TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
25/2019

# Specialized Cyber Red Team Responsive Computer Network Operations

BERNHARDS BLUMBERGS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Software Science
The dissertation was accepted for the defence of the degree of Doctor of Philosophy (cyber security) on 2nd of April, 2019

**Supervisor:** Dr. Rain Ottis,
Department of Software Science, School of Information Technologies,
Tallinn University of Technology
Tallinn, Estonia

**Co-supervisor:** Dr. Risto Vaarandi
Department of Software Science, School of Information Technologies,
Tallinn University of Technology
Tallinn, Estonia

**Opponents:** Professor Dr. Hiroki Takakura,
National Institute of Informatics,
Tokyo, Japan

Fregattenkapitän PD Dr. Dr. habil. Robert Koch,
Bundeswehr University of Munich,
Munich, Germany

**Defence of the thesis:** 27th of May, 2019, Tallinn

**Declaration:**
*Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology, has not been submitted for any academic degree elsewhere.*

Bernhards Blumbergs _____
signature

European Union
European Social Fund
Investing
in your future


ISSN 2585-6898 (publication)
ISBN 978-9949-83-413-6 (publication)
ISSN 2585-6901 (PDF)
ISBN 978-9949-83-414-3 (PDF)

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
25/2019

# Vastutegevusele orienteeritud punase meeskonna küberoperatsioonid

BERNHARDS BLUMBERGS

TAL TECH
KIRJASTUS

# Contents

## LIST OF PUBLICATIONS

The thesis is based on the following publications:

**Publication I**: B. Blumbergs, M. Pihelgas, M. Kont, O. Maennel, and R. Vaarandi. Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels. In B. B. Brumley and J. Röning, editors, *Secure IT Systems: 21st Nordic Conference, NordSec 2016*, pages 85–100, Oulu, Finland, November 2016. Springer International Publishing

**Publication II**: B. Blumbergs and R. Vaarandi. Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 707–712, Baltimore, USA, November 2017. IEEE

**Publication III**: B. Blumbergs. Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 88–99, Prague, Czech Republic, February 2019. SCITEPRESS

**Publication IV**: R. Vaarandi, B. Blumbergs, and E. Çalışkan. Simple event correlator - Best practices for creating scalable configurations. In *2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 96–100, Orlando, USA, March 2015. IEEE

**Publication V**: R. Vaarandi, B. Blumbergs, and M. Kont. An Unsupervised Framework for Detecting Anomalous Messages from Syslog Log Files. In *IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, Taipei, Taiwan, April 2018. IEEE

**Publication VI**: A. Fárar, H. Bahsi, and B. Blumbergs. A Case Study About the Use and Evaluation of Cyber Deceptive Methods Against Highly Targeted Attacks. In *Proceedings of Cyber Incident 2017*, pages 1–7, London, UK, June 2017. IEEE

**Publication VII**: M. Kont, M. Pihelgas, K. Maennel, B. Blumbergs, and T. Lepik. Frankenstack: Toward Real-time Red Team Feedback. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 400–405, Baltimore, USA, November 2017. IEEE

**Publication VIII**: M. Schmitt, L. Vihul, D. Akande, G. Brown, P. Ducheine, T. Gill, W. Heintschel von Heinegg, G. Hernandez, D. Housen-Couriel, Z. Huang, E. Talbot Jensen, K. Kittichaisaree, A. Kozik, C. Kreiss, T. McCormak, K. Nakatani, G. Rona, P. Spector, S. Watts, and B. Blumbergs. *Tallinn Manual 2.0 on the International Law Applicable to Cyber Operations*. Cambridge University Press, Cambridge, UK, 2017 (This publication has not been included in the appendices of this thesis due to the publishing house copyright restrictions.)

**Publication IX**: D. Mucci and B. Blumbergs. TED: A Container Based Tool to Perform Security Risk Assessment for ELF Binaries. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 361–369, Prague, Czech Republic, February 2019. SCITEPRESS

**Publication X**: B. Blumbergs, R. Ottis, and R. Vaarandi. Crossed Swords: A Cyber Red Team Oriented Technical Exercise. In *18th European Conference on Cyber Warfare and Security, ECCWS 2019*, Coimbra, Portugal, July 2019. ACPI. (Accepted paper)

## AUTHOR'S CONTRIBUTIONS TO THE PUBLICATIONS

Contributions to the publications in this thesis are:

**I** In Publication I, as the main and leading author of this publication, the author proposed a problem that IPv6 based transition mechanisms can be abused for undetectable covert channel establishment. The author developed, described and prototyped the IPv6 transition mechanism-based covert channels, created and published the *nc64* and *tun64* tools. The author designed the test network, implemented common covert channel mechanisms for comparison, and provided the guidelines and requirements to the evasion detection team. Additionally, the author successfully applied the developed *nc64* and *tun64* tools in practice within the NATO CCD CoE executed cyber defence exercise "Locked Shields".

**II** In Publication II, as the main and leading author of this publication, the author identified the problem of analysing and attacking the binary network protocols. The author developed, described and prototyped the bit-aware fuzzing framework *Bbuzz* for binary network protocol reverse-engineering, which requires the minimum effort from the human expert to start the network protocol reverse-engineering or vulnerability identification. The author addressed the problem by introducing one bit as the smallest unit for fuzzing test-case creation, implemented automated network protocol sample analysis and test-case creation. Additionally, the author used the developed methodology and prototyped *Bbuzz* tool to successfully reverse engineer the NATO Link-1 binary protocol to inject fake aeroplane tracks on the radar screen.

**III** In Publication III, as the main and the only author for this publication, the author discovered vulnerabilities in major industrial Ethernet protocols (PROFINET IO, IEC-104) and devices (Martem GW6e-TELEM). The author performs and describes the reverse-engineering of the industrial control system network protocols, discloses technical details on identified vulnerabilities, addresses their mitigation by reporting to the vendor and the security community, and proposes the methods for critical information infrastructure protection. Additionally, the author implemented the found vulnerabilities and designed the attacks into the NATO CCD CoE technical exercises "Locked Shields" and "Crossed Swords" as a part of cyber red team attack campaign.

**IV** In Publication IV, as the co-author of this publication, the author provided attack test cases, their execution approaches from the attacker perspective, and the validation of results.

**V** In Publication V, as the co-author of this publication, the author provided the *Bbuzz* tool developed by the author to conduct the experiments against the corporate production network systems, guided on the tool implementation, its use cases and performed the assessment of the results for the *Bbuzz* test cases. Additionally, the application of the *Bbuzz* tool allowed to uncover unexpected system log and error messages, allowing to further fine-tune the detection of anomalous messages.

**VI** In Publication VI, as the co-author of this publication, the author developed a multi-segmented network for the experiment execution, guided the deception mechanism deployment according to the cyber kill-chain phases, and provided guidelines for the security expert engagement for the verification of the implemented cyber deception effectiveness.

**VII** In Publication VII, as the co-author of this publication, the author proposed the problem of near real-time feedback necessity to increase the training benefit for the cyber red team participants. The author created, designed and led the development of the "Crossed Swords" exercise network, which was used for the real-time data acquisition of the executed attacks. Furthermore, provided the requirements for cyber red team situational awareness feedback and and expected result guidelines to the group of technical experts working on creating the *Frankenstack*. Additionally, provided guidance for the tool assessment and applicability, and performed the validation and verification of the implemented solution for the real-time cyber red team feedback. Moreover, the *Frankenstack* was successfully implemented and tested in the NATO CCD CoE technical exercise "Crossed Swords".

**VIII** In Publication VIII, the author was the only technical expert and advisor to the international group of legal experts, who were working on describing the international law applicability to cyber operations. The author was engaged in discussions and drafting of the technical scenarios for various cyber operations for legal analysis, advised on the technical principles of cyberspace and operations, reviewed the manuscript at its drafting stages, and performed a thorough review and update of the technical scenarios presented in the first edition of the manual.

**IX** In Publication IX, the author proposed an idea of automated solution with minimal dependencies to be used for system security baselining, vulnerability assessment, and incident response, locally on the system itself. Author contributed with the ideas and suggestions of applicable tool inclusion into the solution, produced result interpretation and representation. Additionally, author is one of the core developers of the "Locked Shields" game network, where the solution was tested to estimate the deployed virtual machine security level before and after their protection. Furthermore, the author led a cyber red teaming campaign against the game network and deployed systems during the exercise, before the security evaluation was conducted for these systems.

**X** in Publication X, as the main and leading author of this publication, the author explains the cyber red team oriented full-spectrum technical exrcise "Crossed Swords" design and development, as well as the core principles of exercise execution and conduct of an offensive cyber-kinetic operation. The author is the creator and technical director for the exercise since 2014.

## Acronyms

| | |
|---|---|
| ACD | Active Cyber Defence |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| APT | Advanced Persistent Threat |
| BT | Blue Team |
| CDO | Cyber Deception Operation |
| CDX | Cyber Defence Exercise |
| CI | Critical Infrastructure |
| CII | Critical Information Infrastructure |
| CNA | Computer Network Attack |
| CnC | Command and Control |
| CND | Computer Network Defence |
| CNE | Computer Network Exploitation |
| CNO | Computer Network Operation |
| CRT | Cyber Red Teaming |
| CVE | Common Vulnerabilities and Exposure |
| DDoS | Distributed Denial of Service |
| DLP | Data Loss Prevention |
| DoS | Denial of Service |
| DPI | Deep Packet Inspection |
| GRE | Generic Routing Encapsulation |
| HIDS | Host-based Intrusion Detection System |
| HVAC | Heating, Ventilation, and Air-Conditioning |
| ICS | Industrial Control System |
| IDS | Intrusion Detection System |
| IFF | Identify Friend or Foe |
| IoC | Indicator of Compromise |
| ISP | Internet Service Provider |
| IT | Information Technology |
| LAN | Local Area Network |
| NIDS | Network-based Intrusion Detection System |
| OODA | Observe, Orient, Decide, Act |
| OPSEC | Operational Security |
| OSINT | Open Source Intelligence |
| PLC | Programmable Logic Controller |
| PoC | Proof of Concept |
| RCD | Responsive Cyber Defence |
| RT | Red Team |
| RTU | Remote Terminal Unit |
| SCADA | Supervisory Control and Data Acquisition |
| SOF | Special Operations Forces |
| TTPs | Techniques, Tools and Procedures |
| TTTPs | Techniques, Tools, Tactics and Procedures |
| VPS | Virtual Private Server |
| WAN | Wide Area Network |

# List of Figures

# List of Tables

“You take the red pill – you stay in Wonderland, and I show you how deep the rabbit hole goes. Remember: all I’m offering is the truth.”

– Morpheus, *The Matrix*

To my white rabbit, whom I follow.

# 1 INTRODUCTION

With the accelerating expansion and development of the Internet and rapid increase of amounts of interconnected nodes, the cyberspace has become one of the most relevant domains not only for information exchange, integration and merger of systems, but also for malicious activities. The increasing volume of revealed sophisticated and debilitating attacks against Critical Information Infrastructure (CII), such as, Stuxnet [33], Duqu [74], Night Dragon [50], GhostNet [14], Red October [30], Shamoon [26], EnergeticBear/Dragonfly [77], BlackEnergy [27], and NotPetya [76] represent the significance of the cyber domain. Sophisticated cyber attacks are constantly forcing changes in the core concepts of cyber security, such as, defensive measures, situational awareness, threat analysis, and response. Technological advancements in areas, such as, intrusion detection, artificial intelligence and big data analysis, have allowed the security community to reveal previously undetectable persistence of threat actors within the information systems. This is represented by the growing numbers of identified threat actors in cyberspace [24] [58] and shows the interest in cyber domain both from state-affiliated and non-affiliated parties.

CII components, such as, Industrial Control System (ICS) and Supervisory Control and Data Acquisition (SCADA) networks are of an essential interest to the state-affiliated and non-affiliated actors, organized crime, terrorists, and malicious insiders, and are the prime targets for cyber warfare [64]. These systems, primarily being designed for dependency and fail-safe, possess common vulnerabilities in data handling, security administration, architecture design, network implementation, and deployed platforms [73]. World energy council has acknowledged the significance of cyber security and recognized cyber attacks as a substantial threat to CII and energy sector [87]. However, the accepted response so far is generally limited to traditional information assurance principles, such as, system hardening and defence in-depth.

The acknowledgement of cyberspace as an operational domain by NATO [52] is the first step towards considering not only military implications, but also realizing that we are already in the age of full spectrum cyber operations and cyber warfare. Furthermore, establishment of NATO Cyberspace Operations Centre (CyOC)[1] [2] directly points to the allied effort for having a stronger cyberspace presence and integration of such operations for allied system defence by year 2023. Robinson, Jones and Janicke [70] acknowledge the dual use of the cyber warfare and present the core principles, which include the lack of physical limitations, stealth, mutability, delivering kinetic effects, infrastructure control, attribution and deterrence challenges, and information as operational environment. Nicholson et.al. [64] recognize, that cyber warfare can be used both for attacking and defending information systems in cyberspace, proposing penetration testing and war-gaming as one of the protection mechanisms. Cyberspace has already become the domain to be reckoned with, and now attracting an even more increased interest on exploring the nature and operational benefits of this domain.

The investment into the cyber weapon, related tool and technique development is increasing both from the nation state and private industry side. Major commercial ini-

[1] NATO. Cyber Defence. `https://www.nato.int/cps/en/natohq/topics_78170.htm`. Accessed: 29/10/2018

[2] Breaking Defense. "NATO To 'Integrate' Offensive Cyber By Members". `https://breakingdefense.com/2018/11/nato-will-integrate-offensive-cyber-by-member-states/`. Accessed: 21/11/2018

tiatives, such as, Zerodium[3], FinFisher[4], Hacking Team[5], Google Project Zero[6] and Ozeda Group[7] strive to develop and offer the exploit (e.g., zero-days) acquisition for governmental and intelligence agencies. This market saturation clearly represents the interest and demand for such solutions. It is very hard to assess and estimate to whom these exploits are being sold, who are already using them, and if their detection mechanisms are already developed by those who have purchased these cyber weapons. This indicates that whoever wants to compete in cyberspace should have the required capabilities to develop own techniques and tools, as well as ability to adjust the available ones according to the operational requirements and circumvent detection. Additionally, government sponsored cyber black-ops, such as, the leaked CIA and NSA cyber weapon tool-kit by the Shadow Brokers[8], represents that nations possess such capabilities and are investing resources in cyber weapon development. This should not be surprising, since such capability establishment is in the interests of nations willing to maintain an advantage in the cyberspace and protect own assets. Cyber red team establishment, required tool-set development, and cyber operation execution is a capability typically under a very strict control and not being exposed by the nation states or sophisticated threat actors. Furthermore, the recent revelation[9] on the identified Russian Federation's Main Intelligence Directorate (RF GRU Cyber Unit 26165) executed cyber attacks against international organizations. These attacks against institutions, such as, the Organization for the Prohibition of Chemical Weapons (OPCW) and Spiez Laboratory, have indicated the NATO member nations are practising responsive cyber defence. As well as, the United States of America countering the RF executed series of cyber attacks, dubbed as the "Dragonfly", against the US energy sector ICS/SCADA systems and its supply chain [77]. Moreover, European Central Bank has released the framework for threat intelligence-based ethical red teaming (TIBER-EU) [21], to test and improve the European financial institution resilience against sophisticated cyber attacks. This further recognizes the cyber red teaming, albeit from a defensive posture, as an applicable approach, which requires further research and development.

## 1.1 Problem Statement

This thesis addresses the problems in the areas of response to the asymmetric threats, cyber red team capability development and computer network operation execution, and technical exercise development tailored to meet the requirements of an operational cyber red team.

*Responding to asymmetric threats.* Responding to the targeted sophisticated cyber attacks executed by a technologically advanced nation state, or threat actor, is becoming ever harder with traditional cyber defence approaches, placing defenders at a losing position when compared to such an adversary. Unconventional, hybrid and asymmetric threats in cyberspace demand an equally executed response. However, such asymmetric response to a stronger adversary needs to be understood better from various perspec-

[3]Zerodium. `https://zerodium.com/`. Accessed: 28/09/2018

[4]FinFisher. `https://finfisher.com/FinFisher/index.html`. Accessed: 28/09/2018

[5]Hacking Team. `http://www.hackingteam.it/`. Accessed: 28/09/2018

[6]Project Zero. `https://googleprojectzero.blogspot.com/`. Accessed: 28/09/2018

[7]Ozeda. `http://www.ozeda-group.com/`. Accessed: 28/09/2018

[8]Vault 7: CIA Hacking Tools Revealed. `https://wikileaks.org/ciav7p1/`. Accessed: 03/10/2018

[9]Press conference by NATO Secretary General Jens Stoltenberg following the meetings of NATO Defence Ministers. `https://www.nato.int/cps/en/natohq/opinions_158705.htm`. Accessed: 04/10/2018

tives, for example, technical, operational, and legal.

*Cyber red team capabilities and operations.* Building the cyber red team capability is every nation's internal process, which is not publicly discussed and therefore not well understood when compared to building conventional armed forces and their general capabilities. Understanding such capabilities is especially important when considering offensive, responsive or active computer network operations. The myth of cyber red teaming being a nation's "secret cyber capability" has to be dispersed and such capability general requirements, applicability and operational conditions have to be researched openly. There is a lack of verified knowledge regarding cyber red team capability development, execution of various cyber operations, including computer network operations, design of such operations, execution, management and governance. Lewis [35] stipulates that few of the NATO member nations tend to be overly secretive regarding their offensive cyber capabilities, even once they have been leaked to the public, with an intention of increasing the likelihood of adversary performing incorrect judgement, when exploring the use of force against the NATO and its allies. This lack of public information and no accepted procedures or doctrines regarding offensive or responsive cyber defence operations can be reasoned from the perspective, that the cyber operations and cyberspace as a domain of operations have been only recently introduced, when compared to land, maritime and air warfare, and it is yet not entirely clear how such environments can be fully utilized, proper capabilities developed, and operations executed.

*Cyber red team readiness and technical exercises.* The various types of trainings (e.g., table-top exercise, decision-making training, and crisis management exercise) have been attempting to integrate cyber component, also specific cyber defence technical exercises are being created aimed at increasing awareness and defensive capacity. Such publicly discussed and advertised trainings and exercises are oriented mainly towards practising cyber defence, with the red team performing adversary simulation. Cyber red team oriented exercises either are not discussed due to its exaggerated political sensitivity, are limited to a very narrow participant group, or exercised within the nation itself being classified. Such lack of openly available information hinders the cyber red team capability evolution and along with it – obscures the progress of defensive measures against a real adversary.

The aim of the thesis is to define specialized cyber red teaming and develop novel Techniques, Tools, Tactics and Procedures (TTTPs), which are adapted to meet the requirements of the Responsive Cyber Defence (RCD) to deliver the desired unconventional thinking and asymmetric response through Computer Network Operation (CNO) against a persistent, well-motivated and advanced threat actors. Responsive cyber defence, defined in Definition 1 on page 33 and explained in more detail in chapter 3.1, is any acceptable activity performed to ensure the protection of the defended systems in response to an incoming or imminent attack. Cyber Red Teaming (CRT) activity is a coordinated and timely reaction, either responsive or pre-emptive, to a threat in cyberspace to ensure defended information system resiliency, security and integrity. Responding to various and changing threats requires a prompt adaptation of novel TTTPs by the cyber red team. This is even more important in the case when responders are in an unequal position, being outnumbered and sometimes with fewer resources available. A specialized CRT responsive operation is an operation which is characterized by at least the following factors: fast-paced, high stealth, small team with multiple specializations, remote attacks with no or very limited physical access to target, customized tool-set, specific TTTPs, and engaged either for responsive or proactive defence operations. Definition 3 on page 37 defines and further explains specialized CRT. To meet the demands of such operation execution, the red team needs to be selected, trained, and their skills need to be kept con-

stantly up to date. Such goals can be met by integrating technical cyber exercises into the CRT development and maintenance life-cycle and by providing the training experience as close to the real operations as possible.

## 1.2 Research Questions

The following research questions address the cyber red teaming capability development and evolution, adversary detection and deception solution applicability to cyber red team activities, and training within technical cyber exercises to practice or prepare for real-life responsive computer network operations. To further clarify the research questions, they are split into sub-questions aiming at seeking more precise answers and addressing more particular nuances of the matter. The question is answered by providing the answers to its sub-question. The research question mapping to the particular publication and to the thesis chapter, where the question is explored and answered, is presented in table 2.

**Q1.** *What are the specialized cyber red team technical capabilities for responsive computer network operations?*

**Q1.1.** *Which features the techniques, tools, tactics and procedures have to possess to be applicable to the specialized cyber red team responsive computer network operation requirements?*

**Q1.2.** *How these proposed techniques, tools, tactics and procedures are suitable to counter the cyber attack kill-chain?*

For the cyber red team to be able to adjust to the operational requirements, such as, stealthy network access, setting up a covert command and control channel, and completing the objectives, proper techniques and tools need to be identified and developed. This question explores the features of the applicable techniques required to enable the cyber red team to build and expand the available arsenal, which is custom tailored to meet the requirements of every particular operational objective. Additionally, the operation execution by the red team to a certain extent would follow the cyber kill-chain [37] phases for which the target network might have implemented relevant protection methods. It is important to assess how can the proposed techniques, tools and procedures be employed to counter the cyber attack kill-chain and what is their suitability for the computer network operation execution. Furthermore, such responsive cyber activities, besides technical considerations, retain legal concerns. The legal implications of such activities need to be introduced. Techniques and tools, explained in the listed research publications (Publication I, Publication II, Publication III), focus on assessing their delivered effects and applicable usage procedures, thus granting the foundation for Cyber Red Teaming (CRT) novel technical capability development. Legal ramifications can be estimated, and guidance obtained from the published work (Publication VIII).

**Q2.** *How to establish the situational awareness for the cyber red team?*

**Q2.1.** *How system log file analysis and cyber deception approaches are relevant to cyber red team work-flow when executing responsive cyber operations?*

**Q2.2.** *In what ways such solutions are applicable to situational awareness, red team operational infrastructure protection, and attack technical attribution?*

Before commencing with any responsive actions within the cyberspace it is crucial to establish the situational awareness, understand if the systems are under attack, perform the

initial technical attribution, and then execute the responsive operations to defend against the threat. For this to be possible, proper tools need to be integrated into the cyber red team's work-flow and environment, allowing high level of automation, easy and scalable deployment, and straightforward customization. To accomplish this, the delivered effects of the system log file-based anomaly and threat detection solutions, and cyber deception techniques need to be assessed for their applicability to complement the cyber red team's operational capabilities. As well as, how such approaches allow the protection of the red team's operational environment to present the defenders with an opportunity to gain advantage over the adversary. The applicable solutions are evaluated based upon listed research publications (Publication IV, Publication V, Publication VI, Publication IX).

**Q3.** *How to prepare and train the cyber red team for responsive computer network operation execution?*

**Q3.1.** *What considerations are applicable for training cyber red team as close to the real-life conditions as possible in a technical cyber exercise?*

**Q3.2.** *What means can be used to assess the cyber red team training objective achievement in near real-time?*

To prepare the cyber red team for the real responsive computer network operations, or to carry out the team selection and assessment, a proper training environment needs to be created. Technical exercises oriented at CRT should be considered delivering maximum possible realism, including fitting scenario, elaborate technical infrastructure, implemented challenges, chain of command, training objectives, evaluation of training goal achievement, and near real-time feedback provision to the training audience. This question explores the ways how such technical exercise can be designed, executing a full-spectrum cyber operation implementing all of the previously mentioned requirements for the CRT technical arsenal, Techniques, Tools, Tactics and Procedures (TTTPs), legal aspects, threat detection and deception mechanisms. The listed research papers (Publication III, Publication VII) provide the required assessment on real-life attack implementation and near real-time feedback provisioning to the training audience.

| Research Question Number | Publication Number | Chapter Number |
|---|---|---|
| Q1. | Publication I | 4.1, 4.2, 4.2.4 |
| | Publication II | 4.1, 4.2, 4.2.4 |
| | Publication III | 4.1, 4.2, 4.2.4 |
| | Publication VIII | 6.4 |
| Q2. | Publication IV | 5 |
| | Publication V | 5 |
| | Publication VI | 5 |
| | Publication IX | 5 |
| Q3. | Publication VII | 6 |
| | Publication III | 6 |
| | Publication X | 6 |

*Table 2: Mapping of research questions and publications to the thesis chapters*

## 1.3 Contribution

This thesis is based on a collection of cited and published publications in international conferences and by publishing houses (IEEE, Springer, and Cambridge University Press). Thesis explores the principles and approaches for the Cyber Red Teaming (CRT) training and required Techniques, Tools, Tactics and Procedures (TTTPs) development allowing to respond to an advanced adversary in an asymmetric manner. The Computer Network Operation (CNO) activities against the attacker are mounted within the responsive cyber defence paradigm, where the principle of a response to an ongoing or imminent threat is exercised. Computer network operations, defined in Definition 2 on page 35 and explored in more detail in chapter 3.2, constitute any set of effect-oriented activities performed by the application of force in the cyberspace by the use of computer networks. In addition to technical aspects of such operations, a brief introduction to legal implications is presented, providing a broader view of the responsive actions executed by the cyber red team. It has to be noted, that rational response can be exercised once enough technical attribution data is gathered allowing identification of the attack, its source, or by profiling the attack patterns and adversary operational actions. The threat detection solutions are evaluated from the perspective of the effects delivered and their integration into the cyber red team work-flow to provide the required operational input and awareness. Furthermore, the principles on training, exercise design, preparing and assessing the red team are considered from the perspective of technical cyber exercises, where all aforementioned aspects are combined to provide the maximum training effect.

The publications Publication I, Publication II, and Publication III are linked together to cover the novel TTTPs applicable to cyber red team activities. The publications Publication IV, Publication V, Publication VI, and Publication IX provide the detection and cyber deception concepts suitable for CRT asset protection, adversary tracking, and technical attribution evidence gathering. Furthermore, publications Publication X, Publication VII, Publication III, and Publication VIII, being supported by all listed publications, represent the cyber red team oriented technical exercise design, development, near real-time cyber red team feedback provision, execution aspects, and exercised activity legal considerations.

The main contributions of this thesis are:

1. A thorough review of operational requirements and proposal of new definitions for Responsive Cyber Defence, Computer Network Operations, and Cyber Red Teaming. This addresses the inconsistency in the various definitions for these concepts, attempts to crystallise the integral ideas, and propose a more thorough new definitions. Furthermore, minimal operational requirements and features for the Responsive Cyber Defence (RCD), CNO, and CRT are assessed and presented;

2. Definition of a new concept for “specialized cyber red team responsive computer network operations.” This focuses on the operational requirements and novel ideas of such operations, with appropriate techniques, tools, tactics and procedures proposed for accomplishing them;

3. Principles and approaches for the development of novel techniques, tools, tactics and procedures to enhance specialized cyber red team responsive computer network operations. Aiming at novel conceptions allowing to develop new TTTPs tailored for the CRT operational requirements. The proposed methods are assessed from their applicability and delivered effect perspective;

4. Review of accepted cyber kill-chain models and a proposal for a new comprehensive “cyber attack kill-chain” model with applicable techniques suggested for countering

the kill-chain. Directed at assessing the strengths and drawbacks of the current accepted models and unifies them to provide a more thorough cyber attack kill-chain. Furthermore, the developed TTTPs are assessed for applicability to bypass the proposed kill-chain at every stage;

5. Concepts for integration of detection and deception to aid the cyber red team operations, provide technical attribution, adversary tracking, threat assessment, and offer new considerations and means for cyber red team operational asset protection. The goal is directed towards integration of novel defensive techniques into cyber red team work-flow and operational infrastructure;

6. Creation of a new technical exercise oriented at cyber red team skill development in a close to real-life conditions. The focus is on defining the design requirements for cyber red team aimed at full-spectrum technical exercise creation, by integrating applicable TTTPs, attack detection and cyber deception methods. The development and novel design principles of a cyber red team oriented technical exercise created by the author is presented. As well as, represents how every listed publication fits into the exercise development and execution; and

7. Cyber red team training objective assessment and near real-time feedback provision through the development of the novel automated cyber attack detection, analysis, and representation framework *Frankenstack*. The aim is at delivering the novel monitoring and visualization solutions targeted at providing the best situational awareness picture possible to the exercise training audience to increase the training benefits and learning experience.

## 1.4 Thesis Structure

This thesis is divided into six chapters. The introduction chapter provides a brief overview of the threat landscape and provides the reasoning for cyber red team applicability to conduct asymmetric responsive computer network operations, as well as, the research questions and contribution of the thesis.

Chapter 2 gives an overview of related work and background in responsive cyber defence, computer network operations, cyber red teaming, and cyber red team oriented technical exercises;

Chapter 3 defines the underlying concepts for specialized cyber red team responsive computer network operations and their requirements;

Chapter 4 explores the relevant techniques and tool development approaches for the cyber red team responsive cyber defence operations, describes applicable tactics and procedures, and assesses such approach relevance to countering the cyber attack kill-chain.

Chapter 5 presents how anomaly detection and deception solutions are applicable to cyber red team activities and asset protection;

Chapter 6 describes the use case of cyber red team oriented full-spectrum technical exercise design, operational requirements, and introduces the legal ramifications of responsive cyber operations;

Chapter 7 concludes this thesis and presents ideas for future work.

# 2 BACKGROUND AND RELATED WORK

This chapter reviews the background and related work from academic and credible non-academic sources, such as, recognized international organizations, military institutions, and renowned think-tanks, in the areas of active and responsive cyber defence, cyber operations, cyber red teaming, and cyber red team oriented technical exercises. Additionally, the gaps are identified and contribution to address these issues is emphasized.

## 2.1 Responsive Cyber Defence and Initial Attribution

Dewar [17] thoroughly explores the concepts of Active Cyber Defence (ACD), which is being adopted by a number of international actors, allowing to identify and stop increasingly occurring cyber incidents, as well as take offensive measures to minimize the attackers' capabilities. Dewar proposes variety of technical solutions to detect, analyse, identify and mitigate threats in real-time, such as, decoys, hacking back, *"white worms"*, address hopping and honey pots. The research points out, that due to extraterritorial and aggressive nature of the ACD there are legal and political ramifications requiring special care when exerting this approach. Furthermore, publication implies that Computer Network Operation (CNO), ACD and non-ACD actions are placed alongside and all fall under the cyber defence category, where all, except CNO, can be used both in peace- and war-time. CNO being exclusively limited to cyber warfare or military engagements employed either pre-emptively or deployed in advance of a kinetic manoeuvre. It is stated, that "ACD can be considered an approach to achieving cyber security predicated upon the deployment of measures to detect, analyze, identify and mitigate threats to and from communications systems and networks in real-time as well as the malicious actors involved." [17] In a recent US Military Joint Publication [80] the new term "Defensive Cyberspace Operation – Responsive Activities" is introduced to represent external cyberspace operations and defined as "[o]perations that are part of a defensive cyberspace operations mission that are taken external to the defended network or portion of cyberspace without the permission of the owner of the affected system". This definition overlaps with the concept of Responsive Cyber Defence (RCD). Additionally, Mansfield [46] discusses the ACD and hacking the hackers from the perspective of botnet takeover, infiltration, controlled destruction and hacking back. And Repik [68] explores the techniques for dynamic network reconfiguration (e.g., address hopping) and decoys (e.g., honeypots, network telescopes) as the means of ACD.

From strategic and legal perspective, Denning [16] looks at active and passive air and missile defence concept applicability to cyberspace. Denning defines the ACD as "direct defensive action taken to destroy, nullify, or reduce the effectiveness of cyber threats against friendly forces and assets" [16] and is characterized by its scope of effects, degree of cooperation, types of effects, and degree of automation. Furthermore, ACD ethical and legal principles are addressed, such as, authority, third-party immunity, necessity, proportionality, human involvement, and civil liberties. Bradbury [11] discusses the principles of ACD and questions its necessity, legal and ethical considerations. Bradbury concludes, that ethical and legal considerations are shifting depending on who is executing ACD. Furthermore, Schmitt [72] considers seven factors of ACD – severity, immediacy, directness, invasiveness, measurability, presumptive legitimacy, and responsibility. Focusing on effects-based approach that looks at the result of an attack, rather than what is used.

Publicly announced nation state initiatives, such as, US DARPA[10] has initiated an ACD project with a scope to develop a collection of synchronized, real-time capabilities to dis-

[10]DARPA. "Active Cyber Defence (ACD) (Archived)". `https://www.darpa.mil/program/active-cyber-defense`. Accessed: 19/09/2018

cover, define, analyse and mitigate cyber threats and vulnerabilities, enabling cyber defender readiness to disrupt and neutralize cyber attacks as they happen, however, these capabilities would be solely defensive in nature and specifically excludes research into cyber offence capabilities. Also, US National Security Agency defines the ACD as "[e]nabling the real-time defense of critical national security networks by leading the integration, synchronization, and automation of cyber defense services and capabilities."[11] It is worth mentioning, the ADHD[12], which is a GNU/Linux distribution collecting the most common tools and solutions to be used for ACD. Additionally, UK's National Cyber Security Centre (NCSC)[13] has started the Active Cyber Defense programme in addressing the cyber attacks in near real-time.

From a strictly theoretical perspective, Lu, Xu and Yi [41] perform the mathematical modelling of ACD based on computer malware and biological epidemic game-theoretic models. In this study the authors acknowledge the *"white worms"* as the only ACD measure. Lu et. al. determine, that defence based on Intrusion Detection System (IDS), firewalls and anti-malware tools is reactive and fundamentally asymmetrical. This can be eliminated by the use of ACD, offering strategic interaction between the attacker and the defender.

In the most significant identified work related to responsive cyber defence, Maybaum et.al. [49] explore the concepts beyond the ACD and address the RCD as a new approach to counter the increasing number of threats that Information Technology (IT) systems have to face. The technical research paper assesses various tools and approaches which can be used for the RCD not only from the technical perspective, but also taking into account the legal implications. Maybaum et.al. define RCD as "[...] any activities conducted to defend one's own IT systems or networks [...] against an on-going cyberattack by gaining access to, modifying or deleting data or services in other IT systems or networks [...] without the supposed or actual consent of their rightful owners or operators." To complement it, Brangetto, Minárik and Stinissen [13] explore the legal implications of RCD from the international and domestic law perspectives. Brangetto et.al. consider this approach only applicable to states and define it as "[...] the protection of a designated Communications and Information System (CIS) against an ongoing cyberattack by employing measures directed against the CIS from which the cyberattack originates, or against third-party CIS which are involved." Article elaborates, that RCD can be seen as the subset for ACD, but RCD is conducted only against actual and ongoing cyber attack, and it cannot be pre-emptive or retaliatory.

It is worth mentioning, an Artificial Intelligence (AI) technologies implemented for fighting cyber attacks in real-time by the *Darktrace*[14]. This approach implements the human immune system principles into an AI driven solution to fight malware and cyber attacks originating from the external networks and malicious insiders. Darktrace elaborates, that nowadays, both the cyber attackers and defenders use the AI technologies either to conduct the attacks or to deliver the defence with a higher levels of sophistication.

**System log file analysis.** The vast topic of log analysis is tackled only from the perspective of unsupervised detection of anomalous messages from system logs as benefiting the

[11]NSA CSS. Active Cyber Defence (ACD). `https://www.iad.gov/iad/programs/iad-initiatives/active-cyber-defense.cfm`. Accessed 19/09/2018

[12]Active Defense Harbinger Distribution. `https://adhdproject.github.io/`. Accessed: 15/09/2018

[13]UK GCHQ NCSC. Active Cyber Defense. `https://www.ncsc.gov.uk/active-cyber-defence`. Accessed: 16/10/2018

[14]Darktrace. `https://www.darktrace.com/en/`. Accessed: 19/10/2018

cyber red team activities and does not constitute the entire literature. Xu, Huang, Fox, Patterson and Jordan [88] suggest an unsupervised method for anomalous event sequence detection using principal component analysis (PCA). By using source code analysis for detecting event type sequences, the vectors are derived with their attributes representing the number of events of a particular type in the sequence. Vectors with frequently occurring patterns are filtered out, assuming that they represent normal event sequences. Oliner, Aiken and Stearley [65] propose an unsupervised *Nodeinfo* algorithm, which analyses network node generated events from past days, divided into hourly frames (*nodehours*), for anomaly detection. Shannon information entropy-based anomaly score is calculated for each nodehour depending on log file word occurrences in network nodes. Fewer occurrences would generate a higher anomaly score. Du, Li, Zheng and Srikumar [19] propose the *DeepLog* algorithm, which uses long short-term memory (LSTM) neural networks for anomaly detection in event type sequences, by predicting the event type probability from previous types in the sequence. Yamanishi and Maruyama [90] suggest an unsupervised method for network failure prediction from system log error events. This is attempted by dividing the log into time-based sessions, which are modelled with hidden Markov mixture models, while model parameters are learned in an unsupervised manner. An anomaly score is calculated for each session and is considered as anomalous if its score exceeds a threshold. In addition to aforementioned methods, a number of other approaches have been suggested for system log anomaly detection, including clustering [43], invariant mining [40], and hybrid machine learning algorithms [36].

Additionally, It is worth mentioning an initial attempt for cyber red team infrastructure command and control server protection solution *RedELK*[15], which is aimed at implementing *ELK Stack* to aggregate the data from the Command and Control (CnC) servers and maintain the visibility over them. This solution is focused on a small fraction of the cyber red team's operational infrastructure protection and could be applicable to countering the CnC phase of the cyber attack kill-chain.

**Cyber deceptions.** Cyber deception is recognized as an applicable set of techniques for the active cyber defence. Provos [66] proposes the virtual honeypot framework design and implementation for tracking malware and deceiving malicious activities. Wagener et.al. [84] and Zhang et.al. [93] identify the shortcomings of traditional honeypots and explore the adaptive and self-configurable honeypots allowing its behaviour adjustment according to the adversary actions and source of incoming connection.

Heckman et.al. [29] [28] admits that traditional approaches to cyber defence are inadequate and explores denial and deception (D&D) as an option within ACD. Heckman proposes deceptions as part of a Cyber Deception Operation (CDO), such as, honeypots, honeyclients, honeytokens, and tarpits against an Advanced Persistent Threat (APT). This paradigm of cyber denial and deception aims at influencing attacker in a way that gives the deceiver an advantage, by forcing adversary to move more slowly, expend more resources, and take greater risks. The concepts were tested in a MITRE organized Red Team (RT) versus Blue Team (BT) real-time war-game experiment, where a MITRE developed *Blackjack* Computer Network Defence (CND) tool was employed for analysing adversary traffic by applying rules engine to enforce policy to each request. The exercise consisted of four teams (white, red, blue CND, and blue D&D), where the BT played the scenario of planning an attack against the enemy compound. Denial and deception techniques were found to be effective. Geers [25] proposes cyber attack deterrence by denial of capability, communication, and credibility. Fàrar, Bahsi and Blumbergs [23] explore the use

[15] RedELK – Red Team's SIEM. `https://github.com/outflanknl/RedELK`. Accessed: 19/10/2018

of cyber deceptions for network intrusion early warning, by focusing on stopping the attacks at their early stages of cyber kill-chain. Cymmetria *MazeRunner*[16] cyber deception solution delivers a novel approach by the use of digital *breadcrumbs* to deceive the adversary leading to its detection and analysis.

**Initial Attribution.** The active cyber defence employed methods, if successfully implemented can gather the required technical information allowing to perform the initial attribution and attempt the pursue of the adversary. Rid [69] introduces the *Q-model* for cyber attack attribution assessment consisting of: concept (tactical/technical, operational, strategic), practice (asking right questions, targeting analysis, language indicators, modality of code, infrastructure, mistakes, stealth, Computer Network Exploitation (CNE), stages of attack, geopolitical context, the form of damage inflicted), communication (releasing details on attack to boost the credibility, presenting more details to aid further attribution). Research proposes attributing offence to the offender to minimize the uncertainty at three levels: tactically, operationally, and strategically. Rid emphasizes that only a technical redesign of the Internet, consequently, could fully fix the attribution problem. European Union Council [20] provides suggestions for decision-making regarding a joint EU diplomatic response to a malicious cyber activity, which requires attribution. The limited working paper presents the following steps: information collection, information assessment, political decision, and designing a response.

## 2.2 Computer Network Operations

Robinson, Jones and Janicke [70] examine the concepts of cyber attack and cyber warfare. Robinson et.al. define cyber attack as "[a]n act in cyber space that could reasonably be expected to cause harm", and cyber warfare as "[t]he use of cyber attacks with a warfare-like intent". The researchers point out, that cyber war occurs only when nation states declare war and employ only cyber warfare. Research lists the following cyber warfare principles: lack of physical limitations, kinetic effects, stealth, mutability and inconsistency (e.g., the dynamic and changing nature of the cyberspace), identity and privileges (e.g., assumption of ones identity to gain the privileges), dual use (e.g., use of cyber weapons for defensive and offensive purposes), infrastructure control (e.g., control over the adversarial or third party systems to gain positional advantage), information as operational environment, attribution, defence and deterrence. Leblanc et.al. [34] assert that, per US Doctrine, Computer Network Operation (CNO) is comprised of Computer Network Defence (CND), Computer Network Attack (CNA) and Computer Network Exploitation (CNE). Additionally, US Joint Chiefs of Staff [78] acknowledge the concept of Information Operations as the integrated information-related capability employment within a military operation in concert with other performed operations to influence, disrupt, corrupt, or usurp the decision-making of existing and potential adversaries while protecting our own. Such operations integrate the employment of the core capabilities of electronic warfare, computer network operations, psychological operations, military deception and operations security. Furthermore, US Military Joint Publication [80] is specifically addressed to explore the cyberspace operations from the perspective of their planning, coordination, responsibility division, execution, and assessment. This publication defines cyberspace operations as "[...] the employment of cyberspace capabilities where the primary purpose is to achieve objectives in or through cyberspace." This Joint Publication also redefines the terms used by the US military, such as, replacing CNA with “offensive cyberspace operations” and CNE with “cyberspace exploitation”. Schmitt et.al. [71] define the cyber attack as "[...]

[16] Mazerunner. `https://cymmetria.com/`. Accessed: 15/09/2018

a cyber operation, whether offensive or defensive, that is reasonably expected to cause injury or death to persons or damage or destruction to objects." NATO Joint Publication [63] promotes defensive cyberspace operations as one of the key protection challenges as the NATO's dependency on such systems is increasing.

Applegate [3] attempts to define the principle of manoeuvre, both in its offensive and defensive forms, within cyberspace as it relates to the traditional concept of manoeuvre in warfare. Applegate asserts, that the points of attack are moved in cyberspace instead of forces and defines the cyber manoeuvre as "[...] the application of force to capture, disrupt, deny, degrade, destroy or manipulate computing and information resources in order to achieve a position of advantage in respect to competitors". The main characteristics of a cyber manoeuvre are: speed, operational reach, access and control, dynamic evolution, stealth and limited attribution, rapid concentration, parallel and distributed nature. Research introduces the following offensive cyber manoeuvre forms – exploitive manoeuvre (capturing information resources to gain a strategic, operational or tactical competitive advantage), positional manoeuvre (capturing or compromising key physical or logical nodes in the information environment which can then be leveraged during follow-on operations), influencing manoeuvre (using cyber operations to get inside an adversary's decision cycle or even to force that decision cycle though direct or indirect actions). As well as presenting the defensive cyber manoeuvre forms – perimeter defence & defence in depth, moving target defence, deceptive defence, and counter attack.

From a political and strategic stance, Mulvenon [61] explores the Chinese People's Liberation Army (PLA) conducted computer network operations from the perspective of the scenarios, doctrine, organization, and capabilities. Mulvenon states that defence and preemption are the core concepts of PLA and utilize offensive CNO as an attractive asymmetric weapon against the high-tech adversaries.

With a theoretical interest, Boukerche et.al. [10] explore an agent-based and biological inspired real-time intrusion detection and security model for computer network operations. Research presents a novel intrusion detection model based on artificial immune and mobile agent paradigms for network intrusion detection and response.

## 2.3 Cyber Red Teaming

From the publicly available authoritative sources, UK Ministry of Defence [75] recognizes increased use of red teaming as an alternative thinking and approach beneficial for defence. From this perspective, an independent Red Team (RT) is formed to subject plans, programs, ideas and assumptions to rigorous analysis and challenge by applying a range of structured, creative and critical thinking techniques. RT activities vary on the purpose and can include war-gaming, technical examination of vulnerabilities, system testing, or providing perspective through the eyes of adversary. Despite not focused on Cyber Red Teaming (CRT) directly, this doctrine grants valuable insight into RT experience, assembly, leadership, tool-set, tasking, and phases of red teaming. Additionally, presents the golden rules of red teaming: timeliness (delivered in right time), quality (usefulness of the deliverables), and access (tailored for the end user comprehensiveness level). It is suggested, that optimum team size is generally considered between five and nine experts. US Joint Force Development [79] looks at command red teaming with a purpose of decision support by providing an independent and skilled capability with critical and creative thought to fully explore alternatives in plans, operations, and intelligence analysis. RT activities include decision support, critical review, adversary emulation, vulnerability testing, operation planning and operational design. US acknowledges, that cyberspace aggressors can be considered as a specialized and highly-focused red teams. Similar to UK perspective,

this doctrine does not implicitly address CRT, but provides applicable aspects to adversary emulation and red cell operations. Additionally, Longbine [39] looks at the red teaming and acknowledges the tendency towards asymmetric warfare, where commanders require independent and skilled RT to understand the battlefield and ultimately achieving success. Longbine explores various RT definitions from the US perspective and highlights the core concepts – decision-making, challenging the thinking, providing alternative analysis and perspective. Despite, not focused on cyber component, the monograph presents key ideas for RT engagements, such as, threat emulation and conducting vulnerability assessment. Furthermore, European central bank's developed unified framework for threat intelligence-based ethical red teaming (TIBER-EU) [21] is focused at describing the controlled conduct of bespoke intelligence-led red teaming to mimic the Techniques, Tools and Procedures (TTPs) of the existing advanced threat actors against the European Union's core financial infrastructure. The process, consisting of four main phases – generic threat landscape, preparation, testing, and closure, attempts to cover a broad attack surface including elements, such as, people, processes, and technologies.

With a practical approach, Brangetto, Çalışkan and Rõigas [12] look at military CRT, which mimics the mind-set and actions of the attacker to improve the security of one's own organization. CRT is defined as "[...] an element that conducts vulnerability assessments in a realistic threat environment and with an adversarial point of view, on specified information systems, in order to enhance an organisation's level of security" [12]. The study explores the CRT purpose, tasking, formation, structure, and legal implications. Additionally, technical exercises and cyber ranges are proposed for testing, evaluating and mimicking the adversarial actions for cyberspace concepts, policies and technologies. Adkins [1] explores the option of cyber espionage and CRT to combat terrorism and its propaganda, recruitment, training, fundraising, communication and targeting. The CRT general principles are proposed for team assembly (penetration testers, social engineers, reverse engineers, Intrusion Detection System (IDS) specialists, and language specialists), equipment (computers, servers, cellphones, tablet computers, penetration testing lab, and IDS solutions), and attack philosophy (avoidance of readily-available and downloadable solutions by the terrorists, limiting the attack spreading just to intended target, connection bouncing through proxy servers).

To expand these concepts, Yuen [91] explores automated CRT with automated planning by the use of Artificial Intelligence (AI), such as, machine learning, hierarchical task networks, planning graphs, and state-space graphs to identify and plan the engagement as fast as possible in the applicable scope. Yuen acknowledges that CRT has a wider scope than penetration testing or vulnerability assessment, and is performed to appraise the infrastructure, processes, and personnel vulnerabilities to cyber attacks. Yuen, Randhawa, Turnbull, Hernandez and Dean [92] [67] propose a *Trogdor* system prototype for automated CRT based on state-of-the-art automated planners and AI techniques to generate attack plans and model the organization's network at multiple levels of abstraction. Proposed system puts emphasis on visual analytics to achieve situational awareness over organization, intrusion paths, attack steps, attack patterns, attack quality and attack impact. It includes reconnaissance, network scanning, vulnerability analysis, penetration testing, attack graph generation, and risk/impact assessment.

## 2.4 Cyber Attack Kill Chain

Hutchins, Cloppert and Amin [42] explores the intelligence driven Computer Network Defence (CND) analysis of adversary, used Techniques, Tools, Tactics and Procedures (TTTPs), operation patterns and intention characteristics. Researchers present an intrusion kill-

chain model to counter the evolving sophistication of cyber attacks. The kill-chain, in a systematic way, targets and engages the adversary at the phases of reconnaissance, weaponization, delivery, exploitation, installation, command and control, and execution of objectives. This approach attempts to detect, deny, disrupt, degrade, deceive or destroy the adversary and its attack at every stage of the kill-chain. To further support this, Lockheed Martin [37] acknowledges the significance of Critical Information Infrastructure (CII) for the national security and prosperity, and explores cyber kill-chain from the adversary and defender perspectives. To protect these infrastructures against an adversary, which is capable of defeating most common computer network defence mechanisms, every intrusion needs to be analysed to understand the adversarial motivations and strategies, allowing to break the attack chain with just one mitigation. Malone proposes to expand the Hutchins et.al. developed cyber kill-chain [44] by focusing on expanding the execution of objectives stage within the target information system. The proposed expansion adds two sub chains – internal kill-chain (reconnaissance, exploitation, privilege escalation, lateral movement, target manipulation), and target manipulation kill-chain (reconnaissance, exploitation, weaponization, installation, execution). Furthermore, Additional attack disruption methods have been proposed and adapted besides the Lockheed Martin's "Cyber Kill Chain" [38], such as, Mandiant/FireEye "Attack Lifecycle Model" [45], Microsoft "Attack Kill Chain" [51], SANS "The Industrial Control System Cyber Kill Chain" [4], and MITRE "ATT&CK" [57].

Kim, Kwon and Kyu [31] propose the modified cyber kill-chain model for multimedia service environments, such as, IoT. Kim et.al. study the limitation of the existing cyber kill-chain models (e.g. Lockheed Martin, FireEye, Command Five) and endorse the information security focus on detecting the actions happening within the network. Research proposes to expand the cyber kill-chain model with the internal network kill-chain stages – internal reconnaissance, weaponization, delivery, exploitation, and installation. Yadav and Rao [89] review the cyber kill-chain and present the popular attack tools and defensive options at every stage of a cyber attack. Al-Mohannadi et.al. [2] look at modelling the cyber attacks, which have not yet happened, by the use of diamond model, kill-chain and attack graphs.

Wen et.al. [85] [86] look at Advanced Persistent Threat (APT) detection as a measurable mathematical problem through Bayesian classification with correction factor. By studying the cyber kill-chain, a solution is proposed for APT detection based on cyber security monitoring and intelligence gathering. The introduced approach utilizes information acquired from the online sources or acquired from real-time detection, and is divided into three categories – attack intelligence (e.g., sensor logs, firewall/anti-virus alerts), attack behaviour (e.g., *phishing*, Denial of Service (DoS)), attack events (e.g., attacker profiling, TTTPs). Furthermore, tool prototype implementation is described based on *JESS* – the rule language engine for the *Java*, but with no real testing and application presented in the paper. Moskal, Yang and Kuhl [59] propose a cyber-based attacker behaviour modelling in conjunction with the "Cyber Attack Scenario and Network Defense Simulator" (CASCADES) to model the interaction between the network and the attackers. The approach simulates and measures the interaction between various generated types of attackers and network configurations under different scenarios. Attacker behaviour and decision-making process modelling is based on a reduced single attack action, which is executed against an integrated cyber kill-chain with fuzzy logic rules for each attack stage.

## 2.5 Cyber Red Team Technical Exercises

In a comprehensive manner, Leblanc et.al. [34] explore and analyse multiple war-gaming exercises and implemented exercise support tool-sets. ARENA provides cyber attack modelling, execution and simulated network construction, primary used to analyse Intrusion Detection System (IDS). Solution is capable to simulate simple attacks ranging from Denial of Service (DoS) to back-door installation. RINSE (Real-time Immersive Network Simulation Environment) is a live simulation of large scale complex Wide Area Network (WAN), where various Local Area Network (LAN) operators have to defend against attacks. It is capable to simulate simple attacks, such as, DoS, Distributed Denial of Service (DDoS), and computer worms. SECUSIM is an application with a purpose to specify attack mechanisms, verify defence mechanisms and evaluate their consequences. OPNET is a network simulator solution, which can be used also for cyber attack simulation aiding the analysis and design of communication networks, devices, protocols and applications. SUNY provides modelling and examining network performances under DoS attacks. NetENGINE is a cyber attack simulation tool allowing to model very large and complex computer networks to train Information Technology (IT) staff in combating cyber attacks. Various generic cyber attacks are executed against simulated networks, as well as against the in game communication channels used by the players. SIMTEX is a simulation infrastructure with various computer network attacks used for training. CAAJED focuses on kinetic effects of cyber attacks in a war situation, where attacks are manually implemented and their effects controlled by the operators. IWAR is a network attacks and defence simulator implementing common trojan horses, vulnerability scanners, malware, DoS and brute-force attacks. RMC is an isolated physical network for Computer Network Operation (CNO) education and training with an attacking team and training supervisors. DARPA National Cyber Range is a project with an aim to simulate cyber attacks on computer networks and help develop strategies to defend against them. CyberStorm is the US Department of Homeland Security developed exercise with the aim of examining readiness and response mechanisms to a simulated cyber event. Participation is strictly limited to Five Eyes alliance members. Piranet is the French developed response plan and a simulation exercise of a major cyber attack against France's Critical Information Infrastructure (CII). Divine Matrix is the India's war-gaming exercise to simulate a nuclear attack accompanied by a massive cyber attack with kinetic effects against India.

Mauer, Stackpole and Johnson [48] look at developing small team-based cyber security exercises for use at the university as a practical hands-on part within the courses. The research explains the management and roles of the engaged parties in the exercise creation and execution. The developed game network is comprised of small set of virtualized machines used by a group of participants to attack, defend and monitor the event. DeLooze, McKeen, Mostow and Graig [15] examine the US Strategic Command developed simulation environment to train and exercise CNO and determine if these complex concepts can be more effectively taught in the classroom. The simulation environment consists of "Virtual Network Simulator", comprised of two or more networked computers designed to represent attack effects in an interactive graphical environment, and the "Internet Attack Simulator", presenting a set of simple attacks, ranging from reconnaissance to DoS, available for launching against the network simulator's virtual network. Researchers confirm the benefit of CNO simulation exercises by measuring the increase of knowledge of the participants.

## 2.6 Identified Gaps

Related work analysis shows that no substantial work has been done in the area of exploring the aspects the cyber red team technical capability development, execution of responsive cyber defence operations, and conducting technical exercises aimed at cyber red team training and skill development. The main drawbacks are the following:

1. red teaming is acknowledged by nations primary as a supportive element to decision making process, providing an alternative analysis on various topics, but not anticipated as an operational capability to actively pursue the commander assigned external targets. The red team, consisting of highly skilled experts, might be engaged in cyber espionage, or targeting organization's own defences to provide adversarial perspective. However, it is not yet accepted to be used actively against any other target or threat actor. A broader perspective and the scope of cyber red team capability applicability has to be expanded;

2. cyber kill-chain has been thoroughly reviewed from the defence perspective, as its primary area of application, however, the view from the adversarial aspect on countering the cyber kill-chain should be explored. By countering the cyber kill-chain, in this work, is understood the red team Techniques, Tools, Tactics and Procedures (TTTPs) allowing to bypass the detection and prevention mechanisms at every phase of the kill-chain. This is especially important when planning the cyber operations against an advanced adversary to ensure the maximum possible success rate;

3. cyber red team technical capabilities and techniques for tool development to be used against a real adversary are not explored fully. The proper techniques and tools are even more important when a small cyber red team (up to 9 members) is considered to be engaged in the computer network operations;

4. the aspects of active cyber defence are explored, but responsive activities are not considered. Despite the responsive cyber defence being a subset for active cyber defence, it possesses distinctive differences, especially that of responsive defence being directed primarily at external networks and systems instead of mainly focusing on activities within own systems;

5. various sources interpret computer network operations differently and present various definitions, ranging from any attacks executed over computer networks up to being only applicable to the cyber warfare. A coherent definition is required to have a solid understanding of such operation applicability and implied characteristics;

6. no special guide, manuals or proposals exist on computer network operation execution by the cyber red team. It might be expected, that such guidelines would be developed for nation's internal use only and not shared publicly, however, there is a lack of publicly available information on this topic; and

7. the tackled issues for the cyber defence exercises are either narrow in scope, specific to a particular nation or small set of nations, restricted only to exercising just the decision making process or a small subset of a full-scale cyber operation, or limited to just simulation of common cyber attacks. Additionally, majority of the exercises are delivered for the *Blue Team* defensive capability building, and the *Red Team* is either simulated or role-playing the adversary. However, a dedicated technical exercise for training cyber red team capabilities is required.

This thesis addresses the identified gaps to introduce a unified understanding and definitions for the core principles, such as, Responsive Cyber Defence (RCD), Computer Network Operation (CNO), Cyber Red Teaming (CRT) and "cyber attack kill-chain", explores the cyber red team offensive and defensive capabilities from the perspective of the cyber red team responsive computer network operations, presents practical and immediately applicable cyber red team TTTPs to complement the conducted responsive operations, proposes automated anomaly detection and cyber deception usage within the cyber red team's operational network for its protection, situational awareness, threat assessment and adversary tracking, and defines the requirements for the technical exercises aimed directly at cyber red team capability development and preparation for real-life full-spectrum cyber operation execution.

# 3 DEFINING SPECIALIZED CYBER RED TEAM RESPONSIVE OPERATIONS

Nations are not well known to announce publicly their offensive cyber capacities and more focused on advertising and promoting cyber defence activities and initiatives. Despite this, the identified major cyber intelligence programmes attributed to nation states, such as, US NSA PRISM[17] and UK GCHQ TEMPORA[18], prove to represent the actual cyber capabilities of some NATO alliance members. Also, collective defence alliances and treaties as of now do not acknowledge cyber offence as a collective approach and leave such initiatives up to every individual nation to assess and develop on its own. However, this does not exclude any other allies either requesting such assistance or joining forces to engage in offensive cyber operations. Furthermore, responsive cyber defence balances between the defensive and offensive posture and can enable defenders to engage the adversary both in their own cyber infrastructure, as well as globally. However, there is no clear understanding on how such activities can be executed and what their operational and technical requirements are.

This chapter addresses responsive activities taken in the cyberspace by introducing and defining the concept of *specialized cyber red team responsive computer network operations*. This concept is divided into three main components: 1) responsive cyber defence – the overarching concept defining the defender's posture to pursue the adversary to ensure the maximum possible protection of the defended assets; 2) computer network operations – are the means of the defender to exercise this posture against the adversary and systems engaged in the attack; and 3) cyber red team – the actual team of experts possessing the required skills and capabilities to engage in such responsive operations.

The following subsections present the individual concepts of responsive cyber defence, computer network operations, cyber red teaming, and assesses their minimal operational requirements. These principles lay the required foundation to define, once combined, the specialized cyber red team responsive operations and specify their requirements.

## 3.1 Responsive Cyber Defence Requirements

Responsive Cyber Defence (RCD) is a controversial, vaguely understood and seldom used term. More commonly, a broader concept of Active Cyber Defence (ACD) is utilized as presented in chapter 2.1. One definition states that ACD is accomplished through denial to information and deception via misleading information [29] [28]. Another definition suggests actively engaging the incoming cyber attack to destroy or reduce its effectiveness [16], or conduct the hack-back [46]. Contrary to this, US DARPA acknowledges ACD (see footnote 10 on page 22) without the component of actively pursuing and interacting with the adversary. Yet another ACD concept is directed to threat assessment and mitigation [17]. Some view this approach purely limited to one set of approaches (e.g. *"white worms"*, address hopping) [41] [68]. One proposed RCD definition is aimed at gaining access to the third-party systems for modification or deletion of data held therein [49]. Another RCD definition is quite broad and permits any applicable defensive actions to be taken against any communication and information system engaged or used in the attack [13]. These various definitions of ACD are mainly focused towards the integrated and synchronized real-time capability for detection, analysis, identification, and mitigation aimed

[17] The Verge. "Everything you need to know about PRISM." `https://www.theverge.com/2013/7/17/4517480/nsa-spying-prism-surveillance-cheat-sheet`. Accessed: 28/09/2018

[18] Wired. "A simple guide to GCHQ's internet surveillance programme Tempora." `https://www.wired.co.uk/article/gchq-tempora-101`. Accessed: 28/09/2018

at degrading or eliminating the adversary offensive capacity. Tackled ACD activities do not necessarily limit the response to be expanded beyond the defender's own systems, however, they are still primarily focused towards combating the adversary and its activities within the defended networks and information systems.

The aforementioned definitions for the RCD already address a more targeted response, going beyond one's own infrastructure, and by actively pursuing the perpetrators through other information systems and networks. The concepts addressed in these definitions are limited in scope and do not include the principle of system monitoring, information gathering or alteration of system behaviour to allow further intelligence gathering and attribution. Such responsive actions should at all times grant the situational awareness and permit collection of information for the responding Cyber Red Teaming (CRT). Such properties are crucial to allow defenders to trace the cyber attack back to its origin. This thesis addresses the various interpretations and concepts of active and responsive cyber defence and defines the responsive cyber defence.

**Definition 1.** Responsive cyber defence is a responsive activity (1) exercised by conducting cyber operations (2) against engaged external information systems (3) to defend own information systems (4).

With the essential elements of the definition being:

1. response to an incoming threat, which can be executed in real-time as the attack commences or asynchronously once enough attribution information is gathered. The incoming attack against the defended information systems is the main precondition for engaging in RCD;
2. cyber operations, such as, computer network operations as discussed in chapter 3.2;
3. the extraterritorial nature of RCD means that the exercised activities are directed either against adversary's or third-party information systems, which are being used by the attacker. Within this thesis, information system[19] is understood as a set of information technologies, software, physical infrastructure, human resources, procedures and regulations, used for creating, storing and processing information belonging to and governed by a certain entity; and
4. the measures required for ensuring the protected system defence, which include the activities, such as, attack detection, identification, analysis, threat assessment, and attribution, aimed at destroying or reducing the effectiveness of the attack.

It has to be noted, that RCD, in contrast to ACD, can only be executed in case of a verified attack or directed aggression. The RCD per se is not pre-emptive but can be exercised as a subset within a broader responsive encounter, such as, political or kinetic, which in turn can be pre-emptive. As such, the RCD capability does not have to be constantly active or exposed as it would be in case of a more permanent ACD techniques, for example, deceptions and honeypots. Such responsive capabilities would be maintained, kept at high readiness level and alert, but activated only once the circumstances necessitate either within the ACD scope or independently. The Fig.1 illustrates the relation between

[19]The Information system definition derived from author's personal experience merged with Encyclopaedia Britannica definition of Information System. `https://www.britannica.com/topic/information-system`. Accessed: 11/11/2018

the ACD and RCD to represent the scope of engagement, dependencies and applicability. As shown, the RCD is oriented at engagements beyond own infrastructure, however, still heavily depending on the initial detection, analysis and identification capability provided either by the deployed threat detection solutions or ACD techniques. To reach the defined goals of the RCD, the CRT would have to trace back the intrusion through the attack paths, which would include the global communication infrastructure owned by the Internet Service Provider (ISP) or unwitting third-party networks. Additionally, taking control or delivering the desired impact to the perpetrator's deployed attack infrastructure, which could consist of digital and physical assets set up or compromised by the attacker. Extraterritorial nature of RCD means that other engaged or involved parties, such as, proxies and breached public services to host the malware or Command and Control (CnC), can be targeted by the responders to trace back or affect the capabilities or paths taken by the attackers. This property of RCD, as well as ACD, is heavily discussed and tackled by the legal experts [13]. Despite implied legal ramifications, this thesis focuses purely on the technical approaches and methods, by providing only a brief insight into the associated legal concerns (see section 6.4 on page 73). The CRT would operate and conduct the Computer Network Operation (CNO) from its own created and built infrastructure separated from the defended network. Due to this, the CRT has to maintain all the capabilities to continuously detect, identify, analyse and assess the threats while executing the CNO within the RCD. This means, the CRT has to persistently follow the principle implied by the Observe, Orient, Decide, Act (OODA) loop[20] to maintain the visibility of the ongoing activities and keep track on the perpetrator movement, actions, origin and presence. However, in this thesis, a fully functional capability and developed operational network is assumed and is not tackled in a greater detail regarding such operational network design, asset procurement, deployment, modification, and destruction, as it is out of the scope of this work. However, some insight in operational network structure prototyping and defence is given in chapter 5.

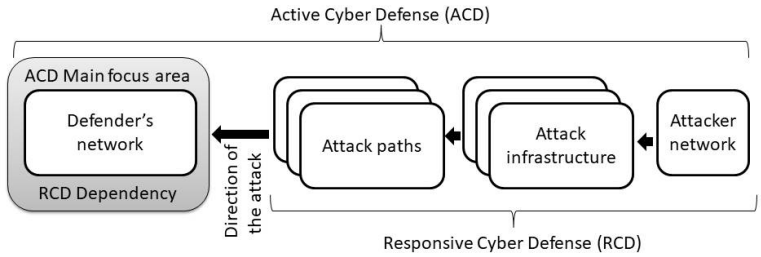



*Figure 1: Active and responsive cyber defence areas of engagement*

Based on the related work, as presented in the chapter 2.1 and discussed here, the RCD has at least the following requirements and characteristics:

1. **Available.** The capability has to be permanently maintained and ready for deployment at the shortest notice possible.

2. **Coordinated.** Has to be in response to an verified aggression and coordinated between the responders at various directions, such as, political, kinetic, or Computer Network Defence (CND).

[20] US Defense Technical Information Center. `http://www.dtic.mil/docs/citations/ADA566700` Accessed: 11/11/2018

3. **Integrated.** Compliant with the initial threat detection, assessment, and related ACD activities.

4. **Synchronized.** Real-time as the attack has been detected and is still ongoing. Additionally, it can be asynchronous in cases when the attack has seized, but the artefacts have been identified through activities, such as, vulnerability assessment, threat hunting, digital forensics, or signature update on the threat detection and analysis systems.

5. **Automated.** Maximum automation to allow the highest level of synchronization and minimize the time required for the early stages of response.

6. **Asymmetric.** Every CRT action has to deliver the maximum possible effect. Which correlates with the high readiness and preparedness level of the CRT.

7. **Omnipresent.** Ability to track and pursue adversary throughout the cyberspace with the highest flexibility possible.

8. **Effective.** Capable of delivering the required impact, such as, destroy or reduce effectiveness of the attack, engage adversary by denial & deception or a hack-back.

9. **Independent.** CRT has to contain all the necessary capabilities to conduct the RCD.

This is not an exhaustive list and more characteristics could be applicable depending on the particular RCD specifics and delivered effect requirements.

## 3.2 Computer Network Operation Requirements

The Computer Network Operation (CNO) definition varies from the perspective and the tier of applicability, such as, political, strategic or tactical, as presented in chapter 2.2. The US doctrine encompasses the computer network operations as a set of various activities within the computer networks [34] or employment of cyberspace capabilities to achieve objectives in or through the cyberspace [80]. One definition offers to treat the cyber operations as a manoeuvre in cyberspace to apply force to achieve a position or advantage in the cyber domain [3]. Another view point states, that CNO is always offensive in nature and is of a dual use [70]. Furthermore, it is inclined, that such cyber operations can be exclusively exercised only within cyber warfare and military engagements [17]. These definitions are mainly focused towards various activities, either defensive or offensive, performed by the use of computer networks, and define the CNO appropriately.

The inconclusiveness brought by few of the definitions regarding CNO being limited to cyber warfare is unclear and restrictive on the applicability of this concept. This thesis addresses the various CNO interpretations and defines computer network operations.

**Definition 2.** Computer network operations are any set (1) of effect-oriented activities (2) performed by the application of force (3) in the cyberspace by the use of computer networks (4).

With the essential elements of the definition being:

1. single or a group of activities supporting either each other, having a common direction or aimed at reaching same effect by various means in cyberspace;
2. instead of being limited to an explicit set of predefined activities, an effect-based approach is considered. The desired effect is specified by the operational requirements or aligned with the supported operation demands. Such

activities can include, for example, defensive, responsive, offensive, intelligence, or information operations;

3. application of force constitutes to specifically directed and focused activities against the elements of the target information system; and
4. the use of computer networks, to achieve the desired operational effect in cyberspace, is the main feature and pre-condition for such operations. It can be a hybrid approach, where the computer network operation delivers one effect in cyberspace with an intention to trigger the desired effects, such as, kinetic or impacting the adversary decision-making process.

The CNO concept can include any subset of activities utilizing computer networks, such as, Computer Network Defence (CND), Computer Network Exploitation (CNE), or Computer Network Attack (CNA). Within this scope, by CNE is understood the offensive use of computer networks to conduct target information system infiltration activities, for example, executing cyber espionage or cyber reconnaissance operations. The CND and CNA are more specific, and each encapsulates purely either the defensive or offensive nature of the computer network use. A responsive CNO is an operation executed within the Responsive Cyber Defence (RCD). These activities do not have to be limited just to these particular tasks and any type of operation, as long as it is dependent on computer networks, can be applicable to CNO. For example, cyber espionage operation, relying on CNE to access and apply CNA to gain position in the target network, would also be considered as a CNO. The operation, as the term implies, is oriented towards delivering effects by application of force in the cyberspace and is not to be confused with a regular use computer networks and their services, such as, browsing the Internet or accessing e-mail. From this perspective, the CNO can carry any desired effect, either again cyber or physical, to be delivered by the means of computer networks, such as, offensive, disruptive, denial, or deceptive. These concepts are represented in Fig.2 from the perspective of peace- and war-time, Active Cyber Defence (ACD), RCD and cyber warfare paradigms. The RCD is an overarching measure, consolidating all of the operation types applicable to both war and peace. The author argues, that in the current political era, it is becoming harder to distinguish the peace- and war-time operations since the thresholds in cyberspace are becoming more obscure as the nation-states practice more unconventional and hybrid warfare.

