CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Cyber Threat Emulation

Exploring the capability of implementing a framework to represent known threat actor groups from public sources with offensive vulnerability assessment tools

Master's thesis in Computer Science and Engineering

Olof Magnusson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

Cyber Threat Emulation

Exploring the capability of implementing a framework to represent known threat actor groups from public sources with offensive vulnerability assessment tools

OLOF MAGNUSSON

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

A Chalmers University of Technology Master's thesis template for LaTeX
Exploring the capability of implementing a framework to represent known threat actor groups from public sources with offensive vulnerability assessment tools
OLOF MAGNUSSON

Supervisor: Miroslaw Staron, Software Engineering and Technology
Advisor: Teodor Sommestad, Swedish Defence Research Agency (FOI)
Examiner: Jennifer Horkoff, Software Engineering and Technology

Master's Thesis 2023
Department of Computer Science and Engineering
Division of Software Engineering and Technology
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2023

Abstract

The rising frequency of cyber attacks presents challenges in protecting confidential data. As technology continues to evolve, it empowers both attackers and defenders to enhance their capabilities to exploit and protect software systems. This dynamic fuels innovation in security measures as organizations seek new and effective methods to mitigate cyber attacks that aim to violate the confidentiality, integrity, and availability of information. To face emerging cyber threats, threat intelligence frameworks have been proven essential for gathering information on the motives, targets, and attack behaviors of threat actor groups. By connecting the information from these frameworks, organizations can facilitate realistic training environments to increase the excellence of security practitioners.

In this master thesis, the capability of linking attack techniques used by known threat actor groups and modules for attack execution is investigated. This is done by developing a mapping framework in an attempt to represent these techniques in SVED and subsequently instantiate this as an attack profile against a virtual organization in CRATE. The results indicate that when a specific CVE-entry (Common Vulnerabilities and Exposures) is prevalent in the incident description, there is a high likelihood of identifying an attack module for an attack technique. However, in the context of the experiment, the thesis has identified difficulties in emulating techniques by known threat actor groups. The evidence implies further research to explore more effective solutions for processing the modules in the attack profiles.

Keywords: computer science, software engineering, security, sved, crate, lore.

Acknowledgements

I would like to express my sincere gratitude to the Swedish Defence Research Agency (FOI) for providing me with the opportunity to collaborate with them on my thesis. In particular, I would like to thank Teodor Sommestad for guidance and feedback on the challenging problems faced in this master thesis. In addition, I would like to express my gratitude to Hannes Holm for explaining the technical systems and addressing my questions regarding Lore and CRATE.
Furthermore, I would like to thank my supervisor Miroslaw Staron and examiner Jennifer Horkoff at the Department of Computer Science and Engineering for honest and valuable input to improve the thesis.

Olof Magnusson, Gothenburg, November 2023

Contents

List of Figures

List of Tables

1
Introduction

In this section, we introduce the motivation and background necessary to understand the cyber security challenges that the work aims to investigate.

1.1 Motivation

Information technology has created a world of opportunities for organizations with the adoption of cloud computing that reaches beyond imagination. Enterprise systems are growing in complexity; devices, machines and particularly, sensors communicate and exchange information with an overwhelming diversity of services. While the benefits of this connectivity are genuinely incredible, it has greatly increased the attack surface. The large attack surface is a consequence of the rapid advancements in software and technology trends that we have witnessed in recent years. It covers every possible path that an adversary can use to gain unauthorized access to an application. The key step in reducing this surface lies in implementing code that effectively secures all data paths entering and exiting an application. By ensuring data flow remains secure in relation to the variables in the code, vulnerabilities can be minimized and unauthorized access attempts thwarted. However, in practice, developing such a software mechanism is complicated because some pathways are unknown to the developer. In addition, certain data paths may intentionally be left accessible to optimize performance or ensure compatibility with other software systems. These complexities and considerations highlight the challenges developers face in balancing robust security measures with the practical requirements of efficient system operation.

