/j3061/

[16] M. Gerlach, A. Festag, T. Leinmuller, G. Goldacker, and C. Harsch, "Security architecture for vehicular communication," in Workshop on Intelligent Transportation, 2007.

[17] S. K. Dhurandher, M. S. Obaidat, A. Jaiswal, A. Tiwari, and A. Tyagi, "Vehicular Security Through Reputation and Plausibility Checks," Systems Journal, IEEE, vol. 8, no. 2, 2014, pp. 384–394.

[18] ISO, "ISO 26262 – Road Vehicles – Functional Safety," 2011.

[19] D. Hatebur and M. Heisel, "A uml profile for requirements analysis of dependable software," in International Conference on Computer Safety, Reliability, and Security. Springer, 2010, pp. 317–331.

[20] C. Valasek and C. Miller, "CAN Message Injection: OG Dynamite Edition," last checked on 05.04.2017. [Online]. Available: http://illmatics.com/can%20message%20injection.pdf

[21] M. Ring, T. Rensen, and R. Kriesten, "Evaluation of Vehicle Diagnostics Security: Implementation of a Reproducible Security Access," Secureware, vol. 2014, 2014.

[22] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to Attribute Based Access Control – Definition and Considerations," 2014, last checked on 12.01.2018. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.sp.800-162.pdf

[23] CVSS Special Interest Group, "Common Vulnerability Scoring Group v3.0 - Specification Document," last checked on 08.01.2018. [Online]. Available: https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf

[24] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, "Deceiving Autonomous Cars with Toxic Signs," last checked on 21.02.2018. [Online]. Available: https://arxiv.org/pdf/1802.06430.pdf

Empirical Case Studies of the Root Cause Analysis Method in Information Security

Niclas Hellesen, Henrik Miguel Nacarino Torres, and Gaute Wangen
Norwegian University of Science and Technology
Gjøvik, Norway
Email: niclash@stud.ntnu.no, henrik.torres@gmail.com, gaute.wangen@ntnu.no

***Abstract*— Root cause analysis is a methodology that comes from the quality assurance and improvement fields. Root-cause analysis is a seven-step methodology that proposes multiple tools per step, which are designed to identify and eliminate the root cause of a reoccurring problem. Lately, the method has been adapted into the information security field, yet there is little empirical data regarding the efficiency of the Root cause analysis approach for solving information security management problems. This paper presents three empirical case studies of root cause analysis conducted under different premises to address this problem. Each case study is qualitatively evaluated with cost-benefit analysis. The primary case study is a comparison of information security risk assessment and root cause analysis results from an analysis of a complex issue regarding access control violations. The study finds that in comparison to the risk assessment, the benefits of the Root cause analysis tools are a better understanding of the social aspects of the risk, especially with regards to social and administrative causes for the problem. Furthermore, we found that the risk assessment and root cause analysis could complement each other in administrative and technical issues. The second case study tests root cause analysis as a tabletop tool by modeling an information security incident primarily through available technical documentation. The findings show that root cause analysis works with tabletop exercises for practice and learning, but we did not succeed in extracting any new knowledge under the restrictions of a tabletop exercise. In the third case study, the root cause analysis methodology was applied in a resource constrained setting to determine the root causes of a denial of service incident at small security awareness organization. In this case, the process revealed multiple previously undetected causes and had utility, especially for revealing socio-technical problems. As future work, we propose to develop a leaner version of the root cause analysis scoped for information security problems. Additionally, root cause analysis emphasizes the use of incident data and we suggest a novel research direction into conducting root cause analysis on cyber security incident data, define some of the obstacles, research paths, and utility of the direction. Our findings show that a problem needs to be costly to justify the cost-benefit of starting a full-scale root cause analysis project. Additionally, when strictly managed, root cause analysis performed well under time and resource constraints for a less complex problem. Thus, the full-scale Root cause analysis is a viable option when dealing with both complex and costly information security problems. For minor issues, a root cause analysis may be excessive or should at least be strictly time managed. Based on our findings we conclude that Root cause analysis should be a part of the information security management toolbox.**

Keywords*— *Information Security; Root cause analysis; Risk Management; Case study; Socio-technical; Empirical.

I. INTRODUCTION

Judging by the available literature on standards and methods, the common approach to dealing with problems in information security (InfoSec) is risk assessments (ISRA). Risk assessment aims to estimate the probability and consequence of an identified scenario or for reoccurring incidents and propose risk treatments based on the results. Although the InfoSec risk management (ISRM) approach is useful for maintaining acceptable risk levels, they are not developed to solve complex socio-technical problems or improve the system performance beyond keeping risk acceptable. For example, given a malware infected network the aim of the ISRA is to identify and deal with unacceptable risk, not to identify and deal with the root cause(s) of the problem. To aid the InfoSec industry in problem elimination, this paper continues the study of applying Root cause analysis (RCA) methodologies in InfoSec [1]. RCA is *"a structured investigation that aims to identify the real cause of a problem and the actions necessary to eliminate it."* [2]. The current RCA is not a single technique, rather, it describes a structured process that comprises of a range of approaches, tools, and techniques to uncover causes of problems, ranging from standard problem-solving paradigms, business process improvement, bench-marking, and to continuous improvement methods [2], [3]. The ISRA and RCA approaches are different in that RCA investigates incidents that have occurred with some frequency aiming to understand and eliminate the problem from a socio-technical perspective, while the objective of ISRM is to manage the risk by keeping it at an acceptable level.

Our literature review found that the application of formal RCA tools in InfoSec is an area that has remained largely unexplored. Therefore, the problem we are addressing in this study is to determine the utility of RCA for InfoSec and if it provides useful input to the decision-making process beyond the ISRA. The contribution of this research is knowledge regarding the application and performance of established RCA methods on InfoSec problems. Specifically, the paper addresses the following research questions:

1) How does the results from running a full-scale RCA extend the findings from the ISRA process?
2) Does the RCA approach have utility in tabletop exercises?
3) How well does the RCA approach work in a resource and time restricted setting?
4) Which RCA tools are suited for InfoSec analysis?

The problems are investigated mainly through case studies, qualitative assessment of results, and cost-benefit analysis.

This paper applies the seven-step process RCA methodology [2] for comparison of results, each step in the RCA method includes multiple tools for completing the step. The data collected for this study was primarily from technical re-

ports, historical observations and data in the target institutions. Together with qualitative interviews of stakeholders in two of the case studies. A key limitation of this study is that the applied RCA all come from the tool selection described in Andersen and Fagerhaug.

The structure of this paper is as follows: The following section addresses previous work on RCA in InfoSec. Section III provides a description of the applied ISRA method for case comparison and an in-depth description of the applied RCA method and the associated tools including statistical analysis. The primary case study presented in this paper extends the ISRA of a complex socio-technical problem with RCA and discusses the cost/benefit of the results. Firstly, we present the results from the ISRA application as a comparison basis followed by the results of a full scale RCA. The case study is of breaches to the access control (AC) security policy (SecPol) with consequent costly incidents, such as access card and Personal Identification Number (PIN) exchange between employees. This complex problem is located at the intersection of the social and technological aspects that many organizations may face. The ISRA and the RCA presents different tools and approaches, but both seeks to treat the problem at hand, which makes the output comparable. The primary case study investigates if RCA can be applied as a useful extension to the ISRM process for the AC SecPol problem. To investigate this issue, we qualitatively assess the results of a RCA conducted as an extension to a high-level ISRA of the problem. The second case study is of RCA performed as a tabletop exercise constrained to technical documentations of the Carbanak incident, in which a group of cyber criminals managed to steal large amounts of money from multiple banks. For this case study we analyze whether the RCA provides a useful insight into the incident. The third case study investigates the root causes of a DDoS attack against a Norwegian security awareness organization. This case was conducted under resource and time restrictions to test RCA performance under these conditions. Furthermore, this paper qualitatively evaluates the performance of RCA tools for InfoSec cases together with cost/benefit analysis. The RCA method suggests incident data as a source of knowledge [2] and we have conducted some preliminary work in applying incident data for RCA. This paper presents insight into key issues together for applying incident data in RCA with a proposal for future work. Lastly, we conclude the results.

II. Related Work

The RCA results presented in this paper represents the summary of the work presented in the Thesis "Root cause analysis for information Security" [4] and is an extension of the conference version of the paper "An Empirical Empirical Study of Root-Cause Analysis in Information Security Management" [1].

RCA was developed to solve practical problems in traditional safety, quality assurance, and production environments [2]. However, RCA has also been adopted in selected areas of InfoSec: Julisch [5] studied the effect of the RCA, by considering RCA for improvement of decision-making for handling alarms from intrusion detection systems. The study provides evidence towards the positive contribution of RCA, but it does not apply the RCA tools as they are proposed in the recent literature [2], [6], [7]. Julisch builds on the notion that there are root causes accounting for a percentage of the alarms, but proposes his tools for detecting and eliminating root causes outside of the problem-solving process, Fig. 1. A more recent study conducted by Collmann and Cooper [8] applied RCA for an InfoSec breach of confidentiality and integrity in the healthcare industry. Based on a qualitative approach, the authors find the root cause of an incident and propose remediation. Their results also show a clear benefit from applying RCA, although their RCA approach seems non-standardized, being primarily based on previously published complex problem-solving research articles. Wangen [9] utilizes RCA to analyze a peer review ring incident, where an author managed to game the peer review process and review his papers. This incident is analyzed by combining RCA tools and the Conflicting Incentives Risk Analysis (CIRA) to understand the underlying incentives and to choose countermeasures. Further, Abubakar et al. [10] applied RCA as a preliminary tool to investigate the high-level causes identity theft. The study applies a structured RCA approach [7] and identifies multiple causes and effects for setbacks to the investigation of identity theft. The Abubakar et al. study shows the utility of RCA for InfoSec by providing an insight into a complex problem such as identity theft. Hyunen and Lenzini [11] discuss RCA application in InfoSec by contrasting the traditional approaches to Safety and Security to highlight shortcomings of the latter. Furthermore, the authors propose an RCA-based tool for InfoSec management to address said shortcomings and demonstrate the tool on a use case. The tool is designed to reveal vulnerable socio-technical factors.

According to Wangen et al. [12] one of the most developed InfoSec risk analysis methods is the Factor Analysis of Information Risk (FAIR) [13]. The authors of FAIR have recognized the need for RCA as an extension of the ISRA method to eliminate problems and they propose a short version of RCA based on flowcharts (p. 366-373). Yet, the book does not go in-depth regarding the RCA method and does not provide any data regarding application. Some of the tools applied in an RCA are also recognizable in the risk assessment literature, for example, instruments such as Flowcharts and Tree diagrams model processes and events visually. Typical comparable examples from risk assessment are Event-tree and Fault-tree analysis, where the risk is modeled as a set of conditional events, however, these approaches are not specifically developed for InfoSec risk analysis. Schneier adapted the Fault-tree analysis mindset and created *Attack Trees* [14]. These tools resemble those of RCA. However, the frame for applying them is different in the sense that attack trees focus on the technical threat and vulnerability modeling, while RCA tools focus on problem-solving.

Although there are a couple of published studies on the application and utility of formal RCA methodologies, the previous work on RCA in InfoSec is scarce, and there is a research gap in experimenting with the RCA tools for solving re-occurring InfoSec problems. The studies we found provided positive results and motivation for further experiments with RCA for InfoSec problems.

III. Method

The research approach was case studies of problems occurring in a Scandinavian R&D institution (primary case study), multiple banks (tabletop exercise) and a small security

awareness company. The case studies were conducted to investigate the complex socio-technical security problems.

Each case study was conducted following the seven step RCA process, Fig. 1. Furthermore, we qualitatively assessed the results. For the primary case study, we also analyzed the differences in approaches between RCA and ISRA, findings, and treatment recommendation. Additionally, we applied a cost-benefit analysis to measure resources regarding time spent on conducting RCA and benefits concerning additional knowledge about the problem.

The following section briefly describes the ISRA approach applied in this study, while the second section describes the RCA approach. The latter contains a description of the seven-step RCA process, overview of the applied tools used, data collection methods, and a brief overview of the statistical methods used for data analysis.

A. ISRA Method for the primary case study

The ISRA was conducted as a high-level risk assessment for the institution, which revealed the need for deeper analysis of the problem. The ISRA has been developed to analyze risks that occur when applying technology to information, and revolve around securing the confidentiality, integrity, and availability of information or other assets [15]. By focusing on assets and vulnerabilities, these assessments tend to have a technical scope [16], [17] with estimates of consequences and respective probabilities of events as key outputs.

The ISRA method applied for the case study is based on the standard ISO/IEC 27000-series [15]. It was further substantiated with the Wangen et al. [18], [12] approaches, which center on estimations of asset value, vulnerability, threat, and control efficiency. These are combined with available historical data to obtain both quantitative and qualitative risk estimations. The applied method identifies events together with adverse outcomes and uses conditional probability to estimate the risk of each identified outcome. The results section provides a summary of the initial ISRA results. To illustrate the risk, we modeled it using the CORAS language [19].

B. Applied Root cause analysis method

In choosing a RCA framework, we looked at comprehensiveness, academic citations, and availability. Based on the criteria, our study chose to follow the seven-step RCA process proposed by Andersen and Fagerhaug [2], as shown in Fig. 1. Each step consists of a set of tools to produce the results needed to complete the subsequent steps, whereas step 7 was out of scope. Depending on the problem one or more tools are required to complete the RCA steps and conclude the root cause(s). As recommended in the methodology, we chose tools per step based on our judgment of suitability. All RCA in this study was conducted by a three-person team supported by a mentor. We have anonymized information according to the employer's requests. The following subsections describe each step in the RCA process and our selected tools starting with the tool applied for the primary case study (see [2] for further description).

1) Problem understanding: The goal of this step is to understand the problem and rank the issues. The tools for understanding the problem are meant to give a better understanding of the problem itself and what aspects in the case one should consider for further investigation. In order

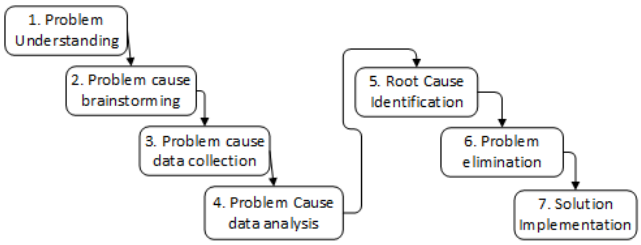


Fig. 1: Seven step process for RCA [2].

to know which tool to use on the problem and to handle the correct problem, it is important to first understand it. Below we will list some of the tools we tried out in our research.

Performance Matrices: are used to illustrate the target system's current performance and importance. The performance matrix contributes towards establishing priority of the different problems, factors, or problems in the system [2] (P.36-41): (i) which part of the problem is the most important to address, and (ii) which problem will reduce the highest amount of symptoms. The problems are qualitatively identified and ranked on a scale from 1 to 9, on performance (x-axis) and importance (y-axis).

Critical Incident: The main purpose of the Critical Incident tool is to understand what are the most troublesome symptoms in a problematic situation. By using the Critical Incident, you will get a better understanding of the aspects of the problem that must be solved, as well as the nature of the problem and its consequences. As with most root cause analysis tools, they are best used by a team to determine the cause of the problem. To work it requires an atmosphere of trust, openness and honesty that encourages people to disclose important information without fear of the consequences. This applies to all tools but especially Critical Incident.

Swim Lane Flowchart: Swim Lane Flowchart shows the flow of events through a timeline and shows connections between events. The chart is divided into players where each player has his horizontal path.

2) Problem Cause Brainstorming: The main idea of this step is to cover other possible issues that may be causing the problem, not thought of in Step 1. Brainstorming is a technique where the participants verbally suggested all possible causes they could think of, which was immediately noted on a whiteboard and summarized together at the end. Brainstorming can take place in different ways, structured or unstructured brainstorming and brain writing. A structured brainstorming is based on the members coming back with suggestions to ensure that no person dominates the process. Unstructured brainstorming allows spontaneous responses from anyone in the group at any time. Brain Writing can be done in two ways. Group members write down their ideas on so-called ID cards, or on a blackboard. During brainstorming, it is important that ideas and suggestions are not criticized until all the ideas and suggestions will be reviewed.

3) Problem Cause Data Collection: The data collection phase helps to make searches for problems more accurate. Random problem solving tends to result in assumptions and guesswork while structured RCA is based on a systematic collection of valid and reliable data that is an important step in

root cause analysis. It is therefore important to plan carefully what tools one might think about using. RCA recommends several data collection techniques [2].

Interviews: For the primary case study, this study chose scientific interviews as the main data collection approach as this study required an in-depth understanding of the motivations for AC SecPol violation problem. The interviews were conducted in a face-to-face setting, and was designed using category, ordinal, and continuous type questions together with open-ended interview questions for sharing knowledge about the problem. The interview subjects were primarily categorized as representatives of key stakeholder groups within the organization and one group of external contractors. Each interview had twenty-six questions with follow-up questions if deemed necessary to clarify the opinion or to extract valuable knowledge from particularly knowledgeable individuals. More informal interviews were also applied as a data gathering method in case study 3.

Check Sheet: is used to systematize collected registered data. The main purpose is to ensure that all data collected complies with reality. Can be used to record the frequency of events that are believed to cause problems.

Incident data analysis: Andersen and Fagerhaug [2] proposes to analyze incident data as a part of the RCA. However, analysis of InfoSec incident data is quite complex and the future work section outlines some of the research problems encountered working with RCA and incident data. Additionally, we propose research directions for solving the problems.

4) Problem Cause Data Analysis: The purpose of this phase is to clarify possible causes before attempting to solve the problem in the final preparatory stage, for example, how are the possible causes related to the problem and what is the most harmful? The purpose of the data analysis phase in the final preparatory stage before attempting to solve the problem is to clarify possible causes. It is important to look at how different aspects of the problem are linked. In data analysis, the following tools can be used:

Statistical analysis: We applied a variety of statistical data analysis methods specified in the results, and the IBM SPSS software for the statistical analysis. A summary of the statistical tests used in this research is as follows.

For *Descriptive analysis* on continuous type questions, we applied the median as the primary measure of central tendency. We also conducted *Univariate* analysis of individual issues and *Bivariate* analysis for pairs of questions, such as a group belonging and a continuous question, to see how they compare and interact. As the Likert-scale seldom will satisfy the requirements of normality and not have a defined scale of measurement between the alternatives, we restricted the use of mean and standard deviation. We analyzed the median together with an analysis of range, minimum and maximum values, and variance. This study also analyses the distributions of the answers, for example, if they are normal, uniform, bimodal, or similar. We used Pearson two-tailed *Correlation test* to reveal relationships between pairs of variables as this test does not assume normality in the sample.

The questionnaire had several open-ended questions, which we treated by listing and categorizing the responses. Further, we counted the occurrence of each theme and summarized the responses.

Affinity diagram: helps to correlate apparently unrelated ideas, conditions, meanings, and reasons so that they can collectively be explored further. When analyzing qualitative data, Affinity Diagram is useful as it groups data and findings of underlying relationships into groups.

Relationship Diagram: Relationship diagram is a tool used to identify logical relationships between different ideas or problems in a complex and confusing situation. In such cases, the strength of the relationship diagram is its ability to visualize such relationships. The main purpose of a relationship diagram is to help identify issues that are not easily recognizable.

5) Root Cause Identification: The goal of this step is to identify the root cause(s) of the problem. From the list of possible causes created and analyzed previously, this step is designed to identify the root cause. With root cause identification, the goal is to develop solutions that will eliminate the symptoms and thus eliminate the problem. In terms of duration and complexity, this stage is rarely the hardest or longest. With thorough preparation, you can usually go through this stage quickly.

Cause-and-Effect chart (Fishbone diagram): Fishbone is a tool that analyzes a relationship between a problem and its causes. It has aspects of brainstorming and systematic analysis to create an effective technique. The main purpose of the tool is to understand what causes a problem together with the secondary causes/factors influencing the problem. It can be used to develop as well as group reasons for a problem. The Fishbone diagram also evaluates systematic causes, finds the most likely root causes and should map to the undesired effect to the problem.

Five Whys is designed to identify a problem then ask why this is a problem. When you get an answer, ask why. This is usually repeated five times until you get to the root cause.

6) Problem elimination: The goal of this step is to propose solutions to deal with the root causes of the problem, Andersen and Fagerhaug [2] describe primarily two types of tools for drafting treatments; one is designed to stimulate creativity for new solutions, while the other is designed for developing solutions. This step is successful if you remove the correct root problem (s), the symptoms will disappear along with the problem and will not resume.

Systematic Inventive Thinking (SIT): It is based on investigating one or more components of the problem. All components should then be assessed using the five SIT principles [1]. These principles are as follows: Attribute dependency: Assess if a change in component will lead to improvement. Component control: examine how the component is connected to the environment around it. Replacement: Replace something in the component with something from the component's environment. Displacement: Assess if the component can increase performance by removing part of the component. Division: Assessing splitting of a component or product's attributes can provide improvement.

Countermeasures Matrix: It is a method to help you prioritize what actions to take. Priority is established by ranking based on the impact and feasibility of recommended measures.

7) Solution implementation: Solution implementation focuses on the implementation phase. This step includes how to organize the implementation of solution implementation and how to develop an implementation plan. We have not been able to implement solution implementations in our task, but we have made suggestions for the execution of the tools to which the book refers.

Tree Diagram: Implementation processes can be complicated, but in order to break down and organize the work, Tree Chart is used to structure the activities. It is a tool that is easy to use to break down major tasks in the business to manageable sizes. Tree Diagram is simply a way to represent a sequence of events.

IV. Primary Case Study: Risk Assessment of Access Control Policy Violations

In this section, we first present a summary of the results from the ISRA, in terms of risk estimation and proposed treatment. Further, we present the results from our RCA for comparison.

The case data was collected from an institution whose IT-operations delivers services to about 3000 users. The organization is a high-availability academic organization providing a range of services to the users, mainly in research, development, and education. The IT Operations are the internal owners of the AC regimes and most of the lab equipment; they represent the principal in this study. The objectives of the IT-operations is to deliver reliable services with minimal downtime, together with information security solutions.

During the last years, the Institution has experienced multiple incidents of unauthorized access to its facilities. The recurring events primarily lead to theft and vandalism of equipment in a range of cost that is deemed unacceptable. Thus, the hypothesis is that this has partially been caused by employees and students being negligent of the SecPol regarding AC, providing unauthorized access to the facilities. While the SecPol explicitly states that both the token and the PIN are personal and shall not be shared, there has been registered multiple incidents of this occurring.

A. The Risk of Access control policy violations

The goal of the ISRA was to derive the annual risk of the incidents. This section summarizes the asset identification and evaluation, vulnerabilities assessment, threat assessment, control efficiency, and outcomes.

The Institution had two key asset groups: (i) hardware and (ii) physical sensitive information, both stored in access controlled facilities. The hardware's primary protection attribute was availability, and the value was estimated in the range of moderate according to the budget, with a low to medium importance in the day-to-day business processes.

The two controls in place are primarily (i) AC mechanisms - physical control in place to prevent unauthorized accesses and mitigate the risk of theft. (ii) The SecPol - administrative control, which is a written statement concerning the proper use of AC mechanisms.

For the vulnerability assessment, experience showed that illegitimate users were accessing the facilities on a daily basis. We identified two primary vulnerabilities; (i) lack of security training and awareness, whereas the stakeholders do not understand the risk exposure of the organization. (ii) Insufficient organizational security policies, whereas the SecPol itself lacks clear consequences for breaches, leaving the personnel complacent. The main attack for exploiting these two vulnerabilities was social engineering, where the attacker either manages to get a hold of a security token and PIN. Alternatively, the attacker manages to gain unauthorized access to the facilities by entering with others who have legitimate access (tailgating). With the number of stakeholders having access, both attacks are easy for a motivated threat actor. The exposure is summarized in Table I.

TABLE I. SUMMARY OF VULNERABILITY ASSESSMENT.

Scenario	Vulnerability Description	Attack description	Attack Difficulty	Vulnerability Severity	Exposure Assessment
A1	Lack of Security Training and Awareness, Insufficient InfoSec Policies	Social Engineering - Employee or Student Gives away Token and PIN (Likely)	Medium	Very High	High
A2	Lack of security training and awareness, Insufficient InfoSec Policies	Social Engineering- Employee or Student leaves doors opened for convenience	Easy	Medium	Medium

For the threat assessment, the experts identified one threat group motivated by a financial incentive with the intent of stealing either physical equipment or sensitive information, with two actors; (i) Actors who frequently steals small items, representing high frequency - low impact risk. (ii) Actors who conduct a few significant thefts, representing the low frequency - high impact risk.

1) Risk Analysis Results.: The risk is modelled in Fig. 2 using the CORAS modelling language [19]. From the model, we have two likely conditional events where (i) the attacker obtains access token and PIN, or (ii) access to the facilities by piggybacking employees or students. The ISRA results showed that the most severe risk facing the organization is theft of sensitive information, while physical theft of equipment is also a grave risk. According to past observations, the risk is greatest during holidays with few people on campus. The two primary risks were major equipment thefts during the holiday season and several minor equipment thefts that aggregated into an unacceptable amount (not differentiated in the CORAS model). In addition, we have the low probability and high impact risk that sensitive information gets compromised through this attack.

2) Implemented Treatment - Camera Surveillance: As a result of the ISRA, the treatment implemented to reduce the two risks was camera surveillance of the main entry points of buildings. Firstly, this treatment has a preventive effect in the sense that it will heighten the attack threshold for threat actors. Besides, it will provide audit trails that will be useful in future investigations. Camera surveillance had also been proven to reduce the number of incidents as well as increasing the amount of solved crimes in similar institutions. This data indicates a high control efficiency; however, the measure also comes with some drawbacks, such as equipment cost together with the required resources to operate the system. Due to the data collection on employees surveillance brings, this risk treatment also subjects the organization to requirements from

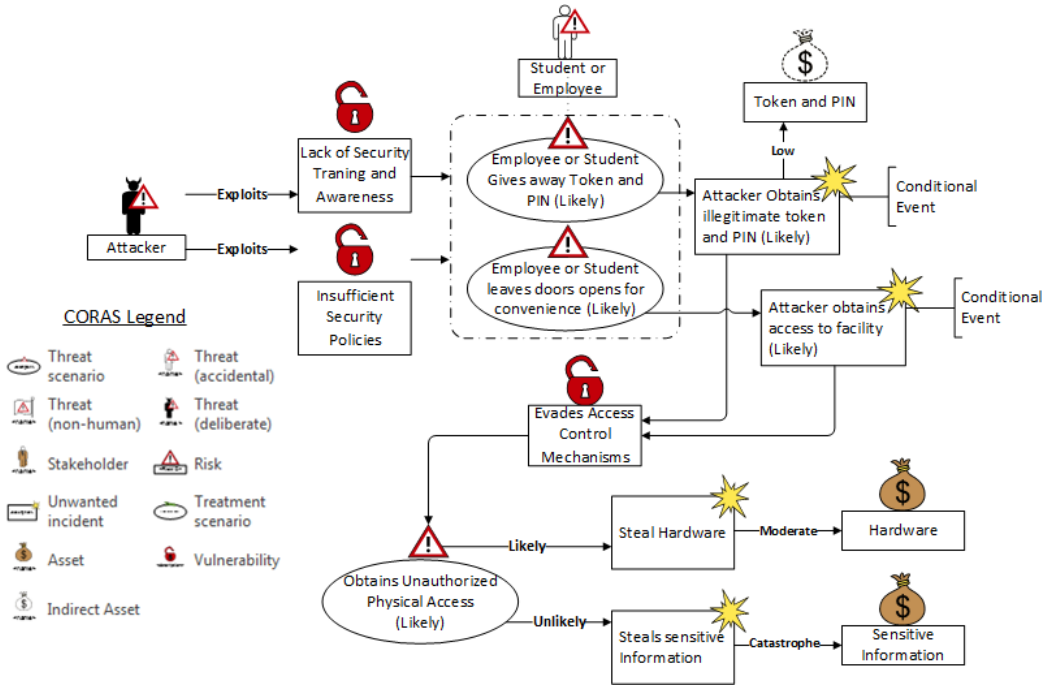


Fig. 2: Risk modelled with CORAS [19]

data privacy protection laws. Neither did it address the socio-technical problem with the SecPol, card swapping, and card lending.

V. Primary Case Study: RCA of Access Control Policy Violations

In this section, we present the results from conducting the RCA according to the method described in Section III-B. The results are derived from conducting RCA on the previously outlined problem and risk; we outline the hypothesized root causes and proposed treatments.

A. RCA Process, Step 1 & 2 - Problem Understanding and Cause Brainstorming

The goal of these steps is to scope the RCA and center on the preliminary identified problem causes. The performance matrix, Fig. 3, is used to rank the identified causes on their *Importance* and *Performance*. With the help of resource persons, the team derived six topics from the preliminary *RCA steps 1 & 2*, Fig. 1): (i) Theoretical knowledge of the SecPol for AC, (ii) Practical implementation of the SecPol for AC, (iii) Consequences for policy breaches, (iv) Security Culture, (v) Backup solutions for forgotten and misplaced cards, and (vi) Card hand out for new employees. The RCA team and the expert ranked the issues and prioritized the data collection step accordingly, illustrated in Fig. 3.

B. RCA Process Step 3 - Data Collection

For the categorical analysis, the team used age, gender, and stakeholder group as the primary categories, with the emphasis on the latter as our hypothesis was that parts of the root cause are found in conflicting interests between internal groups. The team interviewed thirty-six people located at the site, Fig. 4 displays the distribution among the six primary

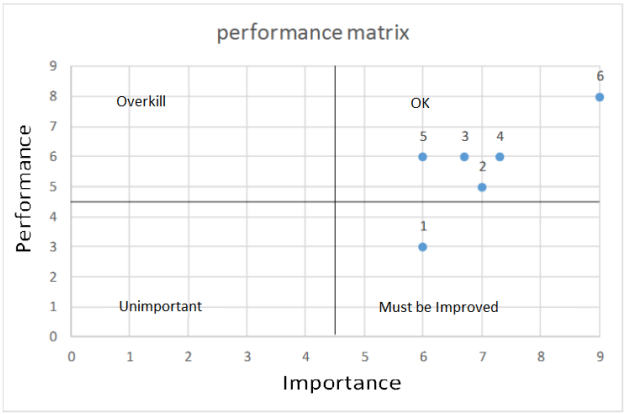


Fig. 3: Performance matrix.

TABLE II. DEMOGRAPHICS INCLUDING AGE AND SEX DISTRIBUTIONS

	Age				Sex		
	Group	Freq.	Percent		Group	Freq.	Percent
Valid	20-29	8	22,2	Valid	Women	10	27,8
	30-39	7	19,4		Men	26	72,2
	40-49	10	27,8		Total	36	100,0
	50-59	8	22,2				
	60-69	3	8,3				
	Total	36	100,0				

stakeholders. The interview subjects for the academic staff, Ph.D. Fellows, and M.Sc. students were chosen at random. The representatives of management and IT and security were key stakeholders in the organization, such as decision-makers

and policy writers.

C. RCA Step 4 - Problem Cause data analysis

The Descriptive analysis showed that about half of the respondents had read the SecPol. All but two reported that it is was not allowed to lend away cards, whereas the remaining two did not know, indicating a high level of security awareness for the issue. Also, the study uncovered uncertainty among the respondents when we asked them about what the potential consequences for breaching the SecPol would bring for the employees. Whereas most of them assumed no consequence, and none perceived any severe consequences. We also uncovered that most people would be reluctant to admit to sharing cards. Further, we asked them *"How often do you think access cards are shared at the Institution?"* on a scale from 1 - 5 (1- Never, Yearly, Monthly, Weekly, 5 -Daily), to which the respondents thought that this is an issue that occurs on at least a weekly basis (Median 4). Using the same scale, the team asked how often the respondents had the need to borrow cards from others. Over half reported to not ever had the need, while twelve reported having had to lend cards on an annual basis, only two reported having the problem more than that. However, half of the respondents said to have been asked by others to borrow cards, which documented the frequency of the problem.

TABLE III. NOTABLE DIFFERENCES BETWEEN GROUPS ON "HOW LONG DID IT TAKE FOR YOU TO GET ACCESS TO THE FACILITIES YOU NEEDED?" (BETWEEN 1 VERY LONG - 6 IMMEDIATE ACCESS)

Category	N	Range	Median	Minimum	Maximum	Variance
Management	3	0	6,00	6	6	0,000
Senior Academic Staff	17	4	6,00	2	6	1,654
Ph.D. Students	7	5	5,00	1	6	3,238
BSc. and MSc. Students	3	4	3,00	1	5	4,000
External Contractors	3	3	4,00	1	4	3,000
Total	33	5	5,00	1	6	2,729

1) Summary of categorical analysis: The statistical analysis showed differences between the responses of men and women; where the latter viewed incidents involving card borrowing among employees more severely than men. The women in our sample also believe that it is more likely that employees admit to borrowing cards. Another visible difference between the stakeholder groups was who had read the policy, where all the representatives of the Management and IT and Security groups had read it. The Ph.D. Fellows and the student groups scored the lowest on having read the policy. Another observable finding was that the waiting time varied between the groups, whereas the permanent employees perceived the shortest waiting times, Table III.

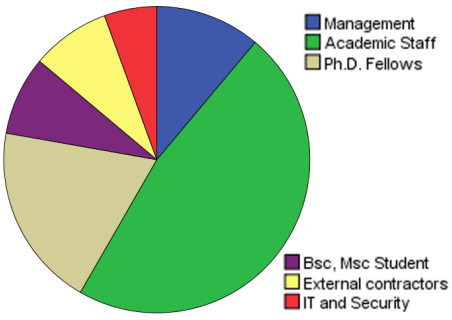


Fig. 4: Distributions of stakeholder groups included in the study

2) Qualitative analysis of differences between groups:

IT and Security. The IT operations owned much of the hardware in the facilities and was in charge of both designing, implementing, and operating the AC policy. Both representatives had read the policy and considered it important that staff and students also know the policy. The IT operations believed that card lending is an increasing problem within the institution, especially in the modern facilities where AC mechanisms are more frequent. One also answered that since he had been involved in developing the policy, he felt more ownership of it and, therefore, experienced a greater responsibility to follow it than other departments. They also felt the legal responsibility not to break the policy due to owning the AC system.

Management. This group consists of middle and upper management, which had all read the SecPol. Half believed it was important to have those who will be subject to the policy involved in the policy development process. When we asked this group about what they saw as the worst scenario, this group had similar opinions: their main concerns was loss and compromise of information together with relevant legal aspects. Two members of this group reported that they did not get the service they expected from IT regarding forgotten cards. Three out of four said that they believed the security culture to be good, while the last one reported the security controls to be cumbersome.

Senior academic staff. Consists of different types of professors, researchers, and lecturers, and represents the majority of employees in the case. This group was the largest with the most widespread opinions. Regarding the SecPol, several expressed discontent and said that it was neither security department or IT service that should be responsible for it. The organization should provide the content of the policy to ensure that it was not an obstacle in the day to day work. Further, delivering on the aims and goals of the organizational assignment should be compared to the potential harm from card swapping incidents, meaning that the policy should be designed with a better understanding of risk. An example of this was that employees must have access to rooms to do their job where a too-strict policy would stand in the way. Regarding this, several mentioned that if the cards were not lent to other employees, it would be very problematic due to the lack of backup solutions. They missed good fallback solution if one had forgotten access card.

Ph.D. Fellows. Out of this group, only one had read the SecPol. Most assumed it was not allowed to lend out their access cards, but two said they did not know. One expressed discontent from not receiving his access card quick enough, which he hypothesized as one of the reasons for borrowing other people's cards. Longer times to hand out access cards may force them to lend cards internally in an office. Another issue was that Ph.D. Fellows occasionally worked with students and that they often needed access to restricted facilities to be able to work. This issue required the Ph.D. fellow either to open the door physically for the students or to loan them their card. When we asked about the security culture, the responses were split: Two did not

know, one thought that security was good, another one said that people trust each other, one said it was wrong, while one said that people knew that they should not lend it to others. The last one said that others could borrow it for practical reasons.

Students. Represents the main bulk of people with access to the main facilities, but with limited access to offices and employee areas. Only one of the students had read the policy, and none of the students who participated knew of any instances of card lending, although two out of three had been asked by someone if they could lend them their cards.

External contractors. Represents the contractors in charge of running the physical facilities, such as cleaning personnel and physical maintenance. In the External group, only one had read the policy. All believed that it was not allowed to borrow cards and that the school saw this as a serious offense. Only one of them reported having had the need to borrow a card.

D. RCA Step 5 - Identified Root causes

The interviews with the groups provided an insight into the many views on this problem and the complexity it entails, visualized with the Fishbone diagram in Fig. 5. Based on our RCA we found five possible root causes:

1. Uncertainty regarding fallback solutions. We found that there was uncertainty surrounding available backup solutions among all the stakeholder groups. Where 14 of the 31 respondents were undecided if there existed any fallback solution, and suggested to create better backup solutions. 17 said there existed backup solutions, but we uncovered different opinions regarding what these were and who was responsible for them. For example, six respondents thought they could summon the IT department, three thought the student help desk, while the remainder thought either management could help or ask a colleague to lend them access cards. Even from the two key stakeholders in IT the replies were contradictory.

2. Discomfort when using fallback solutions. Two of our respondents reported to have forgotten their cards and had contacted the on-campus card distributor to use the fallback solution. The respondents meant they had not been well-received and had not gotten the help they needed. Overall, they reported the situation to be discomforting, which was unfortunate, as this may lead to the employees using different methods for solving the problem.

3. Misaligned SecPol regarding authorization. Our interviews highlighted that being able to do their work is the most important goal for every employee. Thus, the SecPol should aim to facilitate this aim. Too strict AC will in some cases lead to obstruction in day-to-day tasks and lead to employees finding workarounds, which may compromise security, such as asking trusted co-workers to borrow cards. Some of the respondents reported not having been included in the development of the SecPol and felt that it was misaligned.

4. Too much security. In especially one of the most modern buildings, there is a very strict AC regime in place, where low-level security rooms and facilities are regulated. Several of the respondents highlighted this as the main reason for card lending. These low-security rooms only required the card and not the PIN code, so the respondents did not consider this a serious breach of policy. Several of our respondents said that this was too much security and could not understand the reasoning underlying this decision.

5. Lack of risk awareness and consequences. 33 out of 36 defined possible negative consequences for the institution, so, the awareness around possible risks for the institution was high. However, we found that less than half of the respondents had read the overarching SecPol and that the respondents were unaware and uncertain about the organization's and their personal risk if their cards went astray. Everybody agreed that it was a bad thing, but nobody could say with certainty what the consequences would be, if any at all.

E. RCA Step 6 - Proposed root cause treatments

Based on our findings we conducted *Systematic Inventive Thinking* and came up with following root cause treatments:

Improve fallback solutions. Regarding root cause 1 and 2, the RCA team proposed to develop a solution for reserve access cards with adequate and tailored room access. The solution should provide basic access to low-security level facilities, with tailored room access according to stakeholder needs. This suggestion should be a public and low threshold offer for those who have forgotten or misplaced their cards.

Align SecPol with objectives. Regarding root causes 3 and 4, the RCA team proposed to risk assess the need for physical security and AC for the facilities based on the organizational goals, employee needs, and the assets stored in the room. Include key stakeholders in the process and focus on balancing productivity and security to revise the security baseline.

Improve the overarching SecPol. Regarding cause 5, the RCA team proposed to improve the overarching SecPol, the suggestions were: (i) clarify consequences for breaches of policy, (ii) assigning a responsible for sanctions per department, (iii) including the employees in the shaping of policy, and (iv) increase the accessibility of the policy.

Improving risk awareness. Regarding root cause 5, we also propose to improve risk awareness among the stakeholders, by running awareness campaigns including both the risks the organization and employees are facing. As a part of this, we proposed to create an information bank regarding risks, fallback solutions, and how to make use of them.

F. Comparison of Risk Assessment and RCA Results

Upon completing the RCA, we see that the results from the ISRA and RCA provide different models of the same problem. The information gathered from the ISRA process was scoped towards technical risks with solutions for reducing probability and consequence. Furthermore, we found the RCA to work better to visualize complexity and providing insight into the human aspects of the problem. However, the RCA process was resource intensive and required extra training to complete. The RCA process also required the inclusion of more stakeholders than the ISRA.

The results show that the benefits of the RCA are a better understanding of the social dimensions of the problem, such as conflicts between users and the security organization. This insight provides an improved decision basis and an opportunity for reaching a compromise with the risk treatment. The risk

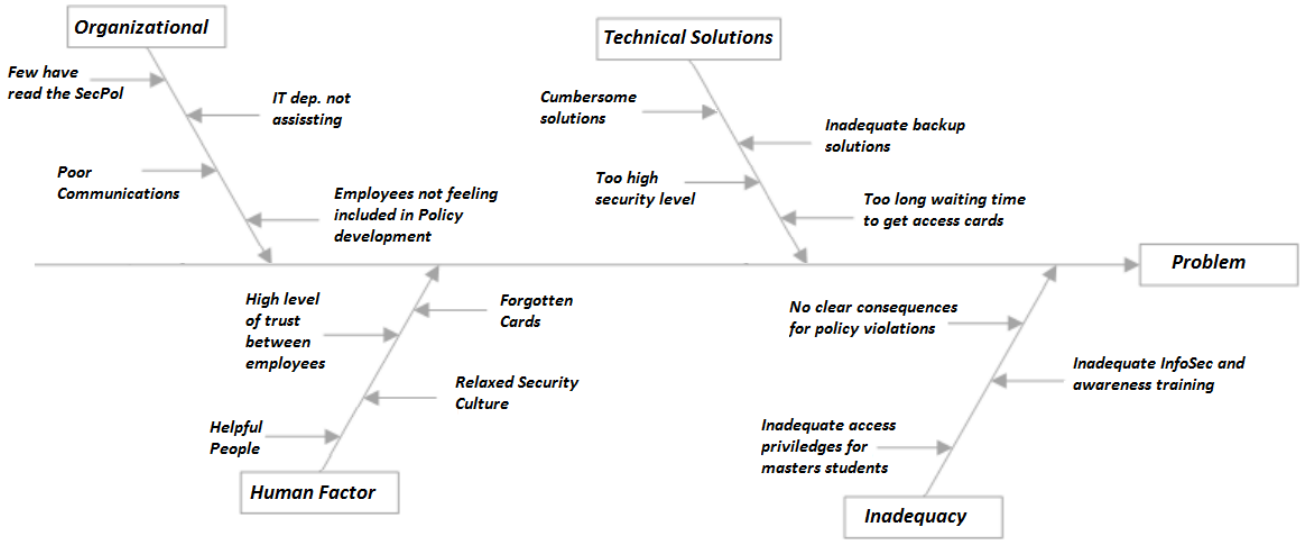


Fig. 5: Fishbone diagram illustrating contributing causes to the main problem.

assessment team were aware of two (cause 3. and 5.) out of the five identified root causes of the problem. Thus, in our case study, the RCA did provide a valuable extension to the risk assessment for solving the problem. The RCA results showed all root causes to be on the administrative and human side of the problem. Thus, the treatments produced from the two approaches were different; ISRA produced a technical treatment in camera surveillance, while RCA produced multiple administrative treatments, each for addressing separate root causes.

Although the ISRA did highlight the vulnerabilities related to the human factor and risk perception as one of the risk factors, in this case, the decision-makers did not opt for revision of the AC policy. To summarize, the ISRA findings viewed card lending as a technical security problem, while RCA extended the knowledge into the administrative problem.

Moving on, the nest section presents the results and evaluation of RCA as a tool for tabletop exercises.

VI. Tabletop RCA case study: Carbanak

The RCA tabletop case was meant as an experiment on how well RCA worked on a case with only historical data and technical documentation available. A tabletop exercise is a discussion-based exercise where personnel meet in a classroom or simulated setting. The group got presented with a scenario to validate the content of plans, procedures, policies, cooperative agreements or other information for managing an incident. Which means that there was restrictions regarding access to new information. This case investigated how the RCA work on a tabletop case.

The case study concerned attacks on multiple banking institutions, where the attackers managed to steal large amounts of money. The attack was often referred to as Carbanak, but also Anunak [20], [21]. The tools used on each step during the analysis of the tabletop case is described in the following sections.

A. Problem understanding

Since we worked only with documentation of the attack we needed to gain an overview of information that was gathered. Based on the constraints we found swimlane flowchart to be the best suited tool for modelling the attack, Fig. 6. As noted in Section III-B this flowchart works by having a swimlane representing each actor on the y-axis, where the lanes progress following the x-axis, which represents the chronological order of actions or events in time. The flowchart had three lanes representing actions taken by either the attacker, the bank employees and administrator. This tool visualized the main events and how one lead to another. We found that the primary way the attackers got into the banks was by using phishing emails that was sent to employees, as documented at page 3 in a Kaspersky report [20]. Furthermore, the attackers exploited vulnerabilities in Microsoft Office and Word before installing the backdoor named Carbanak. In a video of a presentation by a Kaspersky employee, the employee said that the attackers escalated their privileges by sending an email from the infected computer to the IT help complaining that the computer ran slow [22]. The IT employee then logged in to the computer and had his or her credentials stolen by a keylogger. The attackers now escalated their privileges by obtaining access to more machines to spy on additional bank employees. The attackers then observed common working patterns and learned how to use the tools the employees was using. This knowledge allowed them to proceed with their attack and steal money from the bank. The swimlane chart helps to visualize the attack flow and allowed us to obtain an overview of the situation, steps taken, involved parties, and the timeline.

The second tool applied in the problem understanding was *Critical Incident*, which is a tool meant to aid in the process of uncovering symptoms of the most problematic root causes [2]. Critical Incident is a two column table where the left column is the name of a type of incident and the right column is the frequency occurrence. Due to the constraints of the tabletop exercise, we did not have the numerical data on the frequency of different incidents that we needed to complete

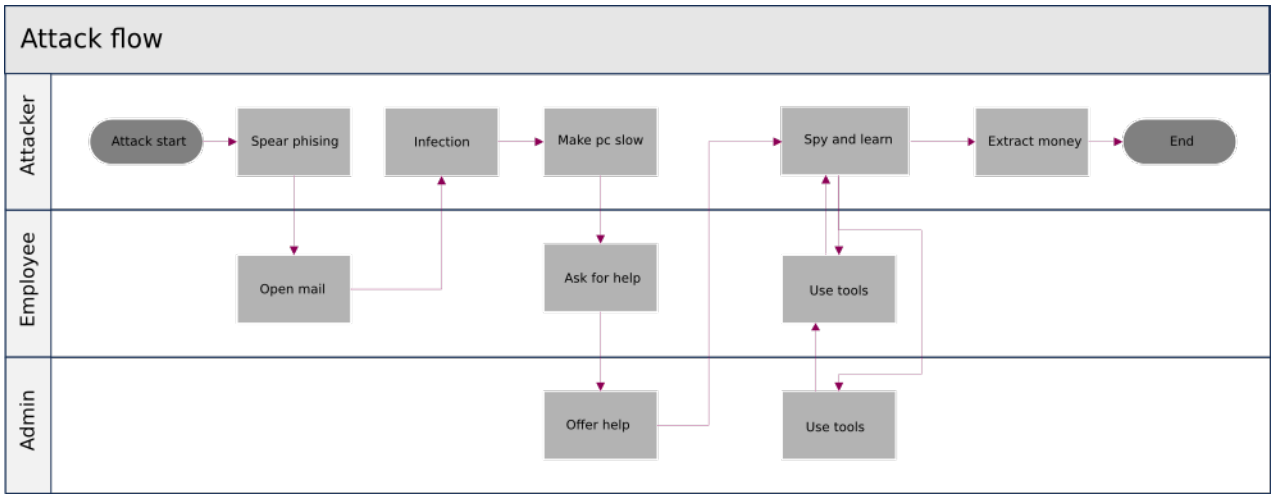


Fig. 6: Swimlane Flowchart illustrating attack flow, the timeline of the attack from left to right.

TABLE IV. CARBANAK CRITICAL INCIDENT TABLE FROM PAGE IN 88

Name	Frequency
Suspicious traffic	High
Monitoring of machines	High
Opening of e-mail attachments	Medium
Policy violations	Medium
Unfaithful employees	Low
Attackers has access to servers	Low
Undiscovered infections in IT systems	Low
Ignorance of spyware	Low
Low security awareness among coworkers	Low

the tabletop case. The solution was to use logical reasoning to estimate which incident most likely had the highest frequency based on the technical documentation. We estimated this using weights ranging from High - happens daily, Medium - weekly, to Low - monthly or less often.

Suspicious traffic has the highest frequency in the table. This traffic represents communication that goes from and between IT equipment inside the banks that are infected and the machines owned by the attackers. The frequency is set to high as we expected that this traffic could reasonably be argued as high. The attackers monitored machines owned by the banks in order to see how the employees operated them, page 21 [20], and this is also placed as high frequency. Since the collected documentation described that the attackers got into the bank through email attachments it may be possible that it is not too uncommon that employees opens and runs files received in mail attachments. An employee may break a policy without being purposefully unfaithful, but rather negligent. Thus it was ranked as with medium frequency. The amount of times it is believed that the attackers felt a need to enter the server infrastructure of the banks is deemed as low. This happens most likely when the attackers wants to place backdoors or start malicious processes.

B. Problem cause brainstorming

Unstructured brainstorming aims to brainstorm on possible causes and present them to the group members. The results were generated as a list of problems that can be improved. We also wanted to identify possible consequences that originated from the problem being analyzed.

The produced list from the brainstorming process is not sorted in any way, and may contain suggestions that more or less overlap. The following step is therefore to sort the list and merge suggestions that overlap and improve upon the suggestions. The list is then sorted according to what is deemed to be the most realistic cause of the problem by the RCA team.

Lastly, in the third step we categorized the proposed problem causes. A total of four categories were created, where the first category deals with the training of employees and the follow-up of the training. This category included the suggestion that there might be a lack of policies or a lack of training and exercise of said policies. The second category referred to weaknesses such as lack of updates and the failing to notice suspicious activity in their systems. The third category referred to monitoring of network and the fourth and last category was about corporate threats.

C. Problem cause data collection

It was not possible to do an active data gathering during the tabletop case since there is no access to personnel to interview or systems to look into.

D. Problem cause data analysis

In this phase we used Relation Diagram and Affinity Diagram III-B. The Relation Diagram, Fig. 7, illustrates the relation between different systems and computers, compared to the Swimlane Flowchart which showed the flow of actions over time. A large circle was drawn on a whiteboard and elements that was viewed as important and overarching was written around the circle. Arrows was then drawn between these items according to relations. In this case, we did not find any new relations that we did not expect already from the documentation and previous RCA steps.

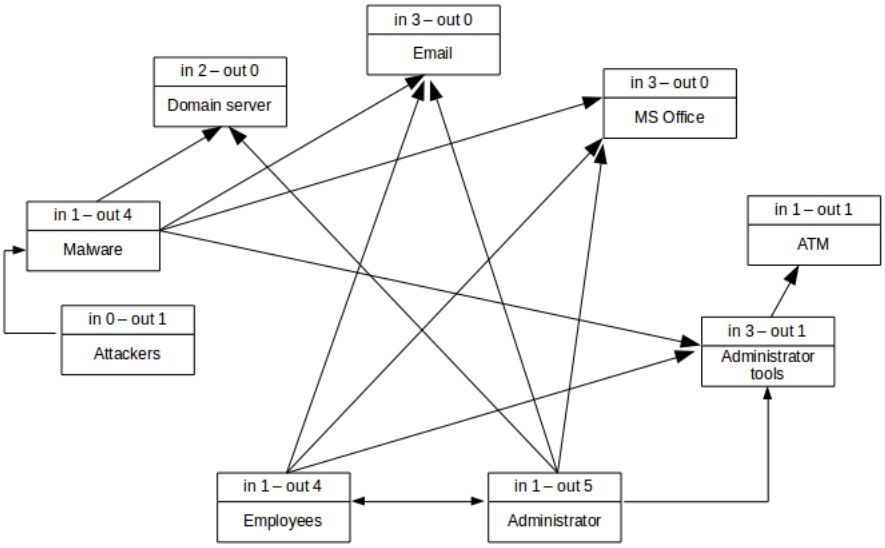


Fig. 7: Relation Diagram illustrating connections between employees and systems.

The Affinity Diagram is a tool for organizing ideas and data. Fig. 8 represents the affinity diagram for the case where we organized the problem situations under the groups malware, training, and environment.

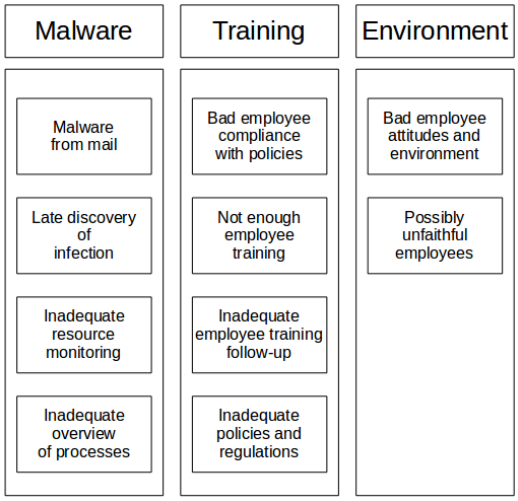


Fig. 8: Affinity Diagram illustrating ordering by context.

E. Root cause identification

The Five Whys approach was used to identify root cause, Table V. The tool was used on the question of why the ATM's gave away money to the criminals. The reasons that answers each of the five why questions was suggestions of what could be realistic, given that this was a tabletop case. The tool did uncover one root cause, however it is not visible if there exists more root causes. The tool appear to be able to isolate the users on one root cause unless it is run several times. The tool was very quick to complete.

TABLE V. CARBANAK TABLETOP CASE, FIVE WHYS

ATM giving away money	Reason
Why?	Because the system was compromised
Why?	Because the attackers exploited a vulnerability when employees opened mail
Why?	Because their software was not up to date
Why?	Because the bank had inadequate update routines
Why?	It was not considered to be critical enough

F. Problem elimination

The tool Countermeasure Matrix, Table VI, was used to suggest worthy countermeasures based on efficiency and feasibility. The way the group solved it was by rating efficiency and feasibility of a counter measure from 1 to 5. The two scales are then summarized, and if the number is ten or above, an action is suggested to be taken. However, for upgrading of legacy systems an action was set to not do anything because it could be unrealistic in many large organizations. We define updating as installing a newer version of a software while patching as installing security patches and bug fixes of a given version of the software. With baseline, it was meant as to have a hash of most files in a system that could be used to detect changes to these files.

G. Solution implementation

A Tree Diagram, displayed in Fig. 9, was used to show which solutions and problems was related to each others, sorted under categories that was linked together with branches that are rooted to the main problem.

H. Assessment of RCA as a tabletop exercise

We found that doing a tabletop case gave us experience on the choosing and execution of RCA tools, but it did not provide any new information about the case being analyzed.

We do see that doing a RCA requires allot of information

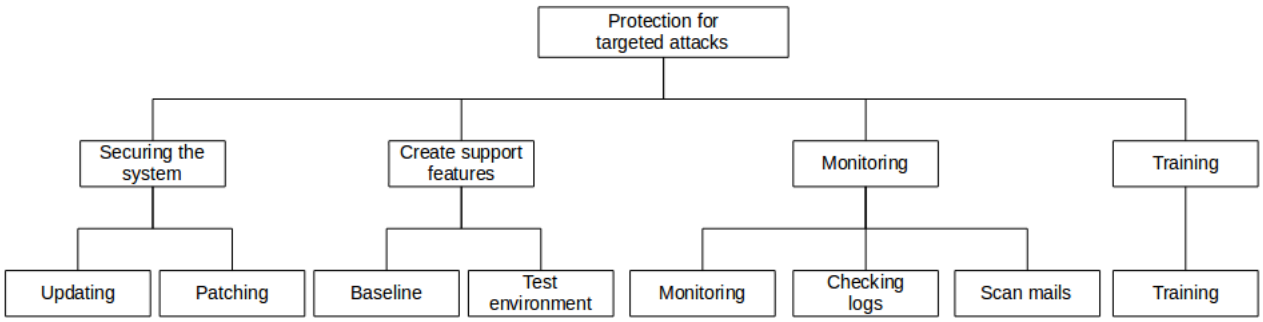


Fig. 9: Tree Diagram illustrating relations between problems and solutions.

TABLE VI. CARBANAK TABLETOP CASE, COUNTERMEASURES MATRIX

Countermeasures	Efficiency	x Feasibility	Sum	Action
Updating	4	5	20	Yes
Patching	4	5	20	Yes
Auto-updates	4	2	8	No
Training	2	5	10	Yes
Baseline	4	3	12	Yes
Monitoring	4	4	16	Yes
Temporarily close bank	2	1	2	No
Trace attack back to attacker	2	1	2	No
Upgrade legacy systems	5	2	10	No
Sandboxing	3	3	9	No
Test environment	4	3	12	Yes
Going through logs	3	5	15	Yes
Scan incoming emails	3	4	12	Yes

about the case. For practice reasons a tabletop case works with providing practice in selecting tools and applying them, but in order to do an actual RCA and discover root causes and implement solutions to them it is necessary to have access to key personnel, logs about what happened, and any other information that can be gathered.

As the tabletop case was dealing with protection versus an APT, we got the impression that completely eliminating a treat of attack from such an opponent is not completely possible. However, eliminating a root cause for an exploited vulnerability increases the organizations resistance towards attacks from the attacker.

VII. Case Study 3: Root cause of DDoS against small security awareness organization

In this case study, we analyzed a case we received from a small Norwegian security awareness organization of a DDoS attack that occurred in May 2015. At the time, the organization consisted of approximately ten employees with the primary objective of preventing and mitigating the consequences of identity theft. In this case we applied the RCA tools to investigate the root cause why their primary website became unavailable during the attack. We also studied whether the solution proposals implemented to date answer all the problems or if any problems remain.

The data sources we had available for the case study was primarily access to key personnel and the police report. Meetings with key stakeholders were conducted in connection with the problem understanding and data collection. An important limitation was that the organization told us that they wanted to largely ignore technical issues, like for example how to avoid DDoS and type of DDoS. The case study was also conducted under time and resource constraints and was conducted to see how the RCA method perform under these conditions. This case study was completed within approximately 150 hours.

A. Problem understanding - Multiple tools

The police report of the attack was an important contribution to the problem understanding and facilitated modelling the problem in a *Swimlane Flowchart*. The model is intended to show the flow through performances and events. The incident involved three stakeholders: The attacker, the website host, and the organization. The incident spanned over two days, following is a description of the timeline and elements in Fig. 10:

7th of May 2015

- 14:30 - Organization is notified via external service that Organization's web pages are unavailable.
- 15:45 - Organization contacts their service provider and is informed that they are working on the matter.
- 16:16 - Organization is contacted by its service provider, who informs that it is a denial of service attack (DDoS)
- 19:50 - Organization is contacted by their service provider explaining that Organization's web pages continuously receive between 40 and 60000 requests from foreign IP addresses and fails due to overload. There was an attempt to block foreign traffic, but due to the challenges of the service provider's network provider, this did not make it possible. It was then attempted to change the IP address of Organization 'web server and update the DNS of the domain Organization.no. This worked for 15-20 minutes. Afternoon/Evening - Organization informs National CERT (NorCERT) and the related security operations centre about the denial of service attack. NorCERT also gets the logs of the attack.

8th of May 2015

- 08:00 - All Organization websites are still unavailable.
- 08:10 - All web pages are available again
- 10:15 - The attack starts again, with the result that all the pages again becomes unavailable

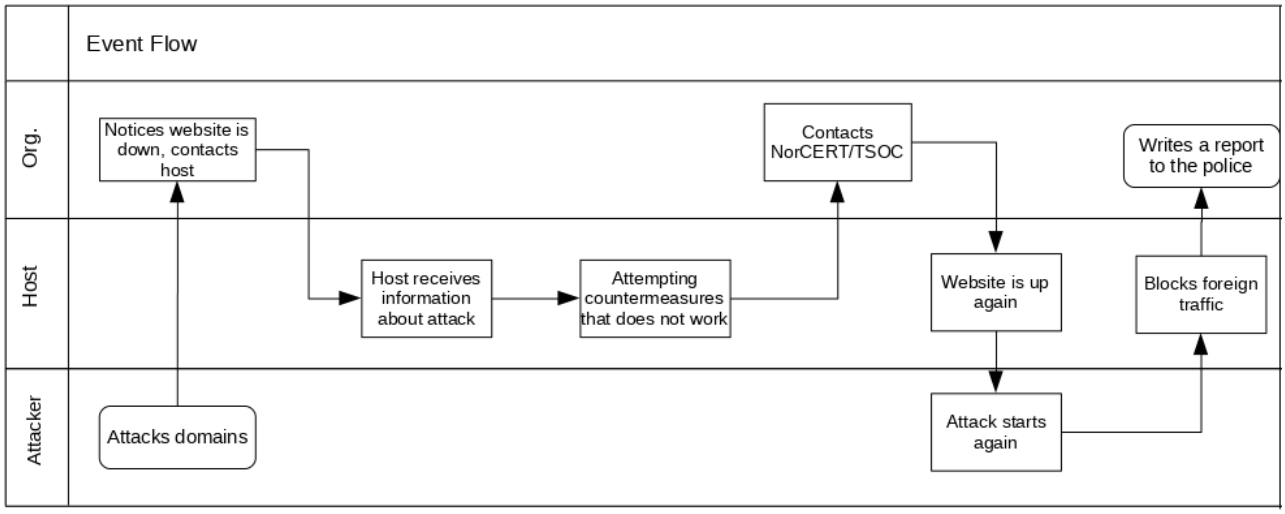


Fig. 10: Flowchart for DDoS case study.

- 12:24 - Service Provider informs that they are now blocking traffic from abroad, and web pages are gradually available for Norwegian and Scandinavian visitors. 12:30 - The local police is contacted for assistance in the case.

Performance Matrix

Performance Matrix was used to find out the most important priorities for the organization and the current performance within these areas. The priorities were identified in co-operation with key stakeholders. The stakeholders said that the most problematic about their site being down was that people could not access the self-help sites for identity theft mitigation. At the time, the organization had no sufficient countermeasure in place against DDoS. They could not quite answer how many times they were exposed to DDoS in a year, because they lacked the overview.

We were told that there was a high number of inquiries to the main website. Another problem was that it could also be difficult to respond to each inquiry within an acceptable time frame and provide sufficient help. Their self-help page and also website uptime were highly ranked within the performance matrix, as it is an organizational objective from them that the self-help site should serve and solve the majority of the inquiries. For the performance matrix we investigated the importance and performance of four areas: (i) Ability to help together with uptime and capacity for managing requests, (ii) Customer contact possibilities, (iii) Response Time, and (iv) Availability of the Self-help page. Fig. 11 shows the performance matrix for the four areas rated in co-operation with the expert.

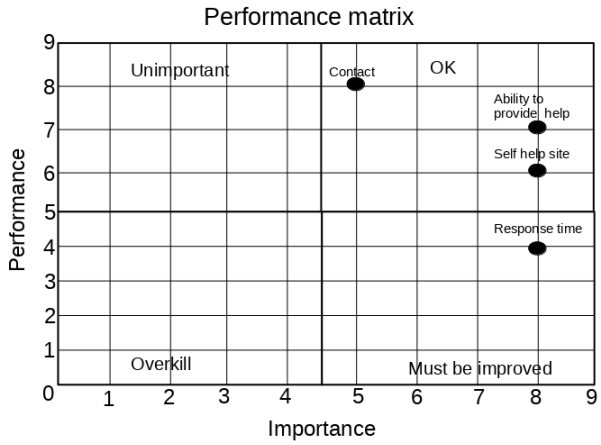


Fig. 11: Performance matrix for DDoS Case Study.

B. Problem cause brainstorming

Here we used unstructured brainstorming to obtain an overview of and consensus on what is being seen as problem causes. We developed a list of expected consequences and causes of the problem or the problems that build up the visible symptoms. The following list is grouped by importance, starting with identified possible consequences of the problem:

1) No accessibility for clients/users of the service.
2) Reduced reputation.
3) If the organization is unable to drive self-help, they will experience increased queues in other channels which they are not staffed to manage.
4) Possible financial problems.

Possible causes of the problem:

1) Service Provider had no tested plan for handling the situation.
2) Not enough knowledge and training in the organization for handling the incident.

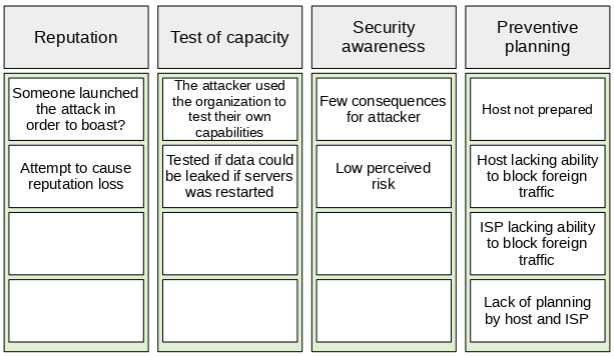


Fig. 12: Affinity diagram for the DDoS case study.

3) The organization is a target because it puts itself in a cross fire between individuals' problems and cyber crime attempting to make money.
4) The attackers can be outside Norway, which limits the jurisdiction and prosecution for domestic authorities.
5) No existing DDoS deterrence.
6) DDoS of the website was not a prioritized risk.
7) Missing/insufficient risk assessment.
8) Insufficient resources spent on server computing power, throughput, and bandwidth.

C. Problem cause data collection

Due to the case constraints and since we already had collected data on the problem, we chose to apply the check sheet approach for systematizing the problem cause data. With the Check Sheet tool we examined how the situation was while under attack and how the situation was afterwards. The desired dividend is to achieve a priority or a ranking of the items to be analyzed.

The points generated in the brainstorming phase were discussed with our contact person at the organization. Each item in the causes and consequences list were discussed and ranked. No graphical models of the check sheet was produced for the case.

D. Problem cause data analysis

Very few of the available RCA tools suited the case study. On the other hand, it was feasible to look for hidden contexts in the data collected. An Affinity Diagram 12 was attempted even though the data was not numeric.

The desire is to look for hidden contexts in the data collected. Following is qualitative assessment of the causes from the data collection phase:

The police did not look at the evidence: The organization collected the available evidence in the form of logs, and these were encrypted and password protected. The organization then reported the attack to the police and sent the encrypted evidence file and wrote that the police could contact them to receive the password for the file. The police never contacted them to obtain the password and dropped the case. Which means that there is a low probability of the attacker experiencing any consequences.

Contact with service provider: The website provider had no procedures on how to handle the situation, and contacted its network provider who could not immediately assist. The time it took to establish contact was relatively short. Slettmeg.no experiences it as an important aspect of this situation that contact time is short and that they themselves know that there are problems with the services they are delivering.

Ad-hoc situation handling: The provider of the web services was not trained in such situations and could not handle it when the attack occurred. Therefore, they contacted their network provider and suggested that they block the traffic from foreign addresses when they discovered that the attack did not originate from Norway. The network provider was also unable to immediately respond to this request. The organization itself had not completed any exercises on DDoS situations, so the incident was handled ad-hoc.

Host unable to block foreign traffic immediately: The website host did not have any mechanisms in place to shutdown network traffic from abroad. Thus, they had to contact the ISP they used, which also had no solution in place. Thus, it took unnecessary time for network traffic from abroad to be closed.

Wide-spanning vulnerability: The crash was aimed at the domain and not the IP address, however, the traffic took down all web services provided by the organization.

Unnecessary resources spent on handling the incident: The attack also led to people having to prioritizing to handle the situation leading to production loss. This situation occurred both at the web host and the ISP.

E. Root cause identification

Due to the restrictions of the case study, no visualization tools were applied to this phase. For this case, we identified 5 primary root causes

1. The attacker's motivation and intention: The attacker is motivated to perform an attack for several different reasons. Some of the reasons may occur from the work on mitigating the consequences of identity theft. There may also exist motivations that are not due to the organization's primary objectives but may be motivated by the attacker attempting to gain recognition. Bragging also deal with hacktivists that is motivated by publicity and fame. There may be a prestige in taking down a website that is managed by a security organization. DDoS is an easy to implement attack and with the right measures it is difficult to reveal the attacker.

2. Low perceived risk: It is costly to track down a moderately skilled attacker on the Internet. This may contribute towards the attacker thinking that there is a low probability of detection and therefore, there are no effective deterrents.

3. Easy to implement: A DDoS attack is easy to conduct. Even with little knowledge, there are standardized tools available for the task, some for free and others for sale. Amplification attacks also contribute contribute to an uneven distribution of power between attacker and defender, as the vulnerable protocols are easy to exploit.

4. Lack of preparation: Neither the organization, website host, or the ISP was prepared to manage the DDoS incident.

5. Lack of security management: The situation was poorly managed and the organization delegated responsibility for the situation without verifying that the host was able to handle such situations.

F. Problem elimination

Systematic inventive thinking is an approach to eliminate the problem. Since the case study had time constraints, the root cause we proposed to address was primarily Lack of security management. Furthermore, we listed components for the problem and according to tool description we took suggestions on components even though they could seem irrelevant. The areas we proposed to improve was:

1) Responsibility and chain of command
2) Security procedures for handling the problem
3) The contract and service level agreement
4) Incident handling cooperation and communication with the supplier
5) Knowledge and experience building of the provider.

These problem-solving components were chosen according to the 5 SIT principles. However, several of the principles turned out to be unworkable on the component, and in our case we chose to leave them blank. During the implementation of an SIT principle, we have described a proposal for improvement based on the purpose of the principle. Following are SIT analysis examples of the three first countermeasures:

No. 1 Responsibility and chain of command.
Component control: Ensure that the responsible person is linked to environments with professional knowledge.

No. 2 is Security procedures:
Attribute dependency: Impose greater control on the purchased services.
Component control: Compare own procedures with best practices.
Procedure: If one compares their procedures with similar organizations and best practices, one can discover weaknesses in their own and get new ideas on how problems can be solved. By carefully examining the service provider, an attempt can be made to reduce possible problem situations in the future.

No. 3 is Contract:
Attribute dependency: The contract should clearly describe the supplier's responsibilities.
Component control: Make sure the contract is equivalent to the environment.
Procedure: If the contract had held the service provider responsible, it could have been possible for the organisation to receive compensation for lost working hours caused by reorganization of work tasks during and after the attack. Contracts can also specify that the provider must have knowledge of how such situations should be treated.

G. Solution implementation

We have presented suggestions for improvement on procedure and contract. Furthermore, it is necessary to determine how the implementation is to be organized. The solution implementation tools available to help explain how the organization should be. Then there is the question of whether a tool is needed to guide, organize and structure the implementation. If the implementation is large or unintentional, it is recommended to use a Tree Diagram. We see from previous analyzes that the three chart has a structured review, as shown in the access card case for access cards and DDoS case. When it is appropriate to make comparisons with other organizations, a Spider Chart can be used.

TABLE VII. TOTAL HOURS SPENT CONDUCTING THE PRIMARY RCA FOR AN UNTRAINED THREE MAN TEAM (APPROXIMATELY 220 HOURS PER TEAM MEMBER)

Step	**Phase**	**Tasks**	**Time spent**
Preliminary	Preparations	Collecting available data	100 hours
Preliminary	Preparations	Testing and choosing tools	72 hours
1	Problem Understanding	Performance Matrix	3 hours
2	Problem cause brainstorming	Brainstorming	1 hours
3	Problem cause Data Collection	Planning interviews	150 hours
3	Problem cause Data collection	Conducting interviews	100 hours
4	Data analysis	Qualitative & Statistical	220 hours
5	Root cause identification	Fishbone	7 hours
6	Root cause elimination	SIT	7 hours
			Total 660 h.
		Only RCA Process	Total 488 h.

H. Assessment of RCA in situations with limited resources and time

Having a limited amount of time and resources on the analysis of the DDoS attack was very demanding. Two analysts completed the case within two weeks ($\sim$ 150 hours of effective work). In this case, the project team would have benefited from more contributors, for example, by identifying more potential problems during the brainstorming phase. Additionally, more project members would have provided a stronger quality control of the RCA process in the early phases. Due to time constraints, the tool selection and model development had less emphasis as the pressure was to deliver results within the time frame. However, going through the RCA process did produce results and insight into the problem. The RCA tools do force a structure onto a complex issue, which makes it more comprehensible. Our results shows that carrying out a RCA can provide a better understanding of the situation even with limited resources. We came up with suggestions for changes that the organization had not considered following the incident, providing evidence that the RCA process does have utility for InfoSec issues in more time constrained environments as well. However, the results would have improved with more time and resources, and more of both would be needed to complete a case with increased complexity and scope of the problem.

VIII. Discussion

This section discusses the cost/benefit of RCA, then evaluates the RCA tools for InfoSec application, and lastly, outlines the limitations and proposals for future work within the field.

A. Cost-benefit analysis

For cost-benefit analysis, we consider time spent on tasks and usefulness of the task. Table VII shows cost in time for

our team from conducting the primary case study. The reported hours are the total amount from start to end without having a budget constraint. The reported hours does contain resources spent beyond the three-man team, e.g., from interview attendance and supervision. Case studies 2 and 3 were conducted within approximately 150 hours per assessment, but without a concrete distribution of hours per task. Because of this, they are left out of the cost benefit discussion.

The most time consuming and crucial tasks were the steps 3 and 4, data collection and analysis. Further, the table shows that the resource demand for the Root cause identification and elimination phases as low, this is because the team primarily identified the root causes during the data analysis. While the main task of the root cause identification phase was to formalize the causes and effects, and the elimination was used to propose treatments.

As the team gain experience with using RCA on cases, the time estimate should be significantly be reduced. For example, our study spent 172 hours in the preparation phases gathering data on the problem and testing tools. With more experience, the preliminary steps will be significantly shortened. Our team also estimated that the whole process itself would become leaner with practice.

To summarize, we derived the primary benefit from the problem cause data collection and analysis phases, which enabled the root cause identification. Furthermore, the group benefited from working on the performance matrix, which set the direction for the remainder of the project. Regarding the remaining tools, the benefits the problem cause brainstorming was that it helped to provide an overview of the problem space and invited creative thinking. The advantage of the Fishbone tool was to group and visualize the identified problems in the context. Further, the process step contributed to determine and analyze causes. The SIT tool has a series of five principles that attempts to discover how to solve the components of the root cause. This tool offers a well-structured way to traverse a problem situation but could be resource intensive when handling many problems with all their components.

Issues of minor importance should not be subject to such an extensive effort as RCA requires. During the preparations for this study, we ran RCA for minor issues and found it not worthwhile as it was unproductive to use a complicated problem-solving process to less costly problems. However, future projects should consider RCA when they perceive the issue as important and do not know its nature or cause. The problem should be expensive, complicated, and cannot be addressed sufficiently with less comprehensive methods. These properties make conducting an RCA on the project justifiable and a valuable addition to the decision-making process.

B. Evaluation of the applied RCA tools

In this section, we evaluate the tools regarding expectation, application, and outcome. The cases are numbered as follows; the case about the access control for the Scandinavian R&D institution is *case one*, the tabletop exercise is *case two*, and the DDoS on the Security awareness website is *case three*. Table VIII shows an overview of the RCA tools applied on each case.

1) Performance Matrix: The performance matrix was applied in two cases. The purpose of this tool is to achieve a better understanding of the problem, prioritization of problem components, and to identify which part of the problem will reduce the largest amount of symptoms if removed. In the primary case study, we interviewed key personnel from the IT department at the institution based on the tool. The difference between what we estimated the answers to be and the responses we got was quite different, which shows how important it is to have key personnel partaking in the process of applying a Performance Matrix. Overall, this tool helped the group to gain a better understanding of the problem.

TABLE VIII. OVERVIEW OF RCA TOOLS USED IN THE CASE STUDIES.

RCA Phase	Tool name	Case 1	Case 2	Case 3
		Card Swap	Carbanak	DDoS incident
Problem Understanding	Performance Matrices	X		X
	Critical Incident		X	(X)
	Swimlane Flowchart		X	X
Problem Cause Brainstorming		X	X	X
Problem Cause Data Collection	Interviews	X		X
	Check Sheet			(X)
	Incident Data Analysis			
Problem Cause Data Analysis	Affinity Diagram	X	X	X
	Relationship Diagram		X	
Root Cause Identification	Fishbone Diagram	X		
	Five Whys		X	
Problem Elimination	Systematic Inventive Thinking	X		X
	Countermeasures Matrix		X	
Solution Implementation	Tree Diagram	X	X	

X = Applied
(X) = Tested, but experienced restrictions

In the DDoS case, it was essential for us to determine what was important for the website owner and how they felt that the functions they offer were working. We did experience that communication was critical, such as our ability to communicate what we were looking for. A note here is that more planning on how to teach the workings of performance matrices to the stakeholders would have made the process easier and quicker than we experienced it to be.

In both cases, we found that performance matrices were worth the effort as they are not time-consuming and they provide valuable insight into the problem.