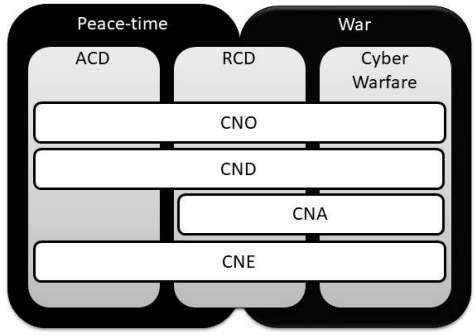



*Figure 2: Computer network operation applicability to operational paradigms*

Based on the related work, as presented in the chapter 2.2 and discussed here, the CNO has at least the following requirements and characteristics:

1. **Hybrid.** CNO can be used to aid both defensive and offensive effects as well as any other activities in support for achieving these effects.

2. **Rapid.** Ability to deliver fast paced operation execution if such capability is prepared and maintained.
3. **Focused.** Possibility to achieve rapid concentration of force in cyberspace on single or a small set of targets.
4. **Dynamic.** Capability to adjust, evolve, and mutate according to the changing environment and operational needs to provide inconsistency.
5. **Agile.** Possesses high manoeuvrability supported by the dynamic nature.
6. **Stealthy.** Granting limited attribution, protection of identity and privileges.
7. **Pervasive.** Has long operational reach and can target any interconnected node remotely in the cyberspace to provide access and control.
8. **Parallel and distributed.** Can be launched from multiple sources in the cyberspace simultaneously against a single or a set of targets.
9. **Effective.** Has the potential capability of delivering cyber and, if designated – kinetic effects.
10. **Asymmetric.** Can deliver higher operational effect in contrast to other operations, if prepared and used accordingly.

This is not an exhaustive list and more characteristics could be applicable depending on the particular CNO specifics.

## 3.3 Cyber Red Teaming Requirements

The Red Team (RT) definition varies on the domain to which this capability is applied, such as – command, military, or cyber, as it is discussed in the chapter 2.3. Most common RT definition revolves around a small, independent team of highly skilled experts, to provide decision support, alternative analysis and approaches to an existing problem [75] [79]. One definition attempts to present military Cyber Red Teaming (CRT) as an adversarial-based assessment of own information system vulnerabilities [12]. Additionally, RT is seen as a decision-making supportive element for conducting asymmetric warfare [39]. These definitions are more focused towards supporting decision-making process or providing a specialized assessment of own assets in cyberspace or physically. It has to be noted, that the CRT is the RT utilizing and performing their assigned tasks mainly within cyberspace and by the use of technology.

The aforementioned definitions are directed towards securing and evaluating defender's systems or course of action, but not at affecting the adversary's information systems or decision-making process. To engage a CRT in conducting a Computer Network Operation (CNO) within the Responsive Cyber Defence (RCD) scope, which has the extraterritorial nature, the CRT has to be structured and possess all the defined qualities and high standards of the RT. However, to separate the RT activities between internal and external, a concept of *"specialized cyber red teaming"*, directed at the adversary, is proposed. This thesis addresses the various red team interpretations, use cases, and their applicability, and defines the specialized cyber red team.

**Definition 3.** specialized cyber red team is an independent team of highly skilled experts (1) tasked to perform activities (2) against the adversary's cyber capabilities (3).

With the essential elements of the definition being:

1. a fully functional and self-sustainable cyber red team operating within the cyberspace;
2. timely and focused activities against the adversary, for example, conducting the adversarial vulnerability assessment, capability and Techniques, Tools, Tactics and Procedures (TTTPs) analysis, or execution of computer network operations; and
3. the cyber red team performed activities being in support of decision-making process, ongoing related operations, or having its own impact delivery.

Such CRT would be tasked to operate on its own or deliver an effect in support of other activities, such as, cyber-kinetic engagements. CRT as such retains the capabilities to operate at any level applicable to deliver the required effect. This could mean operating through kinetic and cyber means to deliver the impact, either directly or indirectly, on the target's cyber or kinetic assets, as long as the execution of operation has the cyber component (Fig. 3). As an example, a CRT might engage in social engineering campaign to gain access to a target server room either physically (e.g., acquiring or copying access cards) or via cyber means (e.g., obtaining remote access password). As well as, for example, gaining remote access to a adversary server room Heating, Ventilation, and Air-Conditioning (HVAC) system and shutting it down to cause physical effects or damage. As for the RT adversary and threat emulation principle, the CRT may exercise these approaches to conduct a *false-flag* or *no-flag* operations. Within such operations, the CRT would pretend and assume the role of another threat actor and adopt its TTTPs to confuse the enemy and make attribution harder or force it to be incorrect. It has to be pointed out, that the CRT is a team with suggested size of no more than ten highly skilled experts to keep it balanced and agile, as it is proposed by the UK doctrine [75]. Such small team requires every participant to assume multiple roles to support all of the CRT operational needs, such as, operational infrastructure planning and deployment, asset acquire, intelligence gathering, Operational Security (OPSEC) assurance, targeting, monitoring, and own asset protection. Depending on the available resources and goals, a separate team can be established to support all of the CRT operational requirements. Most importantly, a specialized CRT is the perfect candidate for executing a CNO within the scope of RCD, hence, combining all of the requirements, benefits and restrictions. To create, develop, maintain and evolve such a complete capability the proper resources have to be allocated and invested as early as possible. However, in this thesis, a fully functional specialized CRT is assumed, and no deeper insights are provided on the assembly, management and its operational requirements, as it is out of the scope of this work.

Based on the related work, as presented in the chapter 2.3 and discussed here, the CRT has at least the following requirements and characteristics:

1. **Small.** A small team of experts for increased ease of management and agility.
2. **Skilled.** Every expert has to be skilled to contribute to the CRT capability.
3. **Specialized.** The team has to self-sustain its independence and operational requirements by providing all the necessary specialized skills.
4. **Innovative.** Finding alternative approaches and identifying vulnerabilities in the designated target systems or processes, by applying creativity, critical review and thinking techniques.
5. **Highly focused.** Aimed at conducting a specific operation to deliver the designated effect.

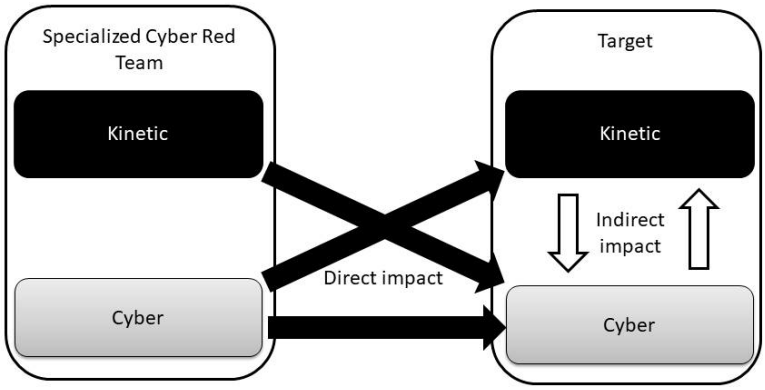



*Figure 3: Special cyber red team operation effects*

6. **Targeted.** Persistent in performing precision strikes to complete the assigned task.
7. **Asymmetric.** A small team with highly skilled experts having the capability on delivering the maximum possible effect.
8. **Adaptive.** Adapting and employing adversarial TTTPs against the target system.
9. **Supportive.** Providing support to the ongoing operations, decision-making processes, and operation planning.
10. **Timely.** Has to deliver the required effects in the defined time frame.

This is not an exhaustive list and more characteristics could be applicable depending on the particular CRT organization and tasking specifics.

# 4 SPECIALIZED CYBER RED TEAM RESPONSIVE OPERATIONS

Concepts, methods and approaches presented in the listed publications (Publication I, Publication II, and Publication III) are applied to fulfil the requirements of specialized cyber red team responsive cyber operations, allowing the creation of custom cyber red team tools and techniques for the responsive operation execution. Furthermore, such technique applicability is evaluated and proposed to counter the cyber kill-chain. Additionally, an insight into legal ramifications of specialized cyber red team responsive operations is presented.

## 4.1 Operational Requirements

The requirements listed separately for Responsive Cyber Defence (RCD), Computer Network Operation (CNO), and Cyber Red Teaming (CRT) have to be combined to define the demands of the specialized cyber red team responsive computer network operations, as it consolidates all of the individual concepts. The techniques and tools, presented in the listed publications, are described and mapped against these criteria to identify how they can benefit, enhance and support the Techniques, Tools, Tactics and Procedures (TTTPs) creation for specialized CRT responsive CNO fulfilment.

**Definition 4.** Specialized cyber red team responsive computer network operations are such computer network operations (1), which are executed by a specialized cyber red team (2) within the responsive cyber defence (3).

With the essential elements of the definition being:

1. computer network operations according to Definition 2 on page 35;
2. specialized cyber red team according to Definition 3 on page 37; and
3. responsive cyber defence according to Definition 1 on page 33.

The table 3 represents the features of every individual RCD, CNO and CRT concept grouped to form the specialized cyber red team responsive computer network operation requirements. These requirements are then mapped against the methods and approaches presented in the publications to introduce TTTPs supporting this objective. In the table, the 'X' denotes that a particular technique, presented in that publication, is applicable to the specialized cyber red team responsive operation specific technique, tool and procedure development. The listed techniques are not exhaustive and do not form a complete list, but represent the ones as introduced in the listed publications. At least the following characteristics are applicable to the proposed new concept:

1. **Stealthy and innovative.** Grasps the core requirements for self-sustainability, evolution and adjustment to the changing environment, adaptation of adversarial TTTPs, independence and innovative skills, to provide the maximum possible level of stealth and make attribution harder;

2. **Agile and available.** A prepared and on demand available, small team with high manoeuvrability;

3. **Focus of force.** Allows achieving rapid concentration of force focused on performing a specific operation against a single or a small set of targets in the cyberspace;

4. **Targeted and pervasive.** Is specifically crafted and designed to deliver the required single impact or a set of parallel activities remotely from any position in the cyberspace;

| RCD | CNO | CRT | Specialized CRT Responsive CNO | Pub.I | Pub.II | Pub.III |
|---|---|---|---|---|---|---|
| Independent | | Skilled | Stealthy and innovative | X | X | X |
| | | Specialized | | | | |
| | Dynamic | Adaptive | | | | |
| | Stealthy | Innovative | | | | |
| Available | Agile | Small | Agile and available | X | X | |
| | Focused | Focused | Focus of force | | X | X |
| Omnipresent | Pervasive | Targeted | Targeted and pervasive | X | X | X |
| | Parallel and distributed | | | | | |
| Automated | | | Rapid and timely | | X | |
| Synchronized | Rapid | Timely | | | | |
| Integrated | | | Integrated and coordinated | | | X |
| Coordinated | | Supportive | | | | |
| | Hybrid | | Hybrid and effective | | X | X |
| Effective | Effective | | | | | |
| Asymmetric | Asymmetric | Asymmetric | Asymmetric | | X | X |

*Table 3: Specialized CRT responsive CNO grouping and TTTP mapping to publications*

5. **Rapid and timely.** Enabling the fast and automated execution of synchronized activities in the designated time or against the adversary as the ongoing attack progresses;

6. **Integrated and coordinated.** Support to other responsive operations and activities in a coordinated manner based on initial threat assessment;

7. **Hybrid and effective.** Can be applicable both to defensive or offensive activities, delivering required impact against kinetic or cyber assets; and

8. **Asymmetric.** Has to provide maximum possible effect with minimum available resources against a stronger adversary.

This is not an exhaustive list and it can be expanded or optimized with the characteristics, which could be applicable depending on specific operational goals, desired effects and available resources.

## 4.2 Techniques, Tools, Tactics and Procedures

The techniques and tools presented in listed publications are logically intertwined to provide the support for the cyber red team operation key stage execution. Those being – gaining initial access (Publication II), establishing a command and control channel (Publication I), and delivering the impact (Publication III). These critical stages are part of nearly any cyber operation (see table 6 on page 52), where the network perimeter needs to be breached to establish initial foothold (e.g., through cyber means or kinetic to cyber methods), a secure channel is required to provide path into the target system for lateral movement and asset location, and finally the designated impact delivery. Every computer

network operation will have its own requirements and effects to be delivered to the adversary information systems. In some cases, to deliver the desired effect, the cyber components are just means of triggering it, such as, disabling the power to force the use of backup generator, conduct a Denial of Service (DoS) to compel to switch to insecure communication lines, target the integrity of a military command and control system to degrade situational awareness and cause incorrect decision-making. The following sections, in a structured way, review the publications and present the fundamental techniques, prototyped tools, and procedures for technique and tool applicability to the specified cyber red team operation requirements (see table 3).

#### 4.2.1 Gaining Initial Access

**Techniques.** Publication II shows the novel and easy to use method for binary protocol bit-aware fuzzing and reverse-engineering [9]. The core concepts introduced by that work rely on creative way for atomic data manipulation exposing vulnerabilities and attack vectors. Assessed network protocol fuzzing frameworks are limited to fuzzing one byte as the smallest unit of data. Such limitation to one byte can be reasonable when assessing the protocols located at the upper layers of the OSI model, such as, FTP, HTTP, and DNS. However, this is a severely limiting factor restricting binary protocol analysis and fuzzing, where the data fields can be of a variable length and not always compatible with a length of one byte. For example, the IPv6 header Flow Label field is 20 bits long and cannot be byte-aligned to comply with the protocol specification. In essence, nearly any network communication protocol can be treated as "binary protocol" since they are composed of multiple variable length bit groups (i.e., bit-fields). The proposed solution accepts one bit as the smallest unit of data, thus allowing flexible fuzzing test-case creation. This permits practically any communication protocol analysis independent of its OSI layer, such as, performing Layer-2 Ethernet frame or Layer-3 IP packet fuzzing. The proposed technique is also applicable to other media, such as, serial line communications, as long as available hardware and software supports it. In the conducted experiments a proprietary serial line protocol ported to TCP/IP stack was successfully targeted.

In general, fuzzing does not immediately guarantee to find exploitable vulnerabilities, since this depends on multiple factors, such as, targeted software development quality, complexity, used programming language, code coverage, test-case correctness, and time. The assessed popular fuzzing frameworks have a complex targeted protocol description and test-case creation process. *Bbuzz* attempts to ease this process, speed-up the increased quality test-case creation, and minimize time investment for attack set-up through systematic and easy to use approach.

**Tools.** Based on these principles, the *Bbuzz* framework for systematic binary protocol fuzzing and reverse-engineering is prototyped, described and applied in practice by the author to confirm its applicability and value for cyber red team operations. This framework allows an automated test-case generation by analysing the available sample traffic and applying methods, such as, bit-group pattern matching, mutable and immutable field identification, and entropy measurements. Such approach, in case of available traffic capture, allows a rapid initial test-case generation and start of the fuzzing process. For effective test-case generation the framework uses various field mutation approaches and produces n-fold Cartesian product of all available payload field mutation sets. This ensures that all possible payload combinations from individual field mutation sets are generated to grant the most complete set of test-cases.

The table 4 briefly summarizes the findings from Publication II to compare the commonly used fuzzing frameworks and tools against the developed *Bbuzz* framework (tools

are listed in **bold** in the table). The comparison criteria is based on the following requirements (listed in *italics* in the table): open-source and available on demand to the CRT depending on the operational requirements; reasonably maintained by the developers to ensure it being up to date as much as possible; is designed or can be applied to support also network protocol fuzzing; can fuzz the protocol starting from OSI Layer-2; the fuzzing test-cases support variable length bit-fields, which can be fuzzed bit-wise with one bit being the smallest fuzzing test-case; can be used to perform network traffic sample analysis to identify features, such as, pattern mining, immutable and mutable bit-field identification, and field Shannon entropy calculations; based on the traffic analysis can automatically create the initial test case; and can monitor the target system under test as much as it is possible depending on the use case. In the table, the field marked with *'Yes'* denotes, that this feature is supported, and *'No'* presents the lack of support for the respective framework. Target monitoring for *Taof* has been labelled as *'Partial'* since the tool expects a reply from the target system and if it is not received, then an exception is assumed. Multiple features for the *Afl* have been marked as *'Partial'* since it is a file format based fuzzer, but can be used to generate test cases from a sample network packet, which have to be wrapped by other means to be sent over the network to the destination. *Bbuzz* is aimed at remote system testing over network and supports ICMP echo messages and port probing to detect a possible exception condition, therefore it has been labelled as *'Partial'*. From the presented results it can be observed, that the *Bbuzz* tool can deliver more flexibility and functional diversity for the CRT when compared to other common fuzzing frameworks.

| | **Spike** | **Sulley** | **Boofuzz** | **Peach CE** | **Taof** | **Zzuf** | **Afl** | **Bbuzz** |
|---|---|---|---|---|---|---|---|---|
| *Open-source* | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| *Maintained* | No | No | Yes | Yes | No | Yes | Yes | Yes |
| *Network-based* | Yes | Yes | Yes | Yes | Yes | Yes | No | Yes |
| *Layer-2* | No | No | No | No | No | No | Partial | Yes |
| *Bit-wise* | No | No | No | No | No | No | Partial | Yes |
| *Traffic sample analysis* | No | No | No | No | No | No | Partial | Yes |
| *Automatic test-cases* | No | No | No | No | No | No | Partial | Yes |
| *Target Monitoring* | No | Yes | Yes | Yes | Partial | No | No | Partial |

*Table 4: Common fuzzing framework comparison to Bbuzz*

As a use case, *Bbuzz* was applied to quickly reverse-engineer key features of the NATO proprietary Link-1 binary protocol, which is used for real-time air picture representation at the military operations centres and air traffic control stations. The protocol property reverse-engineering, such as, flight number, coordinates, altitude, bearing, velocity, and Identify Friend or Foe (IFF) code, allowed the injection of fake aeroplane tracks via the computer network. This attack vector was used to degrade the situational awareness and decision-making process of a NATO operation, performed within the NATO response force readiness exercise STEADFAST COBALT 2017[21]. The *Bbuzz*[22] framework, written in Python 3, has been released publicly on GitHub under the MIT license and is freely available to

[21]SHAPE. "Exercise Steadfast Cobalt set to get underway in Lithuania." `https://shape.nato.int/news-archive/2017/exercise-steadfast-cobalt-set-to-get-underway-in-lithuania`. Accessed 23/09/2018

[22]Bbuzz. `https://github.com/lockout/Bbuzz`. Accessed: 01/10/2018

everyone for usage and further customization.

Additionally, the published work had the following impact on the international security community: Link-1 attacks were implemented in the game network of the NATO CCD CoE executed Cyber Defence Exercise (CDX) "Locked Shields 2017" and successfully executed by the CRT against defending *Blue Team* systems; enhanced discussions at NATO Communications and Information Agency (NCIA) on accelerating the Link-1 protocol revision and its long-term deprecation plans; as well as further applicability for Industrial Control System (ICS)/Supervisory Control and Data Acquisition (SCADA) protocol reverse-engineering and attacks [6]; and generating malicious traffic for testing the unsupervised framework for detecting anomalous Syslog messages [82].

**Tactics and procedures.** It has to be noted, that vulnerability identification and exploit development can be a lengthy and tedious task, however, the cyber red team toolset, such as, *Bbuzz*, provides the necessary means to ease, automate and deliver results faster. Though the approach not being stealthy in its nature, it gives the innovative way on finding targeted vulnerabilities in the adversary's information systems and developing custom exploits, which will raise the level of stealth and increase operational success. For this to be possible either an initial intelligence information is required or the reconnaissance needs to be performed. Intelligence information provided either by the intelligence service or collected by the red team through open source intelligence (OSINT) will give a starting position on understanding if such technique and tools are applicable to achieving this goal. If the operational and time constraints allow and this is deemed as a valid option to be pursued, then further data might be required. Additional information, depending on the target exposure could be collected via the means of active (e.g., network port scans, banner grabbing) or passive (e.g., using online solutions such as *Shodan*, or any other applicable OSINT technique) reconnaissance. The gathered information would allow the cyber red team to attempt to replicate the target system in a controlled and closed testing environment. Targeting software and communication protocols via methods, such as, fuzzing, can be applicable throughout the cyber red team attack life-cycle to find vulnerabilities or ways on how to otherwise abuse the target under test.

From the perspective of custom exploit development for the initial adversary network targeting to gain the foothold, this can be seen as one of the valid options for stealthy network entry. Especially favoured in case when traditional entry methods, such as, *spear-phishing* campaign or known exploit execution might be not desired as they could raise attention and trigger alerts. As well as, in cases when the external network services have limited attack surface and only few options to attempt remote attacks are viable. To attempt such an attack, the initial information is preferred well in advance due to time requirements for reference system creation, vulnerability identification, and exploit verification. However, fuzz-testing can be integrated in any phase of activity to explore alternative ways while pursuing ready-available attacks paths, since it is always available and relatively easy and fast to be set-up.

Such focus of force on one or small set of targets for finding vulnerabilities has its risks and benefits. Investing resources in finding a targeted vulnerability in a small set of services, instead of attempting on finding attack vectors in every exposed asset, has to be well assessed from the perspective of sensibility of attack, likelihood of possible flaws and expertise required to transform them into functioning exploit. However, in a successful case such attack vector can be a significant asset when executing computer network operations or assisting other related activities. Not only support to achieving the initial foothold can be obtained but depending on the possible and desirable effects also other direct or indirect impact can be inflicted on both the cyber and kinetic compo-

nents. Not always the targeted element of the system is the intended target, but it can serve as an indirect mean to accomplish the desired effect, such as, obscuring adversary's situational awareness. Taking into consideration all the aforementioned limitations and advantages, the successfully identified custom vulnerability will grant a small cyber red team a higher success of operation execution.

The presented and verified concepts and approaches contribute to the following operational requirement development: stealthy and innovative, agile and available, focus of force, targeted and pervasive, rapid and timely, effective, and asymmetric.

### 4.2.2 Establishing Command and Control Channel

**Techniques.** Publication I presents the novel and simple ways for covert channel creation based on IPv6 transition technologies [8]. The fundamental approaches are based on innovative and creative use of existing technology present in current computer networks. The covert channel creation approach relies on IPv6 transition mechanisms, such as, dual-stack, encapsulation and tunnelling. These technologies exist in vast majority of current computer networks and are supported by nearly all network communication devices and operating systems. Based on this, it is possible, for example, to create an egress covert channel from the dual stack network, where network engineers have implemented IPv4 addressing scheme, but are not controlling the IPv6, either because not being aware of it or not having a full understanding on how to properly implement, control and secure it. One method uses the IPv4 as a transport layer to establish an IPv6 connectivity by the means, such as, *6in4* encapsulation or Generic Routing Encapsulation (GRE) tunnelling. Additionally, for a dual-stack network, it is possible to establish multiple simultaneous IPv4 and IPv6 connections to various destination IP addresses and exfiltrate data over those. As verified in the experiment, the Network-based Intrusion Detection System (NIDS) would not be able to establish the context, correlate the packets and perform analysis which are split and sent over IPv4 and IPv6 in a randomly selected order. This happens because two different IP stacks are used for packet delivery and tested NIDS solutions do not treat such split packets as belonging to the same network data stream. This was identified as a fundamental flaw in the NIDS implementations requiring their redesign and detection algorithm remodelling.

**Tools.** Based on these principles, two tools – *tun64* and *nc64*, are prototyped, described and thoroughly tested by the author and a team of anomaly and intrusion detection experts. Tests are performed against a set of commercial and open-source solutions, such as, *Snort*, *Suricata*, *Bro*, and *Moloch*. The commercial tools are not mentioned explicitly due to discretion and vendors not agreeing to allow their names to be announced, however, among those are the market leaders for Intrusion Detection System (IDS), NIDS and Data Loss Prevention (DLP) products. Prototyped attack tools are tested alongside with other common covert channel creation techniques, such as, HTTP, DNS, ICMP, SSH and *netcat* based tunnels, running on applicable and various common TCP and UDP ports. In a conducted experiment, the proposed approaches are verified to be capable of successfully bypassing the NIDS detection and allowing covert channel establishment to be used for various purposes, such as, Command and Control (CnC) channel establishment and data exfiltration. Furthermore, the *nc64* tool has been successfully implemented into the cyber red team tool-set for data exfiltration and attack execution against the defending blue teams within the largest international live-fire technical cyber defence exercise "Locked Shields". Both prototyped tools[23] [24], written in Python, are publicly released on

[23] tun64. `https://github.com/lockout/tun64`. Accessed: 01/10/2018

[24] nc64. `https://github.com/lockout/nc64`. Accessed: 01/10/2018

| | **Snort SF** | **Snort ET** | **Suricata** | **Bro** | **Moloch** |
|---|---|---|---|---|---|
| *http-t-80-4* | N | N | V | V | V |
| *iodine-u-53-4* | N | N | Y | P | V |
| *ptunnel-icmp* | N | Y | N | V | V |
| *netcat-t-80-6* | N | N | N | V | N |
| *ssh-t-80-6* | N | N | V | P | N |
| *tun64-t-80-t6over4* | N | Y | Y | P | N |
| *nc64-t-80-6/4* | N | N | N | P | V |
| *nc64-t-443-6* | N | N | N | V | N |

*Table 5: Common tunnelling method detection comparison to nc64 and tun64*

GitHub under the MIT license and available to be used by everyone.

The table 5 represents a shortened and condensed version of findings from Publication I related to developed tool *nc64* and *tun64* comparison to other commonly used protocol tunnelling method (represented in *italics* in the table) detection by popular open-source monitoring, NIDS, and Deep Packet Inspection (DPI) solutions (represented in **bold** in the table). The chosen monitoring tools are *Snort* NIDS with Source Fire (SF) and Emerging Threats (ET) signatures, Suricata NIDS, *Bro* and *Moloch* DPI solutions. The tested popular commercial NIDS and DPI solutions are not listed or mentioned due to signed non-disclosure agreements, however, the prototyped approaches were able to circumvent the detection with high success ratio. The listed tool configuration notation follows the following convention *tool_name-transport_protocol-port-IPversion(s)*, for example, the *nc64* tool running over TCP to a destination port 80 using IPv6 and IPv4 interchangeably would be written as *nc64-t-80-6/4*. The test outcomes are labelled as follows: a positive match (denoted by letter Y in the table) clearly identified a malicious activity and triggered alerts, partial or abnormal footprint (denoted by letter P) raised the alert but did not provide appropriate information, potential visible match (denoted by letter V) requires human analyst or sophisticated anomaly detection for a positive match verification, and the worst case (denoted by letter N) does not generate any visible alerts or logs. From the presented results it can be seen, that the *nc64* tool is successful on evading the implemented automated threat detection solutions and has a higher evasion success rate than other protocol tunnelling methods, as well as, it has not been fingerprinted in comparison to the well-known *netcat* tool.

Furthermore, these techniques support target system remote access from nearly any location in the cyberspace. Major global initiatives executed by the global standardization and industry leaders, such as, “World IPv6 Day”, are accelerating the introduction of IPv6 throughout the Internet with its backbone already being fully IPv6 but lagging at the network edges. To support the IPv6 connectivity for the IPv4 networks, the transition mechanisms are introduced and maintained until the Internet has fully migrated to IPv6. Widespread deployment and availability of IPv6 and required transition mechanisms make this attack approach, as implemented in the *nc64* and *tun64*, global and available throughout the cyberspace. Furthermore, it can be assumed, that IPv4 enabled networks with transition mechanisms enabled will remain for an undefined period of time.

Additionally, the published work had the following impact on the international security community: EUROPOL European Cybercrime Centre (EC3) released a security warning on IPv6 vulnerabilities [22]; Forum of Incident Response and Security Teams (FIRST) annual

conference sparked discussions on IPv6 security[25]; IETF discussions on protocol specification updates and transition mechanism deprecation (private e-mail exchange between the IETF representatives and the author); NIDS vendor system updates (private e-mail exchange between the vendor and the author); and multiple news articles[26] [27].

**Tactics and procedures.** The implemented tools and approaches provide high level of flexibility and usage applicability since they are built on top of a network protocol stack supported by nearly all modern network devices and operating systems. Such capability yields to the cyber red team with extra level of stealth due to readily-available technology use in the modern computer networks. Both the *tun64* and *nc64* tools are applicable to achieving this goal, however, *nc64* showing better results in circumventing network security solutions. Since the *tun64* amd *nc64* tools, alongside with the automated testing network creation scripts, have been publicly released, they are freely available for the cyber red team to be used for any tailored access operations.

The main area of such technique applicability is for maintaining control over the compromised assets in the target network, either in the first stages of the attacks or when moving laterally in the network and searching for the intended target. Such created CnC channels permit not only the control over the specific systems, but also any other data exchange, such as valuable data exfiltration. Furthermore, when engaging in lateral movement within the dual-stack network such approach can be used to maintain the desired level of stealth. This can be accomplished either by moving from one compromised host to another by the means of such techniques, or interconnecting internal network nodes to create a path which is harder to be traced back to its origin and initial entry point. When targeting multi-tiered networks such as ICS/SCADA, consisting of various in-depth network segments, maintaining a stable CnC channel is of a high priority to deliver the designated impact to the target systems.

Detecting such implemented covert channel was identified to be extremely hard, even by a human analyst, since there were no known patterns or signatures to be matched against the large volume of data collected by the DPI. The common CnC channels would rely on using typical protocols such as DNS, HTTP, and within the MS Windows network – SMB, to carry the data in the protocol payload fields. These approaches are well known to the analysts, even if the covert method is not known. If the IPv6 transition mechanism based covert channel is implemented properly, such as, network port aligned with its expected payload headers (e.g., HTTP headers over TCP/80), then the chances of remaining undetected are increasing. This consideration is applicable to any other technique used by cyber red team, for example, when targeting or impersonating a particular network service, the performed activities have to comply with at least the expected patterns of timing, network protocol, source and destination ports, and protocol payload main features. From the conducted experiments, it was identified, that the prototyped *nc64* tool had the best detection evasion indicators when using the following ports – UDP/22, TCP/443, UDP/443, and UDP/80. The presented and verified concepts and approaches contribute to the following operational requirement development: stealthy and innovative, agile and available, and targeted and pervasive.

[25] FIRST Conference 2018. F.Herberg, SWITCH. `https://www.first.org/resources/papers/conf2018/Herberg-Frank_FIRST_20180624.pdf`. Accessed: 23/09/2018

[26] InfoSecurity. "NATO CCDCoE: IPv6 Transition Opens Up Covert Info Exfiltration." `https://www.infosecurity-magazine.com/news/nato-ipv6-transition-opens-up/`. Accessed: 23/09/2018

[27] Slashdot. "Tunnelled IPv6 Attacks Bypass Network Intrusion Detection Systems." `https://tech.slashdot.org/story/17/04/09/0452220/tunnelled-ipv6-attacks-bypass-network-intrusion-detection-systems`. Accessed: 23/09/2018

### 4.2.3 Delivering the Impact

**Techniques.** Publication III displays the novel and automated approaches for ICS/SCADA protocol, system takeover and process control [6]. The basic ideas represented in this work are vulnerability location methods, verification and weaponization for critical impact infliction. Utilizing these approaches, the ICS/SCADA network protocols – PROFINET IO and IEC-104, dominantly used in European automation and power grids, were reverse-engineered by the author and approaches for malicious command injection developed. The described attack vectors, explained by Proof of Concept (PoC) code, were exploited to successfully compromise the industrial and electrical power grid process. The techniques of identifying such attack vectors and exploiting them rely primarily on the protocols lacking the security features. Even if security mechanisms exist, such as, IEC-104 security extensions they are seldom implemented by the vendors and even more rarely deployed by the system engineers in the production environment. The protocols, developed for air-gapped systems aimed at high availability and safety, lack the required security features, such as, integrity and authentication. This becomes even more critical when such initially serial-line proprietary communications are merged with the TCP/IP protocol stack, connected over industrial Ethernet, and commuted by the use of traditional IT equipment. The expected separation between the operational and information technology is becoming very vague and such systems can be targeted to deliver serious impact by the attacker originating from the Internet.

**Tools.** The described four novel and critical attacks against Critical Infrastructure (CI) components researched and developed by the author allow to deliver a devastating impact on the affected and vulnerable systems, either by compromising the whole industrial process, controlling it, or inflicting potential physical damage to the ICS equipment. First attack, aimed at globally deployed and used PROFINET real-time protocol PROFINET IO, allows to inject rogue control frames on the network to control the industrial process. Second one (CVE-2018-10603; CVSSv3 10.0 [CRITICAL IMPACT]), targets a major industrial Ethernet protocol IEC-104 which is used worldwide in energy sector and permits to inject control commands allowing to tamper with the power grid and disable the power supply. Third (CVE-2018-10607; CVSSv3 8.2 [HIGH IMPACT]), aims at causing the DoS condition in the IEC-104 enabled systems through improper use of the protocol, which denies the supervision and control of the power grid. Fourth (CVE-2018-10605; CVSSv3 8.8 [HIGH IMPACT]), being targeted at a Martem TELEM-GW6e protocol gateway – Remote Terminal Unit (RTU), a critical component of the ICS enabling communication and control of the deployed field devices, permits full remote takeover of the device and full compromise of the controlled industrial process. Additionally, XSS vulnerability was identified by the invited expert in the RTU web-based management console (CVE-2018-10609; CVSSv3 7.4 [HIGH]). All of these attack vectors were responsibly disclosed to the vendor, US DHS ICS-CERT, and the international CSIRT community. After the US-CERT and vendor released security advisories, and only once the patches for the vulnerabilities were released, the PoC code[28] [29] [30] [31] was made publicly available on GitHub under the MIT license for attack vector testing, verification, and mitigation.

[28] iec104inj. `https://github.com/lockout/iec104inj`. Accessed :01/10/2018

[29] profinet-poc. `https://github.com/lockout/iec104inj/tree/master/poc/porfinet-poc`. Accessed: 01/10/2018

[30] iec104dos-poc. `https://github.com/lockout/iec104inj/tree/master/poc/iec104dos-poc`. Accessed: 01/10/2018

[31] gw6e-poc. `https://github.com/lockout/iec104inj/tree/master/poc/gw6e-poc`. Accessed: 01/10/2018

Furthermore, the published work had the following impact on the global ICS/SCADA security community: vulnerabilities were reported to the vendors and the US DHS ICS-CERT, which resulted in security advisories published [18] [47] and patches developed; vulnerability technical information and PoC attack scripts were disclosed to the international CERT community and ICS operators; four Common Vulnerabilities and Exposure (CVE) numbers assigned [53] [54] [55] [56]; attacks were implemented in the game network of the NATO CCD CoE executed CDX "Locked Shields 2018" and cyber red team oriented technical exercise "Crossed Swords 2018", and successfully executed by the red team; and was addressed by multiple major vulnerability tracking databases[32] and news articles[33].

**Tactics and procedures.** Form the operational perspective these vulnerabilities give their user superiority and allow to inflict debilitating damage to the target system, thus making them powerful weapons in the CRT arsenal. It has to be noted, that reaching such critical systems in most cases will require a successful execution of previous attack stages, such as, initial foothold, command and control, lateral movement, and asset identification. There are cases when such systems are either directly exposed to the Internet or are available in one network hop distance from the initial foothold. More realistic, in case of military network critical components is that they either would not be connected to the Internet or would use dedicated and encrypted communication lines or channels. In such cases any other alternative options have to be explored by the cyber red team, such as, breach of supply chain integrity, physical access or removable media dissemination. However, to deliver an impact to a military system it is not necessary to target it directly. Such impact, either kinetic or cyber, can be achieved by targeting any other system or network on which it depends. Reaching the final objective would be assisted by successfully executing all previous attack stages, which are supported by the stealthy entry and control channel.

As is the case with the initial attack vector identification and exploit development, also imposing a specific effect on the target system, might require time and in advance preparation. The actual time required for developing attacks against ICS components took the author no more than three days, however, the most time intensive parts were the reference network creation, implementation and configuration, and afterwards – the developed attack verification under various circumstances and configuration settings. Such additional crucial activities are time intensive and demand significant amount of time depending on the complexity of the target system, required resources and skills. Reference system development and verification of attacks for the author took around one month working together with the vendor engineers.

Developing custom attacks is not a straightforward approach and has multiple fundamental requirements, such as, expertise and experience, knowledge of right approaches and techniques, ability to use existing tools and develop new ones, and having an idea where to look for potential vulnerabilities. New ideas on finding and developing unknown vulnerabilities (i.e., "zero-days") will present the CRT with an significant advantage of inflicting a critical damage while circumventing the target defences. Depending on the allocated and available time-frame, such attack development might not always be plausible, but if intelligence information is provided and assessed early enough, such attacks can be developed in advance and used in the responsive computer network operations. De-

[32]SecurityFocus. Multiple Martem Products Multiple Security Vulnerabilities. `https://www.securityfocus.com/bid/104286`. Accessed 23/09/2018

[33]SecurityWeek. Vulnerabilities Found in RTUs Used by European Energy Firms. `https://www.securityweek.com/vulnerabilities-found-rtus-used-european-energy-firms`. Accessed: 23/09/2018

veloped tools can be launched in a coordinated manner in support for other responsive activities and can inflict damage both directly and indirectly to cyber and kinetic components of the target system. Such bespoke approach allows to focus the attack on the most critical components of the target system to impose a significant damage and, if combined with the covert channel techniques, can be executed remotely to incur a significant operational advantage over the adversary.

For a weakness to become a vulnerability it has to be exposed and an applicable method for its exploitation has to be identified. Such process can be time, skill and resource demanding can have a high failure ratio and varies from target to target with no predefined exploit development path available. Common Weakness Enumeration by MITRE corporation[34] strives to deliver a list of known weaknesses, which can be targeted to potentially discover an exploitable vulnerability for the system under test. General types of weaknesses identified leading to their exploitation and industrial process control as described in Publication III: missing authentication for a critical function (CWE-306) (e.g., no authentication or encryption) allows the extraction of the protocol payload and its reverse engineering; missing integrity verification (CWE-353) (e.g., no firmware update integrity checks performed) permits crafted malicious system update to be committed and executed on a remote system; incorrect default permissions (CWE-279) (e.g., limited user can overwrite system files) grants an opportunity to modify or replace critical system files thus compromising the whole system; and use of hard-coded and insufficiently protected credentials (CWE-798, CWE-522) (e.g., default credentials embedded in the software) permits easy extraction of default credentials and unsanctioned access to the remote system.

In a very broad strokes, the vulnerability identification process can be split in the following general steps: target examination (e.g., protocol analysis, system examination, software debugging, sand-boxing); potential weakness identification (e.g., expert analysis, security auditing, fuzz-testing); weakness exploitation attempt and verification (e.g., attack prototyping, condition verification); and final exploit preparation. To speed up this process, applicable techniques and tools need to be utilized, such as, use of suitable programming language to for tool development and exploit prototype. Apart from C/C++ programming languages, Python language is often used for attack prototyping and has been extensively used by the author (Publication I, Publication II, and Publication III). The choice for Python in most cases is favoured due to reasonably easy learning curve, source code readability, and vast community support with third-party modules and frameworks oriented at exploit development and attack prototyping, to mention few, *Scapy* packet manipulation tool, *Radare2* reverse engineering framework, *Capstone* disassembler, *Unicorn* CPU emulator, *Snap7* Siemens S7 communication suite, *Metasploit* exploitation framework, and *Veil-Evasion* common anti-virus solution bypass.

The presented and verified concepts and approaches contribute to the following operational requirement development: stealthy and innovative, focus of force, targeted and pervasive, integrated and coordinated, effective, and asymmetric.

### 4.2.4 Countering the Cyber Attack Kill Chain

The main concepts of the cyber attack kill-chain are introduced in chapter 2.4. The three most widely acknowledged and used cyber attack kill-chain approaches are the Lockheed Martin "Cyber Kill Chain" (Fig.4) [38], Mandiant/FireEye "Attack Lifecycle" (Fig.5) [45], Microsoft "Attack Kill Chain" (Fig.6) [51], SANS "The Industrial Control System Cyber Kill Chain" [4], and MITRE "ATT&CK" [57]. When compared, the differences between the pro-

[34] Comprehensive CWE Dictionary. `https://cwe.mitre.org/data/definitions/2000.html`. Accessed: 28/10/2019

posed methodologies are different. Lockheed Martin puts the most effort in identifying and stopping the attacks in their early stages but does not focus too much on the adversary performed activities internally. Mandiant/FireEye, on the other hand try to grasp both initial and internal attack stages in a very broad strokes, not being too granular on every specific stage. However, Microsoft proposed approach elaborates very deeply on the malicious activities performed within the network, but leaving the initial attack stages not deeply covered. All of these methods have their strengths and weaknesses, but when rationally combined should provide the most comprehensive approach bringing out the most important stages of the attack. From a different angle, MITRE ATT&CK maps adversarial Techniques, Tools, Tactics and Procedures (TTTPs) to the various stages of the attack preparation[35] and execution [36]. Also, SANS proposed kill-chain complies with the main attack stages, as described by other approaches, but adds extra nuances related to learning and targeting the industrial control systems and processes.


*Figure 4: Lockheed Martin Cyber Kill Chain [38]*

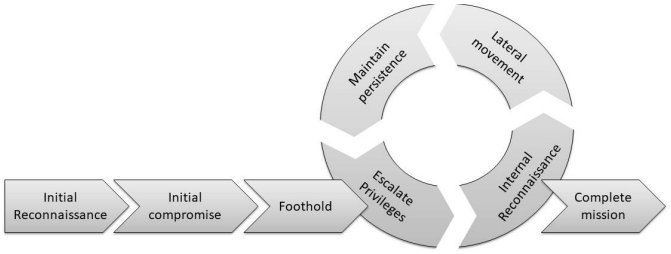



*Figure 5: Mandiant/FireEye Attack Lifecycle [45]*

By design the cyber attack kill-chain is developed from the defensive perspective to eliminate the adversarial threat and cyber attack at every stage of its execution. When considering the kill-chain from the attacker's perspective it is critical to be able to counter the kill-chain by applying novel TTTPs to successfully execute every attack stage without being detected and disrupted. Therefore, cyber red team engaged in computer network operations have to employ adequate TTTPs to raise the level of success. The table 6 on page 52 attempts to represent the stages of every individual approach, and groups them to create the cyber attack kill-chain with the most important stages of the attack, to give enough detail to the initial attack, internal lateral movement, and asset reconnaissance stages. In the table, the *'X'* denotes, that a particular technique is directly applicable for countering a particular stage of the cyber attack kill-chain, and *'S'* identifies, that this method can be used to support countering that particular stage in combination with other applicable approaches and tools. The cyber attack kill-chain stages are mapped against the methods and techniques presented in the publications, which are applicable to countering or mitigating the effects of every cyber attack kill-chain stage.

[35]MITRE PRE-ATT&CK Matrix. `https://attack.mitre.org/matrices/pre/`. Accessed: 27/10/2018

[36]MITRE ATT&CK Enterprise Matrix. `https://attack.mitre.org/matrices/enterprise/`. Accessed: 27/10/2018

| Cyber Kill Chain | Attack Lifecycle | Attack Kill Chain | Cyber Attack Kill Chain | Pub.I | Pub.II | Pub.III |
|---|---|---|---|---|---|---|
| Reconnaissance | Reconnaissance | Reconnaissance | Reconnaissance | S | | |
| Weaponization | Initial compromise | Initial compromise | Initial compromise and foothold | S | X | |
| Delivery | Foothold | | | | | |
| Exploitation | Escalate privileges | | | | | |
| Installation | | | | | | |
| Command and control | | | Command and control | X | | |
| | Internal reconnaissance | Internal reconnaissance | Internal reconnaissance | S | | |
| | Lateral movement | | Lateral movement | X | | |
| | | Local privilege escalation | Privilege escalation | | X | S |
| | | Compromise credentials | | | | |
| | | Admin hunting | | | | |
| | | Remote code execution | | | | |
| | Persistence | Domain dominance | Persistence | | S | |
| | | Remote code execution | Asset reconnaissance | S | | |
| | | Asset reconnaissance | | | | |
| | | Local privilege escalation | | | | |
| Actions on objectives | Complete mission | Asset access | Objective completion | S | X | X |
| | | Exfiltration | | | | |

*Table 6: Cyber attack kill-chain grouping and mapping to countering techniques in publications*

This allows the Cyber Red Teaming (CRT) to adapt and assess their Techniques, Tools and Procedures (TTPs) applicability to allow the Computer Network Operation (CNO) execution with a higher success rate.

To address the strengths, weaknesses, and synergies of the popular cyber kill-chains, the author proposes an unified cyber attack kill-chain approach (Table 6 on page 52, column "Cyber Attack Kill Chain"). The proposed model (Fig. 7) consists of the following phases: reconnaissance, initial compromise and foothold, command and control, internal reconnaissance, lateral movement, privilege escalation, persistence, asset reconnaissance, and objective completion. In this model, the reconnaissance and persistence loops are intertwined, with internal reconnaissance permitting lateral movement, privilege es-

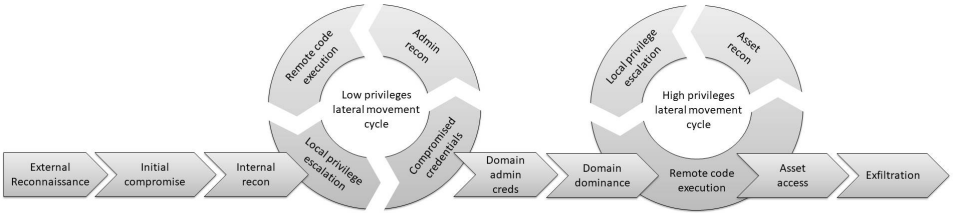



*Figure 6: Microsoft Attack Kill Chain [51]*

calation and persistence, and allowing to conduct further reconnaissance activities on intended targets. If any applicable information or vulnerability is identified within the reconnaissance and lateral movement phases, the privilege escalation and persistence can be performed if necessary. However, not always privilege escalation and persistence are required to achieve the designated effect, therefore objective can be accomplished as soon as the intended target is located and the adequate network position and level of privileges have been obtained to perform the assigned task.

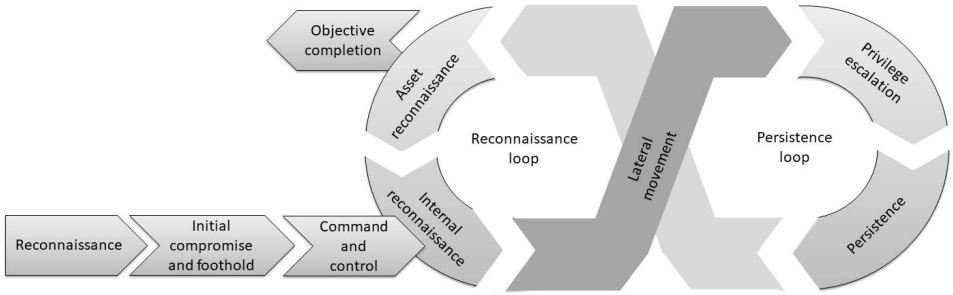



*Figure 7: Proposed cyber attack kill-chain model*

As stated in the chapter 4.1, such specialized cyber red team responsive operations should be capable of delivering the maximum possible effect. With this in mind, the red team needs to ensure highest success rate at every performed stage matching the cyber attack kill-chain. Such procedures would allow the applicable technique and tool proper use for accomplishing the mission objectives, maintaining the Operational Security (OPSEC), and own asset protection.

## 4.3 Chapter Conclusions

The table 3 on page 41 shows that the proposed methods and techniques in the research papers, are applicable to the specialized CRT responsive CNO and cover all of the aspects of such operational TTTPs and demands. The majority of the proposed methods support the custom technique and tool development aimed at specific elements of the targeted system. This, with according tactics and procedures, results in a higher stealth level since these attacks are tailored to the intended objective to deliver the desired operational effects with highest impact possible. Additionally, these techniques are aimed at high stealth to provide a higher operational success rate, such as, establishing Command and Control (CnC) channel to circumvent the Network-based Intrusion Detection System (NIDS). The tables 4 and 5 compare the author's developed methods against the other popular solutions. These unique and bespoke techniques are more effective when

compared to relevant and commonly used ones (Publication I), grant the ease and flexibility to increase the speed of operation execution (Publication II), and allow focus of force on the most critical components of the target system (Publication III). Based on the validated research in the publications, there are no other identified known approaches publicly available to match the level of the developed and presented techniques and tools. All of the proposed new techniques in this chapter have been implemented into the cyber red team oriented technical exercise series “Crossed Swords” and is described in a more detail in the chapter 6.

A new unified cyber attack kill-chain method is introduced (Fig. 7) to find the synergies among the accepted kill-chain models to mitigate the gaps and emphasize their strengths. The methods and approaches, proposed in the publications, are aimed at producing the desired effect and objective completion in the final stages of the cyber attack kill-chain. As it is identified, they cover all of the kill-chain stages either directly or can be used to support countering most of the stages, such as, allowing a local privilege escalation exploit delivery to the target system over the established covert CnC channel. The ultimate goal for the presented techniques is to grant the most benefit to the CRT from both the operation execution and objective accomplishment. When comparing their applicability both to specialized CRT responsive CNO execution and countering the cyber attack kill-chain, it can be assessed, that they provide a balanced approach and support to achieving the presented demands.

# 5 ADVERSARY DETECTION AND RED TEAM ASSET PROTECTION

The cyber red team has to maintain the visibility over the defended infrastructure and its own assets to ensure tracking of malicious activities, perform adversary assessment, gather further technical information aiding the attack trace-back and attribution, and protection of red team's operational infrastructure. Such detection and asset protection techniques benefit the cyber red team responsive operation execution and from the Observe, Orient, Decide, Act (OODA) loop perspective they contribute to the observation and orientation actions. Additionally, Operational Security (OPSEC) requirements have to be complied with to protect cyber red team assets, which are composed not only from the hardware and deployed software entities in cyberspace, but also the team identity and their operational goals. Applicable techniques and solutions are presented in the listed publications (Publication IV, Publication V, and Publication VI) and suitable use cases are assessed.

In contrast to a widely accepted belief, that attack is the best defence, the attackers might forget about defending their own assets and fail at insuring appropriate operational security measures. Such mistake can jeopardize the whole operation and lead to the full compromise of the cyber red team's infrastructure. This, most importantly, comes into consideration when the adversary is not only engaged in attacking the target infrastructure, but might pursue the defenders engaged in a responsive cyber defence. It has to be assumed that for every action there could be a counter action performed, thus leading to counter-red team operations and even to counter-counter-red team operations. Defending party, executing a responsive cyber operation, is conducting operations from the infrastructure outside their defended network, which requires an equal and, in most cases, better protection. Such requirements stem from not only the threat detection perspective, but own asset, position, sensitivity of the pursued operational goals, infrastructure and identity protection requirements.

To provide such capabilities, the cyber red team has to have the expertise to implement, configure, supervise and monitor the threat detection solutions. Such defence has to be implemented to cover the red team infrastructure's perimeter, as well as, activities happening within. For the execution of crucial cyber operations the required red team environment would be custom made and deployed according to the operational needs, and afterwards securely destroyed. To ensure such demands, the implemented defensive solutions have to meet at least the following criteria: readily-available, ease of deployment and management, flexible and scalable, high level of automation, and lightweight. Additionally, such solutions should be easily integrated with the technologies already present in the deployed network infrastructure elements, such as, *rsyslog* or *Syslog-ng* system logging services. Furthermore, these techniques, as well as the rest of the infrastructure, should be maintained to be as untraceable as possible and complicating the attribution. For such matters well developed, readily-available, non-commercial or open-source solutions providing high customization and flexibility would be applicable. Two approaches complying with these requirements have been identified as system log (i.e., Syslog) based analysis and cyber deceptions.

In addition to automated syslog-based detection and cyber deception techniques, other threat assessment solutions can be used as well. Such solutions should comply with the mentioned criteria to be applicable for the cyber red teaming requirements. The proposed framework, named *TED* (Publication IX), can be used on any modern GNU/Linux system with operating-system-level virtualization solution *Docker* deployed. Foremost, such approach would be primarily used in the first stages of incident response on the systems, where no acceptable level of protection and threat detection solutions have been

deployed or requiring additional in-depth assessment. *TED* bundles most common local system and *ELF* binary file security assessment tools, such as, *checksec.sh*, *Lynis*, and *Spectre/Meltdown* checkers. All of the tools, their dependencies, management and orchestration scripts are included in a *Docker* container, which can be used either in Internet connected or disconnected systems, and requiring only single requirement of *Docker* engine being present. This engine is included in majority of the popular modern GNU/Linux distribution repositories and, in most cases, already being installed on systems, such as, application, and web servers. Such lightweight solution allows it to be easily used to establish system security level and identify potential binary file attacks, without introducing significant changes or requirement of installing other solutions on the target system, thus minimizing the contamination of the examined system. The potentially compromised system analysis has to be performed according to the digital forensics requirements, with assessing the system compromise likelihood and at least acquiring memory and disk images, before proceeding with more intrusive activities. Such activities may provide the needed situational awareness picture and assist initial technical attribution establishment, before engaging the specialized cyber red team into the computer network operations against a suspected actor in the cyberspace. Whenever the cyber red team moves from their defended information system into their operational infrastructure, from where the responsive operation will be carried out, the deployed assets in that infrastructure, in most cases, become disposable if deemed compromised. *TED* may be used to assess the required hardening level of the initially deployed operational systems or in specific cases – to examine the cause of their compromise. The primary solutions, compliant with the requirements and to be used within the defended and operational infrastructures, consist of system log file-based anomaly detection and cyber deception solutions.

## 5.1 System Log File-Based Anomaly Detection

System log file monitoring and analysis has been acknowledged as an important network and system management technique, as well as, granting the possibility of detecting anomalies and security violations. Under normal circumstances, all network devices, such as, hosts, sensors, networking equipment or sophisticated data parsing and threat detection solutions, should be able to generate alerts and system logs in a textual format. It is highly recommended for such data to be delivered to a centralized storage location for retention, further analysis and correlation. Depending on the solution, the generated alerts and system logs can have a different representation, which might be parsed either without additional processing or requiring normalization or transformation to an unified standard. Despite this, easy to deploy and flexible system log processing and correlation method implementation into the cyber red team tool-set could grant the required visibility over the defended and protected assets.

Such system log file analysis should be performed for the defended network infrastructure, as well as, the operational one used by the red team. Despite both environments collecting the log files, their design implementations could differ. Defended infrastructure could generate and record events in the log files, which are then transferred to the central location for retention and analysis, depending on the security policies and other requirements. The protected cyber red team operational infrastructure would follow similar approach, however, it might be beneficial not to keep the log files on the red team hosts or its supporting infrastructure, but directly delivered to the central secure location for immediate analysis and threat assessment. Such approach would be considered in case of a likely counter red team operation execution by the adversary and attacker possibly gaining access to the defending red team's infrastructure elements. This potentially could minimize

the exposure regarding the executed responsive activities, intended goals and attribution.

Event correlation has established itself as a prominent and recognized monitoring technique, which is essential for establishing situational awareness. The Simple Event Correlator (SEC) (Publication IV), written in Perl, runs on all modern UNIX based platforms, has been used for a wide range of purposes and environments as an efficient open-source alternative to the commercial solutions. This solution is designed for real-time event processing and incorporates event matching, parsing, and output generation. The SEC uses scalable rules for event correlation, which can be applied to a range of inputs, including the Syslog files.

This lightweight, flexible and real-time nature of SEC complements the cyber red team operational requirements, allowing focus on the operation itself while gathering and receiving real-time alerts. Such visibility, enhanced by the user interface such as *Kibana* from the *ELK Stack*, can deliver the required situational awareness regarding the cyber red team's operational infrastructure. Gained awareness not only allows pinpointing the system failures or monitor the infrastructure performance, but allows to detect anomalies, such as, attacks or unsanctioned access attempts.

Despite the SEC being powerful event correlation solution, it relies on its rule-sets, which have to be created by the human analyst. A novel data mining-based framework is developed (Publication V), requiring no human intervention, for fully automated rule discovery for real-time detection of anomalous messages from Syslog-based logs. This approach possesses the capability to adapt to the changes in the system and employs the *LogCluster* algorithm for data mining. Human expert can extend the framework with own created rules, thus aiding the anomaly detection or adjusting the system to the specific design implementation or operational goals.

The detection of previously unknown error conditions and anomalies has been a difficult problem, however, the proposed framework provides a one practically applicable solution to it. The cyber red team implementing this solution on the centralized Syslog collector, to which the individual host system logs are transferred over a reliable and encrypted channel, gains an immediate benefit with least efforts required. As the conducted experiments over a larger period of time have indicated, the anomalies and unexpected system state conditions have been successfully identified. Furthermore, the test involving the running of *Bbuzz* fuzzing framework for abnormal network communication generation, clearly confirms that unknown anomalous or malicious requests, resulting in the Syslog entries, are detected and reported. It has to be noted, that cyber red team protected operational infrastructure should implement also the traditional defensive mechanisms (i.e., passive defence), such as, packet filtering on all network hosts (e.g., *iptables*, *ip6tables*, *Firewall & network protection*), deploy proper system hardening (e.g., *SeLinux*, application white-listing), host protection with Host-based Intrusion Detection System (HIDS), system disk encryption (e.g., *LUKS*, *BitLocker*, *VeraCrypt*), and user privilege level separation and control. When deploying the defensive measures, they have to be assessed from the operational security considerations to confirm, that no information is leaked out from the infrastructure to the third parties, such as, anti-virus, HIDS or Intrusion Detection System (IDS) alerts and metrics. When considering a potential counter-red team operation executed by an adversary the similar attack approaches have to be expected and system protection should be augmented by the out-of-band (e.g., hardware or software-based port mirroring) network traffic monitoring and analysis solutions, such as, *Snort*, *Suricata* and depending on available resources and team capabilities – *Bro* and *Moloch*. These solutions in turn would also generate the alerts and system logs, which can be transferred to the central location for the data mining, correlation and security incident infor-

mation extraction. If the cyber red team assumes the high risk of the counter-red team operation plausibility, then further in-depth system monitoring solutions can be considered for deployment to deliver additional visibility, such as, system performance metrics (e.g., *sysdig*, *Telegraf*), kernel requests (e.g., *Snoopy*), unsanctioned system use (e.g., *tripwire*).

## 5.2 Cyber Deception-Based Detection

All of the presented passive monitoring and threat detection techniques should be augmented with the active cyber defence elements within the cyber red team's operational infrastructure. The most prominent one, as described in the chapter 2.1, has been identified as the cyber deception approach. If deployed either in the defended system or in the protected cyber red team operational network this set of techniques would grant the immediate feedback to the cyber red team on any identified suspicious activities. Cyber deceptions can further be integrated to deliver their system logs and alerts to the central Syslog processing location. Various types of cyber deceptions can provide different levels of adversary tracking and technical attribution information gathering. The solutions, such as honeypots, would be able to identify unsanctioned access to them and collect information on adversary's activities within the decoy network allowing a better insight into their capabilities, intentions and Techniques, Tools, Tactics and Procedures (TTTPs). Honeytokens could potentially allow the trace-back to the location from where such data has been executed or accessed.

The following cyber deception frameworks were implemented in the reference network, against which the targeted attacks were launched (Publication VI) – *T-pot*[37], SHIVA[38], YALIH[39], KFsensor[40], and ADHD (see footnote 12 on page 23). To conduct the experiments the target network consisted of various zones, such as, demilitarized network hosting external web and e-mail services, internal network for business purposes hosting MS Windows and GNU/Linux clients, and a sub-net for information system security monitoring solutions. Every sub-net represents location for deployment of an applicable cyber kill-chain technique both from the attacker and the defender perspectives. The cyber deceptions, according to their applicability and expected use cases were deployed in the network segments to match the defender's requirements on stopping the attack according to the cyber attack kill-chain stages. Every particular framework or a collection of tools is either oriented towards being deployed on its own client operating system (e.g., GNU/Linux, MS Windows), is a high- or low-interaction honeypot oriented at deceiving the adversary or is a set of tools to be used to actively detect, counter or degrade the attacker's capabilities (e.g., port-spoofing, *tarpit*). The experiment results identified, that out of all implemented technologies the multi-honeypot platform *T-pot* proved to be the most efficient in identifying and detecting attack or malicious activities. *T-pot* includes a set of *dockerized* instances of multiple solutions, such as, *Glastopf, Kippo, Honeytrap, and Dionaea* honeypots, *Suricata* Network-based Intrusion Detection System (NIDS), *ELK* stack, and *ewsposter* for honeypot data sharing. The cyber deception is a valid method for detecting, deceiving, disrupting, degrading, denying, and defending against the adversary.

[37] DTAG Community Honeypot Project. `http://dtag-dev-sec.github.io/`. Accessed: 05/10/2018

[38] Spam Honeypot with Intelligent Virtual Analyzer. `https://github.com/shiva-spampot/shiva`. Accessed: 05/10/2018

[39] Yet Another Low Interaction Honeyclient. `https://github.com/Masood-M/yalih`. Accessed: 05/10/2018

[40] Advanced Windows Honeypot System. `http://www.keyfocus.net/kfsensor/`. Accessed: 05/10/2018

*Detect*. As this being one of the core requirements for any defensive activity, it is required to establish the situational awareness and allow pursuing responsive activities against the detected threat. All of the mentioned passive methods and further described active ones, contribute to achieving this goal. To complement attribution via the active detection means, the most prominent approach could be various cyber deception techniques, such as, *honeytoken* documents rigged with an executable code, or strategically placed decoy data leading the attacker to the deployed honeypots. Rigged documents, such as, attack orders, technical documentation, or other sensitive information, once exfiltrated from the defended infrastructure and opened by an attacker potentially could reveal the adversary position in the cyberspace by *beaconing* its IP address and other sensitive data collected and sent by the embedded computer code. Depending on the adversary's operational security methods, such as, rigged files might be either stripped of executable scripts, opened on an isolated or third-party system, or modified to deliver false information. Despite these concerns, this option should be practised by the cyber red team to raise the level of attribution in case the cyber deceptions are well crafted and placed or if the adversary is careless and makes a mistake.

*Deceive*. Part of defensive activities, especially ones implemented in the cyber red team's operational infrastructure, should be aimed at confusing and deceiving the adversary by any means possible. The techniques, such as, decoys and cyber deceptions will not only allow the identification and tracking of the adversary, but would also allow its attribution, technical capability and goal assessment. The concept of cyber deceptions would include solutions, such as, honeypots which would spoof the attack surface by deploying large quantities of simulated hosts or network topologies (i.e., honeynets). Besides providing deception this technique also supports the degradation of attacker capacity, since adversary would be entangled in sorting identified systems between the real and fake ones. Both low- and high-interaction honeypots benefit the attribution process, additionally granting the insight into adversarial TTTPs in case of a high-interaction honeypot usage. It has to be noted, that high-interaction honeypots might require a larger effort for maintenance and upkeep in contrast to the low-interaction ones but would also give a better understanding of the adversarial objectives and motivation. However, it would be highly suggested to have at least the low-interaction honeypots embedded in the red team's infrastructure. To further control and deceive an adversary decoys could be placed throughout the red team's operational infrastructure. The strategically placed decoys (sometimes called "breadcrumbs"), for example, in the form of cached credentials, saved remote connection sessions or mapped network drives, would try to force the attacker use this seemingly valuable information to follow the predefined path designed by the defender leading to a decoy system. Once this decoy information is used against the decoy system (e.g., honeypot), the alert is triggered, and further perpetrator actions can be monitored and traced. One of the rising solutions in providing such cyber decoy systems based on sophisticated scenario development, "breadcrumb" creation and placement is the *Mazerunner* (see footnote 16 on page 25).

*Disrupt*. If a detected and identified attack becomes too intrusive or tries to overwhelm the protected infrastructure, it is possible to disrupt the connections either by firewalling or *null-routing* them. Such action does not directly aid the attribution or adversary assessment, however, can be used to indirectly limit available attack paths thus potentially steering the perpetrators towards the desired communication channels or approaches which can be controlled or monitored by the cyber red team. Depending on the situation, it might be chosen not to disrupt the adversary activities within the protected networks, but to carry on observing and analysing them to gain more intelligence

information on the threat actor.

*Degrade*. Any active or responsive action depends on time, thus deception and degradation approaches are used for the defenders to gain additional time to respond while opponent is being tricked in performing useless activities. Degradation by inflicting the time penalty is typically executed by at least the following approaches – honeypots to artificially increase the network complexity and size for the attacker to spend time on probing it, network host attack surface spoofing to force attacker wasting time on performing endless network port scans, and *tarpitting* incurring a serious time cost when dealing with slowly responding services to all requests. In some cases, degradation would be seen as better approach instead of disrupting, due to the fact that the adversary will still attempt to invest time instead of leaving or searching alternatives. In cases when active engagement against enemy communication channels or capabilities is executed, the opponent will comprehend that they are being deceived or caught and might change their tactics and approaches. Depending on the situation this might be desirable to observe the true potential and capacity of the threat actor.

*Deny*. If disrupt and degrade are active engagements against already ongoing attack and its paths, the denial would proactively assess and anticipate attacker's goals and eliminate the attack paths even before they are pursued. This could be linked together with detection and deception techniques, where adversary presence and goals are identified and isolated to limit their movement within the defended network. For such option to be feasible, all of the infrastructure hosts should be centrally manageable through the solution, such as, *Salt* stack. However, if such central management solution is taken over by the adversary then the cyber red team could potentially lose all their assets and the operation becoming fully compromised.

*Defend*. The overarching concept merging all of the individual approaches to provide the unified goal for guarding either the defended information systems or protecting the red team's assets and operational infrastructure.

## 5.3 Cyber Red Team Operational Infrastructure Protection Considerations

The proposed conceptual model for cyber red team's operational infrastructure (Fig. 8) consists of multiple logical network zones, designated by their purpose and usage. This comprehensive infrastructure model is just one of the possible variations, and will be dependant on the operational requirements, resources and time available. The figure represents the infrastructure concept design, functional area purpose and host description, and deployed defensive measure applicability. The key factor in this operational network prototype is to assess which of the described and researched techniques and tools for the detection and deception are applicable and in what ways. This cyber red team operational infrastructure model has been implemented to a certain extent in the technical cyber exercise "Crossed Swords" and is covered in more details in chapter 6.1.

From the previously assessed approaches on Syslog-based anomaly and cyber deception based detection, the following techniques are considered:

1. *passive defence*, supporting defend, disrupt, and deny activities, focuses on host and network device hardening, packet filtering, NIDS, communication channel redundancy, security and encryption, and network segmentation;

2. *Syslog analysis*, supporting detect activity, aims at anomaly and threat detection by analysing aggregated system log and alert information from the network nodes;

3. *Attack surface spoofing*, supporting degrade and deceive activities, is a form of cyber deception for spoofing a large attack surface to confuse and stall the adversary;

4. *honeypot*, supporting deceive, degrade, and detect activities, is a form of active threat detection and can be used for adversary activity tracking, attack surface spoofing (e.g., honeynets) and stalling the movement of attacker within the network;

5. *cyber decoy*, supporting deceive, degrade, and detect activities, is a form of a decoy, such as, the "honeytoken" or "breadcrumb" information deployed on network hosts, to lure the adversary towards a detection system (e.g., honeypot) or to reveal its current position in the cyberspace (e.g., beaconing); and

6. *tarpit*, supporting degrade activity, directly aims at stalling the incoming attack progression by responding slowly to received requests.