1.2 Background

Growing issues about keeping information private in the software environment raise several concerns for organizations. The evolving landscape of cyber threats presents considerable challenges when it comes to maintaining robust security measures. Notable cyber security breaches, such as the Uber data breach [1], Wannacry [2], and RockYou2021 [3], have gained attention due to their severity and sophistication. Most security incidents arise from malicious software that is designed to take advantage of vulnerabilities in an application. A vulnerability, in this context, denotes a weakness or flaw inherent in the design or implementation. This malicious software is hence crafted to induce undesirable behavior in the execution of a legitimate program to inflict damage to data resources, disrupt services, or expose sensitive

critical information. However, the issue lies in the fact that, as time has passed, malicious code or malware has become more complex, posing a challenge in its identification due to the advanced skills of those who deploy such harmful code. This complexity is evident in malware compile times, which suggest that threat actor groups have consistently updated their tools over the past several years. In threat intelligence, efforts are made to address this challenge by identifying and naming their activities. The process involves linking them with specific types of malware or agendas in threat intelligence frameworks like MITRE ATT&CK, VERIS, and STIX. These frameworks are widely recognized as state-of-the-art in the community for describing and grouping incidents related to information security.
As a response to the evolving landscape of cyber threats, threat emulator tools have emerged as a promising research area [4, 5, 6]. These tools offer a means to execute predefined scripts or tools that simulate real-world cyber attacks. Several studies [7, 8, 9] have highlighted their clear benefits, including reduced costs of penetration activities, smaller red teams, and automated security audits. By connecting information from the above threat intelligence frameworks, attack simulations can be trained on real-world observations that range from defense evasion, reconnaissance, privilege escalation, execution, and more techniques. Lore, specifically, is an offensive vulnerability assessment tool that uses boolean logic and trained models to automatically select and execute red team actions [10] to assess the security of operational systems.
The threat emulators stated above have been effective in understanding the potential harm under different attack assumptions [7]. This benefit indicates the value of using threat emulators to assess security risks and software defenses. However, because cyber threats emerge daily, it is essential to develop and test new threat profiles to understand adversary behavior. Security threats come from all possible directions, and modeling threat actors is fundamental in developing better incident management on how to respond to simulated threat agents. Based on limited research, this master thesis investigates how attack techniques from known threat actor groups described in public sources can be represented with offensive vulnerability assessment tools.

1.3 Problem

In this thesis we address the problem of representing threat actors described in public sources. The main purpose of this research is to develop profiles that reflect known threat actors described in public sources so that these threat actors can be represented with offensive vulnerability assessment tools for cyber defense exercises. This is done by developing a mapping framework that link attack techniques from known threat actors to software modules for attack execution. The framework is then tested through an experiment to assess the extent to which attack techniques of known threat groups can be emulated. Hence, the research questions are as follows:

- **R1**. **To what extent is it possible to model threat actor profiles described in public sources?** Addressing this research question provides insight into what extent offensive vulnerability assessment tools can represent known threat actor groups described in public sources for cyber defense exercises.

- **R2**. **To what extent is it possible to emulate the attack techniques of known threat actors described in public sources?** Addressing this research question provides insight into what extent offensive vulnerability assessment tools can execute attack techniques from known threat actor groups described in public sources for cyber defense exercises.

1.4 Delimitations

An important aspect is that virtualized machines or systems in cyber ranges are a collection of distinct computers. In theory, any attack sequence from a known threat actor group can be fully replicated by employing the precise malicious file or code used in the attack. However, this approach is complicated because of the reverse engineering challenges. Furthermore, the large volume of adversarial data collected in threat intelligence frameworks indicates that investigating all threat actor groups may not be feasible.
Because this paper focuses on the mapping process between attack techniques and modules for attack execution, setting up the environment and implementing new functionality in CRATE [11] is beyond the scope of this study. Finally, as part of our evaluation process, we will assume that every machine in the network has weak software protection, thereby granting us the means to bypass security measures such as firewalls and antivirus protection.