2) Critical Incident: This tool was used in case two and three. Our expectations of the tool in these cases were that it would give a canonical and graphical display of the most frequent incidents. In both cases, we realized that it was not possible to generate actual numerical frequency labels. However, our experience in case two shows that it was possible to substitute these numerical frequencies with variables such as ”low, medium, and high,” as long as a description of what these ranges are. In case three, this issue was solved by creating questions that we gave them, and they ranked them according to what they found the most problematic. Our experience shows that it was difficult to acquire numerical frequencies of InfoSec incidents or other events, which rendered the critical incident tool having low utility. We managed a workaround using subjective values but quantified numbers are less prone to biases, and an approach for quantifying InfoSec incidents is proposed in the future work section.

Under the premises for the case studies the critical incident tool still provided information regarding problem causes without numerical data. However, frequencies of incidents should be in place before using the tool.

3) Swimlane Flowchart: This tool was first used in the tabletop exercise and then in case three regarding the DDoS incident. When using swimlanes, we wanted to investigate the flow of actions and get a visual representation of the incident. The goal was to obtain a better understanding of the incident details and detect connections between elements, which otherwise would not be easy to spot. The tool resulted in a graphical representation of the links and relationships between events and summarizes the events and their occurrence. We found this tool useful for visualizing the problem flow, involved stakeholders and actions taken. Flowcharts have a low cost to produce and a high utility.

4) Problem cause Brainstorming: This tool was applied in all cases with the aim to generate a list of probable causes and unify the project members views as a foundation for the next steps. Further, we also wanted the brainstorming process to help us identify possible consequences from the problems brainstormed. The tool worked well in categorizing the issues as well as bringing forth information about how some problems may relate to others. The brainstorming together with the problem understanding sets the scope for the remainder of the RCA and is therefore crucial step in the process.

5) Interview: Interviews were used in the case 1 and 2, and we experienced it as a good way to obtain contact with stakeholders, establish a network, and collect useful data for the RCA process. The interviews had to be planned with due care and tailored for the interview subject. With case one from the R&D organization we experienced that doing interviews revealed the attitude on card lending between the employees. Additionally, interviewing the primary stakeholder in case 3 provided invaluable information and insight into the problem space. Interviews were very time consuming, but did also provide the most reward through insight into the problem.

6) Check Sheet: By using Check Sheet in case three we wanted to achieve a ranking on either a prioritization or ranking of problems that has occurred. We also wanted to gain experience on the usage of the tool and evaluate how the tool worked in the given situation. It was not possible to obtain the frequencies of the events, which we then had to solve by asking questions concerning the problems that we had listed. The check sheet also partly relies on incident or problem frequencies as some tools rely on them to work as intended.

The check sheet tool did not provide the information needed to continue the case study and had to be exchanged with interviews.

7) Incident data analysis: This tool was not applied in the case studies but is discussed under future work.

8) Affinity Diagram: In case one, the goal of the Affinity Diagram was to categorize the suggested solutions to the problem, and then research which category the interview objects was the most interested in. We addressed this task by using a number on each proposed solution and summarized the numbers in the top of each column.

The affinity diagram worked well in our case studies to categorize and sort the identified elements. For each case study, we identified multiple elements rendering a high problem complexity. The affinity diagram aided in categorizing these elements and reducing them to a manageable problem. In case three our goal using Affinity Diagram was to discover hidden relationships in the data. However, no hidden relationships were found.

We experienced that the tool was useful for categorizing elements and reducing complexity. But it did not reveal or correlate any hidden relationships between the causes of the problem. For overview purposes, the tool has high utility and low cost. However, the utility is more uncertain when it comes to revealing hidden relationships.

9) Relationship Diagram: We applied the Relationship Diagram in case two where we aimed to see relationships between elements in the diagram and how they affect each other. We did not find any previously undiscovered relationships, and the tool did not have utility for advancing the RCA. However, the tool might be helpful in communication settings and is has a low cost time-wise to implement.

10) Fishbone Diagram: Fishbone Diagram was used on solving case one in the root cause identification stage. The diagram displays the causes leading up to the card lending problem for then to use this information to uncover the root causes. We experienced that it was difficult to generate the categories and the elements in the diagram. However, as Fig. 5 shows, this is one of the highest utility tools in the RCA toolbox. It visualizes the problem space and the contributing causes in a comprehensive way, which also aids in stakeholder communication. The time spent using the tool and making the diagram was worth the time, and the cost will diminish with more practice.

11) Five Whys: The RCA tool Five Whys was used in case two in the root cause identification phase. The tool itself has a low cost. However, the completeness of the process is questionable as there might be more causes than five. It would have been preferable that the tool opens up for more possibilities. A modification could be to run it in more than one iteration to see if more possible causes to the problem could be generated. The tool is easy to understand and to implement, with a time-wise low cost.

12) Systematic Inventive Thinking (SIT): SIT is designed to find the problem-causes where solutions could be applied to eliminate the occurrence of the problems overall. We experienced that the tool worked well for its purpose, but the cost was high to complete the process, and it was time-consuming to deal with all the small components as well as it was error-prone. We expect the amount of work needed to apply this tool will vary depending on the task size. Designate enough time to this tool and try to have as much overview of the problem and its environment before embarking on it. In case three, the SIT helped us discover components we earlier did not notice, so, it has the tool has utility.

With smaller problems SIT can be useful, however, the amount the work grows proportionally with the size of the problem. The utility of running this tool is high as it provides insight into the problem and strategies to eliminate it.

13) Countermeasure Matrix: Countermeasure Matrix was used in case two to determine which countermeasures would best solve the problem. The tool takes into account risks and

costs associated with applying the solutions. We found that one of the tool's limitations was that it was not able to take into consideration the consequence a countermeasure could present. Meaning that implementing a control can solve the problem, but also likely introduces a new risk or problem into the system. The countermeasures matrix is a useful tool for sorting problem treatments and ranking them according to estimated efficiency and feasibility. However, the tool could also benefit from estimating treatment cost.

14)Tree Diagram: In case one, the Tree Diagram was used to present a structured plan for implementation of solutions found while using SIT to the card lending problem. The tool was able to display the order of the steps to be taken as well as what category the action belongs. While applying this tool, key personnel should be actively partaking in the process. In case two, the goal with the tool was to generate a structure of the solution implementation tasks and to visualize the links between these tasks and their respective activity. Tasks are represented by leaves and activities are represented by the root and the branches. Since all the cases were limited to proposing solutions and not implementations, we do no estimate the utility of the tool.

C. Limitations & Future Work

The case study presented in this article is specific to the organization and culture; thus our results have limited generalizability, but the RCA method and results provide an insight into what to expect from the process. Another aspect is that our RCA team was inexperienced and other more experienced teams will run the process more efficiently with a better cost-benefit.

An important limitation for this study was that we limited the tool selection to the method proposed by Andersen and Fagerhaug [2]. We did this to limit the complexity of the process and tool selection. Future studies may wish to include tools from other RCA methodologies and frameworks. We found interviews have the highest value in the data collection process. Similarly, questionnaires were not included in our cases, but they have the potential for reaching a broader audience and can also contribute to the RCA process.

Another issue is if a similar insight could have been gained if we delegated a similar amount of resources into the ISRA to investigate the problem. It is possible that the results of the ISRA would have overlapped more with the RCA with more time and resources spent on the former. However, the ISRA process does not argue for such a deep dive into the problem as the RCA process and does not provide tools for doing so. It is therefore unlikely that a more thorough ISRA process would have produced a similar result. However, the incentive for such an investigation was not there, and we perceive the ISRA methodologies as immature in this area [12]. Instead of considering the RCA as an extension of the ISRA, a possible path for future work is to conduct case studies where the researchers invest a similar amount of resources into both the RCA and ISRA and then compare results.

An additional direction for future work is to apply RCA to more and diverse case studies to get a better understanding of the contributions and limitations of the approach for InfoSec. Recent work has also proposed a novel approach for conducting socio-technical security analysis [11], and a path for future work is to adapt, develop, and improve RCA tools for InfoSec. Furthermore, the future efforts could research RCA efficiency through automation of tasks and build knowledge repositories. Regarding the latter, a repository of tools for data collection would help streamline step 3 in the RCA process.

D. A proposal for RCA of InfoSec incidents

Andersen and Fagerhaug [2] proposes the use of incident data analysis for use in RCA. An InfoSec incident is in short a violation of the integrity, availability, or confidentiality of information assets or resources that fall under the security constituency. If logged properly, incident data documents a security incident from its detection until it is solved, including measures taken by the incident handler to solve it. Thus, incident data is a reliable and important source both for RCA and risk analysis. However, we have conducted preliminary research into utilizing RCA for InfoSec incident data and encountered several challenges that must be solved for the data to readily lend itself to RCA:

1) *No two incidents are the same.* Both incident frequencies and risk quantification requires incidents to be counted. So, to determine whether an incident is re-occurring or not, we need to be able to quantify incidents. However, in our preliminary work we found that no two incidents are identical. For example, we might be facing two incidents that are caused by compromised accounts, but the incidents do not involve the same account, and the initial compromise and malicious actions are likely different. The root-cause might be vulnerable account security, but we need a framework to classify and quantify to determine the frequencies of re-occurring incidents. There are already taxonomies of computer security incidents [23], [24] that provide a nice starting point, but as the threat landscape changes an update to these are needed and a higher granularity is also desirable for incident analysis.
2) *A security incident has at least one cause and one malicious action.* Trying to analyze a security incident one quickly reaches the conclusion that there are at least two parts of the incident that must be quantified. There is both an observable cause and an observable outcome of an incident, where the latter is often what is detected and triggers the incident. An example of a typical incident:
   "Incident topic: Compromised user
   User XX is sending spam email internally to employees. The email contains a suspicious link to a foreign IP address."
   In this case, the cause of the incident would classify as a compromised user account, while the observable outcome and malicious action is sending spam email. Typically, the paper trail of an incident consists of the original incident report or trigger, which varies quite a lot depending on how the incident was detected. Further, the incident handler logs each step he takes to solve the incident and all correspondence with affected parties. In short, all correspondence, analysis, follow ups, and treatments are present in the logs. Both the cause and the outcome of the incidents should be observable from the incident data, so, a

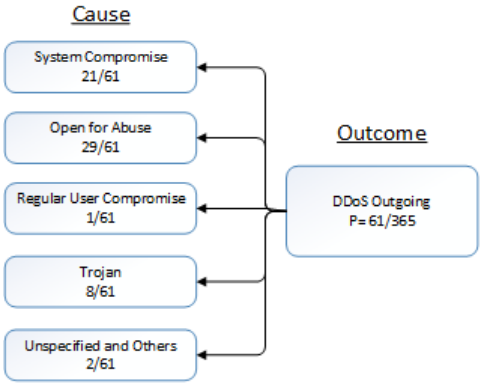


Fig. 13: Distributions of causes for DDoS Outgoing

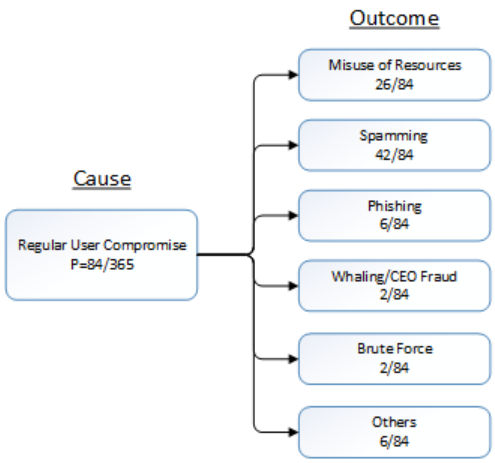


Fig. 14: Distributions of User Compromise outcomes

comprehensive classification scheme should aim to classify both. However, our research into state-of-the-art frameworks suggests that no such classification scheme exists.

3) *Organization maturity.* The organization must be sufficiently mature to have a developed and repeatable process for handling and documenting incidents. A non-standard process will generate a large variety of incident logs, which will not easily lend itself to incident quantification.

In our preliminary analysis, we have tried to determine the root causes of outgoing DDoS attacks for an organization, where an attacker abuses vulnerable systems through for example amplification attacks. In Fig. 13, we have applied a preliminary incident classification framework and attempted to classify all the the causes leading to the 61 security incidents.

Further, by classifying both the cause and the outcome we can also analyze the outcomes of a particular incident cause. This approach is useful for determining attacker motivation once he has exploited a vulnerability and gotten a foot-hold inside the network. In Fig. 14, we have classified 84 incidents as ”Regular User Compromise” and mapped the malicious actions of each incident with frequency distributions.

The incident data does show great promise as an addition to the InfoSec management and resource allocation. From a management perspective, the causes can be addressed by likelihood reducing measures, while the outcomes can be addressed with consequence reducing measures. However, there are some challenges that need to be overcome in order to adapt RCA into incident analysis. We have conducted some preliminary research into the topic, but more research is needed particularly into framework development.

IX. Conclusion

This study has applied RCA tools to propose a solution to a complex socio-technical InfoSec problem and found the RCA method a valid but costly extension to the ISRA. Running a full-scale RCA requires a lot of time and resources and the problem should be expensive enough to justify the RCA. The results from the RCA overlapped slightly with the initial ISRA. The main differences were that the RCA team proposed administrative treatments aimed at solving problems in the social domain, while the ISRA produced a more technical analysis and treatment of the problem. We conclude that practitioners should look at these two approaches as complimentary for dealing with complex socio-technical risks and problems. The combination of the ISRA and RCA will also have utility when planning for defense-in-depth, where administrative and technical risk controls can work in coherence to mitigate threats.

This study found that the RCA process does lend itself to the constrictions of a tabletop exercise for training purposes. RCA did not reveal any additional root causes. The group has to manage the limitations of not having access new information for solving the case. So, RCA has utility for exercise and experimenting with the tools on different types of data, but it is unlikely to provide any additional knowledge.

Applying the RCA under the time and resource-restricted setting did generate valuable insight into the root causes of the problem. For the case study of the DDoS attack, the process revealed multiple causes that were previously undetected by the principal. Several of these causes were in the socio-technical domain, and are not likely to be found using typical InfoSec analysis approaches. Therefore, we conclude the RCA process worked well under resource and time restricted setting.

Several RCA tools proved useful for addressing til InfoSec problems, with an overarching process tailored for problem-solving. Examples of tools that worked well for our case-studies for problem understanding was performance matrices and swimlane flowcharts. For data collection, interviews had the highest utility. We found the affinity diagram to have the highest cost-benefit in the problem cause data analysis phase. One of the best tools in the RCA process for visualizing several existing causes in the problem and communicating was the fishbone diagram. Although SIT has some drawbacks regarding problem scaling, it worked well to provide solutions to identified root causes.

The main drawback of RCA was that our cost-benefit analysis of the time and resources invested in case one is on the borderline of being justifiable, and the cost of the problem should be considered before launching a RCA. However, RCA performed well under time and resource constraints for a less

complex problem. Thus, the full-scale RCA is a viable option when dealing with both complex and costly InfoSec problems. For minor issues, a RCA may be excessive or should at least be strictly time managed. Based on our findings we conclude that RCA should be a part of the InfoSec management toolbox.

ACKNOWLEDGEMENTS

The authors acknowledge the help and support from Erlend Brækken, Professor Einar Snekkenes, Christoffer Hallstensen, and Stian Husemoen. We also extend our gratitude to all the participants in our study and to the anonymous reviewers.

REFERENCES

[1] G. Wangen, N. Hellesen, H. M. N. Torres, and E. L. Brækken, "An empirical study of root-cause analysis in information security management," in *SECURWARE*, vol. 2017. IARIA, 2017, pp. 26–33.

[2] B. Andersen and T. Fagerhaug, *Root cause analysis: simplified tools and techniques*. ASQ Quality Press, 2006.

[3] M. F. Peerally, S. Carr, J. Waring, and M. Dixon-Woods, "The problem with root cause analysis," *BMJ Qual Saf*, vol. 26, no. 5, pp. 417–422, 2017.

[4] H. M. N. Torres, N. Hellesen, and E. L. Brækken, "Bruk av rotårsaksanalyse i informasjonssikkerhet," B.S. thesis, NTNU in Gjøvik, 2016.

[5] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM transactions on information and system security (TISSEC)*, vol. 6, no. 4, pp. 443–471, 2003.

[6] P. F. Wilson, *Root cause analysis: A tool for total quality management*. ASQ Quality Press, 1993.

[7] A. M. Doggett, "Root cause analysis: a framework for tool selection," *The Quality Management Journal*, vol. 12, no. 4, p. 34, 2005.

[8] J. Collmann and T. Cooper, "Breaching the security of the kaiser permanente internet patient portal: the organizational foundations of information security," *Journal of the American Medical Informatics Association*, vol. 14, no. 2, pp. 239–243, 2007.

[9] G. Wangen, "Conflicting incentives risk analysis: A case study of the normative peer review process," *Administrative Sciences*, vol. 5, no. 3, p. 125, 2015. [Online]. Available: http://www.mdpi.com/2076-3387/5/3/125

[10] A. Abubakar, P. B. Zadeh, H. Janicke, and R. Howley, "Root cause analysis (rca) as a preliminary tool into the investigation of identity theft," in *Cyber Security And Protection Of Digital Services (Cyber Security), 2016 International Conference On*. IEEE, 2016, pp. 1–5.

[11] J.-L. Huynen and G. Lenzini, "From situation awareness to action: An information security management toolkit for socio-technical security retrospective and prospective analysis," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy*, 2017, pp. 213 – 224.

[12] G. Wangen, C. Hallstensen, and E. Snekkenes, "A framework for estimating information security risk assessment method completeness," *International Journal of Information Security*, Jun 2017. [Online]. Available: http://dx.doi.org/10.1007/s10207-017-0382-0

[13] J. Freund and J. Jones, *Measuring and Managing Information Risk: A FAIR Approach*. Butterworth-Heinemann, 2014.

[14] B. Schneier, "Attack trees," *Dr. Dobbs journal*, vol. 24, no. 12, pp. 21–29, 1999.

[15] *Information technology, Security techniques, Information Security Risk Management*, International Organization for Standardization Std., ISO/IEC 27005:2011.

[16] G. Wangen and E. Snekkenes, "A taxonomy of challenges in information security risk management," in *Proceeding of Norwegian Information Security Conference - NISK 2013 - Stavanger*, vol. 2013. Akademika forlag, 2013, pp. 76–87.

[17] P. Shedden, W. Smith, and A. Ahmad, "Information security risk assessment: towards a business practice perspective," in *Australian Information Security Management Conference*. School of Computer and Information Science, Edith Cowan University, Perth, Western Australia, 2010, pp. 119–130.

[18] G. Wangen, A. Shalaginov, and C. Hallstensen, "Cyber security risk assessment of a ddos attack," in *International Conference on Information Security*. Springer, 2016, pp. 183–202.

[19] F. den Braber, I. Hogganvik, M. S. Lund, K. Stølen, and F. Vraalsen, "Model-based security analysis in seven steps - a guided tour to the coras method," *BT Technology Journal*, vol. 25, no. 1, pp. 101–117, 2007.

[20] Kaspersky, "Carbanak apt the great bank robbery," 2015. [Online]. Available: https://securelist.com/files/2015/02/Carbanak_APT_eng.pdf

[21] "Anunak: Apt against financial institutions," Group-Ib, Fox-It. [Online]. Available: https://www.group-ib.com/resources/threat-research/Anunak_APT_against_financial_institutions.pdf

[22] J. van der Wiel, "How did the carbanak cybergang steal $1 billion from banks? (1/3)," accessed 22-Feb-2016. [Online]. Available: https://www.youtube.com/watch?v=csc9VDuHBNU

[23] M. Kjaerland, "A taxonomy and comparison of computer security incidents from the commercial and government sectors," *Computers & Security*, vol. 25, no. 7, pp. 522–538, 2006.

[24] S. Hansman and R. Hunt, "A taxonomy of network and computer attacks," *Computers & Security*, vol. 24, no. 1, pp. 31 – 43, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404804001804

Optimal Security Protection for Sensitive Data

George O. M. Yee
Computer Research Lab, Aptusinnova Inc., Ottawa, Canada
Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
email: george@aptusinnova.com, gmyee@sce.carleton.ca

Abstract—**The growth of the Internet has unfortunately been accompanied by an increasing number of attacks against an organization's computing infrastructure, leading to the theft of sensitive data. In response to such incursions, the organization installs security measures (e.g., intrusion detection system) for protecting its sensitive data. However, this installation is often done haphazardly, without any objective guidance regarding how many vulnerabilities must be secured in order to achieve an acceptable level of protection. This paper shows how an organization can calculate estimates of security protection, and objectively use them to adjust the number of security measures installed, until an optimal level of protection is achieved, subject to certain constraints. This work extends the paper "Assessing Security Protection for Sensitive Data" published in SECURWARE 2017. Additional explanations, application examples, and related works have been included.**

Keywords-optimal; security protection; assessment; sensitive data; vulnerability.

I. INTRODUCTION

This work extends Yee [1] by adding explanations, application examples, and related works.

Recent attacks against computing infrastructure, resulting in the theft of sensitive data, have grabbed the headlines, and have devastated the victim organizations. The losses have not only been financial (e.g., theft of credit card information), but more importantly the damage to the organization's reputation. Consider the following data breaches that happened in 2016 [2] and 2017 [3]:

- February, 2016, University of Central Florida: Data breach affected approximately 63,000 current and former students, faculty, and staff, with the theft of information including social security numbers, first and last names, and student/employee ID numbers.
- February, 2016, U.S. Department of Justice: Hackers released data on 10,000 Department of Homeland Security employees one day, and the next day released data on 20,000 FBI employees. Stolen information included names, titles, phone numbers, and email addresses.
- March, 2016, Premier Healthcare: Theft of a laptop containing sensitive data pertaining to more than 200,000 patients, including names, dates of birth, and possibly social security numbers or financial information.
- March, 2016, Verizon Enterprise Solutions: Hackers stole information for about 1.5 million customers; the information was found for sale in an underground cybercrime forum by cyber security journalist Brain Krebs.
- September, 2016, Yahoo!: The company announced that a hacker had stolen information from 500 million accounts in 2014. The hacker, believed to be working for a foreign government, stole email addresses, passwords, full user names, dates of birth, telephone numbers, and in some cases, security questions and answers.
- February and April, 2017, InterContinental Hotels Group (IHG): The company that owns popular hotel chains like Crowne Plaza, Holiday Inn, and Kimpton Hotels, announced in February a data breach that affected 12 of its properties. This number was enlarged to 1,200 properties in April. Malware was found on payment processing servers. The stolen data included cardholder names, card numbers, expiration dates, and internal verification codes.
- March, 2017, Dun & Bradstreet: This business services company found its marketing database with over 33 million corporate contacts shared across the web. The company claimed that the breach occurred to businesses, numbering in the thousands, that had bought its 52 GB database. The leak may have included full names, work email addresses, phone numbers, and other business-related data from millions of employees of organizations such as the US Department of Defense, the US Postal Service, AT&T, Walmart, and CVS Health.
- July, 2017, Verizon: 14 million Verizon subscribers may have been affected by a data breach simply by having contacted Verizon customer service in the past 6 months. The customer service records were kept on a server controlled by Israel based Nice Systems. The leaked data consisted of log files generated when Verizon customers contacted the company by phone.
- September, 2017, Equifax: This is one of the three largest credit agencies in the US. It announced a breach that may have affected 143 million customers, one of the worst breaches ever due to the sensitivity of the data stolen. The compromised data included social security numbers, driver's license numbers, full names, addresses, birth dates, credit card numbers, and other personal information. Hackers had access to the company's system from mid-May to July by exploiting

a vulnerability in website software. Equifax discovered the breach on July 29, 2017.

- November, 2017, Uber: Uber revealed that it became aware of a data breach in late 2016 that potentially exposed the personal information of 57 million Uber users and drivers. However, Uber chose to pay the hackers $100,000 to keep the breach a secret instead of immediately alerting the affected victims. The hackers gained access to the data stored on GitHub, which was used by Uber engineers for collaboration, and included names, email addresses, and phone numbers of Uber users worldwide.

This is only a sampling, as there were many more breaches in 2016 and 2017, and in fact, no year can be said to have been breach-free. Moreover, the problem appears to be getting worst, as 2017 has been mentioned [4] as a "record-breaking year for the numbers of publicly reported data breaches and exposed records in 2017 worldwide: a total of 5,207 breaches and 7.89 billion information records compromised."

To protect themselves from attacks, such as the ones described above, organizations determine their vulnerabilities to attack, and then secure the vulnerabilities with security measures. Common measures include firewalls, intrusion detection systems, two-factor authentication, encryption, and training for employees on identifying and resisting social engineering. However, today's organizations install security measures without any way of calculating the overall level of protection that will result. They proceed based on recommendations from consultants, in reaction to attacks that have been observed, or worst, as a result of having suffered an attack themselves. And in many cases, they are forced to stop this deployment once their security budget runs out. It would be far better if an organization can follow a top-down approach, by setting a target level of protection and then install security measures to achieve the target. The target would be set according to the expected threat situation, the nature of the business, the sensitivity of information kept, and an estimated financial budget. Before this can be done, it would be useful to have quantitative estimates of the level of protection based on the number of vulnerabilities secured. This work derives such estimates and shows how to apply them to not only set a protection target, but also how security measures can be installed to achieve the target.

The objectives of this work are i) derive estimates of the resultant protection level obtained by an organization through the installation of security measures to secure vulnerabilities, ii) show how these estimates can be calculated, iii) show how the estimates can be applied in a structured, objective, quantitative approach to secure an organization, and finally iv) illustrate ii) and iii) using examples.

The rest of this paper is organized as follows. Section II discusses the nature of sensitive data and derives the estimates. Section III explains how the estimates are calculated and applied in a structured, objective, quantitative approach to secure an organization. Additional application areas are also included. Section IV presents two application examples. Section V discusses related work. Finally, Section VI gives conclusions and future research.

II. Estimating Security Protection Levels

Before deriving estimates of security protection levels, it is useful to examine the nature of sensitive data.

A. Sensitive Data

We all have some sense of what is meant by sensitive data: first and foremost it is data that must be safeguarded from falling into the wrong hands, the consequence of which would be damaging to an individual or an organization.

For an individual, sensitive data usually means private information, which is information about the individual and is owned by that individual. The individual's privacy then refers to his or her ability to control the collection, purpose of collection, retention, and distribution of that information by another party. Private information is also called personal information or personally identifiable information because it can be used to identify the individual. For example, an individual's height, weight, or credit card number can all be used to identify the individual and are therefore considered as personal information. Continuing this example, the extent to which the individual has control over who collects this information, the purpose for which the collector will use this information, how long the collector will retain this information, and to which other parties the collector will disclose this information, determines the individual's degree of privacy. The nature of private information will not be explored further here but the reader is encouraged to consult [5] for more details.

For an organization, sensitive data may encompass private information, but may additionally include information that may compromise the competitiveness of the organization if divulged, such as trade secrets or proprietary algorithms and secret formulas. For government organizations, sensitive data may include information that is vital for the security of the country for which the government organization is responsible. For this work, sensitive data is defined as follows:

DEFINITION 1: *Sensitive data* is information that must be protected from unauthorized access in order to safeguard the privacy of an individual, the well being of an organization, or the well being of an entity for which the organization has responsibility.

This work considers losses arising from sensitive data or sensitive information being in the possession of unintended malicious parties or entities. This covers theft and any unintended exposure of sensitive information such as accidental leakage or posting. Per Definition 1, "sensitive data" and "sensitive information" are used interchangeably in this work. Some researchers make a distinction between these terms but the popular usage calls for no distinction.

A. Attacks on Organizations

Attacks carried out against sensitive information

residing with organizations may be categorized as "outside attacks" and "inside attacks". We define these as follows.

DEFINITION 2: An *attack* is any action carried out against sensitive information held by an organization that, if successful, results in that information being in the hands of the attacker. An *outside attack* (A_o) is an attack that is carried out by an outsider of the organization (i.e., the attacker is not associated with the organization in a way that gives her special access privileges to sensitive data, e.g., a regular member of the public). An *inside attack* (A_i) is an attack that is carried out by an insider of the organization (i.e., someone who has special access privileges to sensitive data by virtue of her association with the organization, e.g., employee).

DEFINITION 3: A *vulnerability* of an organization is any weakness in the organization's infrastructure, platform, or business processes that can be targeted by an attack with some expectation of success. A *secured-vulnerability* was originally a vulnerability that has had protective security measures put in place so that it is no longer a vulnerability. For example, a vulnerability is private information stored in the clear. This becomes a secured vulnerability if the private information is encrypted.

Outside attacks target a range of security vulnerabilities, from software systems that can be breached to access the sensitive information to simple theft of laptops and other devices used to store sensitive information. An example of an outside attack is the use of a Trojan horse planted inside the organization's computer system to steal sensitive information.

Inside attacks arise from the attacker making use of her privileged position (e.g., as an employee) to cause a loss of sensitive data. In this case, the attack is often difficult to detect, since it would appear as part of the normal duties of the insider attacker. An example of an inside attack is where a disgruntled employee secretly posts the organization's sensitive information on the Internet to try to harm the organization. An inside attack can also be unintentional (e.g., an employee casually providing client names for a survey).

Both outside and inside attacks target the organization's vulnerabilities. Vulnerabilities that invite outside attacks include the use of badly provisioned firewalls, the failure to encrypt data, and simple carelessness (e.g., leaving a laptop containing sensitive information in a car). Vulnerabilities that attract inside attacks include a) poor business processes that lack mechanisms to track which data is used where, used for what purpose, and accessed by whom, b) poor working conditions that give rise to employees feeling unfairly treated by management which can lead to employees seeking revenge, and c) poor education and enforcement of company policies regarding the proper care and handling of sensitive information (e.g., the above survey example).

The location of an attacker carrying out an attack does not determine whether the attack is an inside attack or an outside attack. An inside attack can be carried out outside the organization's premises; similarly, an outside attack can be carried out inside the premises.

We have so far used the expressions "level of protection" and "protection level" informally relying on their everyday meaning. We now formalize this meaning in terms of vulnerabilities, introducing the idea of "security protection level".

DEFINITION 4: An organization's security protection level (SPL) is the degree of security protection from attacks that results from the organization having secured q vulnerabilities, leaving p vulnerabilities unsecured, where the organization has a total of $p+q$ vulnerabilities. Each pair of values (p, q) corresponds to a different SPL.

Suppose an organization has a total of N vulnerabilities, where $p + q = N$. Then each organization has a value of N that corresponds to a set of SPL points lying on a straight line in the (p, q) plane, where the higher values of q correspond to higher or greater security protection levels. Figure 1 shows this relationship for two organizations having $N=50$ and $N=100$.

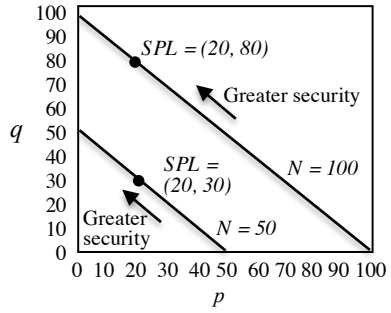


Figure 1. SPL points on lines corresponding to two organizations, one with N=50 and the other with N=100.

B. Deriving the Estimates

Intuitively, for the same organization, SPL A is more capable of protecting from sensitive information loss than SPL B if A is composed of more secured vulnerabilities than B, where all vulnerabilities have roughly the same level of loss risk. This is the idea behind the derivation below.

We seek the capability C of an organization's SPL to protect sensitive data. Suppose that an organization's SPL has p vulnerabilities and q secured-vulnerabilities, where no distinction is made between outside and inside attacks. The number of original vulnerabilities before any vulnerabilities were secured is $p+q$. Let $P(e)$ represent the probability of event e. For convenience, "data" is understood to be "sensitive data". We have

$$C = P(\text{no data losses}) = 1\text{-}P(\text{data losses}) \quad (1)$$

Since a data loss is the result of a successful attack on a vulnerability,

$$P(\text{data losses}) \approx p/(p+q) \quad (2)$$

where we have applied the additive rule for the union of probabilities of attacks on the p vulnerabilities, assuming that 2 or more attacks do not occur simultaneously. This is a

fair assumption confirmed by experience. Substituting (2) into (1) and adjusting for a possible zero denominator gives

$$C \approx 1-[p/(p+q)] = q/(p+q) \quad if\ p+q > 0 \qquad (3)$$
$$= 1 \quad if\ p+q = 0 \qquad (4)$$

Since C is a probability, its value is between 0 and 1, attaining 0 if the organization has no secured vulnerabilities ($q=0$, (3)) and 1 if either all of its vulnerabilities are secured ($p=0$, (3)) or if the organization has no vulnerabilities ($p+q=0$, (4)). Since an organization having no vulnerabilities is highly improbable, (4) is unlikely to apply.

The above derivation can be done within each of the categories of outside attacks and inside attacks (we did not distinguish between outside and inside attacks above). Let C_o, C_i represent the capabilities of an organization's SPL to protect sensitive information from outside attacks and inside attacks, respectively. Let p_o, p_i represent the number of vulnerabilities to outside attacks and inside attacks, respectively. Let q_o, q_i represent the number of secured vulnerabilities to outside attacks and inside attacks, respectively. Then, repeating the above derivation for outside attacks and inside attacks gives

$$C_o \approx q_o/(p_o+q_o) \quad if\ p_o+q_o > 0 \qquad (5)$$
$$\approx 1 \quad if\ p_o+q_o = 0 \qquad (6)$$
$$C_i \approx q_i/(p_i+q_i) \quad if\ p_i+q_i > 0 \qquad (7)$$
$$\approx 1 \quad if\ p_i+q_i = 0 \qquad (8)$$

As above, C_o (C_i) have values between 0 and 1, attaining 0 if the organization has no secured vulnerabilities to outside (inside) attacks ((5) and (7)) and 1 if either all of the vulnerabilities are secured ((5) and (7)) or if the organization has no vulnerabilities ((6) and (8)). Since an organization having no vulnerabilities to outside and inside attacks is highly improbable, (6) and (8) are unlikely to apply.

The estimates of data protection capability are now assigned as follows for a given SPL (no distinction between inside and outside attacks), SPL_o (for outside attacks), and SPL_i (for inside attacks). Let E be an estimate of data protection capability, where no distinction is made between outside and inside attacks. Let E_o be an estimate of data protection capability against outside attacks. Let E_i be an estimate of data protection capability against inside attacks. Then for the SPL, SPL_o, and SPL_i

$$E = q/(p+q) \quad if\ p+q > 0 \qquad (9)$$
$$= 1 \quad if\ p+q = 0 \qquad (10)$$
$$E_o = q_o/(p_o+q_o) \quad if\ p_o+q_o > 0 \qquad (11)$$
$$= 1 \quad if\ p_o+q_o = 0 \qquad (12)$$
$$E_i = q_i/(p_i+q_i) \quad if\ p_i+q_i > 0 \qquad (13)$$
$$= 1 \quad if\ p_i+q_i = 0 \qquad (14)$$

E has the advantage of providing a single number for ease of comparison between different SPLs within an organization. A threshold T for E may be pre-determined such that for E above T, the security measures installed by the organization to secure vulnerabilities against both outside and inside attacks (corresponding to a SPL) are deemed adequate. For given SPL_o and SPL_i, E_o *and* E_i have the advantage of focusing in separately on where an organization stands in terms of its security measures against outside and inside attacks. Thresholds T_o and T_i may be pre-determined for E_o and E_i respectively, such that for both estimates above their respective thresholds, the corresponding installed security measures against outside and inside attacks are deemed adequate. If this is the case, we call the corresponding SPL an *adequate SPL*. In practice, E_o and E_i may be expressed as percentages that define a region in a 100 x 100 plane in which an organization's capability to protect data is adequate (acceptable), as represented by the shaded region in Figure 2. Each point in this shaded region corresponds to an adequate SPL. An organization strives to have the "best" adequate SPL (one which has highest number of security measures possible against both outside and inside attacks) as allowed by its financial budget for adding security measures (see Section III).

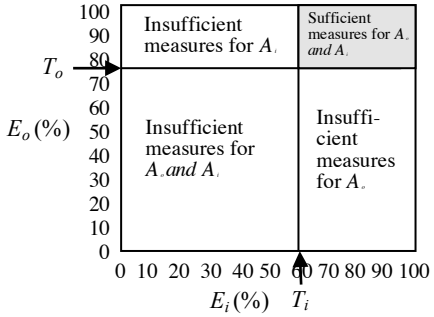


Figure 2. Sufficiency of Security Measures Against Outside Attacks (A_o) and Inside Attacks (A_i)

III. APPLYING THE ESTIMATES

This section shows how an organization may use the estimates to establish "best" adequate SDLs as permitted by its financial budget. The description below separates outside attacks from inside attacks since organizations would need to account for them separately.

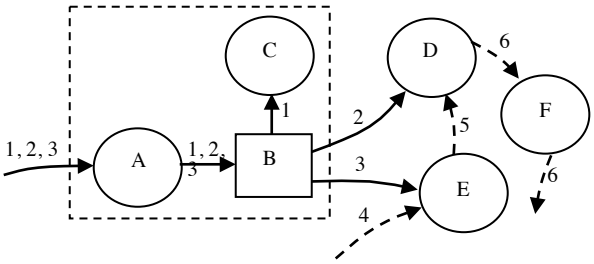
A. Determining the Vulnerabilities

For outside attacks, we recommend a threat analysis of security vulnerabilities in the organization's systems that could allow outside attacks to occur. Threat analysis or threat modeling is a method for systematically assessing and documenting the security risks associated with a system (Salter et al. [6]). Threat modeling involves understanding the adversary's goals in attacking the system based on the system's assets of interest. It is predicated on that fact that an adversary cannot attack a system without a way of supplying it with data or otherwise accessing it. In addition, an adversary will only attack a system if it has some assets of interest. The method of threat analysis given in [6] or any other method of threat analysis will yield $N_o = p_o + q_o$, which is the total number of vulnerabilities to outside attacks. The method presented here for threat modeling is based on [6], and consists of the following steps:

1. Identify threats: examine all available details of the system and enumerate possible threats.

2. Create attack trees for the system: for each threat, take the attacker's view and find the weak points in the system and the paths that can lead to realizing the threat.
3. Apply weights to the leaves: for each leaf, assign qualitative values for risk, access, and cost to the attacker.
4. Prune the tree so that only exploitable leaves remain: prune leaves that represent objectives that are beyond the attacker's capabilities or that offer an inadequate return.
5. Generate corresponding countermeasures: identify security measures for rendering the threat non-realizable.

As an illustration of the above method for threat analysis, consider the hypothetical software system of an online seller of merchandise (e.g., Amazon.com). Figure 3 shows the essential components of this system, using solid arrows to represent sensitive data flow, dashed arrows to depict non-sensitive data flow, circles to represent processing modules, squares to represent data storage, and a dashed square to enclose components that execute on the same computing platform. Additionally, data items are identified using numbers; all other components are identified with letters.



Legend:
A: receive and store data
B: database
C: print shipping label
D: pack item for shipping
E: charge credit card
F: send shipping status to buyer

1: name and address
2: item selected
3: credit card number
4: company account number
5: payment status
6: shipping status

Figure 3. Software system of an online seller of merchandise.

Suppose the goal of an attacker is to steal sensitive data from this system. The above steps are applied as follows:

1. Identify threats: An examination of the system in Fig. 3 found the following threats: a) theft of sensitive data flowing into A, D, and E, b) theft of sensitive data from A, C, D, and E, and c) theft of sensitive data from B.
2. Create attack trees: the weak points in the system that can lead to realizing the threats found in step 1 are: i) the paths for data flowing into A, D, and E can be exploited using man-in-the-middle attacks, ii) the processing modules A, C, D, and E can be exploited using Trojan horse or hacker attacks, and iii) the database B can be exploited using SQL attacks. These locations correspond to vulnerabilities in the system. Note that the paths of data flow into B and C are excluded as weak points because they are not considered externally accessible, due to the fact that A, B, and C all run on the same computing platform. This attack tree can be represented using hierarchical numbering as follows:

0 Theft of sensitive data from system
- 1.1 Theft of sensitive data flowing into A, D, E
  - 2.1 Man-in-the-middle attack on data paths Into A, D, E
- 1.2 Theft of sensitive data from A, C, D, E
  - 2.2 Trojan horse or hacker attack on A, C, D, E
- 1.3 Theft of sensitive data from B
  - 2.3 SQL attacks on B

This can also be depicted graphically as the attack tree in Figure 4, using the same number labels as above.

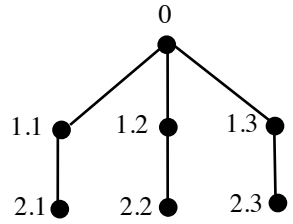


Figure 4. Attack tree for the online seller system in Fig. 3

3. Apply weights to the leaves: the leaves are 2.1, 2.2, and 2.3. Assigning weights L (low), M (medium), and H (high) to each of risk, access, and cost in turn yields (L, H, L) for 2.1, (L, M, L) for 2.2, and (L, M, L) for 2.3. This is read as low risk, high access, and low cost for 2.1, for example, meaning that there is low risk to the safety of the attacker (he is attacking from a remote location), high access to the path, and low cost to the attacker in carrying out the attack. The attacker's access for 2.2 and 2.3 are rated as medium because he or she has to actually get into the system.
4. Prune the tree so that only exploitable leaves remain: in this case, no pruning is necessary because all the leaves are within the capability of the attacker to exploit. If there was a leaf with a weighting of say, (H, L, M), i.e., high risk to the attacker's safety, low access, and medium cost, then this leaf may be pruned, in that the attacker would be unlikely to exploit the leaf.
5. Generate corresponding countermeasures: the countermeasures or security measures for the

vulnerabilities identified in step 2 are: i) encrypt the data that flow along paths into A, D, and E, ii) use firewalls to protect against Trojan and hacker attacks on A, C, D, and E, iii) use a combination of firewall and database hardening to defend against SQL attacks on B.

The threat analysis may be carried out by a project team consisting of the system's design manager, a security and privacy analyst, and a project leader acting as facilitator. In addition to having expertise on privacy and security, the analyst must also be very familiar with the organization's systems.

For inside attacks, we recommend that the above project team carry out a special insider threat analysis, to identify vulnerabilities to inside attacks and identify measures to secure these vulnerabilities. The team would accomplish this by brainstorming answers to the questions in Table I, or other questions from experience, identifying the vulnerabilities and measures to secure the vulnerabilities in the process. In Table I, questions 1 to 6 address motivational or environmental vulnerabilities, which may also be "secured" by applying mitigating measures. Questions 7 and 8 address security vulnerabilities. In identifying vulnerabilities to inside attack, the project team may weigh the vulnerabilities in terms of how likely they are to lead to attacks, and eliminate the unlikely ones. The weighing process may consider such factors as risk to the attacker that she could be caught as well as her motivation for the attack. The value of $N_i = p_i + q_i$ would be determined at the end of this process.

B. *Determining the Thresholds T_o* and T_i

The values of T_o and T_i should be determined by the same threat analysis team mentioned above. The values would depend on the following:

- The potential value of the sensitive data – the more valuable the data is to a thief, a malicious entity, or a competitor, the higher the thresholds should be.
- The damages to the organization that would result, if the sensitive data were compromised – of course, the higher the damages, the higher the thresholds.
- The current and likely future attack climate – consider the volume of attacks and the nature of the victims, say over the last 6 months; if the organization's sector or industry has sustained a large number of recent attacks, then these thresholds need to be higher.
- Consider also potential attacks by nation states as a result of the political climate; attacks by individual hacktivist groups such as Anonymous or WikiLeaks may also warrant attention.

In general, an organization would like to be as secure as possible and establish a "best" adequate SPL. Therefore, values above 80% would not be uncommon. However, whatever the thresholds, the organization must find them acceptable after considering the above factors. The financial budget available for securing vulnerabilities also plays an important role here, since higher thresholds call for securing more vulnerabilities, which means more financial resources will be needed.

TABLE I. QUESTIONNAIRE TO IDENTIFY VULNERABILITIES TO INSIDE ATTACK

	Question	Rationale
1.	Is the sensitive information of high value to outside agencies or a competitor?	The higher the value, the more an inside attacker will be tempted to steal and sell the information.
2.	Does the organization have an employee assistance program that includes counselling and help with financial difficulties?	Such a program may eliminate some financial motivation for an inside attack.
3.	Does the organization have an ombudsman or other impartial agent to assist employees with their grievances?	Such an impartial agent may eliminate or reduce the motivation to seek revenge by committing an inside attack.
4.	Does the organization have a history of perceived injustices to employees?	If the answer is 'yes', employees may be motivated by revenge to commit an inside attack.
5.	Does the organization conduct a stringent background and reliability check on a candidate for employment prior to hiring the candidate?	While a background and reliability check is not guaranteed to weed out potential inside attackers, it should eliminate those with criminal pasts.
6.	Does the organization require candidates for employment to disclose any potential conflicts of interest they may have with respect to their new employment and any outside interests prior to hire? Does the organization require ongoing disclosure of conflicts of interest after hire?	Eliminating conflicts of interest should reduce related motivations for malicious inside attacks. For example, an inside attacker may secretly compromise private information in favour of an outside interest, believing that the compromise is undetected.
7.	What are some possible ways for an insider to gain access to sensitive information she should not be accessing? How to secure?	This question will identify security weaknesses.
8.	What are some possible ways for an insider to transmit sensitive information outside the organization undetected? How to secure?	This question will identify additional security weaknesses.

C. *Applying the Estimates to Determine Optimal or "Best" Adequate SPLs*

We now have values for the following: $N_o = p_o + q_o$, $N_i = p_i + q_i$ (Section IIIA), and T_o, T_i (Section IIIB). Rewriting (11) and (13) and using the ceiling function to avoid fractional numbers of secured vulnerabilities gives:

$$q_o = \lceil N_o E_o \rceil \quad \text{where } T_o \le E_o \le 1 \qquad (15)$$

$$q_i = \lceil N_i E_i \rceil \quad \text{where } T_i \le E_i \le 1 \qquad (16)$$

Equations (15) and (16) give all possible values of q_o and q_i such that the associated E_o and E_i (with $p_o = N_o - q_o$ and $p_i = N_i - q_i$) fall within the shaded region of Figure 1. In other words, these equations give all possible values of q_o and q_i

for adequate SPLs. The ceiling function biases the security level upward by taking the number of secured vulnerabilities to the next higher integer where applicable, which should be fine since more security should be better than less security. The quantities $q_o = \lceil N_o T_o \rceil$ and $q_i = \lceil N_i T_i \rceil$ from (15) and (16), termed respectively the threshold q_o and the threshold q_i, will be useful below.

To obtain an optimal "best" adequate SPL_o and an optimal "best" adequate SPL_i from among the adequate SPLs generated by (15) and (16), the organization applies the constraint that the total cost of implementing the $(q_o + q_i)$ security measures from (15) and (16) must be less than or equal to the financial budget for security measures. The organization separately prioritizes its outside attack and inside attack vulnerabilities in terms of urgency, and then selects them for securing in order of high priority to low priority, until both the financial budget is exhausted and the number of secured vulnerabilities are at least as great as the threshold q_o and the threshold q_i. In this way, the organization determines the q_o and q_i, as well as the p_o and p_i (which are just $N_o - q_o$ and $N_i - q_i$ respectively) that define its "best" adequate SPL_o and "best" adequate SPL_i, respectively. This procedure may be precisely described in the form of a computer algorithm as follows. Let $u_1, u_2, \ldots u_{No}$ and $v_1, v_2, \ldots v_{Ni}$ be the organization's outside attack and inside attack vulnerabilities prioritized in terms of urgency, respectively, such that u_1 has higher or equal priority (urgency) than u_2, u_2 has higher or equal priority (urgency) than u_3, and so on. Similarly, v_1 has higher or equal priority (urgency) than v_2, v_2 has higher or equal priority (urgency) than v_3, and so on. Figure 5 illustrates these relationships in which equal priority does not occur. Let B_o and B_i represent the budgets for securing against outside and inside attacks, respectively. Let C_o and C_i be the costs of securing the vulnerabilities to outside and inside attacks respectively. Let k be a counter variable. Then the pseudo code in Figure 6 comprises a computer algorithm for obtaining "best" adequate SPLs. Running this algorithm will produce the following: a) q_o and q_i, defining the "best" adequate SPL_o and "best" adequate SPL_i, or b) one or two "insufficient budget" messages, in which case the organization has to increase the corresponding budgets and re-run the algorithm. Only result a) would be acceptable. If result b) is obtained, decreasing the thresholds T_o and T_i may result in fewer vulnerabilities needing to be secured, and may therefore generate result a). However, decreasing T_o and T_i is not recommended at this point, since these values were determined only after thorough analysis and consideration (see Section IIIB).

Prioritizing the vulnerabilities may be based on four aspects of an attack, namely "risk", "access", "cost", and the resulting damages from the attack, where "risk" is risk to the safety of the attacker, "access" is the ease with which the attacker can access the system under attack, "cost" is the monetary cost to the attacker to mount the attack, and resulting damages is self evident. A full explanation of this prioritization procedure is given in Yee [5].

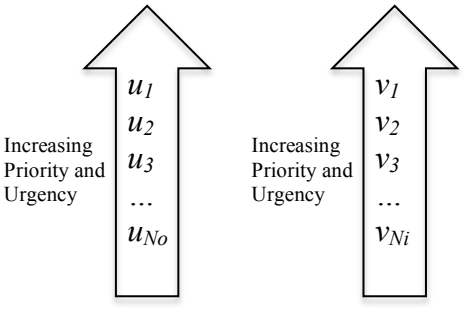


Figure 5. Vulnerabilities with increasing priorities and urgencies.

```
Begin;
   Co = 0; Ci = 0; k = 0;
   While k ≤ No and Co ≤ Bo;
      k = k + 1;
      Co = Co + cost of securing uk;
   EndWhile;
   If (k ≥ threshold qo) qo = k;
   Else Print "qo unavailable -insufficient budget";
   k = 0;
   While k ≤ Ni and Ci ≤ Bi;
      k = k + 1;
      Ci = Ci + cost of securing vk;
   EndWhile;
   If (k ≥ threshold qi) qi = k;
   Else Print "qi unavailable – insufficient budget";
End;
```

Figure 6. Algorithm for obtaining a "best" adequate SPL.

D. Application Areas

The main application area for the approach proposed in this paper is to secure an organization from attacks that target the sensitive data in the organization's computer system. An organization would implement security using the following steps:

1. Identify vulnerabilities to inside and outside attacks using threat analysis and prioritize them in terms of urgency. Identify the costs of security measures required to secure the vulnerabilities.
2. Identify "how secure" the organization needs to be given its nature and the existing attack environment, i.e., determine the values of T_o and T_i. Identify the organization's security budget. Identifying how secure it needs to be is necessary, since securing all vulnerabilities is probably not feasible due to a finite financial budget.
3. Run the algorithm in Fig. 6 to obtain q_o and q_i. Secure the prioritized outside and inside vulnerabilities up to and including q_o and q_i respectively.

The above steps may be carried out by a security consulting firm or by the organization's security department,

depending on the latter's level of confidence. A good compromise may be to hire a consultant for steps 1 and 3, since step 2 involves careful consideration that may be better done internally within the organization. The above steps can also be performed for an organization that already has some vulnerabilities secured. In this case, the already secured vulnerabilities would simply not be part of the above steps and the security budget and q_o, q_i found would be for the unsecured vulnerabilities.

Another application area is in marketing and gaining consumer confidence. Organizations that hold sensitive private information and provide services to the public may wish to advertise the fact that they have high SPLs in order to gain consumer confidence and gain competitive advantage.

A third application area lies in standardization. Studies could be undertaken by standards bodies to determine recommended "best" adequate SPL values for organizations, based on organization type, size, activity, the quantity and nature of the sensitive data held, history of information breaches, and so on. These values could be published and made available as targets or guidance for organizations seeking to implement security using this approach.

One comment received while presenting the original paper at SECURWARE 2017 was that the structured approach of securing vulnerabilities against an objective security target is not how "things are done" in industry and therefore this approach would be useless. Well, of course its not how security is implemented today – that's the whole point of this paper, to suggest a better way, one that is more quantitative, and structured, as opposed to guessing and reacting emotionally when an attack has occurred. How implementing security is currently done was described at the beginning of this paper.

IV. Application Examples

This section presents two examples of applying the proposed approach. The first example is that of an online seller of merchandise implementing security for the first time. The second example looks again at the online seller in the first example, but 5 years later than in the first example, when the online seller decides to replace its computer system for a new more high performance model. Naturally, this introduces new vulnerabilities that need to be secured.

A. Implementing Security for the First Time

Alice Inc., an online seller of goods (e.g., Amazon.com), has an objective to secure its vulnerabilities to outside and inside attacks and to establish corresponding "best" adequate SPLs using the approach in this work. The company hires a security consulting firm to perform threat analyses of its systems, resulting in a report of vulnerabilities found that could be targeted by outside and inside attackers. The report also provides values for the number of vulnerabilities as $N_o = 10$ and $N_i = 8$, and includes prioritizations of outside and inside vulnerabilities. For each type of vulnerability (i.e., outside or inside) the prioritizations identified which vulnerability required securing first, which one second, and so on, in declining order of urgency. Based on the consultant's recommendations, as well as its own internal deliberations, Alice Inc. assigned the following values:

$T_o = 0.80$, $T_i = 0.90$, $B_o = \$100{,}000$, $B_i = \$150{,}000$

Therefore

threshold $q_o = \lceil N_o T_o \rceil = \lceil 10 \times 0.80 \rceil = 8$
threshold $q_i = \lceil N_i T_i \rceil = \lceil 8 \times 0.85 \rceil = 7$

meaning that at least 8 vulnerabilities to outside attacks and 7 vulnerabilities to inside attacks must be secured in order to have "best" adequate SPL_o and "best" adequate SPL_i. Table II identifies the costs of securing the prioritized vulnerabilities where vulnerability 1 has the highest priority (urgency), vulnerability 2 has the next highest priority (urgency), and so on.

TABLE II. Costs of Securing Outside and Inside Vulnerabilities

u_k	Cost of Securing	v_k	Cost of Securing
1	\$7,000	1	\$10,000
2	\$15,000	2	\$40,000
3	\$5,000	3	\$5,000
4	\$10,000	4	\$20,000
5	\$8,000	5	\$40,000
6	\$20,000	6	\$5,000
7	\$10,000	7	\$30,000
8	\$5,000	8	\$5,000
9	\$3,000		
10	\$2,000		

As in Section III, outside and inside vulnerabilities are denoted as u_k *and* v_k respectively. Running the algorithm in Figure 6 yields $C_o = \$85{,}000$ at $q_o = 10$ and $C_i = \$150{,}000$ a*t* $q_i = 7$. The budget for securing outside vulnerabilities was more than enough to secure all outside vulnerabilities. The budget for securing inside vulnerabilities was only enough to secure 7 inside vulnerabilities. Given the existing budgets, Alice Inc.'s "best" adequate SPL_o is realized with $q_o = 10$, $p_o = 0$ and its "best" adequate SPL_i has $q_i = 7$, $p_i = 1$. Any additional security measure against inside attacks would require an increase in the budget.

B. Securing Additional Vulnerabilities

We return to Alice Inc., 5 years after implementing security in the first example. The company has grown rapidly and decides to replace its aging computer system with a new more high-performing one. However, a new system brings new vulnerabilities, so Alice Inc. decides to re-apply the proposed approach to generate new "best" adequate SPLs to make sure that the new vulnerabilities are secured. The company hires the same security consulting firm as before (the firm does good work) to perform threat analyses of its systems, resulting in a report of vulnerabilities found that could be targeted by outside and inside attackers. The report also provides values for the number of vulnerabilities as $N_o = 6$ and $N_i = 3$, and includes

prioritizations of outside and inside vulnerabilities. The vulnerability numbers are lower than 5 years ago because i) the new system did not introduce very many additional vulnerabilities, ii) the company's internal work processes have not changed very much, so that there are not many additional vulnerabilities to inside attacks, and iii) vulnerabilities secured 5 years ago are still secured. For each type of vulnerability (i.e., outside or inside) the prioritizations identified which vulnerability required securing first, which one second, and so on, in declining order of urgency. Based on the consultant's recommendations, as well as its own internal deliberations, Alice Inc. assigned the following values:

$T_o = 0.80$, $T_i = 0.90$, $B_o = \$60{,}000$, $B_i = \$30{,}000$

Therefore

threshold $q_o = \lceil N_o T_o \rceil = \lceil 6 \times 0.80 \rceil = 5$

threshold $q_i = \lceil N_i T_i \rceil = \lceil 3 \times 0.90 \rceil = 3$

meaning that at least 5 vulnerabilities to outside attacks and 3 vulnerabilities to inside attacks must be secured in order to have "best" adequate SPL_o and "best" adequate SPL_i. Table III identifies the costs of securing the additional vulnerabilities, where vulnerability 1 has the highest priority (urgency), vulnerability 2 has the next highest priority (urgency), and so on, as in example 1.

TABLE III. COSTS OF SECURING ADDITIONAL OUTSIDE AND INSIDE VULNERABILITIES

u_k	Cost of Securing	v_k	Cost of Securing
1	\$9,000	1	\$7,000
2	\$10,000	2	\$10,000
3	\$7,000	3	\$8,000
4	\$8,000		
5	\$16,000		
6	\$10,000		

Running the algorithm in Figure 6 yields $C_o = \$60{,}000$ at $q_o = 6$ and $C_i = \$25{,}000$ at $q_i = 3$. The budget for securing outside vulnerabilities was just enough to secure all outside vulnerabilities. The budget for securing inside vulnerabilities was more than enough to secure all 3 inside vulnerabilities. Given the existing budgets, Alice Inc.'s "best" adequate SPL_o is realized with $q_o = 6$, $p_o = 0$ and its "best" adequate SPL_i has $q_i = 3$, $p_i = 0$. Alice Inc. has budgeted enough funds to obtain "best" adequate SPLs that secured all vulnerabilities.

V. RELATED WORK

Related work found in the literature includes risk and threat analysis applied to various domains as well as research on vulnerabilities and countermeasures. No work was found that is similar to this work. One work, Duffany [7], is related in that it looks at protecting an enterprise's information infrastructure. This author develops an economic model for optimal resource allocation in terms of countermeasures to protect an enterprise information infrastructure. The model is solved as a linear program to determine the optimal resource allocation. However, the author does not distinguish between sensitive and non-sensitive data, but considers the organization's overall information infrastructure, including its computing devices. In addition, the author employs an economic model for optimization whereas this work optimizes based on the increase in security obtained through the addition of security measures.

In terms of risk analysis, Jing et al. [8] present an approach that uses machine learning to continuously and automatically assess privacy risks incurred by users of mobile applications. Aditya et al. [9] catalog privacy threats introduced by new, sophisticated mobile devices and applications. Their work emphasizes how these new threats are fundamentally different and inherently more dangerous than prior systems, and present a new protocol for secure communications between mobile devices. Islam et al. [10] present a risk assessment framework specifically tailored for the automotive industry. The framework starts with a threat analysis followed by a risk assessment to estimate the threat level and the impact level. This leads to an estimate of a security level, which is used to formulate high level security requirements. It is interesting that these authors also consider security levels, although the levels they use are only descriptive, such as "low", "medium", and "high".

In terms of threat analysis, Schaad and Borozdin [11] present an approach for automated threat analysis of software architecture diagrams. Their work shows that automated threat analysis is feasible. Shi et al. [12] describe a hybrid static-dynamic approach for mobile security threat analysis, where the dynamic part executes the program in a limited way by following the critical path identified in the static part. Sokolowski and Banks [13] describe the implementation of an agent-based simulation model designed to capture insider threat behavior, given a set of assumptions governing agent behavior that pre-disposes an agent to becoming a threat. Panou et al. [14] propose a cyber investment management framework named RiSKi that detects and continuously monitors insiders' societal behavior, as permitted by law, to proactively treat implied anomalies, threats, and their potential business impacts and risks. RiSKi also provides access to security incidents data to enable businesses to advance their understanding of cyber security and breaches. Sanzgiri and Dasgupta [15] present a taxonomy and classification of insider threat detection techniques based on strategies used for detection. Their classification should assist researchers and readers of this work to better understand the insider threat landscape. Baluta et al. [16] propose a discrete-event simulation model to investigate the effect of insider threats on system vulnerabilities. Their model considers both users and computer systems along with their interactions. The authors claim that the model is useful for "what-if" analysis and for gaining insights into anti-cyber intrusion strategies. Kul et al. [17] present two attack models that pose high risks for sensitive data stored in an organization's database. They discuss the complexities of both models and the defense mechanisms available in the literature.

With regard to vulnerabilities, Gawron et al. [18] investigate the detection of vulnerabilities in computer systems and computer networks. They use a logical representation of preconditions and postconditions of vulnerabilities, with the aim of providing security advisories and enhanced diagnostics for the system. Spanos et al. [19] look at ways to improve the open standard to score and rank vulnerabilities, known as the Common Vulnerability Scoring System (CVSS). They propose a new vulnerability scoring system called the Weighted Impact Vulnerability Scoring System (WIVSS) that incorporates the different impact of vulnerability characteristics. In addition, the MITRE Corporation maintains the Common Vulnerability and Exposures (CVE) list of vulnerabilities and exposures [20], standardized to facilitate information sharing. In terms of vulnerability mitigation, Oladimeji et al. [21] present a goal-centric and policy-driven framework for obtaining security and privacy risk mitigation strategies for health information interchange. They use scenario analysis and other techniques to model security and privacy objectives, threats, and mitigation strategies. Alqahtani et al. [22] propose a security vulnerability analysis framework that establishes bi-directional traceability links between security vulnerability databases and traditional software repositories. Their framework allows researchers to take advantage of semantic inference services to determine both direct and transitive dependencies between reported vulnerabilities and potentially affected software projects.

VI. Conclusion and Future Work

Organizations need to protect their sensitive data from outside and inside attacks against their computer systems that store the data. This protection is achieved by adding security measures to secure vulnerabilities to attack. However, organizations have been implementing security measures without any way of setting security protection level targets, or knowing how an added security measure contributes to the protection target. Organizations also did not have a way of selecting which security measures to implement in order to stay within the financial budget. This work proposes a structured, objective, quantitative approach to estimate, set, and achieve safe, acceptable security protection levels in terms of securing outside and inside vulnerabilities. In addition, the work proposes an algorithm for selecting which security measures to implement in order to achieve optimal adequate protection levels against outside and inside attacks, within the allowable financial budget.

This work has extended [1] in terms of updating the definition of sensitive data and adding a) more examples of attacks on sensitive data, b) more explanation and Fig. 1 on the nature of SPLs, c) an explanation and example of how to do threat analysis, d) a second application example, and e) additional references.

Future work includes investigating other formulations of security protection levels, such as incorporating the effectiveness of security measures, as well as improving the methods for threat analysis and prioritization. In addition, it would be interesting to explore how this work complements existing work in the standardization community.

References

[1] G. Yee, “Assessing Security Protection for Sensitive Data,” Proc. The Eleventh International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2017), pp. 111-116, 2017.

[2] Identity Force, “The Biggest Data Breaches in 2016,” retrieved: July, 2017, https://www.identityforce.com/blog/2016-data-breaches

[3] Identity Force, “2017 Data Breaches – The Worst So Far,” retrieved: February, 2018, https://www.identityforce.com/blog/2017-data-breaches

[4] Dark Reading, “2017 Smashed World’s Records for Most Data Breaches, Exposed Information,” retrieved: February, 2018, https://www.darkreading.com/attacks-breaches/2017-smashed-worlds-records-for-most-data-breaches-exposed-information/d/d-id/1330987?elq_mid=83109&elq_cid=1734282&_mc=NL_DR_EDT_DR_weekly_20180208&cid=NL_DR_EDT_DR_weekly_20180208&elqTrackId=700ff20d23ce4d3f984a1cfd31cb11f6&elq=5c10e9117ca04ba0ad984c11a7dfa14b&elqaid=83109&elqat=1&elqCampaignId=29666

[5] G. Yee, “Visualization and Prioritization of Privacy Risks in Software Systems,” International Journal on Advances in Security, issn 1942-2636, vol. 10, no. 1&2, pp. 14-25, 2017, http://www.iariajournals.org/security/

[6] C. Salter, O. Saydjari, B. Schneier, and J. Wallner, “Towards a Secure System Engineering Methodology,” Proc. New Security Paradigms Workshop, pp. 2-10, 1998.

[7] J. Duffany, “Optimal Resource Allocation for Securing an Enterprise Information Infrastructure,” Proc. 4th International IFIP/ACM Latin American Conference on Networking (LANC ’07), pp. 35-42, 2007.

[8] Y. Jing, G.-J. Ahn, Z. Zhao, and H. Hu, “RiskMon: Continuous and Automated Risk Assessment of Mobile Applications,” Proc. 4th ACM Conference on Data and Application Security and Privacy (CODASPY '14), pp. 99-110, 2014.

[9] P. Aditya, B. Bhattacharjee, P. Druschel, V. Erdélyi, and M. Lentz, “Brave New World: Privacy Risks for Mobile Users,” Proc. ACM MobiCom Workshop on Security and Privacy in Mobile Environments (SPME '14), pp. 7-12, 2014.

[10] M. Islam, A. Lautenbach, C. Sandberg, T. Olovsson, “A Risk Assessment Framework for Automotive Embedded Systems,” Proc. 2nd ACM International Workshop on Cyber-Physical System Security (CPSS ’16), pp. 3-14, 2016.

[11] A. Schaad and M. Borozdin, “TAM2: Automated Threat Analysis,” Proc. 27th Annual ACM Symposium on Applied Computing (SAC ’12), pp. 1103-1108, 2012.

[12] Y. Shi, W. You, K. Qian, P. Bhattacharya, and Y. Qian, “A Hybrid Analysis for Mobile Security Threat Detection,” Proc. IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), pp. 1-7, 2016.

[13] J. Sokolowski and C. Banks, “An Agent-Based Approach to Modeling Insider Threat,” Proc. Symposium on Agent-Directed Simulation (ADS '15), pp. 36-41, 2015.

[14] A. Panou, C. Ntantogian, and C. Xenakis, “RiSKi: A Framework for Modeling Cyber Threats to Estimate Risk for Data Breach Insurance,” Proc. 21st Pan-Hellenic Conference on Informatics (PCI 2017), article no. 32, pp. 1-6, 2017.

[15] A. Sanzgiri and D. Dasgupta, “Classification of Insider Threat Detection Techniques,” Proc. 11th Annual Cyber and Information Security Research Conference (CISRC ’16), article no. 25, pp. 1-4, 2016.

[16] T. Baluta, L. Ramapantulu, Y. Teo, and E. Chang, “Modeling the Effects of Insider Threats on Cybersecurity of Complex Systems,” Proc. 2017 Winter Simulation Conference (WSC), pp. 4360-4371, 2017.

[17] G. Kul, S. Upadhyaya, and A. Hughes, "Complexity of Insider Attacks to Databases," Proc. 2017 International Workshop on Managing Insider Security Threats (MIST '17), pp. 25-32, 2017.

[18] M. Gawron, A. Amirkhanyan, F. Cheng, and C. Meinel, "Automatic Vulnerability Detection for Weakness Visualization and Advisory Creation," Proc. 8th International Conference on Security of Information and Networks (SIN '15), pp. 229-236, 2015.

[19] G. Spanos, A. Sioziou, and L. Angelis, "WIVSS: A New Methodology for Scoring Information System Vulnerabilities," Proc. 17th Panhellenic Conference on Informatics, pp. 83-90, 2013.

[20] MITRE, "Common Vulnerabilities and Exposures", retrieved: July, 2017, https://cve.mitre.org/

[21] E. Oladimeji, L. Chung, H. Jung, and J. Kim, "Managing Security and Privacy in Ubiquitous eHealth Information Interchange," Proc. 5th International Conference on Ubiquitous Information Management and Communication (ICUIMC '11), article no. 26, pp. 1-10, 2011.

[22] S. Alqahtani, E. Eghan, and J. Rilling, "SV-AF – A Security Vulnerability Analysis Framework," Proc. 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), pp. 219-229, 2016.

RMDM – Further Verification of the Conceptual ICT Risk-Meta-Data-Model

Verified with the underlying Risk Models of COBIT for Risk and COSO ERM 2017

Martin Latzenhofer [1] [2]
[1] Center for Digital Safety & Security
Austrian Institute of Technology
Vienna, Austria
email: martin.latzenhofer@ait.ac.at

Gerald Quirchmayr [2]
[2] Multimedia Information Systems Research Group
Faculty of Computer Science, University of Vienna
Vienna, Austria
email: gerald.quirchmayr@univie.ac.at

Abstract— **The aim of this article is to introduce an approach that integrates the different models and methods currently applied for risk management in information and communication technologies (ICT). These different risk management approaches are usually bound to the organization where they are applied, thus staying quite specific for a given setting. Consequently, there is no possibility to compare or reuse risk management structures because they are individual solutions. In order to establish a common basis for working with different underlying risk models, a metamodeling approach from the area of disaster recovery is used. This contribution describes a comprehensive mapping of information artefacts from both the COBIT for Risk and the COSO Enterprise Risk Management (ERM) framework in its new 2017 version which are then lifted to the meta-level of the proposed ICT risk-meta-data-model in order to be able to work with them in a consolidated way. Through this mapping process, all information artefacts are extracted, consolidated and harmonized to minimize the number of relevant objects. It has turned out that both the list of consolidated objects and the derived describing attributes can in general be incorporated into the proposed ICT risk-meta-data-model (RMDM). The results show that it is worth examining a data-structure-oriented approach to develop both a model and a data structure for further framework-independent processing.**

Keywords-information and communication technology risk management; ICT risk-meta-data-model; COBIT for Risk; COSO ERM 2017; metamodeling; UML.

I. INTRODUCTION

In literature and in practice, many different risk management approaches and models can be found for the area of information and communication technology (ICT) systems. Even within the field of ICT, these approaches and models are tailored quite narrowly to specific areas and are typically restricted to one single organization. Therefore, the information on risk management is usually not comparable and transferrable between different organizations. This means that the risk model, the established risk management method, the concrete process implementation, the required input data and the resulting outcome have to be adapted to the current requirements of an organization every time the risk management process is set up. This often leads to high efforts for an organization or a company because they have to initialize and re-establish the risk management frameworks and related processes each time. It is evident that these parameters result in a smaller degree of reusability of a given risk management process and less comparability of the information obtained from it.

When interpreting this problem as a pure ICT issue, an explicit ICT solution is required. This leads to the main research question of this paper, i.e., whether it is possible to develop a common risk management model, which is flexible enough to be applicable in different fields of the ICT area as well as among different organizations. To achieve that, it is crucial to define a suitable level of modeling. Therefore, the goal of the introduced approach is to design a meta-model for ICT risk management. By integrating different existing ICT risk management models, which are suitable for various fields of application into a meta-model, a generic data structure that focuses on common aspects of these models can be developed. This umbrella model simply obtains data from the underlying specialized models that have been defined by different frameworks. In this work, the first mapping was performed with the risk model included in COBIT for Risk. Subsequently, the same transformation method was applied to another risk model that forms part of the COSO ERM 2017 version. The approach introduced in this article postulates a superordinate meta-model for ICT risk management and represents it as a data model, which is expressed as a UML class diagram. The iterative performance of the mapping strengthens the first result of the meta-model and ensures the detailed design of classes, attributes, and methods. Considering the application of ICT risk management in practice, the state-of-the-art frameworks are well-established in the daily business of organizations. Consequently, it is not realistic to replace them by a new, universally valid model. The ICT risk-meta-data-model approach introduced here firstly establishes a common data base of risk information gathered by different risk management frameworks, secondly makes data retrieved from different sources comparable, and thirdly verifies its practical applicability by describing real-life use cases, shown as an instantiation of the ICT risk-meta-data-model.

The main goal is to specify the meta-model as a substantial data model. Using such a precise data model, the meta-model is directly applicable to real-life scenarios and enables the implementation of a dedicated ICT application or

data structure. The data model is directly applicable for ICT tasks, provides a concrete ICT data structure where risk information can be stored, and is a fundamental (data) basis for ICT risk management applications.

The authors' conference paper for SECURWARE 2017 discussed a first comprehensive mapping of a concrete risk model – provided by COBIT for Risk – to the ICT risk-meta-data-model [1]. This present journal paper now integrates a second risk model from COSO ERM 2017 into the proposed RMDM, which has led to further adjustments and thus a more sustainable structure of the RMDM. The originating idea of the conceptual ICT risk-meta-data-model was first introduced as a draft proposal at DACH Security 2016, in Klagenfurt, Austria [2].

This article is divided into five main sections. Following this introduction, Section II starts with a short introduction of the common risk management framework COBIT for Risk, which was selected for the first mapping of risk models to the meta-model level. It discusses the processes of COBIT for Risk which are relevant for managing risk in detail. Section II continues to shortly introduce the COSO ERM 2017 framework as the second risk model that has been mapped. Subsequently, it describes the fundamentals of the applied metamodeling approach and concludes with discussing related work. In Section III, the conceptual data model RMDM, described in Unified Modeling Language (UML) is introduced. The version of the RMDM that is presented in this work represents the current state of the model after the two mappings mentioned above have been performed. Section IV firstly discusses the mapping of the information artefacts, input and output components of COBIT for Risk, which are the core of the derived risk model, the objects of the proposed ICT risk-meta-data-model (RMDM) and the results of the mapping in detail. In line with this approach, the second part of Section IV documents the mapping of COSO ERM 2017 and the respective findings, which resulted in a slight refinement of the UML classes. The general objective of this section is to apply the postulated meta-model by modeling an instance of two concrete risk models. Both mappings represent an analysis of whether modeling at the meta level works in general. The concluding Section V outlines the results and proposes further research that is needed to refine the ICT risk-meta-data-model (RMDM).

II. FUNDAMENTALS

Typically, organizations have a continuous need to manage the risks in their business environment. Such a need due to extrinsical factors is often motivated by legal requirements. Organizations have to ensure compliance with regulations, especially relating to finance and public accounting. Therefore, the responsible person implements risk management – in this case limited to the ICT area – by doing research and building upon already existing risk management structures. Special risk management frameworks that are applicable to ICT, e.g., International Organization for Standardization (ISO) 31000 [3], National Institute of Standards and Technology (NIST) Special Publication (SP) 800-30/-37/-39 [4] [5] [6], Committee of Sponsoring Organizations of the Treadway Commission (COSO) Enterprise Risk Management (ERM) [7], Management of Risk [8] or COBIT for Risk [9], have proven to be effective within one single organization. These frameworks set up a baseline in an organization when it comes to implementing risk management structures. This usually generates isolated solutions. The different risk management frameworks are characterized by relatively similar objects and terms but very different artefacts, which cannot be related, compared, or summarized. One important issue is to harmonize the semantic differences between the various risk management frameworks, and even within one single framework.

A. COBIT for Risk

COBIT for Risk [9] is a special publication edited by Information Systems Audit and Control Association (ISACA, since 2008 the acronym itself is used as a brand name) [10] and is entirely based on Control Objectives for Information and Related Technology (COBIT, since version 5 only the acronym itself is used as a brand name) 5.0 [11], a framework for governance and management of Enterprise ICT, especially for the interaction between ICT and classic business objectives. COBIT for Risk is a comprehensive guide for risk professionals. It elaborates the driving aspects for risk management in COBIT – principles and enablers – and extends the framework with risk scenarios. Furthermore, it provides suggestions for appropriate response measures using a combination of enablers. It has – similar to ISO 31000 [3] – a two-tier approach: the risk management perspective puts the high-level principles into practice and the risk function view seeks to identify relevant COBIT processes, which support the risk management, as depicted in Figure 1. In this figure, the two core risk processes are shown in light blue, the other twelve key supporting processes are colored in dark red.

The COBIT for Risk framework was chosen as a first candidate for the intended mapping because of its good balance between general applicability for risk management topics and very specific statements in form of concrete control objectives for risk management. It definitely provides much more topic-oriented reference-points than standard COBIT. The framework is clearly structured and its description is not too narrative. A highly narrative framework might increase the effort for identifying class objects. In summary, all these characteristics were considered to be good prerequisites for the practical mapping work. Other frameworks, e.g., ISO 31000 [3], might be too generic in order to derive substantial class objects to a sufficient extent or, e.g., NIST [4] [5] [6], is too text-heavy for an efficient proof of concept. Consequently, all the other frameworks are rather suitable for verifying the ICT risk-meta-data-model in a more advanced state of development.