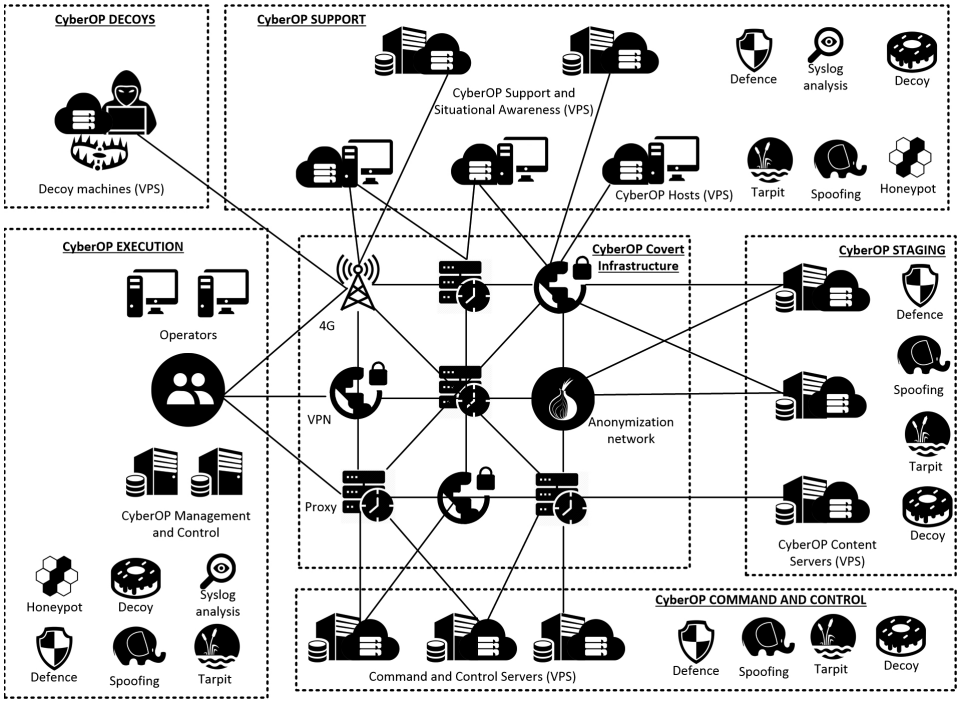



*Figure 8: Cyber red team operational infrastructure concept model and defence measures*

Cyber operation covert infrastructure is a set of interconnected nodes on the Internet by the use of various communication technologies, such as, mobile 4G data connections, VPN tunnels, proxy servers, connection bouncers, or anonymization networks (e.g., TOR, i2p). The purpose of the covert infrastructure is to ensure the maximum possible untraceability to the origin of the cyber red team operators connecting to their deployed assets in the cyberspace. Depending on the operational requirements, at least two network hops should be taken. Relying upon the design of such covert network, the interconnections between the nodes can be changed dynamically whilst maintaining and ensuring the overall connectivity between the two or more interconnected nodes or groups of nodes engaged in the cyber red team operation. Various design approaches can be chosen and implemented to design the overall cyber red team infrastructure and divide it in logical groups basing on their purpose and usage. This might depend on various factors, such as,

if a particular adversary TTTPs are impersonated for the "false-flag" operations, complexity and sensitivity of the performed computer network operations, available resources, and if the cyber red team wants to expose the advanced capacities and capabilities it possesses. For example, a covert network for a high value computer network operation aiming at executing a sophisticated targeted attack against adversary information systems and establishing a Command and Control (CnC) control over the compromised nodes, might have the following architecture requiring applicable defensive measures (Fig. 8):

1. Cyber operation execution area, where the cyber red team is located and the connections to their assets in cyberspace over the covert infrastructure is established. This *home* network would host not only the computer systems from which the operation is originating, but also supporting systems, such as, the situational awareness provision, operation tasking and organisation, collected information cataloguing, indexing and management, operational network supervision, testing, research and development systems, and covert infrastructure automated creation, administration, adaptation and destruction. In this area all of the protection techniques, passive defence, Syslog analysis, spoofing, honeypots, cyber decoys, and tarpits, are applicable to provide maximum possible defence for the cyber operation origin and execution orchestration;

2. Cyber operation support area, where hosts engaged in conducting and supporting the computer network operation are located. All of the nodes are automatically deployed, hosted and afterwards securely destroyed (e.g., overwriting the encrypted file system with random data, such as */dev/urandom*) on a purchased Virtual Private Server (VPS) operator infrastructure. These nodes are supported by the deployed VPS systems, such as, operational infrastructure supervision, Syslog aggregation and analysis, and secured immediate operational information storage. The same protection mechanisms as for the cyber operation execution area are applicable here as it is a vital set of assets on which the success of the operation depends;

3. Cyber operation staging area hosts the exposed VPS nodes engaged in delivering or hosting the attack artefacts. This would include systems, such as, SMTP servers for sending e-mails to the target, web servers for hosting the malware and dive-by exploitation kits, and target network reconnaissance tools. These hosts are used for direct interaction with the adversary for activities, such as, reconnaissance or initial foothold establishment. Taking into account that these systems, due to the nature of their usage, can be identified by the target, a pool of active and stand-by systems are required to be changed once they have been discovered or have fulfilled their intended purpose. Furthermore, this area can include not only the assets deployed by the cyber red team on the public VPS, but also any other public or cloud-based solutions, for example, *Dropbox* for hosting malicious payloads, *Google docs* end for credential harvesting, and *Twitter* feed for CnC command issue. Such public service utilization raises the level of scalability, set-up and destruction, minimizes the expenses and time investment, as well as benefits to raising the level of anonymity. This group of assets due to their high exposure nature and direct engagement with the target have to possess a set of protection mechanisms allowing to estimate asset compromise and tracking of adversarial activities. The techniques, such as, passive defence, spoofing, tarpitting, and decoys, are applicable to ensure a decent level of protection, however, without having a straightforward link, even using covert infrastructure, back to the cyber operation support area. Instead decoys would try to lure the attacker towards the cyber decoy area VPS hosts instead, which then

would report any detected activities back to the cyber support area servers. Varied choice for protection technique tools and implementations should be employed across all protected assets to limit the cyber infrastructure host fingerprinting and identification within the cyberspace based on the defensive and techniques in use;

4. Cyber operation command and control area is the set of VPS based nodes hosting the CnC VPS servers. Upon a successful attack from the cyber operation staging area, the compromised hosts are designed to call back to the intended CnC servers. The connection to the CnC servers is handled via the redirectors in the cover infrastructure, thus limiting the direct exposure of this critical asset. However, the possibility of their detection and targeting by the adversary exists, therefore a set of active and hot stand-by CnC servers is required to transfer the control from one to another in case one has been identified or compromised. Exactly the same protection considerations are applicable to this area as for cyber operation staging hosts due to the high exposure and direct interaction with the target information systems; and

5. Cyber operation decoy area is the landing area to where the stored decoys would attempt to lure the adversary, which is trying to take control over the cyber red team assets and tracing back to the origin of attack. This area would host nodes, such as, honeypots, and sensors for *honeytoken* beacons. The detected interactions with these decoys would be logged and sent to the cyber operation support area servers, such as, Syslog collection and analysis. In this area, systems, such as, the honeypots, decoy destination, and beaconing detectors, are deployed. Since these hosts are designed for actual interaction with the adversary, it would be advisable for them to be designed to look as legitimate as possible to make the attacker believe that some actual cyber red team operational hosts have been reached and thus reveal their position and TTTPs. However, still having some level of hardening implemented to deny full compromise of the hosts. Depending on the cyber operation infrastructure goals, the high- and low-interaction honeypots or a mix of both could be implemented, taking into consideration, that high-interaction systems, in comparison to low-interaction ones, would deliver a more believable experience, but demand higher maintenance and can be potentially fully compromised. Adaptive self-configurable honeypots [84] [93] allowing the adjustment to the incoming attack to provide the highest possible level of interaction and experience for tracking the adversary and permitting to perform threat assessment.

Author acknowledges that Artificial Intelligence (AI) technologies, such as, artificial neural networks and machine learning, should be considered for cyber red team operational network design, implementation and protection. Also, the MITRE ATT&CK and PRE-ATT&CK [57] can be considered for cyber red team OPSEC requirements and when being engaged in false-flag operations with particular threat actor's TTTPs. Furthermore, the protection has to be ensured for cyber red team obtained and controlled assets, such as, DNS names, virtual private servers, bullet-proof hosting services, social network profiles, Open Source Intelligence (OSINT) tools (e.g., Shodan, Maltego, VirusTotal Application Programming Interface (API)), cloud services (e.g., MS Office 365) and payment methods. However, such protection mechanisms would more revolve around using privacy and anonymity ensuring services, use of covert infrastructure for their access, have unique accounts and associated registration data, payments by hard to trace methods (e.g., crypto-currency, )

## 5.4 Chapter Conclusions

This chapter examines the use and applicability of the detection and deception solutions for integration into the cyber red team life-cycle and asset protection. The described novel log-based anomaly detection approaches (Publication IV, Publication V) and existing cyber deception solutions (Publication VI) deliver the required effects to benefit the Cyber Red Teaming (CRT) conducted operations, such as, adversary tracking, threat assessment, technical attribution, red team asset protection, OPSEC improvements, and situational awareness. While conducting the responsive computer network operations the CRT has to bear in mind, that the pursued adversary might engage in the counter-red team activities, thus endangering the not protected CRT assets and jeopardizing the whole responsive operation. Additionally, solutions, compliant with cyber red team operational requirements, can be used for initial response to confirm the defended system security breach and gather initial technical attribution information (Publication IX). The proposed concept model for the cyber red team operational infrastructure (Fig. 8) displays and explains the reasoning and benefits of the detection and deception methods deployed therein. Such model, if adapted and customized according to operational requirements, would provide the necessary situational awareness, increased stealth, and benefit to gaining advantage over the adversary.

# 6 CYBER RED TEAM TRAINING

Majority of the exercises are cyber defence oriented with the Blue Team (BT) being the primary training audience and the Red Team (RT) role-playing the adversary to provide the learning experience for the defenders. However, technical exercises, oriented at advancing the readiness level and experience of a cyber red team, are lacking, limited in scope, not mentioned or described publicly. To enable the development of defensive approaches both approaches – blue and red, have to be exercised, especially if they are dependent on each other both in technical exercises and in real-life operations for protected information system defence. To integrate and explore the proposed concepts of Responsive Cyber Defence (RCD), Computer Network Operation (CNO), Cyber Red Teaming (CRT), red team Techniques, Tools, Tactics and Procedures (TTTPs), detection mechanisms, cyber deceptions, and red team operational infrastructure, a unified environment is required. A technical cyber exercise oriented not only at training single facet of the cyber red team, such as, solving technical challenges, but implementing the presented components to create a cyber operation environment would increase the red team training experience. Additionally, cyber red team interactions and inter-dependencies with other operational entities, such as, conventional kinetic or special operations forces, should also be explored to see how cyber operations fit within the larger picture and not on its own.

This chapter, published in Publication X, unifies all of the listed publications including Publication VII, with their respective contributions being implemented and assessed in the cyber red team oriented technical cyber exercise "Crossed Swords" created and the development being led by the author since year 2014. This exercise and its design considerations are represented as a use case analysis in this thesis. The following sections, in a structured manner, introduce and explore the various aspects of the "Crossed Swords" exercise series as a case study, and emphasize the listed publication concept implementation and their assessment.

## 6.1 Cyber Red Team Exercise Design

Cyber exercise "Crossed Swords" (XS) [62], organized jointly by NATO CCD CoE and CERT.LV, is an annual international technical exercise oriented at training cyber red team with the latest technologies and striving to deliver high realism and training benefits. This exercise, created and with core technical aspects managed by the author, was initially introduced in year 2014 as a red team workshop aiming at training the allied cyber red teams and increasing their readiness for real-life operations. Since its inception, the exercise has grown in complexity and size with the development and management team consisting of over thirty renowned experts in the various areas of technology, red teaming, strategy and leadership, kinetic warfare, international law, and research. The exercise spans across three consecutive days representing a 24-hour fast-paced and intensive operation. More importantly, this exercise has served as a platform for implementing, testing, confirming, and conducting studies in the areas of author's research. The proposed concepts, techniques, tools, and procedures as presented in the listed research papers have been applied and tested within this exercise.

To collect the feedback regarding the XS exercise, a survey was prepared and sent out to all former participants, with the purpose to establish a quick high-level overview. The created survey, aimed at assessing the technological advancement, realism, complexity, learning benefits, and real-time feedback value, was created by the author and sent out to all participants since 2014. Out of 100 participants 33 have provided their feedback, which has been assembled and presented in appendix on page 197. The survey asks to evaluate

the following aspects of the exercise: level of realism of the executed cyber operations at the exercise, diversity of target systems implemented in the game network, technical challenge complexity level, exercise benefits and training outcome value, and the value of provided detected attack feedback. The respondents are requested to grade each of these categories in the scale from zero to five, where the 0 represents the very low realism, diversity, complexity, no value at all, and 5 – very high realism, diversity, complexity, and extremely valuable. The anonymous results gathered from the various alliance member nation participants, mainly representing military organizations, acknowledge the high realism of the exercise (graded 4 out of 5), high diversity of target systems (graded 4 out of 5), high complexity of technical challenges (graded 4 out of 5), with exercise training benefits being extremely valuable (graded 5 out of 5), and provided detected attack feedback being both extremely and very valuable (graded 4 and 5 out of 5). The grading values are a direct representation of the survey charts and are presented here with an illustrative purpose.

## 6.2 Exercise Training and Mission Objectives

The exercise aims to address at least the following principles:

1. Cyber red team assembly and structure. Addresses how the cyber red team can be assembled and structured to accomplish the laid out operational tasks with as less management overhead as possible;

2. Cyber operation execution. Explores how the assembled CRT performs and exposes the ability to accomplish the mission;

3. Information and attack campaign management. Assesses the ways on how the CRT collaborates for information sharing and mission goal objective tasking;

4. Red team coordination. Evaluates the CRT capability to coordinate the particular attacks to avoid tasking collisions, such as, interfering with other sub-team activities (i.e., “friendly-fire”) or doing tasks already performed by other sub-team;

5. Increased level of stealth. Ability of the CRT to apply proper TTTPs to avoid detection at the best level possible;

6. Technical sophistication and related TTTPs. CRT capability to find innovative ways apply applicable TTTPs to accomplish mission objectives;

7. Fast-pace and time pressure. Explores how the CRT is able to perform under constant time pressure and conducts objective prioritization;

8. Situational awareness. Looks at the possibilities to provide the situational awareness to the CRT to improve the learning benefits and outcomes;

9. Cyber red team asset protection and Operational Security (OPSEC). Identifies the applicability of OPSEC requirements to increase the level of stealth, make attribution harder, and protect the CRT deployed assets;

10. Adversary assessment, cyber intelligence and technical attribution. Inspects the CRT ability to gather information allowing adversary threat assessment and technical attribution;

11. Target network infiltration and precision take-down. CRT capability for covert target network infiltration, asset identification and precision take-down;

12. Legal ramifications of the cyber attack. Discovers the possible legal ramifications of the CRT performed activities and overall operation; and

13. Cyber-kinetic interdependencies. Explores the ways on how cyber and kinetic operations can be integrated to assist each other within all applicable domains of operation.

These principles are selected to represent and prototype of a full-spectrum cyber operation and to be used as a basis for the exercise development and execution. Within this scope, the exercise is designed to implement the following cyber red team training objectives (TO):

TO1. Perform defended system compromise assessment, practice evidence gathering and information analysis for technical attribution, identify the origins of malicious activities and take actions to stop them;

TO2. Execute a responsive cyber defence scenario for adversarial information system infiltration;

TO3. Employ stealthy attack approaches, and evaluate applicable TTTPs for fast-paced covert operations;

TO4. Exercise working as a united team in achieving the laid out mission objectives;

TO5. Develop specialized cyber red teaming soft and technical skills needed for operation management, information flow, and target information system takeover; and

TO6. Explore and evaluate the full-spectrum operation's cyber-kinetic interdependencies.

Depending on the individual exercise over-arching scenario, the mission goals for the cyber red team might slightly vary, but the essential mission objectives (MO), which the red team has to accomplish are the following:

MO1. Maintain situational awareness, ensure resiliency of the defended systems, and eradicate adversarial presence; and

MO2. Deter or destroy the incoming larger adversarial kinetic and cyber attack by the cyber-kinetic means.

Within the "Crossed Swords 2019", as for the past iterations, all of the set training objectives were implemented and exercised towards accomplishing the mission objectives.

Majority of specified exercise design principles and training objectives, excluding RT management related objectives, are linked to the listed publications, and they are mapped in a following way:

1. Increased stealth, technical sophistication, target network infiltration and related TTTPs – Publication I, Publication II, Publication III, and Publication VII;

2. Situational awareness, red team asset protection, adversary assessment and technical attribution – Publication IV, Publication V, and Publication VI; and

3. Cyber attack legal ramifications and cyber-kinetic effects – Publication VIII.

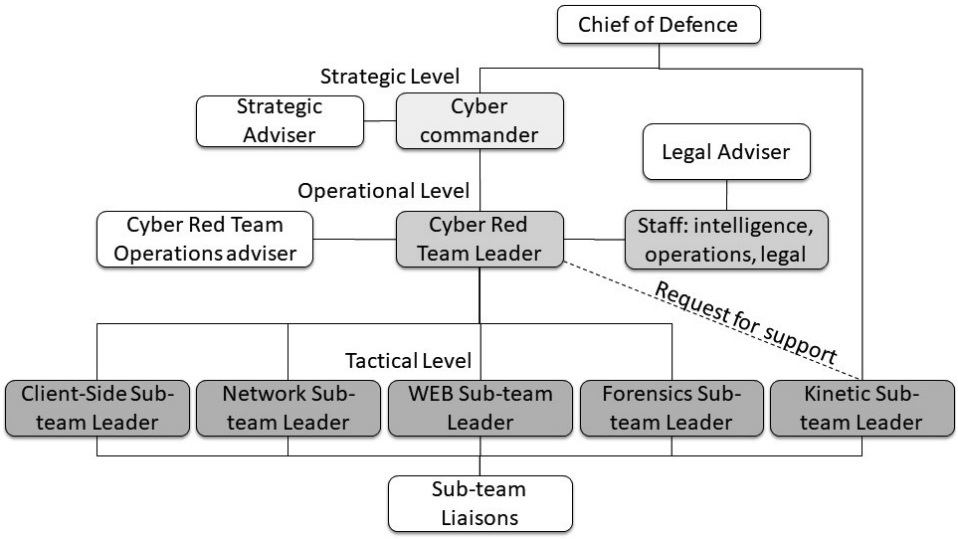



*Figure 9: "Crossed Swords 2019" cyber red team chain-of-command*

## 6.3 Cyber Red Team Structure and Chain-of-Command

The exercise developers, execution managers, and the participants are allocated to various teams and sub-team based on the specifics and activity focus area. The exercise has the following teams based on the area of operations: cyber-kinetic operations team (Red Team – RT), adversary and user simulation team (Blue Team – BT), exercise control and scenario management (White Team – WT), near real-time attack and situational awareness team (Yellow Team – YT), and game network infrastructure development and support team (Green Team – GT). It has to be noted, that the structure and chain-of-command for such cyber-kinetic operations has not been publicly discussed or disclosed by any nation, therefore, this exercise strives to experiment and uncover the organizational model providing as simple as possible chain-of-command and separation of duties.

The designed chain-of-command model for upcoming "Crossed Swords 2019" exercise is depicted in Figure 9, where the grey boxes represent the cyber red team at political, strategic and tactical levels (with respective grey colour shading for every level), and the white boxes indicate the white team presence and assistance to the red team. In the "Crossed Swords" exercise the CRT is divided in to the sub-teams based on the particular expertise in technologies to be targeted (e.g., web applications, Industrial Control System (ICS), network protocols), however, the division can be performed also based on the delivered effect, such as, adversary assessment and reconnaissance, perimeter breach and initial foothold, and particular target objective completion. Both, and possibly more approaches, are applicable on how to structure the CRT based on the operational requirements. This exercise favours the speciality based sub-team creation to allow participant engagement throughout the exercise game-play and not only for explicit phases of the cyber operation.

These teams are subdivided into sub-teams according to the specialization and operational management level:

1. **Red Team**: being the largest team of around fifty experts consists of exercise training audience. The chain-of-command and various sub-teams are the following:

(a) *Cyber commander.* The top-level officer in charge of commanding the cyber operation at a political level. This position is offered to the commanders of the NATO member nation cyber commands. Cyber commander, as part of the training audience, manages and coordinates the cyber operation to reach the set mission goals and coordinates the high-level activities, based on the desired effects, of the sub-teams. Even if the role of the exercise cyber commander might be underestimated it serves as a learning opportunity to the commander on how such cyber-kinetic team would be managed, how to coordinate the activities whilst maintaining the situational awareness. Additionally, this serves an experience for the technical cyber-kinetic team members to always have a clear understanding of the higher commander intentions, provide the situational reports in an understandable manner, and suggest course-of-action options for the mission objective accomplishment;

(b) *Strategic adviser.* Is a member of the exercise development and management team (WT) with the role to provide the advice to the cyber commander, and, if needed, give minor hints to keep the red team activities on the course of designed scenario, as much as it is possible. This option allows exercise developers to explore the nuances and alternative paths for the developed scenario, allowing deviations as long as the end result objectives are met;

(c) Red team leader group. This small group of experts, operating at a strategic level and under direct command of the cyber commander, are responsible for fulfilling assigned operational effects by working with the sub-team leaders and ensuring that objectives, force protection, and intelligence activities are correctly executed and reached. This group consists of three experts: the overall red team leader, OPSEC officer, and an intelligence officer;

(d) *Sub-team leaders.* The main red team consists of five expert-focused sub-teams, at a tactical level, which are represented by their leaders. The purpose of every sub-team, consisting of up to ten experts, is to deliver the intended effects at their responsibility area by performing covert infrastructure maintenance, asset protection, stealthy attacks, infiltration of adversary's information systems, precision take-downs, information extraction and gathering, adversary assessment, and cyber-kinetic engagements. Over the exercise iterations it has been identified, that the most applicable size, ensuring its management and providing required capability, of the sub-team is from six to ten experts. The role and purpose of every sub-team is as follows:

   i. *Client-side attack sub-team.* This team focuses on executing attacks targeted at exploiting end-user (i.e., human vulnerabilities) to get the initial foothold, such as, creating a *spear-phishing* campaign, setting up *watering-hole* or *drive-by* attacks. Once initial breach has succeeded this team performs privilege escalation by exploiting the vulnerabilities on the tar-get client operating system (e.g., MS Windows, GNU/Linux), perform MS Windows domain take-over, conduct lateral movement, control critical processes, such as, unmanned aerial or ground vehicle management graphical user interface, and ensure persistence in the target computer network;

   ii. *Network attack and exploit development sub-team.* The goal for this team is to target exposed computer network services, gain control over them by abusing the misconfiguration, poor implementation, abuse, or developing and exploiting software vulnerabilities. Additionally, this team is

conducting IP network (IPv4 and IPv6) and service mapping, and executing attacks against specialized systems, such as, tactical radio networks, mobile operator base stations, and ICS elements;

iii. *Web-application attack sub-team.* For this team all web-based systems and technologies, such as, web-applications, services, and back-end relational databases, are the target. This team extracts valuable information from the web-applications, such as, user credentials, e-mails, or application source code, as well as breaches the security of an exposed web-application to gain access to the internal network services, and establish persistence;

iv. *Digital battle-field forensics sub-team.* The main effort of this team is to perform data carving and artefact extraction from various sources, such as, hardware devices (e.g., smart phones, portable computers and other electronics), computer memory or hard disk images, by applying digital forensics techniques. This team uniquely serves as the bridge between the cyber and kinetic operational components as it is tasked to perform analysis of forensic evidence extracted either by the cyber sub-teams or brought in from the field by the kinetic team. The goal of the forensic team is to extract valuable evidence or information, such as, intelligence information, sensitive documents, malware command and control server addresses, passwords, or enemy communication channel encryption keys; and

v. *Kinetic forces sub-team.* This team formed from trained military and law-enforcement experts performing various kinetic operations, such as, forced entry, covert access, hardware extraction, target capture or take-down, intelligence collection, surveillance, or kinetic activities on enemy terri-tory. The interaction between the red team provides the one of the key aspects for cyber-kinetic game-play and operation execution. This team is managed and trained by industry experts (e.g., HTCI – High-Tech Crime Institute) and Special Operations Forces (SOF) instructors. The created scenario is designed to have the interdependencies within the red team and anticipates the cyber-kinetic cooperation. For example, the cyber red team might identify a lead, by collecting and assessing the digital evi-dence, to a crucial asset, such as, air-gapped server containing adversary communication encryption keys, which is not directly reachable by cyber means and requests the kinetic force engagement for planning and ex-ecuting the operation to acquire it. In this example, the kinetic force is trained to identify needed hardware, learn its extraction techniques, and preserve digital evidence. Also the kinetic force team might depend on cyber team's support, for example, when executing a forced entry into adversary's data centre protected by a blast-door this can be either can be achieved either by kinetic attack (e.g., use of explosives or physical force) or by cyber means (e.g., targeting the ICS/SCADA system controlling the doors or cutting the power to trigger fail-safe procedures). In this example, depending on the entry method, the kinetic force team might have less or more time to perform the intended activities.

(e) *Sub-team liaisons.* For every mentioned sub-team there is an attached WT liaison responsible for observing and, if required, providing minor hints to the sub-team leader to ensure that the team is not wasting too much time on

some targets, such as, cyber decoys and honeypots, and does not deviate from the intended scenario too much; and

(f) *legal advisers.* With advisers embedded to assist every level – political, strategic, and tactical. The role of these experts is to provide their assessment of the activities form the international and domestic law perspectives. They are not allowed to break the game by denying certain actions, but more giving an insight to the participants on legal implications and consequences of their activities. These experts are heavily used by the training audience to verify and confirm the legality of their planned actions and choose the one compliant or with least consequences. This gives not only the experience and comprehension of ramifications for various taken activities to the technical members of the red team, but also the training experience to the legal advisers as they have to tackle the questions and situation which they might not encounter on day-to-day basis.

2. *Blue Team.* Is a small team of up to four experts experienced in conducting cyber red team activities. This team is under direct control and supervision by the white team and is used to manage the cyber red team's progression within the adversary's computer networks. The main tasks for this team are user and adversary simulation. As a user simulation role-player, they are directly engaged in client-side conducted activities, such as, examining and deciding to open received malicious attachments, visiting the web links, or browsing the in-game web services. With this role their task is to observe, assess and deny or permit the cyber red team's initial foothold, based on the quality and delivery method sophistication level. As an adversary simulation team, their task is to detect the cyber red team's presence, related Indicator of Compromise (IoC), assess the scale, sophistication and red team objectives, and, if permitted, compromise the red team operational infrastructure in a counter-red team operation. By adjusting the level of the resistance of the adversary's information system the cyber red team will have to take this into account when executing their attacks, maintaining OPSEC, and protecting their assets;

3. *White Team.* Is a small group of experts, typically no more than two, responsible for controlling and steering the exercise according to the developed scenario. As mentioned before, deviations from scenario are accepted and some-times encouraged, as long as, the overall focus is not lost, and mission objectives can be reached. It has to be noted, that the exercise does not have the ultimate goal of succeeding in accomplishing the intended scenario and fulfilling entirely the laid mission goals by any means necessary. Depending on the activities pursued by the red team, their course-of-action and time limitations, the mission might be a failure, as it can happen in real life. Within the past iterations of the exercise the red team only once successfully accomplished all of the laid mission objectives, and in other cases completing them partially.

4. *Yellow Team.* This team, composed of experts, focuses various areas of threat and anomaly detection, such as, monitoring, big data analytics, intrusion detection, and situational awareness. The tasks and produced results of this team are described in more detail in chapter 6.5. The most crucial task for this team is to provide the near-real time situational awareness picture to the red team, which represents how the red team operation looks, the detected tools, made mistakes, and identified TTTPs. This fed-back allows the cyber red team to immediately spot the made mistakes

and adjust their operations and tool usage to avoid the detection, therefore not only increasing the level of stealth, but also having a better understanding on the used tools and performed actions; and

5. *Green Team*. Is responsible for tasks, such as, maintaining the cyber range platform, supporting the game network technical requirements, developing the game network hosts and targets, and integrating new technologies either virtual or physical. The game network and introduced technologies are explained in more depth in chapter 6.4.

The described full-spectrum cyber operation execution, tasking, and command structure represents the technical and human environment with its various interdependencies and nuances, where the current work from the listed publications has been implemented and assessed. This includes:

1. Red team – uses the proposed TTTPs, if they are applicable for a particular objective or to deliver the desired effect. Gaining initial access by developing a custom exploit or fuzzing a proprietary network protocol (Publication II) has been successfully executed by the cyber red team against the developed in-game target systems, such as, finding a vulnerable command in a designed network service, allowing stack-based memory buffer overflow exploitation and target system take-over. Established initial foothold on a dual-stack host allows the Command and Control (CnC) channel establishment back to the red team's operational infrastructure attacking hosts, for specific tasks, such as, covert information exfiltration or individual high-value target system control the custom covert channels have been successfully established (Publication I). In addition to these tools, the red team extensively uses also third-party collaboration and post exploitation frameworks (e.g., Cobalt-Strike, Empire, Pupy), which rely on traditional protocols used for CnC communications, such as, HTTP, DNS, and SMB. However, when compared to the proposed *nc64* capabilities, these typical channels are relatively easily detected by the yellow team, and the red team has to make sure, that they are properly configured and adjusted to minimize their detection. For such channels, for example, the compliance with expected protocol payloads have to be ensured not to be easily distinguished from the overall traffic, the beaconing (i.e., call-back) timing and randomization has to be considered to avoid behaviour pattern-based matching and used CnC protocols have to make sense for the target hosts. The ICS oriented zero-day development (Publication III) has been implemented in the game environment, and the red team is faced with identifying weaknesses and exploiting them. All of the described attack vectors in that publication have been successfully also transferred into the exercise and cyber red team has developed the exploits under supervision of an instructor. Due to exercise time restrictions, the instructor gives some leading hints in form of a question allowing the training audience to understand and successfully accomplish the goal within a reasonable time frame. For example, the PROFINET IO attack has been used to control the adversary's data centre bunker doors, and IEC-104 and Martem RTU attacks – to disable the enemy's military base power supply;

2. Adversary simulation team (BT) – it is not mandatory for this team to use any of the proposed cyber-red team oriented TTTPs, as for the adversary simulation team it is not the main purpose of remaining undetected. In some cases, the techniques and tools as described in Publication II have been used against the cyber red team's operational infrastructure, but in most cases the red team OPSEC failures are used

against themselves, such as, unchanged default passwords on the attacking machines, not removed metadata in the delivered infected documents, or leaving unattended back-doors;

3. Legal advisers – according to the scenario, as introduced in the following chapter 6.4, the artificial conflict tackles both international and domestic law. In cases, when international law applies, the work in Publication VIII is used to consult the situation and assess its legal implications;

4. Yellow team – to provide the near real-time feedback and situational awareness picture, as described in detail in chapter 6.5, the solutions for Syslog data aggregation, parsing and correlation (Publication IV, Publication V) are used alongside with the other technologies allowing threat detection and visualization. The cyber red team has to be trained to identify the cyber deceptions and honey-pots (Publication VI), therefore a set of various honey-pots (e.g., low- and high-interaction, and adaptive) and decoys with planted *breadcrumbs* are placed within the adversary's computer networks. This serves not only as a method to represent the situational awareness to the red team but attempts to teach the experts that not all systems have to be targeted and how to identify the decoys. The *Frankenstack* framework (Publication VII), consisting of multiple interconnected tools and solutions, is the core solution developed by the yellow team to provide near real-time situational awareness to the cyber red team; and

5. Green Team – to make the exercise challenging, technically interesting and represent real-life systems as much as possible, the research results (Publication III) are adapted and re-implemented from real cyber operations. The goal for such system integration is to give opportunity to the red team to practice attacks against real systems, which are not available to the participants on day-to-day basis. The introduced technical challenges are covered in more detail in the upcoming chapter 6.4.

## 6.4 Technical Environment, Exercise scenario and Legal Considerations

**Technical environment and scenario.** The "Crossed Swords" (XS) exercise game network is hosted on a cyber range running VMware ESXi hypervisor and it consists of around 200 virtual machines for in-game core networking, simulated Internet, cyber red team segment, and a set of target networks. Not all intended technical game-play elements can be virtualized, therefore the game network is expanded by connecting physical hosts and systems through the cyber range infrastructure. Before creating the overarching geopolitical scenario, the technical scenario is established based on the core development team ideas and intended technical game-play intentions. Due to XS being relatively small, with respect to the game-network scale and training audience size, experimentation and introduction of new, recently prototyped, and orthodox technologies can be afforded making the technical game-play more attractive and as close to the real-life as possible. The network also uses the traditional Information Technology (IT) systems to provide the networking and common workstation operating systems, such as, MS Windows and GNU/Linux, to provide replicate the structure of a regular office and business networks. The following list briefly summarizes some of the technologies introduced in the XS game series, to highlight the technical level:

1. Bunker door – a system, running a set of interconnected Siemens developed S7-1200 Programmable Logic Controller (PLC) based PROFINET IO-devices, is control-

ling the bunker door. The cyber red team has to analyse and reverse-engineer the PROFINET IO RT protocol to inject remotely the commands to open or close the bunker door;

2. Alarm system – a bunker door is protected by the Paradox supplied alarm system and, before the door can be opened, the alarm system has to be targeted remotely by analysing the used bus-protocol, and capturing and decoding the PIN code;
3. CCTV IP camera – attack implies the cyber red team finding and exploiting the flaws in the actual IP-based surveillance camera web interface to gain full control remotely;
4. Distributed power-grid – a system based on IEC-61850 and IEC-60870-5-104 industrial Ethernet protocol series and a Martem produced Remote Terminal Unit (RTU) is used to manage and supervise the distributed power-grid. The red team has to reverse-engineer the IEC-104 protocol and perform remote command injection to control the power supply either by turning it off or on;
5. Anonymization network – red team has to infiltrate the real *i2p* anonymization network to intercept and modify the CnC communication channel running over that network;
6. Unmanned aerial vehicle (UAV) – adversary's Threod manufactured UAVs, flying over or approaching the protected territory, have to targeted to gain control over the provided video stream, taking over the steering, or destroying the UAVs;
7. Unmanned ground vehicle (UGV) – Milrem developed UGVs serve as an adversary-controlled tank force and the cyber red team is tasked to take full control over them by targeting either the used network protocols or the controlling workstation;
8. Maritime navigation – a vessel's steering and tracking system based on the OTH-GOLD (Over-The-Horizon GOLD) and AIS (Automatic Identification System) maritime protocols is targeted by the red team to gain control over the ship and inject naval tracks to confuse the situational awareness;
9. Radio communication network – the network based on Harris constructed military-grade radio stations has to be infiltrated by the red team via extracting the encryption keys for the communications running over the radio carrier;
10. Mobile network base stations – the cyber red team has to infiltrate the LMT (Latvian Mobile Telephone) operator provided base stations connected to the actual mobile network, analyse and parse the intercepted communications to decode the adversary agent's message exchange (SMS) and pinpoint their physical location;
11. Mobile 4G network – red team is tasked to gain access to the Ericsson developed 4G mobile network equipment and execute further attacks against connected nodes;
12. Railroad control station – a system based on Siemens created S7-1200 PLC, running s7comm+ protocol, is used to control the in-game railroad network. The red team is tasked to gain control over the railroad control stations either to stop or derail the train; and
13. Environment monitoring wireless sensors – the red team has to control the Defendec provided Smartdec wireless sensors to track the physical location of an adversary troops.

The various technical challenges implemented across nearly all of the game-net systems, are designed in a way, that no single sub-team of the red team can solve them on its own. To achieve this, cooperation, information exchange, objective tracking, and operation management is emphasized to provide the collaborative training experience and attempting to push the participants out of their comfort zones. The technical scenario, being time limited and fast-paced, cannot be fully solved, therefore the cyber red team has to consider ways and approaches on how to prioritize the technical objectives and manage the focus of force to accomplish the overall mission objectives within the exercise time.

The integration of real-life vulnerabilities and systems, such as, the ones as described in Publication III, deliver the learning perspective to the exercise participants. Examining, developing exploits, and attacking the systems which are widely used for automation and industrial process control are challenging and allow the training audience to comprehend the actual state of security for such industrial components. Furthermore, some participants might have such systems in their organizations, but are not allowed to executed attacks or tests due to them being in a production state. Cyber red team members, with some guidance by the instructors, follow the full weakness identification, vulnerability determination, and exploit development life-cycle as described in the chapter 4.2.3. Such approach has allowed the participants to successfully exploit the industrial control protocols and devices (Publication III).

**Exercise scenario.** The technical scenario, describing the interdependencies, attack vectors, and alternative paths, only covers the part for the actual work to be conducted by the exercise participants. To deliver the context, reasoning, and clear objectives, the overarching scenario is required. This scenario provides the elements, such as, the state-of-the-world background, geo-political situation, intelligence information on what has happened, what is the impact suffered, why the response is being triggered, what are the objectives and rules of engagement. The main geo-political story revolves around a fictitious group of Cyberbian islands, where every island is a country with its technological advancements, political stance, alliances, and intentions. The three island-countries are Berylia, Crimsonia, and Revalia. Berylia being the smallest with a modest military force, part of NATO alliance, and its main economic income originating from the electronics manufacturing. Crimsonia is the largest island with a strong military, rich in natural resources, not part of any alliance, and is expressing some signs of aggression against its neighbouring island-countries. Revalia is a small, self-sustained, and politically neutral country. Within the scenario, the exercise participants assume the role of Berylian rapid response team, which is assembled to address the looming crisis, maintain the resiliency of the national critical information infrastructure, and accomplish the mission goals. Every year, with a new exercise edition, the scenario evolves and the tensions between Berylia and Crimsonia have been escalating, ranging from Crimsonia conducting a series of debilitating cyber attacks against Berylian Critical Information Infrastructure (CII), abuse of a neutral nation infrastructure for operation conduct, placing insiders and double-agents, forming military blockades, up to launching a military invasion of Berylia. The various levels of conflict are designed to explore the technical, cyber-kinetic, and legal game-plays as every particular state opens new opportunities and provides flexibility in conducting the responsive computer network operations. The operational environment for the kinetic force's unit is extremely important, as this restricts, or enables, some types of activities to be exercised.

**Legal considerations.** A part of the exercise scenario consists of legal game-play. Despite the exercise not having legal aspects as the primary objectives, the legal guidance and considerations are incorporated in the form of legal scenario injects aimed to trigger the discussion and legal implication consideration during the situational report meetings.

Legal advisers are assigned to every level of the chain-of-command to assess and consult the exercise participants. The legal aspects of the conducted cyber-kinetic operations and applied TTTPs, within the context of the scenario, tackle at least the following legal considerations as covered in Publication VIII:

1. *Applicable law (Part II)*. Depending on the circumstances ruling at the time, the lawyers are tasked to ascertain which regimes of public international law apply to the cyber operations occurring during the exercise. The cyber attack campaigns of the exercise range from those occurring in peacetime to those endangering national and international security. The storyline generally avoids situations of armed conflict. The exercise scenario attempts to bring all of the five operational domains (i.e., land, sea, air, space, and cyber) into the game-play, extended by espionage, and cyber-kinetic operations. Such activities may be addressed by the international human rights law, diplomatic and consular law, law of the sea, air law, space law, and international telecommunications law;

2. *States entitled to take countermeasures (Rule 24)*. Only state affiliated institutions and organizations, such as, military or intelligence, can conduct responsive activities on the state's behalf as long as the activities they engage in do not constitute an internationally wrongful act. The cyber operations by private entities, such as, business companies or non-governmental organizations (NGO), can never constitute countermeasures in the legal sense. Therefore, the players assume the role of a rapid response team, assembled on the order of Berylian government, which is placed under the supervision of the military command;

3. *Effect of RCD on third parties (Rule 25)*. Due to the fact that RCD has extraterritorial nature and implicates pursuing the adversary, as well as, performing malicious service take-down within the cyberspace, the legal advisers are required to assess the legality of the RCD effects on the third parties. These activities may include operations, such as, third-party Virtual Private Server (VPS) take-over, hacking adversary controlled CnC servers on the Internet, back-dooring or re-weaponizing the malicious code used by the adversary and compromising public web services to plant the targeted exploit-kits. Since such operations are intentional, both from the adversary and defender side, they have an effect on third-party owned systems or against ones residing in a neutral state (i.e., Revalia). For the red team to complete their mission objectives, the RCD activities have to be deemed lawful, the various possible paths have to be explored, their effects evaluated, and necessary precautions taken, if such are reasonably possible;

4. *Limitations on RCD (Rules 23, 26, 72, 73, 113)*. Depending on the legal qualification of the RCD operations, various limitations, such as, concerning necessity, proportionality, imminence and immediacy, are attached to this operation. The legal advisers are tasked to identify any applicable limitations, such as, requirements for the RCD to be necessary and proportionate, and provide these legal implications to the commander or sub-team leaders. The scenario addresses both the cyber and kinetic attacks and activities performed by the adversary against the defended state and its CII. Those include malicious operations, such as, serious cyber attacks, espionage, sabotage, and deployment of malware (e.g., condition-activated logic-bombs) with the goal to debilitate the state's critical services and capabilities. Adversary's kinetic activities include operations, such as, armed drone attacks, navy blockades, placement of a large amount of troops on the borders, and expeditionary force deployment. The game-play is initiated by the series of escalating events leading to

ongoing or imminent threat, which may serve to prove the necessity of the taken responsive operations, which are evaluated by their proportionality and delivered effect. As derived from the scenario, the defending state has to immediately respond to an ongoing or imminent threat to ensure the resiliency and protection of critical assets relevant to the security, functionality and well-being of the state;

5. *Self-defence against an armed attack (Rule 71)*. The scenario is designed in such a manner that the severity offensive action against the victim state amounts to to an armed attack, thus permitting to respond in self-defence with an immediate asymmetric responsive cyber operations against a stronger and advanced adversary. The scenario can be designed more subtle and with less tensions, however, in such case the training audience would struggle proving the necessity to pursue the adversary and infiltrate their network, which would have an impact on the overall game-play and achieving the currently set mission and training objectives;

6. *Geographical limitations of cyber operations (Rule 81)*. The effects of a cyber operations have to be limited to the intended target information systems and geographical locations. This, although not always being possible to limit geographically, is taken into consideration by the red team when executing the cyber operation which may include the activities, such as, placement of drive-by exploit-kits on third-party services. The red team might not be aware of geographical location of a targeted asset in cyberspace, however, when such attack campaign is executed, measures are taken to restrict its spread within the intended target's network address ranges if it is possible;

7. *Means and methods of cyber warfare (Part IV, Chapter 17, Section 5)*. The exercise scenario plays on the various levels of aggression and conflicts without entering the state of war. Despite cyber warfare not being applicable directly it is still to be considered and applicable methods have to be evaluated accordingly, since, within the played-out high-tension scenario, it could unexpectedly escalate to an armed conflict and the state of war;

8. *Precautions (Part IV, Chapter 17, Section 7)*. For the executed cyber operations, the red team is asked to exercise constant care, perform verification of targets, choice of means or methods, choice of targets, evaluate proportionality, and estimate the effects of cyber attack whenever it is reasonably possible and applicable;

9. *Cyber operations in neutral territory (Rule 151)*. The adversary may proxy their cyber attacks or route the kinetic attack, such as, drone flying through neutral state's air space before heading to the intended target. In such cases, the red team's response might have uncertainty and limitations on taken actions in the neutral state's cyberspace. The red team might be tasked with pursuing alternative paths or collecting more attribution evidence, before executing cyber operations within the targets in neutral state's cyberspace; and

10. *False-flag and no-flag operations*. For the red team to protect their identity, assets and intended objectives, a false-flag or no-flag operation would be considered to be executed to imply uncertainty and make attribution harder. From the technical perspective, the cyber red team might adapt the known TTTPs of a chosen threat actor to deceive the adversary. From the legal point of view, it is not clear if such operations are permitted when, for example, impersonating and adversarial profile of a threat actor with high certainty attributable to a third state.

## 6.5 Training Assessment and Real-time Feedback

One of the key aspects of the "Crossed Swords" exercise is to provide the environment, where the cyber red team can experiment, practice applicable TTTPs and observe their effects in near real-time. Such opportunity provides the necessary feedback to the exercise participants for their tool and procedure stealthiness and efficiency, as well as, to the exercise management to evaluate the progress of the red team and the fulfilment of training objectives. To accomplish this, a dedicated framework, called the *Frankenstack* and described in Publication VII, is developed to deliver the required visibility through meaningful visual means and notifications.

The *Frankenstack* development is facilitated and coordinated by NATO CCD CoE since 2016, with a group of an international team of experts. The development team is assembled from technical experts in the field of monitoring, data visualization, threat detection and assessment, and big data analytics. The contributions include NATO CCD CoE partners, such as, Arc4dia, Stamus Networks (Suricata IDS), Greycortex, Cymmetria, Tallinn University of Technology, CERT.LV, and CERT-EE. The *Frankencoding* events[41] have resulted in an ongoing *Frankenstack* development with its source code released publicly[42] on GitHub under the MIT license.

**Goals and design principles.** The operational principles of the developed framework are defined as follows:

1. *increase stealthiness* of the red team executed operation, by providing comprehensive and real-time situational picture;
2. *improve RT coordination* through granular multi-layered view of the detected attacks;
3. *analyse TTTPs* and allow the red team to identify the weaknesses and improve the operational performance;
4. *open-source* tool usage as the building blocks for the *Frankenstack* to lower the needed costs and increase the framework maintainability;
5. *customizable* and interchangeable open-source modules to be replaced for more efficient operations or introduced to expand the existing functionality;
6. *high automation* demands to minimize the latency introduced by human-in-the-loop, by applying solutions, such as, data clustering and pattern mining algorithms;
7. *real-time* provision of the identified threats to allow the adjustment of the red team campaign in a timely manner;
8. *transparency and visibility of actions* delivering the processed information for further analysis at different levels, such as, host-based, or network-based, to the red team members;
9. *adequately detailed* information to be provided to the red team not experienced in system monitoring or data analysis, while still providing enough detail in line with the current game-play progress of the red team; and

[41] Frankencoding. `https://github.com/ccdcoe/Frankencoding`. Accessed: 23/11/2018
[42] Frankenstack. `https://github.com/ccdcoe/frankenstack`. Accessed: 23/11/2018

10. *correlated attack visualisation* to make the detected attacks and threats easily understandable to the exercise participants, exercise control, and observers. This has been implemented by a novel web-based Event Visualization Environment (EVE) and displaying the detected attacks on an interactive network map.

**Integration in the game network.** The solution is easily deployable in the game network and can accept any possible sources of information to be further processed, which can be from at least the following origins:

1. *ERSPAN (Encapsulated Remote Switched Port ANalyser)* traffic mirror collecting all the network data recording, parsing, and deep packet inspection;
2. *NetFlow* from game network routers for traffic statistical analysis and evaluation;
3. *data from the systems*, such as, system performance metrics (e.g., CPU load, HDD utilization, network interface card statistics), and logs (e.g., Syslog, and application textual log-files);
4. *honeypots and cyber decoys* placed in the network to attract and deceive the cyber red team into revealing its TTTPs; and
5. *aggregates the information from all sources* in textual format allowing this to be reduced to a log correlation and analysis problem.

**Assessment.** During the "Crossed Swords 2017" execution the members of the white team performed the assessment of the deployed *Frankenstack* solution for its usefulness and training benefits. The identified findings were addressed and incorporated into the following exercise editions. The conducted expert qualitative interviews and online survey results reflected the following:

1. the deployed tools themselves do not increase the learning perspective, but is up to how red team members perceive and use the tools;
2. the addition of situational awareness solutions to the exercise is welcome and seen as a necessary component;
3. the four large screens in the execution room, showing the yellow team provided information, was preferred and checked approximately every 45 minutes by the majority of the training audience;
4. exercise participants also used the opportunity to access the *Frankenstack* dashboards locally on their computers and dig deeper when attempting new attack vectors;
5. *Alerta* tool, showing the identified attacks as priority categorized alerts, was found most useful by the majority of the trainees;
6. it was acknowledged, that ease of use should be further improved especially when considering the merger of high intensity technical exercise with monitoring tools not known to all participants;
7. majority of the training audience strongly agreed that the provided situational awareness was beneficial to the learning process, was accurate and delivered in acceptable speed;

8. the larger part of the training audience agreed that they learned more regarding how their actions can be detected and tried to be stealthier; and

9. integration of various tools into the *Frankenstack* has to be evaluated carefully to avoid visual distractions and making the output more self-explanatory.

## 6.6 Chapter Conclusions

The technical exercise "Crossed Swords", created and led by the author, integrates all of the research from the listed publications to deliver the innovative and novel training environment for the cyber red team. The flexibility and agility of the exercise permits it to be used as a platform for experimentation, research, and verification of new ideas and concepts. By exploring yet not fully understood concepts of cyber red team assembly and structure, cyber operation management and execution, TTTPs applicability and stealth, near real-time feedback, and intense game-play scenario, permits to identify and verify the functional aspects of cyber operation nuances, which can be integrated into actual cyber operation execution. The exercise strives to provide to the training audience the increased and recognized training experience and benefits by combining the technical, operation management, and legal aspects. The conducted survey (see appendix on page 7.3), with an intention to gather overall feedback and impressions from the exercise participants since year 2014, identifies the learning benefits acknowledged by the training audience. The unique near real-time feedback, delivered to the cyber red team via the *Frankenstack*, permits the team members to assess and increase their skills, employed techniques and tools, adapted tactics and procedures, and promotes deeper understanding of the situational awareness and cyber red team executed responsive computer network operation.

# 7 CONCLUSIONS AND FUTURE WORK

## 7.1 Summary and conclusions

This thesis addresses the lack of public information, understanding, and knowledge on the responsive computer network operation execution within the recently recognized domain of cyber operations. As well as, explores the cyber red teaming applicability to such operation execution, operational asset protection, and required training for the cyber red team. Thesis examines three main closely tied concepts: (1) cyber red team capabilities, novel Techniques, Tools, Tactics and Procedures (TTTPs), and engagements in responsive computer network operations; (2) utilization of anomaly detection and cyber deception novel methods for adversary tracking, situational awareness, technical attribution, and red team asset protection; and (3) cyber red team oriented technical exercise design, structure, execution, legal implications, and novel near real-time feedback provision to the training audience.

Chapters 3 and 4 identify and define the building blocks to determine the definition for specialized cyber red team computer network operations. The author addresses the inconsistency and variety of definitions for the concepts of Responsive Cyber Defence (RCD), Computer Network Operation (CNO), and Cyber Red Teaming (CRT), and defines a new concept of specialized CRT responsive CNO applicable throughout the thesis. As well as, the main characteristics of such operations are estimated. The work in related publications (Publication I, Publication II, Publication III) is assessed from the perspective of novel TTTPs development applicable to the responsive computer network operation execution by the cyber red team. These techniques and tools are compared and evaluated alongside with other popular and commonly used solutions to elaborate the advantages of the proposed ones. The novel TTTPs are mapped against the identified characteristics of specialized CRT responsive CNO, to present their suitability for such operation execution and designated effect delivery.

When considering the specialized cyber red team responsive computer network operations, their extraterritorial nature and offensive capacity has to be acknowledged. For the CRT to identify, track, and pursue the adversary within the cyberspace, the red team would, to a certain extent, follow the cyber attack execution phases (e.g., reconnaissance, initial foothold, command and control). Such phases, from the defender's perspective, are mapped to a cyber kill-chain, allowing to identify and possibly stopping the incoming attack at every stage of the kill-chain. When assessing the popular and accepted cyber kill-chain methods (e.g., Lockheed Martin, Microsoft, Mandiant/FireEye), it can be seen, that there are areas, where a particular method is stronger or weaker. To address these gaps and emphasize the strengths, a new unified model of "cyber attack kill-chain" is proposed by the author. For the CRT to increase the chances of success, the proper techniques for countering the detection at every phase of the cyber attack kill-chain need to be identified. The proposed cyber red team novel TTTPs are mapped against the cyber attack kill-chain to identify their applicability on countering or supporting it at most of its stages.

Chapter 5 explores the effects granted by the novel anomaly detection and current cyber deception solutions and approaches. These methods are explored from the perspective on how they can benefit the cyber red team conducted operations. The novel system log file based anomaly detection and cyber deception techniques are evaluated to grant the CRT the needed scalability, efficiency, ease of automation and management. These approaches fit into the CRT life-cycle and provide the solutions required for passive defence, syslog analysis, attack surface spoofing, honeypots, cyber decoys, and tarpits. From the delivered effect point of view, the proposed techniques help to detect, deceive,

disrupt, degrade, deny, and defend against the adversarial activities. The responsive CNO executed by the CRT would require some form of infrastructure to operate from. This operational infrastructure, depending on the tasks and set objectives, can be used to conduct at least the minimum set of the response related activities, such as, target reconnaissance and vulnerability assessment. The proposed conceptual model for the CRT operational infrastructure includes the following logical network zones – cyber operation execution area (location and initial origin of the cyber red team), support area (location, where attacking hosts are deployed with the attack execution supporting resources), staging area (network hosts used to directly interact with the target information system), command and control area (where the Command and Control (CnC) servers are deployed), and the decoy area (hosting the decoys to where suspicious activity is redirected for further examination). Every particular area has the applicable detection and cyber deception solutions deployed to allow the CRT maintain the visibility over the deployed technical assets, provide means for adversary tracking in case of a possible counter operation, and contribute to increasing the operational security.

Chapter 6 examines the use case of the created and developed technical exercise "Crossed Swords" aimed at training the cyber red team in a close to real-life conditions. This exer-cise integrates all of the listed research work and publications by the author and indirectly serves as the test-bed for novel research conduct and result verification. Besides primar-ily being a technical exercise, also cyber red team management, leadership, information exchange, and legal issues are tackled, to provide the training experience as close to the real cyber operation execution as possible. One of the cornerstones for the exercise is to explore the cyber kinetic interactions by integrating military units (e.g., special opera-tion forces, military police, and army) into the operational game-play. Such kinetic force game-play is intertwined with the cyber operation to find the synergies, where mutual benefits and cooperation can be identified, emphasized and exercised. The training audi-ence, assembled into the CRT and pursuing the assigned mission objectives, practices the existing and novel TTTPs for situational awareness, attribution evidence gathering, target identification, target information system covert infiltration, stealthy activities, and mission objective completion. For this to be successful, a near real-time situational awareness is provided to the participants by the means of novel collection of monitoring, threat detec-tion, and visualization solutions, called the *Frankenstack*. This information gives a timely feedback to the participants to observe the reasons behind the detection of the conducted attacks and allows to identity the needed improvements to increase the level of stealth.

## 7.2 Answering the research questions

Each of the research questions is answered by providing the answers to the respective subquestions.

*[Q1.] What are the specialized cyber red team technical capabilities for responsive computer network operations?*

*[Q1.1.] Which features the techniques, tools, tactics and procedures have to possess to be applicable to the specialized cyber red team responsive computer network operation requirements?* The identified characteristics of the specialized cyber red team computer network operations include at least the following ones: stealthy and innovative, agile and available, targeted and pervasive, rapid and timely, integrated and coordinated, hybrid and effective, and asymmetric. The applicable techniques, tools, tactics and procedures have to support these operational characteristics as much as it is possible. The innovative tools developed based on the suitable techniques have to be at least available on demand, customizable, increase stealth, support asymmetric response, and allow targeted focus of

force. This thesis introduces multiple techniques for initial target network access, command and control channel establishment, and impact delivery, which have resulted in the prototyped tools (*tun64*, *nc64*, *Bbuzz*, *iec104inj*, and other proof-of-concept scripts). The suitability of these tools is assessed by comparing them to other relevant and common tools and solutions, and through conducting practical experiments. The results presented in the listed publications (Publication I, Publication II, Publication III) and summarized in thesis chapter 4.1 on page 40, confirm, that the proposed techniques possess the required features and are applicable to develop suitable tools for specialized cyber red team responsive computer network operation execution, with appropriate tactics and procedures being applied. Furthermore, the proposed techniques and prototyped tools can be also applicable to a wider range of engagements, such as, penetration testing, cyber exercise development and execution, and related cyber operations.

*[Q1.2.] How these proposed techniques, tools, tactics and procedures are suitable to counter the cyber attack kill-chain?* To provide a more comprehensive cyber attack kill-chain, the existing recognized kill-chains are analysed and consolidated to introduce a new unified model attempting to cover all attack stages as thoroughly as possible. The proposed cyber attack kill-chain, introduced and described in chapter 4.2.4 on page 50, presents the following phases: reconnaissance, initial compromise and foothold, command and control, internal reconnaissance, lateral movement, privilege escalation, persistence, asset reconnaissance, and objective completion. For the cyber red team to increase the success of targeting and infiltrating the identified adversarial information systems, suitable TTTPs have to be used to counter possible detection and operation disruption methods at every kill-chain stage. These techniques can be used to support countering all of the cyber attack kill-chain stages either by being directly applied or indirectly supporting other possible methods for countering that specific stage. This is accomplished through granting novel ways for initial compromise and foothold establishment, command and control channel creation, and targeting particular components of the information system. For example, methods for initial foothold can be applicable for gaining further access within lateral movement phase and attempting persistence. Also, established covert command and control channel supports all further activities performed within the target network starting from initial foothold up to objective accomplishment. As well as, targeted vulnerability identification techniques are applicable to support exploitation and privilege escalation attempts.

*[Q2.] How to establish the situational awareness for the cyber red team?*

*[Q2.1.] How system log file analysis and cyber deception approaches are relevant to cyber red team work-flow when executing responsive cyber operations?* The novel system log file-based correlation and anomaly detection tools in combination with cyber deception solutions, as covered in chapter 5 on page 55, are applicable to specialized cyber red team responsive computer network operations. For such techniques to be relevant for the inclusion into cyber red team work-flow, they have to possess at least the following characteristics: readily-available and can be obtained by the red team on demand; easily deployable and manageable to lessen the administration overhead and conserve time; flexible and scalable, allowing to be adapted for operational requirements in various environments; highly automated to further reduce the system administration upkeep and human expert involvement; lightweight, permitting fast deployments and having low computing resource consumption requirements; and can be integrated with other technologies already present on the systems. The introduced novel log-based anomaly detection approaches and tested cyber deception solutions (Publication IV, Publication V, and Publication VI) comply with the presented requirements and are applicable for in-

troduction into cyber red team work-flow.

*[Q2.2.] In what ways such solutions are applicable to situational awareness, red team operational infrastructure protection, and attack technical attribution?* The chapter 5.3 on page 60 introduces and explains the conceptual model for the cyber red team operational infrastructure, where the system log file-based anomaly detection and cyber deception solutions are deployed at various logical network segments. The placement of log analysis and deception tools within those network segments is clarified and their deployment reasoned. The primary goal of such techniques is to aid the cyber red team with providing at least the situational awareness, permit adversary tracking and threat assessment, conduct technical attribution, and protect the cyber red team assets. The proposed techniques and approaches confirm their applicability for cyber red team executed responsive computer network operations.

*[Q3.] How to prepare and train the cyber red team for responsive computer network operation execution?*

*[Q3.1.] What considerations are applicable for training cyber red team as close to the real-life conditions as possible in a technical cyber exercise?* The chapter 6 on page 65 explains and analyses the use case of the cyber red team oriented technical exercise "Crossed Swords" series (Publication X). The identified applicable considerations are at least the following: training environment providing freedom and flexibility to the exercise participants; cyber red team structure design according to the training objectives, mission goals, and cyber operation governance; clear chain-of-command establishment for progress tracking and effective cyber red team management and tasking; sophisticated technical challenges and game-network development, implementing real-life use cases and new technologies (Publication III); appropriate over-arching scenario design to provide the reasoning for the responsive activity immediacy; appropriate technical solution and TTTPs implementation (Publication I, Publication II, Publication III) allowing to increase the level of stealth and execution speed; near real-time cyber attack detection and feedback provision for the training audience (Publication VII); interaction with other operational elements, such as, kinetic forces team, in an integrated game-play to explore the interdependencies, challenges, and benefits; and relevant considerations affecting responsive cyber operation execution, such as, legal ramifications (Publication VIII). These considerations are confirmed to increase the level of realism and boost the learning experience of the training audience.

*[Q3.2.] What means can be used to assess the cyber red team training objective achievement in near real-time?* Cyber red team oriented exercise training and mission objectives are aimed at conducting responsive cyber operations to ensure the security and resiliency of protected systems, stop the malicious activities, infiltrate target information system, maintain stealth, and use applicable TTTPs. As such, cyber red team conducted responsive operation would closely follow cyber attack patterns and employ offensive techniques to accomplish the set objectives. The requirement for cyber red team executed activity feedback and visualization in near real-time implies high or full automation of various threat detection and attack representation solutions. Chapter 6.5 on page 78 describes the novel approach used to develop the collection of monitoring, threat detection, data mining, and visualization solutions, called the *Frankenstack*, for the near real-time feedback provision to the cyber red team (Publication VII). The learning benefits gained by the use of *Frankenstack* have been verified within the multiple iterations of the "Crossed Swords" exercise. As well as, conducting a brief survey among the exercise participants to receive their feedback regarding the exercise and provided situational awareness solutions. Both, the conducted verification and survey, have identified the learning benefits and have been

acknowledged by the training audience to boost their experience and skills. Furthermore, the exercise management team has the ability to track the operation execution progress, assess the training objective accomplishment, and evaluate the performance improvement of the training audience throughout the whole exercise in near real-time.

## 7.3 Future Work

The author acknowledges, that while this thesis significantly advances the collective knowledge on cyber red teaming, it can and should be followed up by subsequent research and development. Author has already started further work on the related topics, such as, applicability and integration of Artificial Intelligence (AI) technologies for the CRT conducted operations, further development of the "Crossed Swords" exercise, cyber red team operational infrastructure automation, and improvement of the *Bbuzz* framework to expand its functionality and apply to network protocol and service vulnerability identification.

To name a few related topics of research to be pursued, such as, cyber command structure for offensive cyber operation execution, cyber weapon development from technical, operational, strategic, and legal perspectives, cyber operation supportive operations, automated exercise game network building by the use of AI technologies, and cyber red team structure design approaches and their strengths.

## References


[1] G. Adkins. Red Teaming the Red Team: Utilizing Cyber Espionage to Combat Terrorism. *Journal of Strategic Security*, 6(5):1–9, 2013.

[2] H. Al-Mohannadi, Q. Mirza, A. Namanya, I. Awan, A. Cullen, and J. Disso. Cyberattack modeling analysis techniques: An overview. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pages 69–76, Aug 2016.

[3] S. D. Applegate. The principle of maneuver in cyber operations. In *2012 4th International Conference on Cyber Conflict (CYCON 2012)*, pages 1–13, June 2012.

[4] M. J. Assante and R. M. Lee. The Industrial Control System Cyber Kill Chain. Whitepaper, SANS Institute, October 2015.

[5] B. Blumbergs. Technical Analysis of Advanced Threat Tactics Targeting Critical Information Infrastructure. *Cyber Security Review*, pages 25–36, 2014.

[6] B. Blumbergs. Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 88–99, Prague, Czech Republic, February 2019. SCITEPRESS.

[7] B. Blumbergs, R. Ottis, and R. Vaarandi. Crossed Swords: A Cyber Red Team Oriented Technical Exercise. In *18th European Conference on Cyber Warfare and Security, ECCWS 2019*, Coimbra, Portugal, July 2019. ACPI. (Accepted paper).

[8] B. Blumbergs, M. Pihelgas, M. Kont, O. Maennel, and R. Vaarandi. Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels. In B. B. Brumley and J. Röning, editors, *Secure IT Systems: 21st Nordic Conference, NordSec 2016*, pages 85–100, Oulu, Finland, November 2016. Springer International Publishing.

[9] B. Blumbergs and R. Vaarandi. Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 707–712, Baltimore, USA, November 2017. IEEE.

[10] A. Boukerche, R. B. Machado, K. R. Jucà, J. ao Bosco M. Sobral, and M. S. Notare. An agent based and biological inspired real-time intrusion detection and security model for computer network operations. *Computer Communications*, 30(13):2649 – 2660, 2007.

[11] D. Bradbury. Offensive defence. *Network Security*, 2013:9 – 12, 07 2013.

[12] P. Brangetto, E. Çalışkan, and H. Rõigas. Cyber Red Teaming - Organisational, technical and legal implications in a military context. Technical report, NATO CCD CoE, 2015.

[13] P. Brangetto, T. Minárik, and J. Stinissen. From Active Cyber Defence to Responsive Cyber Defence: A Way for States to Defend Themselves – Legal Implications. *NATO Legal Gazette. Legal Issues Related to Cyber*, pages 17–28, December 2014.

[14] R. Deiber and R. Rohozinski. Tracking GhostNet:Investigating a Cyber Espionage Network. White paper, Munk Centre for International Studies, University of Toronto and The SecDev Group, March 2009.

[15] L. L. DeLooze, P. McKean, J. R. Mostow, and C. Graig. Simulation for training computer network operations. In *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, pages 329–334, June 2004.

[16] D. E. Denning. Framework and principles for active cyber defense. *Computers & Security*, 40:108 – 113, 2014.

[17] R. S. Dewar. Active Cyber Defense. Technical report, Center for Security Studies (CSS), ETH Zurich, June 2017.

[18] DHS ICS-CERT. Advisory (ICSA-18-142-01) Martem TELEM-GW6/GWM. `https://ics-cert.us-cert.gov/advisories/ICSA-18-142-01`, 2018. Accessed: 04/06/2018.

[19] M. Du, F. Li, G. Zheng, and V. Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1285–1298, New York, NY, USA, 2017. ACM.

[20] EU Council General Secretariat. Non-paper on Attribution of Malicious Cyber Activities in the Context of the Framework for a Joint EU Diplomatic Response to Malicious Cyber Activities. Limited working paper, Council of the European Union, June 2018.

[21] European Central Bank. TIBER-EU Framework: How to implement the European framework for Threat Intelligence-based Ethical Red Teaming. Framework, European Central Bank, May 2018.

[22] European Cybercrime Centre. Cyber Bits: IPv6 vulnerability. Intelligence Notification, Europol Unclassified - Basic Protection Level (BPL), EUROPOL, 2017.

[23] A. Fárar, H. Bahsi, and B. Blumbergs. A Case Study About the Use and Evaluation of Cyber Deceptive Methods Against Highly Targeted Attacks. In *Proceedings of Cyber Incident 2017*, pages 1–7, London, UK, June 2017. IEEE.

[24] FireEye. Advanced Persistent Threat Groups. Who's who of cyber threat actors. `https://www.fireeye.com/current-threats/apt-groups.html`, 2018. Accessed: 12/07/2018.

[25] K. Geers. The challenge of cyber attack deterrence. *Computer Law & Security Review*, 26(3):298 – 303, 2010.

[26] GReAT. Shamoon the Wiper – Copycats at Work. `https://securelist.com/shamoon-the-wiper-copycats-at-work/57854/`, 2012. Accessed: 11/07/2018.

[27] GReAT. BlackEnergy APT Attacks in Ukraine employ spearphishing with Word documents. `https://securelist.com/blackenergy-apt-attacks-in-ukraine-employ-spearphishing-with-word-documents/73440/`, 2016. Accessed: 11/07/2018.

[28] K. E. Heckman, F. J. Stech, B. S. Schmoker, and R. K. Thomas. Denial and deception in cyber defense. *Computer*, 48(4):36–44, Apr 2015.

[29] K. E. Heckman, M. J. Walsh, F. J. Stech, T. A. O'Boyle, S. R. DiCato, and A. F. Herber. Active cyber defense with denial and deception: A cyber-wargame experiment. *Computers & Security*, 37:72 – 77, 2013.

[30] Kaspersky GReAT. "Red October" Diplomatic Cyber Attacks Investigation. https://securelist.com/red-october-diplomatic-cyber-attacks-investigation/36740/, January 2013. Accessed 13/09/2018.

[31] H. Kim, H. Kwon, and K. Kyu Kim. Modified cyber kill chain model for multimedia service environments. *Multimedia Tools and Applications*, 04 2018.

[32] M. Kont, M. Pihelgas, K. Maennel, B. Blumbergs, and T. Lepik. Frankenstack: Toward Real-time Red Team Feedback. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 400–405, Baltimore, USA, November 2017. IEEE.

[33] R. Langner. To Kill a Centrifuge. Technical report, The Langner Group, 2013.

[34] S. P. Leblanc, A. Partington, I. Chapman, and M. Bernier. An overview of cyber attack and computer network operations simulation. In *Proceedings of the 2011 Military Modeling & Simulation Symposium*, MMS '11, pages 92–100, San Diego, CA, USA, 2011. Society for Computer Simulation International.

[35] J. A. Lewis. The Role of Offensive Cyber Operations in NATO's Collective Defence. The tallinn papers, NATO CCD CoE, 2015.

[36] Z. Liu, T. Qin, X. Guan, H. Jiang, and C. Wang. An integrated method for anomaly detection from massive system logs. *IEEE Access*, 6:30602–30611, June 2018.

[37] Lockheed Martin. Gaining the Advantage: Applying Cyber Kill Chain Methodology to Network Defense. White paper, Lockheed Martin corp., 2015.

[38] Lockheed Martin. The Cyber Kill Chain. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html, 2018. Accessed: 22/09/2018.

[39] D. F. Longbine. Red Teaming: Past and Present. Monograph, School of Advanced Military Studies, United States Army Command and General Staff College, May 2008.

[40] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li. Mining invariants from console logs for system problem detection. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, USENIXATC'10, pages 24–24, Berkeley, CA, USA, 2010. USENIX Association.

[41] W. Lu, S. Xu, and X. Yi. Optimizing active cyber defense. In S. K. Das, C. Nita-Rotaru, and M. Kantarcioglu, editors, *Decision and Game Theory for Security*, pages 206–225, Cham, 2013. Springer International Publishing.

[42] E. M Hutchins, M. J Cloppert, and R. M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1, 01 2011.

[43] A. Makanju, A. N. Zincir-Heywood, E. E. Milios, and M. Latzel. Spatio-temporal decomposition, clustering and identification for alert detection in system logs. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 621–628, New York, NY, USA, 2012. ACM.

[44] S. T. Malone. Using an Expanded Cyber Kill Chain Model to Increase Attack Resiliency. Blackhat USA, August 2016. Presentation.

[45] MANDIANT. Practice responding to real-world threats — without the real-world consequences. Data sheet, FireEye Inc., 2018.

[46] S. Mansfield-Devine. Hacking the hackers. *Computer Fraud & Security*, 2009(6):10 – 13, June 2009.

[47] Martem AS. Configuration tool TELEM -GWS User Manual. Chapter 8. Security Considerations. `https://phobos.martem.ee/shr/dsum/Telem-GWS_usermanual.pdf`, 2018. Accessed: 14/07/2018.

[48] B. Mauer, W. Stackpole, and D. Johnson. Developing small team-based cyber security exercises. In *SAM'12 - The 2012 International Conference on Security and Management*, July 2012.

[49] M. Maybaum, B. Blumbergs, E. Çalışkan, I. Leesi, M. Pihelgas, R. Peterson, T. Minárik, and J. Stinissen. Responsive Cyber Defence: Technical and legal analysis. Restricted technical research paper, NATO CCD CoE, 2013.

[50] McAfee Foundstone Professional Services and McAfee Labs. Global Energy Cyberattacks: "Night Dragon". Technical report, McAfee, February 2011.

[51] Microsoft Threat Analytics. What threats does ATA look for? `https://docs.microsoft.com/en-us/advanced-threat-analytics/ata-threats`, 2018. Accessed: 22/09/2018.

[52] T. Minàrik. NATO Recognises Cyberspace as a 'Domain of Operations' at Warsaw Summit. `https://ccdcoe.org/nato-recognises-cyberspace-domain-operations-warsaw-summit.html`, 2016. Accessed: 11/07/2018.

[53] MITRE. CVE-2018-10603. `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-10603`, 2018. Accessed: 04/06/2018.

[54] MITRE. CVE-2018-10605. `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-10605`, 2018. Accessed: 04/06/2018.

[55] MITRE. CVE-2018-10607. `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-10607`, 2018. Accessed: 04/06/2018.

[56] MITRE. CVE-2018-10609. `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-10609`, 2018. Accessed: 04/06/2018.

[57] MITRE corp. MITRE ATT&CK. `https://attack.mitre.org/`. Accessed: 27/10/2018.

[58] MITRE corp. MITRE ATT&CK Groups. `https://attack.mitre.org/groups/`. Accessed: 27/10/2018.

[59] S. Moskal, S. Yang, and M. E Kuhl. Cyber threat assessment via attack scenario simulation using an integrated adversary and network modeling approach. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 15:13–29, 08 2017.

[60] D. Mucci and B. Blumbergs. TED: A Container Based Tool to Perform Security Risk Assessment for ELF Binaries. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 361–369, Prague, Czech Republic, February 2019. SCITEPRESS.

[61] J. Mulvenon. PLA Computer Network Operations: Scenarios, Doctrine, Organizations, and Capability. In R. Kamphausen, D. Lai, and A. Scobell, editors, *Beyond the Strait: PLA Missions Other Than Taiwan*, chapter 8, pages 253–285. US Strategic Studies Institute, The address of the publisher, April 2009.

[62] NATO CCDCOE. Exercise Crossed Swords Practised Cyber-Kinetic Operations in Latvia. `https://ccdcoe.org/exercise-crossed-swords-practised-cyber-kinetic-operations-latvia.html`, 2018. Accessed: 04/06/2018.

[63] NATO Chiefs of Staff. Allied Joint Doctrine AJP-01. Joint publication, NATO/OTAN, February 2018.