1.5 Thesis organization

The thesis is structured as follows. Chapter 2 identifies the theoretical concepts that are necessary to understand the details of the thesis. Chapter 3 identifies the current state-of-the-art in threat modeling and examines previous work in the field. Chapter 4 identifies the methodology used in this study, while Chapter 5 reports on the results obtained. Chapter 6 concludes the thesis with a discussion of the findings, while Chapter 7 examines the validity threats identified. Finally, Chapter 8 presents a conclusion that answers the research questions.

2

Theory

In this section, we introduce the concept of cyber security exercises and threat emulators with an emphasis on the red team [12] emulation tool Lore. Additionally, we provide an overview of threat intelligence platforms and their role in describing incidents related to information security.

2.1 Cyber security and challenges

The rising frequency of cyber attacks has forced organizations to look for innovative approaches to improve their defense capabilities and assess potential software risks. However, the most common problem observed in enterprise systems is the ability to reproduce cyber attacks in a safe training environment. Although the ideal solution is a dedicated testing environment that can virtually represent a large system, this option is usually too expensive and resource constrained for most organizations. As an example, Locked Shields and similar enterprise vulnerability assessments have demonstrated that it takes weeks to prepare a cyber defense exercise (CDX) [13].

2.1.1 Cyber ranges

Infrastructure and resources for replicating sessions involving cyber threats are vital in the development of a cyber defense exercise. In response to this, numerous studies have put forward the concept of cyber ranges to study and analyze cyber threats [14, 15, 16, 11]. A cyber range can be defined as a specialized computing facility that combines a variety of virtual machines and physical devices to represent a large organization or system. Some of the cyber ranges will be described in further detail.

2.1.1.1 KYPO

Masaryk University, located in the Czech Republic, maintains the cyber range called KYPO [16]. This cyber range operates on the OpenStack cloud platform and provides a robust infrastructure to effectively manage and conduct realistic attack simulations against a cluster of virtual servers. KYPO was developed with a primary focus on cyber research, development, and testing, as well as digital forensic analysis and training in the field of cyber security.

2.1.1.2 RGCE

As part of JAMK University of Applied Sciences in Finland, the JYVSECTEC cyber security Center (Jyväskylä Security Technology) manages the cyber range RGCE (Realistic Global Cyber Environment) [17]. The cyber range combines virtualization techniques, physical devices, and domain-specific systems for training, research, or development needs. RGCE was developed primarily to simulate organizational environments or external *Internet users* controlled from a centralized system.

2.1.1.3 CRATE

The Swedish Defence Agency (FOI) is a research institute in Sweden that develops and maintains the cyber range CRATE (Cyber Range And Training Environment) [18, 11]. CRATE is made up of a cluster, which is a collection of servers and resources that virtualize over five to twenty nodes depending on the operating machine. In this architecture, nodes are represented in different networks called organizations. The organizations are divided into multiple segments that consist of virtual machines that operate on varying operating systems with applications and services. Figure 2.1 provides an illustration of a virtual environment in CRATE featuring six distinct types of organizations. Within these organizations, virtual machines may have software configured to replicate specific vulnerabilities, versions, and services, thus creating a simulation of a real-world system.

2.1.2 Cyber security exercises

The fundamental concept of a cyber security exercise is to implement a scenario to safely execute malicious code. These exercises are designed to create realistic and sophisticated simulations with various types of cyber threats, including malware, denial of service, and social engineering tactics. By participating in these exercises and executing attacks within the scenarios, organizations can significantly improve their excellence in handling security incidents and identifying vulnerabilities [19].

2.1.2.1 CDX

A common form of exercise used to improve security awareness is cyber defense exercise (CDX). This exercise serves multiple purposes, including training incident response, testing defense capabilities, and providing individuals with an opportunity to practice protecting IT systems. In a typical setting, participants are divided into two teams: a red team and a blue team. Both teams have the ability to assume the adversary role and target the opposite team within the CDX. More commonly, this format is referred to as live-fire exercises in the community.