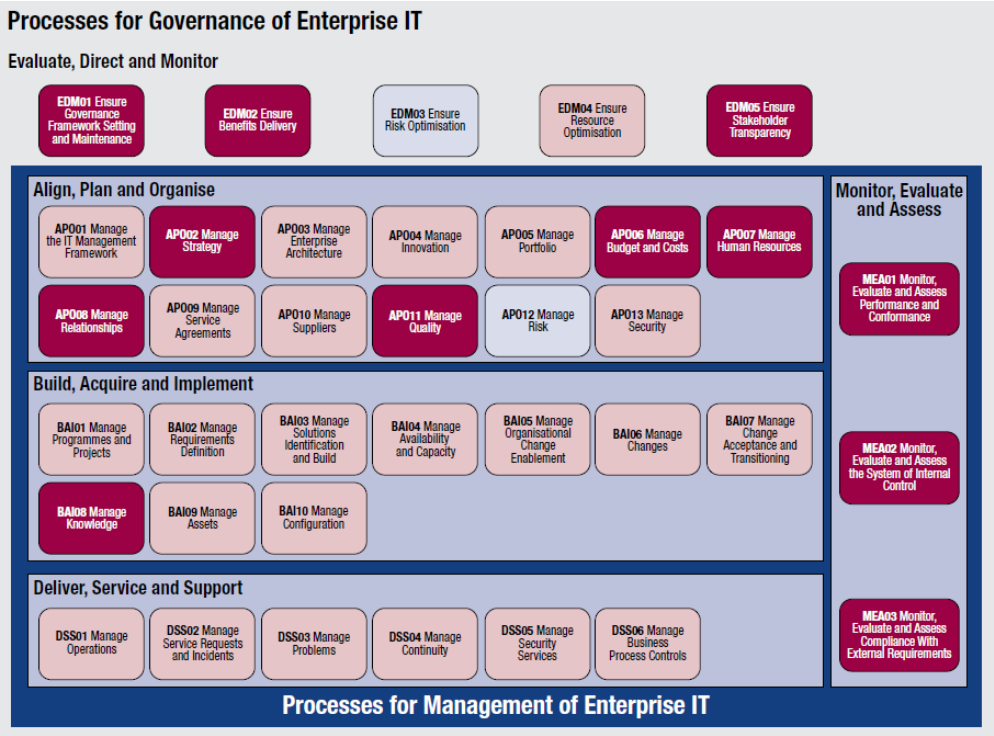


Figure 1. Supporting COBIT processes for the risk function [9, p. 35]

B. COSO ERM 2017

The first version of the "Enterprise Risk Management – Integrated Framework" (ERM) [12] was published by the Committee of Sponsoring Organizations of the Treadway Commission (COSO) [13] in 2004 and was an extension of their first "Internal Control – Integrated Framework" publication from 1992 [14]. In the middle of 2017, COSO fundamentally revised the comprehensive ERM framework in cooperation with PricewaterhouseCoopers (PwC) [15] to address the raising complexity of risk in the business environment. The ERM framework aims at providing guidance to managers on how to handle different kind of risks in an appropriate way. The main objective of this framework is to offer managers specific methods and techniques for managing risks in their organization and to provide them with different use cases.

In general, the COSO ERM 2017 framework emphasizes the relationship of risk management activities with the organization's strategy, business objectives and current performance in order to raise the value that is derived from the organization's mission, vision, and core values. This requires consistent risk management activities at all levels of the enterprise. The main objective is to establish a balance between risk and performance by developing a coherent risk profile based on an appropriate risk appetite that depends on the individual situation of the organization on the market. Typically, the performance targets and risks vary to a certain degree, which is referred to as tolerance in performance and risk capacity of the organization. Consequently, the organization seeks to reach the best possible performance within the given restrictions over time.

The framework itself is a set of 20 principles that are categorized by five interrelated components, which are illustrated in Figure 2. The first component "Governance & Culture" addresses the setup of the organization, which includes the organizational structure, the definition of core values and of behavioral expectations, and the organization's reliability and accountability. The second component "Strategy & Objective Setting" defines input requirements for the risk management in the organization: e.g., business context, risk appetite, alternative strategies and business objectives. Once the risk management framework has been set up, the organization can conduct the operational risk management process, which is described in the component "Performance". In this process, the risk manager identifies, assesses and prioritizes risks, implements risk responses and develops an oversight portfolio view of all risks. In the "Review & Revision" component, the risk manager reviews the changes in risk and performance. The last component, "Information, Communication & Reporting", deals with communication and reporting issues.

Although this framework follows a typical top-down approach, it differentiates between the different levels in an organization – i.e., governance and strategic level (the first two components) and operational level (especially the third component). However, it also addresses the guiding processes for reviewing the risks in the fourth and communication in the fifth component; as it is also done in

Figure 2. COSO ERM 2017 Risk Management Components and Principles [15, p. 21, Fig. 5.1 / pp. 22, Fig. 5.2]

ISO 31000, COBIT for Risk, and the relevant NIST special publications. The 20 principles provide a good set of key principles which management has to follow in order to establish mature risk management processes in the organization. COSO ERM 2017 discusses the problem of cascading risks, continuous changes of risks and the required adjustments of the risk profiles and established responses, and it covers cost implications.

The COSO ERM 2017 framework was mainly chosen as the second candidate for the mapping because of its dense content and because its narrative description is similar to COBIT for Risk. Additionally, the new version of COSO ERM from June 2017 provides an up-to-date perspective on the current business complexity and risk management measures. Consequently, COSO ERM 2017 represents a suitable framework for further verifying and developing the RMDM after the first verification by the risk model of COBIT for Risk.

C. Metamodeling Approach

The semantic meaning of a risk model must be transferred to the meta-level. A formal, scientific approach to build a consistent umbrella is missing. The meta-modeling process helps to create a common basis for standardization. The instantiation procedure of the meta-model down to the distinct risk management framework provides rules for transferring data from a concrete model up to the meta-model, and is in that way working as a normalization process. The first advantage of representing the risk-meta-model as data model is the immanent design of a structured data management based on a semantic model. It must be verified whether the general concepts can be divided from content-specific aspects in such a way that the interaction between meta- and model-level still remains efficient. The data model works as a structure model and holds static information. The risk management process and corresponding workflows change this data dynamically, providing a data model for the whole risk management life cycle. However, this article focuses on the verification of the basic content and on whether the data model can process the information. In addition, the meta-model approach for standardizing risk management information can be implicitly verified by setting up the data model, at least for those risk models which have been analyzed earlier. Certainly, it is no evidence for its comprehensiveness that all existing risk models still fit in the proposed meta-model. In fact, some models might be unsuitable for mapping. However, re-performing the transformation process for a specific number of widely accepted risk frameworks ensures that the meta-model is sufficiently applicable for risk management tasks in organizations.

In the context of a metamodeling hierarchy according to Karagiannis and Kühn [16] (cf., Figure 3), the ICT risk-meta-data-model is situated on Level 2 – Metamodel, described by the Metamodeling Language UML. The selected risk management framework, e.g., COBIT for Risk [9], corresponds to Model on Level 1. It is described by means of the published framework, here in a semi-narrative way. The underlying Original itself can in fact be referred to as Level 0, and represents the organization's risk management structure facing a concrete risk situation. On the top of the hierarchy, the Meta²-Model on Level 3 defines the structural elements of the general UML class diagram. The Meta²-Modeling Language can be understood as the modeling language UML used to describe the ICT risk-meta-data-model.

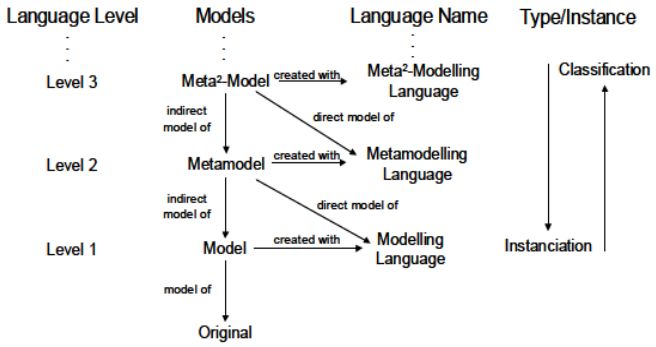


Figure 3. Metamodeling hierarchy [16]

D. Related Work

A lot of academic work has been published on selecting the best risk model or the most promising method to identify and assess ICT risks. Literature research on the general subject of risk management modeling has shown that there are certain common issues that are typically encountered. One issue is the complex application field which the conventional risk management models cannot satisfactorily cope with. Typical examples for difficulties are cascading effects of risks, a broad diversity of possible risks not being able to cover the complete risk landscape or even a high degree of uncertainty. Some recent works deal with ICT risks in established domains which remain hard to manage, e.g., outsourcing, project management, software development; relatively novel innovative environments like cloud computing, internet of things (IoT) with still diffuse risk implications; or previously independent and even stand-alone domains which are now merging, e.g., safety and security or cyber-physical systems (CPS) in relation to ICT. It should be noted that risk management structures and measures, including many different and widely accepted models, have already been established in organizations. However, they tend to struggle with different types of models because in many cases none of them fits their individual requirements completely.

At least four different approaches can be identified. The first category of scientific approaches the proposal of "yet another new/integrated/simplified/formally structured model". The following citations refer to representative examples for this category that were found during the literature research and are definitely not exhaustive: e.g., for cloud computing risks [17], enterprise risk [18], outsourcing risks [19], or even general information security risks [20], which may be appropriate for the very specific situations and circumstances for which these models have been developed for. However, this interpretation generates even more models and leads to the fact that models are not universally applicable. A second approach of the scientific community is to apply combinations of different already existing models or methods to get a more accurate and/or comprehensive result. A representative example for outsourcing can be found in [21], for project management in [22], or risk mitigation decisions in [23]. Thirdly, digging a bit further into the inherent structure of risk models, there are other remarkable approaches which link different methodological approaches together, e.g., simulation, empirical studies and/or modeling approaches [24] [25]. The fourth strategy – which seems to be the most promising approach for the present task – is to generalize the models and to transfer them to a superordinate meta-level. Obviously, meta-modeling is an interesting approach when all the challenges discussed in the previous paragraph arise: firstly, a complex and diverse application domain (e.g., ICT risk management) which cannot be analyzed satisfactorily with one single model among the variety of different and already applied structural frameworks (e.g., all the widely accepted risk management frameworks like ISO 31000, NIST, COSO, COBIT for Risk). However, the risk models do not fit in every aspect (e.g., strategic or operational, business or technical risks) and sometimes even combinations of different approaches (e.g., modeling or simulation, quantitative or semi-quantitative if possible versus qualitative approaches) lead to different and partially contradictory results. In fact, meta-modeling promises good results when a methodological superstructure does not exist. Representative contributions for this approach which address the inherent complexity of specific domains can be identified for e.g., safety and security [26], big data [27], cyber-physical environments [28], or even information system security risk management [29].

The approach introduced in this article is explicitly inspired by similar work in the field of disaster recovery [30] [31], which introduced a meta-model integrating data from different natural disaster scenarios. Othman and Beydoun have implemented a data model in order to store data relevant to disaster recovery and have conducted a proof of concept for two natural disaster incidents of recent history, the Christchurch earthquake and Fukushima nuclear incident [31]. In this article, their approach is shifted into the ICT risk management domain while verifying whether it is a sustainable method for risk management.

III. ICT Risk-Meta-Data-Model (RMDM)

This section introduces the proposed ICT risk-meta-data-model and its current status of development up to now, followed by a discussion about the main components.

A. General Requirements

One of the main objectives of the conceptual ICT risk-meta-data-model is to record key information of any underlying risk model in a way that it can be compared, consolidated, merged and subsequently analyzed from an abstract meta-perspective. This approach ensures that risk management models that have already been implemented in organizations in practice continue to be used, at least the most commonly applied frameworks. Furthermore, this abstraction step reduces the information risk managers work with to the really essential requirements needed to establish the risk management framework and to perform the risk management process. This transformation from the risk model to the more abstract and general meta-level must follow specific rules and definitely causes some information loss. To succeed it is necessary to strike a viable balance between the appropriate level of detail of the information content – by selecting only the key data, combining it

semantically correct and transferring it to the meta-level – and the complexity level of the risk-meta-data-model. The authors assume that an adequate level of abstraction is reached when three to four structurally different risk models can be consistently represented as instances of the ICT risk-meta-data-model. This iterative refinement of the risk-meta-data-model through the analysis of different underlying risk models enhances its sustainability and robustness for practical application. The major advantage of formulating the ICT risk-meta-data-model as an ICT data model is that this allows organizations and companies to apply it in practice. By depicting the meta-model as unified modeling language (UML) classes diagram the modeler can immediately generate the corresponding data structure, implementing a demonstrator, which can serve as a proof of concept. Consequently, the ICT risk-meta-data-model itself constitutes an ICT application that can be applied in practice. In other words, the ICT problem to merge data from different risk models requires an ICT solution, which can immediately be applied by IT means.

The first draft of the ICT risk-meta-data-model was developed based on literature research on different risk management frameworks, which all propagate distinct risk models but use the same or similar terms. The literature research also indicated that there is a need to reflect on the exact meaning of the used terms, even if they seem to be identical. A feasible mapping of the concepts used in different risk models is a prerequisite for successfully raising the key information of the risk model up to the meta-level. This requires the definition of consistent concepts on the meta-level in order to prevent overlapping of concepts and resulting misinterpretations. However, it depends on the specific framework whether the risk model can be derived directly from the publications. ISO 31000 [3], for example, is formulated in a generic way, thus leaving room for interpretation. COBIT for Risk [9], does in contrast provide very specific control objectives for the key and supporting processes on a more detailed level. This characteristic was the main reason for selecting COBIT for Risk for the first mapping of a risk model to the ICT risk-meta-data-model.

The conceptual model aims at reflecting both the fundamental framework establishment and the operative risk management process that covers the risk management lifecycle. This dual perspective is a key feature of many frameworks and easily visible in, e.g., ISO 31000 [3], NIST [4] [5] [6], or even COBIT for Risk [9] or COSO ERM 2017 [15]. A core aspect was to identify appropriate objects, which represent the focus points within the risk management structure. These objects are further described by dedicated attributes, which are the variables for storing the relevant risk management information. These attributes can be changed, modified, extended, and adapted by specific methods. By setting up this data structure it is possible to transfer all relevant risk management data from the origin model up to the ICT risk-meta-data-model. A very first draft of the modeling was already introduced in [2]. This article included a first draft of the ICT risk-meta-data-model and a possible approach for a proof of concept by applying COBIT for Risk as the underlying risk model. The first version of the ICT risk-meta-data-model was the result of a creative process. This process followed the life cycle of risk management: starting with the identification of risk factors, followed by the analysis of the resulting risk by linking it to the current challenges that the organization has to cope with, and finally the evaluation of the risk. Furthermore, the data-model may represent the monitoring of established treatment activities. As a consequence, the data model fulfills the essential requirements of the risk management process as suggested in [3]. The next step was to perform a precise mapping of information artefacts propagated by COBIT for Risk [9] as described in Section IV A and B. The logical next step was to repeat the mapping with another fully applicable risk management framework. The selected COSO ERM 2017 framework provides a similarly dense narrative description, which was a good prerequisite for the further verification of the RMDM. Another reason for choosing this framework obviously was that the publishing organization COSO thoroughly updated it in June 2017. The second mapping and its results are discussed in Section IV C and D.

B. Main Components

Figure 4 shows the status quo of the advanced ICT risk-meta-data-model (RMDM) after the second mapping. Classes or relationships written in italics are represented in the UML diagram. The RMDM can be divided in five main components. On an abstract level, all classes are derived from class Organisation and further divided in *Input*, *Process*, *Output* and *Actor*. These classes of the first component introduce a fundamental structure to group the other classes within the risk management process. This construction with generalization relationships both introduces an additional inherent structure of the data model and applies generalization and inheritance of attributes by superior classes in order to cope with the rising complexity. A particularity of the class *Actor* has to be underlined. The class *Actor* represents all persons and their responsibilities taken over by organizational entities, persons or roles, e.g., by the risk manager. *Actor* also *owns* a *Process*. However, especially the class *Process* should also be able to summarize all important processes, policies, standards and guidelines that form the operational environment. It is not only an abstract data structure, but rather a hybrid class.

The operative part of the conceptual model and the linked classes can be further divided into three virtual processing parts, which are not explicitly included in the UML diagram in Figure 4 but structurally grouped in line with the virtual workflow collecting input, processing and controlling risks. In the first phase, which summarizes all the different input factors and puts them into a common context, the conceptual model shows the detailed causal chain from the single risk factors to the identified risk, which is in fact a prerequisite for performing an operational risk management process. This architectural characteristic enables a possible ex-post analysis or simulation by means of the provided background information in the input classes to examine their practical influences on the final risk. The resulting risk is only a product of its input factors. The appealed causal chain starts on the left side with a pure *Hazard*, which *threatens* a

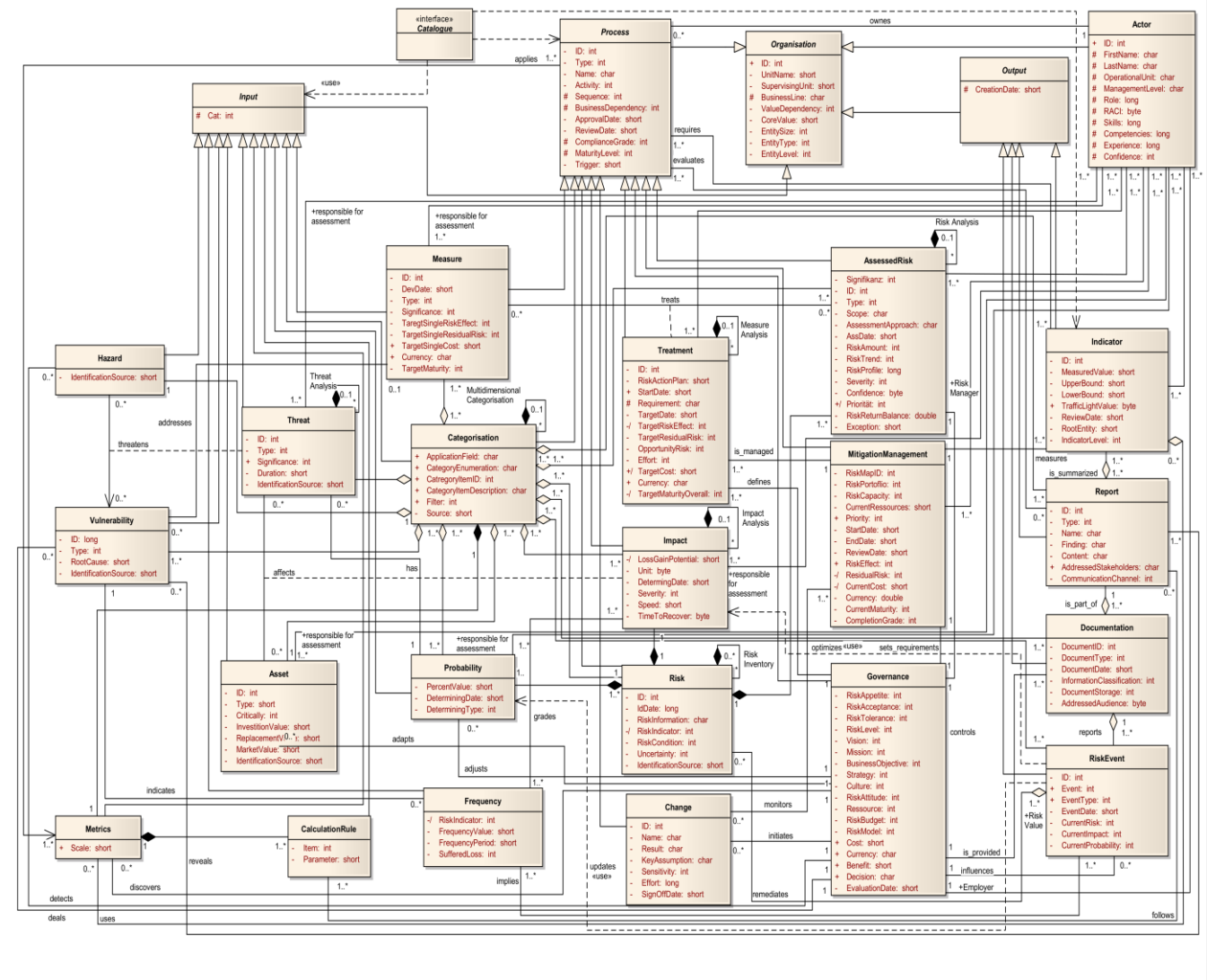


Figure 4. Conceptual ICT risk-meta-data-model (RMDM) described as UML class diagram after the COSO ERM 2017 mapping [own research]

particular *Vulnerability*, resulting in the associated class *Threat*. Hence, the *Threat* explicitly *affects* an *Asset* of the organization, leading to one main risk factor *Impact*, which is also designed as an associated class. The *Threat has* also some *Probability* to materialize, which is the second immediate main risk factor. Typically, the *Risk* can be characterized by its essential components *Impact* and *Probability*, which are often shown in a risk matrix and here designed as composition of both risk factors. However, the *Risk* reflects only the identified risks and does not yet link to a detailed assessment, which the organization is required to do as a next step.

The second phase of the risk management process further processes the risk. It involves the assessment of the previously identified raw risks and linking them with the given influencing factors and framework conditions. Accordingly, the class *Risk* is a composition for *AssessedRisk*. This class records all necessary evaluations of the risks. A *Measure* treats *AssessedRisk*, but there is no indication whether these measures are really applied in this stage. This is indicated by the associated class *Treatment*. In this way, a gradual filter starting from *Risk*, via *Measure* to *Treatment* can be applied. This filter – which could also be interpreted as a second causal chain in the same sense as the risk factors discussed above – allows focusing only on those risks, which should be actively addressed in the risk management process and further reduces the complexity of the model to the high-risk areas according to the individual risk level. Consequently, all selected *Treatments* are managed by *MitigationManagement* during their whole lifecycle. The class *MitigationManagement* represents the fundamental risk portfolio from a manager's point of view. Thus, this class represents the core structure for performing the risk management process within the defined risk management framework over time. After the identification of risks, this stage effectively reduces the amount of relevant risk on which the organization should focus. Finally, management will control the remaining risks by applying a

continuous risk management process. The underlying idea is that the relevant risks which need ongoing attention will be filtered, reduced in their amount and monitored by a cyclic quality process that forces the organization to regularly look at the remaining risks.

The third stage of the operative risk management process represented in the ICT risk-meta-data-model addresses the management's governance and its supporting elements, e.g., key output, risk events, or metrics. The class *Governance* establishes requirements for the class *MitigationManagement* and subsumes all the influencing factors to set up the appropriate risk environment. It holds management information about finance, vision, mission, business objectives, strategy, culture and business context, risk model, attitude, appetite and tolerance etc. It is supported by ongoing *Changes*, which subsume all ancillary activities that support risk management activities, i.e., projects, changes. The class *Categorization* addresses all forms of structuring, e.g., categories, graduations, risk scales, and cluster definitions in the context of risk management efforts, and provides additional structure, while it leaves enough leeway for individual metrics. Consequently, the *Categorization* class has relationships with all the classes that need such a structuring. It is also possible to integrate external catalogues, frameworks, and regulations into the risk management model through the interface class *Catalogue*. The intention of this part is to reflect on the necessary high-level governance of risk management in the responsible organization.

The fifth and final part covers all aspects that are relevant for documentation and measuring performance. This fifth part enables to take a current snapshot of the risk situation and forms a new starting point for a further cycle of the risk management process. Additionally, this part also provides concrete information on actual risk, which helps to achieve a higher maturity degree of the risk management lifecycle. *Documentation* in any form, especially *Reports* or (Key Risk) *Indicators*, has specifying classes, which are implemented as aggregations from the generic structure (*Documentation*) to more quantifiable information (*Indicator*). *Documentation* covers all documents that are relevant for governance decisions and thus creates an information repository. *Metrics* with specified *CalculationRules* stores all kinds of calculation bases, e.g., for Balanced Scorecard, Key Risk Indicators, or Process Performance. This ICT risk-meta-data-model also includes an important feedback loop. The class *RiskEvent* ensures the remediation of risk information based on new findings due to incidents based on real-life incidents. In combination with the class *Frequency*, the quantification of already suffered risk events enables the adjustment of the underlying risk factors, thus increasing the accuracy of further assessments. Finally, intended self-referencing relationships for the classes *Categorization*, *Threat*, *Impact*, *Risk*, *AssessedRisk*, and *Treatment* enable further substantial analysis, e.g., multidimensional assessments of cascading effects if needed. These five main parts of the ICT risk-meta-data-model interlock with one another. Thus, both the continuous elements of the risk management process and the different perspectives of operative process performance and strategic embedding in the organization – in fact the apparent two-tier approach of the discussed risk management frameworks can be reflected in the model.

IV. MAPPING

A. Method for the COBIT for Risk Mapping

The critical success factor for the proper functioning of the meta-modeling idea is the coherent transformation of the information of the selected risk model up to the meta-model while at the same time sufficiently reducing the information content. This transformation is in fact a mapping of all the relevant pieces of information that is necessary for performing risk management with the selected risk model. The risk model COBIT for Risk was selected as the first proof of concept for the metamodeling approach. It provides an appropriate degree of concreteness in order to verify the draft concept that was first introduced in [2].

In a first step, both risk management core processes Evaluate, Direct and Monitor (EDM) 03 "Ensure Risk Optimisation" – the setup of the risk management environment in the organization – and Align, Plan and Organise (APO12) "Manage Risk" – the risk management process as discussed above – were analyzed. All information artefacts mentioned as input or output objects and in the description of the risk specific activities were extracted to a list. These have a different degree of concreteness, which was also assessed. This step was repeated for each of the other twelve supporting processes, which are marked in dark red in Figure 1. This finally resulted in a list of 1619 identified information artefacts, but this list included duplicates, synonyms, and different notations of the same objects, cf. Figure 5. In a second step, all these entries were consolidated in order to even out differences and reduce the amount of information artefacts for further analysis. All entries were transformed into a consolidated object, in fact performing a form of abstraction. This transformation resulted in a list of 26 objects, which corresponds to the column 'synonym' in Figure 5. The purpose of these objects was to set up a data store, leading to a UML class at the end of this process. This abstraction process was conducted as iterative working step because the consolidated object list initiated continuous improvement actions in order to get a coherent list for the subsequent steps. Once the list of consolidated objects had been verified, the consolidated object list was mapped to the classes in the UML diagram. In a third step, the class attributes were revised so that the essential data for risk management fit properly into the appropriate classes.

B. Results of the COBIT for Risk Mapping

The mapping process showed that it is generally possible to transform the essential risk management data from COBIT for Risk up to the meta-level. Small amendments to the draft version of the ICT risk-meta-data-model were necessary after completing the mapping process, e.g., the introduction of the new class *Changes*, which reflects all current change management activities in the considered organization. The

transformation is highly dependent on how concrete the specification of the risk model and its components is. If the risk model leaves too much room for interpretation inconsistencies may appear in the instantiation of the ICT meta-data-risk-model itself. This means that activities without inputs or outputs should be scrutinized. Almost all inputs, outputs and standard COBIT 5 activities specified in the twelve risk supporting processes were unsuitable for the mapping. Thus, certain problems are expected when using ISO 31000 as base risk model because of its highly generic approach. This means that not every risk management framework may be suitable for the mapping due to the different levels of detail of the different frameworks. Furthermore, the framework must provide storage of all kind of documentation that supports the functioning of the management system. Currently, the meta-model includes the dedicated class *Documentation* for this issue. It was originally intended only for risk management documentation, but it has a broader scope, providing a repository for all documentation produced by the applied management system.

Process	Source	Artefact	Level of Detail	Synonym
APO012.01	1	analysis method	medium	Process
APO012.01	1.1	analysis model	low	Process
APO012.01	1.4	assessment of risk attribute	medium	Assessment
APO012.01	2.2	audit	medium	Actor
APO012.01	2.2	business source	low	Catalogue
APO012.01	2.2	CIO office	medium	Actor
APO012.01	1	classification method	medium	Category
APO012.01	1.1	classification model	high	Category
APO012.01	4.1	collected data	medium	Catalogue
APO012.01	1	collection method	medium	Process
APO012.01	1.1	collection model	low	Process
APO012.01	2.3	competition within industry	low	Metrics
APO012.01	2.3	competitor alignment	low	Metrics
APO012.01	2.2	compliance	medium	Requirement
APO012.01	4.1	contributing factor	high	Risk Factor
APO012.01	4.3	contributing factor	high	Risk Factor
APO012.01	3.1	data collection model	low	Process
APO012.01	1.4	data for incentive setting (risk-aware culture)	low	CorporateGovernance
APO012.01	2	data on enterprise`s operating environment	medium	Catalogue

Figure 5. Excerpt of the list of information artefacts of COBIT for Risk [own research]

The first mapping extends the proof of concept that was outlined in [2] to all affected risk management processes of the COBIT for Risk framework. Some small adjustments of the first draft of the ICT risk-meta-data-model were made, but no fundamental changes of the inherent structure of the classes or relationships were necessary. This shows that the ICT risk-meta-data-model is able to represent and store the necessary information for applying the COBIT for Risk framework in principle.

C. Method for the COSO ERM 2017 Mapping

In order to further develop the RMDM in the version of [1], a second mapping with another risk model is performed. The objective of this second round of mapping is to verify the defined class structure and refine the attributes, which were established after the COBIT for Risk mapping. This second mapping aims at ensuring a consistent structure of the RMDM for both underlying frameworks. If this second mapping was also successful, the RMDM could be applied at least for those two risk models, would be more robust, more mature and expected to be applicable in a more flexible way to other frameworks. The assumption is that the adjustments which will be needed to integrate a second risk model should be limited. On the other hand, it is important that the modifications are done carefully because the class structure and especially its attributes must be valid for both risk models.

In contrast to COBIT for Risk, which is in fact a specification of a broad, comprehensive governance framework for risk management, the whole framework COSO ERM 2017 [15] is explicitly designed for risk management purposes. This implies that all five components and the 20 principles – as depicted in Figure 2 – had to be analyzed. According to the applied mapping method, all information artefacts mentioned in these principles have been identified in the narrative description of the framework, finally resulting in 1 935 entries. This list, which is illustrated in Figure 6, has the same structure as the list that was generated for the COBIT for Risk mapping earlier, so that the entries can be easily compared later on. It was remarkable that the information artefacts could be directly mapped to an existing class object that already existed in the RMDM. Consequently, the consolidating intermediate stage for categorizing, harmonizing and finally minimizing of the information artefacts was not needed. Additionally, the first mapping was helpful to leave no room for interpretation. It also ensured a kind of quality control in order to avoid inconsistencies during both mapping processes. Furthermore, it showed that the class structure already has a stable form and is ready for a third risk model integration.

D. Results of the COSO ERM 2017 Mapping

Since the attributes had already been aligned to COBIT for Risk, the necessary adjustments to the attribute's names had to be done very carefully. Therefore, no attributes were deleted and there was no need for this either. The main objective was to provide sufficiently clear mapping paths from an information artefact on the risk model level to a class attribute on the meta-level, regardless of whether COBIT for Risk or COSO ERM 2017 was applied. The second mapping resulted in small adjustments in the names of attributes and even in additional attributes. The detailed changes that were introduced to the RMDM as a result of the COSO mapping were as follows. A self-referenced relationship of the class *Risk* depicts a risk inventory. A direct relationship between *Actor* and *Process* reflects the process ownership. An additional relationship of the *Documentation* and the *Report* class to the *Categorization* class helps to better categorize the different types of documentation and reports. The further attribute 'identification source' defines the origin of the different risk factors. More attributes were added to the classes *Organisation*, *Governance*, *MitigationManagement*, *AssessedRisk* and *Risk*. It can be argued that the reason for these additional attributes is the stronger top down approach of COSO ERM 2017 compared to COBIT for Risk. The additional attributes provide points of references for vision, mission, risk attitude, risk model, culture, risk capacity, risk portfolio, current [assigned] resources, core values, size, type

Process	Source / Section	Information Artefact	degree of cor	Synonym
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	strategy	low	Governance
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	strategy	low	Governance
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	supporting assumptions relative to business context, resources, capabilities	medium	Treatment
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	technical expertise	medium	Actor
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	types of risk	medium	AssessedRisk
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	vision	low	Governance
P8: Evaluates Alternative Strategies	Understanding the Implications from Chosen Strategy	working capital	low	Asset
P9: Formulates Business Objectives	Understanding Tolerance	achievement of business objective	medium	Governance
P9: Formulates Business Objectives	Understanding Tolerance	achievement of strategy	low	Governance
P9: Formulates Business Objectives	Understanding Tolerance	approach for measuring	medium	Process
P9: Formulates Business Objectives	Understanding Tolerance	boundary of acceptable variation	medium	MitigationManagement
P9: Formulates Business Objectives	Understanding Tolerance	business objective	medium	Governance
P9: Formulates Business Objectives	Understanding Tolerance	enhancing value	medium	Asset
P9: Formulates Business Objectives	Understanding Tolerance	management	high	Actor
P9: Formulates Business Objectives	Understanding Tolerance	mission	low	Governance
P9: Formulates Business Objectives	Understanding Tolerance	objective	medium	Governance
P9: Formulates Business Objectives	Understanding Tolerance	outcomes	medium	Governance
P9: Formulates Business Objectives	Understanding Tolerance	performance	medium	Indicator
P9: Formulates Business Objectives	Understanding Tolerance	range of tolerance	high	MitigationManagement
P9: Formulates Business Objectives	Understanding Tolerance	risk	medium	Risk
P9: Formulates Business Objectives	Understanding Tolerance	risk appetite	high	Governance
P9: Formulates Business Objectives	Understanding Tolerance	strategy	low	Governance
P9: Formulates Business Objectives	Understanding Tolerance	tolerance	high	MitigationManagement
P9: Formulates Business Objectives	Understanding Tolerance	vision	low	Governance
P10: Identifies Risk	Using a Risk inventory	category	high	Categories
P10: Identifies Risk	Using a Risk inventory	impact of risks	high	Impact
P10: Identifies Risk	Using a Risk inventory	number of risks identified	high	AssessedRisk
P10: Identifies Risk	Using a Risk inventory	opportunities	medium	Risk
P10: Identifies Risk	Using a Risk inventory	risks	low	Risk

Figure 6. Excerpt of the list of information artefacts of COSO ERM 2017 [own research]

and level of entity. The current status of the RMDM after the second mapping is shown in Figure 4, which can be compared with its previous version in [1, Fig. 3], if needed.

It is remarkable that the concepts of terms and definitions used in the COSO ERM 2017 framework are more consistent than in the COBIT for Risk, resulting in fewer synonyms and discrepancies in terms and notations and making it easy to find double entries. The terms were applied in a highly consistent way throughout the whole framework. Therefore, it was much easier than in the first mapping to perform the consolidation phase. It was almost a straight-forward process to select the right class objects from the RMDM. A key difference compared to COBIT for Risk is that COSO ERM 2017 starts with the risk itself and does not even analyze the previous risk factors before. Consequently, 451 identified information artefacts could be subsumed under the *Governance* class, as many as 66 remained to be subsumed under the class *MitigationManagement* and 223 under the *AssessedRisk* class. In total, this amounts to almost 42 percent of all identified information artefacts of COSO ERM 2017. In contrast, in COBIT for Risk the number of assigned artefacts to these three classes amounts to only 304 information artefacts.

Finally, it shows that all the risk management information that is necessary for applying the COSO ERM 2017 risk model can also be covered by the existing RMDM classes. Due to the fact that the terms and definitions in COSO ERM 2017 are more robust than in COBIT for Risk, some light adjustments of the attribute names make sense and are important. Based on the higher consistency of terms and definitions in the COSO ERM 2017, both the RMDM classes and their attributes pass a kind of consistency check when these adjustments of attributes are performed.

E. Further Research

Further research is still needed to verify the transformation process with two or three other risk management frameworks. This verification should definitely be done for ISO 31000 [3], despite the above-mentioned expectation that the framework will be too generic. The suitability of ISO 31000 should be verified because of its outstanding importance as a world-wide standard. The NIST Special Publications 800-30/-37/-39 [4] [5] [6] also provides the more detailed content that is necessary for the mapping and is thus a good candidate. Moreover, its importance in the US strongly suggest an integration into the RMDM. If it is possible to map their information requirements in the same way as it has been done for COBIT for Risk and COSO ERM 2017, the ICT risk-meta-data-model can be applied at least for these four risk management frameworks, in this way providing an adequately sustainable meta-model solution.

However, the top down approach of the COSO ERM 2017 mapping reveals the inherent problem of adequately reflecting abstract concepts of terms like culture, code of conduct, behavior, expectations or business context, which are not easy to present in the data structure. For the moment, all these aspects have been subsumed under culture and it has not been decided yet how to integrate them into the data structure in more detail.

If the mapping process has been applied several times and the attributes are almost stable (except for a refinement of the definite data types and the visibility properties), the methods can be refined next. The methods of a class should be able to support the complete lifecycle of the concerning attributes. The third area in which refinements are needed is the relationships. It must be verified whether a direct data exchange between the different objects is needed or transitive relationships achieve the same result. Once these

three research questions have been solved, the ICT risk-meta-data-model can be implemented as a first demonstrator, thereby starting the technical verification process. Analyzing these research questions is an ongoing process in order to verify the applicability and utility of the ICT risk-meta-data-model.

In the currently ongoing CERBERUS project of the Austrian National Security Research Program KIRAS [32], the RMDM can serve as a basis for the required overall data model. The objective is that the CERBERUS model holds static data about critical infrastructure objects and combines them with dynamic data obtained from simulations and analyses to represent cross-sectoral cascading risks. The RMDM can serve as a starting point for the development of the CERBERUS data model and can provide structural inputs for risk describing concepts.

The fundamental idea of aggregating risk management data that is stored in different risk models and can be effectively applied when different risk information, e.g., from different companies or organization units that still apply different risk models, need to be migrated. This might be necessary when different companies merge or Comparisons across industry sectors are needed. Another possible application is to use the meta-data-model for training purposes. The model helps to highlight the key elements which are essential for a comprehensive risk management. Moreover, the differences between the risk management frameworks in terms and structure of different risk management models can be illustrated to future risk managers. The implicit comparison between the different approaches gives the training participants an overview of existing risk management approaches that are used in practice.

V. CONCLUSION

This article shows the basic instantiation of two specific risk models – in this case the risk models of COBIT for Risk and COSO ERM 2017 – by means of the conceptual ICT risk-meta-data-model. The objective of the research design is to introduce an ICT risk-meta-data-model for ICT, and to embed it in the context of different established risk models that are commonly applied in the ICT area. The approach of designing a consistent superstructure in form of a meta-model with no need for replacement of the already established ICT risk management models is based on the principle of an ex-post adjustment. Additionally, it provides a data-oriented and more formalized way of overcoming the current organizational and model-related restrictions. The meta-model addresses the whole risk management lifecycle as recommended in [3], from identification, analysis, evaluation to treatment. It reflects both the risk management context and the monitoring and communication requirements for the process. The three main components and the conceptual background of the involved objects are discussed. The findings can be summarized as follows:

- An instantiation of the ICT risk-meta-data-model is generally possible and is a promising possibility to overcome the current situation in ICT, where many different risk models and methods are applied.
- The critical success factor is the coherent transformation of the information of the selected risk model up to the meta-model, while at the same time sufficiently reducing the information content. All essential data of the risk model have an equivalent reference in the superstructure.
- It is crucial to repeat the mapping with other appropriate ICT risk models in order to strengthen the ICT risk-meta-data-model. Moreover, this will reconfirm the general applicability of the meta-data-model and will increase its utility due to having several different risk models mapped to a meta-level.
- The methods and relationships of the objects in the ICT risk-meta-data-model need to be refined before a practical demonstrator can be implemented that can be fed with risk management use cases.

Results show that transferring the general information artefacts specified by COBIT for Risk as well as COSO ERM 2017 into the classes of the meta-model is feasible and promising. The future refinement effort will iteratively improve the ICT risk-meta-data-model in order to further develop and evaluate it and strengthen its applicability for ICT risk management.

ACKNOWLEDGMENT

This work received financial support from the CERBERUS (Cross Sectoral Risk Management for Object Protection of Critical Infrastructures) Project (No. 854766) of the Austrian National Security Research Program KIRAS, Call 2016/2017.

REFERENCES

[1] M. Latzenhofer and G. Quirchmayr, “RMDM – A Conceptual ICT Risk-Meta-Data-Model – Applied to COBIT for Risk as underlying Risk Model,” presented at the SECURWARE 2017: The Eleventh International Conference on Emerging Security Information, Systems and Technologies, Rome, 2017, pp. 117–124 [Online]. Available: www.thinkmind.org

[2] M. Latzenhofer, “Ein Meta-Risiko-Datenmodell für IKT,” in *Bestandsaufnahme, Konzepte, Anwendungen, Perspektiven*, Klagenfurt, pp. 161–173.

[3] International Organization for Standardization (ISO), Ed., *ISO 31000:2009 Risk management - Principles and guidelines*. ISO, Geneva, Switzerland, 2009.

[4] National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Joint Task Force Transformation Initiative Information Security, Ed., “NIST 800-30: Guide for Conducting Risk Assessments.” Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-8930, USA, Sep-2012 [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf

[5] National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Joint Task Force Transformation Initiative Information Security, Ed., “NIST 800-37: Guide for Applying the Risk Management Framework to Federal Information Systems: A Security Life Cycle Approach.” Computer Security Division,

Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-8930, USA, Feb-2010 [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-37r1.pdf

[6] National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Joint Task Force Transformation Initiative Information Security, Ed., "NIST 800-39: Managing Information Security Risk - Organization, Mission, and Information System View." Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-8930, USA, Mar-2011 [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-39/SP800-39-final.pdf

[7] Committee of Sponsoring Organizations of the Treadway Commission (COSO) and Price Waterhouse Cooper (PwC), "Enterprise Risk Management - Aligning Risk with Strategy and Performance (Public Exposure Draft)." Jun-2016.

[8] The Stationary Office (TSO), Ed., "Management of Risk: Guidance for Practitioners." 2010.

[9] Information Systems Audit and Control Association (ISACA), Ed., "COBIT 5 for Risk." Information Systems Audit and Control Association, Rolling Meadows, IL 60008 USA, 2013 [Online]. Available: http://www.isaca.org/COBIT/Pages/Risk-product-page.aspx

[10] Information Systems Audit and Control Association (ISACA), "ISACA." [Online]. Available: https://www.isaca.org/Pages/default.aspx. [Accessed: 24-May-2018]

[11] Information Systems Audit and Control Association (ISACA), Ed., "COBIT 5: A Business Framework for the Governance and Management of Enterprise IT." Information Systems Audit and Control Association, Rolling Meadows, IL 60008 USA, 2012 [Online]. Available: http://www.isaca.org/cobit/Pages/CobitFramework.aspx

[12] Committee of Sponsoring Organizations of the Treadway Commission (COSO), *Enterprise Risk Management Framework*. Committee of Sponsoring Organizations of the Treadway Commission, 2004.

[13] Committee of Sponsoring Organizations of the Treadway Commission (COSO), "COSO." [Online]. Available: http://www.coso.org/. [Accessed: 24-May-2018]

[14] Committee of Sponsoring Organizations of the Treadway Commission (COSO), "Internal Control - Integrated Framework." 1992.

[15] Committee of Sponsoring Organizations of the Treadway Commission (COSO) and Price Waterhouse Cooper (PwC), *Enterprise Risk Management - Integrating with Strategy and Performance*, vol. 1. 2017.

[16] D. Karagiannis and H. Kühn, "Metamodelling Platforms," in *E-Commerce and Web Technologies*, vol. 2455, K. Bauknecht, Am. Tjoa, and G. Quirchmayr, Eds. Springer Berlin Heidelberg, 2002, p. 182 [Online]. Available: http://dx.doi.org/10.1007/3-540-45705-4_19

[17] N. Rödder, R. Knapper, and J. Martin, "Risk in modern IT service landscapes: Towards a dynamic model," in *Service-Oriented Computing and Applications (SOCA), 2012 5th IEEE International Conference on*, 2012, pp. 1–4.

[18] Y. Yu, Q. Hao, and P. Hao, "The research and application of enterprises' dynamic risk monitoring and assessment model based on related time series," in *Chinese Automation Congress (CAC), 2017*, 2017, pp. 7407–7410.

[19] T. R. Bezerra, S. Bullock, and A. Moura, "A simulation model for risk management support in IT outsourcing," in *Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH), 2014 International Conference on*, 2014, pp. 339–351.

[20] D. Wawrzyniak, "Information security risk assessment model for risk management," in *International Conference on Trust, Privacy and Security in Digital Business*, 2006, pp. 21–30.

[21] M. Kakvan, M. A. Mohyeddin, and H. Gharaee, "Risk evaluation of IT service providers using FMEA model in combination with Multi-Criteria Decision-Making Models and ITIL framework," in *Telecommunications (IST), 2014 7th International Symposium on*, 2014, pp. 873–878.

[22] N. C. Pa and B. Anthony, "A model of mitigating risk for IT organisations," in *Software Engineering and Computer Systems (ICSECS), 2015 4th International Conference on*, 2015, pp. 49–54.

[23] M. L. Yeo, E. Rolland, J. R. Ulmer, and R. A. Patterson, "Risk mitigation decisions for IT security," *ACM Trans. Manag. Inf. Syst. TMIS*, vol. 5, no. 1, p. 5, 2014.

[24] C. A. Pinto, A. Tolk, and M. McShane, "Emerging M&S application in risk management," in *Proceedings of the 2011 Emerging M&S Applications in Industry and Academia Symposium*, Boston, Massachusetts, 2011, pp. 92–96.

[25] Q. Zheng and H. Na, "Insurance IT outsourcing risk assessment modeling and empirical study," in *Information and Financial Engineering (ICIFE), 2010 2nd IEEE International Conference on*, 2010, pp. 202–206.

[26] Y. Zhang, B. Hamid, and D. Gouteux, "A metamodel for representing safety lifecycle development process," presented at the 6th International Conference on Software Engineering Advances (ICSEA 2011), 2011, pp. 550–556.

[27] B. Yang and T. Zhang, "A Scalable Meta-Model for Big Data Security Analyses," in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on*, 2016, pp. 55–60.

[28] F. Cicirelli, G. Fortino, A. Guerrieri, G. Spezzano, and A. Vinci, "A meta-model framework for the design and analysis of smart cyber-physical environments," in *Computer Supported Cooperative Work in Design (CSCWD), 2016 IEEE 20th International Conference on*, 2016, pp. 687–692.

[29] M. Findrik, P. Smith, J. H. Kazmi, M. Faschang, and F. Kupzog, "Towards secure and resilient networked power distribution grids: Process and tool adoption," in *Smart Grid Communications (SmartGridComm), 2016 IEEE International Conference on*, Sydney, Australia, 2016, pp. 435–440.

[30] S. H. Othman and G. Beydoun, "Metamodelling approach to support disaster management knowledge sharing," presented at the 21st Australasian Conference on Information Systems (ACIS), Atlanta, GA, USA, 2010, pp. 1–10 [Online]. Available: http://ro.uow.edu.au/cgi/viewcontent.cgi?article=10789&context=infopapers

[31] S. H. Othman and G. Beydoun, "Model-driven disaster management," *Inf. Manage.*, vol. 50 (2013), no. Elsevier, pp. 218–228, Apr. 2013.

[32] Federal Ministry for Transport, Innovation and Technology (BMVIT) and Austrian Research Promotion Agency (FFG), "KIRAS Security Research: CERBERUS," 2016. [Online]. Available: http://www.kiras.at/en/projects/detail/d/cerberus/. [Accessed: 25-May-2018]

A Survey on Open Forensics in Embedded Systems of Systems

From Automotive Considerations to a Larger Scope

Robert Altschaffel, Kevin Lamshöft, Stefan Kiltz, Mario Hildebrandt, Jana Dittmann
Advanced Multimedia and Security Lab
Otto-von-Guericke-University
Magdeburg, Germany
email:Robert.Altschaffel|Kiltz|Mario.Hildebrandt|Jana.Dittmann@iti.cs.uni-magdeburg.de; Kevin.Lamshoeft@st.ovgu.de

***Abstract*—Embedded systems form the foundation for most electronic systems in our everyday environment. Complex systems of embedded systems, usually connected via communication buses, are fundamental for a broad range of applications like industrial control systems (ICS) or transportation. While the complexity of these systems of systems is ever increasing, our understanding of these systems is decreasing. This not only leads to problems in designing safe and secure systems - but also in reconstructing the chain of events that led to an unwanted outcome regarding the system of systems in question. While previous work gave a survey on the means to perform event reconstruction (hence, forensic investigations) into automotive environment, this work extends the point of view to include other systems of embedded systems connected using communication bus technologies. Hence, the main contribution is a survey on the field of forensics in generalized control systems. This includes the identification of implications for forensics in ICS, the discussion of approaches and tools already in existence and those still needed to perform a forensic investigation in ICS environments.**

Keywords- automotive; computer forensics; embedded systems; forensic processes; industrial control systems; safety & security.

I. INTRODUCTION

Most complex electronics-based systems rely on embedded systems connected to actuators and sensors to perform their given task. They often implement a number of (open/closed) control loops using sensors as input and actuators as means to influence their physical environment - earning them the alternative designation as cyber-physical (cp) systems. Prime examples for such control systems are automotive systems [1] and industrial control systems. In these domains, embedded systems control even fundamental, safety-critical functions. Hence, problems with the security of such control systems can affect its safety - e.g., a vehicle might crash or an industrial robot might hurt a worker if something goes awry.

While the paramount objective is to prevent such incidents from happening altogether, this objective seems to be out of reach forever. Hence, in the case of such an incident, (computer) forensic investigations are necessary in order to identify the course of events that led to a given outcome. The results of this investigation can help to fix vulnerabilities or act as a foundation for an assessment of legal responsibility. Especially in the latter case, it is necessary that such an event reconstruction follows generally accepted, scientific and well-proven principles. These principles are referred to as a forensic process. A forensic process requires traces used for event reconstruction to be gathered and analyzed in an authentic fashion (originating from the subject of the investigation) and with integrity (unaltered by external influences or during the course of the investigation) assured. Further, the whole process is required to be comprehensively documented. The process also needs to ensure protection of personal data in accordance to applicable regulations and laws. Further, the process needs to respect regulations concerning the collection of data, especially in consideration of privacy laws and human rights.

In the beginning of an investigation it is very often unclear if an incident arises from an error or an attack or if an investigation will be escalated to include legal authorities. An investigation thus should follow the same principles without regards to the starting hypothesis of the investigator.

Previous work [1] discussed this topic with a scope on automotive systems while this article contributes an extended scope with a generalized view on control systems.

In this broader scope, the same principal challenges as within the automotive domain are apparent: there is a lack of forensic processes focused on (industrial) control systems (so called ICS) that are openly discussed and peer-reviewed in the scientific community. This holds particularly true for the basic process control, where often non-standard IT-equipment is used [2], and affects the identification, acquisition, investigation and analysis of potential case-relevant data. While isolated solutions are applied, these are often hidden behind heavy regulations regarding intellectual property. Hence, this article aims at discussing the possibility of applying well-researched approaches from the domain of desktop-IT forensics to control systems.

In order to achieve this, this article is structured as follows: Section II gives an overview on the technical background of control systems and forensics. Section III discusses the forensic process in the context of control systems. Currently available tools, which might support the forensic process in ICS and their suitability for forensic use, are discussed in Section IV. Section V discusses the requirements for tools geared towards supporting forensics in ICS, while Section VI concludes this article.

II. TECHNICAL BACKGROUND

This section gives a brief overview on the topic of forensic in classical desktop IT and a basic understanding of (industrial) control systems in order to bring these topics together in the following sections. In order to compare forensics in the ICS domain to the automotive domain, a brief recap of automotive IT is given as well.

A. Forensics in Desktop IT

The forensic process aims at finding traces that support the reconstruction of an event. In order to increase the validity of the reconstruction, these traces have to be gathered in a way to preserve authenticity (trace origin) and integrity (trace is unaltered). Additional challenges arise from the need to protect personal data in accordance to applicable regulations and laws as well as respecting regulations concerning the collection of data, especially in consideration of privacy laws and human rights.

To ensure this, a range of models for the forensic process exist, both for classical crime scenes [3], as well as for digital forensics in Desktop IT [4]. These models are often practitioner driven and usually break down the forensic process into distinct phases. For this article, we use the forensic process from [5], as it contains both the practitioner's and the computer scientist's view (see [6]), the latter often being omitted in an attempt to provide guidelines for practitioners only. This model includes investigation steps (practitioner's view), data types (computer scientist's view) and methods for data access (computer scientist's view). Thus, by adhering to this model, both the research aspect as well as the implementation of forensic procedures in practice is supported.

For this first survey on automotive IT forensics we rely on the investigation steps:

- **Strategic preparation** (*SP*) represents measures taken by the operator of an IT-system, prior to an incident, which support a forensic investigation.
- **Operational preparation** (*OP*) represents the preparation for a forensic investigation after a suspected incident.
- **Data gathering** (*DG*) represents measures to acquire and secure digital evidence.
- **Data investigation** (*DI*) represents measures to evaluate and extract data for further investigation.
- **Data analysis** (*DA*) represents the detailed analysis and correlation between digital evidence from various sources.
- **Documentation** (*DO*) represents the detailed documentation of the investigation.

The forensic process is furthermore also divided into live forensic and post-mortem forensics. Live forensics covers the part of the forensic examination performed while the system under investigation is still active. Post-mortem forensics covers all the part of the forensic examination while the system under investigation is powered-off. Live forensics offers the possibility to find traces in highly volatile areas such as main memory but often comes with the implication of substantially altering the state of the system under investigation - either by letting it perform its current operations or by querying the system for certain information from the main memory, which actively alters the state of the system. Post-mortem forensics allows access to lesser volatile mass storage and analyze it in ways ensuring integrity of the mass storage device (typically by using write-blocking devices) but cannot gain insight into the main memory contents. The consideration when to power off a system under investigation and switch from live forensics to post-mortem is to be decided on a case-by-case basis and represents a crucial decision in every forensic examination.

B. Automotive IT

As discussed in previous work [1], modern cars consist of components with fixed logic (or none at all) and components with (re-)programmable logic. The latter often include embedded systems and thus are more important for this article, although being only useful in conjunction with electronic devices with fixed or no built-in logic. Of particular relevance for our discussion are:

- **Sensors** measure the conditions of the vehicle's systems and environment (e.g., pressure, speed, light levels, rain intensity etc.) as well as user input.
- **Actuators** are electrically operated and manipulate their environment in non-electric aspects (e.g., mechanics, temperature, pressure, etc.).
- **Electronic Control Units** (ECUs) electronically process input signals acquired via sensors and relay commands to actuators. Some units control critical systems, such as the engine or safety-critical systems like ESC (Electronic Stability Control) or SRS (Supplemental Restraint System), while others control comfort functionality (e.g., door control units). ECUs are custom-tailored compact, embedded systems. Due to high cost constraints in the automotive industry, they operate on a minimum set of resources regarding CPU computing power, mass storage and main memory. Common exceptions are ECUs that handle multimedia functionality. The number of ECUs embedded with a vehicle is still rising - while a luxury car in 1985 contained less than 10 ECUs, the numbers increased to more than 100 in 2010 [7].
- **Direct analogue cable connections** connect sensors and actuators directly to a specific ECU.
- **Shared Digital Bus Systems** are used for communication among ECUs [8]. In modern cars, several different technologies for digital automotive field bus systems are used with different capabilities, requirements and cost factors. The most common automotive field bus system, often forming the core network of vehicle systems communication, is the Controller Area Network (CAN) [9]. This CAN network is often divided into sub-networks such as powertrain/engine, diagnostics, comfort or infotainment. ECUs are connected to the sub-network and these sub-networks interconnect using a CAN Gateway ECU, which handles the routing of messages to different sub-networks. The CAN

message consists of several flags, the CAN ID and the payload. The CAN ID represents the type of a message and implies a certain sender and receiver for the message. It is assumed that a message with the corresponding ID is sent by the ECU normally responsible for this message. In addition, the CAN ID serves as priority. A lower CAN ID corresponds to a higher priority.

The above implement essential instrumentation and control circuits for the functionality of today's vehicles.

C. *Industrial Control Systems (ICS)*

On a fundamental level, ICS consist of the same basic building blocks found in automotive IT. Again, sensors collect information about the environment while actuators manipulate the environment. Just like ECUs in automotive systems, so called Programmable Logic Controller (PLCs) process input taken from sensors and relay commands to actuators. The components are usually arranged in more or less complex hierarchies, as shown in Fig. 1. Since these actuators could be used to manipulate hazardous substances or objects (e.g., hot steam, poisonous gas, heavy loads), some of them might be regarded as safety-critical. Some ICS might also control critical infrastructures [10] such as emergency services or traffic control. Communication with users and/or supervisors is performed using dedicated human-machine-interfaces (HMI) or supervisory control and data acquisition (SCADA) control systems. PLCs need to communicate with each other using various communication buses, i.e., field buses such as PROFIBUS [11] or industrial Ethernet such as PROFINET [12] or Modbus TCP [13].

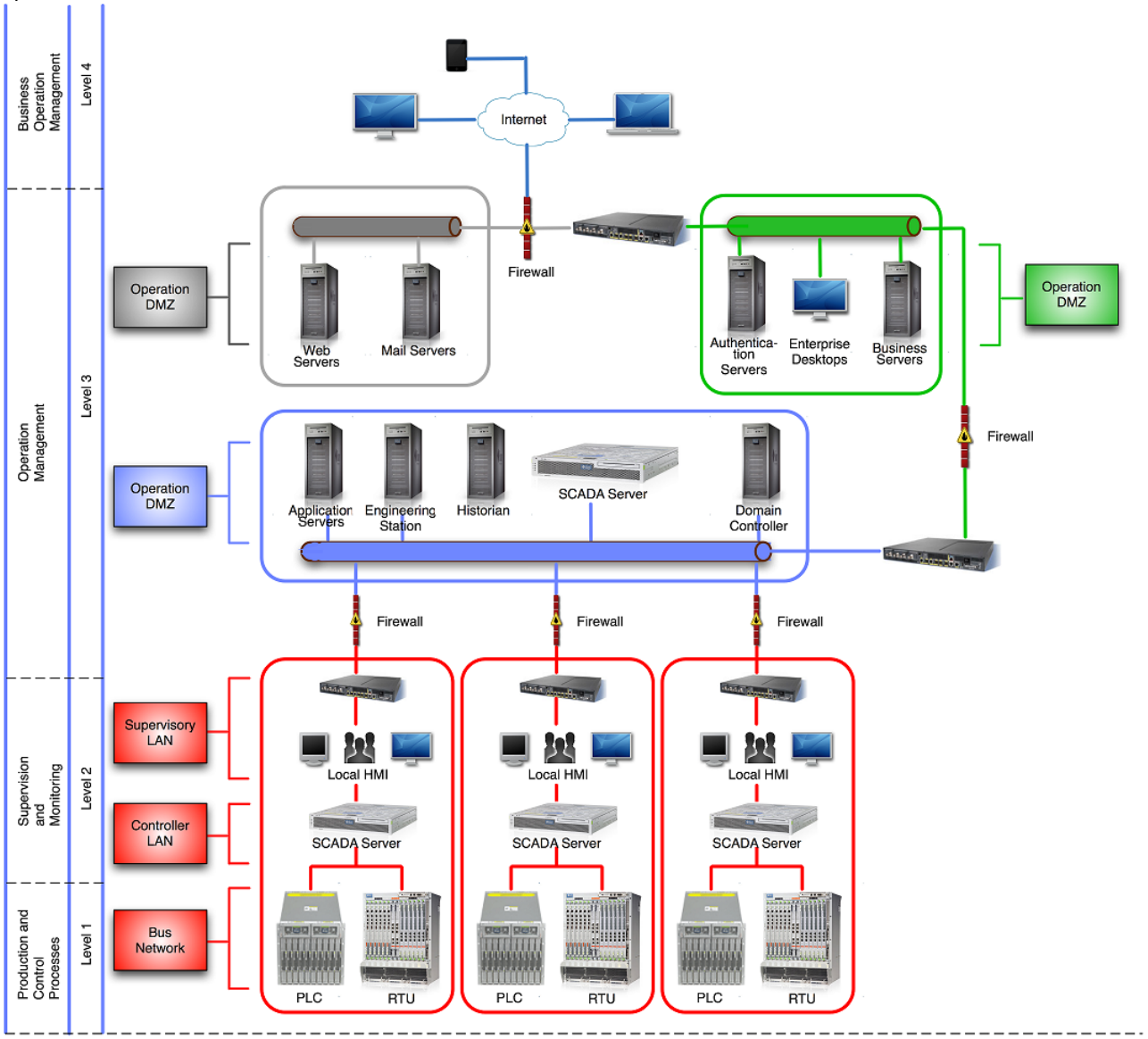


Figure 1. Exemplary Industrial Control Systems, based on [15]

Fig. 1 shows a typical hierarchy for ICS. This figure also shows, that ICS often form a part of larger company networks. Hence, ICS are often connected to classical IT.

A model for these hierarchical networks is given by the Purdue Enterprise Reference Architecture [14]. Today this architecture is used in various iterations, for example, the ISA99 variation.

According to [15], this model defines three different zones within an enterprise network, as pictured in Fig. 2. The zones serve as an indicator for communication flows - communication is only possible within a given zone or to the neighboring zone. That means that there is no direct communication between Enterprise Zone and Cell/Area Zone.

The specific levels are defined in [16]:

- *Level 0 — The physical process — Defines the actual physical processes.*
- *Level 1 — Intelligent devices — Sensing and manipulating the physical processes. Process [sic] *sensors, analyzers, actuators and related instrumentation.*
- *Level 2 — Control systems — Supervising, monitoring and controlling the physical processes. Real-time controls and software; DCS, human-machine interface (HMI); supervisory and data acquisition (SCADA) software.*
- *Level 3 — Manufacturing operations systems — Managing production workflow to produce the desired products. Batch management; manufacturing execution/operations management systems (MES/MOMS); laboratory, maintenance and plant performance management systems; data historians and related middleware. Time frame: shifts, hours, minutes, seconds.*
- *Level 4 — Business logistics systems — Managing the business-related activities of the manufacturing operation. ERP is the primary system; establishes the basic plant production schedule, material use, shipping and inventory levels. Time frame: months, weeks, days, shifts.*

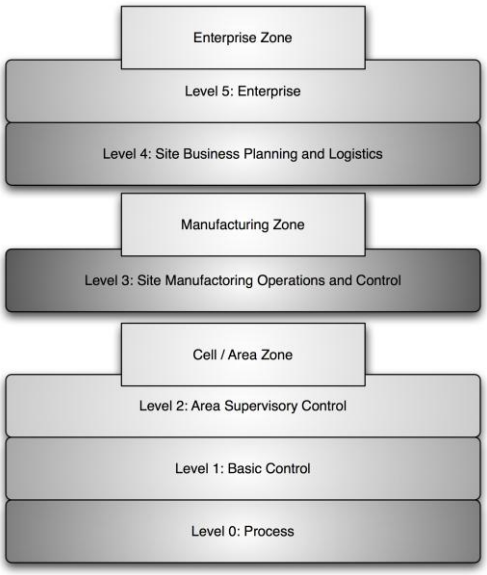


Figure 2. Purdue Model in the variant of ISA99, according to [15]

According to this, activities on level 4 and above are not the scope of this work since they basically represent the classic IT domain - Industrial Control Systems correspond to the components on levels 0 to 3 of the hierarchy defined in [16] for the scope of this article. Hence, ICS contain PLCs, sensors, actuators, HMIs, and SCADA.

III. Review of the forensic Process in Control Systems

This section discusses the implications originating from the involvement of ICS in a forensic process. After Section II covered the fundamental structure of ICS, this section will start with an overview of the data present in this structure. Following this, an overview on the nature of concurrent attacks targeting cyber-physical systems is given in order to identify components and data affected by these attacks. After that, implications for the forensic process will be formulated, based on these observations and the specifics of ICS.

A. Data Streams in Cyber-physical Systems

Components in ICS share the same possible locations of data as classical IT systems or components in automotive IT. These possible locations of data are referred to as data streams [1]. In general, three data streams can be identified in components:

Communication describes the data transmitted over networks. In an ICS environment this network might consist of field bus systems or dedicated serial lines.

Volatile memory represents the non-persistent part of a component's memory. In the context of ICS, this would include the main memory of PLCs.

Non-Volatile memory is the persistent part of a component's memory. It consists of mass storage integrated in the PLC as well as additional memory cards.

B. Attacks on Cyber-physical Systems

In recent years, cyber-physical systems are subject to a growing number of attacks. In order to show the challenges when investigating cyber attacks, a brief review of these cyber attacks is necessary. While these attacks all focus on ICS, they carry wildly different implications for the forensic process. These implications are discussed in Section III.C.

1) Dragonfly (aka Energetic Bear)

Dragonfly was an attack aimed against energy suppliers [17]. While BlackEnergy disrupted, Dragonfly gathered information by infecting the targeted systems with the Dragonfly Remote Access Trojan horse (RAT). This remote access enabled attackers to use the specialized Havex malware on the systems. According to [18]: "*Havex used an OPC malware scanning module to gather information about ICS devices and send that data back to Command and Control (C&C) servers used by the Dragonfly group.*

The malware used an industrial protocol scanner to find networked devices on TCP ports 44818, 102 and 502. Automation companies such as Siemens and Rockwell Automation use these ports for ICS system communication. The industrial processes using the protocols are found in consumer goods manufacturing and packaging applications."

This quote shows that while this attack was aimed towards ICS, it did not touch the ICS in question. This attack played out solely above level 3 on the hierarchy of ICS discussed in Section II.C.

2) BlackEnergy

BlackEnergy started out as a popular crimeware (malware designed to automate criminal activities) [19] but is mostly known for its use during the cyber attacks on Ukrainian power grids in late 2015. These attacks are a prime example of the cyber kill chain [20]. The attackers used an approach with various, distinguishable steps [21]. In a preparation stage, access to the network was obtained by using spear-fishing attacks deploying the BlackEnergy 3 malware. An extensive reconnaissance stage followed, allowing the tailoring and planning of further actions. When the final stage of the attack started, the attackers sent the signal to open the power breakers, overwrote the firmware of serial-to-ethernet-converters (thereby destroying the link between SCADA and the PLC), disabled the uninterruptable power supply and, finally, wiped the hard drives of the SCADA systems. The last three steps aimed at making recovery from this cyber attack more difficult. However, the attack did not aim at 'hacking' the ICS in question - the attackers essentially took over the SCADA system, sending perfectly legitimate commands to the ICS in question. Hence, this attack can be described as a case of 'SCADA Hijacking' [22].

3) Stuxnet

Stuxnet is a major example for an advanced persistent threat (APT). This highly sophisticated attack followed a multi-step approach. This approach consisted of infecting systems in the target network, then infecting the programming environment for the ICS and finally, the ICS in question [23]. In essence, the final stage of the Stuxnet attack was the injection of malicious logic into the ICS code while using rootkit techniques to hide this logic from the programmer. The modified ICS code (containing the malicious logic, still hidden from the programmers' view) was then loaded on the ICS and executed - leading to the breakdown of actuators (permanent physical destruction).

4) PLC- Blaster

PLC-Blaster [24] is a malware executed directly on a PLC. This malware is a worm, which resides within a PLC and during execution scans the attached network for possible targets. Once targets have been identified, the malware infects the PLC in question. The infection takes place using network connections of the PLC. PLC-Blaster initiates a transfer of software to the PLC in question and basically updates the PLC with the malicious code. The implementation demonstrated in [24] shows various possible malicious functions, which might be implemented in order to illustrate the possible impact of such a PLC-resident malware.

C. The Nature of Forensic Investigations into ICS

These four examples given under Section III.B all have in common that all attacks have an impact on the security of ICS. However, while ICS have always been the target, they played varying roles in the attacks itself.

1) Dragonfly (aka Energetic Bear)

In attack 1 (Dragonfly) the ICS within the system was not even touched - there were no unauthorized, malicious messages or anything outside of the normal operating procedures. From the point of view of the ICS, no attack happened at all. The forensic implication is that the forensic investigation in this case has to concentrate on the surrounding IT-infrastructure. Thus, the gathering, investigating and analyzing data is covered by conventional Desktop-/Server-IT. The first challenge (as part of strategic preparation) is to provide access to network, main memory and mass storage data streams to a potential broad range of Desktop/Server IT Systems. The next challenge, in operational preparation, is to decide, which data streams from which systems need to be considered.

2) BlackEnergy

Attack 2 (BlackEnergy) is different in that, at the final stage, a malicious command was transmitted to the ICS in question. While the command itself seemed authentic to the ICS, it altered volatile (main) memory and caused actions. The forensic implication is the need to consider both, a conventional Desktop-IT system and an ICS, in this forensic investigation. Here the complex interchange using networks communication towards the ICS is the primary challenge. The mass storage, main memory and network data streams have to be gathered, investigated and analyzed using conventional Desktop-IT forensics. Additionally, access at least to the network data stream of the ICS system is necessary for the subsequent forensic investigation of the ICS system. The later involves non-standard data gathering, investigation and analysis techniques.

3) Stuxnet

Attack 3 (Stuxnet) went further - here the ICS in question was directly infected with malicious code. This not only altered the volatile memory and caused action, but also altered non-volatile (mass) memory. Similarly to Attack 2, the forensic implication is, that the forensic investigation in this case has to consider both a conventional Desktop-IT system and an ICS.

Additional to the access to mass storage, main memory and network data streams of the Desktop-IT system and the network data stream towards the ICS system, also access to the main memory and mass storage streams of the ICS are needed.

4) PLC-Blaster

Attack 4 (PLC-Blaster) was even more extreme in that only the PLCs were altered at all. Any attached control systems where totally left out of the loop. The forensic implication of this attack is, that forensic capabilities to gather, investigate and analyze data in ICS are necessary to detect the attack at all. This includes access to all data streams (mass storage, main memory, network), involving non-standard investigation techniques.

In order to clarify the scope of this article, it is necessary, to have a closer look on the forensic examinations to be performed to investigate these attacks.

1) Dragonfly (aka Energetic Bear)

In the case of attack 1 (Dragonfly), no forensic investigation into the ICS targeted by the attack takes place. Although, obtaining information on ICS is the ultimate goal of the attacker, all systems attacked belong to the classic IT domain. A forensic investigation would therefore take place in this domain. In addition, it would be utterly impossible to find any traces of such attacks inside the ICS systems in question.

2) BlackEnergy

Attack 2 (BlackEnergy) is a borderline scenario - again, all systems attacked belong to the classic IT domain. However, in this case, a command is transmitted to the ICS, leaving potential traces in the memory of the ICS.

3) Stuxnet

In this complex attack, various systems are attacked, including the ICS itself. Hence, the ICS is infected by malicious code, leaving traces.

4) PLC-Blaster

In attack 4 (PLC-Blaster) the PLC in question is the only component attacked and possibly containing forensic evidence.

As shown in these diverse steps needed to investigate these different attacks, all these attacks led to wildly different locations of possible traces during a forensic investigation. Table I maps these possible traces to the hierarchy levels in automation and the different data streams (see Section II.C for further information).

1) Dragonfly (aka Energetic Bear)

In this case, no traces are left on level 1 and level 2. Communication on level 3 and level 4 will be altered and could lead to possible traces.

2) BlackEnergy

BlackEnergy will cause traces on level 2, since volatile memory, non-volatile memory and communication of level 2 systems is altered. Level 3 and level 4 might offer traces of the spear fishing campaign used to access level 2. However, the PLC in question is not altered and it is very unlikely that it will contain any useful forensic traces if the communication (including the valid, but malicious, requests from the level 2 systems) is not completely captured.

3) Stuxnet

While Stuxnet is a highly advanced malware, it leaves (well-hidden) traces at anything it touches. Stuxnet alters the software on level 1 and level 2. Hence, volatile memory and non-volatile memory of the PLC in question offers another source of forensic data.

4) PLC-Blaster

PLC-Blaster only alters the PLC in question and the communication between various PLCs. Since level 2 is sometimes used in order to enable various PLCs to communication with each other, a complete capture of the communication on level 2 might serve as a source of forensic traces. However, the main source will be the volatile and non-volatile memory of the PLC in question.

These examples show the diversity of ICS-centered attacks and that these are often accompanied by attacks on classical IT systems. This is based on the fact that attackers first need to gain access to a system connected to the targeted ICS. If the ICS in question were, for example, directly connected to the internet (which, at the time of writing, happens dangerously often [25]), this step would be necessary. Hence, most forensic investigations in ICS will have points of contact with forensic investigations into classical IT in order to identify used attack vectors.

TABLE I. LOCATION OF POSSIBLE TRACES FOR DIFFERENT TYPES OF ICS-CENTERED ATTACKS

Attack	**Possible Traces in ...**			
	Level 1	***Level 2***	***Level 3***	***Level 4***
Dragonfly	no	no	Communication	Communication
BlackEnergy	Communication	Communication Volatile Memory Non-Volatile Memory	Communication Volatile Memory Non-Volatile Memory	Communication Volatile Memory Non-Volatile Memory
Stuxnet	Communication Volatile Memory Non-Volatile Memory	Communication Volatile Memory Non-Volatile Memory	Communication Volatile Memory Non-Volatile Memory	Communication Volatile Memory Non-Volatile Memory
PLC-Blaster	Communication Volatile Memory Non-Volatile Memory	Communication	no	No

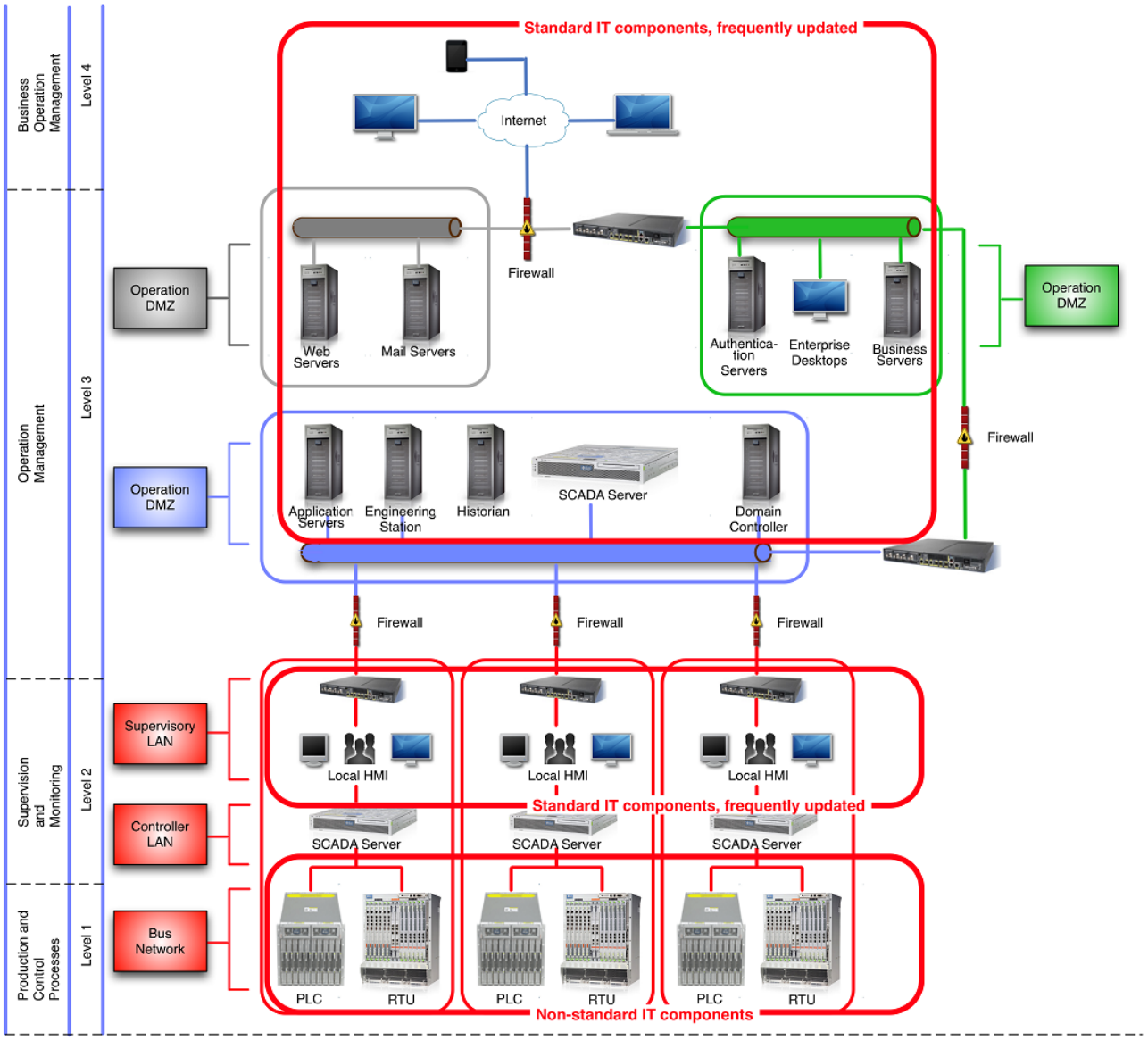


Figure 3. Relationship between ICS forensics and the traditional domain of digital forensics, according to [25]

This relation is shown in Fig. 3, taken from [26]. ICS are usually connected to classical IT environments. There is a clear overlap if ICS employs standard IT components for isolated solutions. Hence, ICS Forensics is a new forensics domain with strong ties and relations to the traditional (IT-) forensic domain.