[64] A. Nicholson, S. Webber, S. Dyer, T. Patel, and H. Janicke. SCADA Security in the Light of Cyber-Warfare. *Comput. Secur.*, 31(4):418–436, June 2012.

[65] A. J. Oliner, A. Aiken, and J. Stearley. Alert detection in system logs. In *2008 Eighth IEEE International Conference on Data Mining*, pages 959–964, Dec 2008.

[66] N. Provos. A virtual honeypot framework. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 1–1, Berkeley, CA, USA, 2004. USENIX Association.

[67] S. Randhawa, B. Turnbull, J. Yuen, and J. Dean. Mission-centric automated cyber red teaming. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 1:1–1:11, New York, NY, USA, 2018. ACM.

[68] K. A. Repik. Defeating Adversary Network Intelligence Efforts with Active Cyber Defense Techniques. Master's thesis, US Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2008.

[69] T. Rid and B. Buchanan. Attributing cyber attacks. *Journal of Strategic Studies*, 38(1-2):4–37, 2015.

[70] M. Robinson, K. Jones, and H. Janicke. Cyber warfare: Issues and challenges. *Computers & Security*, 49:70 – 94, 2015.

[71] M. Schmitt, L. Vihul, D. Akande, G. Brown, P. Ducheine, T. Gill, W. Heinstchel von Heinegg, G. Hernandez, D. Housen-Couriel, Z. Huang, E. Talbot Jensen, K. Kittichaisaree, A. Kozik, C. Kreiss, T. McCormak, K. Nakatani, G. Rona, P. Spector, S. Watts, and B. Blumbergs. *Tallinn Manual 2.0 on the International Law Applicable to Cyber Operations*. Cambridge University Press, Cambridge, UK, 2017.

[72] M. N. Schmitt. Wired warfare: Computer network attack and jus in bello. *Revue Internationale de la Croix-Rouge/International Review of the Red Cross*, 84(846):365–399, 2002.

[73] J. Stamp, J. Dillinger, W. Young, and J. DePoy. Common Vulnerabilities in Critical Infrastructure Control Systems. Technical report, Sandia National Laboratories, May 2003.

[74] Symantec Security Response. W32.Duqu - The precursor to the next Stuxnet. Technical report, Symantec, November 2011.

[75] The Development, Concepts and Doctrine Centre. Red Teaming Guide, 2nd ed. Joint doctrine publication, UK Ministry of Defense, January 2013.

[76] UK Foreign Office. Foreign Office Minister condemns Russia for NotPetya attacks. https://www.gov.uk/government/news/foreign-office-minister-condemns-russia-for-notpetya-attacks, 2018. Accessed: 04/06/2018.

[77] US-CERT. Alert (TA18-074A). Russian Government Cyber Activity Targeting Energy and Other Critical Infrastructure Sectors. https://www.us-cert.gov/ncas/alerts/TA18-074A, 2018. Accessed: 12/07/2018.

[78] US Joint Chiefs of Staff. Information Operations. Joint publication, US Joint Staff, November 2014.

[79] US Joint Chiefs of Staff. Command Red Team. Joint doctrine note, US Joint Force Development, May 2016.

[80] US Joint Chiefs of Staff. Cyberspace Operations. Joint publication, US Joint Staff, June 2018.

[81] R. Vaarandi, B. Blumbergs, and E. Çalışkan. Simple event correlator - Best practices for creating scalable configurations. In *2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 96–100, Orlando, USA, March 2015. IEEE.

[82] R. Vaarandi, B. Blumbergs, and M. Kont. An Unsupervised Framework for Detecting Anomalous Messages from Syslog Log Files. In *IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, Taipei, Taiwan, April 2018. IEEE.

[83] T. Väisanen, A. Farar, N. Pissanidis, C. Braccini, B. Blumbergs, and E. Diez. Defending mobile devices for high level officials and decision-makers. White paper, NATO CCDCOE, 2015.

[84] G. Wagener, R. State, T. Engel, and A. Dulaunoy. Adaptive and self-configurable honeypots. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, pages 345–352, May 2011.

[85] S. Wen, N. He, and H. Yan. Detecting and predicting apt based on the study of cyber kill chain with hierarchical knowledge reasoning. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, ICNCC 2017, pages 115–119, New York, NY, USA, 2017. ACM.

[86] S. Wen, Y. Rao, and H. Yan. Information protecting against apt based on the study of cyber kill chain with weighted bayesian classification with correction factor. In *Proceedings of the 7th International Conference on Informatics, Environment, Energy and Applications*, IEEA '18, pages 231–235, New York, NY, USA, 2018. ACM.

[87] World Energy Council. World Energy Perspectives 2016. The Road to Resilience - Managing Cyber Risks. https://www.worldenergy.org/wp-content/uploads/2016/09/20160906_Resilience_Cyber_Executive_Summary_WEB.pdf, 2016. Accessed: 11/07/2018.

[88] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles*, SOSP '09, pages 117–132, New York, NY, USA, 2009. ACM.

[89] T. Yadav and A. M. Rao. Technical aspects of cyber kill chain. In J. H. Abawajy, S. Mukherjea, S. M. Thampi, and A. Ruiz-Martínez, editors, *Security in Computing and Communications*, pages 438–452, Cham, 2015. Springer International Publishing.

[90] K. Yamanishi and Y. Maruyama. Dynamic syslog mining for network failure monitoring. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pages 499–508, New York, NY, USA, 2005. ACM.

[91] J. Yuen. Automated Cyber Red Teaming. White paper, Australian Department of Defence, Defence Science and Technology Organisation, Cyber and Electronic Warfare Division, April 2015.

[92] J. Yuen, B. Turnbull, and J. Hernandez. Visual analytics for cyber red teaming. In *2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–8, Oct 2015.

[93] Y. Zhang, C. Di, Z. Han, Y. Li, and S. Li. An adaptive honeypot deployment algorithm based on learning automata. In *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pages 521–527, June 2017.

## ACKNOWLEDGEMENTS

I, the author, would like to express my deepest gratitude to my family and the beloved ones for the invaluable support throughout the studies. To my supervisors, Rain and Risto, for encouraging and enduring the endeavour into the darker side of the cyberspace. To my unofficial supervisor, Olaf, for providing the valuable guidance and sharing his opinions. To Raimo and Baiba for supporting the research and granting academic freedom. To Michael and Liis for offering me the journey into an unknown world of legal aspects of the cyber warfare and operations. And to all my colleagues and co-researchers we have cooperated to conduct sophisticated projects and advanced research.

# ABSTRACT
## Specialized Cyber Red Team Responsive Computer Network Operations

This thesis, based on the collection of published and cited publications, explores the aspects of cyber red team responsive computer network operations, addresses the aspects of an asymmetric response to a stronger adversary, assesses threat detection and cyber deception method applicability, and examines the training requirements for the cyber red team.

In the age of state-affiliated and non-affiliated actors pursuing their agendas within and through the cyberspace, anyone can fall as a potential victim or unwitting accomplice to conducted cyber operations, such as, cyber espionage, cyber sabotage, spread of malware, creation of botnets, conduct of cyber attacks, on-line fraud, identity stealing, information warfare, and information system infiltration. Sophisticated and resourceful threat actors, deciding to maintain persistence in their victim's computer networks, can potentially inflict significant damage, such as, sensitive information exfiltration, data modification, sabotage the integrity of the targeted processes, and presumably inflict kinetic impact. For a targeted victim nation or nation-affiliated entity to respond to such a threat, the proper threat detection and situational awareness techniques need to be deployed in the first place. To support the response against an ongoing or expected cyber attack, the proper capabilities and methods need to be established and created. Cyber red team, equipped with specialized techniques, tools, tactics and procedures, and tasked with engaging the adversary as part of the responsive cyber defence, can provide an unconventional and asymmetric response to the threat. This thesis proposes novel techniques, which are applicable to cyber red team to develop new tools, apply tactics and procedures to conduct the responsive computer network operations. Proposed techniques, prototyped in tools, are tested and compared against other common and popular solutions in that area. The results in the listed publications and as presented in this thesis, show the strengths and advantages of the proposed techniques.

For the response to be possible, some level of initial attribution is required to identify the source of attack, the attack paths taken, and start tracing back to it's origin. To enable this, proper solutions and adequate approaches are needed in order to have situational awareness and possibilities for adversary identification. Within the first stages of the response, before engaging the specialized cyber red team, a combination of passive and active defence solutions have to be employed, such as, network intrusion detection and prevention system, log analysis and anomaly detection, honeypots, honeynets, cyber decoys, and honey-tokens. Specialized cyber red team, engaged in responsive computer network operations, requires equal capabilities and visibility to continue tracking the adversary, while pursuing it through the cyberspace, beyond the protected systems. Such techniques and solutions become part of the cyber red team's OODA (Observe, Orient, Decide and Act) loop to aid in detecting and observing the adversary. This becomes more important, if the adversary decides to engage in counter-cyber red team operation, thus potentially endangering the execution of an ongoing responsive computer network operation. System log-based anomaly detection and cyber deception have established themselves as prominent technologies, advancing beyond regular detection and defence. This thesis assesses and confirms the applicability of such technique integration into the cyber red team work-flow to assist with at least situational awareness, threat detection, adversary assessment, technical attribution, and cyber red team's operational infrastructure and asset protection.

The cyber red team selection, training, and skill-set advancement is a necessary part of such capability establishment and development. This becomes essential, when considering the training designed to prepare the team for real-life engagements and cyber operations. Technical exercises, developed to meet the real-life operations as close as possible, provide one of the options for such training and skill development. This thesis explores a cyber red team oriented technical exercise as a use case to establish the exercise design goals, training and mission objectives, the level of technical challenge sophistication, red team management and chain-of-command challenges, applicability of novel techniques, tools, tactics and procedures, cyber-kinetic interaction, legal implications, and situational awareness feedback. The author's created and managed technical exercise combines all of the explored aspects within this thesis, conducted research, and listed publications. One of the main corner-stones of the described exercise is the near real-time situational awareness and attack detection feedback to the training audience. This feedback is intended to provide immediate visibility on how the conducted attacks appear, are being detected by various solutions, and allow the cyber red team to improve their approaches by mitigating the identified drawbacks and mistakes. With this unique visibility, the red team members are able to verify their tools, apply new ones, experiment with various tactics and procedures, to observe the detection of their actions, and identify in what ways the level of stealth can be improved. Thesis verifies the benefits of the presented exercise, its technical concepts, employed techniques, tools, tactics and procedures, mutual interdependencies with kinetic game-play, and the near real-time feedback to the cyber red team.

The author acknowledges the variety of different penetration testing and attack tools already available publicly on source code sharing services, such as, GitHub and SourceForge. Despite some of those projects being applicable to cyber red teaming, even less of them can be used for the responsive computer operations. This thesis acknowledges and assesses some of the most prominent tools, which can be used to further benefit the cyber red team campaign. However, the emphasis of this thesis is put towards the novel techniques and ideas, which can be applied by the cyber red team to develop specialized tools tailored for computer network operations. It has to be noted, that presented techniques and approaches are applicable not only to cyber red team executed responsive computer operations, but can be applicable to other red teaming activities, penetration testing, and more cyber operations, such as, computer network attack and exploitation. Additionally, the author recognizes the significant work already done by various cyber security companies, threat detection and assessment enterprises, and governmental organizations and initiatives, for identifying and assessing global threat actors and their techniques, tools, tactics and procedures. The knowledge and information made public by these entities and initiatives is used within this thesis to be built on top of it and use it as a catalyst for further advancements. Furthermore, the author is fully aware of the ethical considerations this work may tackle, such as, the proposed technique usage to do harm, inflict damage, and conduct malicious cyber attacks. It has to be accepted, that any development, especially in cyber security, may have dual-use implications to benefit both the attacker and the defender. This, for sure, is part of the endless quest between defence and attack, however, the author hopes that the presented work will allow the defenders to increase the protection of the information systems by introducing the concept of counter-attack as part of the responsive cyber defence.

## KOKKUVÕTE
## Vastutegevusele orienteeritud punase meeskonna küberoperatsioonid

Käesolev doktoritöö põhineb autori publikatsioonidel ja uurib punase meeskonna küberoperatsioone vastusena vastase operatsioonidele, tugevamale vastasele asümmeetrilise vastupanu osutamist, ohu avastamise ja petteoperatsioonide kasutusvõimaluste hindamist küberruumis ning nõudeid punase meeskonna väljaõppele.

Kuna erinevad riiklikud ja mitte-riiklikud aktorid kasutavad oma eesmärkide saavutamiseks küberruumi, võib igaüks langeda mingi küberoperatsiooni ohvriks või tahtmatuks kaasosaliseks, sh puutuda kokku küberspionaaži, kübersabotaaži, pahavara, botivõrkude, küberrünnete, netipettuste, identiteedivarguse, infosõja ja infosüsteemidesse tungimisega. Kõrgeltarenenud ja leidlikud aktorid võivad põhjustada oma ohvrite võrkudes olulist kahju, sh tundliku info vargused, andmete muutmine, protsesside terviklikkuse saboteerimine, millega võib kaasneda füüsiline kahju. Selleks, et sihikule võetud riigil või organisatsioonil oleks võimalus sellistele ohtudele vastu seista, on esmajärjekorras vaja sobivat ohtude avastamise ja situatsiooniteadlikkuse süsteemi. Samuti on vaja käimasolevale või oodatavale küberründele vastamiseks sobivaid võimekusi ja meetodeid. Küberruumis opereeriv punane meeskond, mis on varustatud vastavate tehnikate, tööriistade, taktikate ja protseduuridega ning millele on antud ülesandeks küberohule reageerimine, võib anda vastasele ebakonventsionaalse ja asümmeetrilise vastuse. Käesolev doktoritöö kirjeldab uudseid tehnikaid punase meeskonna küberoperatsioonideks vajalike tööriistade, taktikate ja protseduuride arendamiseks. Välja pakutud tehnikad on rakendatud prototüüptööriistades ning testitud ja võrreldud olemasolevate populaarsete lahendustega. Doktoritöö aluseks olevates artiklites detailselt kirjeldatud tulemused näitavad välja pakutud tehnikate tugevusi ja eeliseid olemasolevate ees.

Selleks, et vastase ründele küberoperatsioonidega vastata, tuleb esmalt rünnet analüüsida, et identifitseerida ründe allikas ja ründega seotud süsteemid. Situatsiooniteadlikkuse omamiseks ja vastase identifitseerimise võimaldamiseks on tarvis sobivaid lahendusi ja lähenemisviise. Enne punase meeskonna kaasamist vastuoperatsioonidesse tuleb rakendada erinevaid passiivseid ja aktiivseid kaitsemeetmeid, nt. sissetungituvastus- ja –tõrjesüsteemid (IDS ja IPS), logianalüüs ja anomaaliatuvastus, meepotid ja –võrgud, peibutised, jne. Vastuoperatsioonidele spetsialiseerunud punane meeskond vajab juurdepääsu ja võimekust vastase jälitamiseks nii oma kui võõrastes süsteemides. Sellised tehnikad ja lahendused on osaks punase meeskonna OODA (Observe, Orient, Decide, Act – märka, orienteeru, otsusta, tegutse) tsüklist, mis aitab vastast avastada ja jälgida. See muutub olulisemaks, kui vastane otsustab omakorda mõjutada punase meeskonna küberoperatsioone ja seab seeläbi ohtu meeskonnale seatud eesmärgi täitmise. Süsteemilogidel põhinev anomaaliatuvastus ja küberpeibutiste kasutamine on osutunud olulisteks tehnoloogiateks, mis laiendavad traditsioonilisi kaitsemeetmeid. Käesolev doktoritöö hindab antud tehnikate integreerimise võimalusi punase meeskonna töövoogudesse, et parandada situatsiooniteadlikkust, ohtude avastamist, vastase (võimekuse) määratlemist, ründe allika kindlakstegemist ning punase meeskonna operatiivtaristu ja varade kaitsmist.

Punase meeskonna liikmete valik, väljaõpe ja oskuste arendamine on selle võime arendamisel tähtis osa, eriti arvestades nende ettevalmistamist päris operatsioonide läbiviimiseks. Realistlikult disainitud tehnilised õppused on üks viis sellise väljaõppe pakkumiseks. See doktoritöö sisaldab juhtumiuuringut ühe punase meeskonna jaoks loodud tehnilise õppuse kohta, sh õppuse eesmärkide, disaini, õppetulemite, tehnilise keerukuse, punase meeskonna juhtimise, uudsete tehnikate kasutuselevõtu, sobivate tööriistade, taktikate

ja protseduuride valiku, küber-kineetilise koostoime, juriidiliste aspektide ja situatsiooniteadlikkuse tagasisidega seonduvate teemade kohta. Neid autori loodud ja juhitud õppuse aspekte ongi käesolevas töös ning selle aluseks olevates publikatsioonides süvitsi uuritud.

Üks antud õppuse nurgakivisid on reaalajalähedane situatsiooniteadlikkuse ja ründetuvastuse tagasiside õppuse sihtgrupile. Tagasiside kaudu saavad õppusel osalejad teada, kuidas nende poolt läbi viidud operatsioon näeb välja vastase ja kõrvaliste osapoolte jaoks, mis omakorda annab neile võimaluse lihvida oma tehnikaid ja protseduure, eksperimenteerida tööriistadega ning vajadusel viia sisse muudatusi oma töövoos. Doktoritöö kinnitab antud õppuse kasulikkust, selle aluseks olevaid tehnilisi kontseptsioone, tehnikaid, tööriistu ja protseduure, kineetilise osa koostoimega kaasnevaid mõjusid küberoperatsioonidele, ning reaalajalähedase tagasiside olulisust punase meeskonna väljaõppes.

Selle töö autor on teadlik paljudest erinevatest läbistustestimise vahenditest ja ründetööriistadest, mis on avalikkusele kättesaadavad koodijagamisteenuste nagu GitHub ja SourceForge kaudu. Teisalt on vaid mõned neist kasutatavad punase meeskonna operatsioonides, ning neist omakorda vaid osa on kasutatavad vastuoperatsioonide kontekstis. Doktoritöö raames vaadeldakse ja hinnatakse olulisemate tööriistade kasutusvõimalusi antud operatsioonidel, kuid töö keskmes on uudsed tehnikad ja ideed, mida saab rakendada punase meeskonna jaoks spetsiaalsete tööriistade loomisel. Antud tehnikad on rakendatavad mitte ainult vastuoperatsioonide, vaid ka nt. läbistustestide, võrgurünnete ja võrguluure kontekstis. Lisaks, autor tunnustab küberturbeettevõtete ja valitsusasutuste tehtud tööd globaalsete ohtude identifitseerimisel ja nende tehnikate, tööriistade, taktikate ja protseduuride hindamisel. Nende avaldatud informatsioon on olnud oluline katalüsaator antud töö raames välja pakutud uudsete lahenduste arendamisel. Lisaks, autor on täiesti teadlik antud tööga kaasnevatest eetilistest küsimustest, kuna valedes kätes võivad loodud lahendused ja teadmised olla ohuks. Kahjuks ei saa küberturbe alases uurimistöös mööda faktist, et paljud leiud on ühtemoodi kasulikud nii kaitsja kui ründaja seisukohast. Autori lootus on, et antud töö annab selles igaveses võidurelvastuses kaitsepoolele eelise, võimaldades juurutada vastuoperatsioonide kontseptsiooni küberkaitses.

# Appendix 1

**Publication I**

B. Blumbergs, M. Pihelgas, M. Kont, O. Maennel, and R. Vaarandi. Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels. In B. B. Brumley and J. Röning, editors, *Secure IT Systems: 21st Nordic Conference*, *NordSec 2016*, pages 85–100, Oulu, Finland, November 2016. Springer International Publishing

# Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels

Bernhards Blumbergs[1(✉)], Mauno Pihelgas[1], Markus Kont[1], Olaf Maennel[2], and Risto Vaarandi[2]

[1] NATO Cooperative Cyber Defense Center of Excellence, Tallinn, Estonia
{bernhards.blumbergs,mauno.pihelgas,markus.kont}@ccdcoe.org
[2] Tallinn University of Technology, Tallinn, Estonia
{olaf.maennel,risto.vaarandi}@ttu.ee

**Abstract.** The Internet Protocol Version 6 (IPv6) transition opens a wide scope for potential attack vectors. IPv6 transition mechanisms could allow the set-up of covert egress communication channels over an IPv4-only or dual-stack network, resulting in full compromise of a target network. Therefore effective tools are required for the execution of security operations for assessment of possible attack vectors related to IPv6 security.

In this paper, we review relevant transition technologies, describe and analyze two newly-developed IPv6 transition mechanism-based proof-of-concept tools for the establishment of covert information exfiltration channels. The analysis of the generated test cases confirms that IPv6 and various evasion techniques pose a difficult task for network security monitoring. While detection of various transition mechanisms is relatively straightforward, other evasion methods prove more challenging.

**Keywords:** IPv6 security · IPv6 transition · Covert channels · Computer network operations · Red teaming · Monitoring and detection

## 1 Introduction

In this work we explore possible uses of IPv6 transition technologies for creation of covert channels over dual-stack and native IPv4 connectivity to exfiltrate information for red teaming [6] purposes. An analysis in Sect. 2 shows that this approach is novel and no implementations of such newly-developed tools have been identified previously.

The main contributions of this paper are:

1. two novel approaches for covert channel creation with IPv6 transition mechanisms;
2. fully self-developed proof-of-concept tools that implement the proposed methods (nc64 and tun64);


B.B. Brumley and J. Röning (Eds.): NordSec 2016, LNCS 10014, pp. 85–100, 2016.
DOI: 10.1007/978-3-319-47560-8_6

3. commonly-used protocol tunneling and developed proof-of-concept tool detection comparison table (Appendix A); and
4. a reproducible virtual lab environment providing detection results using open-source network security monitoring tools.

The Internet is in a period of tremendous growth, currently evolving toward the Internet of Anything (IoA). The more widely-deployed IPv4 standard and IPv6 are incompatible, and they can communicate only via transition mechanisms and technologies [38,44]. This introduces an additional layer of complexity and inherent security concerns for the transition and co-existence period [1]. The adoption of IPv6, and availability per the core backbone of the Internet infrastructure and edge networks, varies [10,12]. IPv6 launch campaigns rapidly increased the number of autonomous systems (AS) announcing IPv6 prefix[1,2]. Nevertheless, connecting to the IPv6 Internet while maintaining scalability and minimal overall complexity often means that edge networks deploy various transition mechanisms [36,44], possibly meaning that local area networks (LANs) will continue to use primary IPv4 for an undefined period.

IPv6 protocol implementations and security solutions are relatively new, already supported by default by modern operating systems, and have not yet reached the level of acceptable quality and maturity [15,44]. The lack of expertise and technological maturity result in IPv6 being considered in most cases as a "back-door" protocol, allowing evasion of security mechanisms [21,23]. This is important particularly when an attack originates from inside the network, as network security devices are commonly configured and placed on the perimeter under the assumption that intruders will always come from outside [39].

In the age of advanced high-profile targeted attacks executed by sophisticated and resourceful adversaries, IPv6 is seen as an additional vector for persistent and covert attacks [29,41]. The length of the transition period cannot be estimated, and it can be assumed that even once the entire Internet is native IPv6, there will still be systems running deprecated IPv6 functionality specifications, or heritage transition mechanisms.

Our research shows that current Network Intrusion Detection System (NIDS) solutions have serious drawbacks for handling IPv6 traffic. Addressing these shortcomings would require redevelopment of the principles how NIDSs reassemble packet streams, and correlation of distinct sessions. The described IPv6 transition-based methods (i.e. nc64 and tun64) use both IP version implementations in the same protocol stack. Attribution of these connections to a covert channel is therefore difficult. By comparison, common protocol tunneling approaches (e.g. SSH, DNS) would be easier to detect by an automated solution or human analyst since their behavior pattern is well known and understood.

In this paper, Sect. 2 reviews background and related work, evasion mechanisms, and covert channels; Sect. 3 describes common protocol tunneling approaches and newly-developed attack tool implementation and design; Sect. 4

[1] IPv6 Enabled Networks, RIPE NCC. http://v6asns.ripe.net/v/6 (Accessed 15/04/2016).
[2] IPv6 CIDR Report. http://www.cidr-report.org/v6/as2.0/ (Accessed 15/04/2016).

describes the attack scenario, simulation environment, and generated test cases; Sect. 5 discusses experiment execution results (presented in Table 1), and additionally gives recommendations for such attack detection and mitigation possibilities; and Sect. 6 offers conclusions and future research directions.

## 2 Background and Related Previous Work

The aim for IPv6 was to evolve and eliminate the technical drawbacks and limitations of the IPv4 standard. However, IPv6 reintroduced almost the same security issues and, moreover, added new security concerns and vulnerabilities [11,19]. Current IPv6 attack tools, such as the *THC-IPv6* [21], *SI6-IPv6*[3], *Topera*[4], and *Chiron*[5] toolkits, include the majority of techniques for abuse of IPv6 vulnerabilities, and can be utilized for network security assessment and IPv6 implementation verification.

Already in 1998, Ptacek and Newsham in their research paper [33] showed that NIDS evasions are possible and pose a serious threat. A proof-of-concept tool, *v00d00N3t*, for establishment of covert channels over ICMPv6 [28] has demonstrated the potential for such approach, though it has not been released publicly. Techniques for evading NIDS based on mobile IPv6 implementations reveal that it is possible to trick NIDS using dynamically-changing communication channels [9]. Also, it could be viable to create a covert channel by hiding information within IPv6 and its extension headers [26]. Network intrusion detection system (NIDS) and firewall evasions based on IPv6 packet fragmentation and extension header chaining attacks, have been acknowledged [1,2,21]. Although current Requests for Comments (RFCs) have updated the processing of IPv6 atomic fragments [17], discarding overlapping fragments [24] and enforcing security requirements for extension headers [20,25], these attacks will remain possible in the years ahead as vendors and developers sometimes fail to follow the RFC requirements or implement their own interpretation of them. General approaches for NIDS evasions have been described and analyzed [3,7,32,43], with the basic principles behind evasions based on the entire TCP/IP protocol stack. Advanced evasion techniques (AETs) involve creating combinations of multiple atomic evasion techniques, potentially allowing evasion of detection by the majority of NIDS solutions [30]. Evasions are possible due to NIDS design, implementation and configuration specifics, and low network latency requirements [15].

Existing approaches and technologies consider native IPv6 network implementation and connectivity, and do not take into account possible methods for network security device evasions and covert channel establishment over IPv6 transition mechanisms, in order to reach the command and control (CnC) servers

[3] SI6 Networks' IPv6 Toolkit. http://www.si6networks.com/tools/ipv6toolkit/ (Accessed 10/11/2015).

[4] Topera IPv6 analysis tool: the other side. http://toperaproject.github.io/topera/ (Accessed 10/11/2015).

[5] Chiron. http://www.secfu.net/tools-scripts/ (Accessed 10/11/2015).

over IPv4 only or dual-stack Internet connectivity. To the best of our knowledge no publicly available tool implements transition technology-based attacks.

## 3 Covert Channel Implementations

### 3.1 Protocol Tunneling

Protocol tunneling and IPv6 tunneling-based transition mechanisms pose a major security risk, as they allow bypassing of improperly-configured or IPv4-only network security devices [11,18,22,23,35]. IPv6 tunnel-based transition mechanisms, as well as general tunneling approaches (e.g. HTTP, SSH, DNS, ICMP, IPsec), can bypass network protection mechanisms. However, IPv6 tunnels add to the heap of possible tunneling mechanisms, leading to unmanaged and insecure IPv6 connections [35]. Moreover, dual-stack hosts and Internet browsers favor IPv6 over IPv4 [10]. Various protocol tunneling approaches can be used to set up a covert channel by encapsulating exfiltrated information in networking protocols. Covert channels based on DNS, HTTP(S), ICMP [5], and SSH [13] protocol tunneling implementations are acknowledged here as the most common approaches for eluding network detection mechanisms, due to both their frequent use and standard network policy, which allows outbound protocols and ports for user requirements and remote network administration needs. For the purposes of the test cases we consider mature and publicly available tools for protocol tunneling establishment.

### 3.2 Proof-of-Concept Nc64 Tool

We have developed a proof-of-concept tool, nc64[6], for the creation of information exfiltration channel over dual-stack networks using sequential IPv4 and IPv6 sessions. The tool's source code is publicly available under MIT license.

Signature-based IDSs reassemble packets and data flows, in order to conduct inspection against a known signature database. This is done on per-session basis (e.g. a TCP session). If the data is fragmented across multiple sessions, then the IDS cannot retrieve the full information to evaluate whether the traffic is malicious. In such scenario NIDS has to be context aware in order to be able to correlate and reconstruct the original stream from multiple sequential ones. This is very challenging due to performance considerations. While any set of networking protocols could be used for a sequential session creation, the security, transition, and immaturity of IPv6 makes it a preferred choice. When considering NIDS separate session correlation possibilities, IP protocol switching would make it harder since destination IPv4 and IPv6 addresses are different. In a dual-stack operating system, IPv4 and IPv6 protocols are implemented side by side, thus adding a layer of separation between the two standards and making it more difficult for IDSs to reassemble data. Additionally, a single host can have multiple global IPv6 addresses, making the correlation to a single host even harder.

[6] nc64 https://github.com/lockout/nc64 (Accessed 12/03/2016).

To exfiltrate data from the source host to a destination CnC server over sequential IPv4 and IPv6 sessions, the data must be split into smaller chunks (i.e. up to IPv6 MTU of 1500B). Alternation between IPv4 and IPv6 per session has to be controlled to minimize the amount of information that is sent over a single IP protocol version in successive sessions (e.g. not allowing three or more sequential IPv4 sessions). This control would avoid partial reassembly and deny successful payload inspection by NIDS.

A CnC server has both IPv4 and IPv6 addresses on which it listens for incoming connections. Once the connection is established, the listener service receives sessions and reassembles data in sequence of reception. This can be hard to accomplish if a stateless transport layer protocol is being used (i.e. UDP) or data chunk size exceeds the maximum path MTU (e.g. causing packet fragmentation).

Our proof-of-concept tool, nc64, is written in Python 3 using standard libraries. It implements the aforementioned principles, and additionally:

1. provides both the listener server and client part in one Python module;
2. accepts user-specified data from a standard input, which provides flexibility and freedom of usage;
3. requires both IPv4 and IPv6 addresses for the destination CnC listener, and can have a list of IPv6 addresses in case the CnC server has multiple IPv6 addresses configured;
4. supports UDP and TCP transport layer protocols, as these are the main ones used in computer networks;
5. enables the destination port to be freely selected to comply with firewall egress rules and match the most common outbound protocol ports (e.g. HTTP(S), DNS), and also allows for setting and randomizing of the source port for UDP-based communications;
6. provides an optional payload Base64 encoding for binary data transmission, and to some degree can be treated as obfuscation if the IDS does not support encoding detection and decoding. It has to be noted that Base64-encoded traffic might reveal the exfiltrated data in the overall traffic since it would stand out, which would also apply when using payload encryption;
7. allows for the setting and randomizing of timing intervals between sequential sessions for an additional layer of covertness and to mitigate possible timing pattern prediction and detection by NIDS;
8. implements control over how many sequential sessions of the same protocol can be tolerated before forcing a switch to the other protocol, ensuring that small files are sent over both IP protocols; and
9. supports additional debugging features, exfiltrated data hash calculation, and transmission statistics.

### 3.3 Proof-of-Concept Tun64 Tool

We have developed a second proof-of-concept tool, tun64[7], which exfiltrates information by abusing tunneling-based IPv6 transition mechanism capabilities

[7] tun64 https://github.com/lockout/tun64 (Accessed 12/03/2016).

over the IPv4-only computer network. The tool's source code is publicly available under MIT license.

Most tunneling-based IPv6 transition mechanisms rely on IPv4 as a link layer by using 6in4 encapsulation [31], whereby an IPv6 packet is encapsulated in IPv4 and the protocol number is set to decimal value 41 (the IANA-assigned payload type number for IPv6). Besides 6in4 encapsulation, we also acknowledge GRE (protocol-47) [14] as an applicable encapsulation mechanism for 6in4-in-GRE double encapsulation. When 6in4 (protocol-41) encapsulation is used, duplex connectivity might not be possible if the network relies on strict NAT. However, for the attack scenario considered in this paper (see Sect. 4.1), a one-way communication channel for information exfiltration to the CnC server is sufficient, making UDP the preferred transport layer protocol [34].

Most of the transition techniques cannot solve transition problems and hence are not appropriate for real-world implementation and widespread deployment [44]. Although tunnel-based transition approaches are considered deprecated by the IETF, some of these technologies continue to be supported by modern operating systems and ISPs. The 6over4 [8], ISATAP [37,40], and 6to4 [27,42] transition mechanisms were selected for implementation in our proof-of-concept tool for tunneling-based information exfiltration. Selection of these mechanisms was based upon the tunnel establishment from the target host or network, their support by either operating systems or local network infrastructure devices [37].

Our proof-of-concept tool, tun64, is written in Python 2.7 using the Scapy library[8]. It implements the aforementioned principles and additionally:

1. provides only the client part, thus relying on standard packet capture tools for reception and reassembly (e.g. tcpdump, Wireshark, tshark);
2. supports TCP, UDP, and SCTP as transport layer protocols;
3. emulates 6over4, 6to4, and ISATAP tunneling by assigning source and destination IPv6 addresses according to the transition protocol specification;
4. enables usage of 6to4 anycast relay routers if the tool is being tested in real Internet conditions, although in our simulated network, 6to4 relay routers or agents are not implemented;
5. allows additional GRE encapsulation to create a 6in4-in-GRE double encapsulated packet, which may allow obfuscation if the NIDS is not performing a full packet decapsulation and analysis;
6. gives an option to freely specify source and destination ports, in order to comply with firewall egress rules; and
7. supports sending a single message instead of files or standard input, a functionality designed with proof-of-concept approach in mind.

## 4 Testing Environment and Test Description

### 4.1 Attack Scenario

Our testing environment and experiments are designed according to the following scenario. The attack target is a small- to medium-sized research organization

[8] Scapy project. http://www.secdev.org/projects/scapy/ (Accessed 10/11/2015).

(up to 100 network nodes). Research organization assumes it is running an IPv4-only network, even though all the network hosts are dual-stack and their ISP just recently started to provide also IPv6 connectivity. Network administrators have implemented IPv4 security policies and only the following most common egress ports and services are allowed through the firewall: DNS (udp/53, tcp/53), HTTP (tcp/80), HTTPS (tcp/443), SSH (tcp/22), and ICMP (echo). All network hosts can establish a direct connection to the Internet without proxies or any other connection handlers. This organization was recently contracted by government to conduct advanced technological research and therefore has sensitive information processed and stored on the network hosts and servers. A red team, assuming the role of reasonably sophisticated attacker with persistent foothold in the research organization's network, is tasked to exfiltrate sensitive information from the target network. The red team has a selection of tools available at its disposal for the establishment of a covert information exfiltration channel, as described in Sect. 3.

### 4.2 Testing Environment

To ensure reproducibility of the testbed, we created several *bash* scripts that leverage the Vagrant[9] environment automation tool. The scripts are publicly available in a GitHub repository[10]. A network map of the virtual testing environment is presented in Fig. 1.

The host and CnC devices were built on 32-bit Kali Linux 2.0, which comes bundled with several tunneling tools. Router1 served as the gateway for the target organization, and Router2 as an ISP node in the simulated Internet (SINET). Both routers were also built as authoritative DNS servers to facilitate usage of the Iodine tool, which was explicitly configured to query them during the tests. Two monitoring machines were built to provide detection capability. The first node was connected with a tap to the network link between the routers and all packets were copied to its monitoring interface. Second node was created to avoid conflicts between monitoring tools, and was therefore not used for capture.

In order to create identical testing conditions, we decided to store a packet capture (PCAP) file for each combination of the exfiltration tool, destination port number, transport layer protocol, and IP version. Additionally, several distinct operation modes were tested for the nc64 (e.g. both plain-text and base64 encoded payload) and tun64 (e.g. ISATAP, 6to4, and 6over4 tunneling mechanism emulation) tools, as these significantly impact the nature of the network traffic. Overall, 126 packet capture files were generated to be used as test cases. In the next phase we used the same monitoring nodes to run a selection of popular detection tools which would analyze these PCAP files, produce connection logs, and possibly generate alerts for suspicious activity.

[9] Vagrant. https://www.vagrantup.com/ (Accessed 07/12/2015).

[10] Automated virtual testing environment. https://github.com/markuskont/exfil-testbench (Accessed 07/12/2015).

We considered a number of open-source monitoring tools that are often used for network security analysis. These include the signature-based NIDSs Snort[11] and Suricata[12], as well as the network traffic analyzers Bro[13] and Moloch[14]. For Suricata, we used the Emerging Threats (ET) ruleset, while for Snort we experimented with rulesets from both SourceFire (SF) and ET signature providers. Furthermore, we consulted with security vendors. In some cases their solutions were based on the same open-source tools, albeit lacking IPv6 support due to small customer demand. Thus, we decided to focus only on evaluating open-source network detection tools. In our tests, the data exfiltrated from the host system comprise the highly sensitive */etc/shadow* file and the *root* user's private *SSH* cryptographic keys. Both of which could be used for gaining unauthorized access to potentially many other systems in the organization.

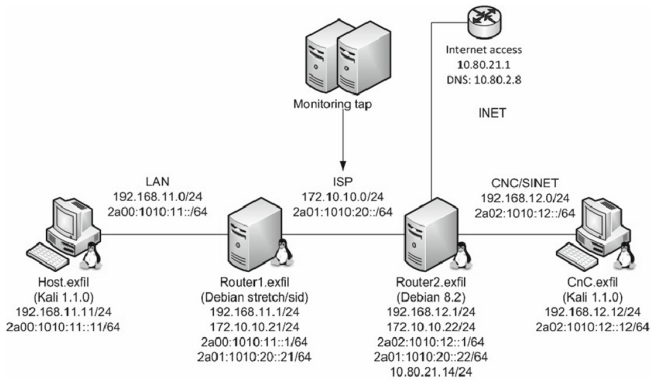



**Fig. 1.** Testing environment network map

## 5 Experiment Execution and Discussion of Results

The results of the experiments are presented in an extensive table (see Table 1 in Appendix A). Each row in the table describes a single attack, while the columns represent a detection tool that was used to attempt its detection. In our results, we distinguished four potential outcomes for a test:

1. a positive match (denoted by letter *Y* and a green cell in the table) was clearly identified as malicious activity with appropriate alerts;
2. a partial or abnormal footprint (*P* and yellow cell) which raised an alert, but the alert did not describe the activity appropriately;

[11] Snort v2.9.8.0. http://manual.snort.org/ (Accessed 07/12/2015).
[12] Suricata v2.1beta4. http://suricata-ids.org/docs/ (Accessed 07/12/2015).
[13] Bro v2.4.1 https://www.bro.org/documentation/index.html (Accessed 07/12/2015).
[14] Moloch v0.12.1. https://github.com/aol/moloch (Accessed 07/12/2015).

3. a potential visible match (*V* and orange cell) from connection logs which requires human analysis or sophisticated anomaly detection for a positive match; and
4. in the worst case, no visible alerts nor connection logs were generated (*N* and red cell).

Firstly, we observed that any exfiltration tool utilizing a specific application layer protocol should adhere to its standard port numbers if the malicious user aims to evade detection. For example, a HTTP tunnel on port 22 triggered an *outbound SSH Scan* alert with the ET ruleset, whereas when port 80 was used, only HTTP connection logs were generated such that we classified the attack as being only *visible*. Note that we marked the *outbound SSH Scan* alert for the HTTP tunnel on port 22 only as a *partial* match because it was incorrectly identified as an outbound SSH connection. Additionally, the same rule was responsible for a partial match against the nc64 technique on port 22. Furthermore, an alert was raised if a SSH header was detected on port 443, or if that port was used to send unencrypted HTTP traffic. Similarly, if abnormal (non-DNS) traffic was identified on UDP port 53, the ET ruleset triggered alerts for either *non-compliant traffic to DNS protocol*, or for being *overly aggressive* (i.e., having too many connections). These signatures were easily bypassed if TCP port 53 was used. However, it has to be noted that most server applications can be bound to any applicable port number (e.g. SSH on tcp/2022, HTTPS console over tcp/8443), and thus can potentially be used to avoid or obscure detection.

The difference between SF and ET rulesets, in their default configurations, is significant. The former seems to focus solely on perimeter intrusions, and hence could not detect any malicious outbound traffic in our tests. Furthermore, the ET ruleset produced slightly different results in Snort and Suricata. Most importantly, the former could clearly identify ICMP Ptunnel as the tool used for traffic exfiltration. Bro does not employ any traditional signatures like Snort or Suricata, but does create logs for all identified connections. As such, it was able to produce log records of all test cases. However, although Bro does not generate alerts, it does have an interesting log file named *weird.log* wherein a record of detected anomalous connections is kept. In fact, during our attacks, several *weird.log* records were generated for non-compliant traffic on port 53. Additionally, Bro's SSH connection parser malfunctioned while processing non-standard traffic, and abnormal logs could be observed in the detection system. Moloch provides no alerts, but is designed as a packet capture, indexing and visualization tool. In the most recent release, at the time of conducting the experiment, Moloch does not support IPv6 due to various limitations when indexing 128-bit IP addresses[15]. Therefore, IPv6-only iterations were unnoticed while IPv4 sessions generated by nc64 in dual-stack configuration were *visible*. The *t6to4* mode in tun64 encapsulates the IPv6 packet as payload making it visible in IPv4

[15] Moloch 0.14.0 2016/05/08 CHANGELOG specifies a notice that "[IPv6] support is experimental, and will change with ES 5.0." https://github.com/aol/moloch/blob/master/CHANGELOG (Accessed 16/08/2016).

indexing system. This was observed only in cases of TCP connections without additional GRE encapsulation.

From the executed test results, detection of malicious activity by NIDS rules was based predominantly on the direction of network traffic, protocol, and destination port. This detection approach is generally favored because it uses resources (e.g. CPU, RAM) efficiently, with an expensive payload analysis attempted only after the preceding match conditions are achieved. In most cases, the nc64 tool avoided being detected, and Table 1 shows which protocol/port combinations can be used to minimize detection by selected NIDS solutions. In comparison with other exfiltration tools, nc64 performed very well on avoiding rule-based detection, and moreover could potentially elude payload inspection. In contrast, the tun64 tool was detected in the majority of cases, since protocol-41 and protocol-47 triggered the rules and generated warning messages by NIDSs. 6to4 tunneling emulation was detected when TCP or 6in4-in-GRE encapsulation was used, suggesting that double encapsulation is considered more suspicious. However, if an organization relies on IPv6 tunneling-based transition mechanisms utilizing 6in4 or GRE encapsulation, such warnings might be silenced or ignored by network-monitoring personnel. In contrast to other tunneling tools the approach taken by tun64 is feasible only if the network conditions comply with the specific operational requirements.

## 6 Conclusions

In this paper, the authors addressed a fundamental problem which could allow to bypass NIDSs by using the IPv6 tunneling-based and dual-stack transition mechanisms in a certain way. The proof-of-concept tools were prototyped to further verify under which circumstances the evasion of major open-source and commercial NIDS and monitoring solutions would be possible. Developed tools, tested alongside with other well known protocol tunneling tools, proved to be able to evade detection and addressed certain shortcomings in the core principles of how modern NIDSs work.

It has to be noted, that any reasonably sophisticated method for exfiltrating data will be hard to detect in real-time by existing NIDSs, especially in situations where the data is split into smaller chunks and the resulting pieces use different connections or protocols (e.g. IPv4 and IPv6). Detecting such activity would require the capability to correlate the detection information in near real-time across different connections/flows. And current NIDS solutions typically lack such capabilities. This is theoretically possible, but would most likely incur a significant performance penalty and an increased number of false positives. There are several possibilities to attempt correlating flows using both IPv4 and IPv6 protocols. If the destination host (i.e. CnC) used in multi-protocol exfiltration has a DNS entry for both A and AAAA records, it would be possible to

perform a reverse lookup to identify that the connections are going to the same domain name using IPv4 and IPv6 protocols simultaneously. This should not happen under normal circumstances, since IPv6 is usually the preferred protocol on dual-stack hosts. Another option would be to rely on source NIC MAC address for aggregating and correlating flows from both IPv4 and IPv6 which are originating from the same network interface. Note, that this requires capturing the traffic from the network segment where the actual source node resides, otherwise source MAC address might get overwritten by network devices in transit. One caveat still remains — distinguishing the flows which are belonging together, especially on busy hosts with many connections. Finally, behavior based detection (e.g. unexpected traffic, malformed packets, specification non-compliance) would provide a way to detect such evasions, at the same time introducing a significant amount of false positives.

It has to be noted that any commercial product which uses an open-source tool for data acquisition is subjected to same limitations of the respective tool. Also, the lack of knowledge regarding IPv6 exploitation methods translate into low customer demand which leads to insufficient IPv6 support in final products. Finally, commercial tools are often too expensive for small and medium sized organizations. Therefore, we did not consider these products in our final evaluation.

Authors believe, that the tendency of use of IPv6 in attack campaigns conducted by sophisticated malicious actors is going to increase; this is also recognized as an increasing trend by the security reports and articles [4,15,16]. Since IPv6 security aspects are being addressed by protocol RFC updates and deprecation of obsolete transition mechanisms, it would be required to focus on these issues at the security solution developer (i.e. vendor) and implementer (i.e. consumer) levels. Adding IPv6 support to the security devices would not solve this problem, since fundamental changes would be required in the way how network traffic is interpreted and parsed, while being able to trace the context of various data streams and perform their correlation. Also, end-users should know how to properly configure, deploy and monitor security solutions in order to gain maximum awareness of the computer network flows under their direct supervision.

Potential future research directions would include advanced insider threat detection, IPv6 protocol stack implementation analysis in the modern operating system kernels and in embedded device micro-kernels.

**Acknowledgements.** This research was conducted with the support of NATO Cooperative Cyber Defense Center of Excellence. The authors would like to acknowledge the valuable contribution of Leo Trukšāns, Walter Willinger, and Merike Käo.

# A Appendix

(See Table 1)

**Table 1.** Protocol tunneling and data exfiltration tool assessment

| Iteration | IP Version | Protocol | Port | Snort SF | Snort ET | Suricata | Bro | Moloch |
|---|---|---|---|---|---|---|---|---|
| http-22 | 4 | TCP | 22 | N | P | P | P | V |
| http-443 | 4 | TCP | 443 | N | Y | Y | V | V |
| http-53 | 4 | TCP | 53 | N | Y | Y | P | V |
| http-80 | 4 | TCP | 80 | N | N | V | V | V |
| Iodine | 4 | UDP | 53 | N | N | Y | P | V |
| nc64-t-22-4-b64 | 4 | TCP | 22 | N | P | P | V | V |
| nc64-t-22-4 | 4 | TCP | 22 | N | P | P | V | V |
| nc64-t-22-64-b64 | 4+6 | TCP | 22 | N | P | P | V | V |
| nc64-t-22-64 | 4+6 | TCP | 22 | N | P | P | V | V |
| nc64-t-22-6-b64 | 6 | TCP | 22 | N | P | P | V | N |
| nc64-t-22-6 | 6 | TCP | 22 | N | P | P | V | N |
| nc64-t-443-4-b64 | 4 | TCP | 443 | N | N | N | V | V |
| nc64-t-443-4 | 4 | TCP | 443 | N | N | N | V | V |
| nc64-t-443-64-b64 | 4+6 | TCP | 443 | N | N | N | V | V |
| nc64-t-443-64 | 4+6 | TCP | 443 | N | N | N | V | V |
| nc64-t-443-6-b64 | 6 | TCP | 443 | N | N | N | V | N |
| nc64-t-443-6 | 6 | TCP | 443 | N | N | N | V | N |
| nc64-t-53-4-b64 | 4 | TCP | 53 | N | N | N | P | V |
| nc64-t-53-4 | 4 | TCP | 53 | N | N | N | P | V |
| nc64-t-53-64-b64 | 4+6 | TCP | 53 | N | N | N | P | V |
| nc64-t-53-64 | 4+6 | TCP | 53 | N | N | N | P | V |
| nc64-t-53-6-b64 | 6 | TCP | 53 | N | N | N | P | N |
| nc64-t-53-6 | 6 | TCP | 53 | N | N | N | P | N |
| nc64-t-80-4-b64 | 4 | TCP | 80 | N | N | N | P | V |
| nc64-t-80-4 | 4 | TCP | 80 | N | N | N | P | V |
| nc64-t-80-64-b64 | 4+6 | TCP | 80 | N | N | N | P | V |
| nc64-t-80-64 | 4+6 | TCP | 80 | N | N | N | P | V |
| nc64-t-80-6-b64 | 6 | TCP | 80 | N | N | N | P | N |
| nc64-t-80-6 | 6 | TCP | 80 | N | N | N | P | N |
| nc64-u-22-4-b64 | 4 | UDP | 22 | N | N | N | V | V |
| nc64-u-22-4 | 4 | UDP | 22 | N | N | N | V | V |
| nc64-u-22-64-b64 | 4+6 | UDP | 22 | N | N | N | V | V |
| nc64-u-22-64 | 4+6 | UDP | 22 | N | N | N | V | V |
| nc64-u-22-6-b64 | 6 | UDP | 22 | N | N | N | V | N |
| nc64-u-22-6 | 6 | UDP | 22 | N | N | N | V | N |
| nc64-u-443-4-b64 | 4 | UDP | 443 | N | N | N | V | V |
| nc64-u-443-4 | 4 | UDP | 443 | N | N | N | V | V |
| nc64-u-443-64-b64 | 4+6 | UDP | 443 | N | N | N | V | V |
| nc64-u-443-64 | 4+6 | UDP | 443 | N | N | N | V | V |
| nc64-u-443-6-b64 | 6 | UDP | 443 | N | N | N | V | N |

(*continued*)

**Table 1.** (*continued*)

| Iteration | IP Version | Protocol | Port | Snort SF | Snort ET | Suricata | Bro | Moloch |
|---|---|---|---|---|---|---|---|---|
| nc64-u-443-6 | 6 | UDP | 443 | N | N | N | V | N |
| nc64-u-53-4-b64 | 4 | UDP | 53 | N | Y | Y | P | V |
| nc64-u-53-4 | 4 | UDP | 53 | N | Y | Y | P | V |
| nc64-u-53-64-b64 | 4+6 | UDP | 53 | N | Y | Y | P | V |
| nc64-u-53-64 | 4+6 | UDP | 53 | N | Y | Y | P | V |
| nc64-u-53-6-b64 | 6 | UDP | 53 | N | Y | Y | P | N |
| nc64-u-53-6 | 6 | UDP | 53 | N | Y | Y | P | N |
| nc64-u-80-4-b64 | 4 | UDP | 80 | N | N | N | V | V |
| nc64-u-80-4 | 4 | UDP | 80 | N | N | N | V | V |
| nc64-u-80-64-b64 | 4+6 | UDP | 80 | N | N | N | V | V |
| nc64-u-80-64 | 4+6 | UDP | 80 | N | N | N | V | V |
| nc64-u-80-6-b64 | 6 | UDP | 80 | N | N | N | V | N |
| nc64-u-80-6 | 6 | UDP | 80 | N | N | N | V | N |
| netcat-t-22-4 | 4 | TCP | 22 | N | N | N | V | V |
| netcat-t-22-6 | 6 | TCP | 22 | N | N | N | V | N |
| netcat-t-443-4 | 4 | TCP | 443 | N | N | N | V | V |
| netcat-t-443-6 | 6 | TCP | 443 | N | N | N | V | N |
| netcat-t-53-4 | 4 | TCP | 53 | N | N | N | P | V |
| netcat-t-53-6 | 6 | TCP | 53 | N | N | N | P | N |
| netcat-t-80-4 | 4 | TCP | 80 | N | N | N | V | V |
| netcat-t-80-6 | 6 | TCP | 80 | N | N | N | V | N |
| netcat-u-22-4 | 4 | UDP | 22 | N | N | N | V | V |
| netcat-u-22-6 | 6 | UDP | 22 | N | N | N | V | N |
| netcat-u-443-4 | 4 | UDP | 443 | N | N | N | V | V |
| netcat-u-443-6 | 6 | UDP | 443 | N | N | N | V | N |
| netcat-u-53-4 | 4 | UDP | 53 | N | Y | Y | P | V |
| netcat-u-53-6 | 6 | UDP | 53 | N | Y | Y | P | N |
| netcat-u-80-4 | 4 | UDP | 80 | N | N | N | V | V |
| netcat-u-80-6 | 6 | UDP | 80 | N | N | N | V | N |
| ptunnel | 4 | ICMP | | N | Y | N | V | V |
| ssh-4-22 | 4 | TCP | 22 | N | N | V | V | V |
| ssh-4-443 | 4 | TCP | 443 | N | Y | Y | V | V |
| ssh-4-53 | 4 | TCP | 53 | N | N | V | V | V |
| ssh-4-80 | 4 | TCP | 80 | N | N | V | P | V |
| ssh-6-22 | 6 | TCP | 22 | N | N | V | P | N |
| ssh-6-443 | 6 | TCP | 443 | N | Y | Y | P | N |
| ssh-6-53 | 6 | TCP | 53 | N | N | V | P | N |
| ssh-6-80 | 6 | TCP | 80 | N | N | V | P | N |
| tun64-t-22-isatap | 4 | TCP | 22 | N | Y | Y | P | N |
| tun64-t-22-t6over4 | 4 | TCP | 22 | N | Y | Y | P | N |
| tun64-t-22-t6to4 | 4 | TCP | 22 | N | Y | Y | P | V |

(*continued*)

**Table 1.** (*continued*)

| Iteration | IP Version | Protocol | Port | Snort SF | Snort ET | Suricata | Bro | Moloch |
|---|---|---|---|---|---|---|---|---|
| tun64-t-443-isatap | 4 | TCP | 443 | N | Y | Y | P | N |
| tun64-t-443-t6over4 | 4 | TCP | 443 | N | Y | Y | P | N |
| tun64-t-443-t6to4 | 4 | TCP | 443 | N | Y | Y | P | V |
| tun64-t-53-isatap | 4 | TCP | 53 | N | Y | Y | P | N |
| tun64-t-53-t6over4 | 4 | TCP | 53 | N | Y | Y | P | N |
| tun64-t-53-t6to4 | 4 | TCP | 53 | N | Y | Y | P | V |
| tun64-t-80-isatap | 4 | TCP | 80 | N | Y | Y | P | N |
| tun64-t-80-t6over4 | 4 | TCP | 80 | N | Y | Y | P | N |
| tun64-t-80-t6to4 | 4 | TCP | 80 | N | Y | Y | P | V |
| tun64-u-22-isatap | 4 | UDP | 22 | N | Y | Y | P | N |
| tun64-u-22-t6over4 | 4 | UDP | 22 | N | Y | Y | P | N |
| tun64-u-22-t6to4 | 4 | UDP | 22 | N | Y | Y | P | N |
| tun64-u-443-isatap | 4 | UDP | 443 | N | Y | Y | P | N |
| tun64-u-443-t6over4 | 4 | UDP | 443 | N | Y | Y | P | N |
| tun64-u-443-t6to4 | 4 | UDP | 443 | N | Y | Y | P | N |
| tun64-u-53-isatap | 4 | UDP | 53 | N | Y | Y | P | N |
| tun64-u-53-t6over4 | 4 | UDP | 53 | N | Y | Y | P | N |
| tun64-u-53-t6to4 | 4 | UDP | 53 | N | Y | Y | P | N |
| tun64-u-80-isatap | 4 | UDP | 80 | N | Y | Y | P | N |
| tun64-u-80-t6over4 | 4 | UDP | 80 | N | Y | Y | P | N |
| tun64-u-80-t6to4 | 4 | UDP | 80 | N | Y | Y | P | N |
| tun64-t-22-isatap-gre | 4 | TCP | 22 | N | Y | Y | P | N |
| tun64-t-22-t6over4-gre | 4 | TCP | 22 | N | Y | Y | P | N |
| tun64-t-22-t6to4-gre | 4 | TCP | 22 | N | Y | Y | P | V |
| tun64-t-443-isatap-gre | 4 | TCP | 443 | N | Y | Y | P | N |
| tun64-t-443-t6over4-gre | 4 | TCP | 443 | N | Y | Y | P | N |
| tun64-t-443-t6to4-gre | 4 | TCP | 443 | N | Y | Y | P | V |
| tun64-t-53-isatap-gre | 4 | TCP | 53 | N | Y | Y | P | N |
| tun64-t-53-t6over4-gre | 4 | TCP | 53 | N | Y | Y | P | N |
| tun64-t-53-t6to4-gre | 4 | TCP | 53 | N | Y | Y | P | V |
| tun64-t-80-isatap-gre | 4 | TCP | 80 | N | Y | Y | P | N |
| tun64-t-80-t6over4-gre | 4 | TCP | 80 | N | Y | Y | P | N |
| tun64-t-80-t6to4-gre | 4 | TCP | 80 | N | Y | Y | P | V |
| tun64-u-22-isatap-gre | 4 | UDP | 22 | N | Y | Y | P | N |
| tun64-u-22-t6over4-gre | 4 | UDP | 22 | N | Y | Y | P | N |
| tun64-u-22-t6to4-gre | 4 | UDP | 22 | N | Y | Y | P | V |
| tun64-u-443-isatap-gre | 4 | UDP | 443 | N | Y | Y | P | N |
| tun64-u-443-t6over4-gre | 4 | UDP | 443 | N | Y | Y | P | N |
| tun64-u-443-t6to4-gre | 4 | UDP | 443 | N | Y | Y | P | V |
| tun64-u-53-isatap-gre | 4 | UDP | 53 | N | Y | Y | P | N |
| tun64-u-53-t6over4-gre | 4 | UDP | 53 | N | Y | Y | P | N |
| tun64-u-53-t6to4-gre | 4 | UDP | 53 | N | Y | Y | P | V |
| tun64-u-80-isatap-gre | 4 | UDP | 80 | N | Y | Y | P | N |
| tun64-u-80-t6over4-gre | 4 | UDP | 80 | N | Y | Y | P | N |
| tun64-u-80-t6to4-gre | 4 | UDP | 80 | N | Y | Y | P | V |

## References


1. Atlasis, A.: Attacking IPv6 implementation using fragmentation. Technical report, Centre for Strategic Cyberspace + Security Science (2011)
2. Atlasis, A.: Security impacts of abusing IPv6 extension headers. Technical report, Centre for Strategic Cyberspace + Security Science (2012)
3. Atlasis, A., Rey, E.: Evasion of high-end IPS devices in the age of IPv6. Technical report, secfu.net (2014)
4. Blumbergs, B.: Technical analysis of advanced threat tactics targeting critical information infrastructure. Cyber Security Review, pp. 25–36 (2014)
5. Blunden, B.: Covert Channels. In: Blunden, B. (ed.) The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System, 2nd edn. Jones and Bartlett Learning, Burlington (2013)
6. Brangetto, P., Çalişkan, E., Rõigas, H.: Cyber Red Teaming - Organisational, technical and legal implications in a military context. NATO CCD CoE (2015)
7. Bukač, V.: IDS system evasion techniques. Master's thesis, Masarykova Univerzita Fakulta Informatiky (2010)
8. Carpenter, B., Jung, C.: Transmission of IPv6 over IPv4 Domains without Explicit Tunnels. RFC 2529, IETF Secretariat, standards Track, March 1999
9. Colajanni, M., Zotto, L.D., Marchetti, M., Messori, M.: Defeating NIDS evasion in Mobile IPv6 networks. In: IEEE (2011)
10. Colitti, L., Gunderson, S.H., Kline, E., Refice, T.: Evaluating IPv6 adoption in the internet. In: Krishnamurthy, A., Plattner, B. (eds.) PAM 2010. LNCS, vol. 6032, pp. 141–150. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12334-4_15
11. Convery, S., Miller, D.: IPv6 and IPv4 Threat Comparison and Best-Practice Evaluation. White paper, Cisco Systems, March 2004
12. Czyz, J., Allman, M., Zhang, J., Iekel-Johnson, S., Osterweil, E., Bailey, M.: Measuring IPv6 adoption. In: ACM SIGCOMM14 (2014)
13. Ellens, W., Żuraniewski, P., Sperotto, A., Schotanus, H., Mandjes, M., Meeuwissen, E.: Flow-based detection of DNS tunnels. In: Doyen, G., Waldburger, M., Čeleda, P., Sperotto, A., Stiller, B. (eds.) AIMS 2013. LNCS, vol. 7943, pp. 124–135. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38998-6_16
14. Farinacci, D., Li, T., Hanks, S., Meyer, D., Traina, P.: Generic Routing Encapsulation (GRE). RFC 2784, IETF Secretariat, March 2000. (standards Track. Supplemented with RFC2890)
15. Fortinet: Biting the Bullet: A Practical Guide for Beginning the Migration to IPv6. white paper, Fortinet Inc. (2011)
16. Data SecurityLabs, G.: Uroburos: Highly complex espionage software with Russian roots. Technical report, G Data Software AG, February 2014
17. Gont, F.: Processing of IPv6 "Atomic" Fragments. RFC 6946, May 2013
18. Gont, F.: Security Implications of IPv6 on IPv4 Networks. RFC 7123, February 2014
19. Gont, F., Chown, T.: Network Reconnaissance in IPv6 Networks. Technical report, IETF Secretariat, February 2015. (internet Draft)
20. Gont, F., Liu, W., Bonica, R.: Transmission and processing of IPv6 options. Technical report, IETF Secretariat, March 2015. (best Current Practice)
21. Gont, F., Heuse, M.: Security assessments of IPv6 networks and firewalls. IPv6 Congress 2013 (2013). (presentation)
22. The Government of HKSAR: IPV6 security. Technical report, The Government of the Hong Kong Special Administrative Region, May 2011

23. Hogg, S., Vyncke, E.: IPv6 Security. Cisco Press, Indianapolis (2009)
24. Krishnan, S.: Handling of Overlapping IPv6 Fragments. RFC 5722, IETF Secretariat, December 2009. (standards Track. Updates RFC 2460)
25. Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., Bhatia, M.: A uniform format for IPv6 extension headers. Technical report
26. Lucena, N.B., Lewandowski, G., Chapin, S.J.: Covert channels in IPv6. In: Danezis, G., Martin, D. (eds.) PET 2005. LNCS, vol. 3856, pp. 147–166. Springer, Heidelberg (2006). doi:10.1007/11767831_10
27. Moore, K.: Connection of IPv6 Domains via IPv4 Clouds. RFC 3056, IETF Secretariat, February 2001. (standards Track)
28. Murphy, R.: IPv6 / ICMPv6 Covert Channels. DEF CON 2014 (2014). (presentation)
29. National Cybersecurity and Communications Integration Center: ICS-CERT Monitor. Technical report, US Dep. of Homeland Security, December 2013
30. Niemi, O.P., Levomki, A., Manner, J.: Dismantling intrusion prevention systems. In: ACM SIGCOMM 2012, August 2012
31. Nordmark, E., Gilligan, R.: Basic transition mechanisms for IPv6 hosts and routers. RFC 4213, IETF Secretariat, October 2005. (standards Track)
32. Pastrana, S., Montero-Castillo, J., Orfila, A.: Evading IDSs and firewalls as fundamental sources of information in SIEMS. In: Pastrana, S., Montero-Castillo, J., Orfila, A. (eds.) Advances in Security Information Management: Perceptions and Outcomes. Nova Science Publishers, New York (2013)
33. Ptacek, T.H., Newsham, T.N.: Insertion, evasion, and denial of service: eluding network intrusion detection. Technica report, DTIC Document, January 1998
34. Sarrar, N., Maier, G., Ager, B., Sommer, R., Uhlig, S.: Investigating IPv6 traffic: What happened at the world IPv6 day? In: Taft, N., Ricciato, F. (eds.) PAM 2012. LNCS, vol. 7192, pp. 11–20. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28537-0_2
35. Degen, S., et al.: Testing the security of IPv6 implementations. Technical report, Ministryof Economic Affairs of the Netherlands, March 2014
36. Skoberne, N., Maennel, O., Phillips, I., Bush, R., Zorz, J., Ciglaric, M.: Ipv4 address sharing mechanism classification and tradeoff analysis. IEEE/ACM Trans. Netw. **22**(2), 391–404 (2014)
37. Steffann, S., van Beijnum, I., van Rein, R.: A comparison of IPv6-over-IPv4 tunnel mechanisms. RFC 7059, IETF Secretariat, November 2013. (informational)
38. Tadayoni, R., Henten, A.: Transition from IPv4 to IPv6. In: 23rd European Regional Conference of the International Telecommunication Society, July 2012
39. Taib, A.H.M., Budiarto, R.: Evaluating IPv6 Adoption in the Internet. In: 5th Student Conference on Research and Development. IEEE, December 2007
40. Templin, F., Gleeson, T., Thaler, D.: Intra-site automatic tunnel addressing protocol (ISATAP). RFC 5214, IETF Secretariat, March 2008. (informational)
41. TrendLabs: targeted attack trends 2014 Report. Technical report, TrendMicro (2015)
42. Troan, O., Carpenter, B.: Deprecating the Anycast Prefix for 6to4 Relay Routers. RFC 7526, IETF Secretariat, May 2015. (best Current Practice)
43. Vidal, J.M., Castro, J.D.M., Orozco, A.L.S., Villalba, L.J.G.: Evolutions of evasion techniques aigainst network intrusion detection systems. In: ICIT 2013, The 6th International Conference on Information Technology, May 2013
44. Wu, P., Cui, Y., Wu, J., Liu, J., Metz, C.: Transition from IPv4 to IPv6: a state-of-the-art survey. IEEE Comm. Surv. Tutorials **15**(3), 1407–1424 (2013)

# Appendix 2

**Publication II**

B. Blumbergs and R. Vaarandi. Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 707–712, Baltimore, USA, November 2017. IEEE

# Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis

Bernhards Blumbergs
NATO Cooperative Cyber Defence Centre of Excellence
IMCS UL, CERT.LV Laboratory
name.surname[a]cert.lv

Risto Vaarandi
Centre for Digital Forensics and Cyber Security
Tallinn University of Technology
name.surname[a]ttu.ee

***Abstract*—Fuzzing is a critical part of secure software development life-cycle, for finding vulnerabilities, developing exploits, and reverse engineering. This relies on appropriate approaches, tools and frameworks. File and protocol fuzzing is well covered, multiple approaches and implementations exist. Unfortunately, assessed tools do not posses the required capabilities for working with protocols, where constructing bit groups are not byte aligned. In this paper, a systematic approach is proposed and tool prototype developed for the cyber red teaming purposes. In a case study, the developed Bbuzz tool is used to reverse engineer a proprietary NATO Link-1 network protocol allowing to inject rogue airplane tracks into air operations command and control system.**

***Index Terms*—computer network operations, remote fuzzing, network security, network protocol reverse engineering, cyber red teaming**

## I. Introduction

When performing a thorough unknown target information system vulnerability test, i.e., a red teaming engagement [1], there are cases when critical network communication protocols need to be targeted and assessed in a limited period of time. In this paper, the term *binary protocol* means any computer network communication protocol consisting of multiple bit groups (i.e., bit-fields) of various length, not always being aligned to the size of one byte. When testing either a known protocol, such as IPv6, or a proprietary one, such as Link-1 [2], proper tools are required to allow testing or reverse engineering with minimum effort.

Fuzzing or fuzz-testing is an integral part of secure software development life-cycle (SDL) [3], creating a process to allow developers build more secure software. Most common open-source fuzzing tools and frameworks used by penetration testers are bundled in such GNU/Linux penetration testing distributions as Kali Linux [4] and Black Arch Linux [5]. Majority of tools are web application oriented, with some being general purpose frameworks, and only few of them developed exclusively for network protocol fuzzing. Out of those, most tools aim at application layer protocols (e.g., FTP, HTTP, SSH), but are not concerned with the underlying transport and Internet layers, which are critical to ensure correct communications. Fuzzing of lower layer protocols would address the critical security issues of the TCP/IP protocol stack implementation in operating system kernel. In assessed tools, a typical test-case definition expects one byte as a minimum amount of data and are not concerned about individual bits and non-byte aligned groups of bits. Unfortunately, a number of widely used protocols support the field sizes of various bit lengths. For example, IPv6 header Flow Label field is 20 bits long and cannot be byte-aligned to maintain the protocol specification. Furthermore, these tools are not applicable for unknown or proprietary network protocols. Proposed *Bbuzz* tool uses one bit as the smallest unit and can operate starting from Layer-2 network frames. Thus, providing features for flexible test-case creation, whilst striving to ensure a balanced simplicity of use. The implemented functionality allows to use Bbuzz for quick assessment of any protocol, its features, information carried within, automatic creation of the initial test-case, and conduct fully automated fuzzing.

This paper addresses the identified drawbacks in open-source tools for network protocol fuzzing and reverse engineering, and provides the following contributions:

1) simple and systematic approach for protocol reverse engineering (described in Section III); implemented in
2) Bbuzz tool – open-source bit-aware network protocol fuzzing framework.

In this paper the Bbuzz tool is used for analyzing network protocols and real case study is presented, Section II gives an overview of related work; Section III explains the systematic approach and describes Bbuzz core concepts; Section IV covers a case study of Link-1 protocol reverse engineering; Section V concludes this paper.

## II. Related Work

Closed source proprietary solutions (e.g., Codenomicon Defensics) tend to be expensive, and in most cases not feasible for short black-box penetration testing or ad-hoc red teaming engagements. In such cases, open-source solutions are explored and adapted to meet the testing requirements [6]. No closed source tools were assessed since no trial versions were made available or provided upon authors request.



978-1-5386-0595-0/17/$31.00 ©2017 IEEE

Tools, such as SNOOZE by Banks et.al. [7], AutoFuzz by Gorbunov et.al. [8], RFSM by Zhao et.al. [9], and T-Fuzz by Johansson et.al. [10] are either designed for application layer network protocol stateful fuzzing or applicable for specific protocols (e.g., LTE, ZigBee), and no public source code was identified. The research paper of T-Fuzz tool hints, that it supports variable bit length test-case specification, however, it is specifically designed as an extension to a proprietary framework to perform telecommunication protocol testing.

SPIKE fuzzing framework [11] is a discontinued solution developed by ImmunitySec. Even though forked SPIKE instances exist, it was disregarded because of lacking support, required steep learning curve and significant time investment to start developing SPIKE-based test-cases.