2.1.2.2 Capture the flag

Another widely represented exercise is Capture The Flag (CTF). CTF involves various scenarios in which the objective of the red team is to attack and find vulnerabilities represented as tokens or artifacts. In contrast, the blue team is tasked with protecting these artifacts by monitoring flows and events in log management tools.

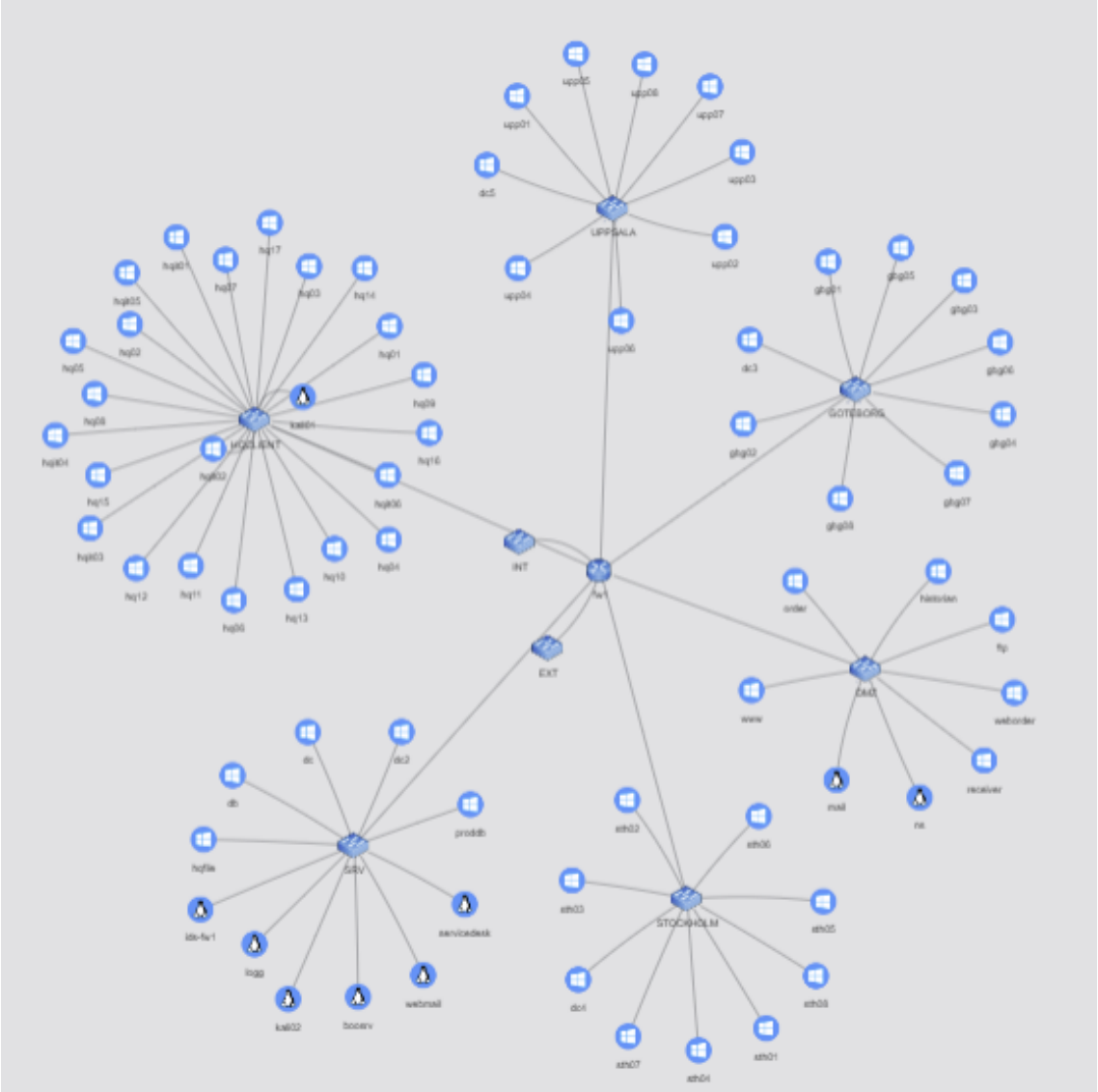

Figure 2.1: Example of a virtual environment in CRATE that represent six segments of virtual nodes.