D. The Nature of Forensics in Automotive Systems

The points discussed in the previous two sections hold also true for other instances of cyber-physical systems. This includes automotive systems, which have been the primary focus of previous work [1]. In the automotive domain, there are examples of direct attacks and manipulations on specific automotive components (e.g., odometer manipulation [27]), attacks on the classical IT systems forming the back-end in order to forge apparently authentic commands to the automotive components and approaches in between (e.g., the complex attacks demonstrated by [28] or [29]).

In this, automotive systems are similar to ICS and hence procedures learned for automotive forensics also hold true for the nascent field of ICS forensics.

E. Implications on Forensic Investigations in ICS

Various factors have implications on the forensic process regarding ICS. These originate from the nature of components, structures and processed used in ICS. Previous work [1] discussed the implications of automotive IT on the forensic process and some of these changes hold true for the ICS domain. In addition, related work ([26], [30], [31], [32], [33]) identifies some constraints. This section aims at summarizing these constraints and discusses the implications. These challenges include:

- The field devices usually have a low storage capacity. This includes storage capability for events, errors or log files. Sometimes fault codes are implemented in a ring buffer where older fault codes are frequently overwritten with newer ones - sometimes field devices do not have any logging mechanisms at all [33]. In addition, it has been found that on devices where extensive logging is supported, this feature is often disabled, or the devices lack sufficient capacity to store enough data to allow analysts to meet forensics requirements [26].
- The often process-critical nature of ICS makes it more unlikely that a system will be powered off for a forensic investigation [31]. This leads to a heavier emphasis on live forensics.
- The nature of communication (network traffic) in ICS differs from classical IT environments. While classical IT networks contain mainly user-generated traffic, SCADA traffic is routine and predictable [32]. In addition, the amount of traffic is - by modern standards - relatively low. This can simplify network forensics in ICS systems considerably.
- ICS systems are built to last for a long time without any update or upgrade. As [33] states: "*It is common for an ICS system to run for 20 or 30 years without update or upgrade*". This leads to an abundance of legacy hardware, which is connected over legacy communication systems. In consequence, ICS tend to be even more heterogeneous that automotive IT. This includes hardware, software, interfaces and communication protocols used. Hence, specialized knowledge is needed to access, obtain and analyze the data obtained in ICS. This knowledge is often hard to access, since vendors rely heavily on their intellectual property to protect their business - reverse engineering is a common (and time-consuming) occurrence in this field of forensics.
- ICS are not geared towards security but towards safety [26] [33]. Most mechanisms aim at achieving availability. This means, that authenticity-centered mechanisms are not that common in ICS. This needs to be kept in mind during forensic investigations.
- Access to mass storage is more complicated compared to Desktop IT. In Desktop IT, mass storage generally can be easily separated from the system under investigation and attached to a forensic workstation. Here, write-blockers are utilized to prevent all write-operations on the mass storage. This guarantees integrity of the data. In ICS mass storage is often part of the MCU silicon itself, rendering the access a very complex issue. However, sometimes the program executed by the PLC in question is stored on a removable memory card. This card can be removed and investigated using read-only hardware, achieving a similar result to classical Desktop IT in terms of integrity. In general, accessing the mass storage requires the stopping of the PLC in question. However, alterations at runtime in main memory of the PLC are possible, which, of course, are not reflected on the program stored on the removable memory card.
- Access to volatile memory is only possible by using built-in diagnostic functions. If such functions are available at all, using them carries a high structural impact. In addition, they are usually only able to access a very limited amount of the volatile memory of the PLC in question. These built-in diagnostic functions might be the target of attackers. In the case of Stuxnet [23], the attackers altered the diagnostic functions in order to deceive the operator (or a potential investigator) by only delivering back information about the volatile memory purged of any traces of potential wrongdoing.
- ICS might also control critical infrastructures [10] such as emergency services, traffic control or power generation. In such cases, the consideration for powering off the given ICS in order to perform a thorough forensic investigation will often favor keeping the systems in question active. On the other hand, there might be some procedures required by law in the case of an incident. This could be the reporting of critical events or suspected attacks.
- Some ICS might feature redundant systems. This could be implemented by performing calculations in multiple PLCs in order to detect failures and hence to increase robustness. Another possible approach would be the inclusion of fallback devices in case one PLC fails. This might allow for a forensic investigation into one of the PLCs while the fallback devices keep the system running.

The limitations discussed in the previous section have a strong impact on the forensic process employed in ICS environments. While related work ([26], [30], [31], [32],[33]) mostly focuses on the classical IT part of SCADA systems, this section will discuss the forensic process aimed at the specific field devices found on level 0 and level 1 of the automation levels.

F. Data Streams in ICS Forensics

As discussed in Section III.A, three data streams can be identified in field devices: *communication*, *volatile memory* and *mass storage*. Forensic traces can be extracted from these three data streams. After discussing the implications of an ICS architecture on forensics in general, it is worthwhile to investigate the impact on the gathering and interpretation of these three distinct data streams.

Communication investigations can be described as network forensics since it encompasses data transmitted over network. In an ICS environment, this network might be field bus systems or dedicated lines. Communication can only be observed at the moment it occurs. Hence, traces originating from the communication data stream can only be gathered during the moment the communication is performed. As [31] puts it: "*Network forensics cannot be performed without mechanisms that systematically capture relevant traffic and state information throughout the network*". The fundamental question is therefore how to access the carrier mediums of the communication in question and how to analyze the captured data. Accessing the carrier medium requires either physical or logical access to the carrier medium or one of the devices involved in the transmission. Physical access would mean tapping directly into the carrier medium, while logical access would imply a device attached to the medium forwarding the communication to the investigator (or, more likely, a tool employed by him). Physical access would be the preferred method, since it allows for a more thorough control of the integrity and authenticity of the capture communication. Also, capturing communication in this manner is a purely passive affair, causing no structural impact on the system in question.

The analysis of the captured communication is generally more complicated compared with a conventional IT environment. Various protocols with proprietary extensions are used, increasing the need of manufacturer cooperation or reverse engineering. On the other hand, the predictive nature and relatively low volume of communication eases the identification of untypical events in the communication stream.

Access to *main memory* in general is only possible by sending requests to the respective PLCs. The accessible data is limited by the diagnostic functions of those PLCs. These diagnostic functions might be extensive in theory but are usually very limited or not available at all. This type of data gathering carries the same implications as in Desktop IT - sending these requests alters the state of the system under investigation (structural impact). Hence, it alters the communication on the field bus system transferring the requests to (and the answer from) the PLC and the specific PLC. While these implications seem grave, it might still be worth acquiring this data when the investigators take these implications into account during the discussion of the conclusiveness of the traces. Hence, the investigator should have an idea of what specific data should be requested in order to keep these implications low and predictable.

Mass storage in ICS consists of mass storage integral to the PLC silicon itself and, optionally, additional memory cards. These memory cards are used to store the executable programs while the integral storage stores the runtime environment. Access to the integral memory in general is only possible by sending (diagnostic) requests to the respective PLC. This carries the same implications and limits as with using (diagnostic) requests to access the main memory: structural impact cannot be avoided as the data gathered is limited by the availability of diagnostic functions.

Memory cards can easily be removed and investigated using write-blockers in order to maintain the integrity of the trace in question. As noted, these memory cards usually contain the executable program in question as well information about the hardware configuration and the project the transferred program belongs to [34]. However, potential alterations to the program after its transfer to the internal main memory of the PLC are impossible to detect with this method.

A serious drawback is the fact that all access to mass storage in ICS environments is only possible if the PLC is in stop mode. This carries the drawbacks of post-mortem forensics (the respective PLC is not available for operation) without offering the increased protection of the integrity of the mass storage since most data gathering will still be performed by sending request to the PLC. It might be possible to circumvent stopping the complete ICS if the system in question is sufficiently redundant. This might allow for the investigation of single PLCs while the system, as a whole, stays operational.

IV. Survey of existing tools and their applicability to the forensic process in ICS

Forensic Investigations in ICS explore a new domain of forensics. They need to be supported by tools in order to gather, evaluate and analyze forensic traces. This section gives an overview of tools and approaches usable during forensic investigations into ICS systems. It discusses the merits and pitfalls of the tools and identifies additional measures needed in order to employ them in a forensically sound manner. While this collection of tools focuses mainly on Siemens systems, many of the observations can be transferred to tools designed to access other hardware.

This survey is structured along the lines of the forensic process detailed in Section II.A and aims at identifying means to acquire the forensic traces identified in Table 1.

C. Strategic Preparation (SP)

This phase represents measures taken by the operator of an IT-system, prior to an incident, which support a forensic investigation. These measures often increase the possibilities available to the investigator during Data Gathering.

The foundation of a forensic investigation is obtaining a deep understanding of the respective system. This includes collecting any documentation of the electronic and electrical system in question. Wiring schemes and electronic parts catalogues, as well as repair manuals, are a vital source of information to decide on further steps to strategically prepare for a forensic investigation. In addition, they can form a solid

foundation to make decisions during the later stages of a forensic investigation. This information is especially important during the Operational Preparation (OP).

A common type of SP is the activation of logging mechanisms already available in the specific components. While this requires only minor reconfiguration, further work might be needed to introduce adequate storage in order to record the resulting logs. This includes addressing the amount of data stored as well as supporting the integrity and authenticity of the data in question. The latter can be done by including cryptographic hashes and using a fixed, reliable time base for the generated logs.

More dedicated means of SP introduce interfaces to specific components that can be used during DG, if appropriate. This would include the installation of data taps for future access (Section IV.D discusses this topic in more detail).

SP also encompasses the installation of additional logging mechanisms geared towards the use during event reconstruction. One such example will now be presented in further detail.

1) Forensic Agents

The introduction of so-called 'Forensic Agents' into the SCADA architecture is discussed in [32]. This forensic agent represents data taps accessing the communication inside the SCADA networks. These taps are attached on level the levels 0, 1 and 2 of the standard Purdue Model.

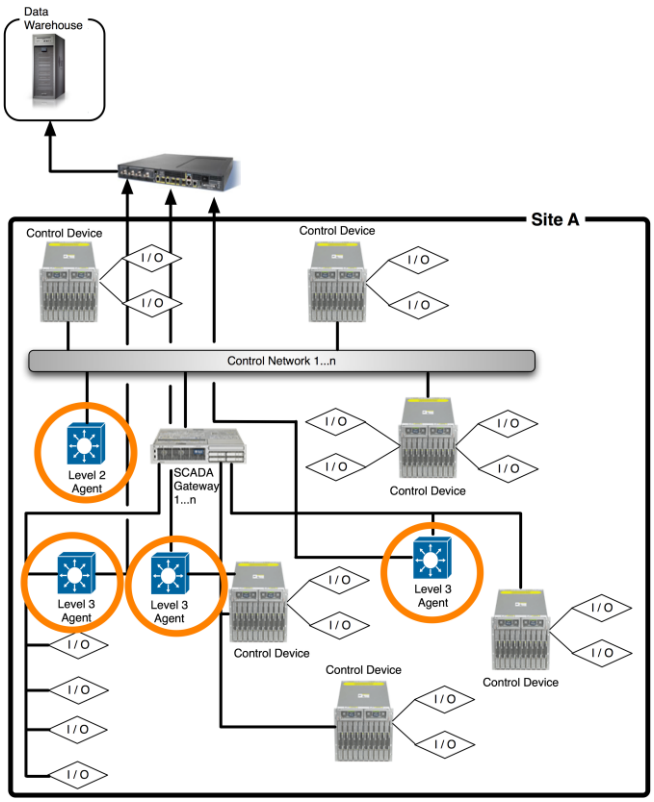


Figure 4. Placement of the Forensic Agents in Industrial Control Systems, according to [32]

Hence, they allow access to the communication inside the Cell/Area zone. The placement of these 'Forensic Agents' is shown in Fig. 4. The captured communication is then stored using data warehouse technology, making the events available for the use as forensic evidence. While this approach addresses the need for storage space, further refinement is needed in order to ensure authenticity and integrity of the respective captures.

This could be achieved by also storing cryptographic hashes generated over specific events (or timeframes) using an algorithm considered secure against collision attacks. Further, the inclusion of timestamps originating from a reliable time source is invaluable in event reconstruction.

Despite this additional effort, the introduction of forensic agents can greatly improve forensic capabilities.

In practice, a scaled down approach using only a subset of data taps or stores only a subset of events could also be viable.

Based on such agents it is advisable to implement means of intrusion detection. Such techniques could help to discover potentially anomalous system behavior as symptoms for initiating a forensic process. Moreover, the approach could be combined with the data reduction strategies, i.e., the complete traffic of all data taps is just recorded if it is justified by the symptom.

D. Operational Preparation (OP)

The Operational Preparation starts after an incident has been observed. In this phase, the fundamental decisions of the forensic investigation are made. The major question is whether to stop the suspect ICS or to keep it active. In an ICS environment, the system might have a critical function e.g. controlling critical infrastructure (emergency services, traffic control, power generation) or ensuring the safety of a plant environment. While in a classical desktop environment the decision on keeping the system productive or performing an extensive forensic investigation is often driven by the interest of stakeholders, ICS scenarios might well be influenced by the need of public safety. On the other hand, ICS scenarios are unlikely to contain much private data. Hence, they are less troublesome for the forensic investigator with regard to ensuring accordance to applicable data collection regulations and laws.

After the decision on keeping the system active or shutting it down (for forensic investigation and offline recovery), useful data sources are identified. As with the decision on the shutdown of the system, these considerations need to rely on the system understanding achieved during the SP. The availability of information on possible consequences as well as available traces greatly increases the ability to perform a well-informed OP.

In addition, there might be legal requirements to report incidents or suspected attacks on ICS systems. This might especially be the case in critical infrastructure domains.

E. Data Gathering (DG)

As discussed in Section III.F, three distinct data streams (*communication*, *volatile memory* and *non-volatile memory*) can provide traces for the forensic investigation.

Communication needs to be captured at the moment it occurs and requires physical access to the carrier in question. Given that access, there are tools for several of the communication bus systems used in ICS environments.

1) PBMaster

The PBMaster project offers an open software implementation of Profibus DP (Process Field Bus Decentralized Peripherals, see [11]) as presented in [35]. This project supports a wide range of hardware integrating UART (Universal Asynchronous Receiver/Transceiver) and RS-485 output. On the hardware-side it supports RS-232/RS-485 converters for Desktop computers, PCI based cards and ARM based boards. The software runs on Linux, FreeBSD, NetBSD and ARM based embedded systems. A Linux Live CD with all required components is provided as well. The project consists of a Profibus FDL master/slave station implementation, FDL/DP frame analyzer, FDL programming interface, a Live Linux CD and TCP/IP server for remote analysis of Profibus network traffic.

This tool does not provide any mechanisms to ensure integrity or authenticity of the gathered data. This can be addressed by using external mechanisms, like cryptographic hashes, to ensure authenticity and integrity of the gathered data. The passive reading access does not come with a structural impact.

However, this useful tool also serves as an example of the highly proprietary and intellectual property-protected tools in the ICS domain. Due to patent violations, distribution of the software is prohibited and limited to members of the Profibus International Organization and therefore not usable for an independent forensic investigator.

As discussed before, the access to *volatile memory* and *non-volatile memory* is indirect, since it relies on querying the PLC is question. In the following, two tools available to perform these queries for contents of *volatile* and *non-volatile memory* are presented.

1) NodeS7

NodeS7 [37] is a Node.js [36] library geared for communication with Siemens S7 PLCs. It provides functions to query values of variables as well as the possibility to write values. In order to work with the S7 1200 and 1500 series, an option called "Enable GET/PUT Access" must be set, which opens the PLC to third party software. This tool can be used to gather information about the current status of the PLC. Hence, it allows access to volatile and non-volatile memory.

To use it in a forensic environment, write operations should be disabled in order to minimize the structural impact. Additionally, no mechanisms to ensure integrity or authenticity of the gathered data are provided. This needs to be addressed by using external mechanisms.

2) Snap7

Snap7 [38] is an open source C++ suite for communication with S7 PLCs. The suite is able to read and write valuable information such as DataArea, DB, IPU, IPI, Merkers, timers, counters and variables. It can list, download, upload and delete blocks of data. Moreover, it is possible to retrieve detailed information on the PLC state, such as IDs, information on the CPU. It also offers the means to start and stop the PLC. The modification of this suite in a way that disables write operations would greatly increase the usefulness during forensic investigations. This could reduce structural impact. As with the other tools presented in this section, no mechanisms to ensure integrity or authenticity of the gathered data are provided. The integrity and authenticity of the gathered data needs to be addressed by using external mechanisms.

3) Soft-Update

Recent research into the behavior of Siemens S7 1516-F PN/DP PLCs lead to the discovery off an unknown behavior which might be useful for forensic investigations [39]. If the CPU of the PLC is in the 'RUN'-state during the loading of a new program, it briefly stops, loads the new program and restarts. During this process, the contents of the non-volatile memory are not overwritten. While this process has been shown in a proof of concept, a dedicated software solution for gathering this data is missing at the point of writing. Additional research into other PLCs might identify further PLCs where this approach of data gathering is applicable.

F. Data Investigation (DI)

Data Investigation represents measures to evaluate and extract data for further investigation. This includes data reduction and the identification of relevant data. It also includes the interpretation from raw data to (human-) readable information. In the classical Desktop domain, file reconstruction would be a prime example for a method used during DI. Another example is the dissection of captured raw communication. In general, tools that are able to 'make sense' (add semantics) of captured raw data by interpreting them are used in this section. One tool, that interprets captured raw data containing *communication* is presented and discussed here:

1) Wireshark S7Comm Dissector

The Wireshark S7Comm Dissector [40] is a tool able to interpret the raw communication between Siemens S7 PLC, the attached HMIs and the control system (the TIA - Totally Integrated Automation Portal, see [41]). This communication relies on a proprietary protocol using ISO-on-TCP packets. The application of this tool allows gaining meaningful insight into the communication between ICS components. While it is always best to use tools on copies of the captured traces, this tool performs no unannounced modifications of the captured trace. However, for a sound forensic process, the usage of external means to ensure integrity and authenticity of the processed traces is strongly advised.

The tool itself comes as a plug-in for the Wireshark [42] network dissector. The dissector for the S7Comm protocol, which is used by older models of the S7-300 and S7-400 series, is included in current Wireshark versions. The dissector for the newer S7 Comm Plus protocol, used by the newer series S7-1200 and S7-1500, needs to be installed as a plug-in (for Windows) or compiled with Wireshark itself (for unix-based Systems).

G. Data Analysis (DA)

Data Analysis brings the different traces gathered in a forensic investigation together. In this step, information is aggregated, correlations found and chains of events identified. While dedicated tools for the use in ICS environments are missing, some of the more generalized tools from the Desktop IT domain can be adopted for use during an investigation into ICS. This refers to tools, which help to create and organize chains of events, like Zeitline (see [43]).

In general, violating the authenticity and integrity of the traces processed in this step can be avoided by using copies of the original traces. The authenticity and integrity of the achieved results should be ensured. While Zeitline has functionality for this, most methods might require the use of external tools to achieve authenticity and integrity.

H. Documentation (DO)

The documentation consists of two parts. First, there is the process of accompanying documentation, which maintains an account of all the actions taken by the examiners. This process should ideally be highly assisted by software, recording all parameters and selected menu items. For desktop IT a range of dedicated IT forensic suites exist (e.g., X-Ways forensics [44]). In the field of non-standard forensic environments such a tool is missing. Hence, the investigator needs to rely on a mostly manual process involving screenshots, digital photographs, etc.

The results of the investigation are then compiled to a final examination report. This report describes the examination process and the results as well the most likely chain of events according to the reconstruction from traces. Usually, no dedicated tools are used for this process - besides a word processor.

V. Design recommendations for future Tools aimed towards forensics in Non-Standard IT environments

As depicted in the prior section and in previous work [1], there is a lack of tools geared towards the use in forensic investigations into non-standard IT. Major challenges are the heterogeneity of the domain (including hardware, software and communication protocols) and the reliance on proprietary and intellectual property-protected solutions.

Major work needs to be done to develop usable interfaces to access communication, volatile and non-volatile memory in order to acquire the forensic traces as identified in Table 1.

Especially for mass storage data streams, i.e., non-volatile memory, on levels 1 and 2 access today is either downright impossible (debug fuses set) or incomplete (e.g., only access to external memory chips) or at best very difficult using debug interfaces with undocumented parameterization and protocols (e.g., JTAG). Full access to the non-volatile memory should be provided, preferably using a serial high-speed interface and measures to ensure integrity and authenticity ensured using up to date cryptographic techniques. The same requirements should be placed towards the data gathering on volatile memory. Additionally, due to the volatile nature of the main memory content, a measure to halt the CPU register and RAM states (e.g., using non maskable interrupts pointing towards an integrity and authenticity ensuring dump routine) would add a new descriptive power to the traces gathered in memory forensics.

Further work is needed in order to interpret the traces gathered from these data streams with regards to semantics in order to support event reconstruction in more detail. This applies to both data investigation (allowing for a data reduction by excluding case irrelevant data) and data analysis (supporting the piecing together of the traces within the respective data stream to get a global picture of events).

For some isolated solutions, tools are available. However, these tools are not geared towards usage in forensic scenarios. Specialized solutions are needed here.

Previous work [1] already discussed criteria for the design of future forensic tools. Further considerations can be found in [45], giving the following requirements:

- the collected/processed data should be useful for the forensic process
- ensure the integrity, authenticity and confidentiality of the collected/processed data
- have a minimized and well-known structural impact, ensuring the integrity of the source data as best as possible
- document the actions performed
- the frequency of possible errors during processing should be known.

VI. Conclusion

This article presents the challenges of forensic investigation into potential security incidents in non-standard IT on the example of ICS and automotive environments. The growing interconnectivity in this domain comes at the price of an increased number of incidents - some of them caused by malicious attacks. This carries the need for forensic investigations into these incidents.

However, this article shows that forensic investigations in ICS environments still have significant shortcomings. The field is hampered by a severe lack of adequate tools owing to the heterogeneity of the ICS domain and the high barriers laid out by proprietary and intellectual property-protected solutions prevalent in ICS.

Approaches from the classical Desktop-IT domain can be adapted in order to preserve authenticity and integrity of the forensic evidence used during an investigation. The same holds true for the need of documentation of the whole

forensic process. Preferably, the tools themselves support the investigator in retaining authenticity and integrity while achieving a thorough documentation. Lacking that, tactics from the classic IT domain, which do not rely on internal tool support, can be applied.

In addition, the protection of personal data in accordance to applicable regulations and laws as well as adhering to regulations concerning the collection of data, especially in consideration of privacy laws and human rights, cannot be solved by the usage of tools alone - strong policies for the gathering and use of forensic evidence are needed.

The main contribution of this paper is the identification of 'white spots' where tailored and adequate solutions are needed in order to perform forensic investigations and giving guidance on the creation of such tailored solutions.

ACKNOWLEDGMENT

This document was produced with the financial assistance of the European Union. The views expressed herein can in no way be taken to reflect the official opinion of the European Union.

In addition, we like to thank our students working on automotive and industrial forensic topics.

REFERENCES

[1] R. Altschaffel, K. Lamshöft, S. Kiltz and J. Dittmann, "A Survey on Open Automotive Forensics", Securware 2017, 2017.

[2] European Union Agency for Network and Information Securit y, "Communication network dependencies for ICS/SCADA Systems", ISBN: 978 - 92 - 9204 - 192 - 2, doi: 10.2824/397676, 2016.

[3] K. Inman and N. Rudin, "Principles and Practises of Criminalistics: The Profession of Forensic Science," CRC Press LLC Boca Raton Florida, USA, ISBN 0-8493-8127-4, 2001.

[4] M. Pollit, "Applying Traditional Forensic Taxonomy to Digial Forensics," IFIP International Federation for Information Processing, Volume 258; Advances in Digital Forensics IV, pp. 17-26, DOI: 10.1007/978-0-387-84927-0_2, 2008.

[5] S. Kiltz, J. Dittmann, and C. Vielhauer, "Supporting Forensic Design - a Course Profile to Teach Forensics," IMF 2015.

[6] S. Peisert, M. Bishop and K. Marzullo, "Computer forensics in forensis", In SIGOPS Operating Systems Review, Volume 42, Issue 3, pp 112-122, ACM, DOI=10.1145/1368506.1368521, 2008.

[7] T. Sugimura, "Junction Blocks Simplify and Decrease Networks When Matched to ECU and Wire Harness," Encyclopedia of Automotive Engineering. 1–7.

[8] A. Hillier, "Hillier's Fundamentals of Automotive Electronics Book 2 Sixth Edition," Oxford University Press, 2014.

[9] Robert Bosch GmbH, "CAN Specification 2.0, 1991" http://esd.cs.ucr.edu/webres/can20.pdf, (18/05/2018), 1991.

[10] Presidential Policy Directive/PPD-21, "Critical Infrastructure Security and Resilience," https://obamawhitehouse.archives.gov/the-press-office/2013/02/12/presidential-policy-directive-critical-infrastructure-security-and-resil (23/05/2018), 2013.

[11] PROFIBUS and PROFINET International: "PROFIBUS," https://www.profibus.com/technology/profibus/ (23/05/2018), 2018.

[12] PROFIBUS and PROFINET International: "PROFINET," https://www.profibus.com/technology/profinet/ (23/05/2018), 2018.

[13] Modbus Organisation: "Modbus," http://www.modbus.org/ (23/05/2018), 2018.

[14] T. J. Williams: "The Purdue enterprise reference architecture: a technical guide for CIM planning and implementation," Research Triangle Park, NC: Instrument Society of America, 1992.

[15] SANS Institute: "Secure Architecture for Industrial Control Systems," https://www.sans.org/reading-room/whitepapers/ICS/secure-architecture-industrial-control-systems-36327 (18/05/2018), 2015.

[16] T. J. Williams: "The Purdue enterprise reference architecture," Computers in industry Vol 24 (2). p. 141-158. 1994.

[17] Symantic: "Dragonfly: Cyberespionage Attacks Against Energy Suppliers," https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/Dragonfly_Threat_Against_Western_Energy_Suppliers.pdf (18/05/2018), 2014.

[18] N. Nelson: "The Impact of Dragonfly Malware on Industrial Control Systems," https://www.sans.org/reading-room/whitepapers/ICS/impact-dragonfly-malware-industrial-control-systems-36672 (18/05/2018), 2016.

[19] F-SECURE LABS: "BLACKENERGY & QUEDAGH - The convergence of crimeware and APT attacks," https://www.f-secure.com/documents/996508/1030745/blackenergy_whitepaper.pdf (23/05/2018), 2014.

[20] LOCKHEED MARTIN: "The Cyber Kill Chain®," https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (23/05/2018), 2015.

[21] P. A.L. Ducheine, J. van Haaster, R. van Harskamp: "Manoeuvering and Generating Effects in the information Environment," in Netherlands Annual Review of Military Studies 2017: Winning Without Killing, 2017.

[22] R. M. Lee, M. J. Assante, T. Conway: "Analysis of the Cyber Attack on the Ukrainian Power Grid," https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf (23/05/2018), 2016.

[23] N. Falliere, L. O Murchu, E. Chien: "W32.Stuxnet Dossier," https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf (18/05/2018), 2011.

[24] Ralf Spenneberg, Maik Brüggemann, Hendrik Schwartke: " PLC-Blaster: A Worm Living Solely in the PLC," https://www.blackhat.com/docs/us-16/materials/us-16-Spenneberg-PLC-Blaster-A-Worm-Living-Solely-In-The-PLC-wp.pdf (23/05/2018), 2016.

[25] S. Gallagher: "Vulnerable industrial controls directly connected to Internet? Why not?," https://arstechnica.com/information-technology/2018/01/the-internet-of-omg-vulnerable-factory-and-power-grid-controls-on-internet/ (23/05/2018), 2018.

[26] M. Fabro, E. Cornelius: "Recommended Practice: Creating Cyber Forensics Plans for Control Systems," https://ics-cert.us-cert.gov/sites/default/files/recommended_practices/Forensics_RP.pdf (23/05/2018), 2008.

[27] European Parliament: "Odometer manipulation in motor vehicles in the EU," http://www.europarl.europa.eu/RegData/etudes/STUD/2018/615637/EPRS_STU(2018)615637_EN.pdf (29/05/2018), 2018.

[28] C. Miller, C. Valesek: "Remote Exploitation of an Unaltered Passenger Vehicle", Black Hat USA, 2015.

[29] Keenlab: "Experimental Security Assessment of BMW Cars: A Summary Report," https://keenlab.tencent.com/en/Experimental_Security_Assessment_of_BMW_Cars_by_KeenLab.pdf (29/05/2018), 2018.

[30] T. Spyridopoulos, T. Tryfonas, J. May: "Incident Analysis & Digital Forensics in SCADA and Industrial Control Systems," 2013.

[31] J. Stirland, K. Jones, J. Janicke, T. Wu: "Developing Cyber Forensics for SCADA Industrial Control Systems," 2014.

[32] T. Kilpatrick, J. Gonzalez, R. Chandia, M. Papa, S. Shenoi: "An Architecture for SCADA Network Forensics," 2006.

[33] P. Van Vliet, M-T. Kechadi, Nhien-An Le-Khac :"Forensics in Industrial Control System: A Case Study," https://arxiv.org/ftp/arxiv/papers/1611/1611.01754.pdf (23/05/2018), 2016.

[34] Siemens: „Structure and Use of the CPU Memory,"https://cache.industry.siemens.com/dl/files/101/59193101/att_897341/v1/s71500_structure_and_use_of_the_PLC_memory_function_manual_en-US_en-US.pdf (29/05/18), 2016.

[35] D. K. Tran, P. Pisa, P. Smolik: "An Open Implementation of Profibus DP," https://static.lwn.net/images/conf/rtlws11/papers/proc/p29.pdf (23/05/2018), 2009.

[36] Node.js, https://nodejs.org/en/ (23/05/2018).

[37] node7s, https://github.com/plcpeople/nodeS7 (23/05/2018).

[38] snap7, http://snap7.sourceforge.net/ (23/05/2018).

[39] (german) Oliver Keil: "Forensik in Automatisierungsystemen - Konzept zur Identifation und Erhebung verschiedener Datenquellen," Bachelor Thesis at Otto-von-Guericke University Magdeburg, 2018.

[40] s7commwireshark, https://sourceforge.net/projects/s7commwireshark/ (23/05/2018).

[41] Siemens: "Your gateway to automation in the Digital Enterprise Totally Integrated Automation Portal," https://c4b.gss.siemens.com/resources/images/articles/dffa-b10161-00-7600.pdf (23/05/2018), 2016.

[42] wireshark: https://www.wireshark.org/ (23/05/2018).

[43] F. Buchholz, C. Falk: "Design and Implementation of Zeitline: A Forensic Timeline Editor," http://www.dfrws.org/sites/default/files/session-files/paper-design_and_implementation_of_zeitline_-_a_forensic_timeline_editor.pdf (23/05/2018), 2005.

[44] X-Ways Software Technology AG, "X-Ways Forensics: Integrated Computer Forensics Software," http://www.x-ways.net/forensics/ (23/05/2018).

[45] M. Hildebrandt, S. Kiltz, J. Dittmann: "A Common Scheme for Evaluation of Forensic Software," In Proceedings of the 6th International Conference on IT Security Incident Management and IT Forensics (IMF2011) Stuttgart, Germany, 10.05.-12.05.2011, ISBN 978-0-7695-4403-8, pp. 92-106, 2011.

Proposal and Study on Implementation of Data Eavesdropping Protection Method over Multipath TCP Communication Using Data Scrambling and Path Dispersion

Toshihiko Kato[1)2)], Shihan Cheng[1)], Ryo Yamamoto[1)], Satoshi Ohzahata[1)] and Nobuo Suzuki[2)]
1) Graduate School of Informatics and Engineering
University of Electro-Communications
Tokyo, Japan
2) Adaptive Communications Research Laboratories
Advanced Telecommunication Research Institute International
Kyoto, Japan
kato@is.uec.ac.jp, chengshihan@net.is.uec.ac.jp, ryo_yamamotog@is.uec.ac.jp,
ohzahata@is.uec.ac.jp, nu-suzuki@atr.jp

***Abstract*— Recent mobile terminals have multiple interfaces, such as 4G and wireless local area network (WLAN). In order to use those interfaces at the same time, multipath transmission control protocol (MPTCP) is introduced in several operating systems. However, it is possible that some interfaces are connected to untrusted networks and that data transferred over them is observed in an unauthorized way. In order to avoid this situation, we proposed, in our previous paper, a new method to improve privacy against eavesdropping using the data dispersion by exploiting the multipath nature of MPTCP. One feature of the proposed method is to realize that an attacker cannot observe data on any path, even if he observes traffic over only a part of paths. Another feature is to use data scrambling instead of ciphering. In this paper, we present the design of this method and the results of performance evaluation. Besides, we discuss how to implement it inside the Linux operating system kernel, using a kernel debugging mechanism called JProbe.**

Keywords- Multipath TCP; Eavesdropping; Data Dispersion; Data Scrambling; JProbe.

I. INTRODUCTION

This paper is an extension of our previous paper [1], which was presented in an IARIA conference.

Recently, mobile terminals with multiple interfaces have come to be widely used. For example, most smart phones are equipped with interfaces for 4G Long Term Evolution (LTE) and WLAN. In the next generation (5G) network, it is studied that multiple communication paths provided multiple network operators are commonly involved [2]. In this case, mobile terminals will have more than two interfaces at the same time.

In order for applications to use multiple interfaces effectively, MPTCP [3] is being introduced in several operating systems, such as Linux, Apple OS/iOS [4] and Android [5]. MPTCP is an extension of TCP, and provides multiple byte streams through different interfaces. It is designed so as for conventional TCP applications to use MPTCP as if they were working over traditional TCP.

MPTCP is specified in three request for comments (RFC) documents provided by the Internet Engineering Task Force. RFC 6182 [6] outlines architecture guidelines for developing MPTCP protocols, by discussing the high level design decisions on selecting the protocol functions from multiple candidates. RFC 6824 [7] presents the details of extensions to the traditional TCP to support multipath operation. It defines the MPTCP control information realized as new TCP options, and the MPTCP protocol procedures for the initiation and association of subflows (TCP connections related with an MPTCP connection), the data transfer and acknowledgment over multiple subflows, and the closing MPTCP connection. RFC 6356 [8] presents a congestion control algorithm that couples the congestion control algorithms running on different subflows.

When a mobile terminal uses multiple interfaces, i.e., multiple paths, some of them may be unsafe such that an attacker is able to observe data over them in an unauthorized way. For example, a WLAN interface is connected to a public WLAN access point without any encryption at the WLAN level, data transferred over this WLAN may be disposed to other nodes connected to it. In order to prevent this eavesdropping, the transport layer security (TLS) is used to provide communication security. Although TLS can be applied to various applications including web access, e-mail and ftp, however, it is widely used only with HTTP, and some applications like VoIP cannot use TLS.

In order to avoid this eavesdropping, we proposed another approach to improve privacy against eavesdropping by exploiting the multipath nature of MPTCP. We called this approach a *not-every-not-any protection*, in our previous paper [1]. Even if an unsafe WLAN path is used, another path may be safe, such as LTE supported by a trusted network operator. So, the proposed method is such that if an attacker cannot observe the data on *every* path, he cannot observe the traffic on *any* path [9]. The feature of the proposed method is to adopt the not-every-not-any protection, and to use the data scrambling instead of ciphering.

In this paper, we present the proposed method in detail, and show the results of processing overhead of the data scrambling/descrambling in the proposed method, and the conventional encryption/decryption methods. We also discuss a study on how the proposed method is implemented in the MPTCP program inside the Linux operating system kernel.

The rest of this paper is organized as follows. Section II explains the overview [10] and the security issues of MPTCP. Section III describes the design of the proposed method protecting against eavesdropping. Section IV gives the performance evaluation on the processing overhead of the proposal method and other ciphering methods. Basically, the content in Sections II through IV comes from our previous paper [1]. Section V shows a study on how to implement the proposed method inside the Linux operating system kernel. In the end, Section VI concludes this paper.

II. Overview and Security Issues of MPTCP

A. MPTCP connections and subflows

As described in Figure 1, the MPTCP module is located on top of TCP. As described above, MPTCP is designed so that the conventional applications do not need to care about the existence of MPTCP. MPTCP establishes an *MPTCP connection* associated with two or more regular TCP connections called *subflows*. The management and data transfer over an MPTCP connection is done by newly introduced TCP options for MPTCP operation.

Figure 2 shows an example of MPTCP connection establishment where host A with two network interfaces invokes this sequence for host B with one network interface. In the beginning, host A sends a SYN segment to host B with a *Multipath Capable (MP_CAPABLE)* TCP option. This option indicates that an initiator supports the MPTCP functions and requests to use them in this TCP connection. It contains host A's *Key* (64 bits) used by this MPTCP connection. Then, host B replies a SYN+ACK segment with MP_CAPABLE option with host B's Key. This reply means that host B accepts the use of MPTCP functions. In the end, host A sends an ACK segment with MP_CAPABLE option including both A's and B's Keys. Through this three-way handshake procedure, the first subflow and the MPTCP connection are established. Here, it should be mentioned that these "Keys" are not keys in a cryptographic sense. As described below, they are used for generating the Hash-based Message Authentication Code (HMAC), but MPTCP does not provide any mechanisms to protect them from attackers' accessing while transfer.

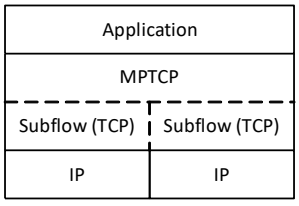


Figure 1. Layer structure of MPTCP.

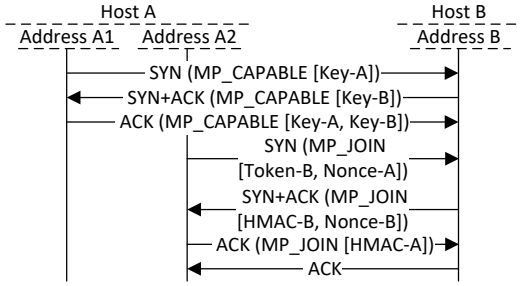


Figure 2. Example of MPTCP connection establishment.

Next, host A tries to establish the second subflow through another network interface. In the first SYN segment in this try, another TCP option called a *Join Connection (MP_JOIN)* option is used. An MP_JOIN option contains the receiver's *Token* (32 bits) and the sender's *Nonce* (random number, 32 bit). A Token is an information to identify the MPTCP connection to be joined. It is obtained by taking the most significant 32 bits from the SHA-1 hash value for the receiver's Key (host B's Key in this example). Then, host B replies a SYN+ACK segment with MP_JOIN option. In this case, MP_JOIN option contains the random number of host B and the most significant 64 bits of the HMAC value. An HMAC value is calculated for the nonces generated by hosts A and B using the Keys of A and B. In the third ACK segment, host A sends an MP_JOIN option containing host A's full HMAC value (160 bits). In the end, host B acknowledges the third ACK segment. Using these sequence, the newly established subflow is associated with the MPTCP connection.

B. Data transfer

An MPTCP implementation will take one input data stream from an application, and split it into one or more subflows, with sufficient control information to allow it to be reassembled and delivered to the receiver side application reliably and in order. The MPTCP connection maintains the *data sequence number* independent of the subflow level sequence numbers. The data and ACK segments may contain a *Data Sequence Signal (DSS)* option depicted in Figure 3.

The data sequence number and data ACK is 4 or 8 byte long, depending on the flags in the option. The number is assigned on a byte-by-byte basis similarly with the TCP sequence number. The value of data sequence number is the number assigned to the first byte conveyed in that TCP segment. The data sequence number, subflow sequence number (relative value) and data-level length define the mapping between the MPTCP connection level and the subflow level. The data ACK is analogous to the behavior of the standard TCP cumulative ACK. It specifies the next data sequence number a receiver expects to receive.

C. Security issues on MPTCP and related work

Some new security issues emerge by the introduction of MPTCP [8]. One is a new threat that an attacker splits malicious data over multiple paths. Traditional signature-based intrusion detection systems (IDSs) suppose that they can monitor all packets of a given flow. If a target system uses

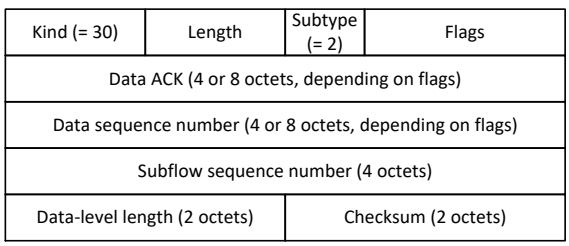


Figure 3. Data Sequence Signal option.

MPTCP and an attacker sends signatures over different subflows, IDSs cannot detect them. Ma et al. [11] proposed a new approach for this problem, where each IDS locally scans and processes its monitored traffic, and all IDSs share asynchronously a global state of string matching automaton.

Another issue is related to MPTCP and privacy. MPTCP has a potential to provide improved privacy against attackers who are able to observe or interfere with subflow traffic along a subset of paths. Dispersing traffic over multiple paths makes it less likely that attackers will get access to all of the data. Pearce and Zeadally [9] suggested the concept of the not-every-not-any protection and introduced some ideas including sending cryptographic signing details using multiple paths and applying cryptographic chaining, such as cipher block chaining (CBC), across multiple paths.

There have been several proposals on the data dispersion over multiple paths. Yang and Papavassiliou [12] provided a method to analyze the security performance when a virtual connection takes multiple disjoint paths to the destination, and a traffic dispersion scheme to minimize the information leakage when some of the intermediate routers are attacked. Nacher et al. [13] tried to determine the optimal trade-off between traffic dispersion and TCP performance over mobile ad-hoc networks to reduce the chances of successful eavesdropping while maintaining acceptable throughput. These two studies use multiple TCP connections by their own coordination methods instead of MPTCP. Gurtov and Polishchuk [14] used host identity protocol (HIP), which locates between IP and TCP to provide multiple paths, and propose how to spread traffic over them. Apiecionek et al. [15] proposed a way to use MPTCP for more secure data transfer. After data are encrypted, they are divided into blocks, mixed in the predetermined random sequence, and then transferred through multiple MPTCP subflows. A receiver rearranges received blocks in right order and decrypts them.

All of those proposals aim at just spreading data packets over multiple paths, and do not consider the coordination over multiple paths. If the transferred data are encrypted before dispersion, it can be said that they are coordinated by the encryption procedure, but the coordination is not realized by the dispersion schemes. In contrast with them, our proposal adopts an approach to improve privacy by coordinating data over multiple paths through data scrambling not encryption.

III. PROPOSAL

A. Requirements and possible approaches

The following are the requirements for designing a not-any-not-every protection method protecting eavesdropping.

- The method needs to cope with two way data exchanges within one MPTCP connection.
- The length of exchanged data should not be expanded.
- Even if there are any bytes with known values, such as fixed bytes in an application protocol header, the method provides protection from information leakage.
- The method does not introduce any new overheads into MPTCP as much as possible.
- The method does not change the behaviors of MPTCP as much as possible.

In designing the proposed method, we have considered the following possible candidates.

(1) Secret sharing method

The secret sharing method is to divide data *D* into *n* pieces in such a way that *D* is easily reconstructed from any *k* pieces, but even complete knowledge of *k* -1 pieces reveals absolutely no information about *D* [16]. Shamir [16] gave an example method based on polynomial interpolation. It is possible to apply the idea of secret sharing to data transfer. Zhao et al. [17] proposed an efficient anonymous message submission protocol based on secret sharing and a symmetric key cryptosystem. It aggregates messages of multiple members into a message vector such that a member knows only his own position in the submission sequence.

Figure 4 shows an idea of applying secret sharing to the eavesdropping protection. It supposes the case that $n = 2$ and $k = 2$. Pieces D_1 and D_2 are generated from an original data and transferred through different paths. An attacker can access only D_2 over an untrusted path, and so he cannot obtain the original data. In this approach, however, the amount of transferred data is increased, twice in this example.

(2) Network coding

The second candidate is the network coding [18]. In this framework, the exclusive OR (XOR) is calculated among multiple packets and the result is transferred instead of packets themselves. Ahlswede et al. [18] mentioned that by employing coding at network nodes, which they referred to as network coding, it is possible to save bandwidth in general. Li et al. [19] proposed a network coding based multipath TCP (NC-MPTCP), which uses the mix of regular subflows, delivering original data, and network coding subflows, which deliver linear combinations of original data. NC-MPTCP

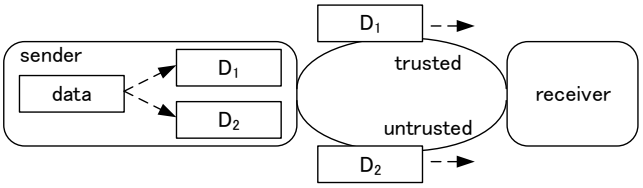


Figure 4. Secret sharing based approach.

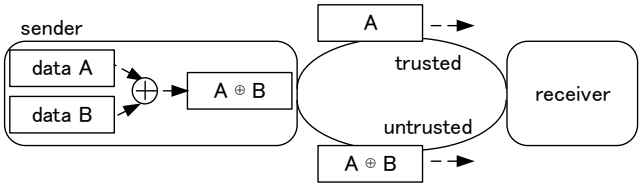


Figure 5. Network coding based approach.

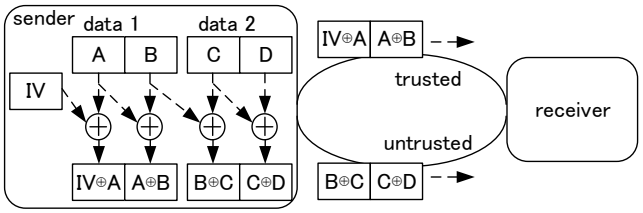


Figure 6. Block ciphering based approach.

achieves higher goodput compared to MPTCP in the presence of different subflow qualities.

Figure 5 shows an idea of applying network coding to the eavesdropping protection. Using data A and B, their XOR (A ⊕ B) is calculated. Through a trusted path, an original data A is transferred, and through an untrusted path, A ⊕ B is transferred. Since an attacker observes only A ⊕ B, he cannot obtain data B without knowledge of data A. This idea can be said a *packet level data scrambling*. Although it can provide the not-every-not-any protection, it introduces an additional overhead due to the variable length packets, and an additional control in MPTCP, such as sending XOR data only over an untrusted path.

(3) Mode of operation in block ciphering

The third candidate is the mode of operation, such as CBC and output feedback (OFB), used in block ciphering [20]. The block cipher defines only how to encrypt or decrypt a fixed length bits (block). A mode of operation defines how to apply this operation to data longer than a block. CBR and OFB introduce a chaining between blocks such that a block is combined with the preceding block by XOR calculation.

Figure 6 shows an idea of applying mode of operation to the eavesdropping protection. Data to be sent (data 1 and 2) are divided into blocks (A through D). The first block is XORed with the initialization vector (IV), and the following blocks are XORed with their preceding blocks. The XORed results are transferred via different paths. In the example, an attacker can only observe B ⊕ C and C ⊕ D, and does not know block B, which is transferred through a trusted path. So, he cannot obtain C and D any more. This idea can be said a *block level data scrambling*. Although it can provide the not-every-not-any protection, it introduces an additional data overhead because the length of packets is not integral multiple of block length in general.

According to those considerations, we select a byte stream based data scrambling approach described below that avoids the issues of the approaches described so far.

B. Detailed design of proposed method

As shown in Figure 7, we introduce a data scrambling function within MPTCP and on top of the original MPTCP. When an MPTCP communication is started, the use of data scrambling is negotiated. It may be done using a flag bit in MP_CAPABLE TCP option.

Figure 8 shows an overview of data scrambling. In the data sending side, an application sends data to MPTCP. It is stored in the send socket buffer, and the data scrambling module scrambles it in a byte-by-byte basis. The result is stored in the send socket buffer again. The data in this buffer is transferred reliably by MPTCP. While sending data, MPTCP tries to send the first packet over an MPTCP connection via a subflow that uses a trusted path. After that, the data transfer by MPTCP is performed according to its native scheduler. We suppose that the distinction of trusted or untrusted path can be done by the IP address of interfaces. In the data receiving side, data is transferred through MPTCP without any losses, transmission errors, nor duplications. The received in-sequence data is stored in the receive socket buffer.

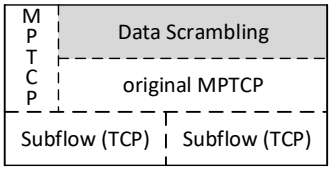
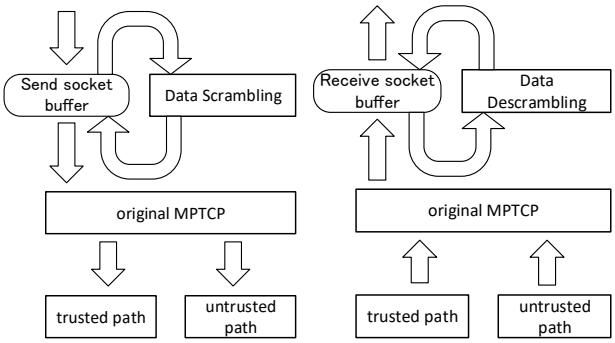


Figure 7. Layer structure of MPTCP with data scrambling.



(a) Sending data (b) Receiving data

Figure 8. Overview of data scrambling processing.

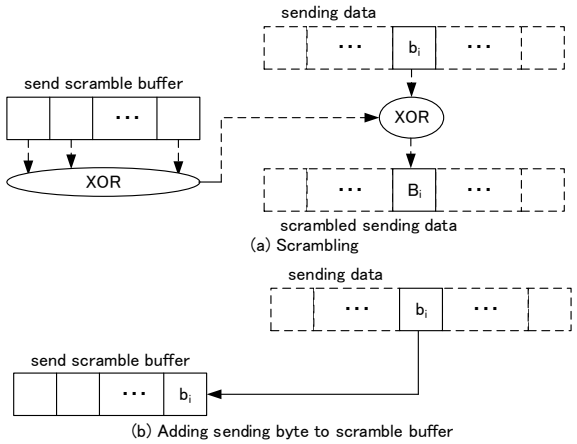


Figure 9. Procedure of data scrambling.

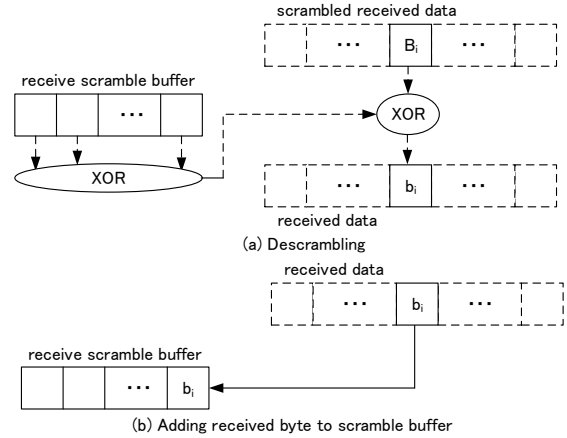


Figure 10. Procedure of data descrambling.

After that, the data descrambling module is invoked to restore the scrambled data to the original one.

Figure 9 shows the details of data scrambling. As described above, the scrambling is performed in a byte-by-

byte basis. More specifically, one byte being sent is XORed with its preceding 64 bytes. In order to realize this scrambling, the data scrambling module maintains the *send scrambling buffer*, whose length is 64 bytes. It is a shift buffer and its initial value is the Key of this side. Since the length of the Key is 8 bytes, the higher bytes in the send scrambling buffer is filled by zero. When a data comes from an application, each byte (b_i in the figure) is XORed with the result of XOR of all the bytes in the send scrambling buffer. The obtained byte (B_i) is the corresponding sending byte. After calculating the sending byte, the original byte (b_i) is added to the send scramble buffer, forcing out the oldest (highest) byte from the buffer. The send scrambling buffer holds recent 64 original bytes given from an application. By using 64 byte buffer, the access to the original data is protected even if there are well-known byte patterns (up to 63 bytes) in application protocol data.

Figure 10 shows the details of data descrambling, which is similar with data scrambling. The data scrambling module also maintains the receive scramble buffer whose length is 64 bytes. Its initial value is HMAC of the key of the remote side. When an in-sequence data is stored in the receive socket buffer, a byte (B_i that is scrambled) is applied to XOR calculation with the XOR result of all bytes in the receive scramble buffer. The result is the descrambled byte (b_i), which is added to the receive scramble buffer.

By using the byte-wise scrambling and descrambling, the proposed method does not increase the length of exchanged data at all. The separate send and receive control enables two way data exchanges to be handled independently. Moreover the proposed method introduces only a few modification to the original MPTCP.

C. Discussions

We need to discuss here about the security scheme of the proposed method. The proposed method does not use the data ciphering, and so it does not protect eavesdropping in a strict sense. It depends on the difficulty of unauthorized data access over networks provided by trusted operators. That is, the intrusion model is that an attacker can access only untrusted networks, such as public access point based WLANs, but he/she cannot access to trusted networks.

We also need to point out that the proposed method gives a small modification to MPTCP. It uses the HMAC value of sender side Key as an initial value of XORing, which means that no additional vulnerabilities are introduced for the initialization vector setting. Besides, as for the dependency between multiple paths that a byte cannot obtained only after the precedence bytes are received, it is intrinsic to MPTCP and is not a defect of the proposed method itself.

Another feature of the proposed method is that it does not introduce any additional control information at all. It just performs XORing a sending byte with bytes in the send scramble buffer. Even if data sending and data receiving are interleaved, the sending byte stream is focused and XORed beyond data receiving. The fact that data is delivered in sequence is assured by MPTCP, because the scrambling is done before data sending, and the descrambling is done after data receiving according to MPTCP.

IV. Performance Evaluation

In this section, we evaluate the processing overhead of the proposed method. In addition, we evaluate the overhead of commonly used cryptographic methods for the purpose of comparison. We adopt the data encryption standard (DES) [21], the triple data encryption algorithm (TDEA) [21], and the advanced encryption standard (AES) [22].

DES is a block based ciphering algorithm standardized by the National Institute of Standards and Technology (NIST). It is designed to encipher and decipher of blocks of data consisting of 64 bits (8 bytes) under control of a 64 bit (8 byte) key. Currently, it has been withdrawn as a standard ciphering method, but the TDEA, a compound operation of DES encryption and decryption operations, can be used as one of cipher suites in TLS.

AES is another block based ciphering algorithm newly standardized by NIST in 2001. It is a symmetric block cipher that can process data blocks of 128 bits (16 bytes), using cipher keys with lengths of 128, 192, and 256 bits (16 bytes, 24 bytes, and 32 bytes, respectively).

In this paper, we used publicly available source programs for DES and AES [23] distributed by PJC, a Japanese software company. They are written in C language. As for the DES algorithm, we prepared 160 blocks ($8 \times 160 = 1280$ bytes) and performed encryption and decryption for those blocks with the electronic codebook (ECB) mode. That is, each block is just encrypted and decrypted independently from other blocks. As for the TDEA algorithm, each of 160 blocks is encrypted or decrypted three times according to the DES algorithm with independent three keys. As for the AES algorithm, we prepared 80 blocks ($16 \times 80 = 1280$ bytes) and used keys with 128, 192 and 256 bit length (AES-128, AES-192 and AES-256). We also used the ECB mode here. It should be mentioned that we suppose 1280 byte long message to be transferred.

As for the proposed method, we introduced two kinds of implementations. One is a straightforward implementation, where the proposed method described in the previous section is programmed in C language as they are. The following are the summary of the straightforward implementation.

- The send/receive scramble buffers are realized by an array of unsigned char type.
- When a byte is scrambled or descrambled, the exclusive OR of all bytes in the scramble buffer is calculated.
- When a byte is scrambled or descrambled, it is added to the scramble buffer by shifting all bytes in the buffer.

The other is a revised implementation, where unnecessary data copying nor exclusive OR calculation are avoided. The following are the summary of the revised implementation.

- The send/receive scramble buffers are realized as ring buffers, by an array of unsigned char type (`sScrBuf[]` and `rScrBuf[]`). In order to avoid unnecessary data copying, the last element (newest element) in the ring buffer is maintained by an index parameter (`sIndex` or `rIndex`).
- The exclusive OR calculation for all bytes in the scramble buffer is performed just once in the beginning.

This result is maintained by a static variable `sXor` or `rXor`.

- When a byte is to be scrambled or descrambled, the static variable (`sXor` or `rXor`) is overwritten by the exclusive OR of the oldest element in the scramble buffer, `sXor` (or `rXor`) and the new byte.
- When a byte is to be scrambled or descrambled, it is added to the scramble buffer just by moving the index parameter (`sIndex` or `rIndex`).

By use of these two implementations, we executed the data scrambling and descrambling for a message with length of 1280 bytes.

We evaluated the performance of those seven methods (DES, TDEA, AES-128, AES-192, AES-256, the proposed method by straightforward implementation, and the proposed method by revised implementation). Table I shows the specification of personal computer used for the evaluation. It is a laptop computer manufactured by Lenovo over which the Linux operating system is installed. We measured the processing time of the encryption and decryption, or the scrambling and descrambling for a message with 1280 byte length. We used Linux `time` command for 10,000 iterations, and calculated the processing time for one operation.

Table II gives the performance results. The encryption and decryption of the DES and AES-128 algorithms require around 2.2 or 2.3 msec. The AES-192 and AES-256 algorithms requires a little more time. The TDEA algorithm requires around 6.7 msec, which is about three time of the DES algorithm. We need to say that we also evaluated the performance of the DES and AES with cipher block chaining (CBC) mode, and obtained the result that the processing time is almost the same with ECB mode.

On the other hand, the straightforward implementation of the proposed method requires around 1 msec. This is smaller than the cryptographic approaches, but the improvement is not large. However, the revised implementation of the proposed method decreases the processing time largely, to around 0.04 msec. It is less than 1/60 compared with the DES and AES algorithms. Although the implementation of DES and AES algorithms is a publicly accessible software, which may be optimized adequately, the obtained results are considered to show that the proposed method is able to decrease the processing overhead of ciphering operations and to provide some level of security against the eavesdropping over untrusted paths in MPTCP communications.

TABLE I. SPECIFICATION OF PC USED IN EVALUATION.

model	lenovo ThinkPad E430
CPU	Intel Core i5-3230M CPU × 4
clock	2.60GHz
memory size	3.7 Gbytes
kernel	ubuntu 16.04 LTS

TABLE II. PROCESSING TIME OF 1280 BYTE MESSAGE.

DES	TDEA	AES-128	AES-192	AES-256	Proposed (straight)	Proposed (revised)
2.24 msec	6.69 msec	2.29 msec	2.80 msec	3.40 msec	0.950 msec	0.0352 msec

V. STUDY ON IMPLEMENTATION

A. How to modify Linux operating system

Since MPTCP is implemented inside the Linux operating system, the proposed method also needs to be realized by modifying operating system kernel. However, modifying an operating system kernel is hard task, and so we decided to use a debugging mechanism for the Linux kernel, called kernel probes [24].

The following are cited from [24]. *A kernel probe is a set of handlers placed on a certain instruction address. There are two types of probes in the kernel as of now, called "KProbes" and "JProbes." A KProbe is defined by a pre-handler and a post-handler. When a KProbe is installed at a particular instruction and that instruction is executed, the pre-handler is executed just before the execution of the probed instruction. Similarly, the post-handler is executed just after the execution of the probed instruction. JProbes are used to get access to a kernel function's arguments at runtime. A JProbe is defined by a JProbe handler with the same prototype as that of the function whose arguments are to be accessed. When the probed function is executed the control is first transferred to the user-defined JProbe handler, followed by the transfer of execution to the original function.*

Figure 11 shows a schematic explanation of JProbe. We assume that there is function `a_func()` inside the Linux kernel, whose symbol is exported. A user may define JProbe handler `ja_func()` whose arguments are exactly the same as `a_func()`. When the Linux kernel is going to call `a_func()`, `ja_func()` is executed in the beginning of `a_func()`. When `ja_func()` returns, the kernel executes `a_func()`. In order to make this mechanism work, a user needs to prepare the following;

- registering the entry by `struct jprobe` and
- defining the init and exit modules by functions `register_jprobe()` and `unregister_jprobe()` [25].

A famous example of JProbe is tcpprobe [26] used to collect TCP sender internal information such as a TCP congestion window value.

B. Design principles

We adopted the following design principles to implement the proposed method inside the Linux kernel.

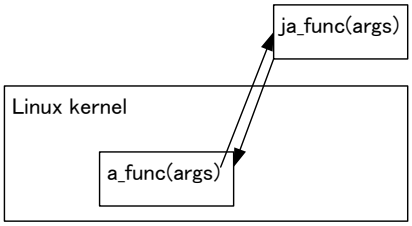


Figure 11. Schematic explanation of JProbe.

- *Use the JProbe mechanism as much as possible.*
  In the Linux kernel, function `tcp_sendmsg()` is called when a user process tries to send data to MPTPCP (actually TCP, too) [27]. So, we define a JProbe handler for this function in order to scramble data to be transferred. On the other hand, function `tcp_recvmsg()` is called when a user process is going to receive data from MPTCP. In this case, however, the descrambling procedure needs to be done in the end of this function. So, we introduce a dummy kernel function and export its symbol. We then introduce a JProbe handler for descrambling. By adopting this approach, we can program and debug scrambling/descrambling independently of the Linux kernel itself.
- *Maintain control variables within socket data structure.*
  In order to perform the scrambling/descrambling, the control variables described in the previous section, such as `sScrBuf[64]` and `sXor`, need to be installed within the Linux kernel. The TCP software in the kernel uses a socket data structure to maintain internal control data on an individual TCP / MPTCP connection [27]. So, we add the control variables for data scrambling to this data structure. Although the kernel modification and rebuild are required, we believe that to insert some variables is an easy task and therefore frequent debugging and rebuilding are not necessary.

C. Detailed design

(1) How to insert control variables in socket data structure

As described above, function `tcp_sendmsg()` is called at data sending. Its prototype in Ubuntu 16.04 LTS is;

```
tcp_sendmsg(struct sock *sk, struct
            msghdr *msg, size_t size).
```

Here, `struct sock` is a type of socket data structure. In the beginning of this function, `sk` is converted to type `struct tcp_sock` in the following way.

```
struct tcp_sock *tp = tcp_sk(sk);
```

`struct tcp_sock` is a type of TCP socket data structure maintaining TCP related members like `rcv_nxt`, which is sequence number of a byte to be expected to receive next, and `snd_nxt`, which is sequence number of a byte to be sent next. It also includes a control information on MPTCP like

```
struct mptcp_cb *mpcb;.
```

Structure `struct mptcp_cb` includes MPTCP related information including keys and tokens like `__u64 mptcp_loc_key;` (local key) and `__u32 mptcp_loc_token;` (local token). Based on these considerations, we decided to insert control variables for data scrambling within structure `struct mptcp_cb` in a way shown in Figure 12. When `tcp_sendmsg()` is called, we can access to these variables in a way like `tp->mpcb->sXor`. It should be noted that `sFirst` and `rFirst` indicate whether the scrambling and descrambling is performed at first or not in this MPTCP connection, respectively.

```
struct mptcp_cb {
 . . . .
  unsigned char sScrBuf[64], rScrBuf[64];
  unsigned char sXor, rXor;
  int sIndex, rIndex, sFirst=1, rFirst=1;
};
```

Figure 12. Control variables for data scrambling.

(2) How to implement scrambling

(2-1) Overview

Figure 13 shows an overview program structure to implement scrambling using JProbe handler `jtcp_sendmsg()`. As described in the previous subsection, `jtcp_sendmsg()` is declared so as to have the same arguments as `tcp_sendmsg()`, as shown in part (i) in the figure. Part (ii) in the figure shows a data structure registering an entry point of the JProbe hander and its related symbol name. Parts (iii) and (iv) are the initialization and exit functions, respectively.

We need to explain about the second argument of `jtcp_sendmsg()`. Structure `struct msghdr` has a linked data structure maintaining one or more members, each of which is expressed by structure `struct iovec`, including pointer to data (`iov_vase`) and its length (`iov_length`). `iov_for_each()` is a macro for traversing individual members, and can be used as a for statement in C language.

```
int jtcp_sendmsg(struct sock *sk,
        struct msghdr *msg, size_t size) {
  struct tcp_sock *tp = tcp_sk(sk);
  struct iov_iter iter;
  struct iovec iov;

  if(tp->mpcb->sFirst) scramble_init(tp);
  iov_for_each(iov, iter, msg->msg_iter) {
    scramble(tp, iov.iov_base, iov.iov_len);
  }
  jprobe_return();
  return 0;
} // (i) JProbe handler

static struct jprobe tcp_sendmsg_jprobe = {
  .kp = {.symbol_name = "tcp_sendmsg",},
        .entry = jtcp_sendmsg,
}; // (ii) Register entry

static __init int jtcp_sendmsg_init(void) {
  ret= register_jprobe(&tcp_sendmsg_jprobe);
  if (ret < 0) return -1;
  return 0;
}  // (iii) Init function
module_init(jtcp_sendmsg_init);

static __exit void jtcp_sendmsg_exit(void) {
  unregister_jprobe(&tcp_sendmsg_jprobe);
}  // (iv) Exit function
module_exit(jtcp_sendmsg_exit);
```

Figure 13. Overview on how to implement scrambling in `tcp_sendmsg()`.

In function jtcp_sendmsg(), control variable tp->mpcb->sFirst is checked in the beginning and, if it is 1, function scramble_init() is called. After that, function scramble() is called for each data contained in msg.

(2-2) Detailed design for scrambling

Figure 14 shows an example of program code for functions scramble_init() and scramble(). scramble_init() is called with an argument tp, which is a pointer to struct tcp_sock data structure. By functions memset() and memcpy(), the send scramble buffer tp->mpcb->sScrBuf[64] is initialized so as to contain the local Key in this MPTCP connection. Then, the XOR result for all bytes in the send scramble buffer is stored in control variable tp->mpcb->sXor. After that, the index parameter indicating the end of the send scramble buffer is settled and tp->mpcb->sFirst is reset.

Function scramble() performs the scrambling procedure for data pointed by argument data whose length is len. In this function, each byte in data is XORed with tp->mpcb->sXor, and index parameter tp->mpcb->sIndex is shifted by one. Then, tp->mpcb->sXor is updated by XORing itself, a byte that is stored in the newly indexed position, and a byte being scrambled (original byte). After that, the original byte is stored in a newly indexed position in the send scramble buffer. In the end, data is changed by the XORed byte for sending.

(3) How to implement descrambling

As mentioned in the previous subsection, the descramble procedure needs to be implemented at the end of function tcp_recvmsg(). In order to realize the descrambling procedure by the JProbe mechanism, we introduced dummy function dummy_recvmsg() just before returning from tcp_recvmsg(). This is shown in Figure 15. For this function, JProbe handler jdummy_recvmsg() is implemented in a similar way with the scrambling procedure.

```
void scramble_init(struct tcp_sock *tp) {
  int i;
  unsigned char x;

  memset(tp->mpcb->sScrBuf, 0, 64);
  memcpy(&tp->mpcb->sScrBuf[56],
         &tp->mpcb.mptcp_loc_key, 8);
  for(i=0,x=0;i<64;i++)
    x = x ^ tp->mpcb->sScrBuf[i];
  tp->mpcb->sXor = x;
  tp->mpcb->sIndex = 63;
  tp->mpcb->sFirst = 0;
  return;
}

void scramble(struct tcp_sock *tp,
              unsigned char *data, size_t len) {
  int i;
  unsigned char x;

  for(i=0;i<len;i++) {
    x = data[i] ^ tp->mpcb->sXor;
    tp->mpcb->sIndex = (tp->mpcb->sIndex+1)%64;
    tp->mpcb->sXor = tp->mpcb->sXor
         ^ tp->mpcb->sScrBuf[tp->mpcb->sIndex]
         ^ data[i];
    tp->mpcb->sScrBuf[tp->mpcb->sIndex] = data[i];
    data[i] = x;
  }
  return;
}
```

Figure 14. Program code for scrambling.

```
int tcp_recvmsg(struct sock *sk, struct msghdr *msg,
    size_t len, int nonblock,int flags, int *addr_len) {
  struct tcp_sock *tp = tcp_sk(sk);
  . . . .
  release_sock(sk);
  dummy_recvmsg(sk, msg, len, nonblack, flags, addr_len);
  return copied;
  . . . .
} // dummy_recvmsg() inserted
EXPORT_SYMBOL(tcp_recvmsg);

void dummy_recvmsg(struct sock *sk, struct msghdr *msg,
     size_t len, int nonblock, int flags, int *addr_len)
{
  return;
} // Defining dummy_recvmsg()
EXPORT_SYMBOL(dummy_recvmsg);
```

Figure 15. JProbe handler for data descrambling.

VI. CONCLUSIONS

This paper proposes a new method to improve privacy against eavesdropping over MPTCP communications, which has become popular among recent mobile terminals. Recent mobile terminals have multiple communication interfaces, some of which are connected to trusted network operators (e.g., LTE interfaces), and some of which may be connected to untrusted network, such as public WLAN hot spots. The proposed method here is based on the not-every-not-any protection principle, where, *if an attacker cannot observe the data on every path, he cannot observe the traffic on any path*. We designed a detailed procedure by following the byte oriented data scrambling in order to avoid unnecessary data length expansion.

We evaluated the processing overhead of the DES, TDEA and AES encryption/decryption and that of data scrambling in the proposed method. The result showed that the optimized implementation of our method requires only less than 1/60 processing time compared with the cryptographic approaches. Although the proposed method is a practical solution, as described above, the processing capability of mobile terminals is still low, and so our proposal is considered to be useful to increase the security against eavesdropping over untrusted mobile communication networks.

Moreover, we discussed how to implement the proposed method in the Linux operating system. We explained about a kernel debugging mechanism called JProbes, which is used to implement tcpprobe. We showed how to make program codes, especially focusing on how to realize the control parameters in the socket data structure and on how to realize the scrambling and descrambling procedures in the JProbe handlers.

We are currently implementing the proposed method on top of MPTCP software in the Linux operating system. We will continue this implementation and conduct the performance evaluation over real networks. Moreover, the proposed method can only prevent eavesdropping, and cannot ensure the integrity of transferred data. We need to improve our method in this aspect.

ACKNOWLEDGMENT

This research was performed under the research contract of "Research and Development on control schemes for utilizations of multiple mobile communication networks," for the Ministry of Internal Affairs and Communications, Japan.

REFERENCES

[1] T. Kato, S. Cheng, R. Yamamoto, S. Ohzahata, and N. Suzuki, "Protecting Eavesdropping over Multipath TCP Communication Based on Not-Every-Not-Any Protection," in Proc. SECURWARE 2017, pp. 82-87, Sep. 2017.

[2] NGNM Alliance, "5G White Paper," https://www.ngmn.org/5g-white-paper/5g-white-paper.html, [retrieved: May 2018].

[3] C. Paasch and O. Bonaventure, "Multipath TCP," Communications of the ACM, vol. 57, no. 4, pp. 51-57, Apr. 2014.

[4] AppleInsider Staff, "Apple found to be using advanced Multipath TCP networking in iOS 7," http://appleinsider.com/articles/13/09/20/apple-found-to-be-using-advanced-multipath-tcp-networking-in-ios-7, [retrieved: May 2018].

[5] icteam, "MultiPath TCP – Linux Kernel implementation, Users:: Android," https://multipath-tcp.org/pmwiki.php/Users/Android, [retrieved: May 2018].

[6] A. Ford, C.Raiciu, M. Handley, S. Barre, and J. Iyengar, "Architectural Guidelines for Multipath TCP Development," IETF RFC 6182, Mar. 2011.

[7] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," IETF RFC 6824, Jan. 2013.

[8] C. Raiciu, M. Handley, and D. Wischik, "Coupled Congestion Control for Multipath Transport Protocols," IETF RFC 6356, Oct. 2011.

[9] C. Pearce and S. Zeadally, "Ancillary Impacts of Multipath TCP on Current and Future Network Security," IEEE Internet Computing, vol. 19, iss. 5, pp. 58-65, Sept.-Oct. 2015.

[10] T. Kato, M. Tenjin, R. Yamamoto, S. Ohzahata, and H. Shinbo, "Microscopic Approach for Experimental Analysis of Multipath TCP Throughput under Insufficient Send/Receive Socket Buffers," in Proc. 15th ICWI 2016, pp. 191-199, Oct. 2016.

[11] J. Ma, F. Le, A. Russo, and J. Lobo, "Detecting Distributed Signature-based Intrusion: The Case of Multi-Path Routing Attacks," in Proc. 2015 INFOCOM, pp. 558-566, Apr. 2015.

[12] J. Yang and S. Papavassiliou, "Improving Network Security by Multipath Traffic Dispersion," in Proc. MILCOM 2001, pp. 34-38, Oct. 2001.

[13] M. Nacher, C. Calafate, J. Cano, and P. Manzoni, "Evaluation of the Impact of Multipath Data Dispersion for Anonymous TCP Connections," In Proc. SecureWare 2007, pp. 24-29, Oct. 2007.

[14] A. Gurtov and T. Polishchuk, "Secure Multipath Transport For Legacy Internet Applications," In Proc. BROADNETS 2009, pp. 1-8, Sep. 2009.

[15] L. Apiecionek, W. Makowski, M. Sobczak, and T. Vince, "Multi Path Transmission Control Protocols as a security solution," in Proc. 2015 IEEE 13th International Scientific Conference on Informatics, pp. 27-31, Nov. 2015.

[16] A. Shamir, "How to share a secret," Communications of the ACM, vol. 22, no. 11, pp. 612-613, Nov. 1979.

[17] X. Zhao, L. Li, G. Xue, and G. Silva, "Efficient Anonymous Message Submission," in Proc. INFOCOM 2012, pp. 2228-2236, Mar. 2012.

[18] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network Information Flow," IEEE Trans. Information Theory, vol. 46, no. 4, pp. 1204-1216, Jul. 2000.

[19] M. Li, A. Lukyanenko, and Y. Cui, "Network Coding Based Multipath TCP," in Proc. Global Internet Symposium 2012, pp. 25-30, Mar. 2012.

[20] ISO JTC 1/SC27, "ISO/IEC 10116: 2006 – Information technology – Security techniques – Modes of operation for an n-bit cipher," ISO Standards, 2006.

[21] Federal Information Processing Standards Publication 46-3, "Annoucing the Data Encryption Standard," Oct. 1999.

[22] Federal Information Processing Standards Publication 197, "Anouncing the Advanced Encryption Standard (AES)," Nov. 2001.

[23] PJC, "Distribution of Sample Program / Source / Software (in Japanese)," http://free.pjc.co.jp/index.html, [retrieved: May 2018].

[24] LWN.net, "An introduction to KProbes," https://lwn.net/Articles/132196/, [retreieved: May 2018].

[25] GitHubGist, "jprobes example: dzeban / jprobe_etn_io.c," https://gist.github.com/dzeban/a19c711d6b6b1d72e594, [retreieved: May 2018].

[26] Linux Foundation Wiki, "TCP Probe," https://wiki.linuxfoundation.org/networking/tcpprobe, [retrieved: May 2018].

[27] S. Seth and M. Venkatesulu, "TCP/IP Architecture, Desgn, and Implementation in Linux," John Wiley & Sons, 2009.