Sulley [12] is a fuzzing framework written in Python, and implements the core concepts of SPIKE. It provides rich features in data generation, monitoring of target system network, process and state activity, and tracking the detected faults. Despite being feature rich, it is not well maintained and has majority of bugs. For network protocol fuzzing, the test-case specifications are automatically byte-aligned by prepending zeroes to reach one byte. Such forced byte-alignment does not allow a required depth of granularity for network protocol testing.

Boofuzz [13] is a fork and successor of Sulley. Besides having an active development, feature addition and code maintenance, it still inherits the same core development principles of Sulley and automatically performs byte alignment of bit fields. This issue [14] was submitted by us and acknowledged by the developer for feature addition. However, up to this date, it is still open and not implemented.

Peach Community Edition [15] is a general purpose fuzzing framework providing diverse feature set and systematic approach to describing and performing testing. Peach allows to define the data and state model, implements monitoring agents and various testing engine strategies and configurations. Due to these features, it is well acknowledged and used by the software vulnerability researcher community [16]. Data and state models are described in XML formatted configuration files (i.e., Pit files). Defining them is a tedious task involving deep in advance knowledge of the target under test. Such knowledge, in most cases, is not available when starting with unknown network protocol analysis, testing or reverse engineering. Furthermore, community edition has lengthy release cycles and infrequent bug fixes, and is lacking features which are implemented exclusively in the commercial edition. When evaluating Peach applicability to IPv6 testing, it was identified, that Pit files are already available for that [17], however, configuration was limited only to connectivity information (e.g., IPv6 and MAC addresses, ports and network interface). Besides that, it was not identified that Peach would support the variable bit-field test-case specification. Peach, being a general purpose solution, tries to cover file-based application as well as network protocol testing.

Other network protocol testing tools, bundled in Kali Linux and Black Arch penetration testing GNU/Linux distributions, such as Taof [18] and Zzuf [19], exist, however, they are limited to known application layer protocols and do not allow bit-field based protocol specification and testing. Also, file-based fuzzing solutions, such as American fuzzy lop [20], were considered to generate network packet mutations to be wrapped for delivery over network. This is not feasible when further payload mutations are required based on the received replies form the system under test.

To address the identified drawbacks of assessed fuzzing solutions, Bbuzz: uses one bit as the minimum size of data insetad of one byte; is protocol independent and can be used to test any network protocol at any layer instead of being limited to a particular protocol or layer; grants flexibility to configure all communications options, if deemed necessary, instead of few fixed ones; provides functionality to aid fast protocol initial assessment and test-case definition without requiring huge time investment; has a clear and strightforward test-case syntax without the need to create complex ones; and is publicly available on GitHub.

## III. Core Concepts and Implementation

Developed Bbuzz tool prototype is written in Python 3, using standard libraries, and is released publicly under the MIT license at *https://github.com/lockout/bbuzz*.

Bbuzz aims to bring a simple and systematic work-flow in subsequent steps: obtaining communication sample, defining the base connection layer, describing the payload, performing the payload mutations, executing the fuzzing, and conducting logging, monitoring and tracking the state of the system under test. Systematic work-flow, implemented in Bbuzz object classes, is described in the following sub-sections. The analyst can interact with all separate object classes either via a Python script, Python interactive sell, or a system shell (e.g., iPython).

### A. Sample Acquisition

Protocol analysis starts either by referring to specification (e.g., IETF RFC) or by capturing traffic samples with a network packet sniffer (e.g., tcpdump or Wireshark). Packet capture gives the required starting information, such as TCP/IP protocol stack layer, payload properties, and delivery method (e.g., unicast, multicast, broadcast). For known protocols, this provides the information about the use and implementation, and in case of unknown – raw data for further analysis. Depending on the complexity of the protocol, the analyst can perform visual analysis of the sample, use the analysis functionality of the network packet capture tools, and use the Bbuzz built-in analysis functionality.

Bbuzz helper tool, written in Python Scapy framework, is used to listen on the network traffic and capture the packets matching the defined filter criteria. Captured packets are converted into their binary representation and written to a file. Bbuzz function *analyze_payload()* parses this capture file and identifies the bit-field patterns which are static (i.e., immutable) among all, and the ones which change (i.e., mutable). This allows to quickly perform pattern matching and identify a possible initial test-case. With this step, the

usage and principal properties of the protocol are identified to continue further analysis. For example, suppose a capture file contains the following data:

```
11100101101011111001101011100110
11100101101011111100100010000110
11100101101011101001101101100110
11100101101011100010011010000110
```

Bbuzz payload analysis generates the following output, representing mutable and immutable bit groups to quickly assess the communication protocol and identify the starting test case:

```
('111001011010111', immutable, 15)
('110011010111', mutable, 12)
('00110', immutable, 5)
```

To speed up the analysis and reverse engineering process of an unknown protocol, the Bbuzz tool calculates Shannon entropy for the whole payload and per each of the identified bit-fields. This unique functionality helps to quickly pinpoint the properties of the payload and its fields, revealing the likely type of information contained therein. For example, ASCII characters having the entropy of around 4.6 and compressed or encrypted data – 7.9 Shannon.

The approach in the case study uses strict filtering of captured packets based on criteria, such as source IP address, destination port, transport protocol, and payload size. This creates uniform packet capture without any noise. For Link-1 communications it is possible to use explicit filtering. There are cases where such filtering is not possible and resulting data-set would contain noise. For mining strong patterns from data-sets with noise, frequent itemset mining (FIM) algorithms have been often suggested [21] [22]. Nevertheless, the use of well-known FIM algorithms, such as Apriori, for packet payload data is complicated for several reasons. Firstly, these algorithms detect unordered groups of items, while reporting the offset of bits in packet payloads is essential for meaningful bit patterns (i.e., reporting a frequent itemset $\{0,1\}$ is not helpful for the end user). Secondly, bit patterns can be quite long (e.g., consisting of hundreds of bits), but mining long patterns from large data sets is known to be computationally expensive [23]. A FIM algorithm called LogHound [23] has been designed for encoding positional information into items (bits) and mining long patterns from larger data sets. During the experiments, we have used LogHound with higher support thresholds (e.g., 80-95%) for filtering out occasional noise and detecting bit patterns which are present in majority of packets.

### B. Establishing Basic Network Connectivity

Basic network connectivity is handled by a *Bbuzz Protocol* class to describe and create the base protocol layer, which is required to send data over the network and meet the requirements of the test. The specified initial base protocol layer allows the testing of further upper layers, those treated as payload to be described as a test-case of a Bbuzz Payload class. Bbuzz Protocol class accepts the base layer to be specified either as "raw2" for Layer-2 connections (e.g., IPv4 or IPv6 testing), "raw3" for Layer-3 connections (e.g., TCP or UDP testing), and "raw4" to specify the Layer-4 connections (e.g., FTP and Link-1 testing). For example, base connection layer "raw2" establishes a Layer-2 connectivity allowing testing of upper layer protocols, such as IPv4 or IPv6. This flexibility allows to create and run test cases without the need to define complex protocol classes and test-cases.

To define the base protocol layer, Bbuzz uses the syntax as presented in example Fig.1, and takes two parameters. The first parameter specifies the initial base layer connection, and the second one is a Python dictionary providing the configuration parameters for that specific layer. Layer-2 connection options require the destination MAC address (DESTINATION_MAC) and Ethernet frame type (ETHER_TYPE). Optionally, source MAC address (SOURCE_MAC) can be specified. The ETHER_TYPE option can be used to provide additional information besides such types as IPv6 (0x86DD) or IPv4 (0x0800), but also, for example, to include IEEE 802.1Q VLAN tagging data (e.g., VLAN tag 0x810000010800). Layer-3 connection options require the destination IP address (DESTINATION_IP) and IP version (IP_VERSION). Optionally, source IP address (SOURCE_IP), SOURCE_MAC and DESTINATION_MAC values are accepted. Layer-4 connection options should include the required ones of the "raw3" accompanied by protocol number (PROTO_NUMBER), for example TCP (SOCK_STREAM = 0x01) or UDP (SOCK_DGRAM = 0x02), and destination port (DESTINATION_PORT). Additionally, source port (SOURCE_PORT) and connection specific parameters can be set, such as configuring the network interface as broadcast by specifying the BROADCAST option. Finally, the network interface identifier (e.g., eth0, enp0s25) should be provided for binding and sending data.

Fig. 1. Example of Link-1 broadcast base Layer-4 creation

```
interface = "enp0s25"
proto = bbuzz.protocol.Protocol(
      'raw4',
      {
         "SOURCE_IP": "10.78.2.169",
         "DESTINATION_IP": "10.78.2.255",
         "PROTO": 0x02,
         "DESTINATION_PORT" : 1229,
         "BROADCAST": True
         }
      )
proto.create(interface)
```

### C. Describing Protocol Fields

Description of various protocol fields is handled by the *Bbuzz Payload* class to provide a structured approach to payload description for a test-case. Bbuzz Protocol class accepts values in different formats to ease the definition of test-cases according to data format. Payload field specifications can be added or loaded from a file to an instantiated Payload class object, therefore providing clear structure and flexibility for test-case definition.

To define the payload, Bbuzz uses syntax as presented in example Fig.2, and accepts two parameters. The first parameter represents the initial value of the data field, and the second one is a Python dictionary describing the data field and providing specific settings for mutation strategies. The value or values, in case of set of fixed known values, of the data field, can either be selected from the captured network traffic, specified according to the protocol specification, or any arbitrary value if it is chosen by the analyst. Payload field options require the following attributes: data format (FORMAT), field type (TYPE), field length in bits (LENGTH), data group (GROUP), and field mutation state (FUZZABLE). FORMAT specifies in what format the data is represented, such as binary, hexadecimal, decimal, octal, string, or bytes values. This eases the test-case definition with the values either from the protocol specification or from packet capture. Based on the format specified, the appropriate data conversion to bits will be chosen by the engine. TYPE indicates what data this field represents, and supports the following types for binary, numeric, string, delimiter, or static data. Based on the data type, the field mutation strategy will be chosen to generate sets of mutated values. LENGTH is the length of field in bits and, when performing the mutations, the size will be aligned to the defined one by the mutation engine. LENGTH set to '-1' represents a variable length field for which mutations of variable length would be produced. This is beneficial for tests of memory buffer overflow conditions. If no length is specified, it is calculated by the Payload class based on the length and type of the presented data. GROUP is a boolean value and when set to True, designates that the data value is a tuple, i.e., a set of comma separated fixed values. This can be used in case of known set of immutable values applicable for a particular field of the payload, for example, a set of Next Header values for IPv6 header. Specifying groups of values minimizes the time needed for testing, and ensures a better code coverage. If not provided, default GROUP value is set to False. FUZZABLE is a boolean parameter to represent either the data is mutable (True) or immutable (False). Additionally, Payload class calculates and assigns an unique identifier (HASH) to every field specified. This is used by some Bbuzz mutation functions to keep track of the mutation state for that particular field.

Fig. 2. Example of IPv6 header Flow Label assignment

```
load = bbuzz.payload.Payload()
load.add('0',
      {
         "FORMAT": "bin",
         "TYPE": "binary",
         "LENGTH": 20,
         "FUZZABLE": True
         }
      )
```

### D. Payload Mutation Engine

Described payload mutations are handled by *Bbuzz Mutate* class to generate mutations for the particular payload field based on specified options. This class converts all the specified data values to binary, orchestrates the mutation process, selects appropriate mutation engine for the particular data type, and produces a mutated payload instance for network transmission.

Bbuzz mutation class is instantiated by *mutagen = bbuzz.mutate.Mutate(load)* and accepts two parameters. The first mandatory parameter presents the defined Payload class object, and a second optional parameter presents Python dictionary to configure the mutation process. Currently, Mutation class provides two main strategies – generating known bad values including ones introduced by coding mistakes for the particular data type, and pseudo random value generation. A third – genetic mutation strategy, will be implemented in upcoming release, in order to precede or obsolete random generation. This approach would produce further mutations for the generated valid payload instances able to solicit a reply from the system under test. The mutation strategies have the goal to attempt to minimize the required time to get testing results and generate a more sensible test-cases. It is not feasible to brute-force, i.e., produce all of the possible mutations for data fields, since that would yield too large set of mutations, be slow and consume huge amounts of computing resources.

Mutation concepts and common coding mistakes (e.g., off-by-one), allowing to trigger software exceptions and vulnerabilities, were identified from [16] [24] [13] to be further adapted and implemented. The mutation engine generates finite amount of mutations out of known bad values according to the specified data type. Binary mutation engine performs bit-wise operations (e.g., inversions, endianess change), bit-shifting (e.g., binary shift right with prepending '1' from left), and pattern insertion. For example, few sample mutations for a bit-field with value of '00110' are *bit-flip: (11001)*, *endianess swap: (01100)* and *bit-shift-right values: (10011, 11001, 11100, 11110, 11111)*. Numeric mutation engine creates mutations based on mathematical coding mistakes, such as turning the value into a two's complement negative, deducting or adding 1 to the value to trigger off-by-one errors. For example, few mutations for initial value of '00110' are *two's complement negative: (11010)*, *addition of '1': (00111)*, and *substracting '1': (00101)*. Delimiter mutation engine performs substitution with other known delimiter values or commonly misplaced characters. For example, replacing delimiter ';' (111011) with characters like ',' (101100) or ':' (111010). String mutation engine performs string encoding modifications, such as encoding the string into UTF-8, UTF-7, or, if enabled – increasing length to trigger buffer overflows. Static defined data fields do not produce any mutations and are treated as immutable. Random mutation engine is started, unless explicitly disabled, by the Mutation class once the generated known value mutation instances have depleted. For all mutation engines, where random values are generated, the seed is constant, unless changed, to ensure that test-cases are reproducible.

Instead of immediately generating the mutation stack containing all possible mutations, a Python generator is used to

produce the next mutation instance upon request, in a fast and memory efficient manner. Mutation generator produces n-fold Cartesian product of all available payload field mutation sets. Meaning, that all possible payload combinations from individual field mutation sets are generated in order to grant the most complete test-case set. In essence, if *F1* is the protocol first field with all generated mutation set *A*, and *F2* is the second field with all generated mutation set *B*, then the Cartesian product would be calculated as follows:

$$A \times B = \{(a, b) \mid a \in A, b \in B\}$$

$$if : A = \{000, 010, 110, 100\}, B = \{11111, 00000\}$$

$$then : A \times B = \{(000, 11111), (000, 00000), (010, 11111),$$

$$(010, 00000), (110, 11111), (110, 00000), (100, 11111),$$

$$(100, 00000)\}$$

### *E. Target Monitoring and Test Execution Logging*

Monitoring the system under test and performing the logging of the fuzzing execution is handled by *Bbuzz Monitoring* class. Logging component provides the back-trace of the fuzzing process to identify which packet or a sequence of packets triggered unexpected behavior or a crash condition. Monitoring component performs tracking of the state of the target system, gathering status information (e.g., system alive state or crash dumps). As well as receiving and tracking the responses from the target system, where applicable, to allow a more intelligent approach to fuzzing process and state machine development.

Components of this class already implemented is the time-stamped logging of the sent packets and a rudimentary approach to tracking the alive status of the target system via ICMP. Further developments are described in section V. To instantiate Monitoring class, it takes an IP address of the system under test – *mon = bbuzz.monitor.Monitor(ip="10.0.244.191").*

### *F. Fuzzing Management*

The management of the fuzzing is performed by *Bbuzz Fuzz* class, which involves actions such as next mutation retrieval from mutation engine, mutated instance delivery over the established network socket, control the timing of the packet sending, logging the sent test-cases with the response from the target system, and managing the process based on the target monitoring metrics. To instantiate the Fuzzer class, it takes the defined Protocol, Mutate and Monitor objects – *fuzzer = bbuzz.fuzz.Fuzz(proto, mutagen, mon).*

The fuzzing process is started with *fuzzer.start(timing=0.5).* This instantiates the base protocol layer for communication establishment, takes next payload mutation, and sends over the network socket. While the fuzzing is ongoing, the Fuzzer class uses Monitoring class functionality to record the sent mutations, their time-stamp, and if possible – the response and state of the target system.

## IV. Case Study

Case study describes a red teaming engagement at NATO's largest technical cyber-defense exercise Locked Shields 2017 [25]. The scope of engagement was to verify the integrity of air command and control (C2) system communications. The desired effect was to present a large amount of fake unidentified aircraft tracks (i.e., volatile data representing an airplane instance at a particular moment in time), and have an impact on the situation awareness and decision making process.

The exercise network, consisting of multiple sub-nets, deployed the NATO air C2 systems in one of the subnets, consisting of ICC (Integrated C2) workstations and NIRIS (Networked Interoperable Real-time Information Services) radar instance. NIRIS is a MS Windows Server 2008 R2 with related software and services required to receive the information from the radar and broadcast this information on the local sub-net for engaged systems, such as ICC client. MS Windows 7 client with ICC client software listens to network broadcast traffic on a defined port and represents the received air tracks and details on the world map. These UDP-based network broadcast communications use Link-1 protocol wrapped in particular NIRIS format. Link-1 protocol, according to its specification, is a proprietary protocol containing groups of variable length bit fields, where each field represents a particular property of an air track (e.g., coordinates, altitude, bearing, flight number). Exercise deployment of air C2 systems were implemented as close as possible to represent real instances, and the cyber defense of these systems was assumed by the exercise participants (i.e., blue teams).

To complete this objective, with not too much information available, red team (RT) approached it as a black-box engagement. Before the start of the exercise, RT was granted access to the exercise network and air C2 component instances therein to allow attack approach assessment and development. The radar system was broadcasting legitimate air tracks on the network, received and displayed on the map by the ICC client instances. RT was able to capture this traffic to start the initial assessment, perform protocol reverse engineering and attempt to broadcast fake air tracks to be plotted on map. The Bbuzz protocol assessment capabilities allowed to quickly identify mutable and immutable bit fields of the Link-1 protocol, which allowed to start the test-case creation. It was assumed, that such properties as aircraft identification number and IFF (Identify Friend or Foe) status should not be changing for the same aircraft, therefore being treated as immutable, and properties, such as coordinates, velocity and altitude should be changing and therefore assumed as mutable. With this limited knowledge the fuzzing case was created and protocol fields mutated in order to produce observable results, which allowed to identify usage of a particular bit fields in Link-1 protocol. The aim was, in the limited amount of time, to identify only appropriate fields which are relevant for the attack, those being, aircraft number, coordinates, bearing, velocity and IFF status.

With the minimum Link-1 implementation information available, the red team used Bbuzz tool to successfuly identify critical parts of protocol in one day and further reverse engineer them. This enabled broadcasting fake air tracks to be displayed on the radar screens of the defending blue teams. In addition, for this attack to be successful, RT had to gain local area network access via other means such as spear-phishing campaign (e.g., e-mails containing malicious attachments or URL links), taking control over the firewalls (e.g., misconfiguration), and exploiting vulnerabilities in the publicly facing services (e.g., e-mail and web servers). Such attack represents a significant impact to real situation awareness, air operation decision making and overloading the air traffic control for handling and routing the fake traffic.

## V. Conclusions and Future Work

In this paper a systematic approach to network protocol fuzzing and reverse engineering is discussed, with a Bbuzz tool prototype implementation described and its applicability in real use cases for red team engagements.

Planned Bbuzz updates involve research and development for system under test state information gathering, crash log collection, genetic payload mutation based on received responses, and target system management. For target system monitoring, approaches, not impacting the state of the system and able to extract information (e.g., core dumps, daemon or service logs, network traces), should be considered. Methods, such as remote probing, agent deployment, or virtual machine (VM) hypervisor management are applicable with their benefits (e.g., hypervisor level data collection) and implicit drawbacks (e.g., using TCP/IP stack under test also for monitoring purposes). Virtualized environments provide scalability and effectiveness of the target system deployment and management (e.g., snapshots, crash state recovery). However, for fuzzing purposes, possibility to interact with the target system via a wrapper or an API should be available. Virtualization platforms, such as VMWare Workstation or SUN VirtualBox, can be considered, both having their strengths and limitations (e.g., limited to VM deployment and management, not allowing interaction with the system itself). Kernel-based virtualization (e.g., KVM, QEMU) are well integrated with GNU/Linux and have API libraries (e.g., libvirt) for most programming languages. However, the extent of the interaction with the target system is not yet researched by the authors. Micro-virtualization solutions (e.g., Docker) allow running core required components, instead of allocating resources to fully featured VM. Nevertheless, this approach requires a deeper understanding if it is enough to conduct a reliable test, and the level of interaction provides valuable information of the target system state.

Future work includes research on the Bbuzz development, and usage for finding vulnerabilities in TCP/IP protocol stack implementation in embedded operating system kernels.

## VI. Acknowledgments

The authors thank Olaf M. Maennel (TUT) and Raimo Peterson (NATO CCD CoE) for supporting this work.

## References

[1] P. Brangetto, E. Çalişkan, and H. Rõigas, "Cyber Red Teaming - Organisational, technical and legal implications in a military context," tech. rep., NATO CCD CoE, 2015.

[2] NATO Standartization Agency, "STANAG 5501, Tactical Data Exchange - Link 1 (Point-to-Point), Ed.7," standartization agreement, North Atlantic Treaty Organization, 2015.

[3] "Microsoft SDL." https://www.microsoft.com/en-us/sdl/. Accessed: 27/03/2017.

[4] "Kali Linux Tools." http://tools.kali.org/tools-listing. Accessed: 28/03/2017.

[5] "Black Arch Linux Fuzzers." https://blackarch.org/fuzzer.html. Accessed: 28/03/2017.

[6] S. D. Zhang and L. Y. Zhang, "Vulnerability mining for network protocols based on fuzzing," in *The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014)*, pp. 644–648, November 2014.

[7] G. Banks, M. Cova, V. Felmetsger, K. Almeroth, R. Kemmerer, and G. Vigna, *SNOOZE: Toward a Stateful NetwOrk prOtocol fuzZEr*, pp. 343–358. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[8] S. Gorbunov and A. Rosenbloom, "AutoFuzz: Automated Network Protocol Fuzzing Framework," *nternational Journal of Computer Science and Network Security*, vol. 10, pp. 239–245, August 2010.

[9] J. Zhao, S. Chen, S. Liang, B. Cui, and X. Song, "Rfsm-fuzzing a smart fuzzing algorithm based on regression fsm," in *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pp. 380–386, October 2013.

[10] W. Johansson, M. Svensson, U. E. Larson, M. Almgren, and V. Gulisano, "T-fuzz: Model-based fuzzing for robustness testing of telecommunication protocols," in *2014 IEEE Seventh International Conference on Software Testing, Verification and Validation*, pp. 323–332, March 2014.

[11] "The Advantages of Block-Based Protocol Analysis for Security Testing." http://www.immunitysec.com/downloads/advantages_of_block_based_analysis.html. Accessed: 30/03/2017.

[12] "Sulley fuzzing framework." https://github.com/OpenRCE/sulley. Accessed: 29/03/2017.

[13] "boofuzz: Network Protocol Fuzzing for Humans." https://github.com/jtpereyda/boofuzz. Accessed: 29/03/2017.

[14] "Fuzz a bitfield of various fixed bit sizes #88." https://github.com/jtpereyda/boofuzz/issues/88. Accessed: 29/03/2017.

[15] "Peach Community Edition." http://www.peachfuzzer.com/resources/peachcommunity/. Accessed: 28/03/2017.

[16] D. egaldo et.al., *Gray Hat Hacking: The Ethical Hacker's Handbook*, ch. 5, pp. 117 – 142. McGraw-Hill Education, 4 ed., January 2015.

[17] "IPv6 Peach Pit Data Sheet." http://www.peachfuzzer.com/wp-content/uploads/IPv6.pdf. Accessed: 29/03/2017.

[18] "Taof - The art of fuzzing." https://sourceforge.net/projects/taof/. Accessed: 29/03/2017.

[19] "Zzuf - Multi-Purpose Fuzzer." http://caca.zoy.org/wiki/zzuf. Accessed: 29/03/2017.

[20] "American fuzzy lop." http://lcamtuf.coredump.cx/afl/. Accessed: 29/03/2017.

[21] Q. Zheng, K. Xu, W. Lv, and S. Ma, "Intelligent Search of Correlated Alarms from Database Containing Noise Data," in *2002 IEEE/IFIP Network Operations and Management Symposium*, pp. 405–419, 2002.

[22] D. Brauckhoff, X. Dimitropoulos, A. Wagner, and K. Salamatian, "Anomaly Extraction in Backbone Networks using Association Rules," in *2009 ACM SIGCOMM Internet Measurement Conference*, pp. 28–34, 2009.

[23] R. Vaarandi, "A Breadth-First Algorithm for Mining Frequent Patterns from Event Logs," in *2004 IFIP International Conference on Intelligence in Communication Systems*, vol. 3283, pp. 293–308, LNCS, 2004.

[24] N. Rathaus and G. Evron, *Open Source Fuzzing Tools*. Syngress Publishing, December 2007.

[25] "Locked Shields 2017." https://ccdcoe.org/largest-international-technical-cyber-defence-exercise-world-takes-place-next-week.html. Accessed: 20/04/2017.

# Appendix 3

**Publication III**

B. Blumbergs. Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 88–99, Prague, Czech Republic, February 2019. SCITEPRESS

# Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems

Bernhards Blumbergs
*CERT.LV, IMCS University of Latvia, Riga, Latvia*
*Centre for Digital Forensics and Cyber Security, Tallinn University of Technology, Tallinn, Estonia*
*name.surname@cert.lv*

Keywords: Cyber Red Teaming, Computer Network Operations, Industrial Control Systems, Exploit Development.

Abstract: Cyber red teaming and its techniques, tactics and procedures have to be constantly developed to identify, counter and respond to sophisticated threats targeting critical infrastructures. This paper focuses on cyber red team technical arsenal development within conducted fast paced computer network operation case studies against the critical infrastructure operators. Technical attack details are revealed, attack tool released publicly and countermeasures proposed for the critical vulnerabilities found in the industrial devices and highly used communication protocols throughout the Europe. The exploits are developed in a reference system, verified in real cyber red teaming operations, responsibly disclosed to involved entities, and integrated within international cyber defence exercise adversary campaigns.

## 1 INTRODUCTION

Significant developments, such as, Industrial Ethernet (Popp and Wenzel, 2001), the merger of Operational Technology with Information Technology and use of wireless radio communications (Åkerberg and Björkman, 2009a), and replacement of proprietary protocols with common information and communication technologies (Paul et al., 2013) cause the conventional borders to disappear (Åkerberg and Björkman, 2009a), open the attacks from the Internet and facilitate malicious intrusions (Maynard et al., 2014). Debilitating cyber operations, such as, Shamoon (GReAT, 2012), EnergeticBear/Dragonfly (US-CERT, 2018), BlackEnergy (GReAT, 2016), and NotPetya (UK Foreign Office, 2018), represent the significant impact inflicted to the critical infrastructure (CI). This clearly indicates that we are already well into the age of cyber arms race, cyber operations and cyber warfare. The known cyber attack sophistication, complexity and identified numbers of state affiliated actors is steadily escalating (FireEye, 2018). Defending and responding to sophisticated adversaries are constantly evolving as an essential capability (Knapp and Langill, 2014) to be facilitated by nation states and private entities. To adjust to adversary employed unconventional tactics and to meet the current security requirements, the defence techniques, tools and procedures (TTPs) must be complied with. The cyber red teaming aims to achieve these demands by providing defensive, as well as, offensive capabilities for proactive and reactive computer network operations. The tool-set for accomplishing such operations with increased efficiency and stealth relies on developing new TTPs and expanding the technical arsenal well in advance.

This research describes executed fast-paced cyber red teaming computer network operations (CNO) against the European energy and automation sector provider systems, where industrial communication protocols and systems were targeted to successfully achieve the desired effect of disabling designated part of the power grid or modifying the industrial process. The attacked internationally standardized industrial Ethernet protocols (PROFINET RT, IEC-104) are most commonly used in Europe and China, and the industrial control device (Martem TELEM-GW6e) – widely deployed in the Baltics and Finland. Additionally, applying such real-life cases to the exercise environment greatly increases the training experience and gives valuable hands-on practice for both the attackers and defenders. Therefore, the vulnerabilities and environment nuances from executed CNOs are transferred to the international live-fire cyber defence exercise game networks, increasing the realism and training benefits.

**Impact of the Work.** The identified attack vectors and new critical vulnerabilities presented in this paper have been responsibly disclosed and coordinated with vendors, US DHS ICS-CERT and international CSIRT community. Vulnerability exploits are fully implemented and tested as working proof-of-concept (PoC) attack scripts, initially distributed only to directly involved or impacted parties. The following list contains identified weaknesses (Common Weakness Enumeration – CWE, classifiers), the newly found critical vulnerabilities (Common Vulnerabilities and Exposure – CVE, numbers) and their severity measurement (Common Vulnerability Scoring System version 3 – CVSSv3, score):

1. Missing authentication for a critical function (CWE-306; CVE-2018-10603; CVSSv3 10.0 [CRITICAL IMPACT]);
2. Incorrect default permissions (CWE-276; CVE-2018-10605; CVSSv3 8.8 [HIGH IMPACT]); and
3. Uncontrolled resource exhaustion (CWE-400; CVE-2018-10607; CVSSv3 8.2 [HIGH IMPACT]).

**Main Contributions.** Conducted research in the field of CI security and cyber red teaming CNOs against ICS/SCADA (Industrial Control Systems/Supervisory Control and Data Acquisition), provides the following contributions both to the ICS and security community:

1. identification and exploitation of security weaknesses in PROFINET IO RT, IEC-104, and TELEM-GW6/GWM;
2. technical disclosure of the developed exploits and their PoC design details;
3. vulnerability responsible disclosure and mitigation coordination;
4. clear and practical approaches for the CI security community to verify, secure and defend their systems;
5. integration of a realistic red teaming campaigns in technical cyber defence exercises; and
6. all of the prototyped attack tools and proof-of-concept code have been made publicly available on GitHub.

In this paper, Section 2 gives an overview of related work, Section 3 describes the threat scenario, identified vulnerability core concepts, and discloses their exploitation, Section 4 proposes defensive measures, and Section 5 concludes this paper.

# 2 RELATED WORK

Pfrang and Meier (Pfrang and Meier, 2018) explores the PROFINET replay attacks and their detection. *"Port stealing"* technique is used to poison the switch port to MAC address binding entries and then a replay packet is injected into the network. Authors note, that system can be harmed even by triggering valid actions in an invalid time slot. Åkerberg and Björkman (Åkerberg and Björkman, 2009b) cover two approaches on attacking the PROFINET, over shared media and packet switched network, based on eavesdropping and Man-in-the-Middle (MitM) attacks to intercept frames, derive correct values and inject or modify the frame sent to the intended destination. Also, Baud and Felser (Baud and Felser, 2006) explore the execution of the MitM attack against PROFINET by using the *Ettercap* tool. Paul, Schuster and König (Paul et al., 2013) indicate potential PROFINET vulnerabilities, review Denial-of-Service (DoS) and MitM attacks without actually executing them. Similarly, Yang and Li (Yang and Li, 2014) analyse the PROFINET from a theoretical view point. Siemens white paper on PROFINET (Siemens Industry Sector, 2008) asserts, that production networks are exposed to the same hazards as office networks, both from internal and external malicious attacks. Even a brief failure or a minor malfunction can inflict a serious damage. Maynard, McLaughlin and Haberler (Maynard et al., 2014) explore the detection and execution of a MitM attack against the IEC-104 protocol, by using a custom *Ettercap* plugin. Pidikiti et.al., (Pidikiti et al., 2013) model the IEC-101 and IEC-104 protocol theoretical vulnerabilities without executing any attacks, and together with Yang et.al., (Yang et al., 2014) and Matoušek (Matoušek, 2017) confirm that both protocols transmit the messages in clear text without any authentication mechanism, encryption or built-in security. Dondossola, Garrone and Szanto (Dondossola et al., 2011) hypothesize on intrusion paths in critical power control scenarios against IEC-104, present a methodology for intrusion process evaluation, and execute a set of simple compromise paths in a control network. Krekers (Krekers, 2017) purely focuses on learning automata from simulators and real devices implementing IEC-104. Zi Bin (Zi Bin, 2008) attempts to use a popular fuzzing framework *Sulley* and vulnerability assessment tool *Nessus* against the IEC-104 protocol and engaged devices. Knapp and Langill (Knapp and Langill, 2014) give an elaborate list on attack threats, attack paths, and lists common attack methods against industrial control systems. Auriemma released an

attack code exploiting several vulnerabilities found in systems used at industrial facilities around the world, however, these vulnerabilities do not pose a high risk as they are oriented towards operator viewing systems and not the ones controlling the critical processes (Zetter, 2011).

**Identified Gaps.** Reviewed work presents various attack vectors and vulnerabilities, however, the significant issue with the related research is that the executed attacks are noisy, can be detected and blocked (e.g., MitM, ARP cache poisoning, gratuitous ARP); the attack chain is long and has many intricate interdependencies; not enough technical detail or actual attack implementation is given, allowing its verification and reproduction; no description of the attack execution is given; attacks have complex requirements and crucial preconditions; are purely theoretical with no real attacks executed; partial attack is performed against network infrastructure without targeting the industrial process; work not publicly made available by the author; the work produces no real usable results; and has no real impact of controlling the industrial process. The work in this paper addresses all identified drawbacks to provide a simple to conduct command injection and device takeover attacks without special dependencies for execution (e.g., MitM). Those being the new findings and strengths, disclosed with related technical details, allowing actual attack verification and reproduction either for real life cyber red teaming CNOs or for integration in cyber defence exercises.

# 3 ATTACKING ICS

## 3.1 Threat Scenario

Multiple attack scenarios exist with the goal of gaining initial access to the SCADA network, such as: directly exposed SCADA components to the Internet, poorly secured and monitored direct vendor access, social engineering attack campaign, breach through the office network, attack through an infected engineer laptop, infected digital media dissemination, insider threat, outdated and loosely exposed internal network services, field station physical access, wireless radio network attacks, breach of physical security, and supply chain targeting. Within the red teaming operation, at least the aforementioned attacks are attempted leading to a possible foothold in the SCADA network. To simulate a real persistent adversary, the red team has to pursue the target no matter how much time it requires to complete the task. The described attacks in further sections rely on such initial position to be obtained to carry out the designated attack against the industrial process.

Following subsections, in a logically structured and methodological manner, describe all the successfully executed attacks, by presenting the attack goals and reasoning, execution environment, attack implementation and identified outcomes. All of the operations were executed against a previously unknown environment with no preliminary information regarding the infrastructure or communication protocols in use. Target network design specifics, such as, devices in use, were provided to the red team only when starting the operation. Additionally, operator and vendor assisted in reference network design and equipment supply, as presented in upcoming sections in a more detail. The rules of engagement were set by the operator to deliver the designated impact only to particular area of their production ICS network, while requesting periodic status updates and confirming red team planned activities. The red team has experience in attacking and reverse-engineering network protocols, developing exploits, and knowledge of various industrial Ethernet protocols and ICS/SCADA design principles. However, the industrial protocols and design of the targeted networks was encountered for the first time. The applied simple tool-set was based upon the custom GNU/Linux distribution with common network analysis tools, such as, *tcpdump* and *Wireshark*, and Python programming language with third party modules, such as, *Scapy* and *Paramiko*.

**Cyber Red Teaming Exercise Integration.** Cyber defence exercises, oriented at blue or red teaming, require realistic adversary scenarios beneficial for both teams. Such attack campaigns should be based upon real life implementations and attack cases to bring as much realism and learning value to the exercise participants as possible. Transferring the knowledge from actual cyber red teaming CNOs allows genuine system integration into game network, creation of sophisticated attack campaigns and valuable adversary threat scenarios. The cyber red teaming operations and identified novel critical vulnerabilities in this paper have been successfully integrated into annual NATO technical exercise series "Crossed Swords" (NATO CCDCOE, 2018a) and "Locked Shields" (NATO CCDCOE, 2018b). The cyber red teaming operations and identified novel critical vulnerabilities in this paper have been successfully integrated into annual international technical cyber defence exercise series. Practical CNO experience integration and merger with technically advanced game environment makes these international live-fire exercise series truly sophisticated. Moreover, identified

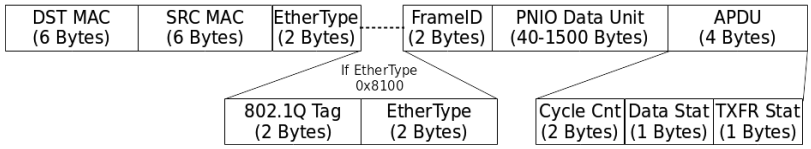



Figure 1: PROFINET IO RT frame structure (Thomas, 2013).

vulnerability disclosure and patching do not mean that the fixes are implemented in the exercise systems by the developers, or more critically – into the real ICS/SCADA by their operators. Patching life-cycle of production systems is a lengthy task, unfortunately, not always being done. It can be assumed, that disclosed critical vulnerabilities will remain not patched for an undefined period of time, allowing them being attacked.

## 3.2 PROFINET IO RT Command Injection Attack

For a cyber red teaming operation against a European industrial automation operator, a task of controlling the industrial process by targeting remotely the programmable logic controllers (PLCs) was assigned.

PROFINET IO RT (PROFINET) (PROFIBUS Nutzerorganisation e.V., 2018), developed by PROFIBUS International, enables PROFIBUS communications over Ethernet. It is an optimized real-time enabled industrial Ethernet global standard defined in IEC 61158-6-10 (Pfrang and Meier, 2018; Burtsev et al., 2017), allowing integrated networking at all automation levels (Siemens Industry Sector, 2008). This real-time Ethernet protocol, with response times in microseconds (Siemens Industry Sector, 2008), is one of the most common automation protocols in Europe and China (Pfrang and Meier, 2018; Paul et al., 2013; Siemens Industry Sector, 2008). PROFINET is described in (Siemens Industry Sector, 2008; Åkerberg and Björkman, 2009b; Knapp and Langill, 2014; Paul et al., 2013; Yang and Li, 2014; Popp and Wenzel, 2001), frame structure is reviewed in (Yang and Li, 2014) and its lack of security reviewed in (Åkerberg and Björkman, 2009b; Knapp and Langill, 2014; Burtsev et al., 2017). PROFINET is based on IEEE 802.3 Ethernet standard (Yang and Li, 2014), PNIO frames (Fig.1) are identified by EtherType 0x8892, and relies on IEEE 802.1Q (VLAN) tags to prioritize the real-time data (Siemens Industry Sector, 2008; Yang and Li, 2014) and to bypass the transport and network layers of the protocol stack (Paul et al., 2013).

**Test Environment.** The reference system, with the assistance of vendor, Siemens representative, engineers, was built by the red team to replicate the target network. The environment is represented in Fig.2 and consists of the following components: (1) MS Windows 7 based engineering workstation (IO-supervisor); (2) two Siemens SIMATIC S7-1200 PLCs, one being the IO-controller (master), and the other – IO-device (slave); (3) the electrical motor (i.e., actuator); (4) electronic switch (i.e., trigger); and (5) HP Procurve network Layer-2 switch with port mirroring support. In this setup, the IO-controller outputs are programmatically mapped to IO-device inputs. The electronic switch is connected to to master inputs (Im) and the motor – to slave outputs (Qs). Once initialization sequence is complete, both devices maintain synchronisation and input/output data exchange over the PNIO real-time frames (Baud and Felser, 2006). Once the electronic switch is triggered, the cyclic IO-data frames carry this information over the Ethernet from the master to slave, and the engine is started according to the programmed PLC ladder logic. For cyclic IO-data, only MAC addresses are used instead of IP addresses for communication between the devices (Baud and Felser, 2006).

**Attack Implementation.** Within three days, the traffic was recorded and learned through a port mirror, control frames identified, and their fields reverse engineered. The following steps were identified to execute a successful attack: identify the frames triggering the action; locate control field in the cyclic IO data; determine IO-device input/output mapping; and inject the command frames. The implementation of attack PoC is developed in *Python* using *Scapy* framework. The following paragraphs describe each attack step in a detail with the related code snippets.

*Frame Analysis.* Performing a network capture of real-time communications, even for a few seconds to intercept the command frames, results in large amount of frames. A 24 second capture has nearly 50,000 frames, where only one or two frames carry the control information triggering the state change of an industrial process. Unique frames are determined by scripting the analysis and using the Python *dictionary* data structure, where every key is unique within

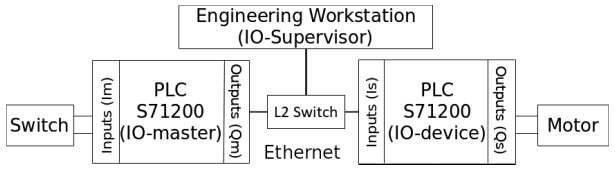


Figure 2: Schematic representation of automation network.

this dictionary. The PNIO data is assigned as dictionary key and the current frame number – as its value. With the following code, the dictionary *uMaster* contains the list of all unique frames originating from the master (*MASTER_MAC*) with the respective last seen frame sequence number:

```
pnioFrames = rdpcap("pnio_capture.pcap")
uMaster = {}; frameNo = 0
for frame in pnioFrames
    if frame.haslayer(Ether) and frame.haslayer(↩
     ↪ Raw):
        IOdata = frame[Raw].load[2:42]
        if frame[Ether].src== MASTER_MAC: uMaster↩
     ↪ [IOdata] = frameNo
    frameNo += 1    # Frame number in PCAP
```

Captured PCAP analysis yields three unique master frames. Out of those, one is the real-time synchronisation frame sent roughly every 34 microseconds, and one of the remaining is the command candidate. To verify, manual inspection of frames is performed and replay is attempted. In the Python code, the extracted unique data bytes are represented by the *FrameBytes* variable. It was identified, that the same control data has to be sent two times in a row, once with the 802.1Q tag (priority set to 6 – RT data), and right after it – untagged:

```
Frame1 = Ether(src=MASTER_MAC, dst=SLAVE_MAC, ↩
     ↪ type=0x8100)/Dot1Q(prio=6L, id=0L, vlan=0↩
     ↪ L, type=0x8892)/Raw(load=FrameBytes)
Frame2 = Ether(src=MASTER_MAC, dst=SLAVE_MAC, ↩
     ↪ type=0x8892)/Raw(load=FrameBytes)
sendp(Frame1, count=1, verbose=False)
sendp(Frame2, count=1, verbose=False)
```

*Input/output Mapping Identification.* With the correct control frame known, the PNIO data unit can be fuzzed with a simple approach, such as, by iterating every byte from 0x00 to 0xFF, injecting and observing the state of slave outputs. Reverse engineered by the author, the PNIO data unit (Fig.3) is used to transfer the data from the IO-master inputs to the IO-slave outputs over the industrial Ethernet. The transfer area, contained in the PNIO data unit, is the representation, in a binary form, of how the PLC inputs are programmatically mapped to the outputs on all master and slave devices, which are part of the same industrial process. The input/output mapping is formed into a maximum of 32 transfer areas represented in a reverse order of their definition. The IO data unit minimum size is 40 bytes, padded with 0x00 to reach this size, and can carry one or few transfer areas. Every transfer area consists of two bytes, where the most significant byte describes the zone status (0x80 – OK), and the least significant byte – the transfer area input/output state mapping. Every mapped bit represents an actual physical output on the IO-device. For example, the transfer area Qm2→Is2 is defined by the engineer, if the Is{2, 4, 5} have to be triggered, then a transfer area mapping is 0b00110100, and the whole transfer area represented by 0x8034. A special case applies to the last or if only a single transfer area is defined. The mapping is stored in the IOxS byte and the least significant byte of the transfer area is zeroed. Reusing the previous example, the transfer area is represented as 0x348000. Consider the case, where two transfer areas are configured Qm1→Is1 and Qm2→Is2, and Is{1.2, 1.7, 2.0, 2.3} are triggered. The whole PNIO data unit in this case is 0x8480098000[...0x00]. With this information, it is possible to inject crafted data frames to control the exact inputs on the slave device. If exact logic and slave outputs are not known where the actuator is connected to, then setting the transfer area mapping byte to 0xFF will trigger all of the inputs. This, of course, can cause unexpected results and possible damage if multiple actuators are controlled by the same transfer area on different outputs.

*Command Frame Injection.* According to the threat scenario, a position in SCADA network is gained. The PLC MAC addresses are quickly learned through a non-intrusive network ARP scan. Additional information can be obtained through passive eavesdropping, locating system documentation on the network systems, or setting a rogue IO-supervisor to automatically learn the network and download the device running configuration. This allows to inject control frames directly on the network by spoofing the master and slave MAC addresses. The lack of security mechanisms in PROFINET allows a spoofed node to impersonate a master node, granting control over all configured slaves (Knapp and Langill, 2014). In principle, to attempt to inflict a permanent mechanical damage to the PLC or the actuator, this attack can be executed in an infinite loop, where output states are alternated between *on* and *off* in timed control frame bursts.

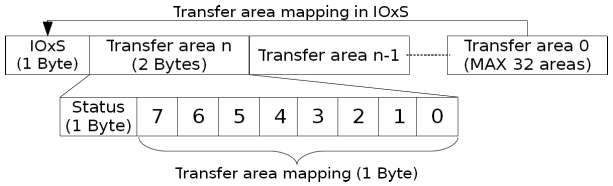


Figure 3: Reverse engineered PNIO data unit fields.

**Attack Execution and Results.** Within the CNO, identified critical flaws in the protocol and its implementation were successfully exploited to inflict severe damage remotely to the distributed industrial automation process. The time needed to complete the operation was split into two phases – three days for reference system analysis and attack development, and three days of active engagement against an automation operator. The devices successfully targeted, to control a part of a designated industrial automation network, were Siemens SIMATIC S7-1200 (v4.0) PLCs, which were the most widely used controllers in the target network. This operation resulted in a responsible vulnerability disclosure to US DHS ICS-CERT, Siemens, and PROFIBUS, with the identified attack vector mitigation solutions still being investigated. The PROFINET attack proof-of-concept code is released publicly on GitHub under the MIT license https://github.com/lockout/iec104inj/tree/master/poc/porfinet-poc.

## 3.3 IEC-104 Command Injection Attack

For a cyber red teaming operation against a European power grid operator a task of disabling the power supply in a dedicated power grid segment, by controlling remotely the circuit breakers, was assigned.

IEC 60870-5-104 is an international standard (IEC, 2006) which is widely used in European control communication for water, gas and electricity (Maynard et al., 2014). IEC-104 allows the transmission of IEC 60870-5-101 (IEC-101) serial frames over TCP/IP (Maynard et al., 2014), which can be utilized for telecontrol tasks in SCADA systems (Yang et al., 2014). Despite the IEC standards being proprietary, the following papers give an overview of the IEC-104 protocol (Yang et al., 2014; Matoušek, 2017; Maynard et al., 2014; Pidikiti et al., 2013), packet structure and message formats (Krekers, 2017; Matoušek, 2017; Maynard et al., 2014).

**Test Environment.** Before targeting the operational environment, the initial work was carried out in a reference environment, which was built by an experienced vendor, Martem, as close to the real-world implementation as possible. The environment is shown in Fig.4, and consists of the following components: (1) MS Windows 10 based SCADA system for industrial process supervision and management; (2) Martem TELEM-GW6/GWM data concentrator (RTU) for industrial process control; (3) two Schneider Electric VAMP-59 and one VAMP-300 controlling, measuring and protecting the power line; (4) three electrical circuit breakers (CB); and (5) HP Procurve network Layer-2 switch with port mirroring support. SCADA system issues the commands over the IEC-104 to the RTU, which translates and relays the commands to the actuators (e.g., VAMPs, circuit breakers) for execution.

**Attack Implementation.** Within a two-day CNO the communications between the RTU and SCADA systems was intercepted, analysed, reverse-engineered and fuzzed to accomplish the desired effect. The following attack steps were identified for a successful command injection and industrial process take over: establish the TCP connection to the RTU; prepare the RTU for data transfer by sending the STARTDT command; inject the IEC-104 commands to control the circuit breakers; close the data transfer by sending a STOPDT command to the RTU; and close the TCP connection. The implementation of attack PoC is developed in *Python 3* using its standard library. The following paragraphs describe each attack step in a detail with the related code snippets.

*Establishing a TCP Connection.* Without firewall, the RTU will accept any incoming connections on TCP/2404 (IEC-104 port). A TCP three-way handshake has to be completed to proceed with the IEC-104 communications. All of exchanged packets will belong to the same TCP stream, which can be a large amount for an intensive industrial process. This can be accomplished by establishing a TCP connection to the RTU with the Python *socket* standard library and creating an object named *sock* (used in this paper's code listings).

*Start Data Transfer.* To prepare the RTU for incoming data transfer a STARTDT (START Data Transfer) command needs to be issued first. This command, according to the standard, has to be sent in

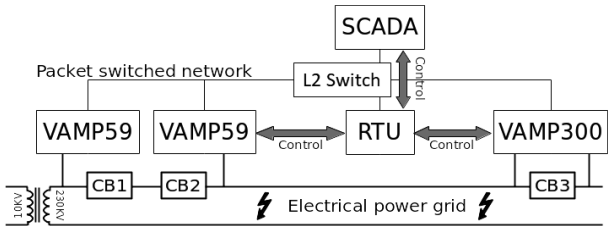


Figure 4: Schematic representation of the power grid networks.

the U-Format (Fig.5a), with the STARTDT Act bit set, thus making the first octet to be 0x07 with the remaining three set to 0x00, the resulting control frame containing "0x07,0x00,0x00,0x00". IEC-104 control command packet is assembled following big-endian order. Control command follows a fixed length telegram format (Fig.6b), which excludes the ASDU. It starts with a byte with value 0x68 (104) indicating the IEC-104 communications, followed by the Application Protocol Data Unit (APDU) length which is fixed to 4 Bytes – the U-Format length, and the STARTDT control frame. Following code assembles and sends the STARTDT command to the RTU:

```
START = b'\x68'                # Startbyte = 0↩
    ↪ x68
FRAME = b'\x07\x00\x00\x00'    # STARTDT Act
ApduLen = b'\x04'              # Payload length
APCI = START + ApduLen + FRAME
sock.sendall(APCI)
```

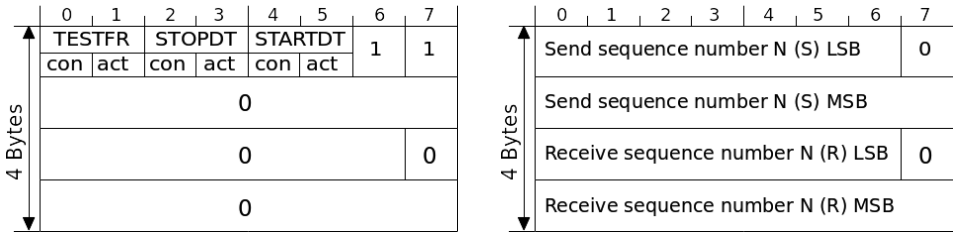
Depending on the targeted RTU maker and model, a sleep time of up to 1 second needs to be added between each sent packet. In the PoC code, importing *sleep* function from the *time* library, 0.5 seconds of sleep proved to work for all of the tested RTUs.

*IEC-104 Command Injection.* To complete this part, information on Application Service Data Unit (ASDU) structure was learned (IEC, 2006; Matoušek, 2017), main data fields identified through *Wireshark* iec104 dissector, data encoding reverse engineered, and control message command fields fuzzed. Depending on the control flow design choice by the engineer, two process telegram types can be used to issue the command to RTU – single command C_SC_NA_1 Act (typeID - 0x2d) and double command C_DC_NA_1 Act (TypeID - 0x2e). This command has to be formed according to the variable length telegram format (Fig.6b) using the ASDU structure (Fig.6a). For these command messages, the ASDU APCI header follows the I-Format structure (Fig.5b), containing the send (Tx) and receive (Rx) transmission sequence numbers. These numbers, similar to TCP sequence numbers, are incremented by the communicating parties to keep track of the exchanged messages. To target an existing ongoing communication, either a sequence number prediction or MitM has to be performed. However, both of these approaches are not so trivial to execute and might trigger an alert. Instead, it was identified, that, after initiating a TCP connection and the data transfer, the (Tx,Rx) can be set to the (0,0) thus tricking the RTU into establishing a new communication session. For the succeeding ASDU fields, the following command option values were assigned: TypeID – single command (0x2d) or double command (0x2e), structure qualifier (SQ) = 0b0 (multiple objects), number of information objects = 0b0000001 (one object), test bit (T) = 0b0 (not in use), positive/negative (P/N) = 0b0 (not in use), CoT (cause of transmission) = 0b000110 (command object), originator address (ORG) = 0x00 (address not used), and ASDU address (COA) = 0x0100 (1 in little-endian format).

Information object address (IOA) is the destination address in the control direction. It is a three byte field, with the first two bytes used for the address value, allowing to address $2^{16}$ (65536) objects. Randomly guessing or brute-forcing this address space is senseless. Either these values are learned by eavesdropping on the communications, or by using the most common used ones. When targeting various implementations, the common pattern for address assignment was in a sequence of 101, 102, 103..., which follows an unwritten SCADA engineering best practice. In the executed attacks these IOA addresses were identified to be assigned to the power grid circuit breakers. IOA value correct encoding is handled by a function using *hexlify* from the *binascii* library to convert hexadecimal values to bytes:

```
def apci_ioa_enc(number):
    hexnum = hex(number)[2:].zfill(6)
    return unhexlify(hexnum)[::-1]
```

The actual value, controlling the state of the circuit breakers (IOA), is either the single command object (SCO) or double command object (DCO) information byte. The byte values are 0x00 (off) and 0x01 (on) for SCO, and 0x05 (off) and 0x06 (on) for DCO. The command injection is performed with the following code:




(a) U-Format. (b) I-Format.

Figure 5: IEC-104 control field format types. (IEC, 2006).

```
# ASDU header
TypeID = b'\x2d'                # C_SC_NA_1 Act
opt = b'\x01\x06\x00'
Addr = b'\x01\x00'              # ASDUaddr=1
IOA = apci_ioa_enc(switchid)    # 101,201...
SCO = b'\x00'                   # 0x00=off,0x01=↩
    ↪ On
ASDU = TypeID + opt + Addr + IOA + SCO
# APCI header
Tx = b'\x00\x00'; Rx = b'\x00\x00' # TxID,RxID↩
    ↪ =0
ApduLen = msg_len(Tx + Rx + ASDU)
APCI = b'\x68' + ApduLen + Tx + Rx
# APDU payload
APDU = APCI + ASDU
sock.sendall(APDU)
```

In principle, to inflict a permanent mechanical damage to the circuit breakers, this attack can be executed in an infinite loop, where state values are alternated between *on* and *off*.

*Stop Data Transfer.* Similar to starting, the stopping of data transmission relies on a control message in the U-Format (Fig.5a), with the STOPDT Act bit set. The control frame "0x13,0x00,0x00,0x00" is sent to the RTU. If this is not executed, then a DoS condition (CVE-2018-10607) may be triggered.

*Close the TCP Connection.* Once the control commands have been injected, the TCP connection is gracefully terminated with `sock.close()`.

**Attack Execution and Results.** Within the CNO, identified critical flaws in the protocol and its implementation were successfully exploited to inflict severe damage remotely to the distributed power grid infrastructure segment. The operation was split and accomplished in multiple phases – two days for attack development in the reference network and four days of active engagement against power grid operators. Within this campaign, various RTU models, implementing IEC-104 communications, were tested and successfully targeted, those being: Martem TELEM-GW6e, Siemens A8000, Ellat, and Londelec. This CNO resulted in responsible vulnerability disclosure to affected vendors, providing recommendations on mitigations depending either the vendor will fix this issue or not, coordination of the vulnerabilities within the involved CSIRT community, and two CVEs assigned (CVE-2018-10603, CVE-2018-10607). The IEC-104 protocol command injection tool *iec104inj* has been released publicly on GitHub under the MIT license at https://github.com/lockout/iec104inj, along with the IEC-104 DoS proof-of-concept code at https://github.com/lockout/iec104inj/tree/master/poc/iec104dos-poc.

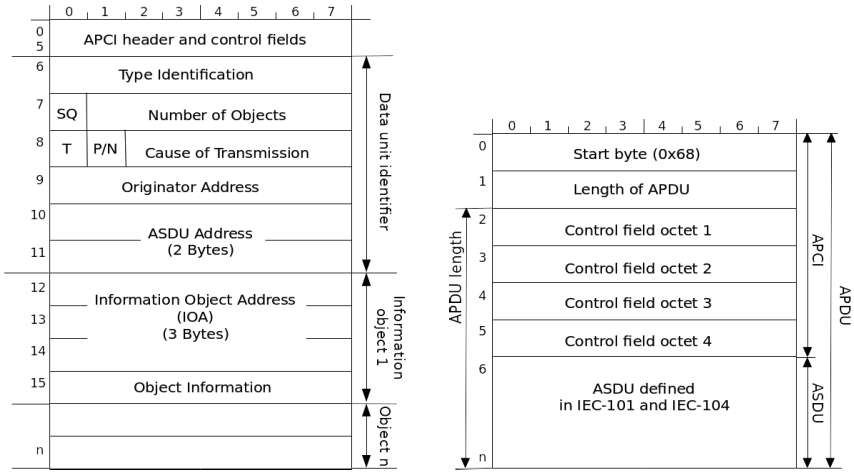
## 3.4 TELEM-GW6/GWM Control Takeover

Within a red teaming CNO against a European power grid operator, the task was to disrupt power supply remotely by any means possible. This implies not only targeting the network protocols, but also the industrial process control devices and their implementation.

Martem AS developed data concentrator TELEM-GW6e is an embedded, GNU/Linux based remote terminal unit (RTU), typically deployed at the remote field facilities to serve as a central point for communication between the controlling SCADA network and the deployed field devices. Besides protocol translation and relay, it has a wide variety of functions, such as, the VPN server, client and a firewall. It is mainly used in the energy and automation sector in the Baltic states and Finland.

**Test Environment.** The same test environment, as described in section 3.3 and represented in Fig.4, is used to find alternative paths on compromising the industrial process remotely.

**Attack Implementation.** The attacks against this RTU were developed and successfully executed within the same time frame as the IEC-104 protocol attacks. The following steps were identified to conduct the attack: find and investigate sensitive default information leak; examine the update process; inspect the configuration file structure; analyse the file system permissions; pack and upload malicious configuration; and execute RTU takeover. The attack PoC




(a) ASDU structure. (b) Telegram format.

Figure 6: IEC-104 payload specification. (IEC, 2006).

is developed in *Python 3* by using third party library *Paramiko* and its module *SCP* for interaction with the SSH. The following paragraphs describe each attack step in a detail with the related code listings. Despite attack technical detail disclosure, few minor details, such as credentials and file names, are redacted. This is done intentionally due to the patch for this vulnerability only recently being released after long development cycle and the sensitive nature of the information held therein. Fixing vulnerabilities was not trivial and required a redesign of the RTU management and configuration process.

*Information Leak Investigation.* The intended way for managing and configuring the RTU is through the vendor developed MS Windows application, freely downloadable from their website. Additional configuration ways include web-console and SSH terminal access. Upon examination of this application it was identified, that: (1) it relies on the *Putty* SSH tool suite packaged into the binary; (2) RTU configuration and management happens over the SSH; (3) has the default RTU access SSH credentials embedded; and (4) provides full action trace log in the console. With this information, it is possible to connect to the RTU over the SSH with the obtained default credentials. These credentials grant limited user rights on the RTU and it is possible to browse the file system, list running processes, network connections and assess its structure as much as the permissions allow. The RTU is running a customized version of embedded GNU/Linux with *BusyBox*. The *root* user account exists on the system, but its password is unknown, and it is not possible to gain escalated privileges with the existing user rights. Therefore, alternative ways for gaining *root* access are further explored. Python *Paramiko* is used to establish a SSH connection to the RTU (*SSHhost*) with the default credentials (*SSHUsername, SSHpassword*), creating an object named *SSHclient* (used in this paper's code listings).

*Update Process Examination.* The configuration utility logs the executed commands with all the parameters. It can be identified, that file system path, from where the current running configuration is read or to where a new one is being written, corresponds to */usr/local/[redacted]*. The file permissions in this directory are with write and read permissions for the default limited user, also used by the configuration utility. Such approach poses a major security risk, since anyone with the default credentials is able to read and write these critical configuration files, however, privilege escalation is not straightforward from this position. The following code retrieves and unpacks the current running configuration from the RTU:

```
with scp.SCPClient(SSHclient.get_transport()) as ↩
    ↪ SCPclient:
    SCPclient.get('/usr/local[redacted].tar.xz')
tar = tarfile.open('[redacted].tar.xz', 'r:xz')
tar.extractall(path="setup"); tar.close()
```

The RTU configuration file is a *.tar.xz Linux tarball* containing the directory structure representing the actual Linux file system. If this tarball is repackaged with existing files modified or new ones added to the archive directory structure, named with an extension of *.new.tar.xz*, stored on the RTU file system configuration directory, and the RTU is rebooted to

initialize the new configuration file, then all the files in the tarball are unpacked and stored at the actual Linux file system upon system start, with the *system* privileges. Such approach is dangerous, since the attacker can package any file, upload it to the RTU and reinitialize it, thus rewriting or modifying any arbitrary protected system file. This implies a rewrite or alteration of the RTU configuration XML file containing all the parameters of the industrial process. The following code packages the new configuration file and uploads it to the RTU:

```
tar = tarfile.open('[redacted].tar.xz', 'w:xz')
tar.add('.'); tar.close()
with scp.SCPClient(SSHclient.get_transport()) as ↩
    ↪ SCPclient:
    SCPclient.put('[redacted].new.tar.xz',
        '/usr/local/[redacted].new.tar.xz')
```

A system watchdog daemon is monitoring the path */var/local/[redacted]* and if a new empty file with a particular name is created, then the RTU is rebooted and reinitialized. This path is writeable with the default limited user permissions. Despite this not endangering the system directly, it can be abused by creating this file every time upon the RTU start, thus throwing it into a reboot loop and causing a DoS condition. The following code creates the new file on the remote system path to reinitialize the RTU via the watchdog daemon:

```
stdin, stdout, stderr = SSHclient.exec_command('↩
    ↪ touch /var/local/[redacted]')
```

*RTU Control Takeover.* Within the red team operation, a new */etc/shadow* and a modified */etc/sshd_config* files were packaged and uploaded to the RTU. During the system reinitialization these two files overwrote the existing system files. This resulted in a permitted *root* SSH login with known, attacker controlled, *root* credentials, granting full system access remotely. Additional attack, including all other possibilities, also includes an overwrite of system *init* files to execute any arbitrary command or a binary, such as, a back-door or a root-kit upon system start.

**Attack Execution and Results.** Within this operation the Martem TELEM-GW6e RTU was targeted and a remote attack executed allowing full compromise of the device and the industrial process. This operation resulted in a responsible vulnerability disclosure to affected vendor, coordination of the vulnerabilities within the involved CSIRT community, and one CVE assigned (CVE-2018-10605). Additionally, a XSS vulnerability in the RTU web configuration console was found (CWE-79; CVE-2018-10609; CVSSv3 7.4 [HIGH]). The Martem RTU takeover proof-of-concept code is released publicly on GitHub under the MIT license https://github.com/lockout/iec104inj/tree/master/poc/gw6e-poc.

# 4 DEFENDING AGAINST ATTACKS

The author worked with industry experts from DHS ICS-CERT and Martem AS to produce security advisories (ICSA-18-142-01, MartemSA1805182, MartemSA1805184) and the following recommendations on securing the industrial control systems (DHS ICS-CERT, 2018): strong user access password policy; authorization with SSH keys instead of passwords; service access control with a device firewall; ingress and egress traffic filtering with the network perimeter firewall; trusted communication partner IP address definitions on the ICS devices; using secure VPN connections; enabling the web interface on the devices only for the required short period of time; keeping the software and firmware up to date; configuration file transport encryption; ensuring control system devices are not accessible from the Internet; and IEC-101/104 security extensions IEC TS 60870-5-7 (implemented rarely due to operational concerns, legacy issues and costs (Maynard et al., 2014)). Additionally, the following security measures were identified in the related work: use of IDS and DPI (Yang et al., 2014; Pfrang and Meier, 2018); NetFlow monitoring (Matoušek, 2017); and hunting for network intrusions (Siemens Industry Sector, 2008). These recommendations are applicable to securing majority of ICS/SCADA environments running various industrial protocols. Moreover, to attempt the detection of the described attacks, the author worked with Check Point to create IDS signatures, released in Check Point application control update 180505.

# 5 CONCLUSIONS AND FUTURE WORK

This paper addresses a fundamental problem in a way how CI protocol specifications are created and maintained by the industry, on how vendors design and support their implementation, and how operators deploy, in most cases, without considering mitigation or monitoring solutions. In this work, attack vectors targeted at fundamental flaws, in the context of cyber red teaming, against PROFINET, IEC-104 and a RTU were explored in a great detail, allowing their assessment and verification. The identified critical vulnerabilities have been responsibly disclosed to involved parties and securely coordinated with the af-

fected community. Furthermore, defensive measures are proposed and IDS signatures created. As well as, developed attack tools released publicly on GitHub. Despite the security countermeasure existence and fixes for the identified critical vulnerabilities being released, it cannot be determined when and if at all these vulnerabilities would be patched or mitigated at the industrial operator side. Therefore, cyber red teaming operations should be considered and executed at a regular basis by the critical infrastructure owners to address these issues.

Future work includes focusing on targeting the IEC 61850 protocol stack and its implementation on broad range of common vendor appliances.

## ACKNOWLEDGMENTS

The author thanks NATO CCD CoE, CERT.LV, ICS-CERT, and Martem. More specificly, Baiba Kaškina and Raimo Peterson for supporting this work with time and technical equipment, Priska Pietra for analysing the PROFINET control data field, Artūrs Daņiļēvičs for CVE-2018-10609, and Rain Ottis, Risto Vaarandi and Olaf Maennel for valuable academic guidance and advise.

## REFERENCES

Åkerberg, J. and Björkman, M. (2009a). Exploring network security in profisafe. In Buth, B., Rabe, G., and Seyfarth, T., editors, *Computer Safety, Reliability, and Security*, pages 67–80, Berlin, Heidelberg. Springer Berlin Heidelberg.

Åkerberg, J. and Björkman, M. (2009b). Exploring Security in PROFINET IO. In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, volume 1, pages 406–412.

Baud, M. and Felser, M. (2006). Profinet IO-Device Emulator based on the Man-in-the-middle Attack. In *2006 IEEE Conference on Emerging Technologies and Factory Automation*, pages 437–440.

Burtsev, A. G., Klishevich, D. M., and Polyanskii, A. V. (2017). Protection of standard network protocols of automated production control systems. *Russian Engineering Research*, 37(3):224–232.

DHS ICS-CERT (2018). Advisory (ICSA-18-142-01) Martem TELEM-GW6/GWM. https://ics-cert.us-cert.gov/advisories/ICSA-18-142-01. Accessed: 04/06/2018.

Dondossola, G., Garrone, F., and Szanto, J. (2011). Cyber risk assessment of power control systems – a metrics weighed by attack experiments. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–9.

FireEye (2018). Advanced Persistent Threat Groups. Who's who of cyber threat actors. https://www.fireeye.com/current-threats/apt-groups.html. Accessed: 12/07/2018.

GReAT (2012). Shamoon the Wiper âĂŞ Copycats at Work. https://securelist.com/shamoon-the-wiper-copycats-at-work/57854/. Accessed: 11/07/2018.

GReAT (2016). BlackEnergy APT Attacks in Ukraine employ spearphishing with Word documents. https://securelist.com/blackenergy-apt-attacks-in-ukraine-employ-spearphishing-with-word-documents/73440/. Accessed: 11/07/2018.

IEC (2006). Internationl Standard: IEC 60870-5-104. Transmission protocols âĂŞ Network access for IEC 60870-5-101 using standard transport profiles. 2nd ed. 2006-06. https://webstore.iec.ch/preview/info_iec60870-5-104%7Bed2.0%7Den_d.pdf. Accessed: 08/07/2018.

Knapp, E. and Langill, J. T. (2014). *Industrial Network Security. 2nd Edition*. Elsevier.

Krekers, M. (2017). Assessing the Security of IEC 60870-5-104 Implementations using Automata Learning. Master's thesis, University of Twente, Enschede, Netherlands.

Matoušek, P. (2017). Description and analysis of IEC 104 Protocol . Tecnical report, Brno University of Technology.

Maynard, P., McLaughlin, K., and Haberler, B. (2014). Towards Understanding Man-In-The-Middle Attacks on IEC 60870-5-104 SCADA Networks. In *Proceedings of the 2Nd International Symposium on ICS & SCADA Cyber Security Research 2014*, ICS-CSR 2014, pages 30–42, UK. BCS.

NATO CCDCOE (2018a). Exercise Crossed Swords Practised Cyber-Kinetic Operations in Latvia . https://ccdcoe.org/exercise-crossed-swords-practised-cyber-kinetic-operations-latvia.html. Accessed: 04/06/2018.

NATO CCDCOE (2018b). Locked Shields 2018. https://ccdcoe.org/largest-international-live-fire-cyber-defence-exercise-world-be-launched-next-week.html. Accessed: 14/07/2018.

Paul, A., Schuster, F., and König, H. (2013). Towards the Protection of Industrial Control Systems – Conclusions of a Vulnerability Analysis of Profinet IO. In Rieck, K., Stewin, P., and Seifert, J.-P., editors, *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 160–176, Berlin, Heidelberg. Springer Berlin Heidelberg.

Pfrang, S. and Meier, D. (2018). Detecting and preventing replay attacks in industrial automation networks operated with profinet io. *Journal of Computer Virology and Hacking Techniques*.

Pidikiti, D. S., Kalluri, R., Kumar, R. K. S., and Bindhumadhava, B. S. (2013). SCADA communication protocols: vulnerabilities, attacks and possible mitigations. *CSI Transactions on ICT*, 1(2):135–141.

Popp, M. and Wenzel, P. (2001). PROFInet-linking worlds. In *ETFA 2001. 8th International Conference on Emerging Technologies and Factory Automation.*

*Proceedings (Cat. No.01TH8597)*, volume 2, pages 519–522 vol.2.

PROFIBUS Nutzerorganisation e.V. (2018). PROFINET - the leading Industrial Ethernet Standard . https://www.profibus.com/technology/profinet/. Accessed: 08/07/2018.