To measure adversary success in a CTF, a scoring system is usually deployed to assign points to each captured token with varying difficulty levels [20].

2.2 Threat emulators

Threat emulators are tools that are used to execute attacks within cyber defense exercises. They are built using a collection of scripts or programs that are constructed to simulate the techniques used by adversaries in actual cyber attacks. The use of threat emulators in cyber security practices provides several advantages over relying solely on hiring security professionals for software system audits. Some of these advantages include greater diversity in exercises, reduced cost of penetration activities, and increased efficiency in auditing software systems [7].

2.2.1 Emulation tools

There are several emulation tools, such as Lore [6], Caldera [21] and Infection monkey [22], which have built-in functionality to automatically execute offensive network assessments. The open-source tools Caldera and Infection Monkey share similarities in that both require the installation of a victim agent before usage and execute commands on controlled machines in an effort to spread laterally. Caldera is developed based on techniques from the MITRE ATT&CK framework and furnishes a platform for attack simulation. The platform includes a vast library of red team actions such as Access, Atomic, and Builder, which contain multiple payloads to test defense mechanisms and incident response capabilities. Similarly, Infection Monkey simulates real-world attacks by deploying multiple propagation techniques and exploits that involve predefined passwords, common logical exploits, and password-stealing software [23].

2.2.2 Lore and SVED

Lore is a tool developed by the Swedish Defence Agency (FOI) to train blue teams in cyber defense exercises. The software architecture comprises a graphical web application, programmable interfaces, and a command line interface [6].
Technically, the tool is developed on an OODA (Observe, Orient, Decide, Act) architecture [6]. The OODA loop is a four-stage process that includes logging activities as they are carried out (observe), analyzing the logs (orient), making decisions based on the current state (decide), and finally, executing the decisions (act). This procedure is repeated until the desired outcomes of the cyber defense exercise are obtained or until the action space in the scenario is exhausted. As of current writing, Lore supports two scenarios that involve compromising IPv4 addresses and obtaining flags on a compromised machine. To enable the setup of these configurations, Lore uses the graphical interface that allows an operator to select a set of this functionality and the command-line interface to execute commands.
SVED (Scanning, Vulnerabilities, Exploits, and Detection) is used to enable programmatic interaction with modules in red team tools. Lore develops its understanding of the situation by parsing the logic from these tools and maintains a

record of the activities in the environment with a red team database. The database uses states that include `INITALIZED`, `SUCCESSFUL`, and `FAILED` to denote the outcome of the operation. This represents the current state of the nodes or machines with a logical decision in the CRATE environment. The boolean selection process will evaluate whether the attack was successful when the attack was deployed against the virtual servers. Thus, an attack sequence evaluated as successful is equivalent to a successful attack in the real world.

2.3 Threat intelligence platforms

A number of frameworks have been developed to create a standardized language to describe and group security incidents, such as VERIS, STIX, and MITRE ATT&CK. These frameworks will be described in further detail.

2.3.1 VERIS

The Vocabulary for Event Recording and Incident Sharing (VERIS) is a platform developed to address the issue of insufficient quality information in security incidents. It uses labels and variables to share incident details and allows customizable associations with actors, actions, assets, and attributes. As an example, it provides information on a security incident where an actor (Internal) executes an action (Malware) that results in an impact on a specific asset (Hosting) and compromises an attribute (Confidentiality). Table 2.1 illustrates the possible associations.

Table 2.1: The four components in the VERIS threat model.