Empirical Analysis of Domain Blacklists

Tran Phuong Thao*, Akira Yamada†, Ayumu Kubota‡
KDDI Research, Inc., Japan
2-1-15 Ohara, Fujimino-shi, Saitama, Japan 356-8502
Email: {th-tran*, ai-yamada†, kubota‡}@kddi-research.jp

***Abstract*—Malicious content has grown along with the explosion of the Internet. Therefore, many organizations construct and maintain blacklists to help web users protect their computers. There are many kinds of blacklists in which domain blacklists are the most popular one. Existing empirical analyses on domain blacklists have several limitations such as using only outdated blacklists, omitting important blacklists, or focusing only on simple aspects of blacklists. In this paper, we analyze the top 14 blacklists downloaded on 2017/02/28 including popular and updated blacklists like *Safe Browsing* from Google and *urlblacklist.com*. We are the first to filter out the old entries in the blacklists using an enormous dataset of user browsing history. Besides the analysis on the intersections and the registered information from Whois (such as top-level domain, domain age and country), we also build two classification models for web content categories (i.e., education, business, etc.) and malicious categories (i.e., landing and distribution) using machine learning. Our work found some important results. First, the blacklists *Safe Browsing version 3 and 4* are being separately deployed and have independent databases with diverse entries although they belong to the same organization. Second, the blacklist *dsi.ut_capitole.fr* is almost a subset of the blacklist *urlblacklist.com* with 98% entries. Third, largest portion of entries in the blacklists are created in 2000 with 6.08%, and from United States with 24.28%. Fourth, *Safe Browsing version 4* can detect younger domains compared with the others. Fifth, *Tech & Computing* is the dominant web content category in all the blacklists, and the blacklists in each group (i.e., small public blacklists, large public blacklists, private blacklists) have higher correlation in web content as opposed to blacklists in other groups. Sixth, the number of landing domains are larger than that of distribution domains at least 75% in large public blacklists and at least 60% in other blacklists. In addition, we collected and analysed the updated version of 11 public blacklists that we downloaded on 2017/11/09, which is over 7 months after the previous blacklist version downloaded on 2017/02/28, and found some new results such as: the number of malicious domains injected by ransomwares is significantly increased (6.67x larger); or many Top Level Domains (TLDs) which belong to the type of *new generic TLD* such as *.forsale*, *.institute*, *.church*, etc., appear in the new blacklist version. We also discussed several challenges on measuring registration time of malicious domains in each blacklist, how to determine a malicious domain, malicious classification using Whois-document-based text mining, and standardization of Whois-attribute extraction.**

***Keywords*—Web Security, Empirical Analysis, Blacklist, Malicious Domain, Whois Information, HTML Document, Text Mining.**

I. INTRODUCTION

The Internet has become very important to our daily life, and thus, the content of the Web has been growing exponentially. According to a research by VeriSign, Inc. [2], the number of domains is already approximately 12 million as of March 31, 2016. Along with that is a huge amount of malicious domains. Just in 2015, the number of unique pieces of malware discovered is more than 430 million, up 36 percent from the year before [3]. Therefore, nowadays there are many competitive services constructed to detect malicious domains. Each service has its own method, which is often not disclosed and always said to be the best service by its authors. Furthermore, each service also has different definition (ground truth) of the term "malicious". For example, a blacklist A defines a domain D to be malicious if D satisfies a condition set AM while another blacklist B defines D to be malicious if D satisfies a condition set BM, which is a subset, superset or completely different from AM. All of these have brought into **a question: how to measure and compare these services**. Many blacklists are freely available on the Internet (called *public blacklists*). However, some vendors do not want to publish their databases and only provide querying services via APIs or portal applications (called *private blacklists*). Our goal in this paper is to perform a large-scale analysis on popular blacklists including both public and private blacklists. We can then indicate the quality of the blacklists in some specific categories. This research can help the users to determine which blacklists should they choose for some conditions, and also can help the blacklist providers assess and improve their blacklists and methods.

A. Related Work

Sheng et al. [4] analyzed phishing blacklists, which are just subset of malicious blacklists that we are focusing on. A malicious domain's purpose includes all kinds of attacks: spamming, phishing, randomware, etc. Kuhrer et al. [5] analyzed malicious blacklists but only focused on constructing a blacklist parser to deal with varied-and-unstructured blacklist formats rather than researching the blacklists themselves. This is because some blacklists solely include domain names, URLs, or IP address. Other blacklists contain more information, such as timestamps or even source, type, and description for each entry. Therefore, their analysis results have poor information that only contains the entries' registration history in each blacklist, the intersection of every blacklist pair, and the top 10 domains in most of the blacklists. Kuhrer et al. [6] then analyzed blacklists via three measures: (i) identifying parked domains (additional domains hosted on the same account and displaying the same website as primary domain) and sinkhole servers (hosting malicious domains controlled by security organizations), (ii) the blacklist completeness by finding the coverage between each blacklist with an existing set of 300,000 malware samples, and (iii) the domains created by Domain Generation Algorithm. However, 300,000 entries in the second measure are not enough to assess the "completeness" because some large blacklists can contain millions of entries.

Furthermore, the ground truth or definition of their malware samples may be different from that of other blacklists, and thus it is unfair when using them to confirm the completeness of other blacklists. The first and third measures are different for our analysis. Vasek et al. [7] only analyzed Malware Domain Blacklist (malwaredomains.com), which is just one of the blacklists in our analysis. Several other papers also performed empirical analysis but are different from our analysis, which focuses on domain blacklists, e.g., [8] analyzed IP blacklists, [9] analyzed email spam detection through network characteristics in a stand-alone enterprise, [10] analyzed spam traffic with a very specific network, [11] analyzed detections of malicious web pages caused by drive-by-download attack, not blacklist analysis, [12] analyzed whitelist of acceptable advertisements.

B. Our Work

In this paper, we do not aim to figure out the ground truth or definition of "malicious", or the factors affecting malicious domain detection in each blacklist. Instead, we attempt to quantitatively measure and compare the blacklists based on seven important aspects: blacklist intersections, top-level domains (TLDs), domain ages, countries, web content categories, malicious categories, differences between the blacklist versions. To the best of our knowledge, we are the first to achieve the followings:

- We deal with top 14 popular blacklists in which there are two special private blacklists given by Google that are Safe Browsing version 3 and 4 (called GSBv3 and GSBv4). These newest versions are being deployed and used parallelly and independently, and have never been analyzed before. In [5], the old version GSBv2 was analyzed in 2011, which was 6 years ago.
- By designing 7 measures in our analsysis, we not only consider the coverage (intersection) as in previous works, but also compare the blacklists based on Whois (TLDs, countries, domain ages), web content categories using IAB [13], which are an industry standard taxonomy for content categorization (e.g., education, government, etc.), malicious categories (landing and distribution), and the differences between the current and the newest updated blacklist versions (as of the time of writing this paper).
- Our analysis is not straightforward, and not just simple statistics. For the measures of web content categories and malicious categories, we construct two supervised machine learning models using text mining, and a combination of text mining with some specific HTML tags to classify the entries in the blacklists, respectively.
- Last but not least, we filter out the active entries in the blacklists instead of old and useless entries as previous works by finding the coverage between each blacklist with a big live dataset.

Roadmap. The rest of this paper is organized as follows. The methodology of our analysis is presented in Section II. The empirical results are given in Section III. The discussion is described in Section IV. Finally, the conclusion is drawn in Section V.

II. Methodology

In this section, we introduce our chosen blacklists, how we pre-processed them, and our analysis design.

A. Blacklists

In this paper, we analyze 14 popular blacklists as described in Table I. Since they have different numbers of entries, which can effect the fairness, we categorize them into 3 groups: (I) *small public blacklists* which have smaller than 1,000,000 unique entries, (II) *large public blacklists* which have equal or larger than 1,000,000 unique entries, and (III) *private blacklists*. In the group (III), we consider separately GSBv3 and GSBv4 although they both belong to the same vendor. This is because they are being deployed and used independently. Furthermore, according to our analysis, they have different API and even database.

TABLE I: 14 POPULAR BLACKLISTS.

No	Group	Abbr.	Blacklists	#Domains
1	(I)	MA	malwaredomains.com	17,294
2		NE	networksec.org	263
3		PH	phishtank.com	9,711
4		RA	ransomwaretracker.abuse.ch	1,380
5		ZE	zeustracker.abuse.ch	382
6		MAL	malwaredomainlist.com	1,338
7		MV	winhelp2002.mvps.org	218,248
8		HO	hosts-file.net	5,974
9	(II)	ME	mesd.k12.or.us	1,266,334
10		SH	shallalist.de	1,570,944
11		UR	urlblacklist.com	2,919,199
12		UT	dsi.ut_capitole.fr	1,346,788
13	(III)	GSBv3	Safe Browsing version 3	Unknown
14		GSBv4	Safe Browsing version 4	Unknown

In Table I, the last column indicates the number of unique domains in each blacklist. All the 14 blacklists were downloaded (in case of public blacklists) or queried (in case of private blacklists) on the same date 2017/02/28. Since the blacklists may contain old entries that attackers no longer use, we extract only active entries by finding the intersection between each blacklist with a real-world web access log that we call AL. AL has 3,991,599,424 records from 5 proxy servers, 9,091,980 raw domains with 80,464,378 corresponding URLs accessed by 659,283 users. The intersections between AL and each blacklist are given in Table II. The number of unique domains in the union of 14 blacklists is 50,519. Instead of the complete blacklists, we use these intersections in our analysis. We should mention that the AL focuses on the users in Japan; and thus, the information of the access records is mainly from Japan (e.g., many domains has the top-level domains (TLD) of *.jp*). Using the AL is not generic; and analyses for other countries are recommended to use the different specific access log. Although there exists a bias when using the AL here, the insights found in our analyses can be widely effective to other countries; for instance, most of the domains are registered from United States and created in 2000; or, GSBv4 can detected younger domains than other blacklists, etc.

B. Analysis Design

In this section, we describe the design of our analysis with the following 6 measures.

TABLE II: ACTIVE MALICIOUS DOMAINS IN 14 BLACKLISTS (INTERSECTIONS WITH AL).

No	Group	Intersection	Abbr.	#Domains	Percentage
1	(I)	AL ∩ MA	AMA	77	0.44%
2		AL ∩ NE	ANE	2	0.76%
3		AL ∩ PH	APH	367	3.78%
4		AL ∩ RA	ARA	3	0.22%
5		AL ∩ ZE	AZE	21	5.50%
6		AL ∩ MAL	AMAL	98	7.32%
7		AL ∩ MV	AMV	2,176	1.00%
8		AL ∩ HO	AHO	5,060	84.70%
9	(II)	AL ∩ ME	AME	19,812	1.56%
10		AL ∩ SH	ASH	32,248	2.05%
11		AL ∩ UR	AUR	33,674	1.15%
12		AL ∩ UT	AUT	24,020	1.78%
13	(III)	AL ∩ GSBv3	AGSBv3	189	unknown
14		AL ∩ GSBv4	AGSBv4	639	unknown

The final column indicates the number of filtered samples over that of original samples in Table I.

1) Measure 1 (Blacklist Intersections): For every blacklist pair with the web access log AL, we find the intersection of their domains. In total we found $\binom{14}{2} = 91$ intersection sets. In our previous article [1], we determined the blacklist pair that has the highest correlation in term of overlapping entries based on the number of entries in the intersections (i.e., the blacklist pair that has largest number of entries in their intersection is the one has highest correlation). However, it is unfair to compare between all the blacklist pairs because each blacklist has a different number of entries. Therefore, in this article version, we determine the blacklist pair that has the highest correlation based on the average of the two percentages of the pairs and then choose the largest one. We will explain the example in Section III-A.

2) Measure 2 (Top-Level Domains (TLDs)): A TLD is the domain in the highest level of the hierarchical Domain Name System. For example, the TLD of the domain *kddi.com* is *com*, the TLD of the domain *yahoo.co.jp* is *jp*. To evaluate this measure, we extract the final string after the dot in each domain name. According to ICANN (the Internet Corporation for Assigned Names and Numbers) [14], there are 1,540 different TLDs as of 2017/11/15 categorized into 6 types: (i) infrastructure top-level domain (ARPA), (ii) generic top-level domains (gTLD), (iii) restricted generic top-level domains (grTLD), (iv) sponsored top-level domains (sTLD), (v) test top-level domains (tTLD), and (vi) new generic top-level domains (new gTLD). The *most common type is the generic top-level domains (gTLD)*, which has two sub-types:

- Original TLDs: consist of *.com*, *.org*, *.net*, *.int*, *.edu*, *.gov* and *.mil*.
- Country-code TLDs: consist of the TLDs of each country or region. For example, *.jp* (Japan), *.us* (United States), *.eu* (European Union), etc.

3) Measure 3 (Domain Ages) and Measure 4 (Countries): To evaluate these measures, we firstly extract the Whois information of each domain in all the intersections between the blacklists and the web access log AL as described in Table II. Whois is the registered information of the domains such as creation date, expiration date, organization, address, registrar server, etc. For the measure 3, we extract creation year (from the creation date) and for the measure 4, we extract the country. Note that, although the measure 2 (TLD) includes country-code TLDs, it does not always show correct countries. For example, the TLD of *jp* not only contains domains from Japan, but also another countries such as United States with a non-small portion. This is why we consider the measure 2 (TLD) and measure 4 (country), separately.

4) Measure 5 (Web Content Categories): This measure aims to classify the blacklisted domains into semantic web content categories, such as education, advertisement, government, etc. Although there are several tools (e.g., i-Filter [15], SimilarWeb [16]) which can be used to categorize a domain into semantic content categories, their coverages are low and they cannot label our entire dataset (this will be explained later). Therefore, to evaluate this measure, we construct our own classification model using supervised machine learning with the help of one of the tools for data labelling. Concretely, we first collect 20,000 URLs and label their semantic contents using i-Filter [15]. However, i-Filter cannot label all the samples but only 14,492 samples (72.46%) into 69 categories. Since the number of categories is quite large for the number of classes in our model, we thus generalize these 69 categories into 17 categories using the standardized category set called IAB [13]. We then extract HTML documents of the 14,492 samples and use *text mining* with Term Frequency-Inverse Document Frequency (TF-IDF) as the feature for the training process. We executed nine different supervised machine learning algorithms: Support Vector Machine (including C-based and Linear-based), Naive Bayes (including Multinomial-based and Bernoulli-based), Nearest Neighbors (including Centroid-based, KNeighbors-based and Radius-based), Decision Tree, and Stochastic Gradient Descent. We assessed the algorithms using k-fold cross validation by setting $k = 10$. We pick up the best algorithm, which has highest accuracy and lowest false positive rate. Thereafter, we extract HTML documents of 50,519 domains in our blacklists. Note that, given a domain, we extract the main URL of the domains by adding prefix *http://www* to the domain. For example: the main url of *google.com* is *http://www.google.com*. We use the model computed by the chosen best learning algorithm to classify the 50,519 domains in the blacklists.

5) Measure 6 (Malicious Categories): There are two types of malicious categories. The first type is about the behaviours of attackers such as phishing, spamming or abusing, etc. This type has already been considered in many previous works. The second type is about the behaviours of the domains/URLs themselves such as *landing* and *distribution*, which are very important properties to understand the attacks but have not been widely considered before. Landing domains are what the web users are often attracted to access, and contain some malicious codes (usually Javascript) which can redirect the users (victims) to another malicious domains called distribution domains. Distribution domains are what the victims are redirected to unconsciously, and really install malwares into the victims' computers. To the best of our knowledge, currently there is a unique tool which can be used to classify a malicious domain into landing or distribution, which is GSBv4. GSBv4 not only is a blacklist (i.e., can detect whether a domain is malicious or benign) but also can classify a malicious domain into landing or distribution category. However, its classification rate is too low (this will be explained later); furthermore, it can

only classify the domains belonging to its blacklist without being able to classify domains in other blacklists. This is why we construct our own classification model using supervised machine learning and only use GSBv4 for data labelling. Concretely, we first randomly collect 31,507 malicious URLs and label them using GSBv4. We then only have 5,772 samples (18.31%), which can be labelled by GSBv4 (4,124 landings and 1,648 distributions). After that, we extracted HTML documents of the labelled 5,772 samples to use in the training process. For feature selection, at first, adapting the idea of [17], we extracted and counted the following special HTML elements in each type:

- Type 1: eight HTML tags, which are used very often in landing domains including: <*script*>, <*iframe*>, <*form*>, <*frame*>, <*object*>, <*embed*>, <*href*>, and <*link*>. This is because these tags allow to place URLs inside, and thus have potential for the redirection, which is a specific characteristic of landing domains.
- Type 2: three elements which are commonly used in distribution domains including *swf*, *jar* and *pdf*. This is because these elements are mostly potential exploitable contents that distribution domains install into victim's computers.

However, our implementation showed that the accuracy of this method is very low (less than 71% using the 9 learning algorithms and 10-fold cross validation). Therefore, we then combine the 2 methods: the above HTML elements (in which the count of all tags in each type is used as one feature) along with text mining on entire HTML documents (in which the TF-IDF of each unique word is used as one feature). As a result, fortunately, we can get 98.07% in accuracy with merely 2.22% in false positive rate. Finally, we use the model of our combining method to classify 50,519 entries in the blacklists.

6) Measure 7 (New Blacklist Version): This measure aims to the new findings on the differences between the new and old versions of the same blacklists. A simple design is based on the number of unique domains in the previous and new blacklist versions. Note that, in this measure, we will directly analyse the blacklists themselves without getting the intersection between the blacklists with the AL. Since private blacklists (group III) such as GSBv3 and GSBv4 do not disclose their number of entries and also their entries in plaintext format (just in a hashed format), we cannot directly analyse them. For this reason, this measure only focuses on public blacklists including small (group I) and large (group II) public blacklists. Besides the differences between the number of domains in the previous and new blacklist versions, we also found some important findings when implementing the TLDs of the new blacklist version.

III. Empirical Results

In our implementation, we use two machines: a computer Intel(R) core i7, RAM 16.0 GB, 64-bit Windows 10; and a MacBook Pro Intel Core i5 processor, 2.7 GHz, 16 GB of RAM, OS X EI Capitan version 10.11.6. Since we do not consider the execution time, it does not matter that the two machines have different configurations. They are just used to speed up our evaluation modules, which can be executed parallelly and independently. We execute the 6 measures using Python 2.7.11 programming language with *pandas* library to deal with big data. Furthermore, we use *python-whois* library version 0.6.5 for Whois extraction of measure 3 and 4. We also use *scikit-learn* library for text mining and *BeautifulSoup* library for HTML extraction of measure 5 and 6.

TABLE III: TOP 10 DOMINANT TLDs IN ALL THE BLACKLISTS.

No	TLD	#Domains	Percentage
1	com	32,691	64.71 %
2	jp	4,277	8.47 %
3	net	3,458	6.84 %
4	org	1,856	3.67 %
5	de	726	1.44 %
6	uk	683	1.35 %
7	au	428	0.85 %
8	edu	375	0.74 %
9	tv	366	0.72 %
10	info	310	0.61 %

TABLE IV: TOP 5 DOMINANT TLDs IN EACH BLACKLIST

No	Blacklist	#Distinct TLDs	1st	2nd	3rd	4th	5th
1	AMA	25	com	jp	pl	net	org
2	ANE	2	com	pl			
3	APH	68	com	net	org	ru	pl
4	ARA	3	to	org	cab		
5	AZE	9	net	com	ua	ru	jp
6	AMAL	22	com	net	it	jp	ru
7	AMV	79	com	net	de	ru	org
8	AHO	145	com	net	org	jp	de
9	AME	113	com	net	org	tv	jp
10	ASH	197	com	jp	net	org	de
11	AUR	180	com	net	org	jp	uk
12	AUT	137	com	net	org	jp	tv
13	AGSBv3	34	com	org	jp	net	cn
14	AGSBv4	61	com	net	top	org	biz

A. Measure 1: Blacklist Intersections

In Table V, we computed the intersections of 91 blacklist pairs (with AL); and not only the number of entries in the intersections (overlapping entries), we also computed the corresponding percentages for each of the blacklist pairs. From the results in Table V, we observe some important information as follows:

- Based on the method used to score the correlation as described in Section II-B1, the table show that the blacklist pair that has the highest correlation in term of overlapping entries is (ME, UT) because the intersection AME ∩ AUT contains 19,598 entries, which occupies 90.25% in average of the two percentages (98.92% AME and 81.59% AUT). This result is different from the previous result in the paper [1], which showed that the pair that has highest correlation is (UT, UR) due to the highest number of entries in the intersection (23,583 domains). Note that, a blacklist pair in which one blacklist is a/an (almost) subset of the other is not always the pair, which has highest correlation in term of overlapping entries. This is because the percentage of the subset blacklist is very

TABLE V: OVERLAPPING NUMBER OF ENTRIES (INTERSECTIONS) OF EVERY BLACKLIST PAIR

	ANE	APH	ARA	AZE	AMAL	AMV	AHO	AME	ASH	AUR	AUT	AGSBv3	AGSBv4
MA	2 AMA: 2.60% ANE: 100%	7 AMA: 9.09% APH: 1.91%	0 AMA: 0% ARA: 0%	0 AMA: 0% AZE: 0%	0 AMA: 0% AMAL: 0%	0 AMA: 0% AMV: 0%	35 AMA: 45.45% AHO: 0.69%	77 AMA: 100% AME: 0.39%	1 AMA: 1.30% AHS: 0%	13 AMA: 16.88% AUR: 0.04%	1 AMA: 1.30% AUT: 0%	1 AMA: 1.30% AGSBv3: 0.53%	4 AMA: 5.19% AGSBv4: 0.63%
NE		0 ANE: 0% APH: 0%	0 ANE: 0% ARA: 0%	0 ANE: 0% AZE: 0%	0 ANE: 0% AMAL: 0%	0 ANE: 0% AMV: 0%	1 ANE: 50.00% AHO: 0.02%	2 ANE: 100% AME: 0.01%	0 ANE: 0% AHS: 0%	2 ANE: 100% AUR: 0.01%	0 ANE: 0% AUT: 0%	0 ANE: 0% AGSBv3: 0%	2 ANE: 100% AGSBv4: 0.31%
PH			0 APH: 0% ARA: 0%	6 APH: 1.63% AZE: 28.57%	14 APH: 3.81% AMAL: 14.29%	42 APH: 11.44% AMV: 1.93%	175 APH: 47.68% AHO: 3.46%	15 APH: 4.09% AME: 0.08%	104 APH: 28.34% AHS: 0.32%	100 APH: 27.25% AUR: 0.30%	51 APH: 13.90% AUT: 0.21%	1 APH: 0.27% AGSBv3: 0.53%	1 APH: 0.27% AGSBv4: 0.16%
RA				0 ARA: 0% AZE: 0%	0 ARA: 0% AMAL: 0%	0 ARA: 0% AMV: 0%	3 ARA: 100% AHO: 0.06%	0 ARA: 0% AME: 0%	2 ARA: 66.67% ASH: 0.01%	1 ARA: 33.33% AUR: 0%	0 ARA: 0% AUT: 0%	0 ARA: 0% AGSBv3: 0%	0 ARA: 0% AGSBv4: 0%
ZE				2 AZE: 9.52% AMAL: 2.04%	2 AZE: 9.52% AMAL: 2.04%	1 AZE: 4.76% AMV: 0.05%	18 AZE: 85.71% AHO: 0.36%	2 AZE: 9.52% AME: 0.01%	6 AZE: 28.57% ASH: 0.02%	6 AZE: 28.57% AUR: 0.02%	1 AZE: 4.76% AUT: 0%	0 AZE: 0% AGSBv3: 0%	0 AZE: 0% AGSBv4: 0%
MAL						21 AMAL: 21.43% AMV: 0.97%	67 AMAL: 68.37% AHO: 1.32%	6 AMAL: 6.12% AME: 0.03%	30 AMAL: 30.61% ASH: 0.09%	36 AMAL: 36.73% AUR: 0.11%	6 AMAL: 6.12% AUT: 0.02%	0 AMAL: 0% AGSBv3: 0%	0 AMAL: 0% AGSBv4: 0%
MV							1,241 AMV: 57.03% AHO: 24.53%	262 AMV: 12.04% AME: 1.32%	1,152 AMV: 52.94% ASH: 3.57%	948 AMV: 43.57% AUR: 2.82%	626 AMV: 28.77% AUT: 2.61%	0 AMV: 0% AGSBv3: 0%	0 AMV: 0% AGSBv4: 0%
HO								754 AHO: 14.90% AME: 3.81%	2,070 AHO: 40.91% ASH: 6.42%	1,733 AHO: 34.25% AUR: 5.15%	1,179 AHO: 23.30% AUT: 4.91%	3 AHO: 0.06% AGSBv3: 1.59%	5 AHO: 0.10% AGSBv4: 0.78%
ME									11,736 AME: 59.24% ASH: 36.39%	19,494 AME: 98.39% AUR: 57.89%	19,598 AME: 98.92% AUT: 81.59%	7 AME: 0.04% AGSBv3: 3.70%	28 AME: 0.14% AGSBv4: 4.38%
SH										19,495 ASH: 60.45% AUR: 57.89%	14,769 ASH: 45.80% AUT: 61.49%	4 ASH: 0.01% AGSBv3: 2.12%	19 ASH: 0.06% AGSBv4: 2.97%
UR											23,583 AUR: 70.03% AUT: 98.18%	7 AUR: 0.02% AGSBv3: 3.70%	29 AUR: 0.09% AGSBv4: 4.54%
UT												7 AUT: 0.03% AGSBv3: 3.70%	25 AUT: 0.10% AGSBv4: 3.91%
GSBv3													170 AGSBv3: 89.95 AGSBv4: 26.60

high but that of the other blacklist can be very small; and that makes the average percentage is not higher than that of other blacklist pairs.

- The results also indicate that the size of the values in this table is *not only dependent on the size of each original blacklist*. For example, ASH = 32,248 and AUR = 33,674 but ASH ∩ AUR = 19,495, which is smaller than AUT ∩ AUR = 23,583 even though AUT = 24,020, which is smaller than ASH.
- Most (not all) of the entries in AGSBv3 is listed in AGSBv4 since the percentage in GSBv3-side is very high (89.95%). Note that, GSBv3 and GSBv4 are the different versions of the same Google's Safe Browsing product but are being deployed parallelly and are using different databases. This result can lead to a hypothesis that, GSBv3 probably will be gradually merged with GSBv4 in near future although there has been no official announcement yet.

B. Measure 2: TLDs

From 50,519 unique domains in all the blacklists, we found 253 different TLDs in totals in which the top 10 dominant TLDs *for all the blacklists* are given in Table III. We then found top 5 dominant TLDs *for each blacklist* as given in Table IV. The third column is the number of distinct TLDs in each blacklist. The fourth until the eighth columns are the top 5 TLDs in descending order. Similar to the measure 1, the number of unique TLDs (the 3rd column) is *not always dependant on the number of entries* in each blacklist. For example, the blacklist HO belongs to the group I (small public blacklists) and AHO has only 5,060 entries but the number of TLDs is 145; meanwhile, the ME belongs to the group II (large public blacklists) and AME has 19,812 entries, which is almost $4\times$ larger than that of AHO, but its number of TLDs is only 113.

C. Measure 3: Domain Ages

Considering the union of all 14 blacklists, there are *34 distinct creation years* (from 1984 to 2017) as given in Figure 1. We can observe that the number of detected malicious domains created *after 1993 increases remarkably* compared to the years before 1993, and *drops down from 2016* (just 1 year before the date that we started our analysis). This indicates that most of the blacklists can *detect the new (young) malicious domains created after 2015 with very low rate*. The top 10 dominant years with corresponding number of domains are given in Table VI. For each blacklist, we also found the top 5 dominant creation years as presented in Table VII. We can observe that the blacklists *MA and GSBv4 can detect younger domains* compared with the other blacklists. Meanwhile, the blacklists *MAL and MV can detect very old domains*.

D. Measure 4: Countries

From the union of 14 blacklists, which contains 50,519 domains, we found 173 distinct registered countries. Note that, some domains are registered under one or multiple countries. That is, the registrator's addresses consist of one or multiple countries. For this reason, we consider each different country even in the same domain instead of just randomly choosing one

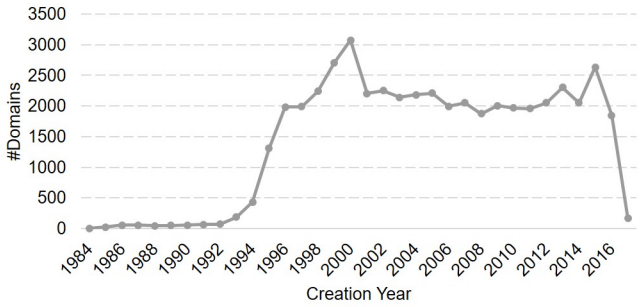


Figure 1: Distribution of Domain Ages (Creation Year).

TABLE VI: TOP 10 DOMINANT CREATION YEARS IN ALL THE BLACKLISTS.

No	Year	#Domains	Percentage
1	2000	3,073	6.08 %
2	1999	2,707	5.36 %
3	2015	2,633	5.21 %
4	2013	2,302	4.56 %
5	2002	2,249	4.45 %
6	1998	2,239	4.43 %
7	2005	2,209	4.37 %
8	2001	2,205	4.36 %
9	2004	2,181	4.32 %
10	2003	2,141	4.24 %

TABLE VII: TOP 5 DOMINANT CREATION YEARS IN EACH BLACKLIST.

No	Blacklist	#Distinct Years	1st	2nd	3rd	4th	5th
1	AMA	16	2016	2015	2014	2013	2012
2	ANE	2	2012	2006			
3	APH	27	2011	2009	2010	1999	2004
4	ARA	3	2014	2013	2008		
5	AZE	12	2007	2004	2001	2008	2006
6	AMAL	25	1999	1997	1998	1996	2005
7	AMV	32	1998	1999	1995	1996	2000
8	AHO	32	2005	2007	2016	1999	2012
9	AME	29	2015	2013	2012	2014	2011
10	ASH	33	2000	1999	2002	2001	1998
11	AUR	33	2015	2013	1999	2000	2007
12	AUT	33	2015	2013	2012	2014	2007
13	AGSBv3	21	2016	2012	2009	2013	2011
14	AGSBv4	21	2016	2015	2014	2012	2013

of the countries for each domain when the domain has multiple countries. The top 10 dominant countries throughout the union of 14 blacklists are given in Table VIII. Besides the union of all the blacklists, we also found top 5 dominant countries in each blacklist as presented in Table IX. The third column is the number of distinct countries in each blacklist. The fourth until eighth columns are the top 5 dominant countries described in descending order. From this table, we can observe that ME and UT have highest correlation because their numbers of distinct countries are almost equal, and the order of their dominant countries from the fourth to the eighth column is exactly same.

E. Measure 5: Web Content Categories

1) Pre-processing and Determining the Best Algorithm for Classification: After labelling 14,492 samples by i-Filter and

TABLE VIII: TOP 10 DOMINANT COUNTRIES IN ALL THE BLACKLISTS.

No	Country	#Domains	Percentage
1	US	12,267	24.28 %
2	JP	7,959	15.75 %
3	CY	3,988	7.89 %
4	PA	3,207	6.35 %
5	RU	1,194	2.36 %
6	AU	1,172	2.32 %
7	FR	1,072	2.12 %
8	DE	1,072	2.12 %
9	CA	994	1.97 %
10	GB	983	1.95 %

TABLE IX: TOP 5 DOMINANT COUNTRIES IN EACH BLACKLIST.

No	Blacklist	#Distinct Countries	1st	2nd	3rd	4th	5th
1	AMA	28	JP	US	CN	CA	FR
2	ANE	2	PL	CN			
3	APH	54	US	RU	AU	DE	BR
4	ARA	3	TO	DE	CA		
5	AZE	11	US	UA	RU	JP	NU
6	AMAL	28	US	IT	RU	JP	KR
7	AMV	81	US	DE	CA	FR	PA
8	AHO	104	US	JP	PA	CN	DE
9	AME	125	US	CY	PA	JP	RU
10	ASH	153	US	JP	CY	PA	DE
11	AUR	152	US	CY	JP	PA	RU
12	AUT	126	US	CY	PA	JP	RU
13	AGSBv3	39	US	JP	CN	RU	PL
14	AGSBv4	58	US	CN	JP	PL	DJ

IAB as mentioned in Section II-B4, we got 17 categories as described in Table X. Note that, the order of the numbers of samples in these categories does not indicate that of the domains in the blacklists. Even the numbers of samples in the categories are varied, for example, the number of samples of *Tech & Comp.* is double that of *Business* in the training dataset, it does not mean that *Tech & Comp.* always has higher order than *Business* in the applied dataset. We used the 14,492 labelled samples for our training dataset and inputted them to the supervised algorithms. We obtained the accuracy and false positive rate for each algorithm as given in Figure 2. We found that Decision Tree gives the best accuracy (99.58%) and lowest false positive rate (0.04%). We thus choose it to classify the domains in our blacklists.

2) Classification Result Using the Best Algorithm: As explained above, we use Decision Tree for the classification of the web content. For the union of all the blacklists, which consists of 50,519 domains, the web content categories with the corresponding number of domains are given in Table XI. We observe that the top 3 dominant categories are *Technology and Computing*, *Business*, and *Non-Standard content* (such as *Pornography*, *Violence*, or *Incentivized*). For each blacklist, the top 5 dominant categories with corresponding number of domains are presented in Table XII. We found that all the blacklists belonging to the group II (large public blacklists including ME, SH, UR, and UT), have higher correlation in web content categories rather than the other blacklists since the number of distinct categories and the order of dominant categories are exactly the same. Furthermore, MV and HO,

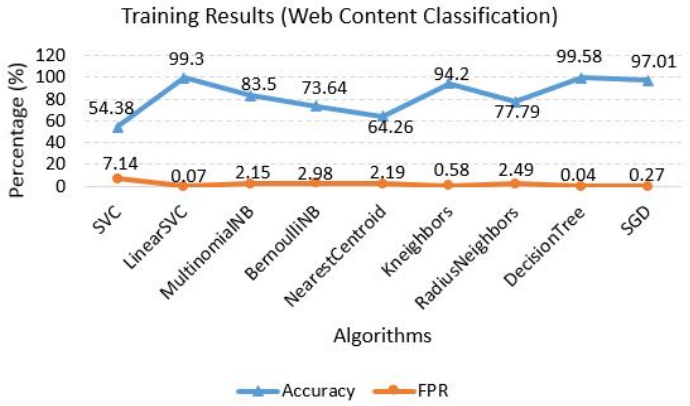


Figure 2: Accuracy and False Positive Rate of Each Algorithm

which belong to the group I (small public blacklists) and GSBv3, which belongs to the group III (private blacklists) also have the same order of dominant categories.

TABLE X: 17 CATEGORIES OF THE WEB CONTENT IN THE TRAINING DATASET.

No	Category	#Samples	No	Category	#Samples
1	Art & Entert.	65	10	Personal Finance	103
2	Automotive	29	11	Real Estate	18
3	Business	4,622	12	Tech & Comp.	7,632
4	Careers	17	13	Society	137
5	Education	15	14	Hobby & Interest	503
6	Shopping	604	15	Non-Standard	490
7	Food & Drink	37	16	News	117
8	Science	8	17	Sports	8
9	Travel	87			

TABLE XI: WEB CONTENT CATEGORIES IN ALL THE BLACKLISTS.

Due to space limitation, we use first three characters in each category as the abbreviation in the 3rd column.

No	Category	Abbr.	#Domain	Percentage
1	Tech & Computing	Tec	13,987	27.69 %
2	Business	Bus	10,259	20.31 %
3	Non-Standard	Non	10,032	19.86 %
4	Shopping	Sho	6,179	12.23 %
5	Hobby and Interest	Hob	2,678	5.30 %
6	Travel	Tra	1,708	3.38 %
7	Education	Edu	994	1.97 %
8	Arts & Entertainment	Art	933	1.85 %
9	Food & Drink	Foo	816	1.62 %
10	Careers	Car	674	1.33 %
11	News	New	628	1.24 %
12	Personal Finance	Per	570	1.13 %
13	Automotive	Aut	446	0.88 %
14	Sports	Spo	231	0.46 %
15	Science	Sci	230	0.46 %
16	Society	Soc	78	0.15 %
17	Real Estate	Rea	76	0.15 %

F. Measure 6: Malicious Categories

1) Pre-processing and Determining the Best Algorithm: Unlike the measure 5, which has 17 labels, this measure only has 2 labels: landing (4,124 samples) and distribution (1,648

TABLE XII: TOP 5 WEB CONTENT CATEGORIES IN EACH BLACKLIST.

No	Blacklist	#Distinct Categories	1st	2nd	3rd	4th	5th
1	AMA	11	Bus	Tec	Non	Sho	Art
2	ANE	1	Bus				
3	APH	16	Tec	Bus	Non	Sho	Hob
4	ARA	3	Sho	Bus	Tec		
5	AZE	5	Tec	Bus	Sho	Hob	Art
6	AMAL	12	Bus	Tec	Non	Sho	Tra
7	AMV	17	Bus	Tec	Non	Sho	Hob
8	AHO	17	Bus	Tec	Non	Sho	Hob
9	AME	17	Tec	Non	Bus	Sho	Hob
10	ASH	17	Tec	Non	Bus	Sho	Hob
11	AUR	17	Tec	Non	Bus	Sho	Hob
12	AUT	17	Tec	Non	Bus	Sho	Hob
13	AGSBv3	14	Bus	Tec	Non	Sho	Hob
14	AGSBv4	15	Bus	Tec	Non	Hob	Sho

samples). We train the dataset using the 9 algorithms and got the results as depicted in Figure 3. Decision Tree gives the best result with 98.07% accuracy and merely 2.22% false positive rate. Therefore, Decision Tree is chosen to classify the entries in the blacklists.

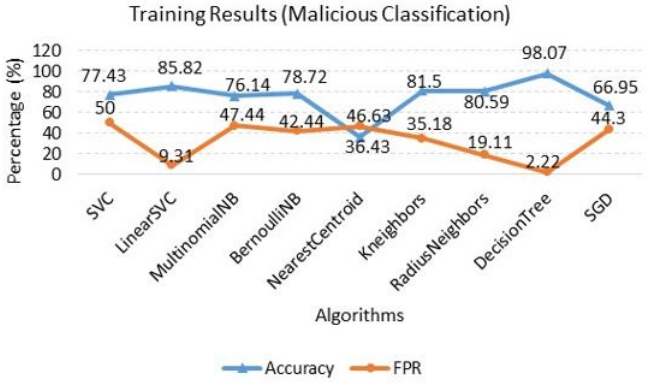


Figure 3: Accuracy and False Positive Rate of Each Algorithm

2) Classification Result Using the Best Algorithm: As explained above, we use Decision Tree for the classification of malicious domains. We got the results as depicted in Table XIII. Most of the blacklists contains larger number of landing domains than number of distribution domains **at least 1.5 times**. This is reasonable because a distribution domain may have multiple corresponding landing domains that redirect users to the distribution domain. Concretely, we found that the landing domains occupy at least 60% of total distinct domains in each blacklist. Especially, in the group II (large public blacklists), the landing domains occupy even larger than 75% of total distinct domains in each blacklist.

G. Measure 7: New Blacklist Version

The new updated public blacklists were downloaded on the same date 2017/11/09, which is over 7 months after the date we downloaded and analyzed the blacklists in the previous paper [1] (2017/02/28) as described in Section II-A. The new version of the blacklists is described in Table XIV. Compared

TABLE XIII: LANDING AND DISTRIBUTION IN THE BLACKLISTS.

No	Blacklist	#Distinct Domains	#Landings	#Distributions
0	Total	50,519	37,815 (74.85%)	12,704 (25.15%)
1	AMA	77	55 (71.43%)	22 (28.57%)
2	ANE	2	0 (00.00%)	2 (100.0%)
3	APH	367	234 (63.76%)	133 (36.24%)
4	ARA	3	3 (100.0%)	0 (00.00%)
5	AZE	21	14 (66.67%)	7 (33.33%)
6	AMAL	98	62 (63.27%)	36 (36.73%)
7	AMV	2,176	1,474 (67.74%)	702 (32.26%)
8	AHO	5,060	3,423 (67.65%)	1,637 (32.35%)
9	AME	19,812	15,232 (76.88%)	4,580 (23.12%)
10	ASH	32,248	24,408 (75.69%)	7,840 (24.31%)
11	AUR	33,674	25,508 (75.75%)	8,166 (24.25%)
12	AUT	24,020	18,411 (76.65%)	5,609 (23.35%)
13	AGSBv3	189	134 (70.90%)	55 (29.10%)
14	AGSBv4	639	389 (60.88%)	250 (39.12%)

TABLE XIV: NEW VERSIONS OF THE BLACKLISTS.

No	Group	Blacklists	#Domains	Downloaded on
1	(I)	MA	14,233	2017/11/09
2		NE	243	2017/11/09
3		PH	9,435	2017/11/09
4		RA	9,204	2017/11/09
5		ZE	367	2017/11/09
6		MAL	901	2017/11/09
7		MV	5,497	2017/11/09
8		HO	333,091	2017/11/09
9		ME	410,212	2017/11/09
10		UT	490,829	2017/11/09
11	(II)	SH	1,254,260	2017/11/09

Note that, the fourth column is the number of *unique* domains. Some raw blacklists contain many redundant domains.

with the previous blacklist version, this new version has several big changes:

- *UR is no longer available from 2017/07/25*. The blacklist provider of *urlblacklist.com* "has closed down, shut of its website, and thrown in the towel, they have refunded current subscribers and closed up shop" [18].
- *ME and UT no longer belong the group II (large public blacklists) but now belong to group I (small public blacklists)* since their numbers of entries are significantly reduced: 3.09 times in case of ME (from 1,266,334 entries to only 410,212 entries) and 2.74 times in case of UT (from 1,346,788 entries to only 490,829 entries). Besides ME and UT, the number of entries in MV is also significantly reduced (39.7 times from 218,248 entries to 5,497 entries); however, MV still belongs to the group I.
- On the contrary, *the numbers of entries in RA and HO are significantly increased*: 6.67 times in case of RA (from 1,380 entries to 9,204 entries) and 55.76 times in case of HO (from 5,974 entries to 333,091 entries). *For RA, which mainly focuses on ransomewares, its significant increase in the number of entries probably indicates the significant increase in the number of domains, which are infected by new ransomwares. The most recently serious ransomwares that have widely affected a lot of computers around the world are*

WannaCry *discovered in 2017/05 [19] and a new variant of* ***Petya*** *discovered on 2017/06/27 [20]*. These two dates fall in the period between the dates of downloading our previous and new blacklist versions.

Besides the differences between the previous and new blacklist versions, we also analysed the TLDs of the new blacklist version and have some new findings:

- Unlike the previous analysis, in this analysis we found *many TLDs in the type of new generic top-level domains (new gTLD)* which is the sixth category mentioned in Section II-B2 such as: *.forsale*, *.institute*, *.church*, *.download*, etc.
- The HO is *surprisingly the blacklist having the highest number of distinct TLDs (506 TLDs), which is even much higher than the number of distinct TLDs in the blacklists that have much larger number of entries*. For example, the number of entries of SH (1,254,260) is 3.77x larger than that of HO (333,091) but the number of corresponding TLDs of SH (506) is 1.52x smaller than that of HO (332).

IV. Discussion

In this section, we discuss several issues that can be addressed in future work.

A. Blacklist Extension

In this article, we analyze 14 popular blacklists. We are planning to analyze other private blacklists. The most prioritized candidate is VirusTotal (virustotal.com). VirusTotal checks domains/URLs by referring 40 other antivirus blacklists (however, all blacklists are not always used). VirusTotal also refers the feedbacks/comments from users. Besides the blacklists and user feedbacks, we currently do not know whether it has its own method to classify a domain/URL into malicious or benign. Furthermore, we plan to extend our analysis from domain blacklists to IP, URL and DNS blacklists. Two prioritized candidates are MXTools or also known as Spamhaus (mxtools.com) and Mxtoolbox (mxtoolbox.com), which provide large number of IP entries.

B. Analysis Extension

We plan to extend our current six measures to three other interesting and important measures, which can help to understand the blacklists better:

1) Measuring the registration time of malicious domains in each blacklist: The registration time here means the response time of each blacklist to a malicious domain. For example, when a domain D becomes malicious on 2017/05/01, blacklist A lists D in its dataset on 2017/05/02 but blacklist B lists D in its dataset on 2017/05/03; and thus, A is better than B. The challenge is that, not all blacklists provide this information. A naive method is to download each blacklist periodically to check whether specific malicious domains appear in each blacklist. For example, [21] analysed the blacklist update frequency by monitoring download site. This method requires high communication costs and also cannot deal with private blacklists which do not allow to directly download blacklists. Therefore, better solutions should be investigated to analyse registration time of malicious domains in blacklists.

TABLE XV: RAW WHOIS OF THE DOMAIN 'DNI.RU'

% By submitting a query to RIPN's Whois Service	
% you agree to abide by the following terms of use:	
% http://www.ripn.net/about/servpol.html#3.2 (in Russian)	
% http://www.ripn.net/about/en/servpol.html#3.2 (in English).	
domain:	DNI.RU
nserver:	ns1.goodoo.ru.
nserver:	ns2.goodoo.ru.
state:	REGISTERED, DELEGATED, UNVERIFIED
org:	OOO "Dni.ru"
registrar:	RD-RU
admin-contact:	https://cp.mastername.ru/domain_feedback/
created:	2000-06-06T14:58:03Z
paid-till:	2018-06-05T21:00:00Z
free-date:	2018-07-07
source:	TCI

2) How to decide whether a domain is malicious based on some blacklists when each blacklist has its own ground truth: A naive method is based on *majority rule*. That is, if a domain is detected by larger than 50% number of blacklists, it can be determined as a malicious domain. Another better method is based on the weight of malicious domain in each blacklist. For example, a blacklist A weights a malicious domain D at 80% while another blacklist B weights it at 30%; then we can weight D at 55%, which is the average weight. Similar to the above analysis about registration time, the challenge is that almost all blacklists do not provide the information about malicious weighting. Therefore, finding how to weight domains in each blacklist is a promising approach to label a domain into malicious or benign.

3) Whois-text-based method for the measure 6 (malicious categories): The method used for the measure 6 (malicious categories) in this is based on some HTML elements that are commonly used in landing and distribution pages along with text mining on the entire HTML documents as described in Section II-B5. In future work, we plan to *implement another method, which has been recently published in [22] and compare with the method used in this paper. The method used in [22] is based on the text mining on entire Whois documents of the domains*. Each domain has each own registration information; and instead of extracting its Whois attributes separately (similar to the measure 3, which uses the attribute *creation date* of the Whois, or the measure 4, which uses the attribute *country* of the Whois), this new method retrieves whole raw texts of the Whois and applies text mining on them. Examples of Whois raw texts are given in Tables XV and XVI.

4) Improving the library of extracting Whois attributes: When extracting Whois attributes (i.e., *creation date* in the measure 3 and *country* in the measure 4), along with using the libraries (e.g., *python-whois* 0.6.5 used in our experiment), high manual operational cost and time-consuming computation are required due to the following challenges.

- First, the Whois information stored in different servers is very unstructured, and some Whois attributes are not always available. For example, we can observe that the Whois structures of *.ru* (Table XV) and *.kr* (Table XVI) are very different. The attributes *last update date* or *address* are not available in the Whois of 'dni.ru', but available in that of 'kddi-research.jp'.

TABLE XVI: RAW WHOIS OF THE DOMAIN 'KDDI-RESEARCH.JP'.

[JPRS database provides information on network administration. Its use is]	
[restricted to network administration purposes. For further information,]	
[use 'whois -h whois.jprs.jp help'. To suppress Japanese output, add'/e']	
[at the end of command, e.g. 'whois -h whois.jprs.jp xxx/e'.]	
[Domain Name]	KDDI-RESEARCH.JP
[Registrant]	KDDI R&D Laboratories, INC
[Name Server]	kddfuji.kddlabs.co.jp
[Name Server]	tao.kddlabs.co.jp
[Signing Key]	
[Created on]	2016/08/01
[Expires on]	2018/08/31
[Status]	Active
[Last Updated]	2017/09/01 01:05:10 (JST)
Contact Information:	
[Name]	KDDI R&D Laboratories, INC
[Email]	email@lan.kddilabs.jp
[Web Page]	
[Postal code]	356-8502
[Postal Address]	2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502, JAPAN
[Phone]	0492-78-7441
[Fax]	0492-78-7510

Second, some attribute values in different Whois servers are very divert. For example, the attribute *country* in different domains is registered as "US", "USA" or "America", which all have the same meaning. Third, we cannot be able to consider all attributes of all domain servers but only some common ones (e.g., creation date, expiration date, registrar, country and organization) since they do not have any standard.

- Even if we use entire Whois text of each domain instead of separately extracting each Whois attribute (as previously discussed in the item 3), there are still some other challenges. First, Whois of a domain can be stored in one or multiple Whois servers. Famous Whois servers can contain Whois of almost all of domains; but for some domains, we need to manually find its corresponding Whois server. Second, English is not always supported in Whois servers. For example, the server *whois.vnnic.vn* only supports Vietnamese, or *ewhois.cnnic.cn* only supports Chinese. Although the Whois are known to be readable-and-understandable by human, semantic language processing is required.

As far as we know, until now there is no library or automatic method which can *completely* standardize Whois information of all kinds of domains. For future work, we plan to improve the existing libraries by adding different patterns for other domains TLDs that the libraries have not supported. Also, we plan to construct English framework for some servers that do not have English Whois.

V. Conclusion

In this article, we analyse 14 popular blacklists downloaded on 2017/02/28 including 8 small public blacklists, 4 large public blacklists and 2 private blacklists by Google. We designed seven important measures including blacklist intersections, TLDs, domain ages, countries, web content categories, malicious categories, and some new findings between the current and new blacklist versions. Especially, we construct our two models using machine learning to analyse the last 2 measures. We finally found several important results: Google is developing GSBv3 and GSBv4 independently; the large public blacklist *urlblacklist.com* contains 98% entries in the blacklist *dsi.ut_capitole.fr*; most of domains in all the blacklists are created in 2000 with 6.08%, and from United States with 24.28%; GSBv4 can detect younger domains compared with other blacklists; (v) *Tech & Computing* is the dominant web content category, and the blacklists in each group have higher correlation in web content than the blacklists in other groups; and (vi) the number of landing domains is larger than that of distribution domains at least 75% in group II (large public blacklists) and at least 60% in other groups. For the final measure, we collected the most updated versions of 11 public blacklists as of this paper (downloaded on 217/11/09), and analysed the differences between the two blacklist versions. We observed some significant changes such as: UR is no longer available from 2017/07/25; ME and UT now belong to group I (small public blacklists) rather than group II (large public blacklists) as in the previous versions; the number of malicious domains injected by ransomwares is significantly increased; and many new-generic TLDs appear such as *.forsale*, *.institute*, *.church* unlike the previous analysis. We also discussed several challenges in analysing registration time of malicious domains, the way to determine a malicious domain, Whois standardization and malicious classification using text mining on entire Whois documents.

Acknowledgement

This research was carried out as part of WarpDrive: Web-based Attack Response with Practical and Deployable Research InitiatiVE, a Commissioned Research of the National Institute of Information and Communications Technology (NICT), JAPAN.

References

[1] T. P. Thao, T. Makanju, J. Urakawa, A. Yamada, K. Murakami, and A. Kubota, "Large-Scale Analysis of Domain Blacklists". In: *Proceedings of the 11th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'17)*, 2017.

[2] VeriSign, Inc., "Internet Grows to 326.4 Million Domain Names in the First Quarter of 2016". Available: https://investor.verisign.com/releaseDetail.cfm?releaseid=980215. Retrieved: 2016/07/19.

[3] Symantec, Inc. "Internet Security Threat Report". Available: https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf.

[4] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, and J. Hong, "An Empirical Analysis of Phishing Blacklists", *6th Conference on Email and Anti-Spam (CEAS)*, 2009.

[5] M. Kuhrer and T. Holz, "An Empirical Analysis of Malware Blacklists", *Praxis der Informationsverarbeitung und Kommunikation*, vol. 35, no. 1, p. 11, 2012.

[6] M. Kuhrer, C. Rossow, and T. Holz, "Paint it Black: Evaluating the Effectiveness of Malware Blacklists", *17th Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pp. 1-21, 2014.

[7] M. Vasek and T. Moore, "Empirical analysis of factors affecting malware URL detection", *eCrime Researchers Summit (eCRS'13)*, pp. 1-8, 2013.

[8] C. J. Dietrich, and C. Rossow, "Empirical research of IP blacklists", *Securing Electronic Business Processes (ISSE'08)*, pp. 163-171, 2008.

[9] T. Ouyang, S. Ray, M. Allman, and M. Rabinovich, "A large-scale empirical analysis of email spam detection through network characteristics in a stand-alone enterprise", *Journal of Computer and Telecommunications Networking*, vol. 59, pp. 101-121, 2014.

[10] J. Jung and E. Sit, "An empirical study of spam traffic and the use of DNS black lists", *4th ACM SIGCOMM conference on Internet measurement (IMC'04)*, pp. 370-375, 2004.

[11] D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: a fast filter for the large-scale detection of malicious web pages", *20th Conference on World wide web (WWW'11)*, pp. 197-206, 2011.

[12] R. J. Walls, E. D. Kilmer, N. Lageman, and P. D. McDaniel, "Measuring the Impact and Perception of Acceptable Advertisements", *Internet Measurement Conference (IMC'15)*, pp. 107-120, 2015.

[13] List of IAB Categories. Available: https://www.iab.com/guidelines/iab-quality-assurance-guidelines-qag-taxonomy/. Retrieved: 2015/09/01.

[14] ICANN, "List of Top-Level Domains". Available: http://data.iana.org/TLD/tlds-alpha-by-domain.txt

[15] Digital Arts. Available: http://www.daj.jp/en/

[16] SimilarWeb. Available: https://developer.similarweb.com/

[17] G. Wang, J. W. Stokes, C. Herley, and D. Felstead, "Detecting Malicious Landing Pages in Malware Distribution Networks", *43rd IEEE/IFIP Conf. on Dependable Systems and Networks (DSN'13)*, pp. 1-11, 2013.

[18] B. E. Nichols, "July 25 2017 - Urlblacklist.com Is No More", https://groups.google.com/forum/#topic/e2guardian/7WeHpD54LE

[19] B. Brenner, "WannaCry: the ransomware worm that didn't arrive on a phishing hook", *Naked Security. Sophos* Retrieved 18 May 2017. Available: https://nakedsecurity.sophos.com/2017/05/17/wannacry-the-ransomware-worm-that-didnt-arrive-on-a-phishing-hook/

[20] "Global ransomware attack causes chaos", *BBC News*. Retrieved 27 June 2017. Available: http://www.bbc.com/news/technology-40416611

[21] Y. Takeshi et al., "Analysis of Blacklist Update Frequency for Countering Malware Attacks on Websites", *IEICE Transactions on Communications*, vol. E97-B, no. 1, pp. 76-86, 2014.

[22] T. P. Thao, A. Yamada, K. Murakami, J. Urakawa, Y. Sawaya, and A. Kubota, "Classification of Landing and Distribution Domains using Whois's Text Mining", *16th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-17)*, pp. 1-8, 2017.

Simulating Blast Effects on High Security Vehicles with 3D Fluid-Structure Interaction

Arash Ramezani, Burghard Hillig, Hendrik Rothe
Chair of Measurement and Information Technology
University of the Federal Armed Forces
Hamburg, Germany
Email: ramezani@hsu-hh.de, burghard.hillig@hsu-hh.de, rothe@hsu-hh.de

***Abstract*—The present time is shaped by a variety of religious, political and military conflicts. In times of asymmetric warfare and constantly changing sources of danger from terrorist attacks and other violence based crimes, the personal need for protection continues to rise. Aside from military applications, there is a large area for the use of high security vehicles. Outwardly almost indistinguishable from the basic vehicles, security vehicles are used for protecting heads of state, as well as individuals. To remain state of the art it is necessary for security vehicles to permanently continue to develop protection against modern weapons and ammunition types. It is enormously cost intensive to check any new technology by firing or blasting of real vehicles. Therefore, more and more calculations of new security concepts and materials are carried out by numerical computer simulations. However, product simulation is often being performed by engineering groups using niche simulation tools from different vendors to simulate various design attributes. The use of multiple vendor software products creates inefficiencies and increases costs. This paper will present the analysis and development of an interface between the most common Computer Aided Engineering applications ANSYS Autodyn and Abaqus to exploit the advantages of both systems for the simulation of blast effects. As a result, the interaction between a shock wave from ANSYS Autodyn and a vehicle model in Abaqus is shown. As part of this publication, the underlying material models and the connection between simulation and interface will be presented.**

Keywords–CFD-FEM coupling methods; fully automatic structure analysis; high-performance computing techniques; blast loading; vehicle structures.

I. INTRODUCTION

This article is based on the developments, which were presented at the SIMUL 2017 in Athens [1]. A new approach has been made to exploit the full potential of the two leading software providers for Computational Fluid Dynamics (CFD) / Finite Element Method (FEM) calculations ANSYS (Canonsburg, USA) and Abaqus FEA from Dassault Systmes (Vlizy-Villacoublay, France). Therefore, an interface between these two software platforms has been developed to combine their individual strengths.

Since the 1960s, the simulation of physical processes has been a steadily growing and integral part of Computer Aided Engineering (CAE). Especially, the Computational Fluid Dynamics and the discretization of complex models using the Finite Element Method have made an impressive development from individual highly specialized applications to the standard of industrial product development [2] [3]. This process was supported by the progressive development of increasingly powerful and less expensive computer hardware. Together with specialized software, a triumph of the simulation of physical processes in everyday technical work has emerged. Positive effects due to the use of simulation tools have been shorter development times, lower production costs, more innovative products, improved security and higher quality. The previous modeling of components and objects of the real world by Computer Aided Design (CAD) software is an important prerequisite for the efficient use of the simulation tools. This has been established as a standard in the automotive industry, so that almost every part of a vehicle can be constructed by using CAD. These complete and realistic vehicle models can be analyzed virtually by available simulation software.

The two leading software providers offer software with similar features available. One key difference is that ANSYS is able to simulate blast propagation and, in contrast, Abaqus does not [4]. Additionally, their performance is characterized by different spreads and focuses. For example, Abaqus and CATIA, a CAD software, which is also distributed by Dassault Systmes, is predominantly used by the automotive industry and provides excellent opportunities for the simulation and modeling of complete vehicles. This includes screwed and adhesive connections. On the other hand, ANSYS offers a wide range of sophisticated simulation capabilities in the field of CFD, which includes the modeling and simulation of explosive detonations and the subsequent propagation of shock waves [5]. The different focuses of the performance of ANSYS and Abaqus yield to a mixed, demand-based use of the software in the research and development area, so that different software is used even within the same company on the same project in different areas of activity. This circumstance is amplified by the fact that product simulations are performed by engineering groups using niche simulation tools from different vendors to simulate various design attributes. Unfortunately, the software providers avoid the effective interaction of their simulation tools due to mutual competition. This complicates the development effort and results in longer development times in research and industry. Particularly, in the area of armored security vehicles, it is necessary to remain state of

the art and to constantly consider the ongoing development of modern weapon and ammunition types. Experimental tests of the harmful effects of new technologies by blast or impact is associated with enormous time and financial costs.

This paper reports on the development of an interface between ANSYS and Abaqus, which will enable combining the strengths of the two leading software providers with the aim of generating synergies that result in short development times and lower costs. This interface allows an iterative transfer of the blast simulation of ANSYS to the structural mechanic solver of Abaqus, which simulates the effects on the vehicle model and vice versa. The development of such an interface represents an absolute novelty. The goal of this development is to combine the advantages of ANSYS, with its advanced CFD / blast simulation capabilities, and Abaqus, which is widely used in the automotive industry. One step to achieve this, is to export the blast simulation data at a certain point depending on space and time. Then the exported data has to be inserted in Abaqus. Hereby, the restrictions for importing data and scripting has to be considered. The newly created interface ensures that Abaqus places the data in the correct position. Now, Abaqus can use this imported data to simulate the interaction between the model and the blast wave. Some results of this working interface are also demonstrated in this paper.

After the introduction in this section, the different methods of space discretization and fundamentals of simulation are described in Section II. In addition, there are short subsections on material models and on ballistic trials where the experimental set-up is depicted, followed by Section III describing the analysis with numerical simulations. The paper ends with a concluding paragraph in Section IV.

II. Materials and Methods

A. State of the Art

A first step in developing an interface between ANSYS and Abaqus has already been reported in [6]. The developed interface allows accessing a set of data and passing them to Abaqus. Python was used as a programming language. ANSYS provides the data records for the interface as .txt files. These files contain data points with Cartesian coordinates, which describe the propagation of shock waves after blasting. The interface takes this data and splits it into separate information. In a further step, the data is stored in a list, linked with the corresponding time points, pressure data and coordinates. It is also possible to use a set of data and to interpolate between the time points to produce a larger data set. After the data has been written and saved in a linked form, the interface retrieves the CAD model. Subsequently, the explosion data can be projected onto a selectable surface of the model. Then, an iterative loop realizes the coupling between CFD and FEM simulations. This approach for a coupled CFD-FEM analysis is called ”strong coupling”. In another approach, the ”semi-strong coupling”, a smaller amount of data is used and mathematically interpolated for a sufficient approximation. The third concept is a ”weak coupling” solution. In the weak or loose coupling methods the coupled problem is partitioned into fluid and structural parts, which are solved separately. The data exchange on the interface is done only once per time step and even not at every time-step. Here, neural networks and deep learning can be used to replicate blast effects on different vehicle structures. Until now, the basic functionality of the interface could be validated on different models, including the model of a safety vehicle.

B. Fundamentals of Simulation

In the security sector, the partly insufficient safety of people and equipment due to failure of industrial components are ongoing problems that cause great concern. Since computers and software have spread into all fields of industry, extensive efforts are currently being made in order to improve the safety by applying certain computer-based solutions. To deal with problems involving the release of a large amount of energy over a very short period of time, e.g., explosions and impacts, there are three approaches, which are discussed in [7].

As the problems are highly non-linear and require information regarding material behavior at ultra-high loading rates, which are generally not available, most of the work is experimental and may cause tremendous expenses. Analytical approaches are possible if the geometries involved are relatively simple and if the loading can be described through boundary conditions, initial conditions, or a combination of the two. Numerical solutions are far more general in scope and remove any difficulties associated with geometry [8].

For structures under shock and impact loading, numerical simulations have proven to be extremely useful. They provide a rapid and less expensive way to evaluate new design ideas. Numerical simulations can supply quantitative and accurate details of stress, strain, and deformation fields that would be very expensive or difficult to reproduce experimentally. In these numerical simulations, the partial differential equations governing the basic physical principles of conservation of mass, momentum, and energy are employed. The equations to be solved are time-dependent and non-linear in nature. These equations, together with constitutive models describing material behavior and a set of initial and boundary conditions, define the complete system for shock and impact simulations.

The governing partial differential equations need to be solved in both time and space domains. The solution over the time domain can be achieved by an explicit method. In the explicit method, the solution at a given point in time is expressed as a function of the system variables and parameters, with no requirements for stiffness and mass matrices. Thus, the computing time at each time step is short but may require numerous time steps for a complete solution. The solution for the space domain can be obtained utilizing different spatial discretization, such as Lagrange [9], Euler [10], Arbitrary

Lagrange Euler (ALE) [11], or mesh free methods [12]. Each of these techniques has its unique capabilities, but also limitations. Usually, there is not a single technique that can cope with all the regimes of a problem [13]. The crucial factor is the grid that causes different outcomes. More details are discussed in Section IV.

Due to the fact that all engineering simulations are based on geometry to represent the design, the target and all its components are simulated as CAD models. Real-world engineering commonly involves the analysis and design of complicated geometry. These types of analysis depend critically on having a modeling tool with a robust geometry import capability in conjunction with advanced, easy-to-use mesh generation algorithms [14]. It often is necessary to combine different simulation and modeling techniques from various CAE applications. However, this fact can lead to major difficulties, especially in terms of data loss and computational effort. Particularly the leading software providers prevent an interaction of their tools with competing products. But to analyze blast loading and its effects on vehicle structures, different CAE tools are needed. Therefore, it is important that an interface is provided that allows a robust interaction between various applications. Using a CAD neutral environment that supports direct, bidirectional and associative interfaces with CAE systems, the geometry can be optimized successively and analysis can be performed without loss of data [15].

Various approaches are possible when it comes to solving problems that involve the release of large amounts of energy in very short periods of time, which then propagate as shock waves or act as impact on structures. Analytic solutions offer a very powerful way to describe such a process. Unfortunately, their applicability is restricted to problems with simple geometries and few boundary and initial conditions. In contrast, numerical simulations offer much more general applications with complex structures and feasible solutions.

The underlying physical model of numerical simulations is provided by physical conservation laws, the equation of state and the constitutive model. Partial differential equations for the conservation of energy, momentum, and mass form the physical conservation laws. Furthermore, the equation of state combines the internal energy or temperature and the density or volume of a material with the pressure. As a result, changes in the density and irreversible thermodynamic processes such as shock-like heating can be considered. In addition, the constitutive model includes the influence of the material to be simulated and describes the effect of deformation, i.e. changes in shape and material strength properties.

Together, these equations form a set of coupled, time- and location-dependent, highly non-linear equations, which can be solved by computer calculations. The governing partial differential equations need to be solved in both time and space domains. The solution over the time domain can be obtained by an explicit method, which is an iterative method and leads to a step by step solution in the time domain. Software for numerical simulation of shock and impact processes is called a hydrocode [16].

C. Methods of Space Discretization

All existing structural dynamics and wave propagation codes obtain solutions to the Differential Equations (DEs) governing the field by solving an analogous set of algebraic equations. The governing DEs are not solved directly, because currently only a handful of closed-form solutions for DEs are available. The equations of structural dynamics, being a coupled set of rate equations, which account for the effects of severe gradients in stress, strain and deformation, material behavior ranging from solid to fluid to gas, temperatures from room temperature to melt temperature are highly non-linear and do not lend themselves to closed-form solutions in the general case.

To get a solution over the spatial domain a discretization of the material with a mesh is necessary. FEM uses such a discretization by dividing the problem space into separate elements. These elements can have different shapes: In two dimensions, the shape of quadrilaterals or triangles, in three dimensions hexahedrons and tetrahedrons are usually used. Even complicated geometries can be formed with these elements. Each FEM element has a certain number of nodes, which are located at its corners and have known spatial coordinates. The displacement of these nodes represents the unknowns of the partial differential equations to be solved. There are multiple, different spatial discretization methods related to FEM, such as Lagrange, Euler, Arbitrary Lagrange Euler (ALE) or mesh free methods. Each of these methods can be used independently, but some specific problems need a combination of different discretization methods.

1) Lagrange: The Lagrange method divides an object into a spatial grid where the grid is fixed to the object and moves with it. The material components within an element do not change. If forces are acting on a node, it is displaced, and thus the forces are transmitted to its neighboring nodes, similar to a spring-mass system. This results in deformations of the grid. The nodes of the edge elements of an object remain unchanged so that the boundary and interface conditions can be easily applied. Clear material boundaries are also available so that space outside the material does not require an extra grid and therefore the conservation of mass is automatically satisfied. Figure 1 shows two objects with its mesh as an example of the Lagrange method.

Two objects consisting of different materials represented by the colors blue and green before (left side) and after impact (right side). The two colors stand for different materials in such a way, that the green material is harder than the blue one. Additionally, the green object has an initial velocity in the direction of the blue object. The right side of the figure shows the discretization dependent deformation after the impact with the Lagrange solver. The mesh is bound to the objects and divides them into multiple elements. After an impact the

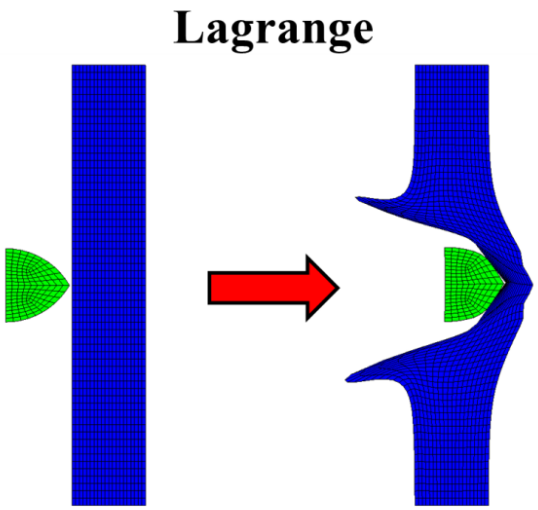


Figure 1. Lagrange method example: A blue target gets hit by a green bullet before (left side) and after the impact (right side).

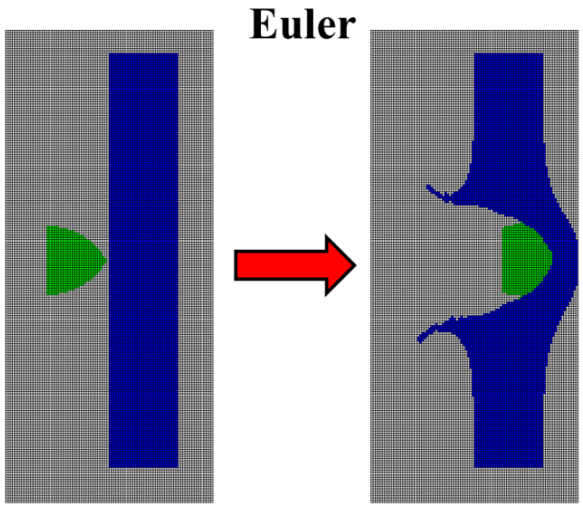


Figure 2. Euler method example: A blue target gets hit by a green bullet before (left side) and after the impact (right side). At both sides, the mesh fills the entire space.

objects deform due to the deformation of the elements. A weak point of the Lagrange method is a strong distortion of the mesh in heavily loaded regions, as shown in Figure 1 in the area adjacent to the green and blue object.

In general the Lagrange method is best suited for complex geometries and structures, projectiles and other solids. A disadvantage of Lagrange is the occurrence of strong distortions of mesh element at high loads. Such a distorted element can adversely affect the temporal solution of the simulation since the time step is proportional to the size of the smallest element.

2) Euler: In the Euler method the coordinates of the nodes are fixed and form the entire mesh of the region to be solved. The material flows through the mesh as a function of time and changes the value of the element, while the spatial coordinates and the nodes remain fixed. This is the reason why no element distortion is possible in the Euler method. In contrast to Lagrange, boundary nodes do not necessarily coincide together with material boundary conditions. Thereby difficulties can arise with the application of boundary and interface conditions. Figure 2 shows two objects and the mesh as an example of the Euler method. Two objects consisting of different materials represented by the colors blue and green before (left side) and after impact (right side). In contrast to the Lagrange frame, the mesh fills the entire space. Again, the green object has an initial velocity in the direction of the blue object. The right side of the figure shows the discretization dependent deformation after the impact with the Euler solver. The mesh is not bound to the objects like in the Lagrange frame. Instead the mesh fills the whole space with the objects and empty space between them. During the simulation the material of the objects is transported through the mesh of the space. After an impact the mesh stays clear but its content is partly deformed.

In general the Euler method is used to model the propagation of gases and fluids as a result of an explosion or impact. In the investigation of solids, the Euler method has a disadvantageous effect, since additional calculations are needed to transport the stress tensor and the history of the material through the lattice. In this case, Euler needs more computing performance and smaller elements to resolve the occurring shock waves.

3) ALE: The Arbitrary Lagrange Euler (ALE) method is a mix of Lagrange and Euler method. ALE allows an arbitrary redefinition of the mesh on each calculation step. Different predefined grid motions can be specified, such as free (Lagrange), fixed (Euler), equipotential, equal spacing and others. As an advantage distortions can be avoided. On the other hand, additional computation steps are necessary to move and to convert the grid. An example of ALE is shown in Figure 3.

Two objects consisting of different materials represented by the colors blue and green before (left side) and after impact (right side). The blue object has an initial velocity in the direction of the green object. The right side of the figure shows the discretization dependent deformation after the impact with the ALE solver. In comparison to the pure Lagrange method (Figure 1), no lattice distortions occur here.

4) SPH: Smoothed Particle Hydrodynamics is a method which is not based on a fixed topological lattice but on a finite set of particles. These particles are embedded to the material similar to the nodes of the Lagrange method, but their connections are not fixed. However, the particles represent not only mass points, but also interpolation points for the calculation of the physical variables. The calculations are based on the data of the neighboring particles and are scaled by a weighting function. Unlike Lagrange, no grid distortion can occur at SPH, since no grid exists. Related to the Euler

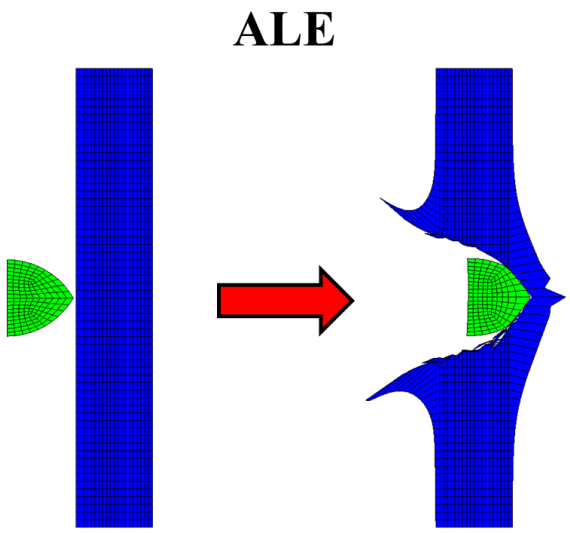


Figure 3. ALE method example: A blue target gets hit by a green bullet before (left side) and after the impact (right side). Between the individual simulation steps the mesh was redefined using either the Lagrange or the Euler method.

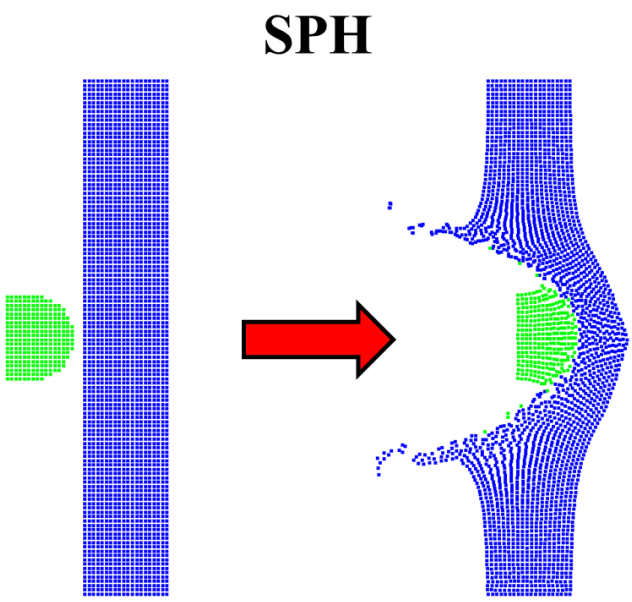


Figure 4. SPH method example: A blue target gets hit by a green bullet before (left side) and after the impact (right side). Both objects consist of a multitude of particles.

method, SPH has the advantage that all material boundaries and interfaces are clearly defined. Figure 4 illustrates two objects consisting of different materials in the SPH frame represented colored particles before (left side) and after impact (right side). The green object has an initial velocity in the direction of the blue object. The right side of the figure shows the discretization dependent deformation after the impact with the SPH solver. As seen in Figure 4 two objects consist of small particles in the SPH frame. Their behavior before and after an impacts differs from the solutions in the Lagrange, Euler or ALE frame.

The SPH method has proven especially useful in the simulation of impact processes on brittle materials [7]. It should be noted that the modeling of the material as particles leads to significantly higher computing effort per time step.

For problems of dynamic fluid-structure interaction (FSI) and impact, there typically is no single best numerical method which is applicable to all parts of a problem. Techniques to couple types of numerical solvers in a single simulation can allow the use of the most appropriate solver for each domain of the problem.

5) Mesh Size: One of the more important issues, which have to be carefully considered is the issue of mesh size. Different results are obtained if the number of cells per unit length is not adequate. For example, it was found that for penetration studies with eroding long rods, the number of cells on the rod's radius should be at least eleven. The same density of cells should be kept in the target, at least for several projectile radii around its symmetry axis. In order to save computing time, the cell size at farther zones can be gradually increased according to their distance from the symmetry axis. The mesh cell size depends on the specific problem. As an example, a small cell size should be considered in cases where there is a fracture in the projectile or target. It is recommended that while preparing the code for its final runs, the numerical convergence with respect to mesh cell size should be checked. Another important issue, especially when material elements are expected to deform considerably, is the issue of erosion with Lagrangian codes. At large deformations the code may run into trouble when treating heavily deformed elements. The use of the erosion threshold condition is then necessary in order to eliminate elements at a predetermined value of the plastic or geometric deformation. The erosion should be monitored constantly, and when it is too high one should replace the Lagrangian with an Eulerian code.

The goal of this paper is to evaluate an interface between different hydrocodes, computational tools for modeling the behavior of continuous media. In its purest sense, a hydrocode is a computer code for modeling fluid flow at all speeds. For that reason, a structure will be split into a number of small elements. The elements are connected through their nodes (see Figure 5). The mesh divides the object into small elements connected by its nodes.

The behavior (deflection) of the simple elements is well-known and may be calculated and analyzed using simple equations called shape functions. By applying coupling conditions between the elements at their nodes, the overall stiffness of the structure may be built up and the deflection/distortion

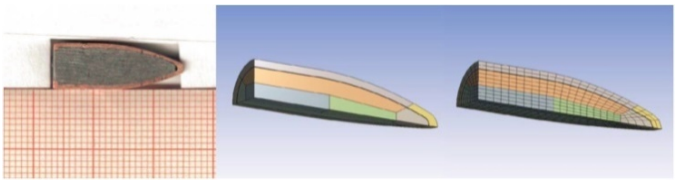


Figure 5. The left side shows the cross-section of a bullet over millimeter paper, which could theoretically divide the bullet into multiple elements. The right side shows an example of a mesh in the numerical simulation. Different parts are color coded in the representation of the bullet.

of any node and subsequently of the whole structure can be calculated approximately [17]. Therefore, several runs are necessary: From modeling to calculation to the evaluation and subsequent improvement of the model.