Siemens Industry Sector (2008). Automate with the open Industrial Ethernet standard and profit now. PROFINET. White paper, Siemens AG.

Thomas, P. (2013). An Introduction to PROFINET Frame Analysis using Wireshark. https://profibusgroup.files.wordpress.com/2013/01/w4-profinet-frame-analysis-peter-thomas.pdf. Accessed: 02/08/2018.

UK Foreign Office (2018). Foreign Office Minister condemns Russia for NotPetya attacks. https://www.gov.uk/government/news/foreign-office-minister-condemns-russia-for-notpetya-attacks. Accessed: 04/06/2018.

US-CERT (2018). Alert (TA18-074A). Russian Government Cyber Activity Targeting Energy and Other Critical Infrastructure Sectors. https://www.us-cert.gov/ncas/alerts/TA18-074A. Accessed: 12/07/2018.

Yang, M. and Li, G. (2014). Analysis of PROFINET IO Communication Protocol. In *2014 Fourth International Conference on Instrumentation and Measurement, Computer, Communication and Control*, pages 945–949.

Yang, Y., McLaughlin, K., Sezer, S., Yuan, Y. B., and Huang, W. (2014). Stateful intrusion detection for IEC 60870-5-104 SCADA security. In *2014 IEEE PES General Meeting. Conference Exposition*, pages 1–5.

Zetter, K. (2011). Attack Code for SCADA Vulnerabilities Released Online. https://www.wired.com/2011/03/scada-vulnerabilities/. Accessed: 08/07/2018.

Zi Bin, C. (2008). Testing and Exploring Vulnerabilities of the Applications Implementing IEC 60870-5-104 Protocol. Master's thesis, KTH Electrical Engineering, Stockholm, Sweden.

# Appendix 4

**Publication IV**
R. Vaarandi, B. Blumbergs, and E. Çalışkan. Simple event correlator - Best practices for creating scalable configurations. In *2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 96–100, Orlando, USA, March 2015. IEEE

2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)

# Simple Event Correlator - Best Practices for Creating Scalable Configurations

Risto Vaarandi, Bernhards Blumbergs
Department of Computer Science
Tallinn University of Technology
Tallinn, Estonia

Emin Çalışkan
Cyber Security Institute
TÜBİTAK
Kocaeli, Turkey

***Abstract*—During the past two decades, event correlation has emerged as a prominent monitoring technique, and is essential for achieving better situational awareness. Since its introduction in 2001 by one of the authors of this paper, Simple Event Correlator (SEC) has become a widely used open source event correlation tool. During the last decade, a number of papers have been published that describe the use of SEC in various environments. However, recent SEC versions have introduced a number of novel features not discussed in existing works. This paper fills this gap and provides an up-to-date coverage of best practices for creating scalable SEC configurations.**

***Keywords—Simple Event Correlator; event correlation; event processing; log file analysis***

## I. Introduction

During the past two decades, event correlation has become a prominent monitoring technique in many domains, including network fault monitoring, system administration, fraud detection, malicious insider and intrusion detection. Also, event correlation is one of the cornerstones for achieving better situational awareness. In order to address event analysis tasks in various domains, many commercial event correlation solutions have been created. Since its introduction in 2001 by one of the authors of this paper [1], Simple Event Correlator (SEC) has become a widely used open source alternative to commercial offerings. During the last decade, a number of papers have been published that describe the use of SEC in various environments, including academia [2], supercomputing centers [3–6], financial institutions [7, 8], telecom companies [8], and military [9]. SEC has been used for a wide range of purposes, including UNIX server log analysis [2], monitoring of supercomputer clusters [3–5], research experiments [6], correlation of large event volumes in centralized logging infrastructures [7], analysis of various security logs [9–11], IDS alarm classification [12], and network management [1, 8, 13]. However, many past papers have provided generic overviews of SEC deployments, and do not cover its advanced features in sufficient details. Moreover, its recent versions have introduced a number of new features that existing works have not discussed. The current paper fills this gap and provides an up-to-date coverage of best practices for scalable deployment of SEC. The remainder of this paper is organized as follows – section II discusses related work, section III presents recommendations for creating scalable SEC configurations, and section IV concludes the paper.

## II. Related Work

Event correlation has received a lot of attention by many researchers, and most papers have adopted the following definition by Jakobson and Weissman [14] – event correlation is a conceptual interpretation procedure where new meaning is assigned to a set of events that happen within a predefined time interval. A number of approaches have been proposed for correlating events, including rule-based [14], graph-based [15], codebook-based [16], and Bayes network based [17] methods. In the industry, event correlation is implemented in most network management and SIEM frameworks, such as HP Openview, Tivoli, ArcSight, and AlienVault. In the open source domain, there are several log monitoring tools with some event correlation functionality – for example, Swatch [18] implements event counting and thresholding operations, while LogSurfer [19] supports pairwise event correlation. Furthermore, NxLog syslog server [20] directly borrows from SEC rule language and re-implements some SEC functionality in its core. ESPER [21] is a development toolkit which allows for augmenting Java and .NET applications with event correlation features. The first papers which provided detailed recommendations on deploying SEC were authored by Rouillard [2] and Vaarandi [10] a decade ago. The treatment by Vaarandi and Grimaila [11] is more recent, but does not address the creation of scalable configurations, and does not describe the new features of the current major release (introduced in 2012). In the following section, we will provide a detailed discussion of these topics.

## III. Best Practices and Recommendations

From its inception, SEC was designed to be as lightweight as possible. For this reason, it was implemented as a UNIX tool which incorporates event matching and parsing, event processing, and output generation into a single program. SEC can be used interactively in UNIX shell pipelines, executed as a daemon (or several daemons), connected to other applications over FIFOs, pipes, and network sockets, etc. Other design considerations were platform independence and ease of installation – since SEC is written in Perl and requires no additions to a standard Perl installation, it runs on all modern UNIX and Linux platforms. SEC uses rule-based approach for event correlation, where rules are arranged into sequences (rulesets), with each ruleset stored in a separate text file. Input events can be received from regular files, FIFOs, and standard

978-1-4799-8015-4/15/$31.00 ©2015 IEEE

input. Input events are typically matched with regular expressions, but for advanced matching and parsing custom Perl functions can be defined. SEC has been designed for real-time event processing only and incoming events are tagged with timestamps of reception. In order to achieve fast memory-based read-write data sharing between rules, event correlation operations, and other SEC entities, SEC has been implemented as a single-threaded tool. Nevertheless, it is straightforward to run many SEC instances with independent rulebases on the same host simultaneously.

### A. Joining Rules Into Event Correlation Schemes

A number of web pages and papers provide examples of one SEC rule which correlates events independently. However, by using *contexts*, *synthetic events*, and other data sharing measures, several rules can be joined together into more powerful event correlation schemes. For example, the ruleset in Fig. 1 has been designed for processing Snort IDS syslog alarms, in order to detect repeated multifaceted attacks from the same host. The ruleset assumes the following alarm format:

*Oct 25 11:36:06 mysensor snort[12341]: [1:16431:5] SQL generic sql with comments injection attempt - GET parameter [Classification: Web Application Attack] [Priority: 1] {TCP} 192.168.17.13:43148 -> 10.12.23.39:80*

```
#
# The rules below are stored in /etc/sec/ids.sec
#

type=EventGroup
ptype=RegExp
pattern=snort\[\d+\]: \[(\d+:\d+):\d+\] .*\
 \{\w+\} ([\d.]+)(?::\d+)? -> [\d.]+(?::\d+)?
context=!IP_$2_ALARM_$1
count=alias ATTACKER_$2 IP_$2_ALARM_$1; \
  create TRIGGER_$1_$2 120 ( unalias IP_$2_ALARM_$1 )
init=create ATTACKER_$2
end=delete ATTACKER_$2
desc=attacking host $2
action=event Multifaceted attack from $2
thresh=10
window=120

type=SingleWithThreshold
ptype=RegExp
pattern=Multifaceted attack from ([\d.]+)
desc=multifaceted attacks from $1
action=pipe 'Continuous multifaceted attacks from $1' \
       /bin/mail root@example.com
thresh=5
window=1800
```

Fig. 1. Ruleset for processing Snort IDS alarms.

In order to start a SEC daemon for processing Snort IDS alarms that will be appended to */var/log/messages*, the following command line can be used:

*/usr/bin/sec --conf /etc/sec/ids.sec --input /var/log/messages --detach*

The first rule depicted in Fig. 1 will match an incoming Snort IDS syslog alarm with the regular expression which sets the *$1* match variable to alarm ID and *$2* match variable to attacker IP address. For example, if the above example Snort alarm is observed, match variables will be set as *$1=1:16431* and *$2=192.168.17.13*. The rule will then substitute match variables in the Boolean expression given with the *context* field, and the expression evaluates TRUE if the context *IP_192.168.17.13_ALARM_1:16431* does not exist. If that is the case, the rule will start an event correlation operation with the ID *<rulefile name, rule offset in rulefile, value of desc field>* which yields *</etc/sec/ids.sec, 0, attacking host 192.168.17.13>*. The operation expects 10 events within 120 seconds as defined with *thresh* and *window* fields of the rule. After the operation has been initialized, it first creates the context *ATTACKER_192.168.17.13* (according to the *init* field). After that, the operation sets up an alias name *IP_192.168.17.13_ALARM_1:16431* for this context as defined with the *count* field. The alias will exist for 120 seconds and will prevent the rule from matching further alarms with this particular combination of attacker IP and alarm ID. The alias lifetime is controlled by the trigger context *TRIGGER_1:16431_192.168.17.13* which will expire after 120 seconds and remove the alias. After creating the context and the alias, the operation sets its event counter to 1.

When further events appear that match the first rule, the operation ID is calculated, and if the operation with the given ID does not exist, it is initialized as described above (since the operation ID contains the attacker IP, there will be a separate event counting and thresholding operation for each attacker). However, if the operation exists, it will receive the matching event and increment its event counter, and also create an alias for attacker IP and alarm ID, in order to avoid counting further alarms of same type for the given attacker within 120 seconds.

If some operation has counted 10 alarms within the last 120 seconds, this indicates the use of different attack techniques from some malicious host within a short time frame. Therefore, the operation generates the synthetic event *Multifaceted attack from attackerIP* (as defined with the *action* field of the rule), and consumes further alarms silently until the end of the event correlation window. Before terminating, the operation will delete the context *ATTACKER_attackerIP* (according to rule's *end* field) which will also destroy all alias names associated with this context, in order to avoid interference with potential further operations for the same attacker IP. Note that alias lifetime triggers don't need removal, since they take no action for non-existing aliases, and potential future recreation of the trigger will destroy any previous instance. If the operation has seen less than 10 alarms for the attacker within the 120 second window, the operation slides the window forward and continues. If no events remain in the window after sliding, the operation terminates.

Synthetic events generated by operations started by the first rule in Fig. 1 are inserted into input buffer of SEC and treated similarly to regular input events from */var/log/messages*. Therefore, these events will match the second rule in Fig. 1 which will start a separate counting and thresholding operation for each attacker IP. If an operation observes 5 events within 1800 seconds for the given attacker, it sends an e-mail warning about repeated multifaceted attacks to *root@example.com*.

### B. Advanced Event Matching with Perl Functions

Although regular expressions allow for flexible parsing of input events, they have some limitations. Firstly, apart from string recognition it is hard to implement other types of

matching, for example, arithmetic filters for numerical fields in input events. Secondly, regular expressions of different SEC rules work independently with no data sharing between them.

For instance, the ruleset in Fig. 1 assumes that the attacker IP is always found in the source IP field of the alarm. However, a number of attacks manifest themselves through specific victim responses to attackers. As a result, the destination IP address field reflects the attacker, for example:

*Oct 25 14:19:03 mysensor snort[12341]: [1:2101201:11] GPL WEB_SERVER 403 Forbidden [Classification: Attempted Information Leak] [Priority: 2] {TCP} 10.12.23.39:80 -> 192.168.11.229:52466*

Unfortunately, it is not straightforward to write a single regular expression for distinguishing external and home IP addresses in relevant alarm fields and setting match variables properly for all scenarios. In order to address complex event matching and parsing tasks, SEC allows for setting up custom Perl functions. Since user-defined code often benefits from external Perl modules, these can be loaded at SEC startup. Fig. 2 presents sample rules for improving the ruleset from Fig. 1.

```
type=Single
ptype=SubStr
pattern=SEC_STARTUP
context=SEC_INTERNAL_EVENT
desc=load Net::IP module and set $homenet
action=eval %ret (require Net::IP); \
  if %ret () else ( logonly Net::IP not found; \
                    eval %o exit(1) ); \
  lcall %ret -> \
  ( sub { $homenet = new Net::IP('10.12.23.32/29'); } )

type=EventGroup
ptype=PerlFunc
pattern=sub { if ($_[0] =~ \
/snort\[\d+\]: \[(\d+:\d+):\d+\] .*\
 \{\w+\} ([\d.]+)(?::\d+)? -> ([\d.]+)(?::\d+)?/) { \
    my $ip = new Net::IP($2); \
    if ($ip->Net::IP::overlaps($homenet) \
    == $Net::IP::IP_A_IN_B_OVERLAP) \
  { return ($1, $3); } else { return ($1, $2); } \
} return 0; }
context=!IP_$2_ALARM_$1
count=alias ATTACKER_$2 IP_$2_ALARM_$1; \
  create TRIGGER_$1_$2 120 ( unalias IP_$2_ALARM_$1 )
init=create ATTACKER_$2
end=delete ATTACKER_$2
desc=attacking host $2
action=event Multifaceted attack from $2
thresh=10
window=120
```

Fig. 2. Using a Perl function for matching and parsing Snort IDS alarms.

The first rule requires the presence of the --*intevents* option in SEC command line which forces the generation of special synthetic events at SEC startup, restarts, log rotations, and shutdown. In order to disambiguate these synthetic events from similarly looking regular input, SEC sets up a temporary context *SEC_INTERNAL_EVENT* which exists only during the processing of these events. The first rule matches the *SEC_STARTUP* event (generated at SEC startup) and loads the *Net::IP* Perl module. The rule also sets the Perl *$homenet* global variable to 10.12.23.32/29. If the module loading fails, the rule logs a relevant error message and terminates the SEC process by calling *exit(1)*. The second rule uses a Perl function for matching IDS alarms which receives the alarm message as its first parameter. The function matches each alarm with the regular expression from Fig. 1, but in addition to match variables *$1* and *$2*, match variable *$3* is set to the destination IP address. Then the *overlaps()* method from *Net::IP* module is used for checking if the source IP address belongs to the home network (represented by *$homenet* variable that was set from previous rule). If that's the case, the function returns alarm ID and destination IP, otherwise the function returns alarm ID and source IP. Outside the function, its return values are mapped to match variables *$1* and *$2*, and thus the *$2* variable always reflects the attacker IP in the rest of the rule definition.

Perl functions can not only be used as patterns for event matching and parsing, but also as additional filters in rule *context** fields. For example, the following rule fields match an SSH login failure syslog event if the connection originates from a privileged port on the client host (the port number of the client host is assigned to the *$1* match variable, and the variable is passed to a Perl function for verifying its value is smaller than 1024):

*ptype=RegExp*
*pattern=Failed [\w.-]+ for \w+ from [\d.]+ port (\d+) ssh2*
*context=$1 -> ( sub { $_[0] < 1024 } )*

In a similar way, many Perl functions can be defined for event matching and parsing which share global data structures (e.g., a hash table of malicious IP addresses). Since including longer functions in rule definitions might decrease rule readability, it is recommended to encapsulate such code into separate Perl modules and load them as depicted in Fig. 2.

### C. Using Named Match Variables and Match Caching

When creating larger SEC rulebases with hundreds of rules, a number of rules might use identical regular expression or Perl function patterns. However, significant amount of CPU time could be spent for matching an event repeatedly with the same pattern. Moreover, the use of numeric match variables (e.g., *$1* and *$2*) assumes that the number of input event fields and their nature are known in advance, but this is not always the case. Finally, variable numbering can easily change if the pattern is modified, making rules harder to maintain. In order to address aforementioned issues, SEC supports named match variables and match caching as depicted by a ruleset in Fig. 3. This ruleset processes Linux *iptables* firewall syslog events which contain a number of fieldname-value pairs, for example:

*Oct 26 11:05:22 fw1 kernel: iptables: IN=eth0 OUT= MAC=XXX SRC=192.168.94.12 DST=10.12.23.39 LEN=52 TOS=0x00 PREC=0x00 TTL=60 ID=61441 DF PROTO=TCP SPT=53125 DPT=23 WINDOW=49640 RES=0x00 SYN URGP=0*

Depending on the nature of network traffic, *iptables* events can contain a variety of different fields, and writing one regular expression for all possible field combinations is intractable. On the other hand, the Perl function in the first rule takes advantage of iterative regular expression matching, in order to parse out each fieldname-value pair and store it into a Perl hash table. Since the function returns a reference to this hash table, named match variables *$+{name}* are created from all fieldname-value pairs in the table. For example, when the

above example event is matched, *$+{SRC}* and *$+{DST}* variables are set to 192.168.94.12 and 10.12.23.39, respectively, and *$+{SYN}* is set to 1 (default when fieldname does not have a value). Therefore, the naming scheme for match variables is dynamic and fully determined by input data. After the event has been matched, the result of parsing is stored in the pattern match cache under the entry *IPTABLES* (the match caching is configured with the *varmap* field of the rule). Note that the pattern match cache is cleared before processing each new input event, and thus all cache entries always reflect parsing results for the currently processed event. Also, each cache entry is implemented as a Perl hash table which can be accessed directly from rule *context** fields (see Fig. 3).

```
type=SingleWithThreshold
ptype=PerlFunc
pattern=sub { my(%var); my($line) = $_[0]; \
    if ($line !~ /kernel: iptables:/g) { return 0; } \
    while ($line =~ /\G\s*([A-Z]+)(?:=(\S*))?/g) { \
      $var{$1} = defined($2)?$2:1; \
    } return \%var; }
varmap=IPTABLES
continue=TakeNext
desc=too many blocked packets from IP $+{SRC}
action=logonly
thresh=100
window=120

type=SingleWithThreshold
ptype=Cached
pattern=IPTABLES
context=IPTABLES :> ( sub { exists($_[0]->{"SYN"}) && \
                            exists($_[0]->{"FIN"}) } )
desc=SYN-FIN flood attempt against IP $+{DST}
action=logonly
thresh=100
window=120
```

Fig. 3. Ruleset for processing Linux iptables firewall events.

Since the *continue* field of the first rule is set to TakeNext, all matching input events are passed to the following rule for further processing. In order to save CPU time, the second rule matches incoming *iptables* events by doing a quick lookup for the *IPTABLES* entry in the pattern match cache (as specified with *ptype=Cached* and *pattern=IPTABLES*). If this entry is found, the *:>* operator in the *context* field passes a reference to the entry into a Perl function which verifies the presence of *$+{SYN}* and *$+{FIN}* variables under the entry. If both variables exist, the rule matches an event, and the *$+{DST}* variable in the *desc* field is set from the *IPTABLES* entry.

Note that named match variables and match caching are also supported for regular expression patterns – for example, the regular expression *Connection closed from (?<ip>[\d.]+)* creates match variables *$1* and *$+{ip}* which are both set to an IP address, and these variables can be cached with the *varmap* statement.

### *D. Arranging rulesets hierarchically*

Each SEC ruleset is stored in a separate text file, and rules from one file are applied to an input event in the order they have been defined in the file. Also, by default rulesets from different files are applied independently against each input event. However, if only few rulesets are relevant for most input events, the use of larger rulebases involves considerable performance penalty, since an input event will be potentially matched against many irrelevant rulesets.

SEC provides several options for addressing this problem. Firstly, if SEC has been started with the *--intcontexts* command line option, reception of any input event will trigger the creation of a temporary context that reflects the source of this event (e.g., *_FILE_EVENT_/var/log/messages*). After all rules have been applied against the input event, the context is deleted immediately. If some rules are designed to match events from specific sources only, such temporary contexts allow for preventing matching attempts for other sources. For example, the following rule fields match the regular expression with input events from */var/log/secure* only (square brackets around *_FILE_EVENT_/var/log/secure* force the check for the presence of this context *before* regular expression matching):

*ptype=RegExp*
*pattern=Connection closed from (?<ip>[\d.]+)*
*context=[ _FILE_EVENT_/var/log/secure ]*

Also, one user-defined context can be set for multiple sources. Prior to SEC-2.7.6, *_INTERNAL_EVENT* context was always used for all synthetic events, while with more recent SEC versions *cevent* and *cspawn* actions can be employed for generating synthetic events with custom contexts.

```
####################
# the content of /etc/sec/main.sec

type=Jump
context=[ _FILE_EVENT_/var/log/messages ]
ptype=PerlFunc
pattern=sub { my(%var); my($line) = $_[0]; \
    if ($line !~ /kernel: iptables:/g) { return 0; } \
    while ($line =~ /\G\s*([A-Z]+)(?:=(\S*))?/g) { \
      $var{$1} = defined($2)?$2:1; \
    } return \%var; }
varmap=IPTABLES
desc=parse and route iptables events
cfset=iptables-events

type=Jump
context=[ _FILE_EVENT_/var/log/secure ]
ptype=RegExp
pattern=sshd\[\d+\]:
desc=route sshd events from /var/log/secure
cfset=sshd-events

####################
# the content of /etc/sec/fw.sec

type=Options
procallin=no
joincfset=iptables-events

type=SingleWithThreshold
ptype=Cached
pattern=IPTABLES
desc=Too many blocked packets to IP $+{DST}
action=logonly
thresh=100
window=120

####################
# the content of /etc/sec/sshd.sec

type=Options
procallin=no
joincfset=sshd-events

…
```

Fig. 4. An example hierarchical ruleset.

Secondly, *Jump* rules can be used for submitting input events to specific rulesets for further processing, and rulesets can be configured to accept input from *Jump* rules only. Fig. 4 depicts an example for three rulesets which are arranged into two-level hierarchy.

From the three rulesets presented in Fig. 4, the ruleset from */etc/sec/main.sec* is applied for recognizing input events and submitting them to two other rulesets which are labeled as *iptables-events* and *sshd-events*. Since both rulesets contain an *Options* rule with the *procallin=no* statement, they will only accept input events from *Jump* rules. As a result, the ruleset in */etc/sec/fw.sec* is restricted to receive *iptables* syslog events from */var/log/messages* which have already been parsed by the *Jump* rule. Also, the ruleset in */etc/sec/sshd.sec* can only process SSH daemon syslog events from */var/log/secure*.

The above example illustrates that ruleset hierarchies can significantly reduce cost of event processing if many rules and rulesets are involved, especially if event parsing is accomplished in top levels of the hierarchy. In more general cases, rulesets can be arranged into graph-like structures which can introduce processing loops. Whenever SEC detects a loop during matching an event against rules, processing for the event is terminated.

## IV. Performance Data and Conclusion

We have used best practices and recommendations from the previous section in a production environment for two years. One of our SEC instances is running on a Linux server and using a hierarchically arranged rulebase of 375 rules, in order to correlate syslog events from many production servers. According to recently collected performance data for 172 days, this SEC instance has processed 1,636,805,087 events during 14,881,059 seconds (109.9 events per second), and 1,331,412,766 events have been matched by rules. During event processing, the SEC instance has consumed 448,150 seconds of CPU time on a single core of an Intel Xeon X5650 processor (about 3% of available CPU time on one core). When we briefly experimented with disabling the hierarchical rulebase arrangement, the CPU load increased 4-5 times.

Although we have reviewed a number of powerful features of SEC for creating scalable configurations, many interesting topics have been left out from this paper due to space limitations. In particular, we haven't provided in-depth discussion on individual rule types, advanced use of contexts for aggregating and reporting event data, actions for working with sockets, clock-triggered event correlation schemes, and integration with other monitoring applications. In order to get a detailed insight into those issues, the interested reader is referred to the SEC official documentation and mailing list, but also to past papers [2, 10, 11].

## Acknowledgment

The author of SEC expresses his gratitude to John P. Rouillard for many great ideas and creative discussions which have been crucial for developing SEC during the last 15 years. The authors also thank Mr. Kaido Raiend and Mr. Ain Rasva for supporting this work.

## References

[1] Risto Vaarandi, "SEC – a Lightweight Event Correlation Tool," Proceedings of the 2002 IEEE Workshop on IP Operations and Management, pp. 111-115.

[2] John P. Rouillard, "Real-time Logfile Analysis Using the Simple Event Correlator (SEC)," Proceedings of the 2004 USENIX Large Installation System Administration Conference, pp. 133-149.

[3] Jeffrey Becklehimer, Cathy Willis, Josh Lothian, Don Maxwell, and David Vasil, "Real Time Health Monitoring of the Cray XT3/XT4 Using the Simple Event Correlator (SEC)," Proceedings of the 2007 Cray User Group Conference.

[4] Ross Miller, Jason Hill, David A. Dillow, Raghul Gunasekaran, Galen Shipman, and Don Maxwell, "Monitoring Tools for Large Scale Systems," Proceedings of the 2010 Cray User Group Conference.

[5] Jason J. Hill, Dustin B. Leverman, Scott M. Koch, and David A. Dillow "Determining the health of Lustre filesystems at scale," Proceedings of the 2011 Cray User Group Conference.

[6] Byung H. Park, Thomas J. Naughton, Pratul Agarwal, David E. Bernholdt, Al Geist, and Jennifer L. Tippens, "Realization of User Level Fault Tolerant Policy Management through a Holistic Approach for Fault Correlation," Proceedings of the 2011 IEEE International Symposium on Policies for Distributed Systems and Networks, pp. 17-24.

[7] David Lang, "Building a 100K log/sec logging infrastructure," Proceedings of the 2012 USENIX Large Installation System Administration Conference, pp. 203-213.

[8] Risto Vaarandi, "Tools and Techniques for Event Log Analysis," PhD Thesis, Tallinn University of Technology, 2005.

[9] Michael R. Grimaila, Justin Myers, Robert F. Mills, and Gilbert L. Peterson, "Design and Analysis of a Dynamically Configured Log-based Distributed Event Detection Methodology," The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 01/2012; 9(3), pp. 219-241, 2012.

[10] Risto Vaarandi, "Simple Event Correlator for real-time security log monitoring," Hakin9 Magazine 1/2006 (6), pp. 28-39, 2006.

[11] Risto Vaarandi and Michael R. Grimaila, "Security Event Processing with Simple Event Correlator," Information Systems Security Association (ISSA) Journal 10(8), pp. 30-37, 2012

[12] Risto Vaarandi and Karlis Podins, "Network IDS Alert Classification with Frequent Itemset Mining and Data Clustering," Proceedings of the 2010 IEEE Conference on Network and Service Management, pp. 451-456.

[13] Risto Vaarandi, "Platform Independent Event Correlation Tool for Network Management," Proceedings of the 2002 IEEE/IFIP Network Operations and Management Symposium, pp. 907-909.

[14] Gabriel Jakobson and Mark Weissman, "Real-time telecommunication network management: Extending event correlation with temporal constraints," Proceedings of the 1995 IEEE International Symposium on Integrated Network Management, pp. 290-301.

[15] Boris Gruschke, "Integrated Event Management: Event Correlation using Dependency Graphs," Proceedings of the 1998 IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, pp. 130-141.

[16] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie, "High speed and robust event correlation," IEEE Communications Magazine 34(5), pp. 82-90, 1996

[17] M. Steinder and A. S. Sethi, "End-to-end Service Failure Diagnosis Using Belief Networks," Proceedings of the 2002 IEEE/IFIP Network Operations and Management Symposium, pp. 375-390.

[18] Stephen E. Hansen and E. Todd Atkins, "Automated System Monitoring and Notification With Swatch," Proceedings of the 1993 USENIX Large Installation System Administration Conference, pp. 145-152.

[19] http://www.crypt.gen.nz/logsurfer/

[20] http://nxlog-ce.sourceforge.net/

[21] http://esper.codehaus.org/

# Appendix 5

**Publication V**
R. Vaarandi, B. Blumbergs, and M. Kont. An Unsupervised Framework for Detecting Anomalous Messages from Syslog Log Files. In *IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, Taipei, Taiwan, April 2018. IEEE

# An Unsupervised Framework for Detecting Anomalous Messages from Syslog Log Files

Risto Vaarandi, Bernhards Blumbergs
TUT Centre for Digital Forensics and Cyber Security
Tallinn University of Technology
Tallinn, Estonia
firstname.lastname@ttu.ee

Markus Kont
Technology Branch
NATO CCDCOE
Tallinn, Estonia
firstname.lastname@ccdcoe.org

***Abstract*—System logs provide valuable information about the health status of IT systems and computer networks. Therefore, log file monitoring has been identified as an important system and network management technique. While many solutions have been developed for monitoring known log messages, the detection of previously unknown error conditions has remained a difficult problem. In this paper, we present a novel data mining based framework for detecting anomalous log messages from syslog-based system log files. We also describe the implementation and performance of the framework in a large organizational network.**

***Keywords—anomaly detection for system logs; pattern mining from log files; LogCluster***

## I. Introduction

Network faults, service failures, security incidents, and other error conditions often trigger log messages which provide detailed error information to system administrators. Therefore, automated system log monitoring for known error conditions is a widely acknowledged practice. Many existing log monitoring tools like Swatch [1] and LogSurfer [2] are rule-based and assume that a human expert defines patterns (e.g., regular expressions) for log messages that require further attention. However, this approach does not allow to identify previously unknown error conditions. For addressing this issue, various anomaly detection methods have been suggested, including hidden Markov models [3, 4], principal component analysis (PCA) [5], decision trees [6], entropy based algorithms [7, 8], support vector machines (SVM) [9, 10], neural networks [11], and logistic regression [12].

For identifying anomalous messages with rule-based log monitoring tools, system administrators often use the following approach – rules are defined for matching all known log messages that reflect normal system activity, while remaining messages are highlighted as anomalous. Since this task requires a lot of expertise, data mining tools have often been suggested for discovering event patterns from log files. However, such tools assume that human experts interpret detected knowledge and create the rules manually which is time-consuming and expensive. In this paper, we propose a novel data mining based framework for *fully automated* rule discovery for real-time detection of anomalous messages from syslog-based logs. Although the framework does not require human intervention and adapts to changes in the system, the human expert can nevertheless augment the framework with hand-written rules (e.g., our framework implementation employs rules for EWMA based alerting). The remainder of this paper is organized as follows – section II reviews related work, section III presents the framework, section IV describes its implementation and performance, and section V outlines future work.

This work has been supported by Estonian IT Academy (StudyITin.ee).

## II. Related Work

Yamanishi and Maruyama [3] have suggested an unsupervised method for network failure prediction from syslog error events. The method divides the log into time slots (sessions) and converts original events into event type symbols. Sessions are modeled with hidden Markov model mixtures, while model parameters are learned in unsupervised fashion. An anomaly score is calculated for each session, with one reported as anomalous if its score exceeds a threshold. Salfner [4] has proposed a supervised failure prediction algorithm which is based on hidden semi-Markov models and takes numerical error event type sequences for its input. The approach employs Levenshtein distance function for grouping similar events under the same event type ID. According to experiments conducted with telecommunication log data, the method compares favorably to other approaches. Xu, Huang, Fox, Patterson and Jordan [5] have suggested an unsupervised method which detects anomalous event sequences using PCA. The method employs source code analysis for detecting event types and event type sequences. Sequences are then converted into vectors, where each vector attribute reflects the number of events of some type in the sequence. During detection process, vectors containing frequently occurring patterns are filtered out, since they are highly likely to correspond to normal event sequences. For detecting anomalous sequences, PCA is applied for remaining vectors. Reidemeister, Jiang and Ward [6] have proposed a supervised method which harnesses two-stage clustering algorithm for mining event type patterns from labeled log files that contain error messages. Detected patterns are then converted into bit vectors that are used for building decision trees with the C4.5 algorithm. Finally, decision trees are employed for detecting recurrent fault conditions from log files. Oliner, Aiken and Stearley [7] have developed an unsupervised algorithm called Nodeinfo which considers events from previous *n* days for anomaly detection. Nodeinfo divides events from each network node into hourly windows (nodehours), and calculates Shannon information entropy

978-1-5386-3416-5/18/$31.00 ©2018 IEEE

based anomaly score for them. If the nodehour contains log file words that have appeared only for few nodes, the nodehour will get a high anomaly score. According to experiments on supercomputer logs, Nodeinfo performs particularly well for groups of similar nodes that produce similar log messages. Makanju, Zincir-Heywood, Milios and Latzel [8] have developed the STAD framework which also uses the concept of nodehour. STAD employs information content clustering for dividing the set of nodehours into clusters, so that nodehours containing similar alert types are assigned to the same cluster. Rule-based anomaly detection is then used for finding clusters that contain nodehours with alert messages. Kimura, Watanabe, Toyono and Ishibashi [9] have proposed a supervised fault prediction method which extracts log message templates (message types) during its first step. Extracted information is used for building log feature vectors that characterize message frequency, periodicity, burstiness, and correlation with maintenance and failures. Vectors are used for training SVM with Gaussian kernel for future fault prediction. Featherstun and Fulp [10] have suggested the use of spectrum-kernel SVM for predicting disk failures on Linux platform from syslog message sequences. The method extracts facility, severity, and specific fault message substrings from each syslog message, and converts them into numerals that are provided to SVM. The method is able to predict hard disk failures one day in advance with an accuracy of 80%. Du, Li, Zheng and Srikumar [11] have proposed the DeepLog algorithm that uses LSTM neural networks for detecting anomalies in event type sequences, predicting the probability of an event type from previous types in the sequence. In addition, DeepLog implements anomaly detection for event type parameters (such as IP addresses). The algorithm also accepts human feedback about false positives for improving its future accuracy. He, Zhu, He and Lyu [12] have compared the anomaly detection performance of logistic regression, decision tree, SVM, clustering, PCA, and invariants mining, applying the methods to event log data in numerical format. During the experiments on two publicly available data sets, supervised methods were found to be superior to unsupervised algorithms.

## III. Detecting Anomalous Event Log Messages

### A. Overview of the Framework

Existing methods described in the previous section have several drawbacks. Firstly, a number of methods are supervised and rely on labeled training data sets [4, 6, 9, 10, 11] which are expensive to produce. Also, such methods need retraining if system changes introduce new log messages. Some methods from section II assume that event logs contain error messages only [3, 4], while in production environments most messages reflect normal system activity. Some methods are designed for specific fields only like disk fault prediction [10], or rely on mining message patterns from source code [5] which might not be always available. Finally, several methods do not report individual anomalous messages, but rather entire time slots that contain such messages [7, 8].

In this section, we present an unsupervised framework for detecting anomalous messages from syslog log files that addresses aforementioned shortcomings. The framework is data mining based and relies on the following assumption – in a well-maintained IT system, most log messages reflect normal system activity, while messages corresponding to system faults, security incidents, and other error conditions appear infrequently (similar assumption has been made in other research papers, e.g., [5]). Therefore, frequently occurring message patterns naturally represent a baseline of normal system activity. Our framework has been designed for logs collected with a widely used syslog protocol [13], and it is assumed that each message has the following attributes – timestamp, sender hostname, facility (type of the sender, e.g., *daemon*), severity, program name, and free-form message text. We also assume that each log file line fully represents some syslog message. Fig. 1 depicts the implementation of the proposed framework. For detecting frequent message patterns, the framework employs the LogCluster algorithm [14] and its Perl-based implementation [15]. We have selected LogCluster, since according to our recent experiments it compares favorably to publicly available implementations of other log mining algorithms [14]. LogCluster based mining module runs daily, in order to keep the database of frequent patterns up to date with changes in the surrounding environment. Patterns from the last $N$ days and $W$ weeks (so called *mining windows*) are then used for creating Simple Event Correlator (SEC) [16] rules that match messages reflecting normal system activity. Any message which does not match these rules is classified as anomalous. Since the mining windows are sliding, the framework is able to adapt to system changes and new log message types.

The rule mining module does not attempt to discover rules for each host separately, since this involves several challenges. Firstly, some hosts do not produce many log messages which complicates the detection of frequent message patterns. Secondly, when a new host appears in the network, it is unclear what filtering rules should be applied to it. For addressing these issues, the mining is conducted for groups of similar hosts which ensures that sufficient amount of past log data is available (the Nodeinfo algorithm uses a similar approach [7]). For the sake of simplicity, the remainder of this section assumes that all hosts belong to one group.

LogCluster is a data clustering algorithm which detects line patterns from textual log file of $n$ lines for the user-given *support threshold* $s$ (*relative* support threshold $r$ means support threshold $r * n / 100$). Each log file line is split into words by user-given separator (default is whitespace). If a word appears in $l$ lines, its *support* is defined as $l$ and *relative* support as $l / n * 100$. LogCluster begins its work with a pass over the log file for finding *frequent words* – words with the support of at least $s$. During the second data pass, LogCluster splits the log file into clusters that contain at least $s$ lines. All lines from the same cluster match the same line pattern of frequent words and wildcards. Each wildcard has the form **{m,n}* and matches at least $m$ and at most $n$ words ($m \leq n$). Finally, each line pattern that represents a cluster is reported to the user, for example, *sshd: Connection from *{1,1} port *{1,1}*. As discussed in [5], a detected pattern is meaningful if its words represent constant parts of the log message, while wildcards capture variable parts (e.g., IP addresses). The support of the pattern is defined as the number of lines in the cluster it represents.

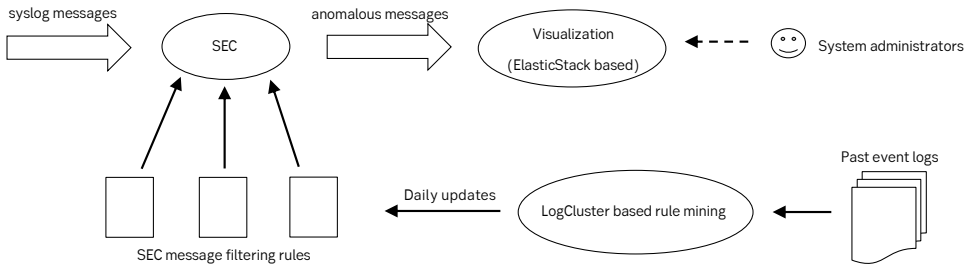



Fig. 1. A framework for detecting anomalous syslog messages.

The latest version of the LogCluster tool [15] can decouple individual phases of the algorithm which allows for parallel execution and pipelining. For example, LogCluster can be configured to detect frequent words only and store them into a *dictionary file* with absolute and relative supports. Also, frequent words can be loaded from dictionary for detecting clusters with a single data pass. Finally, clusters can be stored to *dump files*, so that repeated post-processing of detected patterns can be accomplished without re-executing the entire mining process. However, LogCluster and other support threshold based log mining algorithms have several drawbacks [14, 17] which complicate automated rule discovery. Firstly, they have a single thread of control which makes them less scalable to very large data sets. Also, with higher support thresholds *overgeneralized* patterns could be detected (e.g., **{2,5} for *{1,10}*) that can mistakenly match many anomalous messages. Furthermore, low support thresholds often lead to *overfitting*, where meaningful clusters are split into subclusters with too specific patterns (e.g., *sshd: Closing connection to 10.1.1.1*). Obviously, such patterns do not cover all messages representing normal system activity. Finally, a rare fault condition can trigger a large amount of similar messages within a short time [1], and a mining module might detect a frequent pattern from them. The rest of this section discusses how the framework addresses these challenges.

For increasing the scalability of the framework, daily rule mining is split into independent tasks which can be executed in parallel. Log file messages are first divided by syslog facility and then by program name. Fig. 2 summarizes the rule mining procedure (it assumes that days are numbered in a consecutive order). For each facility, the procedure first identifies the names of *frequent programs* that have produced at least $T_{prog}$ messages during the previous day. If a frequent program has been frequent during at least $K * N$ days in the $N$ day mining window, the framework executes pattern discovery procedure for the message text field of that program. Parameter $K$ is called *daily relevance threshold* ($0 \le K \le 1$), and setting it to higher value prevents learning filtering rules for programs with rare but intensive logging activity. The pattern discovery procedure has been summarized in Fig. 3 and will be described in subsection IIIB. Since log messages from infrequent programs might also contain long-term patterns that reflect normal system activity, a separate pattern discovery procedure with the mining window of $W$ weeks and *weekly relevance threshold* $L$ ($0 \le L \le 1$) is executed for such messages. This procedure will be discussed in subsection IIIC.

### B. Pattern Discovery from Program Message Texts

For mitigating overgeneralization and overfitting, we first attempted to find a method for selecting a single support threshold value that would generate all required meaningful patterns. We evaluated several methods, most notably head/tail breaks hierarchical clustering algorithm [18]. The algorithm iteratively divides data points with heavy-tailed distribution into head and tail by average or mean. We used variants of this algorithm for clustering words by their occurrence time (word occurrence times are known to have a heavy-tailed distribution [19]). Our aim was to identify a support threshold for capturing proper amount of frequent words for creating suitable patterns, but experiments did not yield acceptable results. Thus, the framework selects a number of support thresholds heuristically, starting from a higher value which is iteratively lowered until a stop condition evaluates true. In the framework implementation, the following relative support thresholds have been used: $s_1 = 5$, $s_2 = 2.5$, $s_3 = 1$, $s_i = s_{i-3} / 10$ for $i > 3$, so that $s_i * n / 100 > 100$ ($n$ is the number of lines in input data set).

```
find_rules(N, K, W, L, T_prog, T_weight, T_weak, T_prec)

D := # of the current day
for each f in { facilities } do
  A_f,D-1 := { names of programs that produced at least
              T_prog messages for facility f at day D-1 }
  O_f,D-1 := { log messages for facility f at day D-1 that
              did not originate from programs in A_f,D-1 }
  for each P in A_f,D-1 do
    n := |{ d | D-N ≤ d ≤ D-1, P ∈ A_f,d }|
    if n ≥ K * N then
      F_f,P := discover_patterns(P, N, K, T_weight, T_weak, T_prec)
      build filtering rules for facility f and program P,
        using patterns from F_f,P
    fi
  do
  I_f := discover long term patterns from log messages in
        O_f,D-1 ∪ … ∪ O_f,D-7*W with relevance threshold L
  build filtering rules for facility f,
    using patterns from I_f
do
```

Fig. 2. Rule mining procedure.

If $S$ denotes the set of selected support thresholds in relative notation (i.e., $\forall s \in S$, $0 < s \leq 100$) and *min* is the smallest threshold from $S$, the set of frequent words for *min* contains all frequent words for higher thresholds. Also, we have found that frequent words appearing during few days only are related to overfitting or bursts of anomalous messages. These observations have motivated the following mining procedure. First, LogCluster is used for creating a dictionary file of frequent words for support threshold *min*. After that, dictionary is used for mining patterns in fast single-pass mode with support thresholds from $S$. For each threshold $s \in S$, frequent word is selected from dictionary if it appears in dictionaries of at least $K * N$ days in the mining window, having support of at least $s$ (in Fig. 3, $s_{d,w}$ denotes relative support of word $w$ at day $d$). For easing further post-processing, patterns detected for each $s \in S$ are stored into a dump file (in Fig. 3, $B_{d,s}$ denotes patterns for support threshold $s$ and day $d$).

In the following, we describe the algorithm that selects patterns for building message classification rules, with $F$ denoting the set of selected patterns. The algorithm begins with initializing $F$ to all patterns from sets $B_{d,s}$ in the mining window. For mitigating overfitting, LogCluster supports a pattern joining heuristic which merges too specific patterns based on word weight [14]. The word weight falls to interval *(0..1]* for each word in the pattern, reflecting how strongly the word is associated with other words in this pattern. If a word weight remains below the word weight threshold, it is replaced with a wildcard and similar patterns are merged. For example, if IP addresses are weakly associated with other words in patterns *Closing connection to 10.1.1.1* and *Closing connection to 10.1.1.2*, merging produces the pattern *Closing connection to *{1,1}*. According to experiments, word weight thresholds *0.5...0.8* produce best results for mitigating overfitting [14]. The pattern joining heuristic is applied to each $B_{d,s}$ with word weight threshold $T_{weight}$, and resulting patterns $R_{d,s}$ are joined to the set of selected patterns $F$ (original patterns are not discarded at this point, since heuristic might accidentally create too generic patterns which will be pruned at further steps).

For excluding overgeneralized patterns from $F$, the framework employs several techniques. First, according to our observation, the pattern has a high likelihood of being too generic if it contains one word and is detected only for one support threshold during daily pattern mining. More formally, we call the pattern a *weak pattern* at day $d$ if it was detected only for one support threshold from log messages of day $d$. If pattern $x$ has been detected during $n$ days in the mining window and $x$ has been weak during $m$ days ($m \leq n \leq N$), $x$ is excluded from $F$ if it has one word and $m / n \geq T_{weak}$.

For measuring the degree of generality of a pattern from $F$, the framework calculates its *precision*, with an overgeneralized pattern receiving a low precision score. If pattern $x$ consists of $k$ elements (words or wildcards), precision of $i$th element $prec(x_i)$ is defined as follows: $prec(x_i) = 1$ if $x_i$ is a word; $prec(x_i) = m / n$ if $x_i$ is a wildcard **{m,n}*. For finding the precision of pattern $x$, we have used the following functions: $prec_j(x) = \sum_{i=1}^{k} prec(x_i) / l_j$, where $j=1,2,3$. Parameters $l_1$ and $l_2$ denote the maximum and minimum number of words pattern $x$ can match, respectively, while $l_3$ is defined as $max(l_2, k)$. Since $\sum_{i=1}^{k} prec(x_i) \leq l_j$ for $j=1,2,3$, then $0 < prec_j(x) \leq 1$. For example, suppose pattern $x$ is **{1,6} error *{0,4}*. Then $k = 3$, $prec(x_1) = 1/6$, $prec(x_2) = 1$ and $prec(x_3) = 0$. Also, $l_1 = 11$, $l_2 = 2$ and $l_3 = 3$, and therefore $prec_1(x) \approx 0.11$, $prec_2(x) \approx 0.58$ and $prec_3(x) \approx 0.39$. In other words, any wildcard not matching exactly one word lowers the precision score, with a wildcard matching word sequences with a wide variety of lengths having greater impact (such wildcards are called *generic wildcards*). Therefore, lower precision indicates that pattern has a generic nature, and the framework excludes a pattern from $F$ if its precision is smaller than $T_{prec}$. The $prec_3()$ function has been used for measuring precision in framework implementation, since unlike $prec_1()$, it does not penalize patterns with many words and few generic wildcards, and unlike $prec_2()$, it does not favor patterns with generic wildcards **{0,n}*.

```
discover_patterns(P, N, K, T_weight, T_weak, T_prec)

D := # of the current day
S := { support thresholds for program P at day D-1 }
W := { frequent words for program P and
       support threshold min(S) at day D-1 }

for each s in S do
  V := ∅
  for each w in W do
    n := |{d | D-N ≤ d ≤ D-1, s_d,w ≥ s}|
    if n ≥ K * N then V := V ∪ { w } fi
  done
  B_D-1,s := { message text patterns mined with LogCluster
               for program P at day D-1 with support
               threshold s and frequent words from V }
done
F := ∅
for each d in (D-N,…,D-1) do
  S := { support thresholds for program P at day d }
  C_d := ∅
  for each s in S do
    R_d,s := { merged patterns for B_d,s with word
               weight threshold set to T_weight }
    F := F ∪ B_d,s ∪ R_d,s
    C_d := C_d ∪ B_d,s ∪ R_d,s
  done
done
E := ∅
for each x in F do
  n := |{d | D-N ≤ d ≤ D-1, x ∈ C_d}|
  m := |{d | D-N ≤ d ≤ D-1, x ∈ C_d, x is weak at day d}|
  if (m / n ≥ T_weak AND x has only one word)
    then E := E ∪ { x } fi
done
F := F \ E; E := ∅
for each x in F do
  if precision(x) < T_prec then E := E ∪ { x } fi
done
F := F \ E; E := ∅
for each x in F do
  n := |{d | D-N ≤ d ≤ D-1, x ∈ C_d}|
  if n < K * N then E := E ∪ { x } fi
done
F := F \ E; E := ∅
for each x in F do
  if (∃y ∈ F, x ≠ y, x ~ y) then E := E ∪ { x } fi
done
return F \ E
```

Fig. 3. Pattern discovery procedure for message text field of a program.

After that, the pattern selection algorithm excludes all patterns from $F$ that have been detected for less than $K * N$ days in the mining window. This step will ensure that only

repeatedly occurring frequent patterns will be kept in $F$, and learning filtering rules from accidental bursts of anomalous messages is avoided. Also, if pattern $y$ matches all events that are matched by pattern $x$, we say that $y$ is more general than $x$ and denote it as $x \sim y$. During its final step, pattern selection procedure excludes all patterns from $F$ that have more general patterns in $F$, since excluded patterns are redundant for deriving filtering rules.

```
# A rule for suppressing messages for successful
# SSH logins

type=Suppress
ptype=RegExp
pattern=sshd(?:\[\d+\])?: Accepted(?:\s+)(?:(?:\S+)
(?:\s+)){1,1}for(?:\s+)(?:(?:\S+)(?:\s+)){1,1}from
(?:\s+)(?:(?:\S+)(?:\s+)){1,1}port(?:\s+)(?:(?:\S+)
(?:(?:\s+)|(?:\s+)?$)){1,1}(?:\s+)?$
desc=Accepted *{1,1} for *{1,1} from *{1,1} port *{1,1}

# A rule for suppressing SNMP daemon messages for
# incoming queries

type=Suppress
ptype=RegExp
pattern=snmpd(?:\[\d+\])?: Connection(?:\s+)from(?:\s+)
UDP\:(?:\s+)(?:(?:\S+)(?:(?:\s+)|(?:\s+)?$)){1,1}
(?:\s+)?$
desc=Connection from UDP: *{1,1}

# A rule for suppressing anacron messages for upcoming
# job executions

type=Suppress
ptype=RegExp
pattern=anacron(?:\[\d+\])?: Will(?:\s+)run(?:\s+)job
(?:\s+)(?:(?:\S+)(?:\s+)){1,1}in(?:\s+)(?:(?:\S+)
(?:\s+)){1,1}min\.(?:\s+)?$
desc=Will run job *{1,1} in *{1,1} min.
```

Fig. 4. Sample automatically created SEC rules for matching messages that reflect normal system activity.

### C. Building Filtering Rules

Discovery of long-term frequent message patterns for infrequent programs is conducted similarly to the algorithm in Fig. 3, except that the mining window size is $W$ weeks and the window is divided into weekly slots. For example, if $W = 4$ and weekly relevance threshold $L = 0.75$, patterns are mined from the logs of last 4 weeks, and patterns detected for less than 3 weekly slots are dropped. Also, LogCluster is executed for messages of *all* infrequent programs of a given syslog facility, and patterns are mined from the concatenation of program name and message text fields. Finally, an additional heuristic is used for excluding overgeneralized patterns – if the pattern has a wildcard for the program name or entire message text (e.g., *sshd: *{1,20}* or **{1,1}: daemon stopped*), the pattern will be excluded from further consideration.

After discovering patterns for frequent and infrequent programs (sets $F_{f,P}$ and $I_f$ from Fig. 2, respectively), the rule mining module derives regular expressions from detected patterns and uses them for creating SEC *Suppress* rules. Fig. 4 depicts some automatically created rules for *sshd*, *snmpd*, and *anacron* that originate from our framework implementation. For efficient processing of incoming messages, the SEC rule base is arranged hierarchically [16] by facility and program name. The rule base also contains dedicated rule files for custom message processing rules written by human experts.

## IV. Framework Implementation and Performance

For measuring the performance of the framework, we have evaluated its implementation during 3 months (92 days) in a large EU organization. The framework was running with the following parameters: $N = 10$, $K = 0.5$, $W = 4$, $L = 0.75$, $T_{prog} = 1000$, $T_{weight} = T_{weak} = T_{prec} = 0.5$. During evaluation, the framework processed OS level syslog messages for *auth*, *authpriv*, *cron*, *daemon*, *kern*, and *mail* facilities from 543 Linux servers with standardized OS configuration. Since the servers generated similar log messages for aforementioned six syslog facilities, we used one host group for them. Altogether, 296,699,550 messages were processed by the framework, with 1,879,209 messages (≈0.63%) passing SEC filtering rules and classified as anomalous. During 92 days, 483-551 SEC rules were automatically created by daily rule mining procedure (412-469 rules were created for frequent programs). Table I depicts message classification data for different facilities.

TABLE I. Message Classification by Syslog Facility.

| Facility | # of all messages | # of anomalous messages | # of servers generating anomalous messages |
|---|---|---|---|
| auth | 3,672,497 | 34,904 | 224 |
| authpriv | 59,667,935 | 234,645 | 465 |
| cron | 22,249,354 | 9,000 | 200 |
| daemon | 198,129,740 | 494,545 | 508 |
| kern | 8,466,606 | 907,186 | 342 |
| mail | 4,513,418 | 198,929 | 153 |

Many anomalous messages represented previously unknown fault conditions (e.g., disk issues) that would have remained unnoticed in organizational monitoring system. As can be seen from Table I, unusually large fraction (10.7%) of kernel messages with *kern* facility were classified as anomalous, and they constituted almost half (48.3%) of all anomalous messages. 54.2% of kernel messages were various SELinux warnings, and 22.6% were generated during system reboots (most reboots were part of regular system maintenance, although reboots can also be triggered by system crashes, e.g., executed by hypervisors after virtualization system failures). From remaining kernel messages, most represented serious system errors like file system corruption and out-of-memory conditions. Majority of these error conditions triggered hundreds or thousands messages – for example, during one system crash 115,197 messages were logged (12.7% of all anomalous kernel messages). We also observed similar error message bursts for other facilities – for example, 75.8% of anomalous messages for *mail* facility were triggered by file-system-full condition on a single server. Anomalous messages for *daemon* facility represented various fault conditions or configuration errors, like temporary network outages or connectivity issues with remote services, failures to restart a daemon due to a syntax error in the configuration file, etc. Most anomalous messages for *auth* and *authpriv* facilities reflected authentication errors, while for *cron* facility execution

errors for scheduled jobs were reported. We also discovered that some anomalous messages represented rare but normal system activity. Such activities included creation of new user accounts (*authpriv* facility), modification of scheduled jobs (*cron* facility), and system service restarts (*daemon* facility).

For evaluating the classification precision and recall of the framework, we reviewed classification results for log messages that were triggered by vulnerability scanning of the internal network during four non-contiguous days. 1454 log messages were triggered on 231 servers, with 779 messages representing normal system activity (e.g., incoming connection to an SSH server) and 675 messages reflecting malicious actions (e.g., SSH password probing for non-existing user accounts). The framework classified 683 messages as anomalous, while 656 of them were malicious messages, yielding the precision of 96.0% and recall of 97.1%. During another 10 hour experiment, we used the Bbuzz protocol fuzzing framework [20] for pen-testing Linux kernel and UDP services of a test server that was connected to the framework. Protocol fuzzing triggered 46,370 and 7,957 error messages from *snmpd* and kernel, respectively, and all messages were classified as anomalous.

After detection, the framework sends anomalous messages to ElasticStack for searching and visualization purposes. Although ElasticStack provides system administrators with an efficient interface for investigating anomalies and fault conditions, it does not allow for distinguishing critical errors (e.g., a system crash) from events of lower priority (e.g., an accidental login failure). As discussed before, critical faults often trigger a large number of anomalous messages within a short time frame. For identifying such faults in a timely fashion, we have added SEC post-processing rules into the framework for EWMA based anomaly detection. With this approach, moving average $\mu_t$ of a time series $\{X_1, X_2, ...\}$ is calculated as $\mu_1 = X_1$, and $\mu_t = \alpha * X_t + (1 - \alpha) * \mu_{t-1}$ for $t > 1$. If the value of $\alpha$ is close to 1, only recent values of a time series influence $\mu_t$, while values close to 0 distribute weight more evenly. In our setup, the stream of anomalous messages is divided into 5 minute time slots for each *(host, facility)* tuple, and moving average $\mu_t$ and standard deviation $\sigma_t$ are calculated for the number of messages (i.e., $X_t$ is the number of messages in time slot $t$). The framework raises an alarm for time slot $t$ and *(host, facility)* tuple if $|X_t - \mu_t| > m * \sigma_t$ and $X_t \geq n$ (i.e., a burst of at least $n$ anomalous messages has been observed during the last 5 minutes, where the number of messages is more than $m$ standard deviations away from average). For post-processing anomalous messages, we have used the following settings: $\alpha = 0.05$, $m = 3$ and $n = 100$. During 92 day experiment, 214 alarms were generated, with more than half of them associated with system reboots. Alarms were also triggered by out-of-memory and file-system-full conditions, file system corruption events, and other major system faults.

## V. Future Work

As for future work, we plan to augment our framework with additional time series analysis methods. Our other research goals include further development of the pattern selection algorithm for automated classification, and studying methods for assigning anomaly scores to individual anomalous messages.


## Acknowledgment

The authors express their gratitude to Dr. Rain Ottis and Prof. Olaf Maennel for supporting this work.


## References


[1] Stephen E. Hansen and E. Todd Atkins, "Automated System Monitoring and Notification With Swatch," in Proceedings of 1993 USENIX Large Installation System Administration Conference, pp. 145-152.

[2] http://logsurfer.sourceforge.net

[3] Kenji Yamanishi and Yuko Maruyama, "Dynamic Syslog Mining for Network Failure Monitoring," in Proceedings of 2005 ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 499-508.

[4] Felix Salfner, "Event-based Failure Prediction: An Extended Hidden Markov Model Approach," PhD Thesis, Humboldt-Universität zu Berlin, 2008.

[5] Wei Xu, Ling Huang, Armando Fox, David Patterson and Michael I. Jordan, "Detecting Large-Scale System Problems by Mining Console Logs," in Proceedings of 2010 International Conference on Machine Learning, pp. 37-46.

[6] Thomas Reidemeister, Miao Jiang and Paul A.S. Ward, "Mining Unstructured Log Files for Recurrent Fault Diagnosis," in Proceedings of 2011 IEEE/IFIP International Symposium on Integrated Network Management, pp. 377-384.

[7] Adam Oliner, Alex Aiken and Jon Stearley, "Alert Detection in System Logs," in Proceedings of 2008 IEEE International Conference on Data Mining, pp. 959-964.

[8] Adetokunbo Makanju, A. Nur Zincir-Heywood, Evangelos E. Milios and Markus Latzel, "Spatio-Temporal Decomposition, Clustering and Identification for Alert Detection in System Logs," in Proceedings of 2012 ACM Symposium on Applied Computing, pp. 621-628.

[9] Tatsuaki Kimura, Akio Watanabe, Tsuyoshi Toyono and Keisuke Ishibashi, "Proactive Failure Detection Learning Generation Patterns of Large-scale Network Logs," in Proceedings of 2015 International Conference on Network and Service Management, pp. 8-14.

[10] R. Wesley Featherstun and Errin W. Fulp, "Using Syslog Message Sequences for Predicting Disk Failures," in Proceedings of 2010 USENIX Large Installation System Administration Conference.

[11] Min Du, Feifei Li, Guineng Zheng and Vivek Srikumar, "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning," in Proceedings of 2017 ACM Conference on Computer and Communications Security, pp. 1285-1298.

[12] Shilin He, Jieming Zhu, Pinjia He and Michael R. Lyu, "Experience Report: System Log Analysis for Anomaly Detection," in Proceedings of 2016 IEEE International Symposium on Software Reliability Engineering, pp. 207-218.

[13] C. Lonvick, "The BSD syslog Protocol," RFC 3164, 2001.

[14] Risto Vaarandi and Mauno Pihelgas, "LogCluster – A Data Clustering and Pattern Mining Algorithm for Event Logs," in Proceedings of 2015 International Conference on Network and Service Management, pp. 1-7.

[15] https://ristov.github.io/logcluster/

[16] Risto Vaarandi, Bernhards Blumbergs and Emin Çalışkan, "Simple Event Correlator – Best Practices for Creating Scalable Configurations," in Proceedings of 2015 IEEE CogSIMA Conference, pp. 96-100.

[17] Adetokunbo Makanju, "Exploring Event Log Analysis With Minimum Apriori Information," PhD Thesis, University of Dalhousie, 2012.

[18] Bin Jiang, "Head/tail Breaks: A New Classification Scheme for Data with a Heavy-tailed Distribution," The Professional Geographer, 65(3), pp. 482-494.

[19] Risto Vaarandi, "A Data Clustering Algorithm for Mining Patterns From Event Logs," in Proceedings of 2003 IEEE Workshop on IP Operations and Management, pp. 119-126.

[20] Bernhards Blumbergs and Risto Vaarandi, "Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis," in Proceedings of 2017 IEEE Military Communications Conference, pp. 707-712.

# Appendix 6

**Publication VI**
A. Fárar, H. Bahsi, and B. Blumbergs. A Case Study About the Use and Evaluation of Cyber Deceptive Methods Against Highly Targeted Attacks. In *Proceedings of Cyber Incident 2017*, pages 1–7, London, UK, June 2017. IEEE

# A Case Study About the Use and Evaluation of Cyber Deceptive Methods Against Highly Targeted Attacks

Alexandria Farar and Hayretdin Bahşi
Department of Computer Science
Tallinn University of Technology
Tallinn, Estonia

Bernhards Blumbergs
Technology Branch
NATO CCDCOE
Tallinn, Estonia

***Abstract*—Traditional defences such as intrusion detection systems, firewalls and antivirus software are not enough to prevent security breaches caused by highly targeted cyber threats. As many of these attacks go undetected, this paper shows the results of a case study which consists of implementation of a methodology that selects, maps, deploys, tests and monitors the deceptions for the purpose of early detection. Metrics are developed to validate the effectiveness of the deception implementation. Firstly, various deception mechanisms are mapped to the first three phases of the intrusion kill chain: reconnaissance, weaponization and delivery. Then, Red Teams were recruited to test the deceptions for two case scenarios. Applying metrics, it is shown that the deceptions in the case studies are effective in the detection of cyber threats before the target asset was exploited and successful in creating attacker confusion and uncertainty about the organization's network topology, services and resources.**

***Keywords— deception; honeypots; highly targeted attack; cyber kill chain;***

## I. INTRODUCTION

Many cyber espionage and cyber crime incidents occur due to the highly targeted attacks that have been conducted against organizations as well as individuals [1]. Although an advanced persistent threat similar to Stuxnet can be perceived as an extreme case in a wide spectrum of highly targeted attacks, the common denominator of these attacks is the requirement of a sophisticated level of expertise and substantial resources. In order to accomplish their mission, attackers use a meticulous approach when planning and implementing a targeted attack. Objectives usually entail establishing a foothold within the information technology infrastructure of the targeted organization, with a primary end goal of data exfiltration; other possible aims include attacks against data integrity or availability of critical production systems [2]. Highly targeted attacks attempt to achieve their goals via multiple stages as characterized by the Intrusion Kill Chain model. This model was developed for advanced persistent threats [3], however, a similar model can characterize the targeted attacks [4]. The first three stages of the model are reconnaissance, weaponization and delivery. Reconnaissance stage covers the activities of attackers for gathering information about the target. The attackers prepare the attack payload in the weaponization stage, then transmit it to the target in the delivery stage.

On the defensive side, strong perimeter protection and traditional intrusion detection systems alone are not enough to deal with the targeted attacks as they can be bypassed through the use of advanced attack methods [4]. False positive results and lack of sufficient log management are other important obstacles that degrade the strength of protection in complex information system environments. Thus, many organizations do not realize that their network has been compromised until weeks, months or even years later. Using those systems is a good defense-in-depth strategy, however, there are still gaps that may be minimized by implementing non-traditional security defense measures such as deception, an active defense, which is designed to trick or confuse the attacker [5]. Deception systems can act as an important complementary component to existing protection mechanisms with the aim of attack detection. These systems can lure the attackers to them in order to create focus points for detection systems. Normal users should not access deceptions and any interaction with them is considered a violation – thus reducing the frequency of false positives as regularly experienced with traditional tools. Additionally, utilization of appropriate deception instruments at each stage of the intrusion kill chain may decrease the false negative detection ratio.

The research that predicates this paper is based on the case studies where appropriate deception mechanisms are mapped to the first three phases of the intrusion kill chain: reconnaissance, weaponization and delivery. This method is chosen because it is imperative that highly targeted attacks be detected at the earliest possible stages in the kill chain – effectively breaking the chain before the target asset is exploited. Additionally, a Red Team Engagement Plan which details the penetration test activity against the target systems is developed and executed to test the

effectiveness of the deceptions, along with two metrics, Dwell Time which is the detection time of the threats and Attacker Deception-Perception Survey that evaluates the perceptions of penetration testers who take part in the execution of engagement plan. The threat model assumes that the attackers are considerably sophisticated and substantially resourced which can be enough to cover an average level of highly targeted attack. Thus, advanced persistent threats which are mostly conducted by state-sponsored actors are out of the scope. The main contribution of this study is the case studies implemented on a cyber range facility and evaluation of deceptions in terms of detection time and attacker perceptions.

## II. Related Works

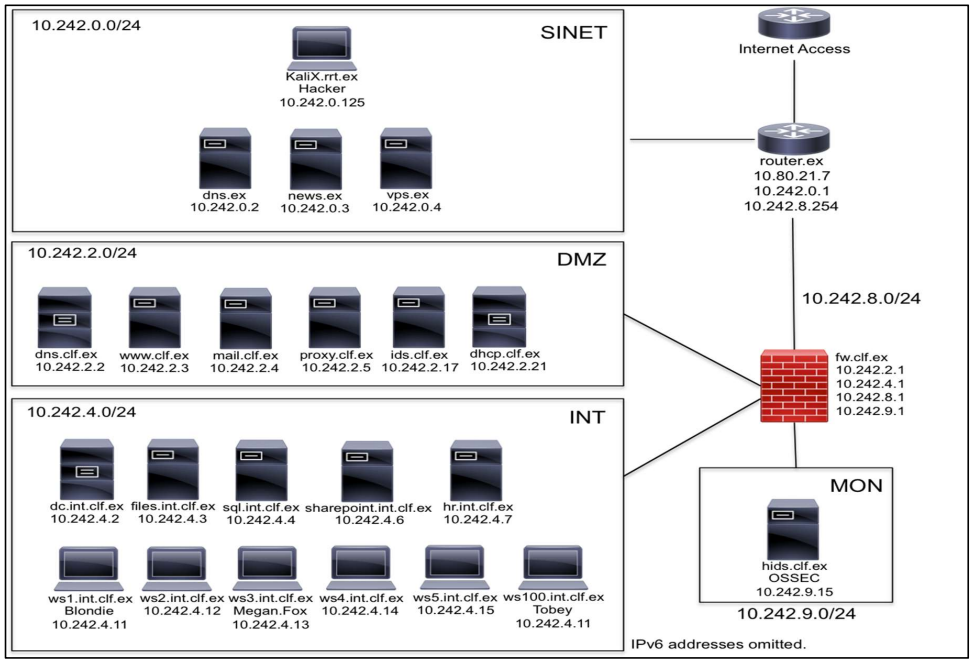
Although there is a multitude of research relating to deception mechanisms, most are focused on only one type of deception, such as a fake network topology [6], a defense against Reconnaissance; or mimicking a web site, a defense against Delivery [7]. Studies show that as the number of deception mechanisms deployed on a network increases, the likelihood of detection also increases [7] [2]. Mapping of existing cyber deception techniques to the different phases of intrusion kill chain is discussed in the studies of Heckman et al. and Briskin et al. [8] [9]. These studies also provide guidance for the planning, preparation and execution of deception systems. However, they lack practical implementations and do not provide a validation method for the effectiveness of deception design. Two fictional case studies about Stuxnet and an APT espionage campaign are given in order to explore the operational aspects of offensive and defensive deception techniques [8].

Wang et al. first introduced the notion of multilayer deception in [10]. Similar to the intrusion kill chain, they modelled a multi-stage attack with three layers of penetration: a human layer (employee information), local asset layer (employee's local machine) and a global asset layer (shared server assets), with deceptions being mapped at these three layers. Additionally, Wang et al. formulated an optimization model that chooses the best location of deceptive components at human and local asset layers and minimizes the total loss in case of an attack. However, this model does not cover all deception layers and the study lacks practical implementation of the proposed system. The idea of early detection of cyber attacks using deception was also demonstrated by Almeshekah et al. in [11], where they used the intrusion kill chain as a framework to show the effectiveness of mapping deception mechanisms at multiple levels in the chain. This paper was mainly theoretical in nature, with no experiment having been performed or metrics developed to test the effectiveness of the deceptions. However, in a dissertation by Almeshekah, in lieu of traditional honeypot scheme, he introduced a centralized deceptive fake server called Deceptiver [12]. The server hooks into a company's internet facing servers and injects deceit when it detects malicious interaction, thus creating a fake view of an organization's resources to either confuse and/or lead attackers astray. Deceptiver was a proof of concept prototype, and in the implementation it was integrated with an Apache Web Server to test it. However, they only measured the performance of the integration of the web server with Deceptiver, as opposed to the actual effectiveness of the deception itself.

## III. Methodology

The experiment was implemented by applying a systematic approach to deploying deception for the early detection of advanced cyber threats. The deceptive methodology entails first selecting the evaluation environment and designing a network topology diagram to model the network infrastructure. Relevant environment can be a cyber range facility where virtual environment is utilized for cyber security trainings or operational environment where the actual business or mission processes take place. Next, an attacker profile is created by defining a Threat Model and Threat Scenario, followed by the selection, mapping and deployment of deceptions, based on the Intrusion Kill Chain. A Penetration testing scheme is also developed, which outlines a strategic deception test plan, and a Red Team Engagement plan is executed. The attacks on the deceptions are monitored and finally, the effectiveness of the deceptions is measured and validated through metrics.

**Figure 1 Network Topology Diagram**




### *A. Evaluation Environment and Network Infrastructure*

The experiment is conducted at the NATO CCDOE Cyber Range facility in Tallinn, Estonia. It is an experimental environment used for Red Team exercises. The virtual environment was hosted on the VMWare ESXi 6.0 virtualization platform.

The devices set up and configured for the exercise experiment to make up the network are represented in a network topology diagram as shown in Figure 1. The overall network consisted of the Internal (INT), demilitarized zone (DMZ), simulated Internet (SINET) and monitor (MON) networks. Servers and workstations were both Windows and Linux-based. Two routers, a DHCP server, IDS, HIDS and a single firewall were also configured.