Actor	Action	Asset	Attribute
Internal	Hacking	Variety	Confidentiality
External	Malware	Ownership	Integrity
Third-party	Social	Management	Availability
	Misuse	Hosting	
	Physical	Accessibility	
	Error	Cloud	
	Environmental	Notes	

2.3.2 STIX

Structured Threat Information Expression (STIX) is a standardized language that is developed to provide a structured framework to describe events and exchange intelligence about cyber threats. It connects multiple objects, including observables, indicators, malware, campaigns, and threat actors, through relationships [24].

- **Observable**: This object provides information about events that occurred on a network or host, including details such as the presence of artifacts, the detection of running processes, or the capture of network packets.

- **Indicator**: This object contains specific evidence that can be used to identify an attack, including suspicious email addresses, domain names, or other indicators that describe malicious activity.
- **Threat Actor**: This object describes the individual or group responsible for executing the attack. It includes information on their motives, capabilities, and any known affiliations. Additionally, it can be linked to the *Identify* object that specifies more details of the threat actor that make use of the attack.
- **Malware**: This object contains different types of malware, such as Trojans, keyloggers, and other forms of malicious code that might have been used in the attack.
- **Attack Pattern**: A TTP (Tactics, Techniques, and Procedures) category that outlines the methods that the adversaries uses to compromise targets.
- **Campaign**: This object represents a series of interconnected attacks orchestrated by a specific group of threat actors.

These interconnected STIX objects provide a comprehensive understanding of a security incident. The *Indicator* object establishes a relationship between the *Threat actor* and the specific indicators of the attack based on the *Observed Data* object. The *Threat actor* is linked to the specific malware used, while the *Campaign* and *Attack Pattern* objects provide identifier of the attack and the threat campaign.

2.3.3 MITRE ATT&CK

The Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) framework is a knowledge base of adversarial tactics and techniques based on observed incidents in actual cyber attacks. The framework consists of fourteen categories called *tactics*, each including subcategories known as *techniques*. These tactics and techniques together form what is commonly referred to as the ATT&CK Matrix. A description of a tactic indicates the objective of the attack and is provided along with a technique executed by a threat actor. Table 2.2 illustrates how descriptions, techniques, and threat actor groups might be interconnected.