Hydrocodes, or wave propagation codes, are a valuable adjunct to the study of the behavior of metals subjected to high-velocity impact or intense impulsive loading. The combined use of computations, experiments and high-strain-rate material characterization has, in many cases, supplemented the data achievable by experiments alone at considerable savings in both cost and engineering man-hours.

A large database exists of high-pressure Equation-Of-State (EOS) data. Considerable data on high rate deviatoric behavior exists as well although, unlike EOS data, it is not collected in a few compilations but scattered throughout a diverse literature. Experimental techniques exist for determining either EOS or strength data for materials not yet characterized under high-rate loading conditions.

By contrast, computations with non-metallic materials such as composites, concrete, rock, soil and a variety of geological materials are, in effect, research tasks. This is due to several reasons: lack of definitive computational models for high strain rate-temperature-pressure response; lack of a database for EOS and high rate strength data for such materials; lack of test methodologies for anisotropic materials subjected to high-rate loading.

A large number of ad hoc models exist for explosives, geological materials, concrete and other non-metallics. Many of these lack a firm theoretical foundation. This is an area where considerable research is required, both to devise appropriate test techniques to measure material response under high strain rates, elevated temperatures and high pressures as well as to develop appropriate constitutive models.

D. Material Models

To understand the completed set-up which produces the necessary data that can be used by the interface, the underlying material models and its interactions have to be examined.

The source of an explosion can be a gas, a liquid or solid material. The consequence of the ignition of such sources is the rapid release of the compressed energy. The rapid release causes relatively huge displacments of the gas molecules in the surrounding air. As the gas molecules interact with each other, a propagating wave is formed. If this wave has its origin in the detonation of an explosive charge, it is called a ”blast wave” [18]. This wave is a non-linear wave, which means that it travels faster than the sound speed in the fluid. Every point in space reached by the wave suffers a rapid and sharp change in pressure, temperature and density of the medium. That is the reason why a blast wave affects the thermodynamic, state and dynamic of ambient air. In addition, when the blast wave reaches a solid target, an interaction of the wave and the solid material in form of stress and deformation must occur. this interaction can be simulated through material models of the solid materials.

1) Air: Air is called the gas mixture of the atmosphere. Dry air mainly consists of the two gases nitrogen and oxygen.

An equation that describes the hydrodynamic response of a material is called equation of state. For gases, a hydrodynamic behavior is assumed, so that their reaction to the dynamic load is done by a variation of pressure as a function of density and internal energy. A hydrodynamic reaction is the primary response for gases and liquids that can not suffer shear stress.

One of the simplest forms of equation of state is that for an ideal gas, which can be used for air. But the ideal gas equation is also used in applications involving the motion of other gases. In ANSYS Autodyn the used ideal gas equation has the following form [19]:

$$P = (\gamma - 1) \cdot \rho \cdot E \tag{1}$$

In this form of equation of state, P describes the pressure, γ the value of the adiabatic exponent, ρ the density and E is the internal energy of the gas. A complete set of values for these properties can be found in Table I.

TABLE I. PROPERTIES AND VALUES OF THE EQUATION OF STATE OF AIR.

Property	*Value*	*Unit*
Density ρ	1.225	kg m^{-3}
Specific Heat C_p	434	J kg^{-1} $°C^{-1}$
Adiabatic Exponent γ	1.4	
Reference Temperature T	15.05	$°C^{-1}$
Specific Internal Energy E	2E+5	J kg^{-1}

An additional strength or failure model for air is not needed because air is a gas and it does not suffer stresses and negative pressures.

2) Explosive: At the beginning of the simulation in ANSYS Autodyn a blast wave is created. As explosive charge TNT is used, which is a very common explosive and used in a wide range of applications.

The Jones-Wilkins-Lee (JWL) equation of state describes the detonation product expansion down to a pressure of 1 kbar for high energy explosive materials and has been proposed by

Jones, Wilkins and Lee [20]. It is a function of density, which is expressed as relative volume $\eta = \rho_0/\rho$, and of the internal energy E [19]:

$$P = A\left(1 - \frac{\omega \cdot \eta}{R_1}\right) \cdot \mathrm{e}^{\frac{R_1}{\eta}} + B\left(1 - \frac{\omega \cdot \eta}{R_2}\right) \cdot \mathrm{e}^{\frac{R_2}{\eta}} + \omega \cdot \rho \cdot E \quad (2)$$

The values of the material constants A, B, R_1, R_2 and ω have been determined in dynamic experiments and are listed in Table II.

TABLE II. PROPERTIES AND VALUES OF THE JWL EQUATION OF STATE.

Property	*Value*	*Unit*
Density ρ	1630	kg m^{-3}
Parameter A	3.7377E+11	Pa
Parameter B	3.7471E+9	Pa
Parameter R_1	4.15	
Parameter R_2	0.9	
Parameter ω	0.35	

An important consequence for the blast wave is that the smaller the relative volume, the higher the pressure generated by the explosion. Figure 6 illustrates this behavior as inverse function of density. The individual parts of the JWL equation of state are colored accordingly.

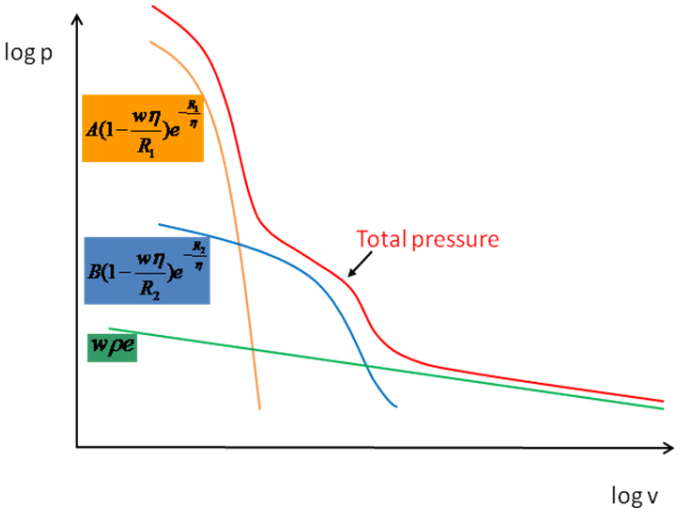


Figure 6. Pressure p as function of relative volume v for the JWL equation of state. Adapted from [21]

3) Steel: Solid materials can react elastically under dynamic loads. If the stress reaches a certain value, the so called yield stress, changes of the shape of an object occur, which are not elastic. If the load continue to increase to a very high dynamic load, the material reach states of stress that exceed their yield strength and plastically deform. This deformation can stay permanent or partly return to the entry state. Material strength laws describe this nonlinear elastic-plastic behavior.

For the modeling of the vehicle in this simulation, the material model of "structural steel" has been used. It consists of several values, which can be found in Table III. It should be noted that this material uses an isotropic elasticity model. In this model the stress-strain behavior equals in every axial direction [19].

TABLE III. PROPERTIES AND VALUES OF THE MATERIAL MODEL OF STRUCTURAL STEEL.

Property	*Value*	*Unit*
Temperature	22	°C
Density ρ	7850	kg m^{-3}
Young's Modulus	2E+11	Pa
Poissons's Ratio	0.3	
Bulk Modulus	1.6667E+12	Pa
Shear Modulus	7.6923E+10	Pa
Specific Heat C_p	434	J kg^{-1} °C^{-1}

The Young's modulus is a material parameter that describes the proportional relationship between stress and strain in the deformation of a solid body in linear-elastic behavior. Another elastic constant is the Poissons's ratio, which is the signed ratio of transverse strain to axial strain. The next material property of Table III is the Bulk modulus. It describes which pressure change is necessary to produce a certain volume change. Finally, the Shear modulus, which provides the information about the linear-elastic deformation of a component due to a shear force or shear stress.

These individual properties form a material model that emulates a material response due to acting forces in the simulation. For this reason, the right choice of material model for a material is critical to the exact result of a simulation.

E. Interface

In general, an interface connects systems that have different properties with the purpose of exchanging information. For computers, this is mainly the case between software, hardware, peripheral devices and humans. Communication at the interface can be either in one direction, such as a remote control or keyboard, or in both directions, such as a touch screen or a network adapter [22].

In the context of numerical simulation of blast and impact processes, an interface is necessary to ensure an effective coupling of CFD / FEM simulations between the software Abaqus and ANSYS. For our research, ANSYS is to be used to provide data from simulated explosions using Euler-Lagrange coupling. On the other hand, the structure, which is affected by the blasting is simulated by Abaqus. The developed Interface has the task of conveying the data between ANSYS and Abaqus, so that the individual simulation steps can be performed successively with respect to the successive transfer of data.

III. Results and Discussion

In computing, an interface is a shared boundary across two separate components of a computer system exchange information. The exchange can be between software, computer hardware, peripheral devices, humans and combinations of these. Some computer hardware devices such as a touchscreen can both send and receive data through the interface, while others such as a mouse, microphone or joystick operate one way only [22].

Coupled FEA/CFD analysis is an alternative technique, where separate FEA and CFD codes are used for solid and fluid regions, respectively, with a smooth exchange of information between the two codes to ensure continuity of blast loading data. The main merit of the approach is to enable users to take full advantages of both CFD and FEA capabilities.

The objective of this work is to develop an interface between ANSYS Autodyn and Abaqus. The software ANSYS is used to solve linear and non-linear problems of structural mechanics, computational fluid dynamics, acoustics and various other engineering sciences [23]. Here, ANSYS will provide data from the simulation of blast effects. The capability to couple Eulerian and Lagrangian frames in ANSYS is helpful in blast field modeling. The Eulerian frame is best suited for representing explosive detonations, because the material flows through a geometrically constant grid that can easily handle the large deformations associated with gas and fluid flow. The structure is modeled with the Lagrangian frame in Abaqus. Abaqus supports familiar interactive computer-aided engineering concepts such as feature-based, parametric modeling, interactive and scripted operation, and GUI customization [24].

First, every possibility of transferring the data from ANSYS outputs to Abaqus inputs has to be detected. A summary of this process is shown in Figure 7. ANSYS will provide the data by generating a data set for the blast loading. Figure 8 shows the color coding of the shock wave goes from low pressures (0 hPa) in blue to high pressures in red ($>$ 3500 hPa). The last picture shows the shock wave when the simulated vehicle is reached. This data set will include snapshots of given points in time. At this stage there is a data set of five points in time, between 0.0291 s and 0.0475 s (after detonation). Related to the points in time this data set includes the pressure values with Cartesian coordinates based on the simulation of the spread of explosive materials. A script is coded to read the blast loading data in Abaqus. This script, coded in Python, uses the line interface in Abaqus directly. First, a blast loading data is generated in ANSYS and saved as a normal text file in .txt format. The data set will be split to separate the different types of information. After that, a list will be created to save the data and connect the related time points to the coordinates and pressure values. At this point, there is a possibility to use linear interpolation between the five time points to generate a larger data base. After reading and saving the data set, the script will load the model used for impact tests in Abaqus. A surface of the model must be selected to project the blast data on it.

The goal is to investigate the impact of the blast data on a full vehicle model in Abaqus. This work (in progress) starts with a less complex model to validate the function of the script and the interface itself. The first model was a basic rectangle to be strained by the pressure data. Afterwards, two more complex models were tested successfully. This approach will lead to a surface similar to the silhouette of high security vehicles (see Figure 9).

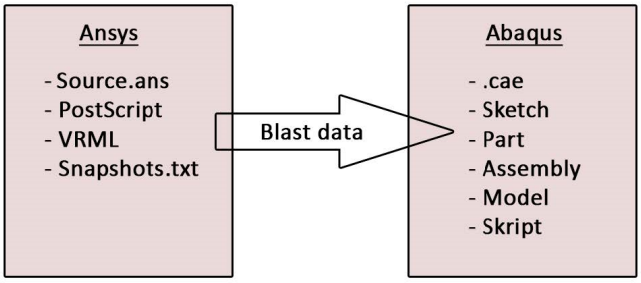


Figure 7. Inputs and outputs for an interface between Ansys and Abaqus. Blast data consisting of several files is exported by Ansys and read via a python script in Abaqus.

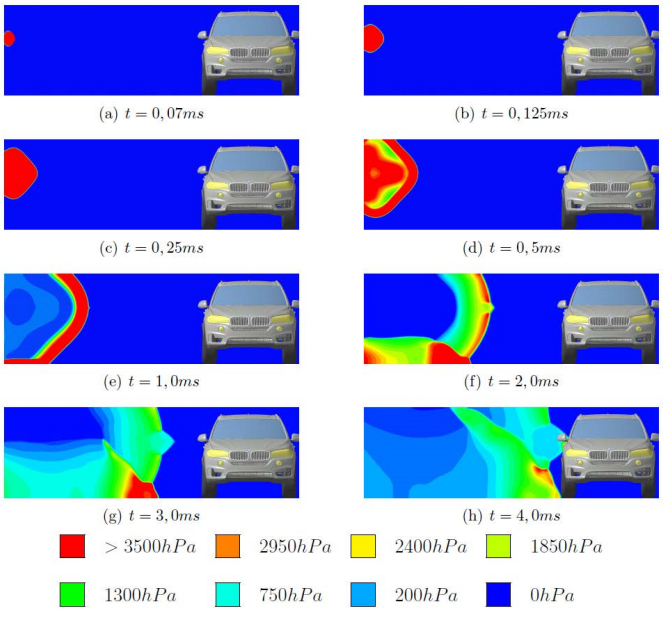


Figure 8. Progressive expansion of a blast in ANSYS Autodyn with a representation for a vehicle on the right side.

The coupling is realized through an iterative loop between the FEA and CFD simulations, with communications ensuring continuity of shock compression data across the coupled boundaries between the FEA and CFD models. In the coupling

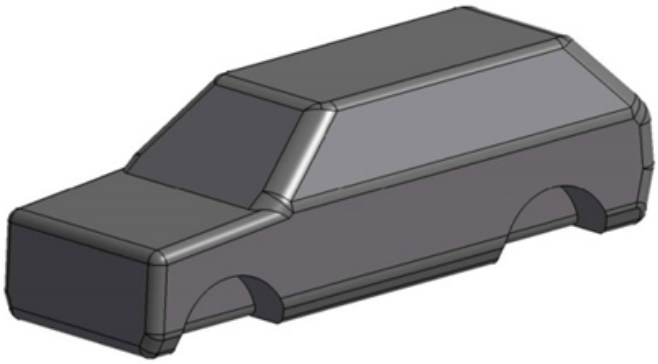


Figure 9. Testing structure in Abaqus with the coarse appearance of a vehicle.

process, intermediate individual FEA and CFD solutions are obtained in turn with dynamically updated boundary conditions.

To avoid exceptional deadlock of the individual CFD simulations, appropriate maximum numbers of iterations are assigned for each CFD model. Testing means that the spatially discretized model is loaded with pressure. The change over time is decisive. An example is shown in Figure 10. The unarmored SUV model was loaded with a typical explosive charge. The load on the vehicle is made visible by color coding from low strain (blue) to very high strain (red). The deformation on the sheet metal body parts is clearly shown. This data can be used to simply analyze vulnerabilities. The goal is, however, to use complete vehicle models and to carry out realistic investigations.

IV. Conclusion and Future Work

A technique for efficiently coupling FEA/CFD for the simulation of blast effects is described. The goal was to combine the strengths of ANSYS in the field of CFD, especially the modeling and simulation of explosive detonations and the subsequent propagation of shock waves, and Abaqus, which is a sophisticated FEA tool an widely used in the automotive area. A newly created interface allows an iterative transfer of the blast data from ANSYS to Abaqus. To enable this, the blast data of a specific simulation step in time and space could be exported. The data sets from ANSYS include snapshots from the blast simulation saved at different points in time. The newly created interface is coded in Python and is able to process the data. One possibility of processing is to use linear interpolation on the data sets. After processing, Abaqus could take the data and insert it in the right positon for solving the FEA. The impact of the blast data on a less complex vehicle model due to the processing of the interface was investigated. In addition, a more complex model was used to the successfully test and validate the interface. The results of this tests are shown in this paper.

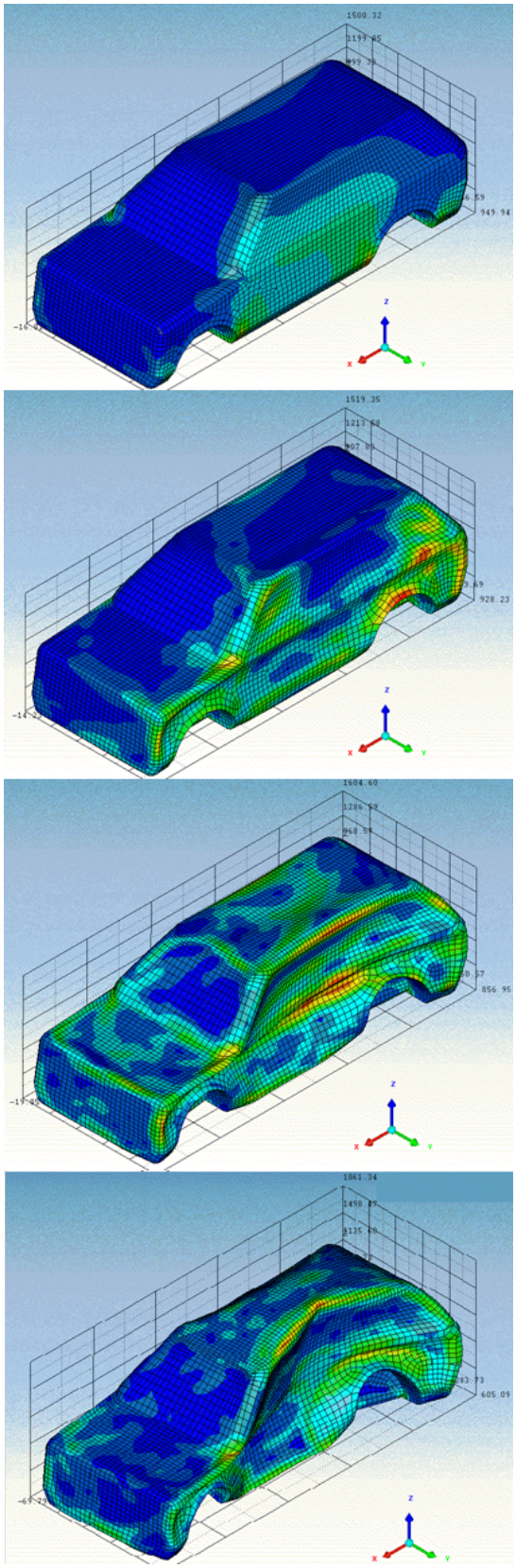


Figure 10. Simulated test structure (unarmed SUV) and deformation process after 3, 5, 10 and 20 ms (images arranged from top to bottom in order of increasing time).

A good agreement of blast load test data and simulation results were observed. Furthermore, it is shown that the coupled solutions can be obtained in sufficiently short turn-around times for use in design. These solutions can be used as the basis of an iterative optimization process. They are a valuable adjunct to the study of the behavior of vehicle structures subjected to high-velocity impact or intense impulsive loading. The combined use of computations, experiments and high-strain-rate material characterization has, in many cases, supplemented the data achievable by experiments alone at considerable savings in both cost and engineering man-hours.

There are a variety of approaches in implementing the coupled FEA/CFD analysis. One is generally called ”strong coupling”, where data have to be transferred between ANSYS Autodyn and Abaqus in every single time step. In a strongly coupled algorithm both parts of the FSI problem are solved simultaneously. For this purpose one system of equations is created after discretising the governing fluid and structural equations and taking into account the boundary conditions on the interface. A ”semistrong coupling” can get along with a smaller set of date, using mathematical interpolation for a sufficient approximation. The third concept is a ”weak coupling” solution. Here, neural networks and deep learning can be used to replicate blast effects on different vehicle structures. These approaches are going to be tested in a next step. Since the fluid and the structural dynamics subproblems are based on different types of partial differential equations, the use of different numerical methods for their efficient solutions might be advantageous.

The main advantage of the loose coupling approach is that it allows already developed efficient and well validated solvers for each of the fluid and structure subtasks to be combined. Therefore, both parts of the FSI problem are solved in the best ways. Depending on the generality of the two codes, arbitrary complex flows and structures can be considered and successfully modelled. The only programming effort lies in creating suitable subroutines for information exchange between the solvers. Unfortunately, due to the explicit nature of this coupling convergence problems may arise. Consequently, there is a restriction on the choice of the time-step even if implicit time-stepping schemes are used by the two solvers. In contrast to the loose coupling approach, the strong coupling algorithms are more difficult to create and to program. The simultaneous solution of the whole FSI problem normally requires reformulation of the systems of equations and sets restrictions on the choice of the numerical methods to be applied. Additionally, special strategies may be needed for modelling the non-linearities in each of the physical domains. This leads to a restriction on the range of tasks that a certain strong coupling algorithm is able to solve. Tremendous programming efforts are needed to create and to validate a new program applicable to various problems. However, because of the simultaneous solution of both parts of the FSI problem, there are no approximation errors and no convergence problems due to the data transfer between the fluid and structural domains. Therefore, the strong coupling strategy is more stable but more difficult to program than the loose coupling approach that is more general but connected with convergence problems.

Furthermore, a larger blast loading data set has to be created in ANSYS. This will allow a more accurate illustration of blast effects on vehicle structures. Smaller time steps will enable a linear interpolation with a higher accuracy. Different explosives are going to be tested to expand the data base. The next step will be a model for the reflection of blast waves and dynamic changes of pressure values. Using a full vehicle model will provide important information about the behavior of armored structures under blast effects. But to validate the results of the simulation, more ballistic trials are needed. Based on the difficulties of full vehicle model simulations, the implementation of an automatic surface detection has to be taken into consideration. This could be helpful if a large number of different vehicles are investigated. In order to create a user-friendly interface it is possible to generate the script as a plug-in which can be started from the Abaqus user surface directly.

By using pre-defined blast data to create forces as vectors on our vehicle structures, the proposal can be generalized. Then, FEA analysis can be done with other software suites as well. Right now, the concept is not applicable to other systems. This is a major disadvantage and part of our future work. Furthermore, a parallelization of the problem should be considered.

REFERENCES

[1] A. Ramezani, B. Hillig, and H. Rothe, “A Coupled CFD-FEM Analysis to Simulate Blast Effects on High Security Vehicles Using Modern Hydrocodes,” in SIMUL 2017 : The Ninth International Conference on Advances in System Simulation (SIMUL 2017) Athens, Greece. IARIA, Oct. 2017, pp. 74–81, ISBN: 978-1-61208-594-4.

[2] J. D. Anderson and J. Wendt, Computational fluid dynamics. Springer, 1995, vol. 206.

[3] O. Ziekiewicz, “The finite element method in engineering science,” McGrawe Hill, London, 1971.

[4] A. Documentation, “Abaqus analysis user guide,” 2014.

[5] “Blast, Explosion and Impact,” 2018, URL: https://www.ansys.com/solutions/solutions-by-industry/construction/blast-explosion-and-impact [accessed: 2018-05-28].

[6] E. Hansen, N. Ehlers, A. Ramezani, and H. Rothe, “Developing an Interface between ANSYS and Abaqus to Simulate Blast Effects on High Security Vehicles,” in SIMUL 2016 : The Eighth International Conference on Advances in System Simulation (SIMUL 2016). IARIA, Aug. 2016, pp. 73–76, ISBN: 978-1-61208-442-8.

[7] A. Ramezani and H. Rothe, “Investigation of Solver Technologies for the Simulation of Brittle Materials,” in SIMUL 2014 : The Sixth International Conference on Advances in System Simulation (SIMUL 2014). IARIA, Oct. 2014, pp. 236–242, ISBN: 978-61208-371-1.

[8] J. Zukas, Introduction to hydrocodes. Elsevier, 2004, vol. 49.

[9] A. Hamouda and M. Hashmi, “Modelling the impact and penetration events of modern engineering materials: characteristics of computer codes and material models,” Journal of materials processing technology, vol. 56, no. 1-4, 1996, pp. 847–862.

[10] D. J. Benson, "Computational methods in lagrangian and eulerian hydrocodes," Computer methods in Applied mechanics and Engineering, vol. 99, no. 2-3, 1992, pp. 235–394.

[11] M. Oevermann, S. Gerber, and F. Behrendt, "Euler–lagrange/dem simulation of wood gasification in a bubbling fluidized bed reactor," Particuology, vol. 7, no. 4, 2009, pp. 307–316.

[12] D. Hicks and L. Liebrock, "Sph hydrocodes can be stabilized with shape-shifting," Computers & Mathematics with Applications, vol. 38, no. 5, 1999, pp. 1–16.

[13] X. Quan, N. Birnbaum, M. Cowler, B. Gerber, R. Clegg, and C. Hayhurst, "Numerical simulation of structural deformation under shock and impact loads using a coupled multi-solver approach," in Proceedings of 5th Asia-Pacific Conference on Shock and Impact Loads on Structures, 2003, pp. 12–14.

[14] N. V. Bermeo, M. G. Mendoza, and A. G. Castro, "Semantic representation of cad models based on the iges standard," in Mexican International Conference on Artificial Intelligence. Springer, 2013, pp. 157–168.

[15] J. Sarkar, Computer Aided Design: A Conceptual Approach. CRC Press, 2014.

[16] C. E. Anderson Jr, "An overview of the theory of hydrocodes," International journal of impact engineering, vol. 5, no. 1-4, 1987, pp. 33–59.

[17] G. S. Collins, "An introduction to hydrocode modeling," Applied Modelling and Computation Group, Imperial College London, unpublished, 2002.

[18] I. Sochet, Blast Effects: Physical Properties of Shock Waves. Springer, 2017.

[19] A. AUTODYN, "Autodyn user's manual, release 18.0," 2017.

[20] J. Kury, H. Hornig, E. Lee, J. McDonnel, D. Ornellas, M. Finger, F. Strange, and M. Wilkins, "Metal acceleration by chemical explosives," in Fourth (International) Symposium on Detonation, ACR-126, 1965.

[21] "ANSYS Mechanical Application 19.0: Explicit Dynamics Analysis Guide," 2018, URL: http://www.ansys.com/ [accessed: 2018-02-27].

[22] K. Breitfelder and D. Messina, "The authoritative dictionary of ieee standards terms," Institute of Electrical and Electronics Engineers (IEEE), 2000.

[23] "ANSYS CAE: Structures: FEA Simulation," 2017, URL: http://www.ansys.com/ [accessed: 2017-05-01].

[24] "Abaqus: Complete Solutions for Realistic Simulation," 2017, URL: http://www.3ds.com/products-services/simulia/products/abaqus/abaquscae/ [accessed: 2017-06-01].

Advanced Sound Static Analysis to Detect Safety- and Security-Relevant Programming Defects

Daniel Kästner, Laurent Mauborgne, Nicolas Grafe, Christian Ferdinand

AbsInt GmbH
Science Park 1, 66123 Saarbrücken, Germany
Email: kaestner@absint.com, mauborgne@absint.com, grafe@absint.com, ferdinand@absint.com

***Abstract*—Static code analysis has evolved to be a standard technique in the development process of safety-critical software. It can be applied to show compliance to coding guidelines, and to demonstrate the absence of critical programming errors, including runtime errors and data races. In recent years, security concerns have become more and more relevant for safety-critical systems, not least due to the increasing importance of highly-automated driving and pervasive connectivity. While in the past, sound static analyzers have been primarily applied to demonstrate classical safety properties they are well suited also to address data safety, and to discover security vulnerabilities. This article gives an overview and discusses practical experience.**

Keywords–static analysis; abstract interpretation; runtime errors; security vulnerabilities; functional safety; cybersecurity.

I. INTRODUCTION

Some years ago, static analysis meant manual review of programs. Nowadays, automatic static analysis tools are gaining popularity in software development as they offer a tremendous increase in productivity by automatically checking the code under a wide range of criteria [1]. Many software development projects are developed according to coding guidelines, such as MISRA C [2], SEI CERT C [3], or CWE (Common Weakness Enumeration) [4], aiming at a programming style that improves clarity and reduces the risk of introducing bugs. Compliance checking by static analysis tools has become common practice.

In safety-critical systems, static analysis plays a particularly important role. A failure of a safety-critical system may cause high costs or even endanger human beings. With the growing size of software-implemented functionality, preventing software-induced system failures becomes an increasingly important task. One particularly dangerous class of errors are runtime errors, which include faulty pointer manipulations, numerical errors such as arithmetic overflows and division by zero, data races, and synchronization errors in concurrent software. Such errors can cause software crashes, invalidate separation mechanisms in mixed-criticality software, and are a frequent cause of errors in concurrent and multi-core applications. At the same time, these defects are also at the root of many security vulnerabilities, including exploits based on buffer overflows, dangling pointers, or integer errors.

In safety-critical software projects, obeying coding guidelines such as MISRA C is strongly recommended by all current safety standards, including DO-178C [5], IEC-61508 [6], ISO-26262 [7], and EN-50128 [8]. In addition, all of them consider demonstrating the absence of runtime errors explicitly as a verification goal. This is often formulated indirectly by addressing runtime errors (e.g., division by zero, invalid pointer accesses, arithmetic overflows) in general, and additionally considering corruption of content, synchronization mechanisms, and freedom of interference in concurrent execution. Semantics-based static analysis has become the predominant technology to detect runtime errors and data races.

Abstract interpretation is a formal methodology for semantics-based static program analysis [9]. It supports formal soundness proofs (it can be proven that no error is missed) and scales to real-life industry applications. Abstract interpretation-based static analyzers provide full control and data coverage and allow conclusions to be drawn that are valid for all program runs with all inputs. Such conclusions may be that no timing or space constraints are violated, or that runtime errors or data races are absent: the absence of these errors can be guaranteed [10]. Nowadays, abstract interpretation-based static analyzers that can detect stack overflows and violations of timing constraints [11] and that can prove the absence of runtime errors and data races [12][13], are widely used for developing and verifying safety-critical software. From a methodological point of view, abstract interpretation-based static analyses can be seen as equivalent to testing with full data and control coverage. They do not require access to the physical target hardware, can be easily integrated in continuous verification frameworks and model-based development environments [14], and they allow developers to detect runtime errors as well as timing and space bugs in early product stages.

In the past, security properties have mostly been relevant for non-embedded and/or non-safety-critical programs. Recently due to increasing connectivity requirements (cloud-based services, car-to-car communication, over-the-air updates, etc.), more and more security issues are rising in safety-critical software as well. Security exploits like the Jeep Cherokee hacks [15], which affect the safety of the system, are becoming more and more frequent. In consequence, safety-critical software development faces novel challenges, which previously only have been addressed in other industry domains.

On the other hand, as outlined above, safety-critical software is developed according to strict guidelines, which effectively reduce the relevant subset of the programming language used and improve software verifiability. As an example dynamic memory allocation and recursion often are forbidden or used in a very limited way. In consequence, for safety-critical software much stronger code properties can be shown than for non-safety-critical software, so that also security vulnerabilities can be addressed in a more powerful way.

The topic of this article is to show that some classes of defects can be proven to be absent in the software so that exploits based on such defects can be excluded. On the other hand, additional syntactic checks and semantical analyses become necessary to address security properties that are orthogonal to safety requirements. Throughout the article we will focus on software aspects only, without addressing safety or security properties at the hardware level. While we

focus on the programming language C, the basic analysis techniques described in this article are applicable to other programming languages as well.

The article is based on [1], amplifying some of the key aspects and following up on ongoing work. It is structured as follows: Section II discusses the relation between *safety* and *security* requirements. After a brief summary of typical requirements and verification goals formulated in current safety standards the role of coding standards is discussed in Section II-B. A classification of security vulnerabilities is given in Section II-C, and Section II-D focuses on the analysis complexity of safety and security properties. Section III gives an overview of abstract interpretation and its application to runtime error analysis, using the sound analyzer Astrée as an example. Section IV gives an overview of control and data flow analysis with emphasis on two advanced analysis techniques: program slicing (cf. Section IV-A) and taint analysis (cf. Section IV-B). Section V concludes.

II. Security in Safety-Critical Systems

Functional safety and security are aspects of dependability, in addition to reliability and availability. *Functional safety* is usually defined as the absence of unreasonable risk to life and property caused by malfunctioning behavior of the software. The main goals of *information security* or *cybersecurity* (for brevity denoted as "*security*" in this article) traditionally are to preserve *confidentiality* (information must not be disclosed to unauthorized entities), *integrity* (data must not be modified in an unauthorized or undetected way), and *availability* (data must be accessible and usable upon demand).

In safety-critical systems, safety and security properties are intertwined. A violation of security properties can endanger the functional safety of the system: an information leak could provide the basis for a successful attack on the system, and a malicious data corruption or denial-of-service attack may cause the system to malfunction. Vice versa, a violation of safety goals can compromise security: buffer overflows belong to the class of critical runtime errors whose absence have to be demonstrated in safety-critical systems. At the same time, an undetected buffer overflow is one of the main security vulnerabilities, which can be exploited to read unauthorized information, to inject code, or to cause the system to crash [16]. To emphasize this, in a safety-critical system the definition of functional safety can be adapted to define cybersecurity as absence of unreasonable risk to life and property caused by malicious misuse of the software.

The convergence of safety and security properties also becomes apparent in the increasing role of data in safety-critical systems. There are many well-documented incidents where harm was caused by erroneous data, corrupted data, or inappropriate use of data – examples include the Turkish Airlines A330 incident (2015), the Mars Climate Orbiter crash (1999), or the Cedars Sinai Medical Centre CT scanner radiation overdose (2009) [17]. The reliance on data in safety-critical systems has significantly grown in the past few years, cf. e.g., data used for decision-support systems, data used in sensor fusion for highly automatic driving, or data provided by car-to-car communication or downloaded from a cloud. As a consequence of this there are ongoing activities to provide specific guidance for handling data in safety-critical systems [17]. At the same time, these data also represent safety-relevant targets for security attacks.

In this section we will first give an overview of typical verification goals and requirements of contemporary safety norms, followed by a brief discussion of relevant coding guidelines. With this background we will present a classification of security vulnerabilities and discuss their relationship with respect to safety requirements. The section concludes with a discussion of algorithmic complexity of safety and security requirements.

A. The Safety Standards' Perspective

Safety standards like ISO-26262 [7], DO-178B [18], DO-178C [5], IEC-61508 [6], and EN-50128 [8] require to identify functional and non-functional hazards and to demonstrate that the software does not violate the relevant safety goals. Some non-functional safety hazards can be critical for the correct functioning of the system: violations of timing constraints in real-time software and software defects like runtime errors or stack overflows. Depending on the criticality level of the software the absence of safety hazards has to be demonstrated by formal methods or testing with sufficient coverage. In this section we will focus on the non-functional aspects at the programming language level, and use the avionics standard DO178C and the automotive standard ISO-26262 as examples.

1) DO-178C: Published in 2011, the DO-178C [5] ("Software Considerations in Airborne Systems and Equipment Certification"), is the primary document by which certification authorities such as EASA or FAA approve commercial software-based aerospace systems. In general, the DO-178C aims at providing "guidance for determining, in a consistent manner and with an acceptable level of confidence, that the software aspects of airborne systems and equipment comply with airworthiness requirements." The *Formal Methods Supplement* DO-333 [19] gives an overview of formal methods, such as Abstract Interpretation (cf. Section III-A), and their application in the software development and verification process.

In the software development process, *Software Design Standards* and *Software Code Standards* have to be taken into account to "disallow the use of constructs or methods that produce outputs that cannot be verified or that are not compatible with safety-related requirements". The *Software Design Standards* (cf. Section 11.7 of [5]) are defined to focus on algorithmic constraints like exclusion of recursion, dynamic objects, or data aliases. They should also include complexity restrictions like maximum level of nested calls. The *Software Code Standards* focus on the programming language. They identify the programming language to be used and should define a safety-oriented language subset (cf. Section 11.8a of [5]). Coding guidelines enforcing compliance with the *Software Design Standard* and the *Software Code Standards* have to be applied.

One objective of the software verification activities is the *accuracy and consistency* verification goal, which aims at determining "the correctness and consistency of the source code, including stack usage, memory usage, fixed point arithmetic overflow and resolution, [...], worst-case execution timing, exception handling, use of uninitialized variables, [...] and data corruption due to task or interrupt conflicts". In particular, this includes runtime errors caused by undefined or unspecified behavior of the programming language. Runtime errors also

have to be addressed when verifying the software component integration, implying, e.g., detecting incorrect initialization of variables, parameter passing errors, and data corruption. All these requirements can be checked using formal analysis (cf. Section FM.6.3.4 of [19]). Furthermore, data and control flow analysis is required at the software architectural level and the source code level to ensure implementation consistency.

2) ISO-26262: In this section we focus on Part 6 of the ISO-26262 standard [20], which specifies the requirements for product development at the software level for automotive applications.

Supporting real-time software and runtime error handling is considered a basic requirement for selecting a suitable modeling or programming language (cf. Section 5.4.6 of [20]). It also states that "criteria that are not sufficiently addressed by the language itself shall be covered by the corresponding guidelines or by the development environment" (cf. Section 5.4.7 of [20]). Coding guidelines have to be applied to support the correctness of the design and implementation. For the programming language C the MISRA C and MISRA AC AGC standards are suggested. The goals to be addressed include "the use of language subsets to exclude language constructs which could result in unhandled runtime errors". The absence of runtime errors has to be ensured by appropriate tools as a part of the development environment.

Among the verification goals of the software design and implementation stage are correctness of data flow and control flow between and within the software units, consistency of the interfaces, and robustness. The standard lists some examples for robustness properties, including implausible values, execution errors, division by zero, and errors in the data and control flow. Furthermore, the compatibility of the software unit implementation with the target hardware has to be ensured.

The software integration phase considers functional dependencies and the dependencies between software integration and hardware-software integration, including non-functional software properties. Again robustness properties have to be taken into account, and it has to be ensured that sufficient resources are available.

The methods for verification of software unit and design and the methods for software verification of software integration include static analysis in general and abstract interpretation (cf. [21]).

B. Coding Guidelines

The MISRA C standard [2] has originally been developed with a focus on automotive industry but is now widely recognized as a coding guideline for safety-critical systems in general. Its aim is to avoid programming errors and enforce a programming style that enables the safest possible use of C. A particular focus is on dealing with undefined/unspecified behavior of C and on preventing runtime errors. As a consequence, it is also directly applicable to security-relevant code.

The most prominent coding guidelines targeting security aspects are the ISO/IEC TS 17961 [22], the SEI CERT C Coding Standard [3], and the MITRE Common Weakness Enumeration CWE [4].

The ISO/IEC TS 17961 C Secure Coding Rules [22] specifies rules for secure coding in C. It does not primarily address developers but rather aims at establishing requirements for compilers and static analyzers. MISRA C:2012 Addendum 2 [23] compares the ISO/IEC TS 17961 rule set with MISRA C:2012. Only 4 of the C Secure rules are not covered by the first edition of MISRA C:2012 [2], however, with Amendment 1 to MISRA C:2012 [24] all of them are covered as well. This illustrates the strong overlap between the safety- and security-oriented coding guidelines.

The SEI CERT C Coding Standard belongs to the CERT Secure Coding Standards [25]. While emphasizing the security aspect CERT C [3] also targets safety-critical systems: it aims at "developing safe, reliable and secure systems". CERT distinguishes between rules and recommendations where rules are meant to provide normative requirements and recommendations are meant to provide general guidance; the book version [3] describes the rules only. A particular focus is on eliminating undefined behaviors that can lead to exploitable vulnerabilities. In fact, almost half of the CERT rules (43 of 99 rules) are targeting undefined behaviors according to the C norm. Addendum 3 to MISRA C:2012 [26] provides an in-depth analysis of the coverage of MISRA C:2012 against CERT C [3].

The Common Weakness Enumeration CWE is a software community project [4] that aims at creating a catalog of software weaknesses and vulnerabilities. The goal of the project is to better understand flaws in software and to create automated tools that can be used to identify, fix, and prevent those flaws. There are several catalogues for different programming languages, including C. In the latter one, once again, many rules are associated with undefined or unspecified behaviors.

C. Vulnerability Classification

Many rules are shared between the different coding guidelines, but there is no common structuring of security vulnerabilities. The CERT Secure C roughly structures its rules according to language elements, whereas ISO/IEC TS 17961 and CWE are structured as a flat list of vulnerabilities. In the following we list some of the most prominent vulnerabilities, which are addressed in all coding guidelines and which belong to the most critical ones at the C programming level. The presentation follows the overview given in [16].

1) Stack-based Buffer Overflows: An array declared as local variable in C is stored on the runtime stack. A C program may write beyond the end of the array due to index values being too large or negative, or due to invalid increments of pointers pointing into the array. A runtime error then has occurred whose behavior is undefined according to the C semantics. As a consequence the program might crash with bus error or segmentation fault, but typically adjacent memory regions will be overwritten. An attacker can exploit this by manipulating the return address or the frame pointer both of which are stored on the stack, or by indirect pointer overwriting, and thereby gaining control over the execution flow of the program. In the first case the program will jump to code injected by the attacker into the overwritten buffer instead of executing an intended function return. In case of overflows on array read accesses confidential information stored on the stack (e.g., through temporary local variables) might be leaked. An example of such an exploit is the well-known W32.Blaster.Worm [27].

In the following we will illustrate this vulnerability and its implications for safety and security with a small toy example.

Consider the following C code fragment:

```
struct {
  char buf[4096];
  unsigned int len;
} msg;

int f1(void) {
  char LBuf [1024];
  unsigned int i;
  GetMsg();
  for (i=0; i<msg.len; i++)
    LBuf[i]=msg.buf[i];
}

void f0(void) {
  ...
  f1(a,b);
  ...
}
```

Function `f0` calls function `f1`, which has two local variables, a `char` array `LBuf` of size 1024 bytes, and an integer `i`. Let's assume the function `GetMsg` updates the global variable `msg` from the environment or a user input. The code expects `msg` to contain a valid string in field `buf`, and the length of this string in field `len`.

There is no explicit check that the value of `len` is not greater than 1024. If `len>1024` a buffer overwrite will occur, and contents of the runtime stack will be overwritten.

A possible stack layout before executing the loop in function `f1` is shown in Figure 1. We assume that the stack grows downwards. The stack frame of `f0` starts with its return address, the previous value of the frame pointer, space for local variables and possibly spilled registers. It is followed by the stack frame of `f1`. Its return address is the instruction following the call instruction in the code of `f0` from which `f1` was invoked. The next cell points to the previous frame pointer, then there are 1024 bytes for the local array `LBuf`, and 4 bytes for the local variable `i`.

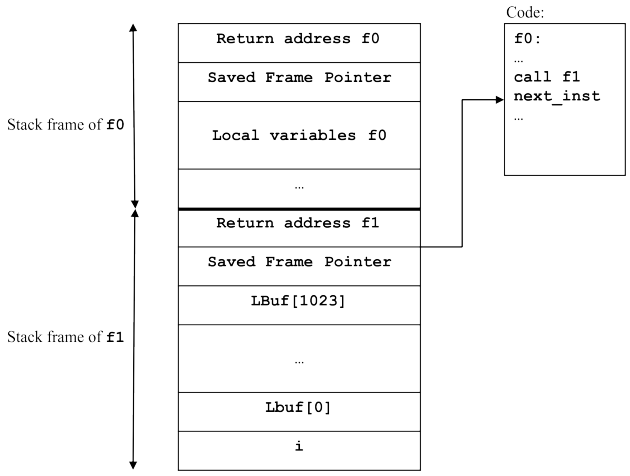


Figure 1. Stack frame before start of loop in `f1`

The loop in `f1` overwrites all elements of the `LBuf` array, starting with index 0. When a buffer overflow happens, the overwriting does not stop in the cell of `LBuf[1023]`, instead the entries of the stack above it will be overwritten, starting with the previous frame pointer, the return address, etc. The expected consequence will be that the program will behave erratically or crash, when trying to continue execution at a wrong (overwritten) return address.

When the situation is exploited in a malicious way, the return address is intentionally overwritten with a carefully selected address. The stack frame of such a code injection attack is shown in Figure 2.

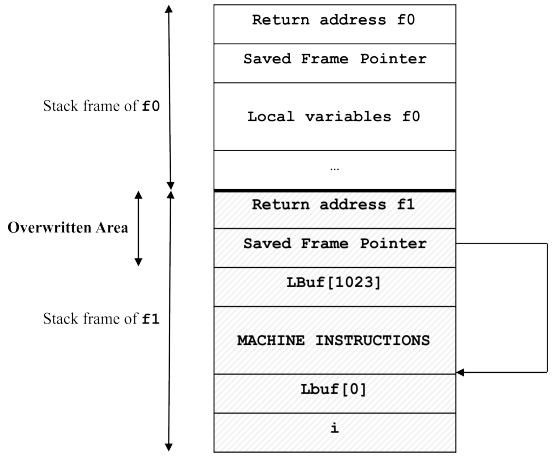


Figure 2. Stack frame after code injection attack

The assumption in that case is that the attacker has gained control to influence the value of the fields in `msg`. The first step is to fill a part of the entries of the source buffer with bytes representing feasible machine code. The `len` parameter is chosen such that the two stack cells above `LBuf` are overwritten. In the second step, the attacker makes sure that the entries in the `msg.buf` array, which will be written in the return address field of the stack, contain the address within `LBuf` where the injected code begins. Then function `f1` will continue normally, but instead of returning to its caller, it will execute the injected code.

2) Heap-based Buffer Overflows: Heap memory is dynamically allocated at runtime, e.g., by calling $malloc()$ or $calloc()$ implementations provided by dynamic memory allocation libraries. There may be read or write operations to dynamically allocated arrays that access beyond the array boundaries, similarly to stack-allocated arrays. In case of a read access information stored on the heap might be leaked – a prominent example is the Heartbleed bug in OpenSSL (cf. CERT vulnerability 720951 [28]). Via write operations attackers may inject code and gain control over program execution, e.g., by overwriting management information of the dynamic memory allocator stored in the accessed memory chunk.

3) General Invalid Pointer Accesses: Buffer overflows are special cases of invalid pointer accesses, which are listed here as separate points due to the large number of attacks based on them. However, any invalid pointer access in general is a security vulnerability – other examples are null pointer accesses or dangling pointer accesses. Accessing such a pointer is undefined behavior, which can cause the program to crash, or behave erratically. A dangling pointer points to a memory

location that has been deallocated either implicitly (e.g., data stored in the stack frame of a function after its return) or explicitly by the programmer. A concrete example of a dangling pointer access is the double free vulnerability where already freed memory is freed a second time. This can be exploited by an attacker to overwrite arbitrary memory locations and execute injected code [16].

4) Uninitialized Memory Accesses: Automatic variables and dynamically allocated memory have indeterminate values when not explicitly initialized. Accessing them is undefined behavior. Uninitialized variables can also be used for security attacks, e.g., in CVE-2009-1888 [29] potentially uninitialized variables passed to a function were exploited to bypass the access control list and gain access to protected files [3].

5) Integer Errors: Integer errors are not exploitable vulnerabilities by themselves, but they can be the cause of critical vulnerabilities like stack- or heap-based buffer overflows. Examples of integer errors are arithmetic overflows, or invalid cast operations. If, e.g., a negative signed value is used as an argument to a $memcpy()$ call, it will be interpreted as a large unsigned value, potentially resulting in a buffer overflow.

6) Format String Vulnerabilities : A format string is copied to the output stream with occurrences of %-commands representing arguments to be popped from the stack and expanded into the stream. A format string vulnerability occurs, if attackers can supply the format string because it enables them to manipulate the stack, once again making the program write to arbitrary memory locations.

7) Concurrency Defects: Concurrency errors may lead to concurrency attacks, which allow attackers to violate confidentiality, integrity and availability of systems [30]. In a race condition the program behavior depends on the timing of thread executions. A special case is a write-write or read-write data race where the same shared variable is accessed by concurrent threads without proper synchronization. In a Time-of-Check-to-Time-of-Use (TOCTOU) race condition the checking of a condition and its use are not protected by a critical section. This can be exploited by an attacker, e.g., by changing the file handle between the accessibility check and the actual file access. In general, attacks can be run either by creating a data race due to missing lock-/unlock protections, or by exploiting existing data races, e.g., by triggering thread invocations.

Most of the vulnerabilities described above are based on undefined behaviors, and among them buffer overflows seem to play the most prominent role for real-live attacks. Most of them can be used for denial-of-service attacks by crashing the program or causing erroneous behavior. They can also be exploited to inject code and cause the program to execute it, and to extract confidential data from the system. It is worth noticing that from the perspective of a static analyzer most exploits are based on potential runtime errors: when using an unchecked value as an index in an array the error will only occur if the attacker manages to provide an invalid index value. The obvious conclusion is that safely eliminating all potential runtime errors due to undefined behaviors in the program significantly reduces the risk for security vulnerabilities.

D. Analysis Complexity

While semantics-based static program analysis is widely used for safety properties, there is practically no such analyzer dedicated to specific security properties. This is mostly explained by the difference in complexity between safety and security properties. From a semantical point of view, a safety property can always be expressed as a trace property. This means that to find all safety issues, it is enough to look at each trace of execution in isolation.

This is not possible any more for security properties. Most of them can only be expressed as set of traces properties, or hyperproperties [31]. A typical example is non-interference [32]: to express that the final value of a variable x can only be affected by the initial value of y and no other variable, one must consider each pair of possible execution traces with the same initial value for y, and check that the final value of x is the same for both executions. It was proven in [31] that any other definition (tracking assignments, etc.) considering only one execution trace at a time would miss some cases or add false dependencies. This additional level of sets has direct consequences on the difficulty to track security properties soundly.

Other examples of hyperproperties are secure information flow policies, service level agreements (which describe acceptable availability of resources in term of mean response time or percentage uptime), observational determinism (whether a system appears deterministic to a low-level user), or quantitative information flow.

Finding expressive and efficient abstractions for such properties is a young research field (see [33]), which is the reason why no sound analysis of such properties appear in industrial static analyzers yet. The best solution using the current state of the art consists of using dedicated safety properties as an approximation of the security property in question, such as the taint propagation described in Section IV-B.

III. Proving the Absence of Defects

In safety-critical systems, the use of dynamic memory allocation and recursions typically is forbidden or only used in limited ways. This simplifies the task of static analysis such that for safety-critical embedded systems it is possible to formally prove the absence of runtime errors, or report all potential runtime errors which still exist in the program. Such analyzers are based on the theory of abstract interpretation [9], a mathematically rigorous formalism providing a semantics-based methodology for static program analysis.

A. Abstract Interpretation

The semantics of a programming language is a formal description of the behavior of programs. The most precise semantics is the so-called concrete semantics, describing closely the actual execution of the program on all possible inputs. Yet in general, the concrete semantics is not computable. Even under the assumption that the program terminates, it is too detailed to allow for efficient computations. The solution is to introduce an abstract semantics that approximates the concrete semantics of the program and is efficiently computable. This abstract semantics can be chosen as the basis for a static analysis. Compared to an analysis of the concrete semantics, the analysis result may be less precise but the computation may be significantly faster.

A static analyzer is called *sound* if the computed results hold for any possible program execution. Abstract interpretation supports formal correctness proofs: it can be proved that

an analysis will terminate and that it is sound, i.e., that it computes an over-approximation of the concrete semantics. Imprecision can occur, but it can be shown that they will always occur on the safe side. In runtime error analysis, soundness means that the analyzer never omits to signal an error that can appear in some execution environment. If no potential error is signaled, definitely no runtime error can occur: there are no false negatives. If a potential error is reported, the analyzer cannot exclude that there is a concrete program execution triggering the error. If there is no such execution, this is a false alarm (false positive). This imprecision is on the safe side: it can never happen that there is a runtime error which is not reported.

The difference between syntactical, unsound semantical and sound semantical analysis can be illustrated at the example of division by 0. In the expression $x/0$ the division by zero can be detected syntactically, but not in the expression a/b. When an unsound analyzer does not report a division by zero in a/b it might still happen in scenarios not taken into account by the analyzer. When a sound analyzer does not report a division by zero in a/b, this is a proof that b can never be 0.

B. Astrée

In the following we will concentrate on the sound static runtime error analyzer Astrée [13][34]. It reports program defects caused by unspecified and undefined behaviors according to the C norm (ISO/IEC 9899:1999 (E)), program defects caused by invalid concurrent behavior, violations of user-specified programming guidelines, and computes program properties relevant for functional safety. Users are notified about: integer/floating-point division by zero, out-of-bounds array indexing, erroneous pointer manipulation and dereferencing (buffer overflows, null pointer dereferencing, dangling pointers, etc.), data races, lock/unlock problems, deadlocks, integer and floating-point arithmetic overflows, read accesses to uninitialized variables, unreachable code, non-terminating loops, violations of optional user-defined static assertions, violations of coding rules (MISRA C, ISO/IEC TS 17961, CERT, CWE) and code metric thresholds. In particular, this includes all error categories mentioned in Section II-C as principle security vulnerabilities.

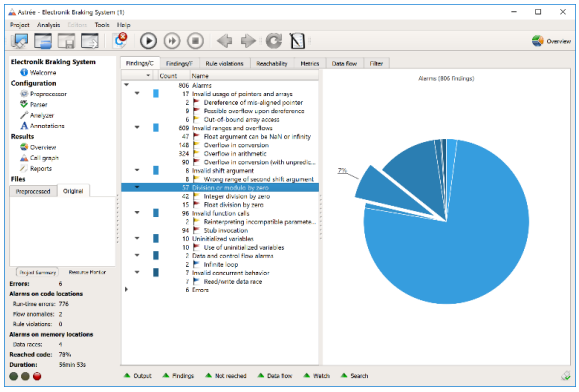


Figure 3. Astrée GUI with alarm overview

Astrée computes data and control flow reports containing a detailed listing of accesses to global and static variables sorted by functions, variables, and processes and containing a summary of caller/called relationships between functions. The analyzer can also report each effectively shared variable, the list of processes accessing it, and the types of the accesses (read, write, read/write).

The C99 standard does not fully specify data type sizes, endianness nor alignment, which can vary with different targets or compilers. Astrée is informed about these target ABI settings by a dedicated configuration file in XML format and takes the specified properties into account.

The design of the analyzer aims at reaching the zero false alarm objective, which was accomplished for the first time on large industrial applications at the end of November 2003. For keeping the initial number of false alarms low, a high analysis precision is mandatory. To achieve high precision Astrée provides a variety of predefined abstract domains, e.g., the interval domain approximates variable values by intervals, the octagon domain [35] covers relations of the form $x \pm y \leq c$ for variables x and y and constants c. The memory domain empowers Astrée to exactly analyze pointer arithmetic and union manipulations. It also supports a type-safe analysis of absolute memory addresses. With the filter domain digital filters can be precisely approximated. Floating-point computations are precisely modeled while keeping track of possible rounding errors.

To deal with concurrency defects, Astrée implements a sound low-level concurrent semantics [36], which provides a scalable sound abstraction covering all possible thread interleavings. The interleaving semantics enables Astrée, in addition to the classes of runtime errors found in sequential programs, to report data races, and lock/unlock problems, i.e., inconsistent synchronization. The set of shared variables does not need to be specified by the user: Astrée assumes that every global variable can be shared, and discovers which ones are effectively shared, and on which ones there is a data race. After a data race, the analysis continues by considering the values stemming from all interleavings. Since Astrée is aware of all locks held for every program point in each concurrent thread, Astrée can also report all potential deadlocks.

In some situations data races may be intended behavior. As an example a lock-free implementation where one process only writes to a variable and another process only reads from it may be correct, although there actually is a data race. However, a prerequisite is that all variable accesses involved are atomic. Astrée explicitly supports such lock-free implementations by providing means to specify the atomicity of basic data type accesses as a part of the target ABI specification. Data race alarms explicitly distinguish between atomic and non-atomic accesses.

Thread priorities are exploited to reduce the amount of spurious interleavings considered in the abstraction and to achieve a more precise analysis. A dedicated task priority domain supports dynamic priorities, e.g., according to the Priority Ceiling Protocol used in OSEK systems [37]. Astrée includes a built-in notion of mutual exclusion locks, on top of which actual synchronization mechanisms offered by operating systems can be modeled (such as POSIX mutexes or semaphores).

Programs to be analyzed are seldom run in isolation; they interact with an environment. In order to soundly report all runtime errors, Astrée must take the effect of the environment

into account. In the simplest case the software runs directly on the hardware, in which case the environment is limited to a set of volatile variables, i.e., program variables that can be modified by the environment concurrently, and for which a range can be provided to Astrée by formal directives written manually, or generated by a dedicated wrapper generator. More often, the program is run on top of an operating system which it can access through function calls to a system library. When analyzing a program using a library, one possible solution is to include the source code of the library with the program. This is not always convenient (if the library is complex), nor possible, if the library source is not available, or not fully written in C, or ultimately relies on kernel services (e.g., for system libraries). An alternative is to provide a stub implementation, i.e., to write, for each library function, a specification of its possible effect on the program. Astrée provides stub libraries for the ARINC 653 standard, and the OSEK/AUTOSAR standards. In case of OSEK systems, Astrée parses the OIL (OSEK Implementation Language) configuration file and generates the corresponding C implementation automatically; in case of AUTOSAR projects the ARXML (AUTOSAR Extensible Markup Language) configuration file is used as an input.

Practical experience on avionics and automotive industry applications are given in [13][38]. They show that industry-sized programs of millions of lines of code can be analyzed in acceptable time with high precision for runtime errors and data races.

IV. Control and Data Flow Analysis

Safety standards such as DO-178C and ISO-26262 require to perform control and data flow analysis as a part of software unit or integration testing and in order to verify the software architectural design. Investigating control and data flow is also subject of the Data Safety guidance [17], and it is a prerequisite for analyzing confidentiality and integrity properties as a part of a security case. Technically, any semantics-based static analysis is able to provide information about data and control flow, since this is the basis of the actual program analysis. However, data and control flow analysis has many aspects, and for some of them, tailored analysis mechanisms are needed.

Global data and control flow analysis gives a summary of variable accesses and function invocations throughout program execution. In its standard data and control flow reports Astrée computes the number of read/write accesses for every global or static variable and lists the location of each access along with the function from which the access is made and the thread in which the function is executed. The control flow is described by listing all callers and callees for every C function along with the threads in which they can be run. Indirect variable accesses via pointers as well as function pointer call targets are fully taken into account. Astrée also provides a call graph visualization enhanced by data flow information, which can be interactively explored (cf. Figure 4). Edges denote function calls, nodes in green represent functions free of alarms, nodes colored red correspond to functions with alarms, and nodes in yellow denote functions which call functions that cause run-time errors but do not cause potential run-time errors by themselves.

More sophisticated information can be provided by two dedicated analysis methods: program slicing and taint analysis. Program slicing [39] aims at identifying the part of the program

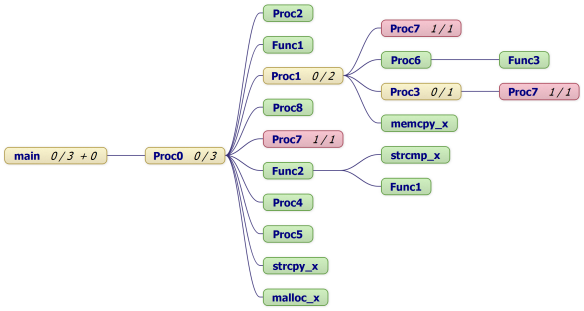


Figure 4. Astrée's Call Graph Visualization.

that can influence a given set of variables at a given program point. Applied to a result value, e.g., it shows which functions, which statements, and which input variables contribute to its computation. Figure 5 shows Astrée's graphical visualization of a slice at the call graph level; green nodes represent functions which contain code which is part of the slice, gray nodes represent functions not contained in the slice.

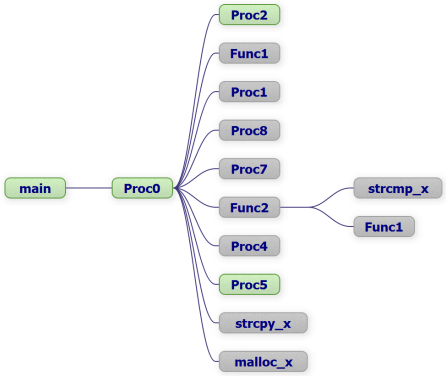


Figure 5. Astrée's Program Slice Visualization.

Taint analysis tracks the propagation of specific data values through program execution. It can be used, e.g., to determine program parts affected by corrupted data from an insecure source. In the following we give a more detailed overview of both techniques.

A. Program Slicing

In this section we will first give a brief overview over classic static slicing. Then, we will introduce a new approach to slicing, which takes into account results from static program analysis via abstract interpretation, and describe its practical application based on an integration in Astrée. Finally, we describe the relevance of these concepts for demonstrating safety and security properties.

A slicing criterion of a program P is a tuple (s, V) where s is a statement and V is a set of variables in P. Intuitively, a slice is a subprogram of P which has the same behavior than P with respect to the slicing criterion (s, V). Computing a statement-minimal slice is an undecidable problem, but, using static analysis, approximative slices can be computed. A well-known algorithm for static slicing first computes a *system*

dependence graph (SDG) and then solves a graph reachability problem on this graph [40]. The system dependence graph is an abstract description of the data- and control dependencies of a computer program.

The precision of this description directly influences the precision of slices computed from the SDG. Computing precise system dependency graphs is a non-trivial task since it requires deriving intricate program properties. These may include points-to information for variable and function pointers, code reachability, context information or possible variable values at certain program points. Astrée's core analysis computes invariants about such properties. We propose *analysis-enhanced slicing*, an approach which feeds some of these invariants to a program slicer contained in Astrée. This slicer can produce sound and compact slices by exploiting points-to and reachability information. A significant precision gain is achieved by reducing the amount of vertices and the amount of data- and control dependences in the system dependence graph. For simple static slicing, over-approximating the set of possible destinations of a pointer variable blows up the size of the system dependence graph as it may add false dependences to statements which contain variables that would otherwise not be included in the slice. This may cause a drastic transitive increase in the number of dependences and vertices. In contrast, interpreting invariants from Astrée's core analysis yields a precise local dependency description of a pointer dereference, which prevents the transitive blow up.

Astrée detects code which is guaranteed to be unreachable for any possible program execution. Ignoring such unreachable code fragments when constructing the system dependence graph further decreases its size. A system dependence graph computed by our approach is a sound abstraction of the data- and control dependences of a computer program. This follows from the soundness of the Astrée core analysis. As a consequence, the resulting slices are also sound. The amount of precision gain depends on the precision of the exported invariants.

We conducted experiments on programs from automotive and avionic industry in order to gauge the effectiveness of analysis-enhanced slicing. In the following, for each run of the analyzer and slicer we list the average execution time and memory consumption of three separate runs. Table I gives an overview of the programs under test. Exporting the invariants to be fed to the slicer does not significantly affect the performance of the analyzer. Run time and memory consumption increase by around 1% on average.

TABLE I. Projects

Project	#Lines	Reached Code	Analysis Time
Avionic1	417,723	98%	1h 41m 59s
Avionic2	71,566	73%	20m 38s
Automotive1	447,188	87%	52m 25s
Automotive2	1,623,140	17%	1h 12m 39s
Automotive3	10,331	92%	4s
Automotive4	1,705	83%	<1s

On these programs, the slicer has been executed in two different modes. In one mode it takes into account exported invariants (EXPORT) and in the other it does not (ALL). In the latter mode it assumes that each pointer may point to every variable of matching type. Table II and Table III show the execution time and memory consumption of the two slicer modes, respectively. For three programs the slicer requires more than 32GB in ALL mode and was stopped. In contrast, the slicer always terminates when considering exported invariants (mode EXPORT). In this mode the run time and memory consumption are much lower.

TABLE II. Slicing Efficiency - Run Time

Project	ALL	EXPORT
Avionic1	n/a	6s
Avionic2	23m 2s	1m 12s
Automotive1	n/a	1h 31m 20s
Automotive2	n/a	1h 13m 52s
Automotive3	2s	<1s
Automotive4	<1s	<1s

TABLE III. Slicing Efficiency - Memory

Project	ALL	EXPORT
Avionic1	>32,000 MB	691 MB
Avionic2	5015 MB	654 MB
Automotive1	>32,000 MB	5268 MB
Automotive2	>32,000 MB	1218 MB
Automotive3	87 MB	<1 MB
Automotive4	<1 MB	<1 MB

Finally, we show the number of lines of the computed slices in Table IV. In general, analysis-enhanced slicing yields significantly smaller slices.

TABLE IV. Slicing Precision

Project	ALL	EXPORT
Avionic1	n/a	661
Avionic3	70.817	67.134
Automotive1	n/a	48.380
Automotive2	n/a	39.507
Automotive3	4362	2868
Automotive5	415	176

So far the discussion of the Astrée slicer has been restricted to its context-insensitive mode. In this mode it always takes into account all contexts (call stack). While efficient, this constitutes another source of imprecision, since not all considered contexts describe actual execution paths. Therefore, the Astrée slicer supports a notion of context-sensitivity. Here, too, it relies on the precision of the Astrée core analysis. The call contexts computed during this analysis can be exploited in two different ways. Either all possible call contexts or a real subset of call contexts can be taken into account. Both modes exclude those function calls from the system dependence graph which do not match any of the specified contexts.

Considering all possible call contexts yields a sound slice with increased precision, when compared to the context-insensitive case. In contrast, the slice computed by taking into account a real subset of call contexts does not capture all possible behaviors of the original program which influence the slicing criterion. Instead, the behavior described by the slice is restricted to execution paths which match one of the specified call contexts. Depending on the choice of contexts, slices computed with this approach may be significantly smaller.

Analysis-enhanced slicing can be extended to context-sensitive slicing as well. Exploiting context-sensitive invariants

in the same way as for context-insensitive slicing is sound and yields an increase of precision. In addition, it is possible to extend the framework by also exporting invariants separately per context. This is especially useful when considering only a small amount of contexts since in this case the invariants may be much more precise. Again, the resulting slice is possibly much smaller.

Another important advantage of analysis-enhanced slicing, in comparison with standard static slicing, is its efficiency. While computing sound slices with standard static slicing requires lots of time and memory, those resources are significantly lower for analysis-enhanced slicing. This is due to the smaller size and smaller complexity of the computed system dependence graphs. This efficiency improvement makes it possible to compute slices for very large programs in reasonable time.

In this work we do not consider dynamic slicing since a dynamic slice does not contain all statements potentially affecting the slicing criterion, but only those relevant for a specific subset of program executions, e.g., only those in which an error value can result. This restriction makes dynamic slicing unsuitable for proving program properties.

The different slicing modes presented in this section are relevant for demonstrating safety and security properties. Sound slices can be computed by context-sensitive analysis-enhanced slicing, when taking into account all possible contexts, or by context-insensitive analysis-enhanced slicing. With these slices it is possible to show that certain parts of the code or certain input variables might influence or cannot influence a program section of interest. They yield a global overview of these properties for the entire program.

In contrast to that, context-sensitive analysis-enhanced slicing, which only considers a subset of possible contexts, is more suitable for investigating the influence of a certain code section, e.g. a function, or a module, on the program location of the slicing criterion.

By considering exactly those contexts that pass through the interesting section, it is possible to prove that the program location of the slicing criterion may be influenced or cannot be influenced by this section. As the slices computed with this approach may be much smaller, this approach may yield much preciser results than investigation using sound slices.

B. Taint Analysis

In literature, taint analysis is often mentioned in combination with unsound static analyzers, since it allows to efficiently detect potential errors in the code, e.g., array-index-out-of-bounds accesses, or infeasible library function parameters [3], [22]. Inside a sound runtime error analyzer this is not needed since typically more powerful abstract domains can track all undefined or unspecified behaviors. Inside a sound analyzer, taint analysis is primarily a technique for analyzing security properties. Its advantage is that users can flexibly specify taints, taint sources, and taint sinks, so that application-specific data and control flow requirements can be modeled.

In order to be able to leverage this efficient family of analyses in sound analyzers, one must formally define the properties that may be checked using such techniques. Then it is possible to prove that a given implementation is sound with respect to that formal definition, leading to clean and well defined analysis results. Taint analysis consists in discovering data dependencies using the notion of taint propagation. Taint propagation can be formalized using a non-standard semantics of programs, where an imaginary taint is associated to some input values. Considering a standard semantics using a successor relation between program states, and considering that a program state is a map from memory locations (variables, program counter, etc.) to values in $\mathcal{V}$, the *tainted* semantics relates tainted states, which are maps from the same memory locations to $\mathcal{V} \times \{\text{taint}, \text{notaint}\}$, and such that if we project on $\mathcal{V}$ we get the same relation as with the standard semantics.

To define what happens to the *taint* part of the tainted value, one must define a *taint policy*. The taint policy specifies:

Taint sources which are a subset of input values or variables such that in any state, the values associated with that input values or variables are always tainted.

Taint propagation describes how the tainting gets propagated. Typical propagation is through assignment, but more complex propagation can take more control flow into account, and may not propagate the taint through all arithmetic or pointer operations.

Taint cleaning is an alternative to taint propagation, describing all the operations that do not propagate the taint. In this case, all assignments not containing the taint cleaning will propagate the taint.

Taint sinks is an optional set of memory locations. This has no semantical effect, except to specify conditions when an alarm should be emitted when verifying a program (an alarm must be emitted if a taint sink may become tainted for a given execution of the program).

A sound taint analyzer will compute an over-approximation of the memory locations that may be mapped to a tainted value during program execution. The soundness requirement ensures that no taint sink warning will be overlooked by the analyzer.

At first sight, it is easy to implement an efficient taint analysis: keeping track of the taint only requires one bit per variable. One bit means that iterating is fast, but when considering the whole set of variables, a sound analysis will require to compute fixpoints over bitvectors, meaning that in the worst case, convergence may take as many iterations as the number of variables in the program to analyze, each iteration requiring to compare two bit-vectors (again a linear cost). Fortunately, modern analyzers use advanced algorithmic techniques to efficiently compute such fixpoints, and basic taint analysis is way less complex than simple interval analysis.

A more precise taint analysis may be needed though: the typical case being taint cleaning functions that may fail to clean the tainting. In such cases, the function will usually return a value describing whether the cleaning succeeded. In order not to raise false alarms, it is then necessary to use a *relational* abstraction, keeping track of the relation between the taint bit, and the values of other variables. Doing so blindly leads directly to unscalable analyzes. In order to preserve the efficiency of the taint analysis, relations should only be kept between some taint values and some variable values, and only in a limited context. This leads to relational analysis through packing, a technique already in use for relational numerical domains, such as octagons or polyhedra, but which requires some expertise to fine-tune the heuristics. Finding the right packs depending on the parametric sources, cleaning and taint

sinks is a research subject that will be our future work.

The tainted semantics can easily be extended to a mix of different hues of tainting, corresponding to an extension of the taint set associated with values. Then propagation can get more complex, with tainting not just being propagated but also changing hue depending on the instruction. This is needed not only to carry different taint analysis in one go, but also as a necessary semantic step to define some multi-level notions security breach. Carefully implemented, such extensions lead to a rather flexible and powerful data dependency analysis, while remaining scalable.

V. Conclusion

In this article, we have given an overview of code-level defects and vulnerabilities relevant for functional safety and security. We have shown that many security attacks can be traced back to behaviors undefined or unspecified according to the C semantics. By applying sound static runtime error analyzers, a high degree of security can be achieved for safety-critical software since the absence of such defects can be proven. In addition, security hyperproperties require additional analyses to be performed, which, by nature, have a high complexity. We have given two examples of scalable dedicated analyses, program slicing and taint analysis. Applied as extensions of sound static analyzers, they allow to further increase confidence in the security of safety-critical embedded systems.

Acknowledgment

This work was funded within the project ARAMiS II by the German Federal Ministry for Education and Research with the funding ID 01—S16025. The responsibility for the content remains with the authors.

References

[1] D. Kästner, L. Mauborgne, and C. Ferdinand, “Detecting Safety- and Security-Relevant Programming Defects by Sound Static Analysis,” in The Second International Conference on Cyber-Technologies and Cyber-Systems (CYBER 2017), ser. IARIA Conferences, J.-C. B. Rainer Falk, Steve Chan, Ed., vol. 2. IARIA XPS Press, 2017, pp. 26–31.

[2] MISRA (Motor Industry Software Reliability Association) Working Group, MISRA-C:2012 Guidelines for the use of the C language in critical systems, MISRA Limited, Mar. 2013.

[3] Software Engineering Institute SEI – CERT Division, SEI CERT C Coding Standard – Rules for Developing Safe, Reliable, and Secure Systems. Carnegie Mellon University, 2016.

[4] The MITRE Corporation, “CWE – Common Weakness Enumeration.” [Online]. Available: https://cwe.mitre.org [retrieved: Sep. 2017].

[5] Radio Technical Commission for Aeronautics, “RTCA DO-178C. Software Considerations in Airborne Systems and Equipment Certification,” 2011.

[6] IEC 61508, “Functional safety of electrical/electronic/programmable electronic safety-related systems,” 2010.

[7] ISO 26262, “Road vehicles – Functional safety,” 2011.

[8] CENELEC EN 50128, “Railway applications – Communication, signalling and processing systems – Software for railway control and protection systems,” 2011.

[9] P. Cousot and R. Cousot, “Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints,” in Proc. of POPL’77. ACM Press, 1977, pp. 238–252. [Online]. Available: http://www.di.ens.fr/~cousot/COUSOTpapers/POPL77.shtml [retrieved: Sep. 2017].

[10] D. Kästner, “Applying Abstract Interpretation to Demonstrate Functional Safety,” in Formal Methods Applied to Industrial Complex Systems, J.-L. Boulanger, Ed. London, UK: ISTE/Wiley, 2014.

[11] J. Souyris, E. Le Pavec, G. Himbert, V. Jégu, G. Borios, and R. Heckmann, “Computing the worst case execution time of an avionics program by abstract interpretation,” in Proceedings of the 5th Intl Workshop on Worst-Case Execution Time (WCET) Analysis, 2005, pp. 21–24.

[12] D. Delmas and J. Souyris, “ASTRÉE: from Research to Industry,” in Proc. 14th International Static Analysis Symposium (SAS2007), ser. LNCS, no. 4634, 2007, pp. 437–451.

[13] D. Kästner, A. Miné, L. Mauborgne, X. Rival, J. Feret, P. Cousot, A. Schmidt, H. Hille, S. Wilhelm, and C. Ferdinand, “Finding All Potential Runtime Errors and Data Races in Automotive Software,” in SAE World Congress 2017. SAE International, 2017.

[14] D. Kästner, C. Rustemeier, U. Kiffmeier, D. Fleischer, S. Nenova, R. Heckmann, M. Schlickling, and C. Ferdinand, “Model-Driven Code Generation and Analysis,” in SAE World Congress 2014. SAE International, 2014.

[15] Wired.com, “The jeep hackers are back to prove car hacking can get much worse,” 2016. [Online]. Available: https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/ [retrieved: Sep. 2017]

[16] Y. Younan, W. Joosen, and F. Piessens, “Code injection in c and c++ : A survey of vulnerabilities and countermeasures,” Departement Computerwetenschappen, Katholieke Universiteit Leuven, Tech. Rep., 2004.

[17] SCSC Data Safety Initiative Working Group [DSIWG], “Data Safety (Version 2.0) [SCSC-127B],” Safety-Critical Systems Club, Tech. Rep., Jan. 2017.

[18] Radio Technical Commission for Aeronautics, “RTCA DO-178B. Software Considerations in Airborne Systems and Equipment Certification,” 1992.

[19] ——, “RTCA DO-333. Formal Methods Supplement to DO-178C and DO-278A,” 2011.

[20] ISO 26262, “Road vehicles – Functional safety – Part 6: Product development at the software level,” 2011.

[21] ISO/DIS 26262, “Road vehicles – Functional safety,” 2016.

[22] ISO/IEC, “Information Technology – Programming Languages, Their Environments and System Software Interfaces – Secure Coding Rules (ISO/IEC TS 17961),” Nov. 2013.

[23] MISRA (Motor Industry Software Reliability Association) Working Group, MISRA-C:2012 – Addendum 2. Coverage of MISRA C:2012 against ISO/IEC TS 17961:2013 ”C Secure”, MISRA Limited, Apr. 2016.

[24] MISRA (Motor Industry Software Reliability Association) Working Group , MISRA-C:2012 Amendment 1 – Additional security guidelines for MISRA C:2012, MISRA Limited, Apr. 2016.

[25] CERT – Software Engineering Institute, Carnegie Mellon University, “SEI CERT Coding Standards Website.” [Online]. Available: https://www.securecoding.cert.org [retrieved: Sep. 2017].

[26] MISRA (Motor Industry Software Reliability Association) Working Group, MISRA-C:2012 – Addendum 3. Coverage of MISRA C:2012 (including Amendment 1) against CERT C 2016 Edition, MISRA Limited, Jan. 2018.

[27] Wikipedia, “Blaster (computer worm).” [Online]. Available: https://en.wikipedia.org/wiki/Blaster_(computer_worm) [retrieved: Sep. 2017].