### *B. Threat Model and Threat Scenario*

The model is based on highly skilled attackers who have substantial resources for the accomplishment of the mission. Advanced persistent threats which require very advanced technical capabilities and huge amount of resources are out of scope. The attackers are assumed to have skills in conducting spear phishing campaigns, compromising the known vulnerabilities and achieving lateral movements between different network segments. The motivation of the attackers is obtaining valuable information which can be used for monetary gain or espionage purposes.

In the threat scenario for this experiment the target is Company Z, a research firm that sells zero-day vulnerabilities to governments, with the average flaw going for $45,000-180,000. Thus, the firm is susceptible to highly targeted cyber threats. Company Z stores a catalog of zero-day exploits and their high profile client list on a file server located in the internal network.

### *C. Penetration Testing Schema*

The traditional goal of penetration testing is to identify the exploits and vulnerabilities that exist within an organization's IT infrastructure and to help confirm the effectiveness of the security measures that have been implemented [13]. In this experiment, the penetration test is designed to test how effective the deceptions are in detecting the cyber threats according to intrusion kill chain steps.

Two professional penetration testers (RedTeam1, RedTeam2) are recruited with each having at least three years of experience. They have been involved in several professional penetration test projects. The duration for the penetration test is three days due to limited availability of testers. Black box penetration test methodology is selected where the adversaries have no knowledge of the network. However, passive reconnaissance information was provided due to the limited time of three days to complete attacker goals.

The Penetration Testing Scheme consists of four parts to include a Red Team Exercise Briefing, Red Team Rules of Engagement, Red Team Diary (RTD) and Red Team Exercise Debriefing. The Red Team Exercise Briefing details key aspects of the exercise. It is provided to RT participants and includes the dates of execution, exercise objectives, exercise outcomes, type

of exercise, exercise environment, and simplified threat model and threat scenarios to preserve results integrity in the case of the black box test. In the Red Team Rules of Engagement, the Red Team is provided general guidelines on how to conduct the penetration test. It consists of attack time limitations (i.e. 3 days), reporting requirements, type of penetration test and general attack guidelines. The Red Team Diary consists of a daily log that the penetration tester uses to document activities performed on the network. It includes such information as timestamps, source IPs, IPs of machines compromised, exploits executed on machines and other details. After the Red Team exercise is complete and the Red Team Diary has been reviewed, a debriefing takes place. The Red Team Debriefing is an interview that takes place between the exercise leader and the Red Team participants in order to ask direct questions regarding the tools, tactics and techniques used as well as why they made the decisions that they made during the attack.

*D.* Deployment of Deceptions and Exercise Execution

Deception mechanisms selected are T-Pot, Spam Honeypot with Intelligent Virtual Analyzer (SHIVA), Yet Another Low Interaction Honeypot (YALIH), KFSensor and Active Defense Harbinger Distribution (ADHD) [14][15][16][17]. All solutions are free and open source except for KFSensor.

In the case of reconnaissance, T-pot, a honeynet is chosen to deceive the attacker regarding the topology and contents of the target organization's network. It also defeats the weaponization phase of the intrusion kill chain, causing the attacker to develop exploits that are ineffectual, as he will fashion them based on a false network topology and non-existent services. Portspoof and KFSensor are also selected to further create a fake topology by emulating services that are non-existence on the network. In particular, Portspoof has the ability to slow down reconnaissance that uses port scanning, while KFSensor implements service emulation and has a built in IDS engine that captures these attacks in real time.

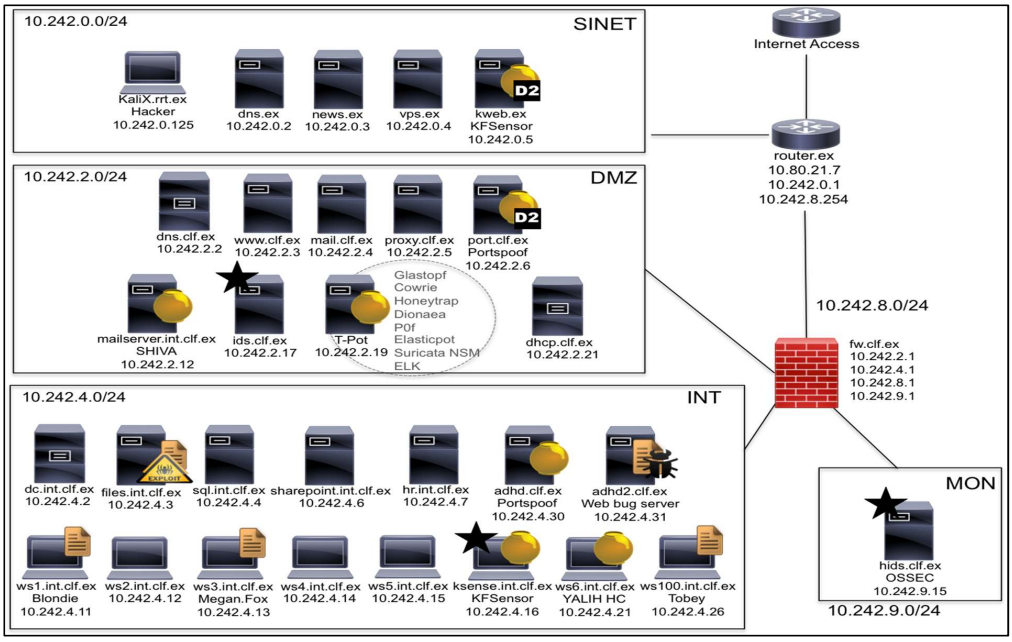
For the delivery phase, T-Pot, YALIH and SHIVA were mapped. T-pot contains vulnerable web (Glastopf), SSH (Cowrie) and malware (Dionaea) server honeypots that the attacker may interact with and be detected. SHIVA is a high interaction SPAM / Open Relay honeypot that analyses SPAM and acts as an Open Relay. YALIH is a honeyclient that retrieves email attachments and URLs and scans them to assess if they are malicious or not. In this experiment, the YALIH email honeyclient was configured to retrieve the user Blondie's email for analysis.

As depicted in Figure 2, two case scenarios are implemented: Deployment1 (D1) and Deployment2 (D2). The deployments are identical, except in the case of D2 where two additional deceptions are added for enhanced deception in-depth.

D1 deceptions are placed in the DMZ and the Internal Network. T-Pot and SHIVA are placed in the DMZ. KFSensor, YALIH honeyclient and ADHD (Web Bug Server and Portspoof) active defenses are placed in the Internal network. The honeytokens were strategically placed in the Documents directory and/or Desktop of three Windows 7 workstations: ws1(10.242.4.11) ws3(10.242.4.13) and ws100(10.242.4.26). Additionally, Honeydocs were placed on the files server, in and around the directory containing the real client list and zero day exploits. In D2, to add more complexity to the network and increase number of deceptions, ADHD Portspoof was also placed in the DMZ, port.clf.ex (10.242.2.6). KFSensor, was added to the SINET, kweb.ex (10.242.0.5). This was done in an effort to detect the ACT before it reaches the DMZ, and confuse the attacker earlier in the attack chain. Portspoof was deployed in the DMZ, further adding to network complexity perception.

The monitoring devices were located in the DMZ (Suricata/T-Pot), INT (KFSensor/IDS) and MON (OSSEC/HIDS) [18]. Log collection and analysis was done both manually and using the Elastic stack (ELK) [19]. RedTeam1 and RedTeam2 were assigned to attack the network for D1 and D2 respectively.

**Figure 2. D1 & D2 Deployments**




The Red Team exercise schedule indicates that each Red Team had three days to accomplish the mission of stealing Company Z's zero-day exploits and the high profile government client list. RedTeam1 and RedTeam2 executed their attacks between May 4th-6th 2016 and May 16th-18th respectively.

*E. Evaluation Metrics*

The two metrics used in the experiment were Dwell Time (DT) and the Deception-Perception Survey. Dwell Time measures how long the adversary is inside your network prior to being detected and the Attacker Deception Perception Survey measures the success of the deceptions in creating confusion and uncertainty on the part of the hacker [20].

Dwell Time is measured by using forensic data (i.e. logs, netflow or pcaps) to trace threats back to their origin (IP Address). In this experiment, Dwell Time is calculated by subtracting the Attack Start Time (AST) from the Time Attack Detected (TAD). These measurements (timestamps) were derived from the RTD and conducting forensic analysis of the captured data (honeypot logs) using the Elastic Stack (ELK) for T-Pot, and manual log analysis for the standalone deceptions.

The Time to Detection (TTD) specifies the maximum amount of time that the attack can remain undetected; and is selected purely based on perceived risk tolerance. If the DT is within the TTD, then the deception is effective. In this scenario, the risk tolerance is low; therefore TTD is set at less than or equal to 60 minutes, and may be adjusted as needed. The Time for Mission Execution (TME) is three days. TME describes thenumber of days allowed for the attacker to accomplish the mission.

The Attacker Deception-Perception measurement is derived from the Red Team Diary Debriefing, and is based on the Likert-type Scale to measure attacker perceptions [21]. Figure 3 gives the list of questions asked to penetration testers.

**Figure 3. Attacker Deception-Perception Survey**

| Attacker Deception-Perception Survey |
|---|
| What was your overall perception of the network, as far as level of difficulty in navigation?<br>☐ 1-Extremely Not Complex ☐ 2-Not Complex ☐ 3- Neutral ☐ 4-Complex ☐ 5-Extremely Complex |
| |
| 1. How likely is it that the machines were decoys and not real?<br>☐ 1-Extremely Unlikely ☐ 2-Unlikely ☐ 3- Neutral ☐ 4-Likely ☐ 5-Extremely Likely |
| 2. How likely is it that you were confused about identifying services or resources?<br>☐ 1-Extremely Unlikely ☐ 2-Unlikely ☐ 3- Neutral ☐ 4-Likely ☐ 5-Extremely Likely |
| 3. How likely is it that you were interacting with honeypots?<br>☐ 1-Extremely Unlikely ☐ 2-Unlikely ☐ 3- Neutral ☐ 4-Likely ☐ 5-Extremely Likely |
| 4. How likely is it that you became frustrated as a result of the complexity of the network, and not being able to locate the client list and exploits?<br>☐ 1-Extremely Unlikely ☐ 2-Unlikely ☐ 3- Neutral ☐ 4-Likely ☐ 5-Extremely Likely |
| 5. How likely is it that your failure to complete the mission due to confusion about the network topology?<br>☐ 1-Extremely Unlikely ☐ 2-Unlikely ☐ 3- Neutral ☐ 4-Likely ☐ 5-Extremely Likely |

The Debriefing consists of two sessions: direct, open-ended questions that the exercise leader asks of the Red Team participants and an Attacker Deception-Perception Survey. The open-ended questions asked are formulated based on the analysis of the Red Team Diary, and are geared toward the attacker's perception of the network, and why certain actions

were taken; but also gives insight into what tools the attacker used and the motivation behind it. The Attacker Deception-Perception Survey makes an assessment of the attacker's view of network complexity and effectiveness of deceptions.

For D1, RedTeam1 successfully exploited and compromised many vulnerable systems, however, the Nmap scanning activity was detected by T-Pot (p0F), and subsequently by the T-Pot (Glastopf) web server honeypot. Table 1 summarizes the detection times and dates of D1. Additionally, almost all of the systems in the SINET and DMZ were exploited. The internal network was breached and the sql.int.clf.ex server was exploited, however, the other servers, workstations and/or deceptions were not. The allowable time to detection is less than or equal to 60 minutes. P0f detected the Nmap scan within seven minutes, while the Glastopf web server honeypot detected the attack at the 15 minute mark.

In D2, RedTeam2 was successful in executing many exploits and conducting ARP spoofs on the DMZ and INT networks. However, the ARP spoofs failed to produce any worthwhile information. The INT was not penetrated before the end of the exercise. Emails sent to user Blondie were not related to the scenario, and although the YALIH honeyclient retrieved the emails and scanned the URL that was delivered by the attacker, neither the suspicious email attachment nor URL was flagged as malicious. As shown in Table 2, the attack began on the network using an Nmap scan on May 16th at 21:00 and was detected by T-Pot's P0f, Dionaea and Glastopf at 21:30. Honeytrap detected the attacker at 00:51 on May 17th, when the attacker probed port 25. KFSensor (kweb) also logged attacker activity on May18, including both source IP addresses the attacker used: 10.242.0.125 and 10.80.100.89 (own machine). Both T-Pot and KFSensor logged over hundreds of attempts by the attacker to find vulnerabilities and exploit them. However, they are too numerous to list in this paper. The allowable time to detection is less than or equal to 60 minutes.

The results of the Attacker Deception-Perception Survey for both deployments revealed that overall, the attackers felt that the network was complex, and were confused at times regarding the identification of services or resources. They felt it was unlikely that the machines were honeypots, although possible, due to unexpected responses to the network probe. Even though the attackers were often times confused and was not confident about network topology, the inability to reach to the target information was stated as "not enough time."

**Table 1. RedTeam1 Attack Timeline**

| Date | Attack Start Time (AST) | Time Attack Detected (TAD) | Deception | Dwell Time (MIN) | IP | Comments |
|---|---|---|---|---|---|---|
| May 4 | 13:40 | 13:47 | T-Pot | 7 | 10.242.2.19 | P0f – Nmap SYN scan |
| | | 13:55 | | 15 | | Glastopf – POST Request |
| | | | | | | |
| | | 16:55 | | - | | Suricata – Port 80 - SQLi |

**Table 2. RedTeam2 Attack Timeline**

| Date | Attack Start Time (AST) | Time Attack Detected (TAD) | Deception | Dwell Time (MIN) | IP | Comments |
|---|---|---|---|---|---|---|
| May 16 | 21:00 | 21:30 | T-Pot | 30 | 10.242.2.19 | P0f; Nmap Scan Port:443, Dionaea; Port 21 |
| | | 21:31 | T-Pot | 31 | 10.242.2.19 | Glastopf; Port 80 |
| May 17 | | 00:51<br>01:08 | T-Pot<br>T-Pot | -<br>- | 10.242.2.19<br>10.242.2.19 | Honeytrap; Port 25 Port 587 |
| May 18 | | 01:56 | KFSensor | - | 10.242.0.5 | ICMP ECHO REQ ARP Spoof 10.242.0.1 |
| | | 2:11 | KFSensor | - | 10.242.0.5 | NBT SMB SYN Scan Port 445 |

## IV. Conclusion and Future Work

This research presents case studies for the implementation and evaluation of deceptive methodology that selects, maps, deploys, tests and monitors various deceptions against highly targeted attacks. The relevant deceptions address the detection of attacks at the reconnaissance, weaponization and delivery stages of intrusion kill chain. Recruited Red Teams attacked the systems after the deployment of deception systems. Additionally, overall system is equipped with many system monitoring tools. In order to evaluate the effectiveness of the deceptions, two metrics, Dwell Time and the Attacker Deception-Perception Survey, are utilized. Dwell Time is obtained from the timestamp of intrusion detection logs generated by the system monitoring tools. The perception of the penetration testers are collected through surveys in order to perceive the effectiveness of deceptions. The results of case studies show that the deception methodology is effective in early detection of highly targeted attacks and creates confusion and uncertainty for the penetration testers.

Future work that would be beneficial to this research would be the development of additional qualitative metrics to test the effectiveness of active defences. Comparing the outcome of white box testing vs. black box testing would also be interesting. Testing the deceptions ability to detect insider threats, both malicious and accidental is much needed research, as they represent the weakest link of the security.

## V. References


[1] Danielle Anne Veluz. (2011, May) Trend Micro. [Online]. http://www.trendmicro.com/vinfo/us/threat-encyclopedia/web-attack/95/understanding-highly-targeted-attacks

[2] Ivan Dimov. (2013, June) Infosec Institute. [Online]. http://resources.infosecinstitute.com/guiding-principles-in-information-security/

[3] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin, "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," *Leading Issues in Information Warfare & Security Research*, vol. 1, p. 80, 2011.

[4] A.K. Sood and R.J. Enbody, "Targeted cyberattacks; a superset of advanced persistent threats," in *IEEE Security & Privacy*, vol. 11, 2013, pp. 54-61.

[5] Bryce Galbraith. (2015, October) Info Security. [Online]. http://www.infosecurity-magazine.com/opinions/apts-anticipatory-active-defenses/

[6] Trend Micro. (ND) Trend Micro. [Online]. http://www.trendmicro.com/vinfo/us/security/definition/targeted-attacks

[7] PwC et al. (2015) PwC. [Online]. http://www.pwc.com/us/en/increasing-it-effectiveness/publications/us-cybercrime-survey-2015.html

[8] K.E. Heckman, F.J. Stech, R.K. Thomas, B. Schmoker, and A.W. Tsow, "Intrusions, Deception and Campaigns," in *Cyber Denial, Deception and Counter Deception*.: Springer International Publishing, 2015, pp. 31-82.

[9] G. Briskin et al., "Design Considerations for Building Cyber Deception Systems," in *Cyber Deception*.: Springer International Publishing, 2016, pp. 71-97.

[10] W. Wang et al., "Detecting targeted attacks by multilayer deception," *Journal of Cyber Security and Mobility*, vol. 2, no. 2, pp. 175-199, 2013.

[11] Mohammed Almeshekah, Eugene Spafford, and Mikhail Atallah, "Improving Security Using Deception," *Center for Education and Research Information Assurance and Security, Purdue University, Tech. Rep. CERIAS Tech Repor*, vol. 13, 2013.

[12] Mohammed H Almeshekah, "Using Deception to Enhance Security," Purdue University West Lafayette, PhD Dissertation 2015.

[13] Rahmat et. al BudLarto, "Development Of Penetration Testing Model For Increasing Network Security," Network Research Group ,School of Computer Sciences, Pulau Pinang, White Paper 2004.

[14] Deutsche Telekom AG Honeypot Project. (2015, March) T-Pot: A Multi-Honeypot Platform. [Online]. http://dtag-dev-sec.github.io/mediator/feature/2015/03/17/concept.html#concept

[15] Sumit Sharma and Rahul Binjve. (2015, November) Shiva-Spampot. [Online]. https://github.com/shiva-spampot/shiva

[16] Masood Mansoori, Ian Welch, and Qiang Fu, "YALIH, Yet Another Low Interaction Honeyclient," in *Proceedings of the Twelfth Australasian Information Security Conference*, Auckland, 2014, pp. 7-15.

[17] (2016) Black Hills Infomation Security. [Online]. http://www.blackhillsinfosec.com/?page_id=4419

[18] OSSEC. [Online]. http://ossec.github.io

[19] Abdelkader Lahmadi and Frederic Beck, "Powering Monitoring Analytics with ELK Stack," in *International Conference on Autonomous Infrastructure, Management and Security (AIMS 2015)*. , 2015.

[20] John N. Stewart, "Advanced Technologies/Tactics Techniques, Procedures: Closing the Attack Window, and Thresholds for Reporting and Containment," *Best Practices in Computer Network Defense: Incident Detection and Response*, vol. 35, pp. 30-42, 2014.

[21] Wade M. Vagias. (2006) Clemson University. [Online]. https://www.clemson.edu/centers-institutes/tourism/documents/sample-scales.pdf

# Appendix 7

**Publication VII**
M. Kont, M. Pihelgas, K. Maennel, B. Blumbergs, and T. Lepik. Frankenstack: Toward Real-time Red Team Feedback. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 400–405, Baltimore, USA, November 2017. IEEE

# Frankenstack: Toward Real-time Red Team Feedback

Markus Kont
NATO Cooperative Cyber Defence Centre of Excellence
markus.kont[a]ccdcoe.org

Mauno Pihelgas
NATO Cooperative Cyber Defence Centre of Excellence
mauno.pihelgas[a]ccdcoe.org

Kaie Maennel
Tallinn University of Technology
kamaen[a]ttu.ee

Bernhards Blumbergs
NATO Cooperative Cyber Defence Centre of Excellence; IMCS UL, CERT.LV Laboratory
bernhards.blumbergs[a]cert.lv

Toomas Lepik
Tallinn University of Technology
toomas.lepik[a]ttu.ee

***Abstract*—Cyber Defense Exercises have received much attention in recent years, and are increasingly becoming the cornerstone for ensuring readiness in this new domain. Crossed Swords is an exercise directed at training Red Team members for responsive cyber defense. However, prior iterations have revealed the need for automated and transparent real-time feedback systems to help participants improve their techniques and understand technical challenges. Feedback was too slow and players did not understand the visibility of their actions. We developed a novel and modular open-source framework to address this problem, dubbed *Frankenstack*. We used this framework during Crossed Swords 2017 execution and evaluated its effectiveness by interviewing participants and conducting an online survey. Due to the novelty of Red Team-centric exercises, very little academic research exists on providing real-time feedback during such exercises. Thus, this paper serves as a first foray into a novel research field.**

***Keywords*—automation, cyber defense exercises, education, infrastructure monitoring, real-time feedback, red teaming**

## I. Introduction

Cyber defense exercises (CDX) are crucial for training readiness and awareness within the *cyber domain*. This new domain is acknowledged by NATO alongside with land, sea, air, and space [1]. Alliance nations are endorsing the development of both defensive and responsive cyber capabilities. In this context, the paper focuses on further evolving the quality and learning experience of CDX, aimed at developing cyber red teaming [2] and responsive skillset. Crossed Swords (XS) [3], a techical exercise developed by NATO Cooperative Cyber Defense Centre of Excellence (NATO CCD COE) since 2014, is used as a platform to create the proposed framework. The solution is applicable to any other CDX where standard network and system monitoring capability is available.

### A. Background

XS is an intense hands-on technical CDX oriented at penetration testers working as a single united team, accomplishing mission objectives and technical challenges in a virtualized environment. While common technical CDX is aimed at exercising defensive capabilities (i.e., Blue Team – BT), XS changes this notion, identifies unique cyber defense aspects and focuses on training the Red Team (RT).

To develop and execute the exercise, multiple teams are involved: rapid response team (i.e., RT); game network and infrastructure development (Green Team – GT); game scenario development and execution control (White Team – WT); defending team user simulation (i.e., BT); and monitoring (Yellow Team – YT).

The RT consists of multiple sub-teams based on the engagement specifics, those being: network attack team, targeting network services, protocols and routing; client side attack team, aiming at exploiting human operator and maintaining access to the hosts; web application attack team, targeting web services, web applications and relational databases; and digital forensics team, performing data extraction and digital artefact collection. These sub-teams must coordinate their actions, share information and cooperate when executing attacks to reach the exercise objectives.

The main goal is to exercise RT in a stealthy fast-paced computer network infiltration operation in a responsive cyber defense scenario [4]. To achieve this, the RT must uncover the unknown game network, complete a set of technical challenges and collect attribution evidence, while staying as stealthy as possible. Note that XS is not a *capture-the-flag* competition, as the RT has to pivot from one sub-objective to another in order to achieve the final mission according to the scenario. Furthermore, Red sub-teams are not competing with each other, and rather serve as specialized branches of a single unit.

### B. Problem Statement

Prior XS iterations revealed several problems with RT learning experience. Primarily, the YT feedback regarding detected attacks from the event logs and network traffic was presented at the end of every day, which was not well suited to the short, fast and technical nature of the exercise. The feedback session addressed only some noteworthy observations from the day, but RT participants need direct and immediate feedback about their activity to identify mistakes as they happen. This feedback needs to be adequately detailed, so that the RT can understand why a specific attack was detected and then improve their approach. Finally, to make the feedback faster, the slowest element in the loop—the human operator—needs to be eliminated.

Therefore, manual data analysis by the YT needs to be automated as much as possible. To achieve this, we used the same open-source tools as in the previous XS iterations, but added in event correlation, a novel query automation tool, and a newly developed visualization solution. We decided to call the framework *Frankenstack*. Fig. 1 illustrates the role of Frankenstack in the XS exercise.

978-1-5386-0595-0/17/$31.00 ©2017 IEEE

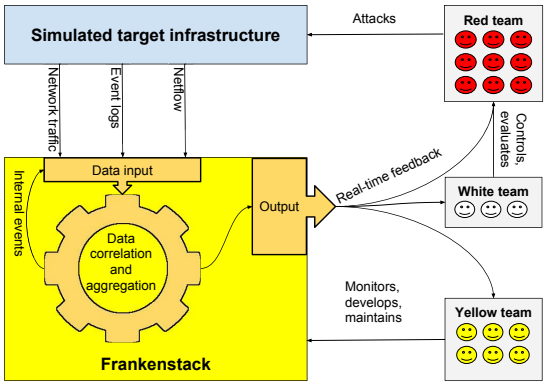



Fig. 1. High-level overview of Frankenstack

The RT has to receive timely and efficient feedback from the YT regarding detected attacks on the target systems. This feedback is critical to raise the level of stealthiness, identify the gaps of RT coordination, and analyze the tools and tactics used for computer network operations. The effectiveness of our framework was assessed during the main execution of XS 2017 (XS17), where the stack provided real-time monitoring feedback to the RT.

The remainder of the paper is organized as follows: section II provides an overview of related work, section III describes our monitoring stack, section IV presents RT feedback results, while section V discusses future work, and section VI concludes the paper.

## II. Related work

For teaching purposes, the benefit of exercises and competitions is generally well accepted and documented [5], [6], [7], [8]. Unfortunately, not much research has focused on the perception of feedback which is provided to the training audience, especially in the context of monitoring technical indicators of compromise in realistic environments. Thus, this section presents research related to both measuring and improving the learning experience as well as situation awareness (SA) during cyber exercises.

Dodge et al. discussed CDX network traffic analysis in [9], a practice that is common in modern exercises not only for situational awareness (SA) but also as educational tool, for elaborating attacker campaigns, for training network analysts, etc. However, this early paper focuses on traffic capture and initial profiling, and does not consider distractions such as traffic generation, increasing infrastructure complexity, host instrumentation, data source correlation, or the need for immediate feedback. In [10], Holm et al. correlated network traffic and RT attack logs from Baltic Cyber Shield, a precursor for Locked Shields and Crossed Swords exercises. However, their goal was to improve existing metrics for vulnerability scoring, as opposed to participant education. Likewise, in [11], Brynielsson et al. conducted a similar empirical analysis on CDXs to profile attacks and create attacker personas.

In [12], Arendt et al. presented CyberPetri, a circle-packing visualization component of Ocelot, which was previously presented in [13] as a user-centered decision support visualization. They presented several use cases of the tool, but their main goal was high-level feedback to network analysts based on target system service availability reports. Although the tool was useful for high-level decision making, technical RT members are more interested in immediate effects of their attacks on target systems. Note that any single system is often a supporting pillar for more complex services, and is not noticeable to end-users. Nevertheless, modern security monitoring is built upon instrumentation of these systems, to find RT *footprints* and to trigger notification upon breaching these digital tripwires.

A paper [14] by Henshel et al. describes the assessment model for CDXs based on the Cyber Shield 2015 example, as well as integrated evaluation of metrics for assessing team proficiency. In addition to data collected during the exercise, they also conducted a pre-event expertise survey to determine possible relationships between prior expertise and exercise performance. For future assessments they suggest that near real-time analysis of the collected data is required—they stress that raw data collection is not a problem, but the capability to meaningfully analyze is the limiting factor. Manual methods do not scale with the huge amounts of incoming data. This closely coincides with our observations in section I-B and this is what we aim to improve.

Furthermore, existing academic research commonly relies on monolithic tools, which are often not accessible to the general public, thus, making experiments difficult, if not impossible, to reproduce. We seek to provide an inexpensive open-source alternative to these products. The next section describes our modular monitoring architecture.

## III. Frankenstack

Commercial tools are too expensive for smaller cyber exercises, in terms of licensing fees, hardware cost and specialized manpower requirements. Detection logic in commercial tools is also not available to the general public, which hinders YT's ability to provide detailed explanations of detected attacks. Frankenstack is easy to customize as individual elements of the stack are industry standard tools which can be interchanged. Note that we opted to use a commercial tool *SpectX* as an element within Frankenstack for log filtering, due to on-site competency and developer support. However, this function could have been achieved with the open-source Elastic stack [15]. Our stack provides a clear point of reference to other researchers and system defenders who wish to compile the monitoring framework in their particular environments, as the overall architecture is novel.

The data available to us during XS included full ERSPAN (Encapsulated Remote Switched Port ANalyzer) traffic mirror from gamenet switches and NetFlow from gamenet routers. This was provided by the GT. Furthermore, we instrumented

gamenet systems to collect numerical metrics (e.g., CPU and memory usage, and network interface statistics) and logs (e.g., syslog from Linux, Event Logs from Microsoft Windows, Apache web server access logs, and audit logs from Linux command line executions). Such host instrumentations are very difficult to implement in a standard CDX with BT training focus: if the intent is to give BTs full control of a simulated infrastructure, then they also have full volition to disable these tools. However, as the XS training audience is the RT, then we could maintain control of all target systems and ensure a constant stream of monitoring data. Moreover, we complemented the list of BT data sources with various YT honeypots and decoy servers.

Detailed overview of the resulting stack, in relation to data processing pipelines, is presented in Fig. 2. The blue area represents available data sources, the gray area stands for data storage, and the yellow area denotes the YT presentation layer (i.e., visualization tools on five monitors). Blue and green elements represent target systems and all other elements outside colored boundaries are processing tools. Custom tools that we developed are highlighted with a dark yellow circle. Note that some tools, such as Moloch, are designed for both data storage and visualization, but are not presented in these respective areas because only their API components were used for processing automated queries.

We opted against using NetFlow data, as modern packet capture analyzers (e.g., Suricata, Bro, and Moloch) can fill this role, albeit by needing more processing power and memory. Additionally, these tools commonly present their output in textual log format, which we fed back into the central logging and correlation engine. Thus, the problem of identifying and displaying high-priority IDS alerts can be simplified into a log analysis problem.

Frankenstack uses event correlation for integrating various information sources as this field has been well researched in the context of log management [16], [17], [18]. We open-sourced the correlation ruleset in [19]. See Listing 1 for an example raw log entry from Snoopy Logger [20] that was converted into a more universal human-readable security event that could be presented to the general audience on various dashboards while preserving the raw message for anyone wishing to drill down. Note that specific IP addresses have been removed from this example. This generalization is necessary for handling and grouping subsequent log entries that continue describing the same event, e.g., additional commands executed on the same host via SSH.

Listing 1. Event generalization by frankenSEC

```
#INPUT
login:administrator ssh:(SRC_IP 58261 DST_IP 22)
username:administrator uid:1001 group:administrator
gid:1001 sid:6119 tty:(none) cwd:/home/administrator
filename:/usr/bin/passwd: passwd administrator

#OUTPUT
SRC_IP->[DST_IP]: Command execution by administrator
over SSH
```

Post-mortem analysis of available data sources has proven effective during prior CDXs for packet capture (PCAP) analysis, but requires a significant amount of time and manual work. Again, this clashes with the short time-frame of a CDX. Furthermore, search queries are often written ad hoc during investigations and subsequently forgotten, making analysis results difficult to reproduce. Thus, we created *Otta* [21], a novel query documentation and automation tool for periodically executing user-defined queries on large datasets and converting aggregated results into time-series metrics. *Otta* enables trend graphing, alerting, and anomaly detection for stored user-defined queries. This reduces time spent on analysis and ensures reproducibility by documenting the queries that produced the results.

We used various open-source tools for timelining metrics and log data, for displaying alerts, and presenting correlated information. There are slight differences in handling various incoming alerts. While many types of alerts (e.g., CPU and disk usage) trigger and recover automatically based on a set of thresholds, there are some types (e.g., IDS alerts) that lack the concept of a recovery threshold. Thus, the alert will never recover once raised, leading to an overabundance of information on the central dashboard. Furthermore, batch bucketing methods and timelines are lossy, as only the most frequent items are considered. The volatile nature of CDXs and an abundance of generated network traffic can therefore cause these dashboards to be too verbose to follow efficiently.

Attack maps are not usable because they rely on geographical data which is completely fictional in many CDX environments. Therefore, we developed Event Visualization Environment, or EVE, a novel web-based tool for visualizing correlated attacks in relation to gamenet infrastructure. The Alpha version of this tool has been made publicly available in [22]. EVE is a web application that shows attacks carried out by the RT in real time with a custom gamenet network map as background. Probes can send information to EVE listener in JSON format. Real-time visualization is using WebSocket technology—eliminating the need to reload the page for updated results.

EVE supports combining multiple events in a short time window, and that share the same source and destination addresses, into a unified attack. Resulting attacks are subsequently displayed as arrows connecting the circles around source and target hosts on the network map, while detailed attack information is displayed next to the network map. Using the gamenet map makes EVE a very intuitive tool for enabling participants and observers alike to comprehend CDX events on a high-level.

During the exercise EVE was available only to YT and WT members, as it revealed the entire exercise network map that RT had to discover on their own. However, EVE has a dedicated replay mode to display all the attacks condensed into a given time period, allowing participants to obtain an overview of the attacks as well as understand the pace and focus of the previous attack campaign. For instance, attacks from the previous day can be replayed in 15 minutes. EVE

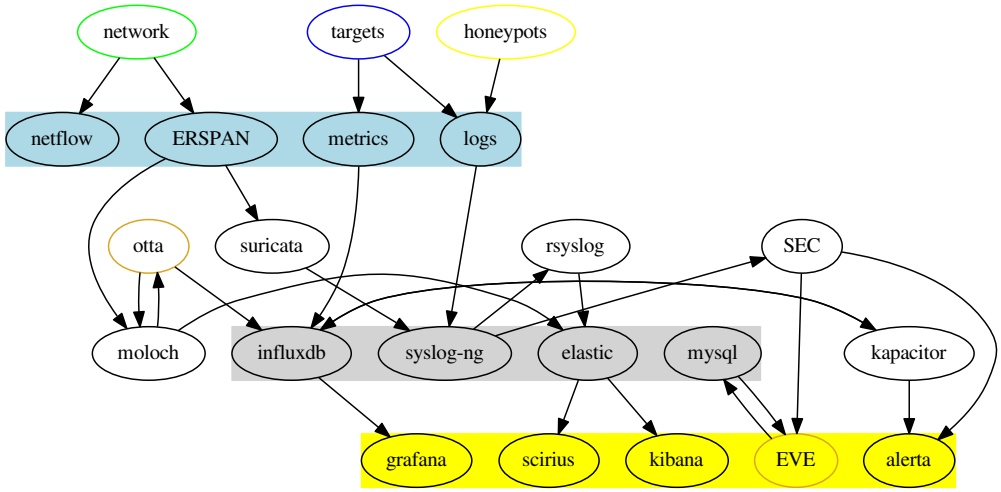



Fig. 2. Data flow between Frankenstack elements during XS17

TABLE I
DEDUPLICATION BY EVENT SOURCE

| Event source | Total events | Unique events displayed | Percentage displayed |
|---|---|---|---|
| Apache | 1908 | 35 | 1.83% |
| IDS | 23790 | 616 | 2.59% |
| Snoopy logger | 2962 | 40 | 1.35% |
| *Total* | *28660* | *691* | *2.41%* |

was shown in replay mode to RT participants after the exercise concluded. This compressed replay was very effective in presenting the most prevalent problems, such as periodic beaconing during otherwise silent night periods and verbosity of particular network sub-team attacks.

Alerta [23] served as the primary dashboard to display alerts to the RT. We used the HTTP API for submitting Frankenstack events to Alerta. The RT had direct access to the Alerta web interface and could write their own filtering rules to view information relevant to their current campaign. Finally, we present Tab. I to illustrate how Frankenstack performed in deduplicating the events that were displayed to the RT on the Alerta dashboard. Note that deduplication was primarily based on the generalized event descriptions (see Listing 1).

## IV. ASSESSMENT

The tools and infrastructure are essential for learning, but they do not make the exercise successful by default. Often human factors, such as how YT and RT members perceive and use the tools, have significant impact.

One essential part of the assessment was to observe the behavior of the RT members and their interaction with Frankenstack during the exercise in order to gain further insights into their progress and learning experience. We carried out qualitative interviews with RT participants, to estimate their reaction to Frankenstack and their overall learning progress. The interviews took place in casual settings during breaks in execution. Furthermore, we conducted a quantitative survey in the form of an online questionnaire. The survey consists of multiple choice or ranking style questions with the ability to provide additional comments for each question. The survey concluded by asking some general questions about meeting the training objectives and overall satisfaction with the exercise.

### *A. Feedback combined from interviews, survey and observations*

This subsection includes the analysis of participants feedback. Improvement suggestions to learning design are presented in the following subsection IV-B.

We received 14 survey responses out of 27 participants (52%). 46% of participants had attended other exercises, but none of those exercises had attempted to provide SA via a similar toolset. The remaining 54% had not previously attended any exercise.

There were four large screens in the training room directed to the RT, displaying Alerta, Grafana, Scirius, and Suricata. A fifth screen displaying EVE was only visible to YT and WT members. Most RT members preferred to view the main screens displayed in training room, and 38% responded that they checked the screens every 60 minutes or less. Another 38% checked the screens every 30 to 50 minutes. RT members were not restricted from accessing any of the Frankenstack web interfaces. The survey revealed that learners did access the monitoring framework on their local computers when attempting new attack vectors. Thus, tools served their intended usage.

Alerta was considered most useful (46%), followed by Moloch (31%). There was no clear result for the least useful tool. The respondents expressed mixed feelings on the ease of

use of the SA tools: 38% equally agreeing and disagreeing, and the remainder (24%) being neutral.

Regarding learning impact, 79% agreed (of those 57% strongly agreed) that the SA given during exercise is useful for their learning process, while 21% were neutral. In terms of the feedback rate, 77% of the respondents considered the speed of feedback to be at the correct level, 15% considered it too slow and 8% considered it too fast. Furthermore, 57% agreed that alerts were accurate and sufficient for their learning process, while 43% were neutral about this question. However, several respondents revealed being too focused on achieving their primary objectives, and thus unable to properly switch between their tools and feedback screens.

In relation to visibility, 45% of the participants agreed that they had learned a lot about how their actions can be detected (i.e., it is useful to see simultaneously what attack method could be detected, and how), and 30% were more careful with their attacks and thus tried to be stealthier than they normally would have been. However, there were some unintended side-effects. The feedback sometimes provided insight into the network map that the RT was tasked to discover independently. For example, if the RT probes a yet unknown node on a network, the logs generated on the host might reveal the target hostname (e.g., *sharepoint* or *ftp*), which consequently implies the purpose of the system—something that would not be apparent from an IP address. Thus, there is a fine line between revealing too little or too much to the training audience.

Furthermore, some comments revealed a loss of emphasis on stealth due to exercise time constraints, i.e., RT members knowingly used more verbose techniques closer to the objective deadline. To clarify, 64% of respondents confirmed that the SA tools were not distracting them nor had negative impact, while 30% agreed that they were distracted. The remaining 6% were neutral. This confirms the challenges of providing instant feedback, as the learning potential is not fully used. The question is how this learning experience is impacting long-term behavior of the participant.

One of the key training aspects is working as a team in achieving goals. Thus, team communication and cooperation are vital. Overall, 83% of respondents indicated some improvement of the skills for these specific training objectives. However, feedback concerning the impact of SA tools on team communication and cooperation is mixed—50% perceived positive impact, whereas 21% were negative and remainder were neutral. Several respondents acknowledged less need for verbal communication, as they could see relevant information on the screens. Unfortunately, not all RT members were able to interpret and perceive this information correctly. This combined with the reduced need for communication meant that not all participants progressed as a team.

Compared to other CDXs, 50% responded that they needed less information from YT members, as they obtained relevant SA on their own. Guidance, however, is a critical success factor for learning, especially in a team setting. 64% of participants said they had sufficient help for their learning process, i.e., when they did not know how to proceed, their team members or sub-team leaders provided guidance. However, 64% is a rather disappointing result and could clearly be increased with improved learning design. Some respondents admitted that they did not know how other teams were progressing and wasted time on targets that were not vulnerable. This caused significant frustration and stress, especially when combined with the compressed timeframe of a CDX.

### *B. Learning improvement suggestions*

Given the amount of work that goes into preparing such exercises, the level of learning potential needs to be maximized. Our analysis suggests that small learning design changes may have significant impact. This section presents the main recommendations derived from these results.

From the learning perspective, we cannot assume that participants know how to use or interpret the results. Lack of in-depth knowledge of monitoring tools (e.g., where is raw data collected, what is combined and how, what needs to be interpreted in which way, etc.) has a negative impact on learning. A dedicated training session or workshop needs to take place prior to execution. Furthermore, in the light of the survey results, inclusion of various tools into Frankenstack needs to be carefully evaluated to avoid visual distractions for RT participants. There is also a need to reduce prior system and network monitoring knowledge by making the output more self-explanatory.

Given the difficulties in switching between multiple screens whilst also trying to achieve an objective in unfamiliar network, one can easily suggest compressing the amount of presentable information to reduce the number of monitoring screens. However, this cannot be attained without reducing the amount of technical information. The purpose of Frankenstack is not to provide SA to high-level decision makers, but to present feedback to technical specialists. Thus, a better approach would be restructuring each sub-team with a dedicated monitoring station with a person manning it, allowing team members to focus on their objectives and get feedback relevant only for their actions. As such, RT members must be given a *hands-on* opportunity to use monitoring systems.

In RT exercises such as XS, there are several main objectives to be achieved by the whole RT. It is challenging to evaluate reaching objectives, since there are many steps involved in reaching a specific objective. Often the tasks or sub-objectives are divided between sub-teams (network, web and client-side) and between individuals in those sub-teams. The difficulty of a specific exploitation depends on the individual's skillset, which varies widely. Hence, there is a trade-off between assigning a task to an experienced member to increase the chance of success, versus teaching a new member. For example, an experienced network administrator is more effective in exploiting network protocols and is likely less visible while doing so, but may not learn anything new.

Discussions and feedback revealed that several respondents felt they were stuck and working alone. Division of the tasks between sub-teams and individuals also diminishes the learning potential. One training design option to alleviate this issue

would be regular *team timeouts* for reflection. Reflective team sharing is crucial for the learning success of each individual, and would overcome the project management approach where each team member focuses only on personal objectives. Higher emphasis should be on offering tips and helping those stuck on an objective to move forward whilst also keeping track of the feedback provided by Frankenstack. The coaching could also be handled in the form of a *buddy system* where RT members are not assigned a sub-task individually, but in groups of two or three. They would then have to share their knowledge and can benefit from different individual backgrounds.

Finally, it is important to have better time-planning during the execution. While it is certainly appropriate to allow for flexibility in the paths that the RT can take to solve the objectives, participants should avoid spending too much time on wrong targets. Nevertheless, the learning impact of the exercise in this format (i.e., with real-time feedback) is very positive. Only 13% of all participants' responses reported no significant change in their skills, while an overwhelming 87% perceived an improvement in their skill level, and 93% agreed that they were satisfied with exercise.

## V. Future work

We encountered several unforeseen problems, as methods for assessing technical RT campaigns have to be incorporated into the game scenario itself. However, most XS17 targets had already been developed before the initial stages of this research. We plan to increase information sharing between Red and Yellow teams to improve RT progress measurement. Thus, we can develop better assessment methodologies for RT skill levels and YT feedback framework.

Development of a new dynamic version of EVE is already underway for the next XS iteration. In addition to the network map view, it can draw the network map dynamically as RT compromises new targets. Currently, EVE can only be used after the end of the exercise. However, in addition to providing more actionable alerting, the new version can also reduce RT work for mapping new systems and allow them to focus on the technical exercise.

## VI. Conclusion

In this paper, we have presented the core challenges in organizing a CDX with Red Team emphasis, such as timeliness and accuracy of feedback, and ensuring participant education without compromising the game scenario. We compiled a novel stack of open-source tools to provide real-time feedback and situational awareness, and conducted surveys among the RT members to assess the effectiveness of this method.

Frankenstack feedback regarding learning impact was mainly positive. However, there are critical questions to answer when designing the RT exercises, such as what is the right balance of information to provide to the RT, does the behavior change due to monitoring or information visible (i.e., learners unconsciously limit themselves by not trying out more risky strategies, etc.). Also, some further learning design changes, and not necessarily only limited to SA, can maximize the return on the significant investment into preparing such RT exercises. We hope to spark a discussion on improving these problems.

## VII. Acknowledgments

The authors would like to thank Mr. Risto Vaarandi, Mr. Hillar Aarelaid and Prof. Olaf M. Maennel for their valuable contributions. This work has been supported by the Estonian IT Academy (StudyITin.ee).

## References

[1] T. Minárik, "NATO Recognises Cyberspace as a Domain of Operations at Warsaw Summit," Available: https://ccdcoe.org/nato-recognises-cyberspace-domain-operations-warsaw-summit.html.

[2] P. Brangetto *et al.*, "Cyber Red Teaming - Organisational, technical and legal implications in a military context," NATO CCD CoE, Tech. Rep., 2015.

[3] "Crossed swords exercise," Available: https://ccdcoe.org/crossed-swords-exercise.html.

[4] P. Brangetto *et al.*, "From Active Cyber Defence to Responsive Cyber Defence: A Way for States to Defend Themselves Legal Implications," Available: https://ccdcoe.org/multimedia/active-cyber-defence-responsive-cyber-defence-way-states-defend-themselves-legal.html.

[5] B. E. Mullins *et al.*, "The impact of the nsa cyber defense exercise on the curriculum at the air force institute of technology," in *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, Jan 2007, pp. 271b–271b.

[6] A. T. Sherman *et al.*, "Developing and delivering hands-on information assurance exercises: experiences with the cyber defense lab at umbc," in *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, June 2004, pp. 242–249.

[7] R. C. Dodge *et al.*, "Organization and training of a cyber security team," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 5, Oct 2003, pp. 4311–4316.

[8] G. H. Gunsch *et al.*, "Integrating cdx into the graduate program," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 5, Oct 2003, pp. 4306–4310.

[9] R. C. Dodge and T. Wilson, "Network traffic analysis from the cyber defense exercise," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 5, Oct 2003, pp. 4317–4321.

[10] H. Holm *et al.*, "Empirical analysis of system-level vulnerability metrics through actual attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 6, pp. 825–837, Nov 2012.

[11] J. Brynielsson *et al.*, "Using cyber defense exercises to obtain additional data for attacker profiling," in *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, Sept 2016, pp. 37–42.

[12] D. Arendt *et al.*, "Cyberpetri at cdx 2016: Real-time network situation awareness," in *2016 IEEE Symposium on Visualization for Cyber Security (VizSec)*, Oct 2016, pp. 1–4.

[13] D. L. Arendt *et al.*, "Ocelot: user-centered design of a decision support visualization for network quarantine," in *2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*, Oct 2015, pp. 1–8.

[14] D. S. Henshel *et al.*, "Predicting proficiency in cyber defense team exercises," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Nov 2016, pp. 776–781.

[15] "Elastic stack," Available: https://www.elastic.co/.

[16] R. Vaarandi *et al.*, "Simple event correlator - best practices for creating scalable configurations," in *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2015 IEEE International Inter-Disciplinary Conference on*, March 2015, pp. 96–100.

[17] R. Vaarandi, "Platform independent event correlation tool for network management," in *Network Operations and Management Symposium, 2002. NOMS 2002. 2002 IEEE/IFIP*, 2002, pp. 907–909.

[18] ——, "Sec - a lightweight event correlation tool," in *IP Operations and Management, 2002 IEEE Workshop on*, 2002, pp. 111–115.

[19] "Frankensec," Available: https://github.com/ccdcoe/frankenSEC.

[20] "Snoopy Logger," Available: https://github.com/a2o/snoopy.

[21] "Otta," Available: https://github.com/ccdcoe/otta.

[22] "Eve - event visualization environment," Available: https://github.com/ccdcoe/EVE.

[23] N. Satterly, "alerta," Available: http://alerta.io/.

# Appendix 8

**Publication IX**
D. Mucci and B. Blumbergs. TED: A Container Based Tool to Perform Security Risk Assessment for ELF Binaries. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 361–369, Prague, Czech Republic, February 2019. SCITEPRESS

# TED: A Container Based Tool to Perform Security Risk Assessment for ELF Binaries

Daniele Mucci[1], Bernhards Blumbergs[1,2]
[1]*Centre for Digital Forensics and Cyber Security*
*Tallinn University of Technology*
[2]*CERT.LV, IMCS UL*
*mucci.daniele@gmail.com, name.surname[at]cert.lv*

Keywords: ELF binary analysis, GNU/Linux system hardening, Vulnerability assessment, Software containers.

Abstract: Attacks against binaries, including novel hardware based attacks (e.g., Meltdown), are still very common, with hundreds of vulnerabilities discovered every year. This paper presents TED, an auditing tool which acts from the defense perspective and verifies whether proper defenses are in place for the GNU/Linux system and for each ELF binary in it. Unlike other solutions proposed, TED aims to integrate several tools and techniques by the use of software containers; this choice created the necessity to compare and analyze the most popular container platforms to determine the most suitable for this use case. The containerization approach allows to reduce complexity, gain flexibility and extensibility at the cost of a negligible performance loss, while significantly reducing the dependencies needed. Performance and functionality tests, both in lab and real-world environments, showed the feasibility of a container-based approach and the usefulness of TED in several use cases.

## 1 MOTIVATION FOR THE PROJECT

From the 1st of January 2017, around 150 different vulnerabilities involving stack or heap overflows and format string bugs have been reported (NIST, 2018). In addition, during this period several hardware vulnerabilities which also involved binaries have been found (e.g., Spectre/Meltdown, Throw-/NetHammer). Also reverse engineering should always be considered as a possible option for targeting binaries, especially where intellectual property is involved. Considering the enormous diffusion of GNU/Linux based systems, it is clear that protecting ELF binaries in such environment is of critical importance, and therefore this is the primary purpose of this project.

## 2 INTRODUCTION

Many tools and techniques exist to protect systems or binaries from a wide range of attacks, and many more are continuously developed. However, despite the large number of protection mechanisms designed, a relatively small and consolidated set of features is commonly used and deployed. This set includes defenses such as Address Space Layout Randomization (ASLR), Data Execution Prevention (DEP, NX/XD), Stack Smashing Protector (SSP) or Stack canaries. For the most recent Linux kernels, this set can be extended with the microcode to protect against some Spectre variants (e.g., Meltdown). On the other hand, verifying whether all these measures are in place is usually done either manually or by the aid of some tools (e.g., *Radare2*, *Lynis*). This action has to be performed separately for each tool used or individually for each binary, making it time consuming, error-prone and inefficient. Moreover, low-level tools often require specific dependencies for libraries or other tools, which might not always be possible to satisfy or whose installation might require additional time and effort.

The solution proposed in this paper consists of using software containers to address the issues mentioned, specifically to integrate and orchestrate multiple tools, to collect and process the different results, and most of all to greatly reduce the dependencies required, allowing to individually pack the needed tools with their dependencies, thus eliminating possible conflicts and the need for their installation. Given the fact that software containers are still rarely used

outside cloud environments, a significant amount of work consisted in analyzing and comparing the various available container engines to select the one that provided the needed functionalities and at the same time did not represent a heavy dependency itself. The tool here presented, TED, implements such solution using Docker containers to perform all the necessary tests aimed to verify whether a selected set of defense measures are in place, and to consequently assess the risk associated with each binary present in the system. Proposed solution would be practically applicable to the cyber security areas such as, vulnerability assessment, incident response, system triaging, and security baseline establishment.

This paper provides the following contributions:

1. discussion of software containers as execution environment and evaluation of the different container engine platforms in the context of vulnerability assessment; and
2. container technology implementation for ELF binary security assessment in an open-source toolset TED.

This paper is organized as follows: Section 3 gives an overview of related work; Section 4 describes the container engine selection process, the design and the implementation of TED; Section 5 provides the result of the evaluation process; Section 6 concludes this paper.

## 3 RELATED WORK

Extensive research already exists on binary security, both from the attack and from the defense perspective. However, very few projects, with a purpose similar to the one presented in this paper, were found.

BitBlaze is a platform developed by Song et al. (Song et al., 2008) which uses both static and dynamic analysis to extract a wide range of security information from a program, without taking into consideration the defense measures applied and relying on custom techniques. The main purpose of BitBlaze is to detect possible vulnerabilities in the program and to identify their root cause, rather than determining what security measures are applied. BitBlaze has several components, in particular a static analysis tool (VINE, available and not maintained for 4 years) and a dynamic analysis component (TEMU, available and not maintained for 3 years). Young-Hyun et al. (Choi et al., 2015) in 2015 developed a project called DBA (Dynamic Binary Analyzer), capable of dynamically detecting vulnerabilities in binaries with taint analysis, which targets x86 (32-bit) Windows binaries. This project focuses on finding vulnerabilities or detecting exploitation at runtime. To perform its analysis, DBA uses QEMU virtual machine to emulate the execution environment for a single binary. TEASER by Ulrich (Ulrich, 2017) is a system, which aims to assess the exploitability of binaries, performing a vulnerability assessment from the perspective of an attacker. TEASER is limited to identifying memory corruption vulnerabilities and is meant to ease the process of detecting bugs which might lead to exploits. It is built on top of other tools, such as *Valgrind*, *PANDA*, *ASan* and *LLVM*, and uses *QEMU* emulation for some steps of its execution. Tang et al. (Feng-Yi et al., 2016) and Wang et al. (Wang et al., 2017) in their projects focused on binary security analysis in terms of performing a diagnosis of memory vulnerabilities. The two projects which can be compared with TED are *checksec.sh* [1] and Lynis[2]. The first is a *Bash* shell script, which shows technical information, including whether some security measures are applied, regarding a binary, a loaded library or the kernel. The main script is not maintained anymore, but a *forked* and maintained version[3] exists. Lynis, on the other hand, is a software aimed to audit, hardening and testing for compliance Unix systems. The software runs a wide range of tests according to what tools are available on the system, and it is publicly available.

In addition to the presented tools, there is a conspicuous number of proposals to protect binaries from a wide range of attacks, however, an evident gap between the academia and the industry emerged. This means that virtually all the novel tools or techniques, such as (Marco-Gisbert and Ripoll, 2013; Solanki et al., 2014; Younan et al., 2006; Chen et al., 2017; Novark and Berger, 2010), developed in the academic environment, independently by their efficacy and security impact, are either unused or extremely rarely deployed in the production environments.

In the related work, multiple limitations and drawbacks have been identified, such as, the need for specific and numerous dependencies, the use of heavy virtualization technologies (e.g., KVM/QEMU) and their configuration, the support only for Windows binaries or for 32-bit architectures, the focus on single binaries rather than on the whole system, the lack of automation and the need of user interaction and finally the use of technologies that make the tool not portable, not easily extensible or not suitable for cloud environments. TED aims to address all the gaps identified by bundling all the tools and dependen-

[1] http://www.trapkit.de/tools/checksec.html
[2] https://cisofy.com/lynis/
[3] https://github.com/slimm609/checksec.sh

cies needed inside Docker containers, which not only make the application extremely portable, but avoid the overhead of an hypervisor, reducing drastically the performance loss; given this approach, the set of dependencies of TED is fixed and it is limited to the Docker platform and a small number of Python packages. Furthermore, TED focuses on 64-bit executables and aims to automate the scanning process, without requiring user interaction, in order to grant scalability to the project and to allow users to run TED periodically. Finally, TED not only collects and integrates information about both the system and the binaries in it, but also provides an assessment based on the information gathered, to allow for easier planning and better focus on critical parts of the system.

# 4 DESIGN AND IMPLEMENTATION

The design and implementation of the tool proposed here, TED, necessitated of several steps. First, the most suitable container engine for the context in which TED operates needed to be selected, then, it was required to individuate the defense measures and techniques to integrate in the tool, and furthermore to establish a corresponding scoring system to measure the system's ELF binary state. Finally, the tool had to be implemented.

## 4.1 Container Engine Selection

The idea of software isolation was born during the 1980s, however, the first use of containers dates back to the early 2000s. Despite this, only from 2015 containers have become a popular and ubiquitous technology, therefore the market is still relatively young, leading to the flourishing of many container engines and orchestration systems. Among these, five available platforms have been taken into consideration: Docker[4], Rkt[5], LXC[6], OpenVZ[7] and Garden[8]. OpenVZ and Garden have been discarded before the detailed evaluation for the following technical reasons: OpenVZ has limited functionalities if it is not run on a specific kernel, imposing a heavy constraint for a tool that aims to be generic and portable; Garden on the other hand is not yet a mature project and it is virtually not used, despite it aims to represent an alternative to Docker.

The remaining platforms – LXC, Docker and Rkt, have been evaluated using three parameters:

1. Availability: what needs to be done to get and install the platform, what dependencies need to be satisfied and also whether there are public repositories from which container images can be obtained.
2. Functionality: whether and how the container engine provides capabilities such as building images, collecting logs and events and sharing portion of the file system with the host.
3. Performance: the overhead caused by executing code inside a container for each engine is measured running 1000 times a test script and comparing its execution time with a baseline established running the same script on the native machine.

The most important criterion is the functionality, since a lack of features would impact the capabilities of TED or increase its complexity. Following the functionality, the availability is considered, as this impacts the dependencies needed by TED, which are meant to be as few as possible. Finally, performances are considered of secondary importance, since the difference in performances between technologies which are fundamentally similar is expected to be small.

*Docker.* Docker is the most popular container engine and the current industry standard. It can be installed in most cases from the package manager and does not need any particular Kernel feature, although *root* access is necessary to run the Docker daemon. The necessity of root access and its security implications are well documented[9] and require special attention. It is important to note that Docker is included in many cloud-oriented operating systems, such as CoreOS Container Linux [10] and RancherOS [11]. Container images that can be run by Docker are publicly available online from a public repository called Docker Hub; the Docker Hub makes extremely easy to find, run and publish an extensive set of images through HTTP. The Docker engine is focused on providing an execution environment for one process only, meaning that in general there should be a 1:1 ratio between Docker containers and applications running, and also that a container terminates when the process with PID 1 exits. Docker offers a rich set of functionalities, including an API to build, run and manage containers; in particular, it offers a simple way to

[4] https://www.docker.com/
[5] https://coreos.com/rkt/
[6] https://linuxcontainers.org/
[7] https://openvz.org/
[8] https://github.com/cloudfoundry/garden
[9] https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface
[10] https://coreos.com/os/docs/latest/
[11] https://rancher.com/rancher-os/

start and stop containers, extract logs or outputs from them, build new container images and share portions of the file system with the host.

*LXC.* LXC is defined as "a userspace interface for the Linux kernel containment features" (LXC-Doc, 2018). Unlike Docker, the purpose of LXC is to provide a full virtual environment without the burden of running a separate kernel and a hypervisor. Thus, in this case, several applications can run inside one single container. There are few hard dependencies for LXC which are likely to be fulfilled on most systems and therefore do not impose a hard constraint. Installing LXC is generally possible through the package manager, although no operating system comes with LXC pre-installed. It is possible to import public "templates" for different operating systems, but there is not any public repository for already made containers, however, it is still possible to distribute images via HTTP with a private web server. The functionalities offered by LXC are many and powerful, but in general require a careful configuration. The building process for an image is very different from the one offered by Docker: with LXC it is possible to "export" and then "import" an image. The export produces usually a *tar.xz* file or two *tarballs* that can be imported, which are more similar to a snapshot of the running container, rather than to a fresh image. Information about container events can usually be found in the host journal, while, unless manually specified, container logs need to be extracted directly from inside the containers. It is possible to share a portion of the file system with the host by simply mounting it inside the container, provided that a correct permission configuration is in place.

*Rkt.* Rkt (Rkt-Doc, 2018), read "rocket", is a container engine developer by CoreOS Container Linux team, and – apart from low-level details – it is very similar to Docker. Rkt containers are meant to run a single application on which their lifecycle depends. Rkt is available from the package manager in most Linux distributions, with the exception of Ubuntu and CentOS, and it is pre-installed on CoreOS Container Linux. The only dependency for running Rkt containers is a kernel version 3.18 or later. Rkt can run both Application Container Images (ACI) and Docker containers. For the firsts, the location of the container needs to be known *a priori* to be downloaded, since a public repository of images is not available, while for the latter it is possible to use the Docker Hub. The functionalities of Rkt, especially from the perspective of this project, are very similar to the ones offered by Docker, except for building a new image. Since Rkt can run Docker images, it is possible to use Docker build feature, but in a more general process, the build of a new container needs to be done through *acbuild*[12] tool, that appears to be not maintained. The logs of Rkt containers can generally be read through the host journal and most of the operations necessary to manage containers can be performed through the Rkt Command Line Interface (CLI) while an API with read-only capabilities is available. Events and log collection is done in a similar fashion to Docker, and similarly volumes are used to share the file system with the host.

**Performance overview.** To measure and compare the performances of the different platforms, a simple script which emulates some functionalities of TED has been created. This script has then been executed 1000 times on the native machine and inside a container run with each of the engines presented. The average execution time on the native operating system was 3.8s, whereas Rkt and Docker offered faster performances – 3.33s (-12.4%) and 3.5s (-7.9%) respectively. The slowest result was obtained by the LXC container, which took an average of 3.9s (+2.6%) to execute the script.

**Conclusions.** Taking into account both the criteria to evaluate the platforms, the information gathered and the results from the performance tests, it is possible to observe that LXC – in addition to being the slowest platform – requires a heavier configuration from the host side compared to Rkt and Docker. In particular, the work required to configure the container network is a consistent overhead to the deployment of the application. Moreover, the way LXC templates are distributed implies that not only a public service needs to be set in place, but also that a new template should be downloaded every time an update to the application is performed. Rkt and Docker, on the other hand, offer similar functionalities and performances, making the choice between them less obvious. However not only the Rkt CLI offers limited capabilities compared to the Docker API, but Rkt has also a much less capillary diffusion and user-base than Docker, it is less tested and it is used mainly in the context of CoreOS Container Linux, whereas Docker is widespread and battle-tested on a vast array of platforms. In the context of this project, Docker does not present any of the drawbacks identified in the other platforms, offers all the functionalities needed, including good performances and a convenient way to manage and distribute images. For all the reasons above, it is clear that Docker results the most suitable candidate to be used in TED.

[12] `https://github.com/containers/build`

## 4.2 Defense Measures Selection and Scoring System

The main functionality of TED is to verify whether certain protection mechanisms for binaries are in place, therefore selecting the appropriate measures becomes of critical importance to ensure the quality of the results. To choose the most suitable defenses, both their popularity and their efficacy have been taken into consideration, to avoid as much as possible false positive results. It is possible to divide the defense measures taken in consideration for being integrated in TED in three main categories: system defenses, ELF defenses against vulnerability exploit and ELF defenses against reverse engineering.

**System wide defenses.** This category includes those measures used by the kernel or applied to all the executables running in a given machine. Their purpose is to offer some level of protection to the environment in which the binaries run, without considering the defenses that each specific binary might or might not include. These include Address Space Layout Randomization (ASLR) and its analogue for the kernel (KASLR), hardware based memory protection (e.g., NX – No eXec, XD – Execute Disable) and a novel mechanism originally designed to increase kernel security, which later was recognized also as a protection from the Meltdown attack, KAISER/KPTI (Gruss et al., 2017). (K)ALSR and NX/XD have been chosen because they offer a good degree of protection from a wide range of memory exploitation attacks, including those belonging to the *ret2** family; however, more advanced attacks exists (e.g., ROP and derivatives) but the defense measures against this kind of attacks are not commonly found on systems and therefore have not been taken into consideration. Finally, KAISER/KPTI at the moment is the only existing defense against Meltdown for Linux systems, and given the critical impact of such attack, it has been included in TED.

**Executable defenses against vulnerability exploits.** This set includes the measures which are applied individually to each executable, for example during or after compilation time, and that aim to protect them from exploits that target vulnerabilities in the code. These measures offer a certain degree of protection independently from the system in which the binary is run. The most popular and commonly used techniques are software canaries, in particular the Stack Smashing Protector (SSP), the software based Data Execution Prevention and its corollary, the W^X (Writable XOR eXecutable) principle. Similarly to what it has been discussed previously, more advanced and novel protections for binaries have been designed, but none of this can be considered as a standard or as commonly deployed, and therefore they have not been integrated into TED.

**Executable defenses against reverse engineering.** The last category contains the protections aimed to counter or make more difficult the reverse engineering of a binary. The only measure which TED checks is the stripping of the binary. Stripping is an extremely weak defense, which makes the debugging of the program slightly more difficult, but it is the only common measure applied. The reason for which no other techniques are considered is that no standard binary protector for Unix systems exist, and that the focus of the research in this field is on software obfuscation. Unfortunately, TED is run on binaries whose content is in general unknown, and therefore it is not possible (or very complex) to verify whether the code is obfuscated or not. This limitation effectively leave stripping as the only measure verifiable by TED.

**Additional tests.** In addition to verify whether the measures mentioned are in place, TED also checks if the kernel is vulnerable to known exploits or to different Spectre variants. The rationale behind this choice is that this is not only a way to verify kernel binaries security, but also a decision which derives from the observation that generic binary attacks can have higher impact on machines where the kernel is vulnerable to exploits, especially to Privilege Escalation exploits. Finally, Spectre attacks could be used to compromise the machine leveraging hardware faults that are not mitigated by the defenses mentioned above (except for Meltdown).

## 4.3 Scoring System

To convert the information that TED gathers into a value that helps the TED user to make plans, two scoring systems are defined – a system score and an ELF score. Each score is computed on a total of 100 points, where a higher score implies a higher chance of exploitation, while 0 means that all the defenses that TED is able to verify are in place. The assignment of a score to each (lack of) defense is purely qualitative and decided after a careful consideration. The system score is so composed:

1. Kernel exploits (40 points). For each confirmed kernel exploit found, a score of $15/2^{n-1}$ is assigned, with $n \in [1,\infty]$, which represents the *n-th* exploit found. Each potential kernel exploit found will add $5/2^{n-1}$ points. Kernel exploits usually have a very big impact on the security of the machine. However, establishing a linear dependency between the number of exploits and the score would generate misleading results. For this

reason, the first exploit found will significantly increase the risk score, while each additional exploit will contribute with a continuously lower weight.

2. Spectre (20 points). Each variant of Spectre to which the system is vulnerable adds 3 points, while 8 points are added if the system is vulnerable to Meltdown. At the time of this writing, 5 variants of Spectre are known. Although Spectre attacks can lead to a severe compromise of the system, because there are not yet known exploits for them, the score assigned is relatively low. Meltdown contributes with a higher score because it is relatively simpler to exploit.
3. ASLR (20 points).The ASLR configuration can lead to a maximum of 20 points, depending on how the randomization is enabled. Specifically, 20 points are assigned if ASLR is disabled, 15 if it is partially enabled (i.e., only on stack, virtual dynamic shared object page, and shared memory), 5 if it is enabled also for the heap and data segment and 0 points if ASLR is enabled fully and additional patches have been applied. ASLR is an effective measure that should be enabled on all the systems, however, the different ASLR configurations restrict or enlarge the attack surface, and therefore are taken in consideration in the assignment of the risk score.
4. NX/XD (20 points). The lack of support from the CPU for the NX/XD bit on memory pages adds 20 points, while no points are added if its support is confirmed. The capability to restrict the execution of certain memory page is a very important measure to stop the simplest attacks and to build additional defenses, and therefore a significant score is assigned in case of missing support for NX/XD.

The binary score is composed as follows:

1. DEP/WˆX (50 points). If the binary is compiled with the stack marked as executable (violating DEP), 30 points are added. If there is any section both writable and executable (violating WˆX), 20 points are added. The possibility to write on the stack contributes with a big score as it enables the most basic attacks. The capability to write on an executable memory section is still a severe vulnerability but in general it is less trivial to exploit and therefore contributes with a lower score.
2. Canaries/SSP (40 points). If the binary is compiled without canaries and without using the Stack Smashing Protector, 40 points are added. Canaries are an orthogonal measure to DEP and also represent an important measure to detect exploit attempts. For this, the lack of canaries increases significantly the risk score of the binary.
3. Stripping (10 points). If the binary is not stripped, 10 points are added. The low score reflects the low effectiveness of stripping a binary to counter reverse engineering.

## 4.4 Implementation

The implementation of TED has been performed using mainly *Python* language, for which a library [13] which implements all the operations normally performed through the *Docker* command or API is available. It is important to note that TED is especially suitable for cloud environments because of several reasons: the first is that in such environments Docker is commonly present, allowing TED to be deployed in a matter of seconds; the second reason is that even though the tools run inside containers, TED can scan and analyze all the binaries in the system, including those inside other containers, without the need to modify the employed container images or their configuration. Moreover, the approach of TED to analyze both the system and the binaries is especially relevant in containerized environments since most of the container-escape strategies rely on kernel/system vulnerabilities. The program is composed by several modules that perform specialized operations. These operations include the acquisition of user input, the collection of the binaries to analyze, the execution of TED's checks, the collection of the results and the score computation. Each check that TED performs aims to verify whether one or more defense measures are in place, therefore they match the defense techniques mentioned in the previous section. In total, five different tests are implemented at the moment, each of which is run inside its own Docker container.

**Kernelpop test.** A custom version of Kernelpop[14] is used to verify whether the kernel is vulnerable to known exploits. This tool has been chosen because it provides fast and accurate results, it is maintained and its code is publicly available.

**Spectre test.** The bash script *spectre-meltdown-checker.sh*[15] is used to verify whether the system is vulnerable to 5 different Spectre variants. This script is comprehensive and actively maintained, and provides results in *json* format and therefore it is very suitable for being integrated inside TED.

**NX/XD support test.** The standard */proc/cpuinfo* is used to verify whether the CPU supports the *nx* bit. Given the fact that containers share the kernel with the

[13] https://github.com/docker/docker-py

[14] https://github.com/Sudneo/kernelpop

[15] https://github.com/speed47/spectre-meltdown-checker

host, the information obtained by */proc* virtual files is accurate and no external tools are needed.

**ASLR test.** In this test both the standard *sysctl* tool and a custom binary are used to detect the ASLR configuration active in the system. While the *sysctl* result provides the information from the kernel perspective, the custom binary uses a direct approach to observe the pattern in addresses assigned to the stack, to *malloc()* calls (heap) and to the environment variables. Analyzing these address and checking for repetitions allows to detect if ASLR is disabled or if it is enabled partially. The two perspectives are compared to provide a more accurate result for one of the most critical defenses.

**ELF test.** In this test *rabin2*, one tool which is part of the *radare2*[16] framework, is used to extract some information from the binaries, together with the standard *objdump* tool. The purpose of this test is to verify whether the binary has been compiled with DEP and W^X, whether canaries are used (and in particular the SSP) and if the binary is stripped. The first test consists in inspecting the flags of the binary sections/segments, the second consists in disassembling the binary and verifying the presence of canary check functions, while the binary is considered stripped if no symbol sections are found. *Radare2* has been chosen because it is one of the most popular tools for reverse engineering in Unix environment and it is actively maintained.