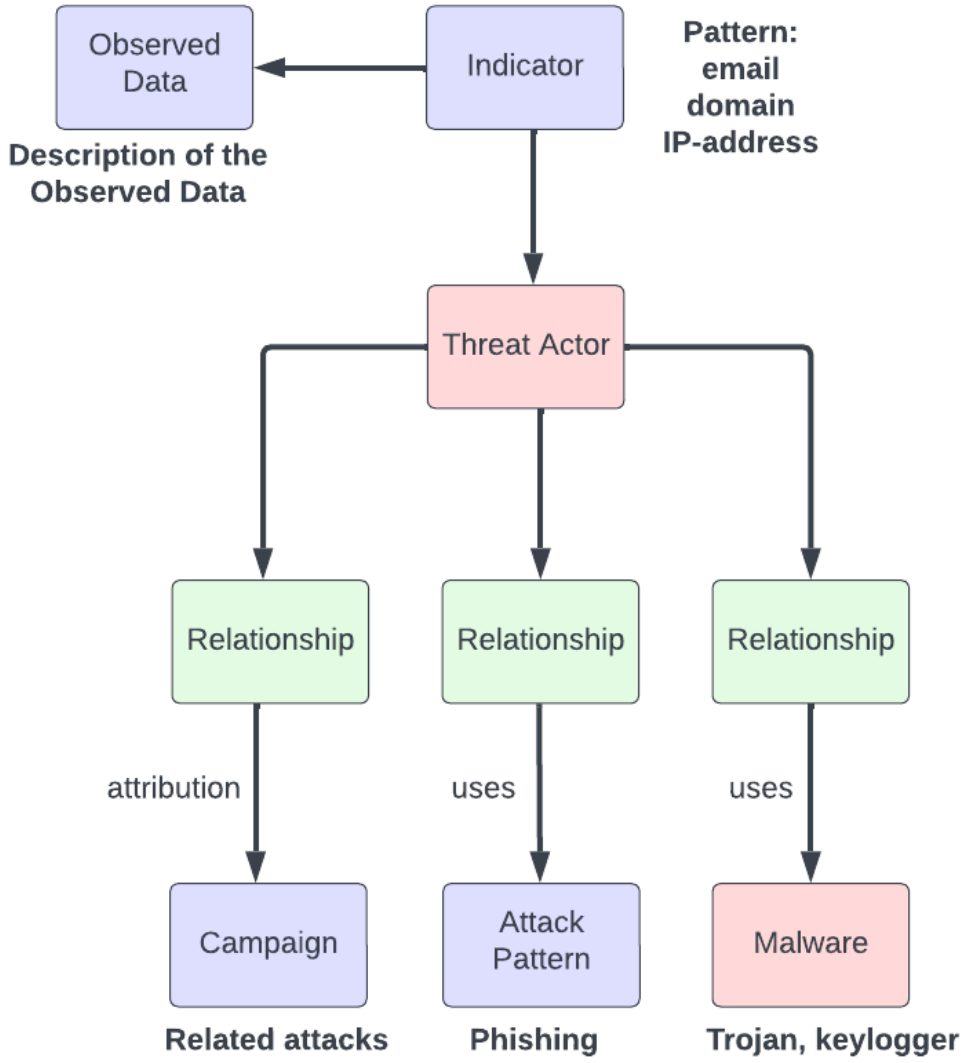

Figure 2.2: Phishing attack illustrated with the STIX threat model. We use colors and icons from the STIX language to provide a more consistent representation.

Table 2.2: Detailed description of each tactic, outlining the objectives and techniques employed by attackers in MITRE ATT&CK.

Tactic (Objective of the attack)	Description	Technique	Threat Actor
Reconnaissance	Plan future operations	Active Scanning	TeamTNT
Resource Development	Establish resources	DNS Server	HEXANE
Initial Access	Trying to get into the network	Phishing	Axiom
Execution	Trying to run malicious code	Powershell	APT28
Persistence	Main their foothold	Account Manipulation	Lazarus Group
Privilege Escalation	Higher-level Permission	Bypass User Account Control	Bad Rabbit
Defense Evasion	Avoid Being Detected	Deobfuscate	BendyBear
Credential Access	Steal Account Names and Passwords	Brute Force	Caterpillar WebShell
Discovery	Figure out your environment	Account Discovery	ShimRatReporter
Lateral Movement	Move through your environment	SSH Hijacking	MuddyWater
Collection	Gather data to reach their goal	Adversary-in-the-Middle	Kimsuky
Command and Control	Take control	Data Encoding	H1N1
Exfiltration	Trying to steal data	Exfiltration Over Physical Medium	AuTo Stealer
Impact	Manipulate, interrupt or destroy data	Data Destruction	PowerDuke

3
Previous work

In this section, we introduce the state-of-the-art in threat modeling and previous work in this field with an emphasis on attack-graph-based and threat emulation tools. Finally, we will highlight this thesis contribution.

3.1 Threat modeling

Threat modeling is a process in cyber security that involves analyzing the impact of potential threats to identify weak points [25] in enterprise systems. In principle, it is used in organizations to secure software systems in the design phase by proactively working on their approach to security. Prior work on threat modeling is already well established. According to a systematic review of the literature [26], threat modeling methods are classified into manual modeling [26, 27], automatic modeling [28, 29], formal modeling [26, 29], or graphical modeling [27, 30, 31]. In the context of graphical modeling, attack trees are primarily used to create conceptual diagrams of possible paths on how a system might be attacked by an adversary. The method offers a systematic approach to construct attacks by breaking down the potential steps or actions to compromise a certain function or aspect of a system.