[28] CERT – Vulnerability Notes Database, “Vulnerability Note VU#720951 – OpenSSL TLS heartbeat extension read overflow discloses sensitive information.” [Online]. Available: http://www.kb.cert.org/vuls/id/720951 [retrieved: Sep. 2017].

[29] NIST – National Vulnerability Database, “CVE-2009-1888: SAMBA ACLs Uninitialized Memory Read.” [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2009-1888 [retrieved: Sep. 2017].

[30] J. Yang, A. Cui, J. Gallagher, S. Stolfo, and S. Sethumadhavan, “Concurrency attacks,” in In the Fourth USENIX Workshop on Hot Topics in Parallelism (HOTPAR12), 2012.

[31] M. R. Clarkson and F. B. Schneider, "Hyperproperties," Journal of Computer Security, vol. 18, 2010, pp. 1157–1210.

[32] A. Sabelfeld and A. C. Myers, "Language-based information-flow security," IEEE Journal on Selected Areas in Communications, vol. 21, no. 1, 2003, pp. 5–19.

[33] M. Assaf, D. A. Naumann, J. Signoles, E. Totel, and F. Tronel, "Hypercollecting semantics and its application to static analysis of information flow," CoRR, vol. abs/1608.01654, 2016. [Online]. Available: http://arxiv.org/abs/1608.01654 [retrieved: Sep. 2017].

[34] A. Miné, L. Mauborgne, X. Rival, J. Feret, P. Cousot, D. Kästner, S. Wilhelm, and C. Ferdinand, "Taking Static Analysis to the Next Level: Proving the Absence of Run-Time Errors and Data Races with Astrée," Embedded Real Time Software and Systems Congress ERTS^2, 2016.

[35] A. Miné, "The Octagon Abstract Domain," Higher-Order and Symbolic Computation, vol. 19, no. 1, 2006, pp. 31–100.

[36] A. Miné, "Static analysis of run-time errors in embedded real-time parallel C programs," Logical Methods in Computer Science (LMCS), vol. 8, no. 26, Mar. 2012, p. 63.

[37] OSEK/VDX, OSEK/VDX Operating System. Version 2.2.3., http://www.osek-vdx.org, 2005.

[38] A. Miné and D. Delmas, "Towards an Industrial Use of Sound Static Analysis for the Verification of Concurrent Embedded Avionics Software," in Proc. of the 15th International Conference on Embedded Software (EMSOFT'15). IEEE CS Press, Oct. 2015, pp. 65–74.

[39] M. Weiser, "Program slicing," in Proceedings of the 5th International Conference on Software Engineering, ser. ICSE '81. IEEE Press, 1981, pp. 439–449.

[40] S. Horwitz, T. Reps, and D. Binkley, "Interprocedural slicing using dependence graphs," ACM Trans. Program. Lang. Syst., vol. 12, no. 1, Jan. 1990, pp. 26–60. [Online]. Available: http://doi.acm.org/10.1145/77606.77608

Integrating Autonomous Vehicle Safety and Security Analysis Using STPA Method and the Six-Step Model

Giedre Sabaliauskaite, Lin Shen Liew, and Jin Cui

Centre for Research in Cyber Security (iTrust)
Singapore University of Technology and Design
Singapore 487372
Email: giedre@sutd.edu.sg, linshen_liew@sutd.edu.sg, jin_cui@sutd.edu.sg

***Abstract*—Safety and security are two inter-dependent key properties of autonomous vehicles. They are aimed at protecting the vehicles from accidental failures and intentional attacks, which could lead to injuries and loss of lives. The selection of safety and security countermeasures for autonomous vehicles depends on the driving automation levels, defined by the international standard SAE J3016. However, current vehicle safety standards ISO 26262 do not take the driving automation levels into consideration. We propose an approach for integrating autonomous vehicle safety and security processes, which is compliant with the international standards SAE J3016, SAE J3061, and ISO 26262, and which considers driving automation levels. It incorporates the System-Theoretic Process Analysis method into autonomous vehicle safety analysis, and uses the Six-Step Model as a backbone for achieving integration and alignment among safety and security processes and artefacts throughout the entire autonomous vehicle's life-cycle.**

Keywords–Autonomous vehicle; safety; security; Six-Step Model; STPA.

I. INTRODUCTION

Autonomous Vehicles (AVs), the self-driving vehicles, are safety-critical Cyber-Physical Systems (CPS) – complex engineering systems, which integrate embedded computing technology into physical phenomena. Safety and security are two key properties of CPSs, which share the same goal – protecting the systems from undesirable events: failures (safety) and intentional attacks (security).

Ensuring the safety of autonomous vehicles, i.e., reducing the number of traffic crashes to prevent injuries and save lives, is a top priority in autonomous vehicle development. Safety and security are interdependent (e.g., security attacks can cause safety failures, or security countermeasures may weaken CPS safety and vice versa), therefore they have to be aligned in the early system development phases to ensure the required level of protection [1][2][3].

Although AVs could be considered to be smaller and/or less complex systems as compared to other CPSs, such as, power plants or water treatment systems, they face some unique challenges, which have to be taken into consideration when analyzing their safety and security.

Firstly, there are six different levels of driving automation ranging from no driving automation (level 0) to full driving automation (level 5), as described by the international standard SAE J3016 [4]. The levels describe who (human driver or automated system) performs the driving tasks and monitors the driving environments under certain environmental conditions. Thus, AV safety and security depend on the driving automation levels and the environmental conditions.

Secondly, the AV domain is relatively new, and therefore, there are no international standards developed specifically for AV safety and security yet. Currently, the ISO 26262 standard, which describes functional safety of road vehicles, is being used for AV safety analysis [5]. However, it is not sufficient for AVs, as argued in [6] and [7]. ISO 26262 addresses the safety of each function, or item, of the vehicle separately, since the driver is responsible for everything what falls outside the item. However, in AV, it is necessary to ensure safety at all times, especially at the high automation levels, when there is no driver in the vehicle.

Moreover, the ISO 26262-recommended hazard analysis techniques, such as Fault Tree Analysis (FTA) and Failure Modes and Effects Analysis (FMEA), are ineffective in identifying the systemic and interaction related problems of complex software intensive E/E systems [8]. In view of that, some recent works (e.g., [9][10]) employ a relatively new hazard analysis technique, System-Theoretic Process Analysis (STPA) [11], to complement the ISO 26262 for a more comprehensive AV safety analysis.

To address vehicle security needs, the SAE J3061 standard has been developed [12]. It defines cyber-security lifecycle of cyber-physical vehicle systems. However, the security lifecycle, defined in SAE J3061, is analogous to the vehicle safety lifecycle described in ISO 26262, and therefore, it is not sufficient for AV cyber-security analysis. ISO and SAE are currently jointly developing vehicle standard ISO 21434 [13], which will replace SAE J3061 in 2019.

How can we analyze AV safety and security throughout its entire life-cycle in a consistent way, and provide required level of protection? In our previous work, we proposed a Six-Step Model for modeling and analysis of CPS safety and security [15][16]. It incorporates six dimensions (hierarchies) of CPS, namely, functions, structure, failures, safety countermeasures, cyber-attacks, and security countermeasures. Furthermore, it uses relationship matrices to model interdependencies between these dimensions. The Six-Step Model enables comprehensive analysis of CPS safety and security, as it utilizes system functions and structure as a knowledge base for understanding the effect of failures and attacks on the system. Furthermore, we presented an initial approach

TABLE I. Driving automation levels [4][14].

Level	Name	Execution of steering/ acceleration/deceleration	Monitoring of driving environment	Performance of DDT fallback	Driving modes (environment and conditions)
0	No automation	Human driver	Human driver	Human driver	N/A
1	Driver assistance	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial automation	System	Human driver	Human driver	Some driving modes
3	Conditional automation	System	System	Human driver	Some driving modes
4	High automation	System	System	System	Some driving modes
5	Full automation	System	System	System	All driving modes

for applying the Six-Step Model for AV safety and security analysis in [1].

In this paper, we extend the initial approach, proposed in [1], to enable a comprehensive analysis of AV safety and security using STPA method and the Six-Step Model, which is compliant with the international standards SAE J3016, SAE J3061, and ISO 26262.

The remainder of this paper is structured as follows. Section II describes the preliminaries. The proposed approach is explained in Section III, and a Six-Step Model example is included in Section IV. Finally, Section V concludes the paper and describes our future work.

II. PRELIMINARIES

This section describes the AVs, their safety and security analysis, the Six-Step Model, and STPA method.

A. Autonomous Vehicles' Main Terms and Definitions

The real-time operational and tactical functions required to operate the vehicle in on-road traffic include lateral and longitudinal vehicle motion control, monitoring the driving environment, object and event response execution, maneuver planning, and enhancing conspicuity via lighting, signaling, etc. [4]. These functions are collectively called the Dynamic Driving Task (DDT) [4]. An automated driving system of an AV performs entire or part of DDT, depending on AV driving automation level. In addition to DDT, AVs implement DDT-fallback – a response mechanism, which enables a human driver or an automated system to take over performance of the entire DDT in case of unexpected situations, e.g., during traffic jams on freeways.

SAE International (SAE) has developed an international standard, SAE J3016 [4], to describe the levels of vehicle driving automation. The standard has been widely adopted by international organizations, such as the National Highway Traffic Safety Administration (NHTSA) [14].

There are six driving automation levels (see Table I):

- Level 0 – the human driver performs entire DDT.
- Level 1 – an automated system on the vehicle can assist the human driver to perform either the lateral or the longitudinal vehicle motion, while driver monitors the driving environment and performs the rest of DDT.
- Level 2 – an automated system performs the lateral and the longitudinal vehicle motion, while driver monitors the driving environment and performs the rest of DDT.
- Level 3 – an automated system can perform entire DDT, but the human driver must be ready to take back control when the automated system requests.
- Level 4 – there is no human driver; an automated system conducts the entire DDT, but it can operate only in certain environments and under certain conditions (driving modes).
- Level 5 – there is no human driver; an automated system performs entire DDT in all environments and under all conditions that a human driver could perform them.

Level 3-5 vehicles are called the highly automated vehicles, since their automated systems (not a human driver) are responsible for monitoring the driving environment [14]. Furthermore, level 1-4 vehicles are designed to operate only in certain environments and under certain conditions, while level 5 vehicles - in all environments and under all conditions.

AV functions can be grouped into three main categories: perception (perception of the external environment/context, in which vehicle operates), decision & control (decisions and control of vehicle motion with respect to the external environment/context that is perceived), and vehicle platform manipulation (sensing, control, and actuation of the vehicle, with the intention of achieving desired motion) [17][18]. An international standard for describing AV functions and functional interfaces, SAE J3131, is currently under development.

AV structural architecture consists of two main systems: a) cognitive driving intelligence, which implements perception and decision & control functions, and b) vehicle platform, which is responsible for vehicle platform manipulation [17]. Each system consists of components, which belong to four major groups: hardware, software, communication, and human-machine interface [18][19]. See Section IV for more details.

B. A Six-Step Model

In our earlier work [15][16], we proposed a Six-Step Model to enable comprehensive CPS safety and security analysis. The model is constructed using the following six steps (see Figure 1):

1) The first step is aimed at modeling the functional hierarchy of the system. The functions are defined using the Goal Tree (GT), which is constructed starting with the goal (functional objective) and then defining functions and sub-functions, needed for achieving this goal. A relationship matrix, F-F, is used to define the relationships between functions, which can be high, medium, low, or very low.
2) In the second step, system's structural hierarchy is defined using the Success Tree (ST) to describe system's structure as a collection of sub-systems and units. Furthermore, the relationships between structure and functions are defined using a relationship matrix S-F, as shown in Figure 1.

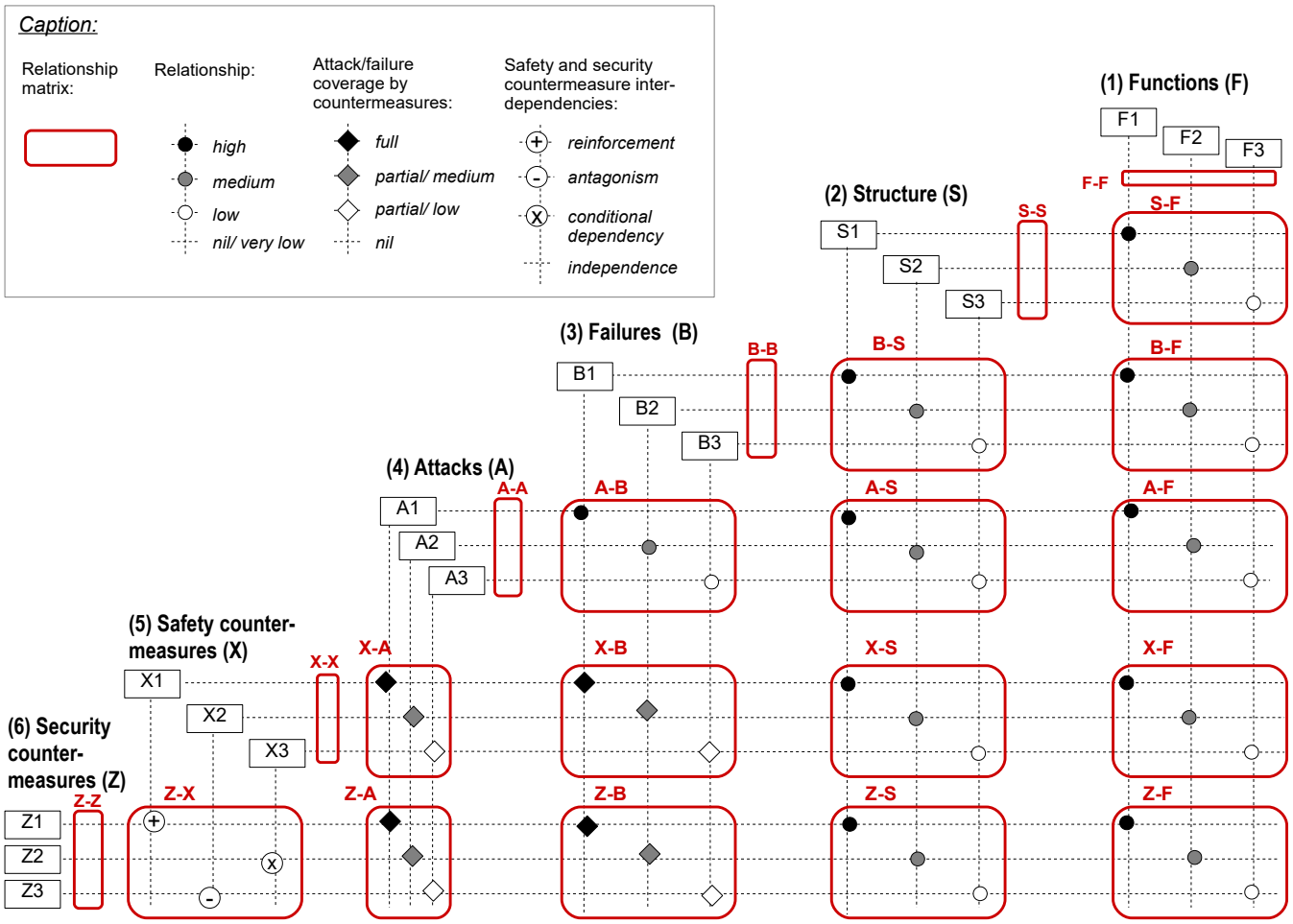


Figure 1. The Six-Step Model.

3) The third step is focused on safety hazard analysis. In this step, system's failures are identified and added to the model. In addition, the relationships between failures, system structure and functions are identified, and the corresponding relationship matrices – B-B, B-S, and B-F – are added to the model.
4) The fourth step focuses on security threat analysis. In this step, attacks are identified and added to the model along with the relationship matrices to describe relationships between attacks, failures, structure and functions. Relationship matrix A-B (attacks – failures) is used to determine the failures, which could be triggered by a successful attack.
5) In the fifth step, safety countermeasures are added to the model and their relationships are identified. Matrices X-A and X-B show the coverage of attacks and failures by safety countermeasures, where white rhombus indicates that the countermeasure provides low protection from attack/failure; gray rhombus - medium protection; black rhombus - full protection (see Figure 1).
6) Finally, in the last step, security countermeasures are added to the model and their relationships are established. Similarly to matrices X-A and X-B from the previous step, two new matrices Z-A and Z-B are added to define the coverage of attacks and failures by security countermeasures. The security countermeasures, added in this step, could be used to protect the system from attacks and failures, not covered by the safety countermeasures. Furthermore, matrix Z-X is used to capture the inter-dependencies between safety and security countermeasures, such as reinforcement, antagonism, conditional dependency, and independence, as defined in [20].

After completion of steps 5 and 6, it is important to analyze if there were any changes made to system's structure, as some countermeasures might require the use of additional components, e.g., sensors or controllers. If the changes occur, it is necessary to return to the step 2 to add the new components and then repeat steps 3-6.

The Six-Step Model, constructed throughout steps 1-6, interconnects six hierarchies of the systems (functions, structure, failures, attacks, and safety and security countermeasures) by forming a hexagon-shaped structure of their relationships, as shown in Figure 2. The relationships help to ensure alignment between these hierarchies, and they have to be maintained throughout the entire system's life-cycle.

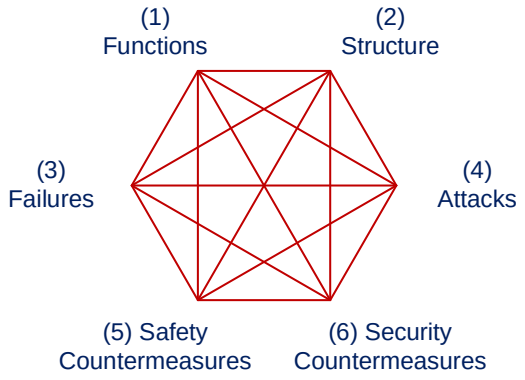


Figure 2. Relationships among hierarchies of the Six-Step Model.

C. AV Safety Analysis

The ISO 26262 standard [5] defines functional safety for automotive equipment applicable throughout the life-cycle of all automotive Electronic and Electrical (E/E) safety-related systems. It aims to address possible hazards caused by the malfunctioning behavior of E/E systems. The safety process consists of several phases, such as concept, product development, and production, operation, service and decommissioning. Hazard Analysis and Risk Assessment (HARA) is performed during the concept phase, where hazardous events, safety risks, and safety goals are identified. These goals are further refined into the safety requirements, and the safety countermeasures are designed and implemented. Fault Tree analysis [19] can be used during HARA to identify the conditions and events that could lead to high-level hazardous events. Fault tree refines top-level hazardous event into intermediate events and basic events, which are interconnected by AND and OR logical operators.

However, as ISO 26262 requires the presence of the human driver inside the vehicle to respond to unexpected environments and conditions, it is likely to be unfit for AVs where humans have little or no part in the driving. In fact, various means have been proposed to complement the ISO 26262 standard in ensuring the safety of complex software intensive E/E systems like AVs. For instance, [6][21][22] highlight the importance of having an adequate item definition; note that the *item definition* is a prerequisite for Hazard Analysis and Risk Assessment (HARA) process as per ISO 26262 standard.

Traditionally, the *item* delivers only one function like steering and braking, and malfunctions caused by interactions between *item* and other entity are simply eliminated by design [21]. In contrast, the *item* for AVs may deliver multiple functions (e.g., a complex braking system, which includes regenerative braking), and thus defining it would require more careful consideration. In view of that, Ibarra et al. [21] model the item definition based on Goal Structuring Notation [23].

Inadequate *item definition* would eventually result in inadequate Safety Goals (SGs). Such SGs could be violated even when the system is fault-free (e.g., having no sensor failure at all). There is an ISO work-group called Safety of the Intended Functionality (SOTIF) aiming to provide a guidance on handling such violations, but its specification is yet to be published [24]. In [6], Warg et al. suggest that the *item definition* should be a product of an iterative process, which comprises three steps: (1) perform HARA where generic operational situation and hazard trees are used to generate a list of Hazardous Events (HEs), and the trees get updated according to the Scope and Requirements of the Function (SRF) updated on last iteration; (2) identify the dimensioning HEs based on a set of rules as described in [25]; (3) refine the SRF according to the dimensioning HEs. The iterative process ends once the HEs and SRF are mature (based on certain criteria) for creating the safety goals and an *item definition*, which are then inputted to the *functional safety concept* phase.

How an inadequate *item definition* would lead to inadequate safety requirements (e.g., Functional Safety Requirements (FSRs) and Technical Safety Requirements (TSRs)) is exemplified in [22], where Bergenhem et al. claim that there is a substantial gap between any adjacent levels of safety requirements (e.g., between SG and its corresponding FSRs, and between FSR and its corresponding TSRs). Such gap necessitates a complex rationale for verification of the completeness and correctness of these requirements. Therefore, they recommend that each safety requirement level be refined to a certain extent - e.g., prior to deriving its FSRs, a high-level SG is translated into multiple lower-level SGs to reduce the gap and subsequently ease the rationale and hence the verification.

In addition, another challenge lies in specifying the HEs for the AVs. Conventionally, in the context of automotive industry, the HEs were identified by using hazard analysis techniques recommended by the ISO 26262 such as Fault Tree Analysis (FTA), and Failure Modes and Effects Analysis (FMEA). A brief review on the effectiveness of these techniques with respect to modern complex E/E systems can be found in [8]. In essence, these traditional techniques are based on simple linear chain-of-event accident causality models, originally intended for systems where the safety issues are mainly caused by random hardware failures; they are unfit for AVs, which could be compromised by software error, dysfunctional interaction, etc. apart from hardware failures.

Recently, a relatively new hazard analysis technique known as STPA has gained popularity among the researchers and practitioners in engineering the safety of complex systems in various domains [26][27][28][29][30]. STPA [11] is developed based on STAMP (Systems-Theoretic Accidents Model and Process) - a novel accident causality model that consider the safety of a complex system as a system control problem rather than a component failure or reliability problem. It aims to identify inadequate control scenarios, which could result in unwanted losses/accidents, and then develop detailed safety constraints to avoid/mitigate such scenarios. Note that the inadequate controls can occur owing to human error, dysfunctional interaction, software failure, etc. Arguments on why traditional safety engineering approaches (including traditional accident causality models) are inadequate for addressing the safety of complex systems can be found in [11]. In fact, some works such as [26] and [27] have demonstrated that STPA is able to identify not only all the hazardous/unsafe scenarios, which FTA identifies, but also those that FTA fails to identify.

To address the safety needs of AV more comprehensively, both [9] and [10] have proposed to integrate STPA into the concept phase of ISO 26262. Figure 3 illustrates the integration between STPA and ISO 26262 concept phase, which consists of five main stages:

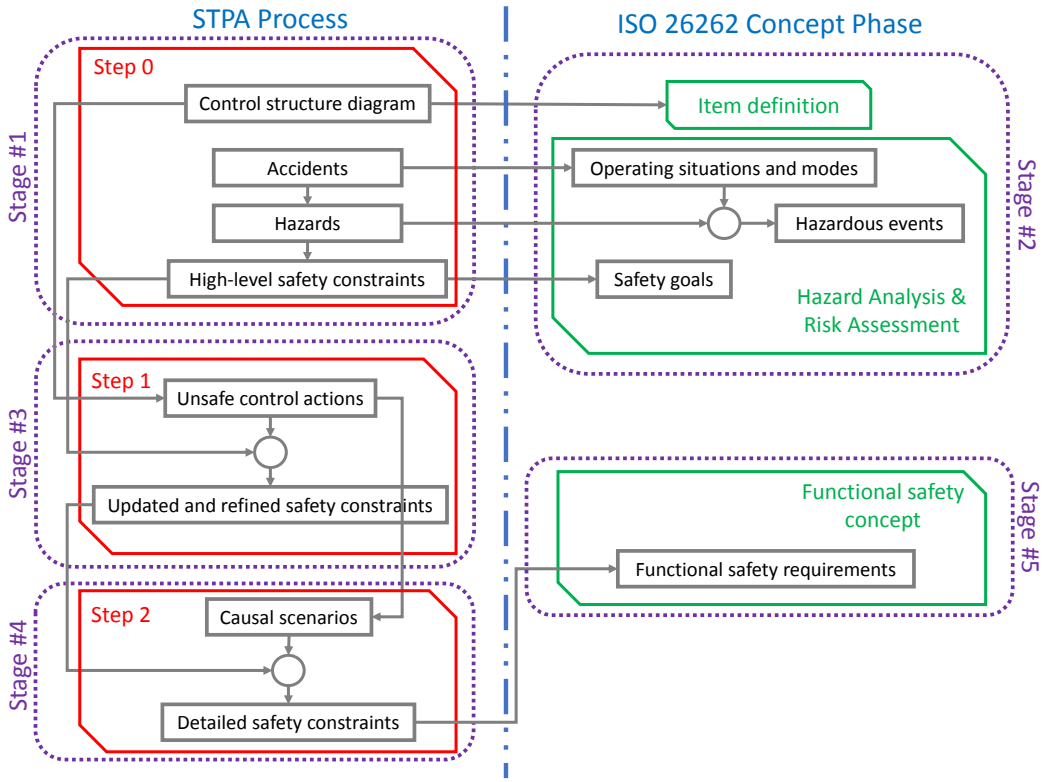


Figure 3. Integration of STPA into the concept phase of ISO 26262.

1) Perform Step 0 of STPA. Firstly, identify the accidents that could happen to the AV. Secondly, identify the high-level system hazards, which could lead to the identified accidents. Thirdly, define the high-level safety constraints for mitigating/avoiding the identified hazards. Finally, draw the system-level control structure - a diagram that represent the functional model of the system, which shows the major components of the system as well as their interfaces and boundaries.
2) Utilize the output of STPA Step 0 in ISO 26262 concept phase. Firstly, the information extracted from the control structure could contribute to a more precise *item definition*. Secondly, the list of accidents is used to derive the operating situations and modes. Thirdly, identified hazards and derived operating situations are combined to form the HEs. Fourthly, the high-level safety constraints are considered in formulating the safety goals for the HEs.
3) Perform Step 1 of STPA. Firstly, identify the control actions from the control structure diagram. Secondly, check if the control actions can be unsafe (i.e causing some previously identified hazards or additional ones), if they are not provided, or incorrectly provided, or untimely provided, or stopped/applied too soon/long. Then, translate the unsafe control actions into safety constraints by using the the guide words like ”shall” and ”must”, which could also be used to refined previously identified safety constraints.
4) Perform Step 2 of STPA. Firstly, identify the causal scenarios for the unsafe control actions, based on the control structure diagram. Secondly, derive new or more detailed safety constraint for the identified causal scenarios.
5) Utilize the output of STPA Step 2 in ISO 26262 concept phase. The finalized and detailed safety constraints are inputted to the *functional safety concept* of ISO 26262 to derive the functional safety requirements.

D. AV Security Analysis

SAE J3061 is a vehicle cyber-security standard, which was developed using the ISO 26262 standard as a base. Thus, both standards consist of similar phases. Security process, defined by SAE J3061, includes concept, product development, and production & operation phases. Threat Analysis and Risk Assessment (TARA) is performed during the concept phase, where threats, security risks, and security goals are defined. In the product development phase, security requirements are defined based on the security goals, and the security countermeasures are developed.

Attack tree analysis [12][31] is often used for performing TARA. It helps to determine the potential paths that an attacker could take to lead to the top-level threat [12]. An attack tree is a graph, where the nodes represent attack events, and the edges - attack paths through system, which could be connected using AND and OR gates.

Behavior diagrams, such as Data-Flow Diagrams (DFD) [32] and Information-Flow Diagrams (IFD) [16] could be used for identifying the attacks to be included in attack trees analysis. DFDs include elements, such as processes, data flows, and data store, and are used to model data flows between

software components. IFDs include units and information flows between them, and could be used to model information flows between software and hardware components, such as actuators, controllers, sensors, etc. In [16], we proposed a method for generating IFDs using the Six-Step model in order to identify possible attacks on CPSs.

III. Integrated Autonomous Vehicle Safety and Security Analysis Approach

This section proposes an approach for integrated AV safety and security analysis, which used the Six-Step Model and STPA methods, and is compliant with the international standards SAE J3016, SAE J3061 and ISO 26262.

Figure 4 describes the proposed approach and shows the relationships between steps of the Six-Step Model and various artefacts from AV definition and design, safety analysis, and security analysis processes.

The steps of the AV Six-Step Model are performed in the following order:

- Steps (1) and (2). Autonomous driving functions and the systems (structure), which implement these functions, are defined during AV definition and design process. As a result, AV functional and structural hierarchies are defined and added to the Six-Step Model, along with their relationships. The functions and structure will be continuously updated based on the results of the safety and security analysis.
- Steps (3) and (4). These steps correspond to AV vulnerability (hazard and threat) analysis. On the safety side, HARA (as defined by ISO 26262) and STPA are performed in order to identify and evaluate hazardous events, and define AV functional safety requirements. At the end of the hazard analysis phase, failures, which are considered in security requirements, are extracted from the fault trees and added to the the Six-Step Model (Step (3)). On the security side, TARA (as defined by SAE J3061) is performed in order to evaluate security threats and derive AV functional security requirements. The AV structural hierarchy, defined in step (2), could be used to define attack surfaces and construct information-flow models (see [16]), which helps to identify possible attacks and construct attack trees, as described in Section II-D. The risks associated with each attack are then evaluated and security requirements are defined. Similarly to failures, the attacks, included in security requirements, are extracted from the attack trees and added to the Six-Step Model (Step (4)). The relationships between attacks, failures, functions, and structures, are also added to the Six-Step model.
- Steps (5) and (6). During these steps, safety and security countermeasures are selected and added to the model along with their relationships to remaining elements of the model. On the safety side, functional safety requirements are refined into technical requirements and corresponding countermeasures are designed for satisfying these requirements. Similarly, on the security side, functional security requirements are decomposed into technical requirements for security countermeasures. The countermeasures from both sides are added to the Six-Step Model to analyze their relationships to the remaining elements of the model. In particular, the matrices are useful to make sure that each countermeasure is really needed (addresses attacks/failures not completely covered by any other countermeasures, shown in matrices X-A, X-B, Z-A, and A-B), and that there are no contradictions among countermeasures (matrix Z-X).

The AV Six-Step Model, constructed during steps (1)-(6), is a backbone of AV vulnerability analysis. It supports three AV processes, namely, AV definition and design, AV safety analysis, and AV security analysis, as shown in Figure 4. Furthermore, it enables integration of safety and security artefacts, developed throughout the entire AV life-cycle (such as failures, attacks, safety and security countermeasures) into AV function and structure hierarchies to assure their consistency and completeness.

The AV Six-Step Model has to be maintained throughout the entire AV life-cycle. This is particularly important for security analysis, as new threats are continually identified and have to be analyzed.

IV. Six-Step Model Example of an AV

This example describes a high automation AV. Its Six-Step Model is shown in Figure 5. Due to space limitations, only an excerpt of the Six-Step Model is included in Figure 5. Furthermore, only the high degree relationships between elements are shown.

The AV, described in this example, performs three main autonomous driving functions, i.e., perception, decision & control, and vehicle platform manipulation, as described in Section II-A. The perception function can be further decomposed into sensing, sensor fusion, localization, semantic understanding, and world model (see [17]). These functions are added at the top of to Six-Step Model and their inter-relationships are identified, as shown in Figure 5 step (1).

The main systems of AV, which implement driving automation functions, are: cognitive driving intelligence, vehicle platform, and communication system [17][18]. The cognitive driving intelligence includes on-board computer and external sensors for perception of environment, such as Light Detection and Ranging (LIDAR), cameras, and ultrasound sensors [33]. The vehicle platform includes controllers (ECUs) and actuators, which implement the desired motion. The communication system includes in-vehicle and V2X (vehicle to vehicle, infrastructure, and humans) communication networks. In this example, only in-vehicle communication is considered. All these structural elements are added to model in step (2).

In steps (3) and (4), we add failures and attacks to the Six-Step Model. In this example, we describe the LIDAR failures and attacks. LIDAR combined with camera are used for navigation in AVs. Together with other sensors, they provide necessary information for performing AV localization function (determining the location of vehicle with respect to its surroundings). LIDAR includes the following components: laser lens filter, receiver, power regulator, rotating mirror, etc. In this example, LIDAR is connected to on-board computer through Ethernet in order to send its readings.

Fault trees are commonly used for safety analysis, as described in Section II-C. An example of LIDAR fault tree is

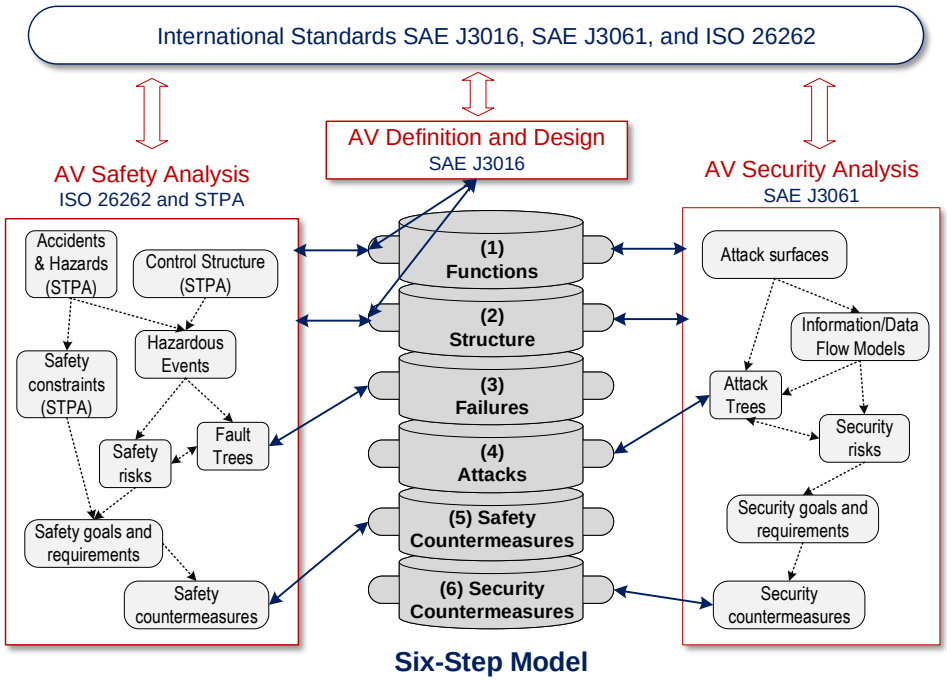


Figure 4. The Six-Step Model as a backbone for integrated AV safety and security analysis.

shown in Figure 6. The top-level undesired event is localization failure, which uses LIDAR readings in combination with other sensors for AV localization. Thus, localization failure could happen if either LIDAR or other sensors fail. LIDAR failure can be further decomposed into electrical, LIDAR component, or LIDAR communication failures, as shown in Figure 6. At the end of fault tree analysis, failures are added to the Six-Step Model. Due to space limitations, only a high-level LIDAR failure is shown in Figure 5.

STPA can be used to complement AV safety analysis and help identify failures, not captured in fault trees, as described in Section II-C. Figure 7 depicts a high-level control structure for a typical AV, which is a prerequisite in STPA for identifying the inadequate controls that could result in hazards or hazardous events. The arrows shown in Figure 7 signify the control relationships between the components. For example, the sensors (e.g., LIDAR) send their readings to the computer for computation purposes, and subsequently the computer commands the ECUs for manipulating the vehicle's motion accordingly. Each control can be evaluated in four different ways (i.e., not provided, incorrectly provided, untimely provided, and stopped/applied too soon/long). For instance, if the LIDAR readings are not provided, then the localization, which is performed by the computer and is dependent on LIDAR readings, is likely to fail. A corresponding safety constraint would be "The computer must always receive LIDAR readings". Then, how each inadequate control could occur is to be identified. For example, the computer receives no data from LIDAR because it is disconnected from the Ethernet. Hence, the corresponding safety constraint would be "LIDAR must be connected to the computer at all times". Certainly, more low-level control structure diagrams could be drawn to show more explicit interaction between the components as well as their corresponding sub-components; thus, one can derive more explicitly the unsafe control actions and their causal scenarios (failures) and hence the safety constraints. At the end of integrated safety analysis, the failures and safety countermeasures, as well as the updated structure and functions, are added to the Six-Step model.

Attack trees can be used for security analysis, as described in Section II-D. They show attack paths through the system. An example of a LIDAR attack tree is shown in Figure 8. As we can see from Figure 8, an attacker can execute either cyber or direct physical attack on the LIDAR. To execute cyber attack with the goal to alter LIDAR readings, an attacker can use Ethernet, since LIDAR is connected to Ethernet. Two common types of attacks, deception and Denial of Service (DoS) can be performed on sensor readings. Deception attack is used to modify sensor readings, while DoS attack - to prevent on-board computer from timely receiving the readings. Alternatively, an attacker can get access to the on-board computer and modify the LIDAR readings received by on-board computer, as shown in Figure 8. Attacks on LIDAR and security countermeasures are summarized in [34]. The information from the LIDAR attack tree (Figure 8) is added to the Six-Step Model in step (4) (see Figure 5).

As we can see from Figure 5, the main function affected by either the LIDAR failure or attack is the sensing function. Furthermore, there is a strong relationship between LIDAR attack and failure, LIDAR attack is strongly related to Ethernet (i.e., an attacker can attack LIDAR through Ethernet).

To mitigate sensor attacks and failures, it is necessary to provide sensor redundancy and perform sensor fusion. A combination of LIDAR, Radar, and Camera provides good coverage of AV tasks in most of the environmental conditions [33]. Radar is added to the model in step (5) as a safety

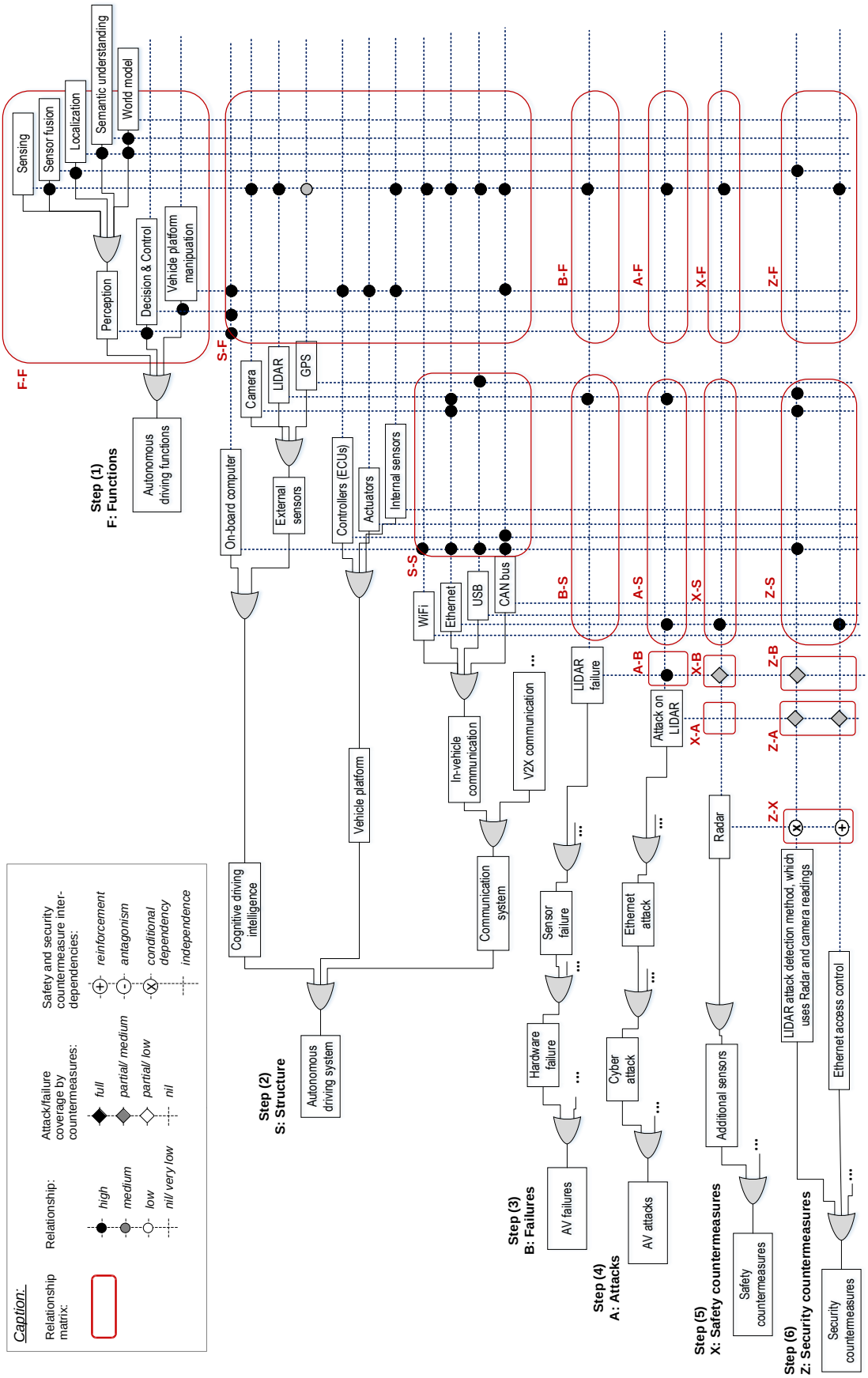


Figure 5. An example of AV Six-Step Model.

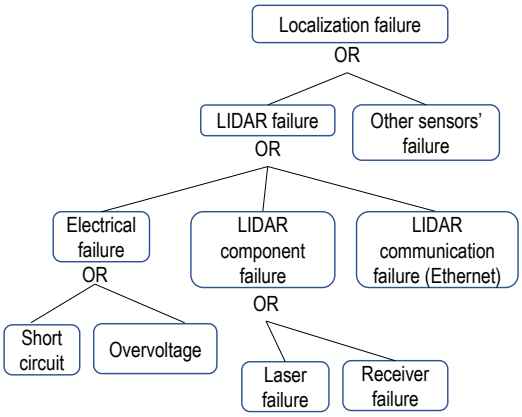


Figure 6. LIDAR failure tree example.

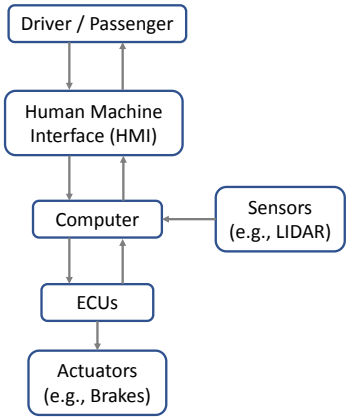


Figure 7. High-level control structure diagram of an AV.

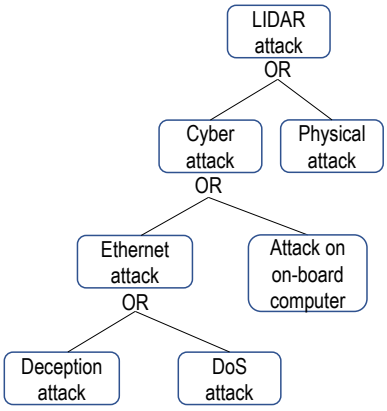


Figure 8. LIDAR attack tree example.

countermeasure. In case of LIDAR failure, Radar and Camera will still be able to perform sensing of the driving environment.

Security countermeasures could include redundancy: multiple LIDARs, or V2X communication to compare measurements of target vehicle with nearby vehicles [34]. However, due to high cost of LIDAR, multiple LIDARs are not considered in this AV. Furthermore, there is no V2X communication in this AV example. If the vehicle had V2X communication, LIDAR attacks could be detected by cross-comparing LIDAR readings of the nearby vehicles.

Various LIDAR attack detection and mitigation methods can be implemented inside on-board computer, e.g., LIDAR attacks can be detected by comparing LIDAR readings to Radar and Camera readings, while shorter or randomized LIDAR scanning interval could help in preventing the attacks [34]. In Figure 5, a security countermeasure, "LIDAR attack detection method, which uses Radar and Camera readings", is added. Additional countermeasure, "Ethernet access control", is used to prevent LIDAR attacks.

Matrices X-A, X-B, Z-A, Z-B, and Z-X are very useful for integrated safety and security analysis. X-B shows that Radar provides partial coverage of LIDAR failure, as Radar cannot fully replace LIDAR. Z-A and Z-B indicate that LIDAR attack detection method will be able to provide coverage not only for LIDAR attacks, but also failures, as it will detect corrupt LIDAR readings, which could happen in either case. Finally, matrix Z-X shows the inter-dependencies between safety and security countermeasures. As we can see from Figure 5, Radar (safety countermeasure) and the LIDAR attack detection method (security countermeasure) share a conditional dependency (denoted by x), i.e., in order to implement the attack detection method, we need a Radar; while Radar and Ethernet access control mechanism reinforce each other.

As the new structural component, Radar, has been added to the model in Step (5), it is necessary to return to the step (2) to include it to AV structural hierarchy and to establish its relationships to the remaining elements of the model.

V. Conclusion and Future Work

In this paper, an approach for integrated AV safety and security analysis is proposed, which is compliant with the international standards SAE J3016, SAE J3061, and ISO 26262. STPA method is integrated into the concept phase of ISO 26262 for acquiring more accurate and detailed lists of functions, failures and safety countermeasures. The proposed method uses the Six-Step Model for achieving and maintaining integration and alignment among safety and security artefacts throughout the entire AV life-cycle. The Six-Step Model incorporates six hierarchies of AVs, namely, functions, structure, failures, attack, safety countermeasures, and security countermeasures. An example of an AV Six-Step Model is included to demonstrate the usefulness of the proposed approach.

Future work will include the refinement of the proposed approach to facilitate its application in industry and the use by other researchers. Furthermore, we are currently extending the proposed approach for application to transportation system (system-of-systems) level.

REFERENCES

[1] G. Sabaliauskaite and J. Cui, "Integrating Autonomous Vehicle Safety and Security," in Proceedings of the 2nd International Conference on Cyber-Technologies and Cyber-Systems (CYBER) November 12–16, 2017, Barcelona, Spain. IARIA, Nov. 2017, pp. 75–81, ISBN: 978-1-61208-605-7.

[2] G. Sabaliauskaite and A. P. Mathur, Aligning Cyber-Physical System Safety and Security. Cham: Springer International Publishing, 2015, pp. 41–53. [Online]. Available: https://doi.org/10.1007/978-3-319-12544-2_4

[3] L. Pièrre-Cambacédès and M. Bouissou, "Cross-fertilization between safety and security engineering," Reliability Engineering & System Safety, vol. 110, 2013, pp. 110 – 126.

[4] SAE J3016: Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. SAE International, Sep. 2016.

[5] ISO26262-2:2011, Road Vehicles – Functional Safety – Part2: Management of Functional Safety. International Organization of Standardization, ISO, 2011.

[6] F. Warg et al., Defining Autonomous Functions Using Iterative Hazard Analysis and Requirements Refinement. Cham: Springer International Publishing, 2016, pp. 286–297. [Online]. Available: https://doi.org/10.1007/978-3-319-45480-1_23

[7] A. Abdulkhaleq et al., "A systematic approach based on stpa for developing a dependable architecture for fully automated driving vehicles," Procedia Engineering, vol. 179, no. Supplement C, 2017, pp. 41 – 51.

[8] Q. V. E. Hommes, "Assessment of safety standards for automotive electronic control systems," 2016, (Report No. DOT HS 812 285).

[9] A. Abdulkhaleq, S. Wagner, D. Lammering, H. Boehmert, and P. Blueher, "Using STPA in compliance with ISO 26262 for developing a safe architecture for fully automated vehicles," CoRR, vol. abs/1703.03657, 2017.

[10] A. Mallya, V. Pantelic, M. Adedjouma, M. Lawford, and A. Wassyng, Using STPA in an ISO 26262 Compliant Process. Cham: Springer International Publishing, 2016, pp. 117–129.

[11] N. G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety.

[12] SAE J3061: Cybersecurity Guidebook for Cyber-Physical Vehicle Systems. SAE International, Jan. 2016.

[13] ISO/SAE AWI 21434, Road Vehicles – Cybersecurity engineering. International Organization of Standardization, ISO, Under development.

[14] Automated Driving Systems 2.0. A Vision for Safety. National Highway Traffic Safety Administration, NHTSA, U.S. Department of Transportation, Sep. 2017.

[15] G. Sabaliauskaite, S. Adepu, and A. Mathur, "A six-step model for safety and security analysis of cyber-physical systems," in the 11th International Conference on Critical Information Infrastructures Security (CRITIS), Oct. 2016.

[16] G. Sabaliauskaite and S. Adepu, "Integrating six-step model with information flow diagrams for comprehensive analysis of cyber-physical system safety and security," in 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), Jan. 2017, pp. 41–48.

[17] S. Behere and M. Törngren, "A functional reference architecture for autonomous driving," Inf. Softw. Technol., vol. 73, no. C, May 2016, pp. 136–150. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2015.12.008

[18] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," IEEE Micro, vol. 35, no. 6, Nov. 2015, pp. 60–68.

[19] P. Bhavsar, P. Das, M. Paugh, K. Dey, and M. Chowdhury, "Risk analysis of autonomous vehicles in mixed traffic streams," Transportation Research Record: Journal of the Transportation Research Board, vol. 2625, 2017, pp. 51–61.

[20] L. Pièrre-Cambacédès and M. Bouissou, "Modeling safety and security interdependencies with bdmp (boolean logic driven markov processes)," in 2010 IEEE International Conference on Systems, Man and Cybernetics, Oct. 2010, pp. 2852–2861.

[21] I. Ibarra, S. Hartley, S. Crozier, and D. Ward, "Iso 26262 concept phase safety argument for a complex item," 2012.

[22] C. Bergenhem, R. Johansson, A. Söderberg, J. Nilsson, J. Tryggvesson, M. Törngren, and S. Ursing, "How to reach complete safety requirement refinement for autonomous vehicles," in CARS 2015 - Critical Automotive applications: Robustness & Safety, M. Roy, Ed., Paris, France, Sep. 2015.

[23] I. Habli, I. Ibarra, R. Rivett, and T. Kelly, "Model-based assurance for justifying automotive functional safety," 2010.

[24] "Standardization efforts on autonomous driving safety barely under way," The Hansen Report on Automotive Electronics, 2017.

[25] R. Johansson, "Efficient Identification of Safety Goals in the Automotive E/E Domain," in 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016), Toulouse, France, Jan. 2016.

[26] J. Chen, Y. Lu, S. Zhang, and P. Tang, "Stpa-based hazard analysis of complex uav system in take-off," in The 3rd International Conference on Transportation Information and Safety, Wuhan, P. R. China, 2015.

[27] T. Ishimatsu, N. G. Leveson, J. Thomas, M. Katahira, Y. Miyamoto, and H. Nakao, "Modeling and hazard analysis using stpa," in Proceedings of the 4th IAASS Conference, Making Safety Matter, Huntsville, Alabama, USA, May 2010.

[28] P. Asare, J. Lach, and J. A. Stankovic, "Fstpa-i: A formal approach to hazard identification via system theoretic process analysis," in 2013 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS), Philadelphia, PA, USA, 2013.

[29] A. Abdulkhaleq and S. Wagner, "Experiences with applying stpa to software-intensive systems in the automotive domain," 2013.

[30] A. Abdulkhaleq, S. Wagner, and N. Leveson, "A comprehensive safety engineering approach for software-intensive systems based on stpa," Procedia Engineering, vol. 128, Oct. 2015, pp. 2 – 11.

[31] B. Schneier, Attack Trees. Wiley Publishing, Inc., Indianapolis, Indiana, 2015, in Book, Secrets and Lies.

[32] Z. Ma and C. Schmittner, "Threat modeling for automotive security analysis," Advanced Science and Technology Letters, vol. 139, 2016, pp. 333–339.

[33] "Beyond the Headlights: ADAS and Autonomous Sensing," 2016, URL: http://woodsidecap.com/wp-content/uploads/2016/12/20160927-Auto-Vision-Systems-Report_FINAL.pdf [accessed: 2018-05-08].

[34] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote Attacks on Automated Vehicles Sensors: Experiments on Camera and LiDAR," in Black Hat Europe, Nov. 2015.

System Integrity Monitoring for Industrial Cyber Physical Systems

Rainer Falk and Steffen Fries
Corporate Technology
Siemens AG
Munich, Germany
e-mail: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Cyber physical systems are technical systems that are operated and controlled using information and communication technology. Protecting the integrity of cyber physical systems is a highly important security objective to ensure the correct and reliable operation and to ensure high availability. A comprehensive protection concept of the system integrity involves several axes: the component level ranging from sensors/actuator devices up to control and supervisory systems, planning and configuration management, and the system life cycle. It allows detecting integrity violations on system level reliably by analyzing integrity measurements from a multitude of independent integrity sensors, capturing and analyzing integrity measurements of the physical world, on the field level, and of control and supervisory systems. Trusted sensors can be used as add-on in existing industrial automation and control systems to allow for cross-checking with sensor measurements of the control system.**

Keywords–system integrity, device integrity; cyber physical systems; Internet of Things, embedded security; cyber security.

I. INTRODUCTION

With ubiquitous machine-oriented communication, e.g., the Internet of Things and interconnected cyber physical systems (CPS), the integrity of technical systems is becoming an increasingly important security objective. This paper is an extended version of [1] that describes an approach for enhanced integrity monitoring of industrial automation and control systems.

Information technology (IT) security mechanisms have been known for many years, and are applied in smart devices (Internet of Things, Cyber Physical Systems, industrial and energy automation systems, operation technology) [2]. Such mechanisms target source authentication, system and communication integrity, and confidentiality of data in transit or at rest. System integrity takes a broader approach where not only the integrity of individual components (device integrity) and of communication is addressed, but where integrity shall be ensured at the system level of interconnected devices. This purpose is in particular challenging for dynamically changing cyber physical systems, that come with the Industrial Internet of Things (IIoT) and Industrie 4.0. Cyber systems will become more open and dynamic to support flexible production down to lot size 1 (plug-and-work reconfiguration of manufacturing equipment), and flexible adaptation to changing needs (market demand, individualized products).

The flexibility starts on the device level, where smart devices allow for upgrading and enhancing device functionality by downloadable apps. But also the system of interconnected machines is reconfigured according to changing needs. Examples are Software Defined Networks (SDN) enabling a fast reconfiguration of the communication infrastructure to adapt flexibly to the communication needs. Another example relates to manufacturing systems (e.g., robots) in industrial automation systems, where smart tools are attached to a robot that in turn feature also a local communication network connecting to the robots network. These tools may be connected only temporarily.

Classical approaches for protecting device and system integrity target at preventing any changes, and compare the current configuration to a fixed reference policy. More flexible approaches are needed to protect integrity for flexibly reconfigurable and self-adapting CPSs.

This paper describes an integrated, holistic approach for ensuring CPS integrity. After summarizing system security requirements coming from relevant industrial security standard IEC 62443 [2] in Section II, an overview for protecting device integrity and system integrity is described in Sections III and IV. The presented approach for integrity monitoring is an extensible framework to include integrity information from IT-based functions and the physical world of a CPS. This allows integrating integrity information from the digital and the physical world. Trusted physical integrity sensors can be installed as add-on to existing automation and control systems, see Section V. Using one-way gateways to extract integrity monitoring information from closed control networks, while ensuring freedom from interference, is described in Section VI. A new approach for integrity monitoring of encrypted communications is described in Section VII. An approach for evaluation in an operational security management setting is outlined in Section VIII. Related work is summarized in Section IX, and Section X concludes the paper.

II. SYSTEM INTEGRITY REQUIREMENTS

Protecting industrial automation control systems against intentional attacks is increasingly demanded by operators to ensure a reliable operation, and also by regulation. This section gives an overview on industrial security, and on the main relevant industrial security standard IEC 62443 [2] and integrity security requirements.

A. Industrial Security

Industrial security is called also Operation Technology security (OT Security), to distinguish it from general information technology (IT) security. Industrial systems have not only different security requirements compared to general IT systems, but come also with specific side conditions that prevent that security concepts established in the IT domain can be applied directly in an OT environment. For example, availability and integrity of an automation system have often a higher priority than confidentiality. High availability requirements, different organization processes (e.g., yearly maintenance windows), and required certifications may prevent the immediate installations of updates.

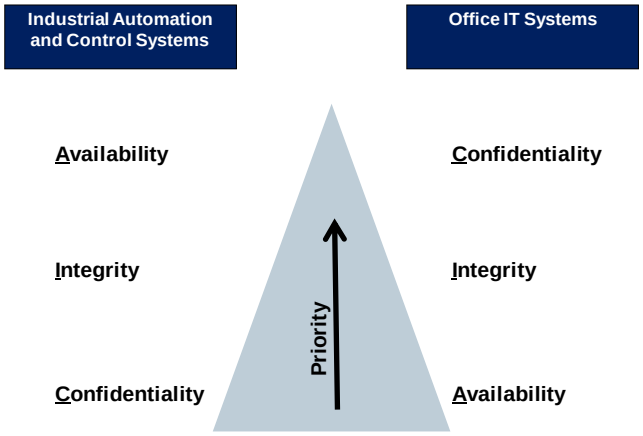


Figure 1. The CIA Pyramid [3]

The three basic security requirements are confidentiality, integrity, and availability. They are also named "CIA" requirements. Figure 1 shows that in common IT systems, the priority is "CIA". However, in automation systems or industrial IT, the priorities are commonly just the other way round: Availability has typically the highest priority, followed by integrity. Confidentiality is often no strong requirement for control communication, but may be needed to protect critical business know-how. Shown graphically, the CIA pyramid is inverted (turned upside down) in many automation systems.

Specific requirements and side conditions of industrial automation systems like high availability, planned configuration (engineering info), long life cycles, unattended operation, real-time operation, and communication, as well as safety requirements have to be considered when designing a security solution. The security requirements, for instance defined in IEC 62443, can be mapped to different automation domains, including energy automation, railway automation, building automation, process automation.

Defined security measures range from security processes, personal and physical security, device security, network security, and application security. No single security technology alone is adequate, but a combination of security measures addressing prevention, detection, and reaction to incidents is required ("defence in depth").

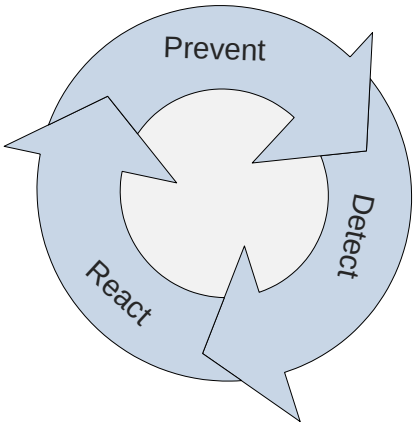


Figure 2. Prevent Detect React Cycle

Also, overall security has to address the areas prevent, detect, and react, see Figure 2. It is not sufficient to only define measures to protect against attacks. The capability has also foreseen to detect attacks, and to define measures to react adequately once an attack has been detected.

B. Overview IEC 62443 Industrial Security Standard

The international industrial security standard IEC 62443 [2] is a security requirements framework defined by the International Electrotechnical Commission (IEC). It is applied successfully in different automation domains, including factory and process automation, railway automation, energy automation, and building automation.. The standard specifies security for industrial automation and control systems (IACS) and covers both, organizational and technical aspects of security. Specifically addressed is the setup of a security organization and the definition of security processes as part of an information security management system (ISMS) based on already existing standards like ISO 27002. Furthermore, technical security requirements are specified distinguishing different security levels for industrial automation and control systems, and also for the used components. The standard has been created to address the specific requirements of industrial automation and control systems. In the set of corresponding documents, security requirements are defined, which target the solution operator and the integrator but also the product manufacturer.

As shown in Figure 3, different parts of the standard are grouped into four clusters covering

- common definitions and metrics;
- requirements on setup of a security organization (ISMS related, comparable to ISO 27001 [4]), as well as solution supplier and service provider processes;
- technical requirements and methodology for security on system-wide level, and
- requirements on the secure development lifecycle of system components, and security requirements to such components at a technical level.

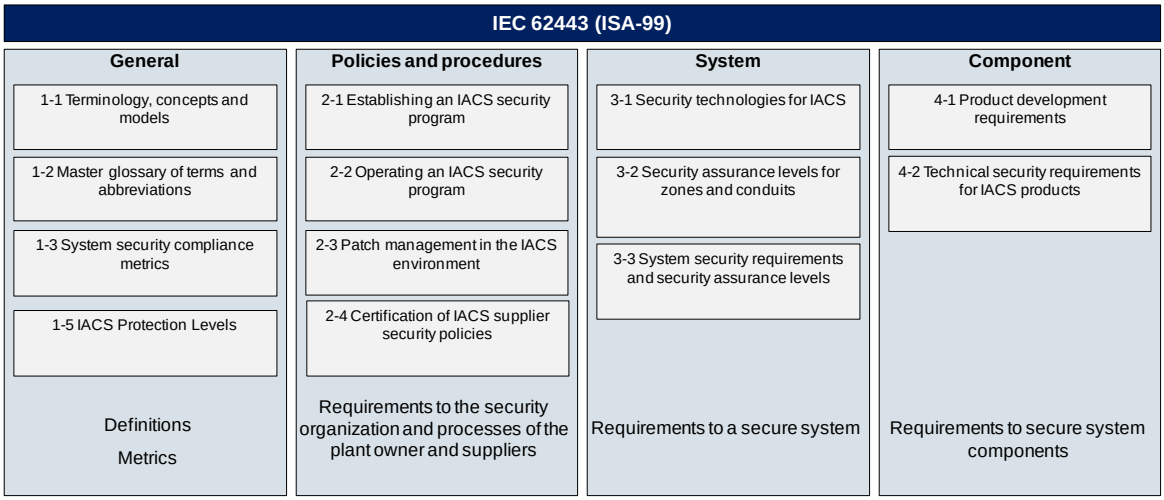


Figure 3. IEC 62443 Industrial Security Standard – Overview

Figure 4 below gives an overview on which parts of IEC 62443 are relevant for the different roles. The operator of an automation system operates the automation and control system that has been integrated by the system integrator, using components of product suppliers.

According to the methodology described in IEC 62443-3-2, a complex automation system is structured into zones that are connected by and communicate through so-called "conduits" that map for example to the logical network protocol communication between two zones. Moreover, this document defines Security Levels (SL) that correlate with the strength of a potential adversary as shown in Figure 5 below. To reach a dedicated SL, the defined requirements have to be fulfilled. IEC 62443 part 3.3 defines system security requirements. It does help to focus only on certain facets of security. The security requirements defined by IEC 62443 part 3.3 help to ensure that all relevant aspects are addressed.

Part 3-3 of IEC 62443 [5] defines seven foundational requirements group specific requirements of a certain category:

– FR 1 Identification and authentication control
– FR 2 Use control
– FR 3 System integrity
– FR 4 Data confidentiality
– FR 5 Restricted data flow
– FR 6 Timely response to events
– FR 7 Resource availability

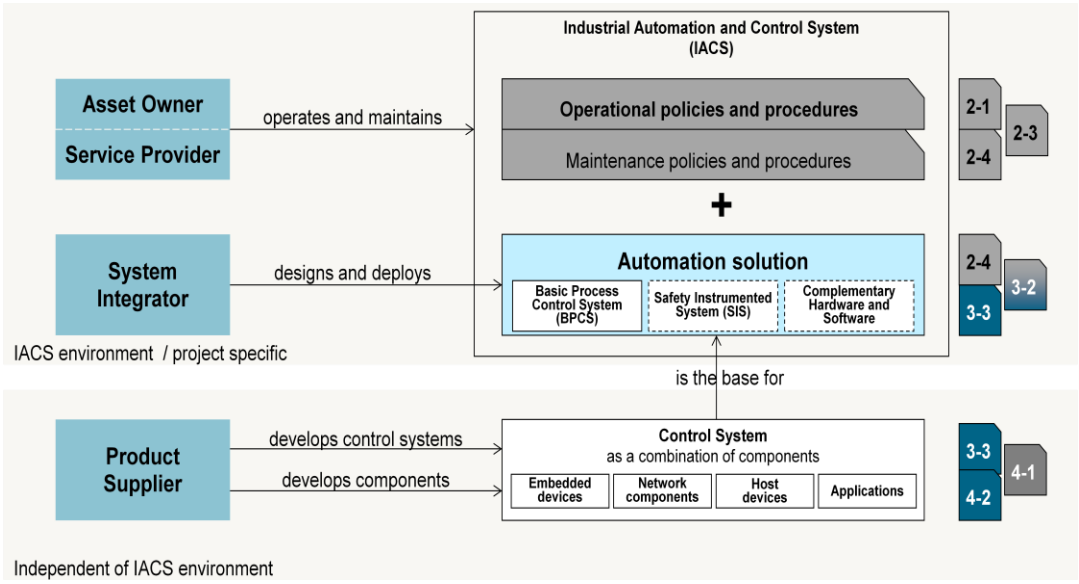


Figure 4. Application of IEC 62443 parts by different roles

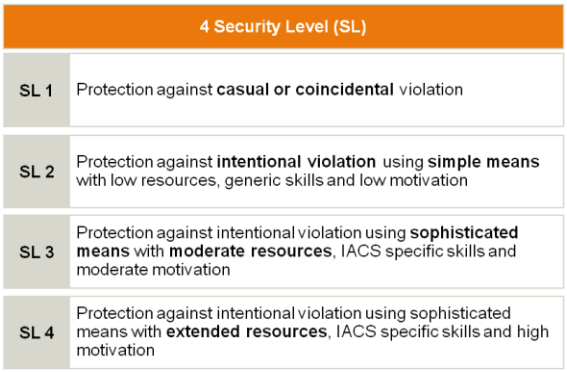


Figure 5. IEC 62443 defined Security Level

For each of the foundational requirements there exist several concrete technical security requirements (SR) and requirement enhancements (RE) to address a specific security level. In the context of communication security, these security levels are specifically interesting for the conduits connecting different zones.

Four Security Levels (SL1, SL2, SL3, SL4) are defined that correlate with the strength of a potential adversary as shown in Figure 5. To reach a dedicated security level, the requirements (SR) and potential requirement enhancements (RE) defined for that security level have to be fulfilled. The standard foresees that a security requirement can be addressed either directly, or by a compensating countermeasure. The concept of compensating countermeasures allows to reach a certain security level even if some requirements cannot be implemented directly, e.g., as some components do not support the required technical features. This approach is in particular important for existing industrial automation and control systems, so called "brown-field installations", as existing equipment can be continued to be used.

The security level of a zone or a conduit (a conduit connects zones) is more precisely a security level vector with seven elements. The elements of the vector designate the security level for each foundational requirement. This allows defining the security level specific for each foundational requirement. If, e.g., confidentiality is no security objective within a zone, the security level element corresponding to FR4 "Data confidentiality" can be defined to be SL1 or even none, although SL3 may be required for other foundational requirements (e.g., for FR1, FR2, and FR3). So, the resulting security level vector for a zone could be SL=(3,3,3,1,2,1,3) or SL=(2,2,2,0,1,1,0).

Different types of SL vectors are distinguished, depending on the purpose:

– SL-T: A target security level vector is defined by the IACS operator based on his risk assessment, defining which security level shall be achieved by each zone and conduit.

– SL-A: The achieved security level vector designates the current status, i.e., the security level that is actually achieved by each zone and conduit. In particular for brown-field installations, it is common that a targeted security level cannot be set-up immediately. The gap between the targeted and the actually achieved security level can be made transparent.

– SL-C: The security level capability describes the reachable security level a component is capable of, if properly configured, without additional compensating counter measures employed. This also means that depending on the SL-T not all security features of a component may be used in certain installations.

C. IEC 62443 Integrity Requirements

One of the seven foundational security requirements defined in Part 3-3 of IEC 62443 [5], targets specifically integrity. Integrity requirements cover in particular the following areas:

– Overall system integrity

– Communication integrity

– Device integrity

The following examples from IEC 62443-3-3 [5] illustrate some of the integrity-related requirements:

– FR3, SR3.1 Communication integrity: "The control system shall provide the capability to protect the integrity of transmitted information".

– FR3, SR3.4 Software and information integrity: "The control system shall provide the capability to detect, record, report and protect against unauthorized changes to software and information at rest."

– FR3, SR3.8 Session integrity: "The control system shall provide the capability to protect the integrity of sessions. The control system shall reject any usage of invalid session IDs."

– FR5, SR 5.2 Zone boundary protection: "The control system shall provide the capability to monitor and control communications at zone boundaries to enforce the compartmentalization defined in the risk -based zones and conduits model."

Corresponding to the system requirements defined in IEC 62443-3-3, also security requirements are defined for individual components (devices). These requirements are defined by IEC 62443 part 4-2 [6] that is currently specified. Different types of components are distinguished, which are "software application", "embedded device", "host device", and "network device".

D. Practical Application of IEC 62443

The standard IEC 62443 has been applied successfully by operators, integrators, and manufacturers in various projects. It is common that documentation and technical designs of real-world deployments are not made public or shared with competitors. However, some examples for applying IEC 62443 are available publicly:

A publication of applying the IEC 62443 standard to the Ukrainian power plant gives some insight concerning how the standard can be applied in a concrete setting [7]. In particular, it shows that a sound, comprehensive security concept is needed that covers security requirements broadly and at a consistent level. The German industrial association "Zentralverband Elektrotechnik- und Elektronikindustrie e.V." (ZVEI) published an overview document on IEC 62443 that includes a simple example, showing the application to a simplified automation system [8].

For the integration of decentralized energy resources into the digital grid, the standard IEC 62351-12 [9] maps the security solution specified for decentralized energy resources to the security requirements in IEC 62443-3-3, arguing that the security requirements are addressed comprehensively.

III. PROTECTING DEVICE INTEGRITY

The objective of device integrity is to ensure that a (single) device is not manipulated in an unauthorized way. This includes the integrity of the device firmware, of the device configuration, but also the physical integrity. Main technologies to protect device integrity are (see Figure 6):

- Secure boot: A device loads at start-up only unmodified, authorized firmware.
- Measured boot: The loaded software modules are checked at the time they are loaded. Usually, a cryptographic hash value is recorded in a platform configuration register of a hardware of firmware trusted platform module (TPM) [10][11]. The configuration information can be used to grant access to keys, or it can be attested towards thirds parties.
- Protected firmware update: When the firmware of a device is updated, the integrity and authenticity of the firmware update is checked. The firmware update image can be digitally signed.
- Application whitelisting: Only allowed, known applications can be started on a device. A whitelist defines which application binaries can be started.
- Runtime integrity checks: During operation, the device performs self-test of security functionality and integrity checks to verify whether it is operating as expected. Integrity checks can verify the integrity of files, configuration data, software modules, and runtime data as the process list, i.e., the list of currently executed processes.
- Process isolation, kernel-based mandatory access control (MAC): Hypervisors or kernel MAC systems like SELinux [12], AppArmor [13], or SMACK [14], can be used to isolate different classes of software (security domains). An attack or malfunction one security domain does not affect other security domains on the same device.
- Tamper evidence, tamper protection: The physical integrity of a device can be protected, e.g., by security seals or by tamper sensors that detect opening or manipulation of the housing.
- Device integrity self-test: A device performs a self-test to detect failures. The self-test is performed typically during startup and is repeated regularly during operation. Operation integrity checks: measurements on the device can be compared with the expected behavior in the operative environment. An example is the measurement of connection attempts to/from the device, based on parameters of a Management Information Base (MIB).

The functionality of some devices can be extended by extensions (App). Here, the device integrity has to cover also the App runtime environment: Only authorized, approved apps can be downloaded and installed. Apps are isolated during execution (managed runtime environment, hypervisor, and container). Host-based intrusion detection systems (HIDS) as, e.g., OSSEC [15] can be used for runtime integrity checks on devices, detecting unauthorized changes to the file system.

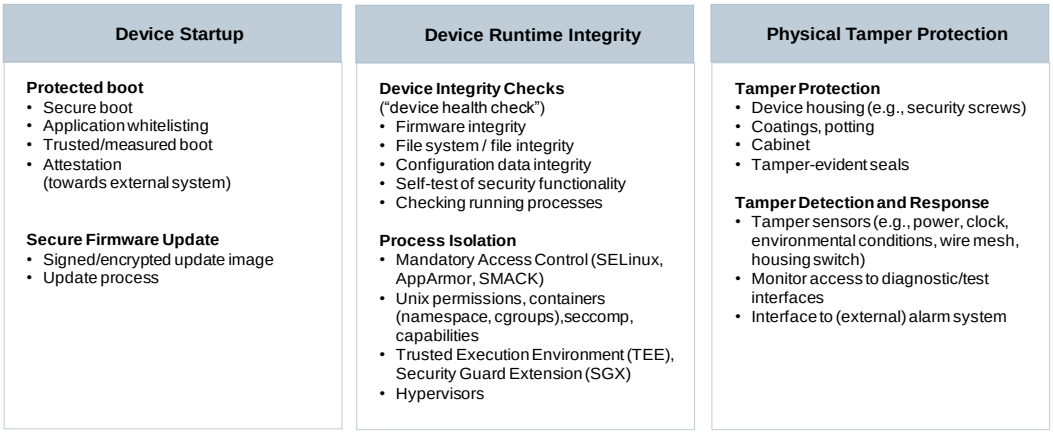


Figure 6. Device Integrity Security Technologies

The known approaches to protect device integrity focus on the IT-related functionality of a device (with the exception of tamper protection). Also, a strong tamper protection is not common on device level. The main protection objective for device integrity shall ensure that the device's control functionality operates as designed. However, the integrity of input/output interfaces, sensors, and actuators are typically out of scope. In typical industrial environments, applying a strong tamper protection to the each control device, sensor, and actuator would not be economically feasible. Therefore, protecting device integrity alone would be too limited to achieve the goal of protection the integrity of an overall CPS.

IV. System Integrity Monitoring

The next level of integrity is on the system level comprising a set of interconnected devices. The main approaches to protect system integrity are collecting and analyzing information on system level:

- Device inventory: Complete and up-to-date list of installed devices (including manufacturer, model, serial number version, firmware version, current configuration, installed software components, location)
- Centralized Logging: Devices provide log data, e.g., using Open Platform Communication Unified Architecture (OPC UA) protocol [16], SNMP [17], or syslog protocol [18], to a centralized logging system.
- Runtime device integrity measurements: A device integrity agent provides information gathered during the operation of the device. It collects integrity information on the device and provides it for further analysis. Basic integrity information are the results of a device self-test, and information on the current device configuration (firmware version, patches, installed applications, configuration). Furthermore, runtime information can be gathered and provided for analysis (e.g., process list, file system integrity check values, partial copy of memory).
- Network monitoring: The network communication is intercepted, e.g., using a network tap or a mirror port of a network switch. A challenge is the fact that network communication is increasingly encrypted.
- Physical Automation process monitoring: Trusted sensors provide information on the physical world that can be used to cross-check the view of the control system on the physical world. Adding trusted sensors to existing installation allows for a smooth migration from legacy systems to systems providing integrated trusted sensors.
- Physical world integrity: Trusted sensors (of physical world), integrated monitoring of embedded devices and IT-based control systems, and of the technical process allow now quality of integrity monitoring as physical world and IT world are checked together.

The captured integrity information can be used for runtime integrity monitoring to detect integrity violations in real-time. Operators can be informed, or actions can be triggered automatically. Furthermore, the information is archived for later investigations. This allows that integrity violations can be detected also later with a high probability, so that corresponding counter-measures can be initiated (e.g., plan for an additional quality check of produced goods). The integrity information can be integrated in or linked to data of a production management system, so that it can be investigated under which integrity conditions certain production steps have been performed. Product data is enhanced with integrity monitoring data related to the production of the product.

A. System Overview

Agents on the system components acting as integrity sensors collect integrity information and optionally determine an integrity attestation of the collected information. To allow for flexibility in CPS, the approach puts more focus on monitoring integrity and acting when integrity violations are detected, than on preventing any change that has not been pre-approved by a static policy.

The approach is based on integrity sensors that provide integrity related measurements. An intelligent analysis platform performs data analysis (e.g., statistical analysis, big data analysis, artificial intelligence) and triggers suitable respondence actions (e.g., alarm, remote wipe of a device, revocation of a device, stop of a production site, planning for additional test of manufactured goods).

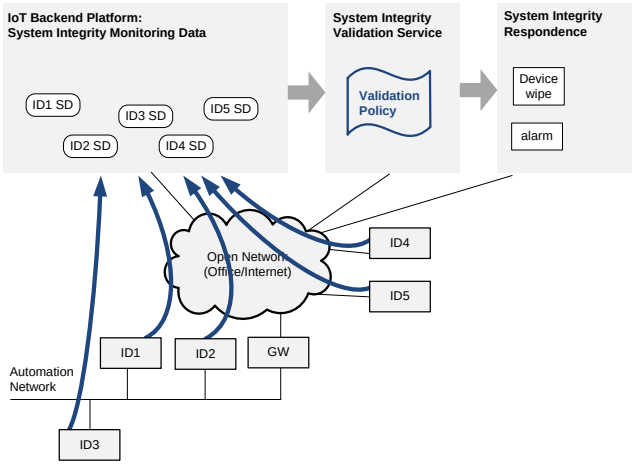


Figure 7. Validation of Device Monitoring Data

Figure 7 shows an example for an IoT system with IoT devices (ID1, ID2, etc.) that communicate with an IoT backend platform. The devices provide current integrity monitoring information to the backend platform. The devices can be automation devices that include integrity measurement functionality, or dedicated integrity sensor devices. The device monitoring system itself has to be protected against attacks itself, following the industrial security standard IEC 62443.