The main accomplishment of TED is not only to integrate and orchestrate the tools mentioned, but also to use containers to pack the needed tools with their dependencies in separate environments. This strategy allows to run easily these tools and also to use at the same time libraries or programs with different versions without creating conflicts. To provide such functionality, several Docker images have been built, while in the cases where a specialized image was not necessary the public *debian:latest* image has been used. The code of TED is publicly available and can be found on a public Github repository[17] and all the Docker images are available either on the public Docker Hub [18] or on the authors' Docker Hub page[19].

# 5 EVALUATION

To verify the feasibility of a container based approach for a security tool and the usefulness of the tool presented in this paper, several tests with different scopes and use-cases have been performed.

## 5.1 Performance Benchmark

The first test is a quantitative analysis with the aim of computing the overhead caused by running the needed tools inside containers. The four heaviest checks (excluding the NX/XD support check) that TED performs have been executed both inside a container and on the native operating system, measuring and comparing the respective execution times. Each check has been executed 100 times, and the average value has then been computed. The results are summarized in Table 1, where the average value for each different check is reported. It can be observed that the overhead added by running inside a Docker container is noticeable in relative terms; however, for each of these tests, a new container has been created and then destroyed. It was therefore possible to suppose that the overhead was caused by the Docker bootstrap routine. To confirm this hypothesis, the two checks with the major absolute difference between values have been modified to be executed inside the same container and the test has been repeated. The result obtained, also shown in Table 1, clearly confirms the hypothesis formulated. If the container bootstrap process is not factored in the execution time, the difference between running the checks inside containers or on the native machine becomes negligible. Moreover, the assumption of reusing a container for multiple tests does not represent a constraint, in fact, TED uses the same approach for its own execution. In conclusion, the performance analysis showed that executing the needed tools inside a container the way that TED does, causes an overhead in the order of few seconds per tool run, which in a non real-time processing context and compared to a global execution time potentially in the order of hours, can be considered negligible; therefore, the container based approach can be considered feasible from the performance perspective.

## 5.2 Functionality Tests

The second phase of testing included a set of real-world experiments to verify the usefulness of TED in real environments. For this, two main scenarios have been used: the first consisted of using TED to assess the security of the binaries present on a public web server and to consequently establish an action plan to resolve eventual issues found; in the second case, TED has been used on a portion of the infrastructure used for the NATO Locked Shields 2018 cyber exer-

[16] https://github.com/radare/radare2
[17] https://github.com/Sudneo/TED
[18] https://hub.docker.com/
[19] https://hub.docker.com/u/sudneo/

Table 1: *Summary result for the benchmark checks.*

| Check | Physical | Docker | Relative Δ | Absolute Δ | Docker no startup | Relative Δ | Absolute Δ |
|---|---|---|---|---|---|---|---|
| ASLR | 0.031s | 0.469s | +1418% | 0.438s | - | - | - |
| Kernelpop | 0.112s | 1.559s | +1284% | 1.447s | 0.134s | +16.4% | 0.022s |
| Spectre | 4.745s | 6.066s | +27.8% | 1.320s | 4.485s | -5.8% | 0.260s |
| ELF | 0.00060s | 0.00055s | -8.4% | 0.00005s | - | - | - |

cise [20] to determine whether it would be a good fit in a complex environment.

**Public web server.** TED full scan on the web server took 84m and 2s to complete and found the issues summarized in Table 2, where the actions that have been established to mitigate them are also reported. In Table 2 it is possible to observe that all the actions defined involve the system and the kernel, while no actions were planned for the ELFs compiled without defenses. The reason for this is multifaced; first, most of the binaries were used for local commands and were not facing the public. Second, none of the binaries had the *setuid* bit set, and finally, to take action and recompile the binaries, a more in depth inspection was necessary to make sure that recompiling the code with DEP or SSP enabled would not have broken any compatibility. For a matter of simplicity, further actions like configuring SELinux profiles or AppArmor for specific binaries, had not been considered. After performing the actions presented in Table 2, all the addressed issues have been resolved, and the following TED scan, which took 152m and 52s, reported a much lower risk level. It was therefore possible to conclude that TED allowed to effectively detect vulnerabilities in the system and also it facilitated the creation of an action plan composed of informed choices.

**Locked Shields infrastructure.** This test consisted of running TED on two sets of machines used during the cyber exercise: the first set included undefended machines, whereas the other set was composed by two machines which belonged to and had been defended by the winning team. The machines used for testing were web servers which run their applications inside Docker containers. It is worth mentioning that all the four servers used for this test did not have any Internet connection (after the game finished) and were accessible only through SSH once connected to the closed virtualized environment hosting the game-network. This condition presented the opportunity to successfully prove that an Internet connection does not represent a hard dependency for TED. Comparing the results of TED's scans on the first server of each set it was possible to observe that the blue team performed some updates (e.g., *sshd* and *apache2* binaries had been modified) and of course installed new software (e.g., *ossec*, *splunk*), but no major modifications to binaries to reinforce their defenses had been performed. One exception was represented by the *mysql* binary that scored 40 points in the undefended machine, but only 20 in the defended one, where canaries had been added. The most interesting observation is that in both cases the system score reported by TED was the same, in particular, the same kernel vulnerabilities were present in all the machines, among which the most interesting was the *dirtyCOW*. This exploit grants a privilege escalation and is especially relevant in contexts where applications run inside Docker containers, since it can allow attackers to break out of such containers [24]. The same observations could be done for the second pair of machines, where the exact same differences could be observed.

**Conclusions**. The results of the real-world tests showed that TED has very limited dependencies, can run without an Internet connection, effectively detects vulnerabilities in the system and helps creating an action plan to resolve them. Despite this, it might be needed to integrate more tools into TED to provide more detailed information, for example about what changed in each binary or whether each executable is malicious or not. It is reasonable to conclude that TED could fit in contexts such as The Locked Shields especially if the defending team knows on which binaries to focus (e.g., web server binaries), reducing the scope of the report and improving the capability to quickly assess the risk on meaningful binaries, to individuate possible system vulnerabilities and to correlate the findings.

# 6 CONCLUSIONS AND FUTURE WORK

In this paper a novel approach to vulnerability assessment – using software containers, is discussed, together with the selection process for the most suitable

[20] https://ccdcoe.org/largest-international-live-fire-cyber-defence-exercise-world-be-launched-next-week.html

[24] https://github.com/scumjr/dirtycow-vdso

Table 2: *TED's findings on the public web server.*

| Type | Description | Action |
|---|---|---|
| System | Kernel vulnerable to dirtyCOW and dirtyCOW poke variant | Upgrade kernel |
| System | Kernel vulnerable to CVE20177308[21] | Disable user namespace usage to unprivileged users. |
| System | Kernel vulnerable to CVE20162384[22] | Update kernel or restrict USB access. |
| System | Kernel vulnerable to CVE20176074[23] | Disable kernel DCCP module. |
| System | Machine vulnerable to Spectre variant 1,2 and Meltdown | Upgrade kernel, enable KPTI |
| ELFs | 71 binaries compiled without canaries of which 26 without NX | - |

containers platform, and the design, implementation and verification in multiple use cases of a tool, TED, which implements this approach. The planned improvements to TED include automated detection of malicious binaries through the integration of an antivirus such as ClamAV. Moreover, to improve the accuracy of the ELFs tests, the capability to analyze each binary to verify that no unsafe functions are used will be added. From the functionality perspective, despite TED runs all the checks it performs inside containers, the main application still runs on the native machine. Fully containerizing TED, including the supporting functionalities, will reduce the dependencies needed to the sole Docker platform, making the tool even more suitable for cloud environments, where containers are already used in many cases and where the Docker platform often is pre-installed, and would allow TED do be easily deployed with orchestration tools such as Kubernetes, increasing drastically the scope and the target audience. Finally, future work includes research on protections of binaries from reverse engineering, which at the moment are noticeably scarce, exception made for code obfuscation techniques.

# REFERENCES


Chen, X., Xue, R., and Wu, C. (2017). Timely address space rerandomization for resisting code reuse attacks. *Concurrency and Computation*, 29(16).

Choi, Y.-H., Park, M.-W., Eom, J.-H., and Chung, T.-M. (2015). Dynamic binary analyzer for scanning vulnerabilities with taint analysis. *Multimedia Tools and Applications*, 74(7):2301–2320.

Feng-Yi, T., Chao, F., and Chao-Jing, T. (2016). Memory vulnerability diagnosis for binary program. *ITM Web of Conferences*, 7.

Gruss, D., Lipp, M., Schwarz, M., Fellner, R., Maurice, C., and Mangard, S. (2017). Kaslr is dead: Long live kaslr. volume 10379, pages 161–176. Springer Verlag.

LXC-Doc (2018). Lxc official reference. `https://linuxcontainers.org/lxc/introduction/`. Accessed on 20/10/2018.

Marco-Gisbert, H. and Ripoll, I. (2013). Preventing brute force attacks against stack canary protection on networking servers. In *2013 IEEE 12th International Symposium on Network Computing and Applications*, pages 243–250.

NIST (2018). Nist nvd, vulnerabilities statistics. `https://nvd.nist.gov/vuln/`. Accessed: 10/06/2018.

Novark, G. and Berger, E. D. (2010). Dieharder: Securing the heap. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 573–584, New York, NY, USA. ACM.

Rkt-Doc (2018). Rkt official reference. `https://coreos.com/rkt/docs/latest/`. Accessed on 20/10/2018.

Solanki, J., Shah, A., and Lal Das, M. (2014). Secure patrol: Patrolling against buffer overflow exploits. *Information Security Journal: A Global Perspective*, pages 1–11.

Song, D., Brumley, D., Yin, H., Caballero, J., Jager, I., Kang, M. G., Liang, Z., Newsome, J., Poosankam, P., and Saxena, P. (2008). Bitblaze: A new approach to computer security via binary analysis. In Sekar, R. and Pujari, A. K., editors, *Information Systems Security*, pages 1–25, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ulrich, F. (2017). *Exploitability Assessment with TEASER*. Msc thesis, Northeastern University.

Wang, R., Liu, P., Zhao, L., Cheng, Y., and Wang, L. (2017). deexploit: Identifying misuses of input data to diagnose memory-corruption exploits at the binary level. *The Journal of Systems & Software*, 124:153–168.

Younan, Y., Pozza, D., Piessens, F., and Joosen, W. (2006). Extended protection against stack smashing attacks without performance loss. In *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*, pages 429–438.

# Appendix 9

**Publication X**
B. Blumbergs, R. Ottis, and R. Vaarandi. Crossed Swords: A Cyber Red Team Oriented Technical Exercise. In *18th European Conference on Cyber Warfare and Security, ECCWS 2019*, Coimbra, Portugal, July 2019. ACPI. (Accepted paper)

**Crossed Swords: A Cyber Red Team Oriented Technical Exercise**

Bernhards Blumbergs[1,2], Rain Ottis[2], Risto Vaarandi[2]
[1] CERT.LV, IMCS University of Latvia, Riga, Latvia
[2] Centre for Digital Forensics and Cyber Security, TalTech, Tallinn, Estonia
bernhards.blumbergs[a]cert.lv
rain.ottis[a]taltech.ee
risto.vaarandi[a]taltech.ee

**Abstract:**
This paper describes the use-case of international technical cyber exercise "Crossed Swords" aimed at training the NATO nation cyber red teams within a responsive cyber defence scenario. This exercise plays a full-spectrum cyber operation, incorporates novel red teaming techniques, tools, tactics and procedures (TTTPs), assesses team design and management, trains the skills for target information system covert infiltration, precision take-down, cyberattack attribution, and considers legal implications. Exercise developers and participants have confirmed the learning benefits, significant improvements in understanding the employed TTTPs, cyber-kinetic interaction, stealthy computer network infiltration and full-spectrum cyber operation execution.

**Keywords:** Technical Cyber Exercise, Cyber Red Teaming, Responsive Cyber Defence, Computer Network Operations

## 1. Introduction

The majority of exercises are cyber defence oriented with the defending blue team (BT) being the primary training audience and the attacking cyber red team (CRT) role-playing the adversary to provide the learning experience for the defenders (Leblanc, et al., 2011; Ogee, et al., 2015; Dewar, 2018; Fox, et al., 2018). However, technical exercises, oriented at advancing the readiness level and experience of a CRT, are lacking, limited in scope, not mentioned or described publicly (Lewis, 2015). To enable the development of defensive approaches, both sides – blue and red, must be exercised, especially if they are dependent on each other in real operations. To integrate and explore the concepts of responsive cyber defence (RCD), computer network operations (CNO), CRT techniques, tools, tactics and procedures (TTTPs), detection mechanisms, cyber deceptions, and cyber red team operational infrastructure, a unified environment is required. A technical cyber exercise oriented not only at training single facet of the CRT but implementing a full cyber operation environment would improve the CRT training experience. Additionally, CRT interactions and inter-dependencies with other operational entities, such as conventional kinetic or special operations forces (SOF), should also be explored to see how cyber operations fit within the larger picture. This paper defines and assesses the CRT oriented technical cyber exercise "Crossed Swords" since year 2014.
The main contribution of this paper is a detailed description of the CRT oriented technical cyber exercise "Crossed Swords" that has been conducted since 2014. To the best of our knowledge, no research papers on CRT exercise design have been published before, and this paper fills this gap. Also, "Crossed Swords" exercise has several unique design features, such as multi-disciplinary nature of the exercise and training audience, complex scenario which adds geo-political, strategic and cyber-kinetic dimensions to advanced technical challenges, and near real-time feedback to exercise participants via situational awareness system.
The remainder of this paper is organized as follows – section 2 provides an overview of related work, section 3 focuses on various design aspects of "Crossed Swords" exercise, and section 4 concludes the paper.

## 2. Related Work

Leblanc et.al. (Leblanc, et al., 2011) explore and analyse multiple war-gaming exercises and implemented exercise support tool-sets, as described in this paragraph. "CyberStorm" is the US Department of Homeland Security developed exercise with the aim of examining readiness and response mechanisms to a simulated cyber event. Participation is strictly limited to Five Eyes alliance members. "Piranet" is the French developed response plan and a simulation exercise of a major cyber-attack against France's Critical Information Infrastructure (CII). "Divine Matrix" is the India's war-gaming exercise to simulate a nuclear attack accompanied by a massive cyber-attack with kinetic effects against India. "Standoff", organized by PHDays conference, involves a competition without fixed scenario

between attackers, defenders, and security monitoring teams. Airbus commercial Cyber-Range platform hosts the playground for "European Cyber Week Challenge Final". Similarly, NATO Cyber-Range is used for running "Crossed Swords" exercise.
Mauer, Stackpole and Johnson (Mauer, et al., 2012) look at developing small team-based cyber security exercises for use at the university as a practical hands-on part within the courses. The research explains the management and roles of the engaged parties in the exercise creation and execution. The developed game network is comprised of a small set of virtualized machines used by a group of participants to attack, defend and monitor the event. DeLooze, McKeen, Mostow and Graig (DeLooze, et al., 2004) examine the US Strategic Command developed simulation environment to train and exercise CNO and determine if these complex concepts can be more effectively taught in the classroom. The simulation environment consists of "Virtual Network Simulator", comprised of two or more networked computers designed to represent attack effects in an interactive graphical environment, and the "Internet Attack Simulator", presenting a set of simple attacks, ranging from reconnaissance to DoS, available for launching against the network simulator's virtual network. Mostow and Graig confirm the benefit of CNO simulation exercises by measuring the increase of knowledge of the participants.
The issues tackled, in the related work for the cyber defence exercises are either narrow in scope, specific to a nation or small set of nations, restricted only to exercising just the decision-making process or a small subset of a full-scale cyber operation, or limited to just simulation of common cyber-attacks. Additionally, the majority of the exercises are delivered for the defensive capability building, and the CRT is either simulated or role-playing the adversary. However, a dedicated technical exercise for training CRT capabilities is required.

## 3. Cyber Red Team Exercise Design

Cyber exercise "Crossed Swords" (XS) (NATO CCD CoE, 2019), organized jointly by NATO CCD CoE and CERT.LV, is an annual international technical exercise oriented at training CRT with the latest technologies and striving to deliver highly realistic training. Since its inception, the exercise has grown in complexity and size. The exercise spans across three consecutive days representing a 24-hour fast-paced and intensive operation. More importantly, this exercise has served as a platform for implementing, testing, confirming, and conducting academic studies in the areas of CRT TTTPs, learning effectiveness, and near real-time situational awareness (Kont, et al., 2017).
The exercise is designed to implement the following cyber red team training objectives (TO):

1. Perform defended system compromise assessment, practice evidence gathering and information analysis for technical attribution, identify the origins of malicious activities and take actions stop them;
2. Execute a responsive cyber defence scenario for adversarial information system infiltration;
3. Employ stealthy attack approaches, and evaluate applicable TTTPs for fast-paced covert operations;
4. Exercise working as a united team in achieving the laid-out mission objectives;
5. Develop specialized cyber red teaming soft and technical skills needed for operation management, information flow, and target information system takeover; and
6. Explore and evaluate the full-spectrum operation's cyber-kinetic interdependencies.

The following subsections will provide a detailed description of the exercise.

### 3.1. Cyber Red Team Structure and Chain-of-Command

The exercise developers, execution managers, and the participants are allocated to various teams and sub-teams based on the specifics and activity focus area. The exercise has the following teams based on the area of operations: cyber-kinetic operations team (Red Team – RT), adversary and user simulation team (Blue Team – BT), exercise control and scenario management (White Team – WT), near real-time attack and situational awareness team (Yellow Team – YT), and game network infrastructure development and support team (Green Team – GT). As stated, the structure and chain-of-command for such cyber-kinetic operations has not been publicly discussed or disclosed by any nation; therefore, this exercise strives to experiment and uncover the organizational model providing simple chain-of-command and separation of duties.
The designed chain-of-command model for "Crossed Swords 2019" exercise is depicted in Figure 1, where the grey boxes represent the cyber red team at strategic, operational and tactical levels (with respective grey colour shading for every level), and the white boxes indicate the white team presence and assistance to the CRT. In the "Crossed Swords" exercise the CRT is divided into the sub-teams based on the expertise in technologies to be targeted (e.g.,


**Figure 1**: "Crossed Swords 2019" chain-of-command

web applications, network protocols). This exercise favours the speciality based sub-team creation to allow participant engagement throughout the exercise game-play and not only for explicit phases of the cyber operation. These teams are subdivided into sub-teams according to the specialization and operational management level:

1. **Red Team**: being the largest team of around fifty experts consists of exercise training audience. The chain-of-command and various sub-teams are the following:
    a. *Cyber commander.* The top-level officer in charge of commanding the cyber operation at a political level. Cyber commander, as part of the training audience, manages and coordinates the cyber operation to reach the set mission goals and coordinates the high-level activities, based on the desired effects, of the sub-teams;
    b. *Strategic adviser.* Is a member of the exercise development and management team (WT) with the role to provide the advice to the cyber commander, and, if needed, give minor hints to keep the red team activities on the course of designed scenario. This option allows exercise developers to explore the nuances and alternative paths for the developed scenario, allowing deviations if the result objectives are met;
    c. Red team leader group. This small group of experts, operating at an operational level and under direct command of the cyber commander, are responsible for fulfilling assigned operational effects by working with the sub-team leaders and ensuring that objectives, force protection, and intelligence activities are correctly executed and reached. This group consists of three experts: the overall red team leader, OPSEC officer, and an intelligence officer;
    d. *Sub-team leaders.* The main red team consists of five expert-focused sub-teams, at a tactical level, which are represented by their leaders. The purpose of every sub-team is to deliver the intended effects in their responsibility area. Over the exercise iterations it has been identified that sub-team size of 6-10 experts offers the best trade-off between required capability and management efficiency. The role and purpose of every sub-team is as follows:
        i. *Client-side attack sub-team.* This team focuses on executing attacks targeted at exploiting end-user (i.e., human vulnerabilities) to get the initial foothold, such as creating a *spear-phishing* campaign, setting up *watering-hole* or *drive-by* attacks. Once initial breach has succeeded this team ensures persistence in the target computer network;
        ii. *Network attack and exploit development sub-team.* The goal for this team is to target exposed computer network services, gain control over them by abusing the misconfiguration, poor implementation, abuse, or developing and exploiting software vulnerabilities. Additionally, this team is conducting IP network (IPv4 and IPv6) and service mapping, and executing attacks against specialized systems, such as tactical radio networks, mobile operator base stations, and ICS elements;
        iii. *Web-application attack sub-team.* For this team all web-based systems and technologies, such as web-applications, services, and back-end relational databases, are the target. This

team extracts valuable information from the web-applications, such as user credentials, e-mails, or application source code, as well as breaches the security of an exposed web-application to gain access to the internal network services, and establish persistence;

iv. *Digital battle-field forensics sub-team.* The main effort of this team is to perform data carving and artefact extraction from various sources, such as hardware devices (e.g., smart phones, portable computers and other electronics), computer memory or hard disk images. This team serves as the bridge between the cyber and kinetic operational components as it is tasked to perform analysis of forensic evidence extracted either by the cyber sub-teams or brought in from the field by the kinetic team; and

v. *Kinetic forces sub-team.* This team formed from trained military and law-enforcement experts performing various kinetic operations, including high interaction with the CRT capability, such as forced entry, covert access, hardware extraction, target capture or take-down, intelligence collection, surveillance, or kinetic activities on enemy territory. The interaction with the rest of the red team provides one of the key aspects for cyber-kinetic game-play. This team is managed and trained by industry experts (e.g., HTCI – High-Tech Crime Institute) and SOF instructors (e.g., NATO SOF School). The created scenario is designed to have the interdependencies within the CRT and anticipates the cyber-kinetic cooperation.

e. *Sub-team liaisons.* For every mentioned sub-team there is an attached WT liaison responsible for observing and, if required, providing minor hints to the sub-team leader to ensure that the team is not wasting too much time on some targets, such as cyber decoys and honeypots, and does not deviate from the intended scenario significantly; and

f. *Legal advisers.* Legal advisers are embedded to assist every level of the chain-of-command. The role of these experts is to provide their assessment of the activities from the international and domestic law perspectives.

2. *Blue Team.* A small team of up to four experts experienced in conducting cyber red team activities. This team is under direct control and supervision of WT and is used to manage the CRT progression within the adversary's computer networks. The main tasks for this team are user and adversary simulation. As a user simulation role-player, they are directly engaged in client-side conducted activities, such as examining and deciding to open received malicious attachments, visiting the web links, or browsing the in-game web services. Their task is to observe, assess and deny, or permit, the CRT initial foothold, based on the quality and delivery method sophistication level;
3. *White Team.* A small group of experts, typically no more than two, responsible for controlling and steering the exercise according to the developed scenario. As mentioned before, deviations from scenario are accepted and sometimes encouraged if the overall focus is not lost, and mission objectives can be reached. The exercise does not have the goal of succeeding in accomplishing the intended scenario and fulfilling entirely the laid mission goals by any means necessary. Depending on the activities pursued by the CRT, their course-of-action and time limitations, the mission might be a failure, as it can happen in real life. Within the past iterations of the exercise the red team only once successfully accomplished all the mission objectives, and in other cases completed them partially.
4. *Yellow Team.* This team focuses various areas of threat and anomaly detection, such as monitoring, big data analytics, intrusion detection, and situational awareness. The most crucial task for this team is to provide the near-real time situational awareness picture to the CRT from the perspective of neutral and hostile actors. This feedback allows the CRT to immediately spot mistakes and adjust their operations and tool usage to avoid detection, therefore not only increasing the level of stealth, but also having a better understanding on the used tools and performed actions; and
5. *Green Team.* Is responsible for tasks, such as maintaining the cyber range platform, supporting the game network technical requirements, developing the game network hosts and targets, and integrating new technologies.

### 3.2. Technical Environment and Technical Exercise Scenario

The “Crossed Swords” (XS) exercise game network is hosted on a cyber range running VMware ESXi hypervisor and it consists of around 200 virtual machines for in-game core networking, simulated Internet, CRT segment, and a set of target networks. Not all intended technical game-play elements can be virtualized, therefore the game network is expanded by connecting physical hosts and systems through the cyber range infrastructure. Before creating the overarching geo-political scenario, the technical scenario is established based on the core development team ideas and intended technical game-play intentions. Due to XS being relatively small, with respect to the game-network scale and training audience size, experimentation and introduction of new, recently prototyped, and unorthodox technologies can be afforded making the technical game-play more attractive and as close to the real-life as possible. The network also uses the traditional IT systems to provide the networking and common workstation operating systems, such as MS Windows and GNU/Linux, to provide replicate the structure of a regular office and business networks. The following list briefly summarizes some of the technologies introduced in the XS game series, to highlight the technical level:

1. Bunker door – control system employing a set of interconnected Siemens developed S7-1200 based PROFINET IO-devices. The CRT must reverse-engineer the communication protocol to inject remotely the commands controlling the bunker door;
2. Alarm system – protected premises by the Paradox alarm system, which must be targeted remotely by analysing the used bus-protocol, and capturing and decoding the PIN code;
3. CCTV IP camera – CRT has to find and exploit the flaws in the IP-based surveillance camera’s web interface to gain full control remotely;
4. Distributed power-grid – based on IEC-60870-5-104 industrial Ethernet protocol series and a Martem produced remote terminal unit (RTU) is used to manage and supervise the power-grid. The CRT has to reverse-engineer the protocol and perform remote command injection to control the power supply;
5. Unmanned aerial vehicle (UAV) – Threod manufactured UAVs flying over the protected territory must be targeted to gain control over the steering and video stream;
6. Unmanned ground vehicle (UGV) – Milrem developed UGVs serve as an adversary-controlled tank force and the cyber red team is tasked to take full control over them by targeting either the used network protocols or the controlling workstation;
7. Maritime navigation – a vessel’s steering and tracking system based on the AIS (Automatic Identification System) maritime protocol is targeted by the CRT to gain control over the ship and inject fake naval tracks;
8. Radio communication network – Harris-based military-grade data network must be infiltrated by the CRT by extracting the encryption keys;
9. Mobile network base stations – the cyber red team must infiltrate the LMT (Latvian Mobile Telephone) operator provided base stations connected to the actual mobile network, analyse and parse the intercepted communications to decode the adversary agent’s message exchange (SMS) and pinpoint their physical location; and
10. Railroad control station – a system based on Siemens created S7-1200 PLC running s7comm+ protocol, controls the in-game railroad network. The CRT is tasked to gain control over the railroad control stations to stop or derail the train.

The various technical challenges implemented across nearly all game-net systems, are designed in a way, that no single CRT sub-team can solve them on its own. Instead, cooperation, information exchange, objective tracking, and operation management is emphasized to provide the collaborative training experience and attempting to push the participants out of their comfort zones. The technical scenario, being time limited and fast-paced, cannot be fully solved, therefore the CRT has to consider ways and approaches on how to prioritize the technical objectives and manage the focus of force to accomplish the overall mission objectives within the exercise time.

The integration of real-life vulnerabilities and systems (Blumbergs & Vaarandi, 2017) (Blumbergs, 2019) deliver the learning perspective to the exercise participants. Examining, developing exploits, and attacking the systems which are widely used for automation and industrial process control are challenging and allow the training audience to comprehend the actual state of security for such industrial components. Furthermore, some participants might have such systems in their organizations, but are not allowed to execute attacks or tests due to them being in a production state. CRT members, with some guidance by the instructors, follow the full weakness identification, vulnerability

determination, and exploit development life-cycle. This allows the participants to successfully exploit the industrial control protocols and devices.

### 3.3. Geo-political Exercise Scenario

The technical scenario, describing the interdependencies, attack vectors, and alternative paths, only covers the part for the actual work to be conducted by the exercise participants. To deliver the context, reasoning, and clear objectives, the overarching scenario is required. This scenario provides the elements, such as the state-of-the-world background, geo-political situation, intelligence information on what has happened, why the response is being triggered, what are the objectives and rules of engagement. The main geo-political story revolves around a fictitious group of Cyberbian islands, where every island is a sovereign country with its technological advancements, political stance, alliances, and intentions. The three island-countries are Berylia, Crimsonia, and Revalia. Berylia being the smallest with a modest military force, part of NATO alliance, and its main economic income originating from the electronics manufacturing. Crimsonia is the largest island with a strong military, rich in natural resources, not part of any alliance, and is expressing some signs of aggression against its neighbouring island-countries. Revalia is a small, self-sustained, and politically neutral country. Within the scenario, the exercise participants assume the role of Berylian team, which is assembled to address the looming crisis. Every year, with a new exercise edition, the scenario evolves and the tensions between Berylia and Crimsonia have been escalating, ranging from Crimsonia conducting a series of debilitating cyber-attacks against Berylian CII, abuse of a neutral nation infrastructure, placing insiders and double-agents, forming military blockades, up to launching a military invasion of Berylia. The various levels of conflict are designed to explore the technical, cyber-kinetic, and legal game-plays as every scenario opens new opportunities and provides flexibility in conducting the responsive computer network operations. The operational environment for the kinetic force's unit is extremely important, as this restricts, or enables, some types of activities to be exercised.

### 3.4. Legal considerations

The legal aspects are incorporated in the form of legal scenario injects aimed to trigger the discussion and legal implication consideration by the command element. Legal advisers are assigned to the chain-of-command to assess and consult the exercise participants. The legal aspects of the conducted cyber-kinetic operations and applied TTTPs, within the context of the scenario, typically tackle at least the following legal considerations as covered in Tallinn Manual 2.0 (Schmitt, et al., 2017):

1. *Applicable law*. Depending on the scenario, the lawyers are tasked to ascertain which regimes of public international law apply to the cyber operations occurring during the exercise;
2. *States entitled to take countermeasures*. Only state affiliated institutions and organizations, such as military or intelligence, can conduct responsive activities on the state's behalf as long as the activities they engage in do not constitute an internationally wrongful act;
3. *Effect of RCD on third parties*. Since RCD has extraterritorial nature and implicates pursuing the adversary, as well as, performing malicious service take-down within the cyberspace, the legal advisers are required to assess the legality of the RCD effects on the third parties. For the CRT to complete their mission objectives, the RCD activities have to be deemed lawful;
4. *Limitations on RCD*. Depending on the legal qualification of the RCD operations, various limitations, such as concerning necessity, proportionality, imminence and immediacy, are attached to this operation. The legal advisers are tasked to identify any applicable limitations, such as requirements for the RCD to be necessary and proportional, and provide these legal implications to the commander or sub-team leaders;
5. *Self-defence against an armed attack*. The scenario is designed in such a manner that the severity of the offensive action against the victim state amounts to an armed attack, thus permitting to respond in self-defence with immediate asymmetric responsive cyber operations against a stronger and advanced adversary;
6. *Geographical limitations of cyber operations*. The effects of cyber operations have to be limited to the intended target information systems and geographical locations. This, although not always being possible to limit geographically, is taken into consideration by the CRT when executing the cyber operation which may include the activities, such as placement of drive-by exploit-kits on third-party services;

7. *Means and methods of cyber warfare*. The exercise scenario plays on the various levels of aggression and conflicts;
8. *Precautions*. For the executed cyber operations, the CRT is asked to exercise constant care, perform verification of targets, choice of means or methods, choice of targets, evaluate proportionality, and estimate the effects of cyber-attack whenever it is reasonably possible and applicable;
9. *Cyber operations in neutral territory*. The adversary may proxy their cyber-attacks or route the kinetic attack, such as drone flying through neutral state's air space before heading to the intended target. In such cases, the red team's response might have uncertainty and limitations on taken actions in the neutral state's cyberspace.; and
10. *False-flag and no-flag operations.* For the CRT to protect their identity, assets and intended objectives, a false-flag or no-flag operation could be considered to be executed to imply uncertainty and make attribution harder. From the technical perspective, the cyber red team might adapt the known TTTPs of a chosen threat actor to deceive the adversary. From the legal point of view, it is not clear if such operations are permitted when, for example, impersonating and adversarial profile of a threat actor with high certainty attributable to a third state.

**3.5. Training Assessment and Real-time Feedback**

One of the key aspects of the "Crossed Swords" exercise is to provide the environment, where the CRT can experiment, practice applicable TTTPs and observe their effects in near real-time. Such opportunity provides the necessary feedback to the exercise participants for their tool and procedure stealth and efficiency, as well as, to the exercise management to evaluate the progress of the CRT and the fulfilment of training objectives. To accomplish this, a dedicated framework, called the *Frankenstack* (Kont, et al., 2017), is developed to deliver the required visibility through meaningful visual means and notifications. The *Frankenstack* development is facilitated and coordinated by NATO CCD CoE since 2016. The development team is assembled from technical experts in the field of monitoring, data visualization, threat detection and assessment, and big data analytics. The contributions include NATO CCD CoE partners, such as Arc4dia, Stamus Networks (Suricata IDS), Greycortex, Cymmetria, Tallinn University of Technology, CERT.LV, and CERT-EE. The *Frankencoding* events (https://github.com/ccdcoe/Frankencoding) have resulted in an ongoing *Frankenstack* development with its source code released publicly on GitHub under the MIT license (https://github.com/ccdcoe/frankenstack).

The solution is easily deployable in the game network and can accept any possible sources of information to be further processed, which can be from at least the following origins:

1. *ERSPAN (Encapsulated Remote Switched Port ANalyser)* traffic mirror collecting all the network data recording, parsing, and deep packet inspection;
2. *NetFlow* from game network routers for traffic statistical analysis and evaluation;
3. *data from the systems*, such as system performance metrics (e.g., CPU load, HDD utilization, network interface card statistics), and logs (e.g., Syslog, and application textual log-files);
4. *honeypots and cyber decoys* placed in the network to attract and deceive the cyber red team into revealing its TTTPs; and
5. *aggregates the information from all sources* in textual format allowing this to be reduced to a log correlation and analysis problem.

During the "Crossed Swords 2017" execution the members of WT performed the assessment of the deployed *Frankenstack* solution for its usefulness and training benefits (Kont, et al., 2017). The identified findings were addressed and incorporated into the following exercise editions. The conducted expert qualitative interviews and online survey results reflected the following:

1. the deployed tools themselves do not increase the learning perspective, but is up to how red team members perceive and use the tools;
2. the addition of situational awareness solutions to the exercise is welcome and seen as a necessary component;
3. the four large screens in the execution room, showing the yellow team provided information, was preferred and checked approximately every 45 minutes by most of the training audience;
4. exercise participants also used the opportunity to access the *Frankenstack* dashboards locally on their computers and dig deeper when attempting new attack vectors;

5. *Alerta* tool, showing the identified attacks as priority categorized alerts, was found most useful by most of the trainees;
6. it was acknowledged, that ease of use should be further improved especially when considering the merger of high intensity technical exercise with monitoring tools not known to all participants;
7. majority of the training audience strongly agreed that the provided situational awareness was beneficial to the learning process, was accurate and delivered in acceptable speed;
8. the larger part of the training audience agreed that they learned more regarding how their actions can be detected and tried to be stealthier; and
9. integration of various tools into the *Frankenstack* has to be evaluated carefully to avoid visual distractions and making the output more self-explanatory.

**4. Conclusions and Future Work**

In this paper, we have presented the "Crossed Swords" exercise which involves intense game-play scenario and near real-time feedback, and explores novel concepts of CRT structure, cyber operation management and execution, and TTTPs applicability and stealth. For the future work, we plan to study the impact of the exercise on participant learning efficiency, on participant knowledge retention and change of perception about red team cyber operations, and on best practices for red team cyber operations.

**Acknowledgements**

The authors thank Liis Vihul and Joonsoo Kim for their valuable contribution.

**References**

Blumbergs, B., 2019. *Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems.* Prague, Scitepress, 5th International Conference on Information Systems Security and Privacy.

Blumbergs, B. & Vaarandi, R., 2017. *Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis.* Baltimore, IEEE Milcom.

DeLooze, L., McKean, P., Mostow, J. & Graig, C., 2004. *Simulation for training computer network operations.* WestPoint, IEEE Fifth SMC Annual Information Assurance Workshop.

Dewar, R. S., 2018. *Cyber Defense Report: Cyber Security and Cyber Defense Exercises,* Zürich: Center for Security Studies (CSS), ETH Zürich.

Fox, D. B., McCollum, C. D., Arnoth, E. I. & Mak, D. J., 2018. *Cyber Wargaming: Framework for Enhancing Cyber Wargaming with Realistic Business Context,* Massachusetts : The MITRE Corporation.

Kont, M. et al., 2017. *Frankenstack: Toward Real-time Red Team Feedback.* Baltimore, IEEE Milcom.

Leblanc, S., Partington, A., Chapman, I. & Bernier, M., 2011. *An Overview of Cyber Attack and Computer Network Operations Simulatio.* SanDiego, Proceedings of the 2011 Military Modeling & Simulation Symposium.

Lewis, J., 2015. *The Role of Offensive Cyber Operations in NATO's Collective Defence.* Tallinn, NATO CCD CoE.

Mauer, B., Stackpole, W. & Johnson, D., 2012. *Developing Small Team-based Cyber Security Exercises.* LasVegas, International Conference on Security and Management.

NATO CCD CoE, 2019. *Crossed Swords.* [Online]
Available at: https://ccdcoe.org/exercises/crossed-swords/

Ogee, A., Gavrila, R. & Trimintzios, P., 2015. *The 2015 Report on National and International Cyber Security Exercises: Survey, Analysis and Recommendations,* Athens: ENISA.

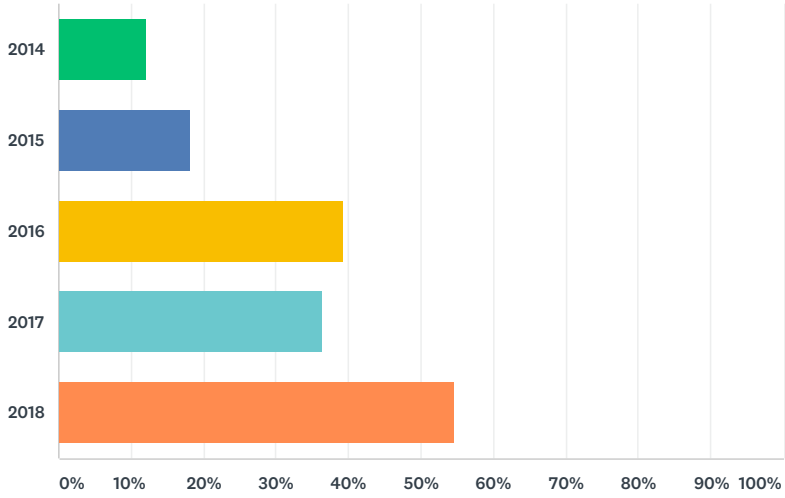
Schmitt, M. et al., 2017. *Tallinn Manual 2.0 on the International Law Applicable to Cyber Operations.* Tallinn: Cambridge University Press.

# Appendix 10

**“Crossed Swords” Exercise Feedback Survey Results**

# Q1 Year(s) of participation in NATO CCDCoE Crossed Swords exercise (formerly known as -- Red Teaming Workshop)

Answered: 33 Skipped: 0




| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| 2014 | 12.12% | 4 |
| 2015 | 18.18% | 6 |
| 2016 | 39.39% | 13 |
| 2017 | 36.36% | 12 |
| 2018 | 54.55% | 18 |
| Total Respondents: 33 | | |

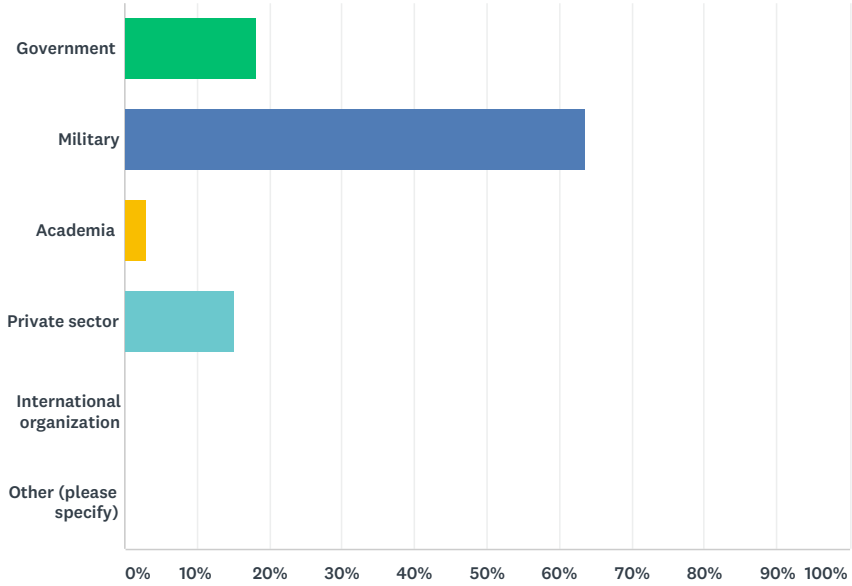
# Q2 Country represented at the exercise (in case of an international organization -- the name of the organization)

Answered: 33 Skipped: 0

| # | RESPONSES | DATE |
|---|---|---|
| 1 | Netherlands | 10/25/2018 5:27 PM |
| 2 | Czech Republic | 10/9/2018 10:47 AM |
| 3 | Germany | 10/4/2018 11:06 AM |
| 4 | Spain - ESP CYBERCOM (MCCD) | 9/25/2018 12:43 PM |
| 5 | Spain | 9/24/2018 9:03 AM |
| 6 | CERT.LV | 9/20/2018 4:51 PM |
| 7 | Austria | 9/20/2018 10:09 AM |
| 8 | POLAND | 9/20/2018 9:21 AM |
| 9 | Spain | 9/19/2018 2:45 PM |
| 10 | MOD NLD | 9/19/2018 1:13 PM |
| 11 | Netherlands | 9/19/2018 12:55 PM |
| 12 | Czech republic | 9/19/2018 10:42 AM |
| 13 | Netherlands | 9/19/2018 10:33 AM |
| 14 | LV | 9/18/2018 11:59 PM |
| 15 | Slovakia | 9/18/2018 8:44 PM |
| 16 | Lithuania | 9/18/2018 6:21 PM |
| 17 | Austria | 9/18/2018 5:29 PM |
| 18 | Poland | 9/18/2018 1:51 PM |
| 19 | Kudelski Security | 9/18/2018 1:47 PM |
| 20 | CZ (GreyCortex) | 9/18/2018 1:40 PM |
| 21 | LV | 9/18/2018 12:44 PM |
| 22 | Canada | 9/18/2018 12:39 PM |
| 23 | Slovakia | 9/18/2018 11:12 AM |
| 24 | Latvia | 9/18/2018 11:01 AM |
| 25 | Greece - HMOD | 9/18/2018 11:00 AM |
| 26 | USA - UTICA College | 9/18/2018 10:21 AM |
| 27 | Spain | 9/18/2018 9:47 AM |
| 28 | netherlands | 9/18/2018 9:34 AM |
| 29 | Germany | 9/18/2018 9:31 AM |
| 30 | Spain | 9/18/2018 9:23 AM |
| 31 | BHC Laboratory | 9/18/2018 9:11 AM |
| 32 | USA - HTCI | 9/18/2018 9:11 AM |
| 33 | BELGIUM | 9/18/2018 8:43 AM |

# Q3 Type of the represented institution

Answered: 33 Skipped: 0

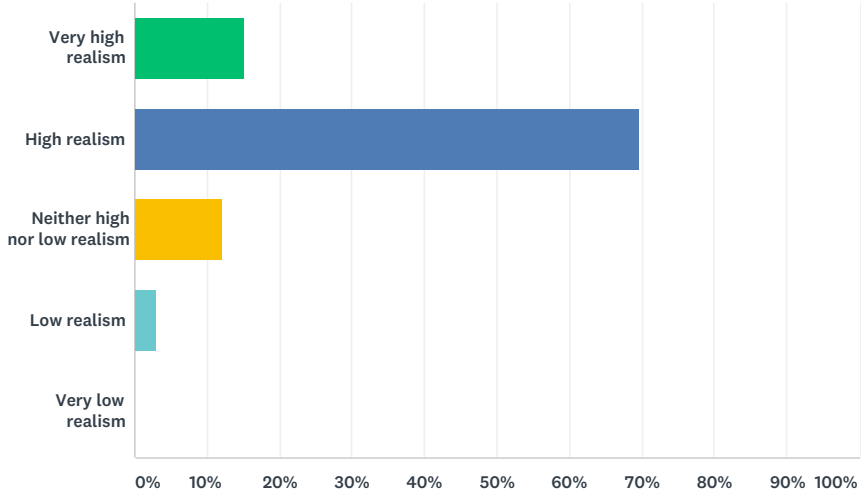



| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Government | 18.18% | 6 |
| Military | 63.64% | 21 |
| Academia | 3.03% | 1 |
| Private sector | 15.15% | 5 |
| International organization | 0.00% | 0 |
| Other (please specify) | 0.00% | 0 |
| TOTAL | | 33 |

| # | OTHER (PLEASE SPECIFY) | DATE |
|---|---|---|
| | There are no responses. | |

# Q4 Level of realism of the executed cyber operations at the exercise (Crossed Swords strives to provide the maximum training benefit and tries to replicate the real operational requirements as close as it is possible for a technical exercise)

Answered: 33 Skipped: 0



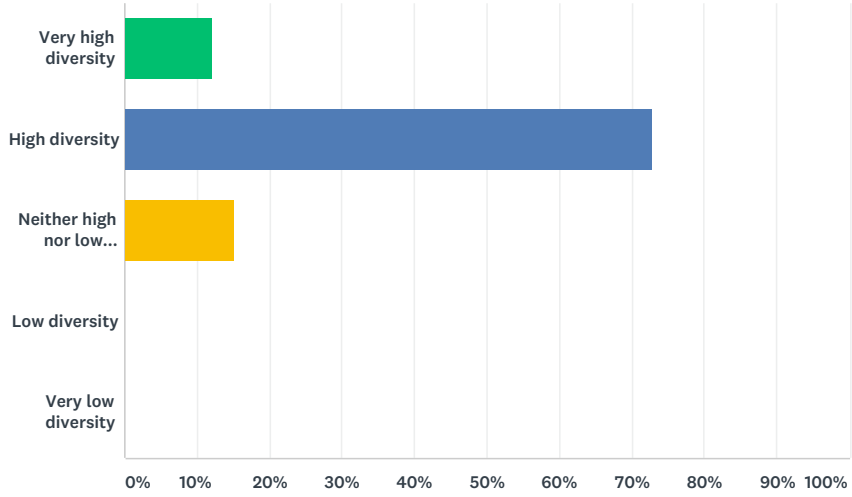

| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Very high realism | 15.15% | 5 |
| High realism | 69.70% | 23 |
| Neither high nor low realism | 12.12% | 4 |
| Low realism | 3.03% | 1 |
| Very low realism | 0.00% | 0 |
| TOTAL | | 33 |

| # | COMMENTS (PLEASE SPECIFY) | DATE |
|---|---|---|
| 1 | high realism on the infrastructure part, low realism on the team communication part, because in my opinion it is rather hard/impossible to get a fully functional team | 9/20/2018 4:51 PM |
| 2 | Technical yes, operational lack of processes not realistic in a military environment. | 9/19/2018 1:13 PM |
| 3 | The client side could have increased their level of sophistication of the execution of their part by being more stealthy and coordinated. This would then increase the level of realism. | 9/18/2018 8:44 PM |
| 4 | there where no actions from blue team side. | 9/18/2018 5:29 PM |
| 5 | Can´t assess, as I haven´t been in situation dealing international/large scale cyber attack. Nevertheless, exercise shows quite well how things should (or shouldn´t :) ) go in case of such incident (chain of command, cooperation between subteams/teamleaders/exercise leads etc.). It is really inspiring and useful in developing procedures for incident response. | 9/18/2018 11:12 AM |
| 6 | You always have to trade some realism to achieve effective training in such an environment. | 9/18/2018 10:21 AM |
| 7 | I took part in forensics spin off exercise | 9/18/2018 9:47 AM |
| 8 | The compresion of time to fully cover all phases of the compromise takes away part of the realism. | 9/18/2018 9:23 AM |
| 9 | It was high realism simulation. | 9/18/2018 9:11 AM |

| | | |
|---|---|---|
| 10 | as far as digital forensics / SOF is concerned at least | 9/18/2018 8:43 AM |

# Q5 Diversity of target systems implemented in the exercise network

Answered: 33 Skipped: 0

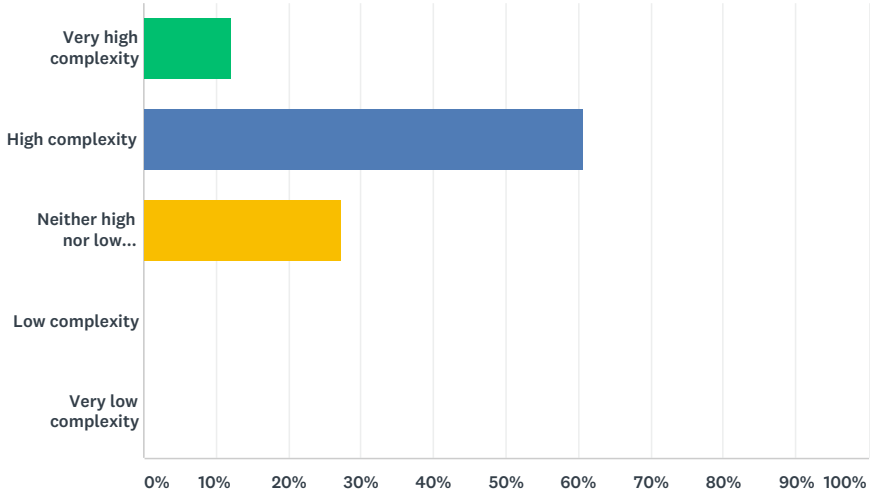



| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Very high diversity | 12.12% | 4 |
| High diversity | 72.73% | 24 |
| Neither high nor low diversity | 15.15% | 5 |
| Low diversity | 0.00% | 0 |
| Very low diversity | 0.00% | 0 |
| TOTAL | | 33 |

| # | COMMENTS (PLEASE SPECIFY) | DATE |
|---|---|---|
| 1 | being on the yt part, it's somewhat hard to judge, but having taken part in this for one year as a red, i'd say it was reasonably well balanced with targets. | 9/20/2018 4:51 PM |
| 2 | Compared to real life, or to other cyber exercises? :) Anyway, I think trere were quite lot of different things to play with, considering this is an exercise. I appreciate that exercise is evolving in this area, too - in 2018 there were definitelly more types of targets than in 2016. | 9/18/2018 11:12 AM |
| 3 | As forensics investigator, changing target for sources of evidence | 9/18/2018 9:47 AM |
| 4 | From my point of view, red team targets vere diverse enough to make necessary some kind of specialization on members of subteams | 9/18/2018 9:23 AM |
| 5 | Sometimes it seemed that diversity was neither high or low it can be improved. | 9/18/2018 9:11 AM |

# Q6 Technical challenge complexity level (taking into account the chain of attack, target system implementation and effort required for accomplishing the goal)

Answered: 33 Skipped: 0

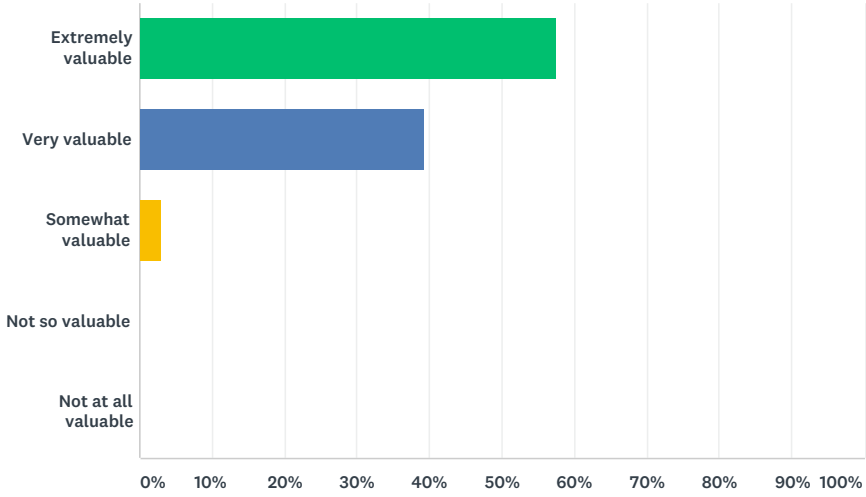



| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Very high complexity | 12.12% | 4 |
| High complexity | 60.61% | 20 |
| Neither high nor low complexity | 27.27% | 9 |
| Low complexity | 0.00% | 0 |
| Very low complexity | 0.00% | 0 |
| TOTAL | | 33 |

| # | COMMENTS (PLEASE SPECIFY) | DATE |
|---|---|---|
| 1 | and again, this comes out based only on one year as a attacker, there were targets that were a low hanging fruit, and then there were plc controllers where there was some time for research needed to understand what's going on there, also buffer overflows/ropchains :> | 9/20/2018 4:51 PM |
| 2 | for all specialisms high complecity, independences inbetween. | 9/19/2018 1:13 PM |
| 3 | A good combination | 9/19/2018 12:55 PM |
| 4 | The technical complexity was well balanced. I think the communication and coordination of operations had higher complexity. | 9/18/2018 8:44 PM |
| 5 | technical challenge for host team was low. | 9/18/2018 5:29 PM |
| 6 | Just right complexity. I remember that some targets were easy in the end, but to find them and find right way of exploiting took some time. Considering limited time of exercise execution, complexity level is IMHO satisfying. | 9/18/2018 11:12 AM |
| 7 | Some of the exercises part of the forensic scenario were just difficult because a hyped storyline | 9/18/2018 9:47 AM |
| 8 | One important thing to learn in this exercise is the necessity of internal collaboration so complexity is not high enough to make exercise impossible for a great majority of participants. | 9/18/2018 9:23 AM |
| 9 | It was very challenging and the tasks very very nicely delivered via chain of attack. | 9/18/2018 9:11 AM |
| 10 | clearly there has been an evolution (in the 2 years I participated / DF-SOF was implied). Some lessons identified have been learned, other / new ones are still on the todo list. | 9/18/2018 8:43 AM |

# Q7 Exercise benefits and training outcome value (the benefit of the exercise for your and development and experience, as well as joint operation in an international context)

Answered: 33 Skipped: 0



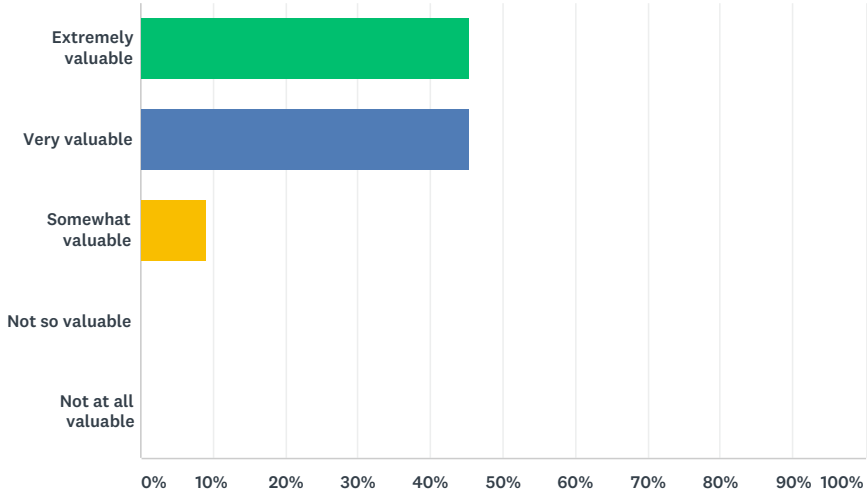

| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Extremely valuable | 57.58% | 19 |
| Very valuable | 39.39% | 13 |
| Somewhat valuable | 3.03% | 1 |
| Not so valuable | 0.00% | 0 |
| Not at all valuable | 0.00% | 0 |
| TOTAL | | 33 |

| # | COMMENTS (PLEASE SPECIFY) | DATE |
|---|---|---|
| 1 | brush up things one has forgotten, learning new things, brainstorming with people who know more than oneself, also getting a grip on teamwork/peopleskillz. | 9/20/2018 4:51 PM |
| 2 | Communication, coordination, leadership, feedback on stealthiness and detectability of operations. | 9/18/2018 8:44 PM |
| 3 | I felt that this provided the most value: the chance to work with other teams, share ideas and techniques, and practice clear communication during an urgent, time-limited exercise. | 9/18/2018 1:47 PM |
| 4 | I was there as an observer and believe the training was extremely valuable as many nations participated. | 9/18/2018 12:39 PM |
| 5 | Great experience, both in technical point of view and in operational/management: It is valuable opportunity to test your managing and communicational skills, and to realize how tough it can be to keep a picture of everything going on, as the exercise goes on with new injects/findings every few minutes. | 9/18/2018 11:12 AM |
| 6 | Training deficiencies were identified as well as ideas being generated for new TTPs. | 9/18/2018 10:21 AM |
| 7 | I personally learnt a lot on working in groups with participants from other countries but the participation of civilians (from civilian companies) as trainees (attacking) made it weird since they will not be involved in a real operation. The use of the force is strictly reserved to government personnel in NATO countries... | 9/18/2018 9:23 AM |

| | | |
|---|---|---|
| 8 | Red team based teamwork was a big benefit and exercise challenges did provide a personal improvements as well. | 9/18/2018 9:11 AM |
| 9 | cfr supra, lessons identified also for the internal (national) side | 9/18/2018 8:43 AM |

# Q8 The value of provided detected attack feedback to the red team (exercise develops and implements novel technologies for attack detection and near real-time information provision to the red team)

Answered: 33 Skipped: 0




| ANSWER CHOICES | RESPONSES | |
|---|---|---|
| Extremely valuable | 45.45% | 15 |
| Very valuable | 45.45% | 15 |
| Somewhat valuable | 9.09% | 3 |
| Not so valuable | 0.00% | 0 |
| Not at all valuable | 0.00% | 0 |
| TOTAL | | 33 |

| # | COMMENTS (PLEASE SPECIFY) | DATE |
|---|---|---|
| 1 | even if it is the low hanging fruit, people on the wire will notice it and one will get burnt, so it's not only on how good one is at the computer, it's also about how good one is to stay below the grid | 9/20/2018 4:51 PM |
| 2 | every detection was very well described and we were informed what changes w can make to improve. | 9/20/2018 9:21 AM |
| 3 | There were to many tools not known previously and was hard to figure out information | 9/19/2018 2:45 PM |
| 4 | deliverd al lot of lessons identified, specially back to the own organisation | 9/19/2018 1:13 PM |
| 5 | I was in yellow team. Feedback from red team was nice! | 9/18/2018 5:29 PM |
| 6 | '\|\|sleep(5)# BR :) | 9/18/2018 1:51 PM |
| 7 | I can't comment this, since I was yellow team :) | 9/18/2018 1:40 PM |
| 8 | No doubt it is really cool to see what´s going on, how your attacks are detected. By answering "Somewhat valuable" I mean that knowledge of us being detected did not influence our actions that much - as the time went, more and more focus was on completing the mission, even if our actions were detected. Generally, this feedback is very valuable as it makes you consider your actions more in real scenarios, outside cyber exercise. | 9/18/2018 11:12 AM |
| 9 | Best part of the exercise to me was to be able to see how my actions were appearing in different log and representation systems and the oportunity to adapt my attacks and try again with a low visibility profile. | 9/18/2018 9:23 AM |

| | | |
|---|---|---|
| 10 | This was best aspect since it is essential to move as quite as possible in red teaming missions. Helps to improve stealthiness. | 9/18/2018 9:11 AM |
| 11 | Not really applicable for digital forensics, since we were acting completely off-line | 9/18/2018 8:43 AM |

# CURRICULUM VITAE

## 1. Personal data

| | |
|---|---|
| Name | Bernhards Blumbergs |
| Date and place of birth | 17 February 1982, Riga, Latvia |
| Nationality | Latvian |

## 2. Contact information


| | |
|---|---|
| Address | Tallinn University of Technology, School of Information Technologies,<br>Department of Software Science,<br>Ehitajate tee 5, 19086 Tallinn, Estonia |
| E-mail | research[at]b2.lv |


## 3. Education

| | |
|---|---|
| 2013–2019 | Tallinn University of Technology, School of Information Technologies,<br>Cyber Security, PhD studies |
| 2003–2005 | Riga Technical University, Faculty of Electronics and Telecommunications,<br>Telecommunications, MSc |
| 2000–2003 | Riga Technical University, Faculty of Electronics and Telecommunications,<br>Electrical Engineering, BSc |

## 4. Language competence

| | |
|---|---|
| Latvian | native |
| English | C2 |
| Russian | B2 |
| German | B1 |
| Estonian | A2 |
| Japanese | A1 |

## 5. Professional employment

| | |
|---|---|
| 2013–2017 | NATO CCD CoE, Researcher |
| 2012–... | CERT.LV, Cyber security expert |
| 2003–2012 | Latvian National Armed Forces, Information systems engineer |

## 6. Training courses

| | |
|---|---|
| 2018 | Industrial Control System Penetration Testing |
| 2016 | Windows Kernel Exploitation |
| 2016 | SANS ICS410: ICS/SCADA Security Essentials |
| 2015 | SANS SEC770: Advanced Exploit Development for Penetration Testers |
| 2014 | SANS SEC573: Python for Penetration Testers |
| 2013 | SANS SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking |
| 2006 | Microsoft Certified System Administrator |
| 2006 | Microsoft Certified Professional |

**7. Computer skills**

- Operating systems: GNU/Linux, MS Windows
- Document preparation: Vim, LaTeX, Libre Office, MS Office
- Programming languages: Python, Bash, C/C++, Intel x86 Assembler
- Scientific packages: MATLAB

**8. Honours and awards**

- 2018, Ambassador for NATO CCD CoE
- 2000–..., Military awards

**9. Defended theses**

- 2019, "Specialized Cyber Red Team Responsive Computer Network Operations", PhD, supervisors Prof. Dr. R. Ottis and Dr. R. Vaarandi, Tallinn University of Technology, Department of Software Science, School of Information Technologies
- 2005, "Mobile IP and IPv6 Implementation for Network Roaming", MSc, supervisor Prof. Dr. A. Kavacis, Riga Technical University, Institute of Telecommunications
- 2003, "Use of Neural Networks for Secure Data Transmission Over Untrusted Networks", BSc, supervisor Prof. Dr. T. Celmins, Riga Technical University, Institute of Electronics

**10. Field of research**

- Cyber Red Teaming
- Network Infrastructure and Network Protocol Attacks
- ICS/SCADA Attacks
- Exploit Development

**11. Scientific work Papers**

1. B. Blumbergs, M. Pihelgas, M. Kont, O. Maennel, and R. Vaarandi. Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels. In B. B. Brumley and J. Röning, editors, *Secure IT Systems: 21st Nordic Conference, NordSec 2016*, pages 85–100, Oulu, Finland, November 2016. Springer International Publishing
2. B. Blumbergs and R. Vaarandi. Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 707–712, Baltimore, USA, November 2017. IEEE
3. B. Blumbergs. Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 88–99, Prague, Czech Republic, February 2019. SCITEPRESS

4. R. Vaarandi, B. Blumbergs, and E. Çalışkan. Simple event correlator - Best practices for creating scalable configurations. In *2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 96–100, Orlando, USA, March 2015. IEEE

5. R. Vaarandi, B. Blumbergs, and M. Kont. An Unsupervised Framework for Detecting Anomalous Messages from Syslog Log Files. In *IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, Taipei, Taiwan, April 2018. IEEE

6. A. Fárar, H. Bahsi, and B. Blumbergs. A Case Study About the Use and Evaluation of Cyber Deceptive Methods Against Highly Targeted Attacks. In *Proceedings of Cyber Incident 2017*, pages 1–7, London, UK, June 2017. IEEE

7. M. Kont, M. Pihelgas, K. Maennel, B. Blumbergs, and T. Lepik. Frankenstack: Toward Real-time Red Team Feedback. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 400–405, Baltimore, USA, November 2017. IEEE

8. M. Schmitt, L. Vihul, D. Akande, G. Brown, P. Ducheine, T. Gill, W. Heinstchel von Heinegg, G. Hernandez, D. Housen-Couriel, Z. Huang, E. Talbot Jensen, K. Kittichaisaree, A. Kozik, C. Kreiss, T. McCormak, K. Nakatani, G. Rona, P. Spector, S. Watts, and B. Blumbergs. *Tallinn Manual 2.0 on the International Law Applicable to Cyber Operations*. Cambridge University Press, Cambridge, UK, 2017

9. D. Mucci and B. Blumbergs. TED: A Container Based Tool to Perform Security Risk Assessment for ELF Binaries. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 361–369, Prague, Czech Republic, February 2019. SCITEPRESS

10. B. Blumbergs, R. Ottis, and R. Vaarandi. Crossed Swords: A Cyber Red Team Oriented Technical Exercise. In *18th European Conference on Cyber Warfare and Security, ECCWS 2019*, Coimbra, Portugal, July 2019. ACPI. (Accepted paper)

11. T. Väisanen, A. Farar, N. Pissanidis, C. Braccini, B. Blumbergs, and E. Diez. Defending mobile devices for high level officials and decision-makers. White paper, NATO CCDCOE, 2015

12. B. Blumbergs. Technical Analysis of Advanced Threat Tactics Targeting Critical Information Infrastructure. *Cyber Security Review*, pages 25–36, 2014

**Conference presentations**

1. B. Blumbergs, M. Pihelgas, M. Kont, O. Maennel, and R. Vaarandi. Creating and Detecting IPv6 Transition Mechanism-Based Information Exfiltration Covert Channels. In B. B. Brumley and J. Röning, editors, *Secure IT Systems: 21st Nordic Conference, NordSec 2016*, pages 85–100, Oulu, Finland, November 2016. Springer International Publishing

2. B. Blumbergs and R. Vaarandi. Bbuzz: A Bit-aware Fuzzing Framework for Network Protocol Systematic Reverse Engineering and Analysis. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 707–712, Baltimore, USA, November 2017. IEEE

3. B. Blumbergs. Remote Exploit Development for Cyber Red Team Computer Network Operations Targeting Industrial Control Systems. In *5th International Conference on Information Systems Security and Privacy, ICISSP 2019*, pages 88–99, Prague, Czech Republic, February 2019. SCITEPRESS

4. M. Kont, M. Pihelgas, K. Maennel, B. Blumbergs, and T. Lepik. Frankenstack: Toward Real-time Red Team Feedback. In *Milcom 2017 Track 3 - Cyber Security and Trusted Computing*, pages 400–405, Baltimore, USA, November 2017. IEEE

5. B. Blumbergs, R. Ottis, and R. Vaarandi. Crossed Swords: A Cyber Red Team Oriented Technical Exercise. In *18th European Conference on Cyber Warfare and Security, ECCWS 2019*, Coimbra, Portugal, July 2019. ACPI. (Accepted paper)

# ELULOOKIRJELDUS

**1. Isikuandmed**

| | |
|---|---|
| Nimi | Bernhards Blumbergs |
| Sünniaeg ja -koht | 17. Veebruar 1982, Riia, Läti |
| Kodakondsus | Läti |

**2. Kontaktandmed**

| | |
|---|---|
| Aadress | Tallinna Tehnikaülikool, Tarkvarateaduse Instituut,<br>Ehitajate tee 5, 19086 Tallinn, Estonia |
| E-post | research[a]b2.lv |

**3. Haridus**

| | |
|---|---|
| 2013–2019 | Tallinna Tehnikaülikool, infotehnoloogia teaduskond,<br>doktoriõpe |
| 2003–2005 | Riia Tehnikaülikool, elektroonika ja telekommunikatsiooni teaduskond,<br>magistriõpe |
| 2000–2003 | Riia Tehnikaülikool, elektroonika ja telekommunikatsiooni teaduskond,<br>bakalaureuseõpe |

**4. Keelteoskus**

| | |
|---|---|
| Läti keel | emakeel |
| Inglise keel | C2 |
| Vene keel | B2 |
| Saksa keel | B1 |
| Eesti keel | A2 |
| Jaapani keel | A1 |

**5. Teenistuskäik**

| | |
|---|---|
| 2013–2017 | NATO CCD CoE, teadur |
| 2012–... | CERT.LV, küberturbe ekspert |
| 2003–2012 | Läti Rahvuslikud Relvajõud, infosüsteemide insener |

**6. Koolituskursused**

| | |
|---|---|
| 2018 | Industrial Control System Penetration Testing |
| 2016 | Windows Kernel Exploitation |
| 2016 | SANS ICS410: ICS/SCADA Security Essentials |
| 2015 | SANS SEC770: Advanced Exploit Development for Penetration Testers |
| 2014 | SANS SEC573: Python for Penetration Testers |
| 2013 | SANS SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking |
| 2006 | Microsoft Certified System Administrator |
| 2006 | Microsoft Certified Professional |

**7. Arvutioskused**

- Operatsioonisüsteemid: GNU/Linux, MS Windows
- Kontoritarkvara: Vim, LaTeX, Libre Office, MS Office
- Programmeerimiskeeled: Python, Bash, C/C++, Intel x86 Assembler
- Teadustarkvara paketid: MATLAB

**8. Autasud**

- 2018, NATO CCD CoE saadik
- 2000–. . . , Sõjaväelised autasud

**9. Kaitstud lõputööd**

- 2019, "Specialized Cyber Red Team Responsive Computer Network Operations", PhD, supervisors Prof. Dr. R. Ottis and Dr. R. Vaarandi, Tallinn University of Technology, Department of Software Science, School of Information Technologies
- 2005, "Mobile IP and IPv6 Implementation for Network Roaming", MSc, supervisor Prof. Dr. A. Kavacis, Riga Technical University, Institute of Telecommunications
- 2003, "Use of Neural Networks for Secure Data Transmission Over Untrusted Networks", BSc, supervisor Prof. Dr. T. Celmins, Riga Technical University, Institute of Electronics

**10. Teadustöö põhisuunad**

- Punased meeskonnad küberturbes
- Infrastruktuuri ja võrguprotokollide vastased ründed
- ICS/SCADA vastased ründed
- Ründetarkvara arendus

**11. Teadustegevus**
Teadusartiklite, konverentsiteeside ja konverentsiettekannete loetelu on toodud ingliskeelse elulookirjelduse juures.