An integrity data validation service checks the obtained integrity measurement data for validity using a configurable validation policy. If a policy violation is detected, a corrective action is triggered: For example, an alarm message can be displayed on a dash board. Furthermore, an alarm message can be sent to the IoT backend platform to terminate the communication session of the affected IoT device. Moreover, the device security service can be informed so that it can revoke the devices access permissions, or revoke the device authentication credential.

B. Integrity Sensors

The integrity monitoring framework foresees to include a variety of integrity measurements. Depending on the specific application scenario, meaningful integrity sensors can be deployed. Depending on the evolving needs, additional sensors can be deployed as needed.

- Physical world (technical process)
- Physical world (alarm systems, access control systems, physical security as, e.g., video surveillance)
- Device world (malware, device configuration, firmware integrity)
- IT-based control systems (local, cloud services, edge cloud)
- Infrastructure (communication networks)

Flexible extension with additional integrity sensors (even very sophisticated as, e.g., monitoring power fingerprint). The described approach is open to develop and realize sophisticated integrity measurement sensors. So the solution is design to allow evolution and innovation. Integrity sensors have to be protected against attacks so that they provide integrity measurements reliably.

C. Integrity Verification

The integrity monitoring events are analyzed using known data analysis tools. The system integrity can be monitored both online. In industrial environments, it is also important to have reliable information about the system integrity of a production system for the time period during which a certain production batch was performed. This allows performing the verification also afterwards to check whether during a past production batch integrity-violations occurred.

The final decision whether a certain configuration is accepted as correct is up to human operators. After reconfiguration, or for a production step, the configuration is to be approved. The approval decision can be automated according to previously accepted decisions, or preconfigured good configurations).

As integrity measurements are collected from a multitude of integrity sensors, integrity attacks can be detected reliably. Even if some integrity sensors should be disabled or manipulated to provide malicious integrity measurements, still other integrity sensors can provide integrity information that allows detecting the integrity violation. Checking integrity using measurements from independent integrity sensors and on different levels (physical level, field devices, control and supervisory systems) allows detecting integrity violations by checking for inconsistencies between independent integrity measurements.

V. Trusted Physical Integrity Sensor

A specific approach described in Section IV is the cross-checking of regular sensor measurements used by the industrial automation and control system with independently obtained sensor measurements that are provided by a trusted sensor node. Trusted sensor nodes can be added as add-on security sensors to existing industrial and automation control systems, providing an additional layer of integrity protection. Those trusted sensors and the corresponding analysis algorithms can be updated flexibly and independently from the actual industrial automation and control system. The specific security measures protecting trusted sensors do not interfere with real-time communication requirements or regulatory certification requirements of the actual automation system.

Trusted sensors are used in specific applications as in smart metering to obtain trustworthy information on consumed energy, or for digital tachographs to obtain trustworthy information on driving time and speed for trucks. However, such security-oriented solutions are quite complex, so that it is not realistic to assume that such solutions replace all sensors (and actuators) in industrial automation and control system. Therefore, the intention is to augment existing automation and control system solutions with specific additional trusted sensors. Such trusted sensors can feature specific security measures to provide trusted, integrity protected sensor measurements for consistency and plausibility checking:

- Physical protection (tamper protection): Trusted sensor nodes can be realized with tamper protected housing, and special tamper protected security controllers. Additional tamper sensors integrated with the trusted sensor can detect when the trusted sensor is relocated from his mounting point.
- Cryptographically protected communication: The communication can be protected using common cryptographic protocols, e.g., the Internet security protocol (IPsec) [19] or the transport layer security (TLS) [20] protocol.
- Source authentication: The trusted sensor node can authenticate to other parts of the system to vouch for its credibility.

VI. Freedom of Interference

When integrating trusted sensors in a real-time critical or safety-critical industrial automation and control system, it has to be ensured reliably that the trusted sensors cannot interfere with the control operations. This can be achieved by separating the control network and the integrity monitoring network physically, or at least logically using virtual networks.

However, integrity monitoring information has to be provided also from closed, isolated control networks. Actually, the most critical control systems are often realized

in isolated networks. Actually, IEC 62443 system security requirement SR5.1(3) requires for security level SL4 to both logically and physically separate critical control networks. A physical isolation goes beyond a physical segmentation of control networks that is already required for security level SL2.

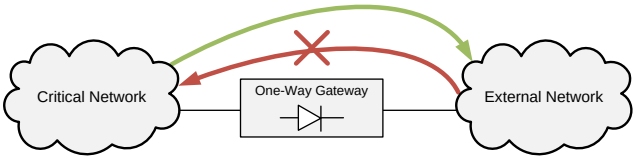


Figure 8. Unidirectional One-Way Gateway

Freedom of interference for network communication can be realized using special one-way gateways [21], as depicted in Figure 8. A one-way gateway ensures that a data communication can take place only in one direction, in particular from a critical control network to an external network. It is not possible to influence or even modify the control communication within the critical control network from the external network, as required by safety authorities and regulator. A data capturing unit (DCU) provides for passive, unidirectional data capture with no interference to the monitored network.

VII. Integrity Monitoring of Encrypted Communications

A specific part of monitoring the system integrity is the network communication. However, network communication is encrypted more-and-more, e.g., using the Transport Layer Security (TLS) protocol [20]. In contrast to earlier versions of the TLS protocol, the most recent version TLS1.3 [22], currently under development, supports only cipher-suites realizing authenticated encryption. Both confidentiality and integrity/authenticity of user communication is protected. No cipher suite providing integrity-only protection is supported by TLS version 1.3, anymore. So, only basic IP header data can be analyzed. This is not sufficient for integrity monitoring of TLS-protected industrial control communication.

A protocol specific solution to enable monitoring of encrypted communication channels by trusted middleboxes is provided by mcTLS [23]. With mcTLS, trusted middleboxes can be incorporated into a secure sessions established between a TLS Client and a TLS Server. Figure 9 shows the basic principle of mcTLS. A TLS authentication and key agreement is performed between a TLS client and a TLS server. As part of the handshake, the TLS client indicates those TLS middleboxes that shall be incorporated within the TLS session. As part of the authentication and key agreement between client and server the middleboxes are incorporated into the message exchange to also possess the (encrypted) key material of the established TLS session using the extension mechanism of TLS.

The basic approach is to perform an enhanced handshake involving middleboxes into the handshake phase of TLS, see Figure 9.

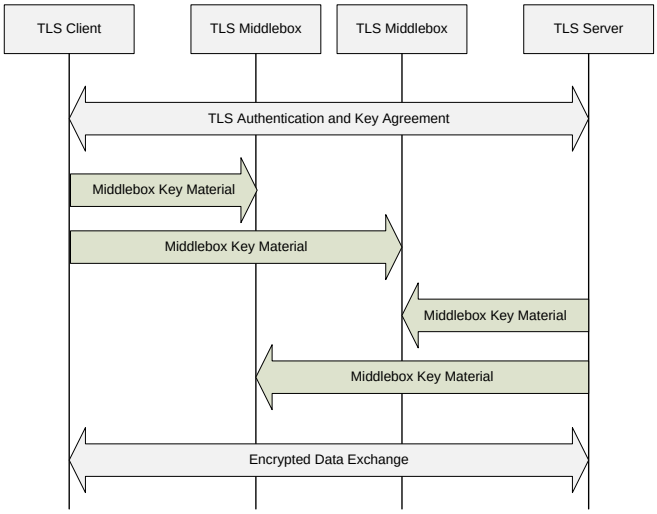


Figure 9. Multi-Context TLS

Specifically, middleboxes are authenticated during the handshake and thus known to both communicating ends. Moreover, each side is involved in the generation of the session key, which is also provided to the middlebox. There is also additional keying performed for the exchange of pure end-to-end keys. Specific key material known to the middlebox is used to decrypt the traffic and check the integrity. The end-to-end based keys are used to protect integrity end-to-end. The latter approach ensures that the middlebox can only read and analyze the content of the communication in the TLS record layer, but any change done by the middlebox is detected by an invalid end-to-end integrity check value. This approach has the advantage that it provides an option to check the associated security policy during the session setup and at the same time monitor traffic as an authorized component. The drawback is that the solution focuses solely on TLS and cannot be applied to other protocols without changes.

The TLS-variant mcTLS allows middleboxes to analyze the TLS-protected communication, e.g., to detect potential security breaches This approach enables communication checking the contents of the communication session without breaking end-to-end security. Hence, with mcTLS, the contents of encrypted data communication, in particular of industrial control communication, can be checked.

Note that mcTLS is only one potential solution for allowing monitoring of encrypted communication. There are further approaches currently being discussed in different standardization groups. Hence, mcTLS is used here just as example as it provides for authenticated and authorized middleboxes, visible and known to the communicating entities.

VIII. EVALUATION

The security of a cyber system can be evaluated in practice in various approaches and stages of the system's lifecycle:

- Threat and risk analysis (TRA) of cyber system
- Checks during operation to determine key performance indicators (e.g., check for compliance of device configurations).
- Security testing (penetration testing)

During the design phase of a cyber system, the security demand is determined, and the appropriateness of a security design is validated using a threat and risk analysis. Assets to be protected and possible threats are identified, and the risk is evaluated in a qualitative way depending on probability and impact of threats. The effectiveness of the proposed enhanced device authentication means can be reflected in a system TRA.

The main evaluation of security tools is performed during secure operation, when as part of an overall operational security management appropriate technologies are deployed that, in combination, reduce the risk to an acceptable level. The new approach presented in this paper provides an additional component, in form of a trusted sensor, integrated into the overall system security architecture that is used to provide additional (secure) measurements to reduce the risk of integrity violations. Compared to existing solutions covering IT-related aspects only, the integrity of the control application and the physical world are interconnected. The solution approach does not intend to have a single technology, but it realizes a system-oriented approach that can evolve as part of the security management life cycle covering prevent, detect, and response, as shown in Figure 2.

Applicability to industrial automation environments of the proposed approach allows for:

- Updatability: integrity monitoring system can be updated independently from control system
- Add-on to existing automation systems (brownfield)
- Freedom of interference (do not invalidate reliable operation or certifications)

IX. RELATED WORK

A security operation center (SOC) is a centralized unit for detecting and handling security incidents. Main functionalities are continued security monitoring reporting, and post-incident analysis [24][25]. Security incident and Event management (SIEM) systems can be used within a SOC to analyze security monitoring data. Compliance management systems support a centralized reporting of server configuration in data centers.

Host-based intrusion detection systems (HIDS) as SAMHAIN [26] and OSSEC [15] analyze the integrity of hosts and report the results to a backend security monitoring system. Network based intrusion detection systems (NIDS) capture the network traffic, e.g., using a network tap or a mirroring port of a network switch, and analyze the traffic. Examples are SNORT [27] and Suricata [28].

Two main strategies can be followed by an intrusion detection system (IDS): Known malicious activities can be looked for (signature based detection), or any change compared to a learned reference network policy is detected (anomaly detection). They can be applied also in industrial automation and control networks. Premaratne, Samarabandu, Sidhu et al. simulated attacks on an energy automation substation and developed an IDS to detect these attacks [29]. The risk of an attack on the energy distribution system of is determined based on the current power consumption. Fovine, Carcano, et al. have proposed a state-based IDS that monitors the cyber-physical state evolution of a supervisory control and data acquisition (SCADA) system [30].

An "automotive thin profile" of the Trusted Platform Module TPM 2.0 has been specified [31]. A vehicle is composed of multiple control units that are equipped with TPMs. A rich TPM manages a set of thin TPMs, so that the vehicle can be represented by a vehicle TPM to the external world. Technical solutions for protecting against tampering of smart meters are described in [32].

Approaches to utilize the context information on the CPS operation, device capabilities, device context to enhance the authentication of a single device, have been described by the authors of this paper in previous work [33]. The effect of an integrity attack on the degradation of a control system has been investigated by Mo and Sinopoli [34].

X. CONCLUSION

Ensuring system integrity is an essential security feature for cyber physical systems and the Internet of Things. The security design principle of "defense in depth" basically means that multiple layers of defenses are defined. This design principle can not only be applied at the system level, but also at the level of a single security mechanism.

This paper proposed a framework for ensuring system integrity in flexibly adaptable cyber physical systems. With new concepts for flexible automation systems coming with Industrial IoT / Industrie 4.0, the focus of system integrity has to move from preventing changes to device and system configuration to having transparency on the device and system configuration and checking it for compliance. This paper focused on integrity of devices, communication, and cyber systems. The addition of trusted physical sensors allows for cross-checking trusted measurements with the state of the industrial automation and control system. One-way data gateways can be used to provide integrity monitoring information from closed control networks to external networks for evaluation. Furthermore, approaches for integrity monitoring of encrypted communications have been presented.

The approaches for integrity monitoring in industrial automation and control systems described in this paper focus on the operation phase. Nevertheless, integrity in a broader sense has to cover the whole life cycle, from development, secure procurement, secure manufacturing, and supply chain security up to the commissioning phase in the operational environment.

REFERENCES

[1] R. Falk, S. Fries, "Enhancing Integrity Protection for Industrial Cyber Physical Systems", The Second International Conference on Cyber-Technologies and Cyber-Systems, CYBER 2017, November 12 - 16, 2017, Barcelona, Spain, available from: http://www.thinkmind.org/index.php?view=article&articleid=cyber_2017_3_30_80031 2018.01.23

[2] IEC 62443, "Industrial Automation and Control System Security" (formerly ISA99), available from: http://isa99.isa.org/Documents/Forms/AllItems.aspx 2018.01.23

[3] R. Falk, S. Fries, "Advanced Device Authentication for the Industrial Internet of Things", International Journal on Advances in Internet Technology, vol. 10, no 1&2, pp. 46-56, 2017, available from: http://www.iariajournals.org/internet_technology/inttech_v10_n12_2017_paged.pdf 2018.05.15

[4] ISO/IEC 27001, "Information technology – Security techniques – Information security management systems – Requirements", October 2013, available from: https://www.iso.org/standard/54534.html 2018.01.23

[5] IEC 62443-3-3:2013, "Industrial communication networks – Network and system security – Part 3-3: System security requirements and security levels", Edition 1.0, August 2013

[6] IEC 62554-4.2, "Industrial communication networks - Security for industrial automation and control systems - Part 4-2: Technical security requirements for IACS components", CDV:2017-05, May 2017

[7] Patrice Bock, Jean-Pierre Hauet, Romain Françoise, and Robert Foley: "Ukrainian power grids cyberattack - A forensic analysis based on ISA/IEC 62443", ISA InTech magazine, 2017, https://www.isa.org/templates/news-detail.aspx?id=152995 2018.01.23

[8] ZVEI: "Orientierungsleitfaden für Hersteller zur IEC 62443" [German], ZVEI Whitepaper, 2017, https://www.zvei.org/presse-medien/publikationen/orientierungsleitfaden-fuer-hersteller-zur-iec-62443/ 2018.01.23

[9] IEC 62351-12, Power systems management and associated information exchange - Data and communications security - Part 12: Resilience and security recommendations for power systems with distributed energy resources (DER) cyber-physical systems, https://webstore.iec.ch/ 2018.01.23

[10] Trusted Computing Group: "TPM Main Specification", Version 1.2, , available from http://www.trustedcomputinggroup.org/resources/tpm_main_specification 2018.01.23

[11] Trusted Computing Group, "Trusted Platform Module Library Specification, Family 2.0", 2014, available from http://www.trustedcomputinggroup.org/resources/tpm_library_specification 2018.01.23

[12] SELinux, "Security Enhanced Linux", available from: https://selinuxproject.org/page/Main_Page 2018.01.23

[13] AppArmor, "AppArmor Security Project", available from: http://wiki.apparmor.net/index.php/Main_Page 2018.01.23

[14] SMACK, "Simplified Mandatory Access Control Kernel", available from: https://www.kernel.org/doc/html/latest/admin-guide/LSM/Smack.html 2018.01.23

[15] OSSEC, "Open Source HIDS SECurity", web site, 2010 - 2015, available from http://ossec.github.io/ 2018.01.23

[16] OPC Foundation, "OPC Unified Architecture (UA)", available from: https://opcfoundation.org/about/opc-technologies/opc-ua/ 2018.01.23

[17] J. Case, R. Mundy, et al., "Introduction and Applicability Statements for Internet Standard Management Framework", RFC3410, available from: https://tools.ietf.org/html/rfc3410 2018.01.23

[18] R. Gerhards, "The Syslog Protocol", RFC5424, March 2009, available from: https://tools.ietf.org/html/rfc5424 2018.01.23

[19] S. Kent, K. Seo, "Security Architecture for the Internet Protocol", RFC4301, December 2005, acailable from https://tools.ietf.org/html/rfc4301 2018.01.23

[20] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, Aug. 2008, available from http://tools.ietf.org/html/rfc5246 2018.01.23

[21] Siemens: "Enhancing the first line of defense - Uni-directional communication to enable a connected world, whitepaper, December 2017, available from https://www.siemens.com/dcu 2018.01.23

[22] E. Rescorla: "The Transport Layer Security (TLS) Protocol Version 1.3", Internet draft (work in progres), September 2017, available from: https://tlswg.github.io/tls13-spec/draft-ietf-tls-tls13.html 2018.01.23

[23] D. Naylor, K. Schomp, et al., "Multi-Context TLS (mcTLS), Enabling Secure In-Network Functionality in TLS," available from http://mctls.org/ 2018.01.23

[24] B. Rothke, "Building a Security Operations Center (SoC)", RSA Conference, 2012, available from https://www.rsaconference.com/writable/presentations/file_upload/tech-203.pdf 2018.01.23

[25] McAffee Foundstone® Professional Services, "Creating and Maintaining a SoC", Intel Security Whitepaper, available from: https://www.mcafee.com/us/resources/white-papers/foundstone/wp-creating-maintaining-soc.pdf 2018.01.23

[26] R. Wichmann, "The Samhain HIDS", fact sheet, 2011, available from http://la-samhna.de/samhain/samhain_leaf.pdf 2018.01.23

[27] "SNORT", web site, available from https://www.snort.org/ 2018.01.23

[28] "Suricata", web site, available from https://suricata-ids.org/ 2018.01.23

[29] U.K. Premaratne, J. Samarabandu, T.S. Sidhu, et al., "An Intrusion Detection System for IEC61850 Automated Substations", IEEE Transactions on Power Delivery, Volume: 25, Issue 4, pp. 2376-2383, Oct. 2010

[30] I.N. Fovino, A. Carcano, T. De Lacheze Murel, A. Trombetta, et al., "IDS Modbus/DNP3 State-based Intrusion Detection System", 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), 2010, IEEE

[31] Trusted Computing Group, "TCG TPM 2.0 Automotive Thin Profile", level 00, version 1.0, 2015, available from http://www.trustedcomputinggroup.org/resources/tcg_tpm_20_library_profile_for_automotivethin 2018.01.23

[32] Texas Instruments: Anti-tamper Techniques to Thwart Attacks on Smart Meters, TI Training, 2018, available from https://training.ti.com/node/1128354 2018.01.23

[33] R. Falk and S. Fries, "Advanced Device Authentication: Bringing Multi-Factor Authentication and Continuous Authentication to the Internet of Things", The First International Conference on Advances in Cyber-Technologies and Cyber-Systems, CYBER 2016, October 9 - 13, 2016 - Venice, Italy , available from http://www.thinkmind.org/index.php?view=article&articleid=cyber_2016_4_20_80029 2018.01.23

[34] Y. Mo and B. Sinopoli, "On the Performance Degradation of Cyber-Physical Systems Under Stealthy Integrity Attacks", IEEE Transactions on Automatic Control 61.9 (2016): 2618-2624.

A Block Cipher Masking Technique for Single and Multi-Paired-User Environments

Ray R. Hashemi
Amar Rasheed
Jeffrey Young
Department of Computer Science
Georgia Southern University, Armstrong Campus
Savannah, GA, USA
e-mails: {rayhashemi, amarrasheed, alanyoung10101}
@gmail.com

Azita A. Bahrami
IT Consultation
Savannah, GA, USA
e-mail: Azita.G.Bahrami@gmail.com

***Abstract*— A ciphertext inherits some properties of the plaintext, which is considered as a source of vulnerability and, therefore, it may be decrypted through a vigorous datamining process. The vulnerability increases when a community of users is communicating with each other. Masking the ciphertext is the solution to this vulnerability. We have developed a new block cipher masking technique named Vaccine for which the block size is random and each block is further divided into segments of random size. Each byte within a segment is instantiated using a dynamic multi-instantiation approach, which means (i) the use of Vaccine does not produce the same masked outcome for the same given ciphertext and key and (ii) the choices for masking different occurrences of a byte are extremely high. Vaccine is tested in both single-paired-user and multi-paired-user communities with the revoking option. A key agreement is used to manage key changes required by the revoking option. For testing in a single-paired-user environment, two sets (100 members in each) of 1K long plaintexts of *natural* (borrowed from natural texts) and *synthesized* (randomly generated from 10 characters to increase the frequency of characters in the plaintext) are built. For each plaintext, two ciphertexts are generated using Advanced Encryption System (AES-128) and Data Encryption Standard (DES) algorithms. Vaccine and two well-known masking approaches of Cipher Block Chaining (CBC), and Cipher Feedback (CFB) are applied separately on each ciphertext. On average: (a) the Hamming distance between masked and unmasked occurrences of a byte using Vaccine is 0.72 bits higher than using the CBC, and CFB, and (b) Vaccine throughput is also 3.4 times and 1.8 times higher than the throughput for CBC and CFB, correspondingly, and (c) Vaccine *masking strength* is 1.5% and 1.8% higher than the masking strength for CBC and CFB, respectively. For testing in a multi-paired-user community with the revoking option, the findings remain the same for every single-paired-user. However, there is an overhead cost related to re-keying and re-profiling, which is caused by the revoking of a user from the community or expanding the community of users. The overhead cost is linearly related to the size of community.**

Keywords- Cyber Security; Masking and Unmasking Ciphertext; Variable-Block Cipher Vaccination; Masking Strength; Key Aggregation; Re-keying; Re-Profiling

I. INTRODUCTION

Protecting sensitive electronic documents and electronic messages from unintended eyes is a critical task. Such protections are provided by applying encryption. However, the encrypted text (ciphertext) is often vulnerable to datamining. The root of such vulnerability is in the *inherited-features* of the plaintext by the ciphertext. We have addressed the problem and its solution in the past for a secure *single-paired-user environment* [1]. (The distinct property of this environment is that there is no community of users, but there are individual pairs of users.) However, we expand the previously reported paper to address not only the single-paired-user environment but also a secure *multi-paired-user* environment with the option of revoking user.

The distinct properties of such an environment are: (i) presenting the environment as a graph in which users make the vertices and the communications established among the users make the edges of the graph, (ii) having a community-based key, which is the aggregation of the single-paired-user keys, and (iii) having the option to revoke the membership of a user in the community. The side effect of the last property demands a re-keying of the remaining single-paired users in the community whenever membership of a user is revoked. Regardless of having a secure single or multi-paired-user environment, the problem of inherited-features will not disappear and the problem is not limited only to primitive encryption modes such as displacement but also it can be observed in the outcome of more sophisticated encryption modes such as CBC and CFB [2][3][4]. The following examples provide some evidence.

As an example related to primitive encryption modes, let us consider the plaintext message of: "*The center is under an imminent attack"*. The plaintext may be converted into the following ciphertext using, for instance, a simple *displacement* encryption algorithm: "xligirxvmwyrhivermqqmrirxexxego". The features of the plaintext are also inherited by the ciphertext—a point of vulnerability. To explain it further, the word "attack" is

among the key words related to security. The characteristics of the word are: (i) length is six, (ii) the first and the fourth characters are the same, and (iii) the second and the third characters are the same. Using these characteristics, one can mine the given ciphertext and isolate the subtext of "exxego" that stands for "attack" which, in turn may lead to decryption of the entire message.

As an example related to more sophisticated encryption modes, the block cipher techniques that employ CBC/CFB encryption mode to produce distinct ciphertexts are vulnerable to information leakage. In the case of CBC/CFB using the same Initial Text Vector (IV) with the same encryption key for multiple encryption operations could reveal information about the first block of plaintext, and about any common prefix shared by two different plaintext messages. In CBC mode, the IV must, in addition, be unpredictable at encryption time; in particular, the (previously) common practice of re-using the last ciphertext block of a message as the IV for the next message is insecure (for example, this method was used by Secure Sockets Layer (SSL) 2.0). If an attacker knows the IV (or the previous block of ciphertext) before he specifies the next plaintext, he can check his guess about plaintext of some block that was encrypted with the same key before (this is known as the Transport Layer Security (TLS) CBC IV attack) [5].

The problem of inherited-features becomes a bigger concern when communication takes place within a secure community of users with a user revoking option. Upon revoking a user from the community, all the keys used for communication between any two users that collectively make a *community-based aggregate key*, need to be changed (re-keying process) on the fly. As a result, the inherited-features problem needs to be addressed in two environments: *Single-paired-user* and *multi-paired-user* [6][7].

In either environment, the fact remains the same that the logical solution for the inherited-features problem is to mask the ciphertext using a masking scheme that is *dynamic* and supports a high degree of *multi-instantiations* for each byte. A dynamic masking scheme does not produce the same masked outcome for the same given ciphertext and the same key. The high degree of multi-instantiation masking scheme replaces the *n* occurrences of a given byte in the ciphertext with *m* new bytes such that *m* is either equal to *n* or extremely close to *n*.

The goal of this research effort has two prongs:

(1) Introducing and building a dynamic masking scheme, named Vaccine for a single-paired-user environment. The Vaccine also supports a high degree of multi-instantiations that can mask the inherited-features of a ciphertext in the eye of a data miner while providing for transformation of masked ciphertext into its original form, when needed and

(2) Adapting the Vaccine for use in a multi-paired-user community that has the revoking option.

The Vaccine has the following three unique traits, which makes it a powerful masking scheme: It (1) divides the ciphertext into random size blocks, (2) divides each block into random size segments, and (3) every byte within each segment is randomly instantiated into another byte. All three traits are major departures from the norm of masking schema.

The rest of the paper is organized as follows. The Previous Works is the subject of Section II. For single-paired-user environment, the Methodology is presented in Section III, the Empirical Results are discussed in Section IV, and the findings are covered in Section V. For a multiple-paired-user environment, the adoption of the Vaccine is the subject of Section VI and the findings are discussed in Section VII. The conclusion and future research are presented in Section VIII.

II. PREVIOUS WORKS

In a secure single-paired-user environment, masking the features of a ciphertext that are either inherited from the plaintext or generated by the encryption scheme itself is the essential step in protecting a ciphertext. The block cipher and stream cipher mode of operations provide for such a step. We are specifically interested in CBC [8][9][10] and CFB [11] as samples of the block cipher and stream cipher mode of operations. They are to some degree comparable to the proposed Vaccine.

CBC divides the ciphertext into fixed–length blocks and masks each block separately. The use of fixed-length block demands padding for the last partial block of the ciphertext, if the latter exists. The CBC avoids generating the same ciphertext when the input text and key remain the same by employing an Initial Text Vector (IV).

CFB eliminates the need for possible padding of the last block (that is considered a vulnerability for CBC [12]) by assuming the unit of transmission is 8-bits. However, CFB also uses IV for the same purpose that it was used for CBC.

In contrast, Vaccine splits the ciphertext into random size blocks and then sub-divides each block into segments of random size. Masking each pair of segments is done by using a pair of randomly generated patterns. As a result, Vaccine needs neither padding nor IV. The randomness of the block size, segment size, and patterns used for instantiation of a given character are the major departure points of Vaccine from the other block and stream cipher approaches.

In a secure multi-paired-user environment with a revoking option, the use of CBC and CFB demand key management, which could be executed either by a key distribution [13][14][15][16] or key agreement [17][18][19] process (both processes are considered as pure overhead.) Key management is also adopted by Vaccine, which in turn requires the re-keying operation of the entire community-based aggregate key. Such overhead may be reduced by using the Vaccine. The reduction in overhead cost comes from the fact that the Vaccine masking is an amalgamation of random patterns, a key, and the use of a randomly generated byte from the plaintext. Therefore, if the revoked

user knows only the keys then, it is not enough to correctly de-mask (or mask) a ciphertext as long as the patterns' profile changes. Changing the patterns' profile is less expensive than the re-keying process. Of course, re-keying always remains as another viable option.

III. SINGLE-PAIRED-USER ENVIRONMENT

First, we present our methodology for instantiation of a byte, which contributes into dynamicity of Vaccine. Second, we introduce our methodology for building Vaccine. The details of the two methodologies are the subjects of the following two subsections.

A. Instantiation

Instantiation is the replacement of a byte, c, by another one, c', such that c' is created by some modifications in c. To perform the instantiation, we present our two methods of *Self-substitution* and *Mixed-Substitution*. Through these methods, a number of parameters are introduced that are referred to as the *masking* parameters. At the end of this subsection, we present the masking parameters as a *profile*.

1) Self-Substitution: Consider byte 10011101 and let us (i) pick two bits in positions p_1 and p_2 such that $p_1 \neq p_2$, (ii) flip the bit in position p_1, and (iii) swap its place with the bit in position p_2—Two-Bit-One-Flip-Circular-Swap technique.

It is clear that the pairs (p_1=1, p_2=7) and (p_1=7, p_2=1) create different instances for the byte. Therefore, the order of p_1 and p_2 is important. The number of possible ways selecting a pair (p_1, p_2) from the byte is 7*8=56, which means a byte may be instantiated by 56 possible different ways using Two-Bit-One-Flip-Circular Swap technique. The technique name may be generalized as *W-R-Bit-M-Flip-Circular-Swap*. For the above example (W=2 and M=1) the technique is shown in Figure 1. As a more general example, (W=8 and M=4) is also shown in Figure 1. As a rule, the value of parameter M is always less than the value of parameter W. This is necessary for not diminishing the effect of the swapping step. (We introduce the parameter R shortly.)

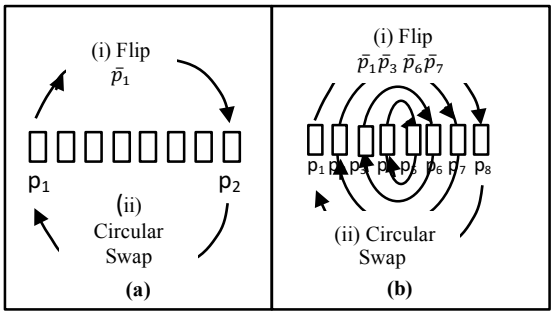


Figure 1. W-Bit-M-Flip-Circular-Swap Technique: (a) W=2 and M=1 and (b) W=8 and M=4

One may pick 3-bits (W=3) to instantiate the byte. Let us assume 3 bits randomly selected that are located in the positions p_1, p_2, and p_3. There are many ways that M-Flip-Circular-Swap technique can be applied:

a. (One-Flip-Circular-Swap) Flip one of the three bits and then make a circular swap among p_1, p_2, and p_3.
b. (Two-Flip-Circular-Swap) Flip two out of the three bits and then apply circular swapping.
c. (Three-Flip-Circular-Swap) is not used because it violates the rule of M being smaller than W.

The number of possible combinations grows to 5040.

Using the W-R-Bit-M-Flip-Circular-Swap for all possible values of W (W=2 to 8) and M (M=1 to W-1) generates the total of (X=1,643,448) possible substitutes for a given byte. If either W=1 or M= 0 then, the self-substitution has not been enforced and in this case X=1 (the byte itself). Now, we explain the role of parameter, R (where, R is a byte long) in the W-R-Bit-M-Flip-Circular-Swap.

Let us refer to the case of W=2 and M=1 one more time where it is able to facilitate the generation of 56 instantiations of a given byte using all the possible pairs of (p_1=•, p_2=•). That is, the two positions of p_1 and p_2 could have any value from 1 to 8 as long as $p_1 \neq p_2$. What if one is only interested in those instantiations resulting from the pairs of (p_1=3, p_2=•), which by definition also includes instantiations resulting from the pairs of (p_1=•, p_2=3)? The chosen value (bit) of interest for p_1 is a value from 1 to 8 that is expressed by setting the bit of interest in R. (Since p1 =3, the bit number 3, in R, is set to 1.) The number of bits that are set to "1" in R is always equal to M. For our example, R="00000100".

The pairs represented by (p_1=v, p_2=•) are the set of seven pairs of {(p_1=v, p_2=1), . . ., (p_1=v, p_2=8)}. The seven pairs are named the *primary set* for the *primary signature* of (p_1=v, p_2=•). The (p_1=*, p_2=v), which is a tweaked version of (p_1=•, p_2=v) is the *complementary signature* of (p_1=v, p_2=•) and stands for the other set of seven pairs {(p_1=8, p_2=v), . . ., (p_1=1, p_2=v)}. These seven pairs make the *complementary set* for (p_1=*, p_2=v). (Values of p_1, in the complementary set, are in reverse order of values of p_2 in the primary set.)

The primary and complementary sets also referred to as the *primary sub-pattern* and *complementary sub-pattern*, respectively. The two sub-patterns collectively make a *pattern* and the triplet of (W=2, M=1, R= "00000100") make the pattern's *stamp*, where W, M, and R are masking parameters. It is clear that M cannot be equal to W, because, when M= W, the primary and complementary sets are the same and they have only one member. This is another reason for supporting the rule of M must be smaller than W.)

The stamp of (W=4, M=3, R = "00001011") means four bits are chosen from the byte out of which three bits (M=3) in positions 1, 2, and 4 are the positions of interest (p_1=1, p_2=2, p_3=4.) Therefore, the primary signature and the Complementary signatures are, respectively, defined as (p_1=1, p_2=2, p_3=4, p_4=•) and (p_1= *, p_2=2, p_3=4, p_4=1).

When none of the bits in R is set to "1", it means R has not been enforced. In this case, we do not have the primary

and complementary sets. However, to apply Vaccine, we ought to have both sets. To do so, the default value of R, which is R with its M least significant bits set to "1" is used.)

2) Mixed-Substitution: We also extend the byte instantiation to the key. In a nutshell, the instantiation of the given byte, c, and each key byte are done separately. One of the instantiated key bytes is selected as the *key image* and the final instance of c is generated by XORing the key image and the instantiated c. The details are cited below.

Application of self-substitution with masking parameters of (W, M, and R) on a given byte generates the primary and the complementary sub-patterns of ($u_p^1 \ldots u_p^n$) and ($u_c^m \ldots u_c^1$). The subscripts p and c stand for these two sub-patterns and there are *n* and *m* members in the p and c sub-patterns, respectively. The key byte B_j is instantiated into another byte using the self-substitution with masking parameters of (W_j, M_j, and R_j, for j=1 to 4). Application of self-substitution on the individual four bytes of the key (B_1 . . . B_4) generates the primary and the complementary sub-pattern for each byte as follows:
($u_p^{1B1} \ldots u_p^{n1B1}$) and ($u_c^{m1B1} \ldots u_c^{1B1}$),
($u_p^{1B2} \ldots u_p^{n2B2}$) and ($u_c^{m2B2} \ldots u_c^{1B2}$),
($u_p^{1B3} \ldots u_p^{n3B3}$) and ($u_c^{m3B3} \ldots u_c^{1B3}$), and
($u_p^{1B4} \ldots u_p^{n4B4}$) and ($u_c^{m4B4} \ldots u_c^{1B4}$).

A byte, say c_1, using the first member of the primary sub-pattern, u_p^1, is instantiated to c_1'. The first byte of the key, B_1, using its first member of the primary sub-pattern, u_p^{1B1}, is instantiated to B_1'. The other three bytes are also instantiated into B_2', B_3', and B_4' using their first member of the primary sub-patterns, $u_p^{1B2}, u_p^{1B3}, and\ u_p^{1B4}$, respectively. The Hamming distance of *HD*(c', B_j'), for j=1 to 4, are measured and B'=Argmax[*HD*(c', B_j'), for j=1 to 4] is the key image. In the case that there are ties, the priority is given to the instantiated byte of B_1, B_2, B_3, and B_4 (and in that order.) The final substitution for c_1 is:

$$c_1'' = (c_1' \oplus B') \quad (1)$$

The next byte, c_2, within a given segment of ciphertext is instantiated to c_2' using u_p^2, and key bytes of B_1, B_2, B_3, and B_4 are instantiated to B_1', B_2', B_3', and B_4' using $u_p^{2B1}, u_p^{2B2}, u_p^{2B3}, and\ u_p^{2B4}$, respectively.

B'=Argmax[*HD*(c', B_j'), for j=1 to 4] and c_2''=(c_2'⊕ B'). The process continues until the segment of the ciphertext is exhausted. The bytes of the next sub-list and the key bytes are instantiated using the complementary sub-patterns. Therefore, the sub-patterns are alternately used for consecutive segments of the ciphertext.

Using mixed substitution, the number of possible combinations for each key byte is equal to X and for the key of four bytes is X^4 (>$1.19*10^{31}$ combinations.) The reader needs to be reminded that the four-byte key may be expanded to the length of N bytes for which the outcome of XOR is one of the X^{N+1} possible combination. For N=16 (128-bit key) The XOR is one of the X^{17} possible combinations (>$4.65*10^{105}$.)

3) Profile: Considering both self and mixed substitutions, the masking parameters grow to five triplets: The first triplet, (W, M, and R) for the instantiation of a byte of segment and the next four triplets of (W_j, M_j, and R_j, for j=1 to 4) for instantiation of the four bytes of the key. Therefore, patterns' profile, or simply profile, includes 15 masking parameters, which are accommodated by a 96-bit long binary string (Figure 2) as described below.

Since the possible values for each of the parameters W and W_j is eight (2 through 8 and value of 1 means the self-substitution has not been enforced), the value of each parameter can be accommodated by 3 bits (the total of 15 bits). Since the three bits make a decimal value between 0 and 7, we always add 1 to the decimal value to get the true value for W or W_j.) The parameters M and M_j have eight possible values (1 through 7 where the value of 0 means that self-substitution has not been enforced) and each parameter can also be accommodated by 3 bits (the total of 15 bits). The parameters R and R_j need eight bits each (the total of 40 bits). In addition, we use twenty-six bits as *prefix* of the profile (five bits as *reserved* bits for possible expansion, sixteen bits as the *Flag bits* and five bits as the *Preference bits*.)

The sixteen flag bits represent a decimal number (Δ) in the range of (0: 65,535). Let us assume that the length of the ciphertext that is ready to be masked is L_{ct}. Three bytes of f_1, f_2, and f_3 of the ciphertext are selected (flagged), which are in locations: $\delta_1 = \delta$, $\delta_2 = \lfloor L_{ct}/2+\delta/2 \rfloor$, and $\delta_3 = L_{ct} - \delta$, where, δ is calculated using the formula (2)

$$\delta = \begin{cases} \Delta\ Mod\ L_{ct}, & \Delta > L_{ct} \\ L_{ct}\ Mod\ \Delta, & \Delta \leq L_{ct} \end{cases} \quad (2)$$

The flagged bytes will not be masked during the vaccination process and they collectively make the *native byte* of $F=(f_1 \oplus f_2 \oplus f_3)$. Since the length of the ciphertext and the length of its masked version remain the same there is no need for including the length of the ciphertext in the profile. The question of why the flagged bytes are of interest will be answered shortly.

The purpose of preference bits is to build a *model*, which is influenced by both the key and flagged bytes. The model is used to create variable length blocks and segments. The minimum number of preference bits is five and can grow up to ten by consuming the reserved bits. The preference bits are partitioned such that the most significant bit is considered partition one and the rest of the bits make partition two.

To build the model, a desired byte number (z) of the key is identified by the bits of partition two. That is, one can select any byte from a maximum of a 512-byte long key. The key is treated as circular and two pairs of bytes $A_1=(z+1||z)$ and $A_2=(z+2||z-1)$ are obtained from the key. A new pair of bytes of $A_3=A_1 \oplus A_2 \oplus (F||F)$ is built. If the bit in the partition one is set to zero then, the model is A_3;

otherwise, the model is $a_1 \oplus a_2$, where, a_1 and a_2 are the pair of bytes in A_3.

Let us assume that there are two similar ciphertexts of CT_1 and CT_2 and we are using the same key and the same profile to mask the two ciphertexts, separately. As long as one of the three flagged bytes in CT_1 and CT_2 is different, the native bytes for the ciphertexts are different and so their models, which in turn make their masked versions different. This is one of the major advantages of Vaccine.

To summarize, using a 4-byte key, the number of bits needed for the profile is 96 bits. Dissection of a pattern profile is shown in Figure 2. The 24 hex digits representing the patterns' profile along with eight hex digits representing the 4-byte key that are collectively called *Masking Image*, may be sent to the receiver in advance or they may hide in the masked ciphertext itself:

a. In a predefined location/locations,
b. In location/locations determined by the internal representation of the key following some formula(s), or
c. A mixture of (a) and (b).

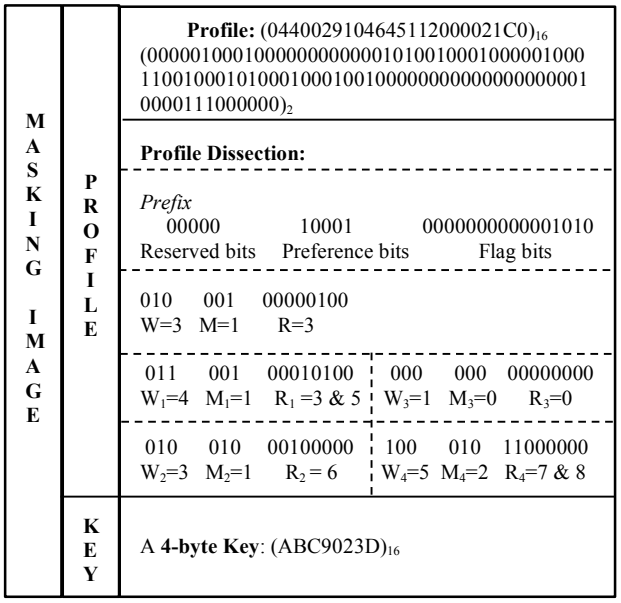


Figure 2. Dissection of the masking Image

B. Vaccine

Vaccine is a variable-block cipher methodology capable of masking and unmasking a ciphertext. The details of masking and unmasking of Vaccine are presented in the following two subsections.

1) Masking of the Ciphertext: Vaccine as a masking scheme is able to mask the features of a ciphertext in the eye of a text miner. Vaccine: (1) divides the ciphertext into random size blocks, (2) each block, in turn, is divided into a number of segments such that the length of each segment is random, and (3) every byte within each segment is randomly instantiated to another byte using self and mixed substitutions. The masking process is encapsulated in algorithm Mask shown in Figure 3.

The algorithm is made up of four sections. In section one, (Step 1 of the algorithm) the profile is dissected to extract masking parameters and they, in turn, generate primary and complementary sub-patterns for five patterns:($Pattern_p^0$, $Pattern_c^0$), ($Pattern_p^{B1}$, $Pattern_c^{B1}$), ($Pattern_p^{B2}$, $Pattern_c^{B2}$), ($Pattern_p^{B3}$, $Pattern_c^{B3}$), and ($Pattern_p^{B4}$, $Pattern_c^{B4}$) used for masking the chosen byte of the ciphertext and the four key bytes, respectively. The array of pt with five elements keeps track of those primary and complementary sub-patterns of the five patterns that are in use. The model is also extracted in this step.

Algorithm Mask
Input: A 32–bit key, a pattern's profile of 96-bit, and a ciphertext, CT.
Output: Delivering IC as the masking version of CT.
Method:
Step1- //Dissection of the profile and initializations
Dissection delivers primary and secondary sub-patterns of five patterns ($Pattern_p^0$, $Pattern_c^0$), ($Pattern_p^{B1}$, $Pattern_c^{B1}$), ($Pattern_p^{B2}$, $Pattern_c^{B2}$), ($Pattern_p^{B3}$, $Pattern_c^{B3}$), and ($Pattern_p^{B4}$, $Pattern_c^{B4}$).
κ ←Model obtained by using Preference bits, Flag bits, and key;
IC ← ""; C ← CT;
pt[5]← 0;//pt gives turn to the primary (pt[•]=0) and complementary (pt[•] =1) sub-patterns of the five patterns for initializing the CurrentP [5];
Step 2-Repeat until C is exhausted
a- Get the set of decimal numbers from κ in ascending order: D ={d_1, d_2, . . . d_{y-1}, d_y};
Get the next random size block,
β_n=Substr(C, 0, d_y);
b- CL = 0; //Current location in C
c- Repeat for i =1 to y-1
//Divide β_n into y-1 segments;
s_i = Substr(β_n, CL, d_i - CL);
CL = CL+ d_i;
CurrentP[m]=$Pattern_{pt}^m$ //for m =0 to 4;
d- Repeat for each byte, c_j, in s_i
d_1- If (c_j is a flagged byte) Then continue;
d_2- If (CurrentP[0] is exhausted)
Then CurrentP[0] = $Pattern_{pt}^0$;
d_3- c_j' = Flip c_j bits using CurrentP[0];
d_4- c_j' = Circularly swap proper c_j bits using CurrentP[0];
d_5- σ = Select(c_j', CurentP[1], CurrentP[2],CurrentP[3],CurrentP[4]);
d_6- a = c_j' ⊕ σ;
d_7- IC←IC || a;
End;
pt[•]++; pt[•] ← pt[•] mode 2;
End;
e- Remove block β_n from C;
f- Apply one-bit-left-rotation on κ;
End;
End;

Figure 3. Algorithm Mask

The second section (Step 2.a of the algorithm) identifies a random size block prescribed by κ—the model.

The identification process is done by creating y binary numbers using κ. The i-th binary number starts from the least significant bit of the κ and ends at the bit with the i-th value of "1" in κ. The binary numbers are converted into decimal numbers and sorted in ascending order, $\{d_1, d_2, \ldots d_{y-1}, d_y\}$. The block, β_n=Substr(C, 0, d_y), where C is initially a copy of the cipher text.

The third section (Step 2.c of the algorithm) divides block β_n into a number of random size segments. The size and the number of segments are dictated by the κ internal representation. Block β_n has y segments: $\{ s_0 \ldots s_{y-1}\}$.

The segment s_i starts from the first byte after the segment s_{i-1} (the location is preserved in variable CL) and contains $\lambda_i = d_{i+1} - d_i$ bytes. The number of segments and their lengths are not the same for different blocks.

To get the next block of the ciphertext, the block β_n is removed from C (Step 2.e) and κ is changed by having a one-bit-left-rotation (Step 2.f). Using the above process along with the new κ, the next block with a different size is identified. This process continues until C is exhausted. It is clear that the lengths of blocks are not necessarily the same. In fact, the lengths of the blocks are random. It needs to be mentioned that the length of blocks β_i and β_{i+8} are the same when κ is one byte long. When κ is two bytes long, the length of the blocks β_i and β_{i+16} are the same. And a block on average is 32,768 bytes long. As a result, the ciphertext, on average, must be longer than 491,520 bytes before the blocks' lengths are repeated.

Algorithm Select
Input: A byte (c), Key, and four patterns for the four key bytes.
Output: key image, k.
Method:
a. Repeat for (w = 1 to 4)
  If (CurrentP[w] is exhausted)
  Then CurrentP[w] = $Pattern_{pt}^{w}$;
End;
b. h ← -1;
c. Repeat for v= 1 to 4;
  i. c_v ← An instantiated version of $KeyByte_v$ using related sub-pattern.
  ii. If HD(c, c_v) >h //HD is Hamming distance function
  Then h = HD(c, c_v); k = c_v;
End;
End;

Figure 4. Algorithm Select

The fourth section (Step 2.d of the algorithm) delivers the masked version of the ciphertext, byte by byte, for a given segment. Flagged bytes are not masked (Step $2.d_1$). If the number of bytes in the segment s_i is greater than the cardinality of the pattern then, the pattern repeats itself (Step $2.d_2$). Each byte, c_j, of the segments s_i (for i=1 to y-1) are masked by applying (i) the relevant member of the current sub-pattern on byte c_j (Step $2.d_3$ and $2.d_4$), (ii) identifying the key image (Step $2.d_5$), by invoking the Algorithm Select (Figure 4), (iii) create c_j', the masked version of c_j, by XORing the outcome of process (i) and process (ii), (Step $2.d_6$), and (iv) concatenate the masked version of c_j, to the string of IC, which ultimately becomes the inoculated version of the inputted ciphertext (Step $2.d_7$).

2) Unmasking of the Ciphertext: For unmasking a masked ciphertext, those steps that were taken during the masking process are applied in reverse order. Therefore, the Algorithm Mask with a minor change in step 2.d can be used for unmasking. We show only the changes to Step d of Figure 3 in Figure 5.

d- Repeat for each byte, c_j', in s_i
  d_1- If (c_j is a flagged byte) Then continue;
  d_2- If (CurrentP[0] is exhausted) Then CurrentP[0] = $Pattern_{pt}^{0}$;
  d_3- σ = Select(c_j', CurentP[1], CurrentP[2], CurrentP[3], CurrentP[4];
  d_4- α = c_j' ⊕ σ;
  d_5- α = Circularly swap bits of α using CurrentP[0];
  d_6- α = Flip a bits using CurrentP[0];
  d_7- UM←UM||α; //UM is the unmasked ciphertext;
End;

Figure 5. The modified part of the Algorithm Mask

IV. EMPIRICAL RESULTS

To measure the effectiveness of the proposed Vaccine, we compared its performance with the performance of the well-established masking algorithms of CBC and CFB. The behavior of Vaccine was observed using three separate profiles of simple, moderate, and complex. These observations are named VAC_s, VAC_m, and VAC_c.

Two plaintext templates of *natural* and *synthetic* were chosen and 100 plaintexts were generated for each template. Each plaintext following the first template was selected from a natural document made up of the lower and upper-case alphabets and the 10 digits—total of 62 unique symbols. Each plaintext following the second template was randomly synthesized using the 10 symbols set of {A, b, C, L, x, y, 0, 4, 6, 9}. The goal was to synthesize plaintexts with high occurrences of a small set of symbols. Each plaintext created under both templates was 1K bytes long.

For each plaintext, two ciphertexts of C_a and C_d were generated using Advanced Encryption System (AES-128) and Data Encryption Standard (DES) algorithms [20][21][22]. The masking approaches of CBC, CFB, VAC_s, VAC_m, and VAC_c were applied separately on C_a and C_d generating the masked ciphertexts of:

$\{C_a^{cbc}, C_a^{cfb}, C_a^{vac_s}, C_a^{vac_m}, C_a^{vac_c}\}$ and
$\{C_d^{cbc}, C_d^{cfb}, C_d^{vac_s}, C_d^{vac_m}, C_d^{vac_c}\}$.

When CFB was applied on C_a and C_d the key lengths were 64-bit and 128-bit, respectively, and the IV was chosen from a natural document. (The least significant 64 bits of the 128-bit key was used as the key when CFB was applied on C_a. The key used by VAC_s, VAC_m, and VAC_c was also borrowed from the least significant 32 bits of the 128-bit key used for CFB.)

Let us consider the first set of masked ciphertexts $\{C_a^{cbc}, C_a^{cfb}, C_a^{vac_s}, C_a^{vac_m}, C_a^{vac_c}\}$ generated from C_a. The

following steps are used to compare the effectiveness of the proposed Vaccine with CBC and CFB. (The same steps are also followed to compare the effectiveness of the proposed Vaccine with CBC and CFB using the masked ciphertexts of $\{C_d^{cbc}, C_d^{cfb}, C_d^{vac_s}, C_d^{vac_m}, C_d^{vac_c}\}$.)

a. Get the list of unique symbols, which makes up the plaintext, List=$\{\sigma_1 \ldots \sigma_m\}$.
b. Get the frequency of symbol σ_i, for i = 1 to m, and calculate the average frequency of the symbols.
c. Repeating the next two steps for every symbol, σ_i, in the list.
d. Identify the locations for all the occurrences of the symbol, σ_i, in the plaintext, ($\ell^i_1 \ldots \ell^i_n$).
e. Identify the bytes in the locations of ($\ell^i_1 \ldots \ell^i_n$) within the $C_a^\bullet$ and calculate the Hamming distance, h_j, between the two bytes in location ℓ_j, for j=1 to n, in the plaintext and $C_a^\bullet$. The overall average of Hamming distance for the symbol σ_i is $h_{\sigma i}$=Average($h_1 \ldots h_n$),
f. Concluding that the underline masking methodology with the highest average values of the Hamming distances have a superior performance.

TABLE I. AVERAGE OF HAMMING DISTANCES BETWEEN THE TWO 100 PLAINTEXTS OF 1K BYTE LONG (GENERATED BY TWO TEMPLATES) AND THEIR RELATED MASKED CIPHERTEXTS: (A) ENCRYPTED BY AES AND (B) ENCRYPTED BY DES

		AES-128				
	Avg. Symb.	*CBC*	*CFB128*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	Dist.	Dist.	Dist.	Dist.	Dist.
Syn.	103	3.568	3.570	4.415	4.373	4.411
Natu.	16.5	3.569	3.561	4.423	4.361	4.411
(a)						
	Avg. Symb.	DES				
		CBC	*CFB64*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	Dist.	Dist.	Dist.	Dist.	Dist.
Syn.	103	3.527	3.526	4.182	4.153	4.223
Natu.	16.5	3.513	3.515	4.176	4.141	4.221
(b)						

TABLE II. THROUGHPUT AVERAGE IN MILISECOND FOR THE TWO 100 PLAINTEXTS OF 1K BYTE LONG (GENERATED BY TWO TEMPLATES): (A) ENCRYPTED BY AES AND (B) ENCRYPTED BY DES

		AES-128				
	Avg Symb.	*CBC*	*CFB128*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	*TPut.*	*TPut.*	*TPut.*	*TPut.*	*TPut.*
Syn.	103	4545	11111	25000	33334	20000
Natu.	16.5	12500	10000	16667	20000	12500
(a)						
	Avg Symb.	DES				
		CBC	*CFB64*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	*TPut.*	*TPut.*	*TPut.*	*TPut.*	*TPut.*
Syn.	103	3846	11111	20000	25000	14286
Natu.	16.5	10000	10000	14286	20000	11111
(b)						

The outcome of applying the above steps on the ciphertexts of $\{C_a^{cbc}, C_a^{cfb}, C_a^{vac_s}, C_a^{vac_m}, C_a^{vac_c}\}$ and $\{C_d^{cbc}, C_d^{cfb}, C_d^{vac_s}, C_d^{vac_m}, C_d^{vac_c}\}$ are shown in Table I.a and Table I.b. We have also used the system clock to calculate the average throughput (in millisecond) for the masking approaches of CBC, CFB, VAC_s, VAC_m, and VAC_c and reported in Tables II.a and II.b.

In addition, a *masking strength* of μ ($0 < \mu < 1$), is introduced that is defined as $\mu = N_{inst} / N_{occ}$, where N_{inst} is the number of unique bytes in the masked ciphertext representing the instantiations of the N_{occ} occurrences of symbol σ_i in the underlying plaintext of the masked ciphertext. The masking strength for CBC, CFB, VAC_s, VAC_m, and VAC_c are presented, respectively, in Tables III.a and III.b.

TABLE III. AVERAGE MASKING STRENGTH FOR THE TWO 100 PLAINTEXTS OF 1K BYTE LONG (GENERATED BY TWO TEMPLATES): (A) ENCRYPTED BY AES AND (B) ENCRYPTED BY DES

		AES-128				
	Avg. Symb.	*CBC*	*CFB128*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	μ	μ	μ	μ	μ
Syn.	103	0.506	0.486	0.451	0.540	0.571
Natu.	16.5	0.882	0.878	0.845	0.890	0.889
(a)						
	Avg. Symb.	DES				
		CBC	*CFB64*	*VAC$_s$*	*VAC$_m$*	*VAC$_c$*
Tem.	Freq.	μ	μ	μ	μ	μ
Syn.	103	0.501	0.494	0.490	0.564	0.570
Natu.	16.5	0.878	0.894	0.880	0.909	0.893
(b)						

V. FINDINGS FOR SINGLE-PAIRED-USER ENVIRONMENT

The performance of the presented new cipher block approach, Vaccine, for masking and unmasking of ciphertexts seems superior to the performance of the well-known masking approaches of CBC and CFB.

The advantages of Vaccine over CBC and CFB are numerated as follows:

a. The key and patterns' profile may hide in the masked ciphertext.
b. The block size for Vaccine is not fixed and it is selected randomly.
c. Each block is divided into segments of random size.
d. The masking pattern changes from one byte to the next in a given segment.
e. Masking a ciphertext using Vaccine demands mandatory changes in the ciphertext. Therefore, the identity transformation could not be provided through the outcome of Vaccine. The simple proof is that the Hamming weight is modified.
f. The results revealed that on average:
   i. The Hamming distance between masked and unmasked occurrences of a byte using Vaccine is 0.72 bits higher than using CBC and CFB.
   ii. Vaccine throughput is 3.4 times and 1.8 times higher than throughput for CBC and CFB.

iii. Vaccine masking strength is 1.5% and 1.8% higher than masking strength for CBC and CFB.
iv. VAC_m masking strength is 3.6% and 3.7% higher than masking strength for CBC and CFB. And VAC_c masking strength is 3.9% and 4.2% higher than masking strength for CBC and CFB.

VI. MULTI-PAIRED-USER ENVIRONMENT

A secure multi-user environment [13][23][24] is represented by a graph, G(V, E), where V is a set of vertices representing the user members of the environment and E is a set of bi-directional edges indicating the communication between paired users. As an example, let us consider a secure multi-user environment, Figure 6, for which V is composed of the set $\{v_1, v_2, v_3\}$ and E is composed of the set $\{e_{1,2}, e_{1,3}, e_{2,3}.)$

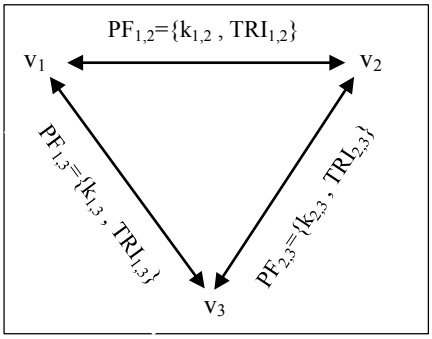


Figure 6. A safe multi-user environment with three cooperative users

There is a pairwise masking image, $MI_{i,j}$ associated with each edge, $e_{i,j}$, that enables the two users of v_i and v_j to communicate securely with each other (i.e., vaccinated messages can flow between v_i and v_j through the edge of $e_{i,j}$.) The pairwise $MI_{i,j}$ has two major components of a pairwise key, $k_{i,j}$, and a pairwise profile, $PF_{i,j}$, Figure 2. For a pairwise $k_{i,j}$ of four-byte long the $PF_{i,j}$ includes a prefix, $PRE_{i,j}$, and five sets of triplets (W, M, R) , $TRI_{i,j} = \{Tri^0_{i,j}, \ldots, Tri^4_{i,j}\}$, that each one prescribes a set of patterns. The number of bits to accommodate one triplet in PF_{ij} is 14. The pairwise key $k_{i,j}$ can grow as many bytes as desired (the maximum of 512 bytes.) The number of triplets in $TRI_{i,j}$ also grows with the growth of key length. For each added byte to the key, a new triplet is added to the PF_{ij}. Beyond the key length of sixteen bytes, the length of prefix also grows one bit at a time. Each added bit to the prefix doubles the length of key in bytes. Since there are five reserved bits in PF_{ij} and they can be used to increase the length of the prefix, the key length can grow up to 512 bytes. Each one of these bytes can be addressed by the second partition of the preference bits (i.e., all the preference bits excluding the most significant bit.) For the key length of L_k bytes, the PF length, L_p, is calculated in bits using formula (3):

$$L_p = 14(L_k+1) +26 \quad (3)$$

Since the pairwise masking images are associated with edges, we also refer to MI, k, and PF as the edge masking image, edge key, and edge profile, respectively.

The secure multi-user environment of our interest has the option of revoking existing users. That is, upon revoking a user the keys become vulnerable and a key management approach needs to be employed. Key management is achieved in two ways: key distribution [13][14][15][16] and key agreement[17][18][19]. In the key distribution approach, each user has its own private key and it is only shared with a key-distribution center (KDC). In the case that v_k needs to communicate with v_m, KDC is asked for a session key that will be generated and delivered to both v_k and v_m.

In the key agreement approach, a number of users agree on having one community-based key, K, and all users participate in building such a key by donating their individual keys. The community-based key remains private to the members.

The revoking option using key distribution has the same overhead cost for using CBC, CFB, and Vaccine. However, adaptation of the Vaccine into the key agreement approach suggests some interesting developments that need to be discussed.

The key agreement approach requires that as soon as a user, v_k, is revoked all the edges be re-keyed, re-profiled, or both (regardless of substituting or not substituting the revoked user.) Let us take a closer look at these three options. During the masking process conducted by Vaccine, edge key and edge profile, play a role. In fact, the masking of a ciphertext is completed by the use of the *model* and *native byte* (both terms explained in Section III), which in turn was generated by employing the prefix (PRE), triplets (TRI), and key (k), along with the plaintext. Therefore, the revoked user cannot correctly mask (or de-mask) a ciphertext as long as one of the three parameters of PRE, TRI and k changes. However, both re-keying and re-profiling may provide a higher masking strength.

To implement the Vaccine adaptation into the key agreement approach we present three algorithms of Keys, Carve, and Profile. The Keys algorithm dynamically creates N pairwise keys of a given length for N edges that collectively make the community-based key, K. The algorithm Carve randomly generates a set of triplets based on the key length and also enforces the internal constraints on the randomly generated triplets. The Algorithm Profile generates either one profile used by all pairwise keys or N profiles for the N edges. The details of the three algorithms are covered in the following three subsections.

A. Algorithm keys

Dynamically creating a Community-based key, K, which is composed of a large number of edge keys (one per edge and for the purpose of re-keying) is encapsulated by the algorithm Keys, Figure 7. In the N iterations of Step 3, which is the number of edge keys, the algorithm delivers N pairwise keys with the length of L_k bytes that are randomly

generated. This is accomplished by randomly generating L_k binary strings with the length of L_k = 256-bit (Step 2.) Eight bits from each one of the L_k strings are selected randomly (Step 3.a.) The obtained bytes are concatenated in a random order to make one edge key of L_k bytes long (Step 3b.) The resulting edge key, k, is added to the Community-based aggregated key, K, (Step 3.c.)

Algorithm Keys (L_k, N)
Input: L_k, which is the length of key in bytes and N is the number of edge keys needed.
Output: A Community-based key, K, made up of N edge keys randomly generated on fly.
Method:
Step 1: k = ""; λ =0; J =0;
Step 2: Create L_k binary numbers of 256-bit long: S_1, . . ., S_{Lk};
Step 3: Repeat (while λ < N)
  a: Select eight bits randomly from each string: B_1, . . ., B_{Lk};
  b: Repeat (while j < L_k)
    a1: Randomly pick a byte, B', from the set{ B1, . . ., B_{Lk}};
    a2: k = k|| B'; j = j++;
  End;
  c: K = K||k;
  d: k = "";
  e: j =0;
  f: λ = λ++;
End;
End;

Figure 7. Algorithm Keys

B. Algorithm Carve

The algorithm Carve, shown in Figure 8, accepts an edge key with length of L_k bytes and randomly creates a binary number, TRI, of the length L_p delivered by formula (3) (Step 2.)

Considering Figure 2, the length of $PRI_{i,j}$ in the edge profile of $PF_{i,j}$ is 26 bits. As we mentioned previously the number of bits needed to express each triplet of (W, M, R) in $PF_{i,j}$ is 14 bits. Therefore, each 14 bits after the 26-bit prefix is made up of three parts (Part1, Part2, and Part3). Part1 (W) is 3 bits long and starts from the location r1 = 26 in the profile. Part2 (M) is also 3 bits long and starts immediately after Part1 (i.e., location r2 = r1 +3.) Part3 (R) is 8 bits long and starts 6 bits after Part1 (i.e., location r3 = r1 +6.) (Step 3 gets the starting points of Part1, Part2, and Part3.)

The locations of Part1of all the triplets are separated by 14 bits and the same is true for the locations of Part2 and Part3 of all triplets. The three parts of each triplet are obtained in Step 4.a and converted to decimal numbers in Step 4.b. In reference to the three parts of the triplets we have three concerns that need to be addressed. These concerns are in reference to the constraints on W, R, and M values in W-R-Bit-M-Flip-Circular-Swap technique.

The first concern is about the validity of the equivalent decimal value carried by (Part1+1) of the triplet. If the value is less than 2, the value changes to 2 (Step 4.c.)

The second concern is about the equivalent decimal value in Part2 that must be at least one less than the value in Part1 (a constraint rule between W and M.) If the value of Part 2 is greater than or equal to the value in Part1, it is reduced to make it smaller such that the value of Part 1 is higher by 1 (Step 4.d.)

Algorithm Carve (k, L_k)
Input: An edge key, k, with the length of L_k bytes.
Output: Dynamically generating a set of triplets, TRI, that is in agreement with k.
Method:
Step 1: i = 0;
Step 2: Create a random binary number of $L_p = 14(L_k+1) +26$ bits long, TRI;
Step 3: $r_1 = 26$; r2 = r_1 +3; r3 = r_1 +6;
Step 4: Repeat while (i <14 L_k)
  a: $Part_i^1$ = SUBSTR(TRI, r_1+i, 3);
    $Part_i^2$ = SUBSTR(TRI, r_2+i, 3);
    $Part_i^3$= SUBSTR(TRI, r_2+i, 7);
  b: f_1 = DECIMAL($Part_i^1$)+1; f_2 = DECIMAL($Part_i^2$) ;
    /*DECIMAL function converts a given binary number into decimal number*/
  c: If (f_1 < 2) Then f_1 = 2;
  d: If ($f_1 \le f_2$) Then $f_2 = f_1$-1;
  e: If (COUNT($Part_i^3$) – f_2 > 0)
    Then randomly flip (COUNT($Part_i^3$) – f_2) bits in $Part_i^3$ to 0;
    If (COUNT($Part_i^3$) – f_2 < 0)
    Then randomly flip |COUNT($Part_i^3$) – f_2| bits in $Part_i^3$ to 1;
  f: SUBSTR(TRI, r_1+i, 3) = BINARY(f_1);
    SUBSTR(TRI, r_2+i, 3) = BINARY(f_2); /*BINARY function converts a given decimal number into binary number*/
    SUBSTR(TRI, r3+i, 8)= $Part_i^3$;
  g: i = i+14;
End;
End;

Figure 8. Algorithm Carve

The third concern is about the validity of the content of Part3 of the triplet. The count of 1s in Part3 must be equal to the equivalent decimal value carried by Part2 of the triplet. If the count of 1s is higher, randomly enough 1s are flipped to zero and if the number of 1s is lower, randomly enough 0s are flipped to one to solve the problem (Step 4.e.)

Replace the three parts of the triplet with their new changes (Step 4.f.) By adjusting the index of i (Step 4.g) all the parts in TRI are inspected and corrected.

C. Algorithm Profile

Now we are ready to look into two cases of using:

(a) a new aggregated community-based key, K, and (i) one new profile for all edges or (ii) one new profile per edge and
(b) the existing aggregate key for the community and (i) one new profile for all edges or (ii) one new profile per edge.

One may raise the question of why case (b) is a valid case to begin with. The question is an important one because

keeping the existing aggregate key may jeopardize the overall security of the system. To answer the question, as it was mentioned before, the PF (composed of PRE and TRI) and k are needed to complete the vaccination. As long as either PF or k changes, the vaccination results are different. Therefore, case (b) is a legitimate one. The set of steps for implementing case (a) and case (b) are given in the algorithm Profile shown in Figures 9.

```
Algorithm Profile (Option, Lk, N, EK)
Input: Option (Option = "10" means a new aggregate key
       and one new profile is needed for the entire
       aggregate key. Option = "11" means a new
       aggregate key and one new profile is needed for
       each edge key within the aggregate key. Option =
       "00" means one new profile is needed for the entire
       aggregate old key. Option = "01" means one new
       profile is needed for each edge key within the old
       aggregate key.), a desired key length, Lk, the number
       of needed edge keys, N, and an existing community-
       based key, EK.
Output: (a) A new community-based key K and one or N
            new profiles or
        (b) One or N new profiles
Method:
 Step 1: Switch (Option)
          a: Case "10":
              K = Keys(Lk, N);
              k = one individual key in the community-
                  based key K;
              Carve(k, Lk);
              Break;
          b: Case "11":
              K = Keys(Lk, N);
              Repeat for every key, ki, in the community-
                | based key K
                Carve(ki, Lk);
              End;
              Break;
          c: Case "00":
              ek = one individual key in EK.
              Carve(ek, Lk);
              Break;
          d: Default:
              Repeat for every key, eki, in EK.
                | Carve(ki, Lk);
              End
 End;
```

Figure 9. Algorithm Profile

The parameter Option is 2 bits long and the values "1" and "0" for the most significant bit represent case (a) and case (b), respectively. The option (i) in both cases is represented by the least significant bit set to "0". The option (ii) in both cases is represented by the least significant bit set to "1".

For case (a) option (i), the algorithm does not re-key the existing community-based key, EK, however, it generates one profile used by all users (Step 1.a) and for the same case option (ii), no re-keying takes place and the algorithm generates a new edge profile for every edge (Step 1.b.) For the case (b) the algorithm reacts the same way that it has reacted for case (a) except that for both options of (i) and (ii) first, a new community-based, K, is generated and then one new edge profile (Step 1.c) or N edge profiles (one per edge) are generated (Step 1.d.)

It is worth mentioning that in the case of the pair-wise community growth, the above algorithms easily can provide for re-keying and re-profiling of only the new edges without disturbing the existing edge masking images.

VII. FINDINGS FOR MULTI-PAIRED-USER COMMUNITY

The findings about the behavior of the Vaccine in a secure multi-paired-user community are the same as those of a single-paired-user environment. The reason stems from the fact that the vaccine performs the same function in a single-paired-user system as it does on each individual paired-user in the community. However, the re-keying and re-profiling are pure overhead and this is true anytime that a key agreement is used. To have a better understanding of this overhead cost we have completed the time complexity calculations for the three algorithms of Keys, Carve, and Profile, Table IV. The notations of L_k and N are used for the key length and the community size, respectively.

TABLE IV. TIME COMPLEXITY FOR ALGORITHMS KEYS, CARVE, AND PROFILE

Algorithm	Time Complexity
Keys	O(N): if $L_k < N$ $O(N^{\lceil L_k/N \rceil})$: otherwise
Carve	O(1)
Profile	*For Option = "10" and "11"* O(N): if $L_k < N$ $O(N^{\lceil L_k/N \rceil})$: otherwise
	For Option = "00" O(1)
	For Option = "01" O(N)

The findings reveal that the performance of the three algorithms in the worst case is linear to the size of the community assuming the size is larger than the length of the key given to the community members. The assumption is not far from reality and, in general, any linear growth in delivery of an algorithm with the large size of community is well accepted behavior.

VIII. CONCLUSION AND FUTURE RESEARCH

A new cipher block approach, Vaccine, for masking and unmasking of ciphertexts was introduced and implemented with one major goal in mind: Removal of inherited-features from a ciphertext. The methodology was first applied to a single-paired-user environment and the performance of the Vaccine was scrutinized by comparing it with the two well-known approaches of CBC and CFB modes. The advantages of the Vaccine application in a single-paired environment were numerated in Section V. The adoption of Vaccine for a multi-paired-user environment with the option of user revocation was also explored, which resulted in a methodology for re-keying

and re-profiling. One major property of the new keys and new profiles was that they were generated by starting with the creation of completely random and long binary strings. In general, using such strings provide a better protection for the keys and profiles against the adversarial attempts.

As future research, building a new version of the Vaccine is currently in progress to make the throughput and the masking strength of the methodology even higher. Study of Vaccine as an authentication method in a secure multi-paired-user environment has been scheduled.

REFERENCES

[1] R. Hashemi, A. A. Rasheed, A. Bahrami, and J. Young, "Vaccine: A Block Cipher Method for Masking and Unmasking of Ciphertexts' Features", The Second International Conference on Cyber-Technologies and Cyber-Systems (CYBER 2017), Barcelona, Spain, Nov. 2017, pp. 41-47.

[2] A. A. Rasheed, M. Cotter, B. Smith, D. Levan, and S. Phoha, "Dynamically Reconfigurable AES Cryptographic Core for Small, Power Limited Mobile Sensors", The 35th IEEE International Performance Computing and Communication Conference and Workshop, pp. 1-7, 2016.

[3] G. P. Saggese, A. Mazzeo, N. Mazzocca, and A. G. M. Strollo, "An FPGA-based performance analysis of the unrolling, tiling, and pipelining of the AES algorithm", LNCS 2778, pp. 292-302, 2003.

[4] N. Pramstaller and J. Wolkerstorfer, "A Universal and Efficient AES Co-processor for Field Programmable Logic Arrays", Lecture Notes in Computer Science, Springer, Vol.3203, pp. 565-574, 2004.

[5] B. Moeller. *Security of CBC Cipher suites in SSL/TLS: Problems and Countermeasures. [Online]*. Available from: https://www.openssl.org/~bodo/tls-cbc.txt

[6] W. Stallings, "Cryptography and Network Security: Principles and Practice", Pearson, 2014.

[7] C. A. Henk and V. Tilborg, "Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial", Springer Science & Business Media, 2006.

[8] N. Feruson, B. Schneier, and T. Kohno, "Cryptography Engineering: Design Principles and Practical Applications", Indianapolis: Wiley Publishing, Inc., pp. 63-64, 2010.

[9] W. F. Ehrsam, C. H. W. Meyer, J. L. Smith, and L. W. Tuchman, "Message Verification and Transmission Error Detection by Block Chaining", US Patent 4074066, 1976.

[10] C. Kaufman, R. Perlman, and M. Speciner, "Network Security", 2nd ed., Upper Saddle River, NJ: Prentice Hall, p. 319, 2002.

[11] National Institute of Standards and Technology (NIST), Advanced Encryption Standard (AES), Federal Information Processing Standards Publications 197 (FIPS197), Nov. 2001.

[12] S. Vaudenay, "Security Flaws Induced by CBC Padding — Applications to SSL, IPSEC, WTLS...", Lecture Notes in Computer Science, Springer, vol. 2332, pp. 534-546, 2002.

[13] S. Tanaka and F. Sato, "A Key Distribution and Rekeying Framework with Totally Ordered Multicast Protocols", Proceedings of the 15th International Conference on Information Networking, 2001, pp. 831-838.

[14] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman Key Distribution Ex-tended to Group Communication", Proceedings of the 3rd ACM Conference on Computer and Communication Security, 1996, pp. 31-37.

[15] C. Becker and U. Wille, "Communication Complexity of Group Key Distribution", In 5th ACM Conference on Computer and Communication Security, 1998, pp. 1-6.

[16] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system", Advances in Cryptology - EUROCRYPT'94, 1994.

[17] M. Steiner, G. Tsudik, and M. Waidner, "CLIQUES: A New Approach to Group Key Agreement", Proceedings of the 18th Internal Conference on Distributed Computing Systems, 1998, pp. 380-387.

[18] G. Ateniese, M. Steiner, and G. Tsudik, "New Multiparty Authentication Services and Key Agreement Protocols", IEEE Journal on Selected Areas in Communications, Vol. 18, No. 4, 2000, pp. 628-639.

[19] Y. Kim, A. Perrig, and G. Tsudik, "Communication-efficient group key agreement", 17th International Information Security Conference, 2001.

[20] H. Kuo-Tsang, C. Jung-Hui, and S. Sung-Shiou, "A Novel Structure with Dynamic Operation Mode for Symmetric-Key Block Ciphers", International Journal of Network Security & Its Applications, Vol. 5, No. 1, p. 19, 2013.

[21] H. Feistel, "Cryptography and Computer Privacy", Scientific American, Vol. 228, No. 5, pp 15–23, 1973.

[22] F. Charot, E. Yahya, and C. Wagner, "Efficient Modular-Pipelined AES Implementation in Counter Mode on ALTERA FPGA", (FPL 2003), Lisbon, Portugal, pp. 282-291, 2003.

[23] S. Mittra, "Iolus: A Framework for Scalable Secure Multicasting", Proceedings of the ACMSIGCOMM , 1997, pp. 277-288.

[24] D. Wallner, E. Harder, and R. Agee, "Key Management for Multicast: Issues and Architectures", RFC 2627, 1999.

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601