3.1.1 Attack graph tools

In recent years, there has been a growing interest and research focus on applying attack-graph-based tools in threat modeling. The concept is to extract attack paths from a specific system setting in order to measure and manage cyber security. There is a wide range of approaches that use attack-graph-based tools with some degree of formalism. For example, MulVAL adopts Datalog [5] and computes different attack paths using probabilistic methods. k-Zero Day Safety [32] extends MulVAL in the computation of zero-day attack graphs. P^2CySeMoL [4] is an attack-graph tool based on CySeMoL [33] that uses a bayesian network to map relationships between attacks and defenses. NAVIGATOR [34] uses attack graphs with graphical user interfaces to visualize the effect of a variety of attack techniques. NetSecuritas [35] uses a scanner output and known exploits to generate attack graphs to target specific system settings. Pwnpr3d [36] provided an approach for automated attack graph generation. While the above studies motivate how attack-graphs tools are used for assessing vulnerabilities, the approaches are limited in the number of techniques found in the threat intelligence platforms.

3.1.2 Threat simulation

There are several studies that have focused on the design and implementation of attack graph tools with a focus on the techniques from the ATT&CK framework. For instance, a study [37] investigates an automated testbed to emulate adversary behavior on a Windows enterprise network. Another practical tool is Caldera [21], which develops models and analyzes adversary success by running attacks in an automated system setting. Similar to practical tools, Lore [6] is a network assessment tool that automatically selects and executes red team actions. The tool uses a large number of tools in the library of actions and has coverage of ATT&CK techniques in the default automation profile [10].
On the side, a study [38] proposed a threat modeling language based on the ATT&CK Matrix for enterprise system models in the domain-specific Meta-Attack Language (MAL). However, the proposal models actions on a high abstraction level that cannot be easily translated into an attack simulation environment.

3.1.3 Thesis contribution

While prior work has focused on model attack techniques and defenses to provide architectural support, there is limited attention to investigating how attack techniques from public sources can be connected to similar constructs in software modules. As a result, we see a research gap where the usability and effectiveness of threat actor models are not satisfactorily considered. Because the offensive vulnerability tool Lore has the capability to execute a variety of techniques similar to those described in public sources, it provides new opportunities to research to which extent techniques from threat actor groups can be connected into modules for attack execution.

4
Method and Procedures

In this section, we introduce the research methodology used in this study to investigate how attack techniques from known threat actors can be translated into modules for attack execution.

4.1 Research Methodology

The research methodology for this thesis is divided into two parts. The first part explains why Design Science Research (DSR) is the methodology chosen for this study, justifying its suitability and advantages in addressing the research questions. The second part outlines the research design for the thesis, where each element of the DSR is operationalized to investigate how threat actor groups from public sources can be represented with offensive vulnerability tools.

4.1.1 Design Science Research

Design Science Research (DSR) is a research methodology that involves the design and evaluation of artifacts for experimental purposes in information systems research [39]. This study is based on the above principle to develop a framework in an attempt to represent known threat actor groups with offensive vulnerability assessment tools. The methodology is relevant because it allows us to connect attack techniques from threat actors described in public sources to attack modules used for executing the attacks, which are then represented as Lore attack profiles (the artifacts). The modules used in this study involve a range of red team tools, such as Nmap, OpenVAS, Metasploit, THC Hydra, hashcat, BloodHound, Responder, and others.

4.1.2 Research Design

The methodology is structured into three logical and sequential parts according to the DSR. The first step is to formulate the problem and find the threat actors in order to construct the attack profiles. To accomplish this, we investigate threat intelligence frameworks and threat profiles, allowing us to diagnose the problem and build a selection of profiles. In the second step, we construct the artifact by developing Lore profiles through a mapping framework that links attack techniques with modules for attack execution. Finally, the artifact is evaluated with domain experts and instantiating threat profiles in a virtual environment. Hence, the activities are as follows:

- **Activity1: Problem formulation:** Formulate the problem and gather information on how to construct attack profiles, including investigating threat intelligence frameworks and selecting threat profiles for artifact creation.
- **Activity2: Artifact design:** Outline the process of creating artifacts using a mapping framework that links attack techniques and software modules.
- **Activity3: Artifact evaluation:** Workshops with domain experts that involve the evaluation of this mapping framework using a workshop. Furthermore, the framework is tested in a virtual environment to assess the feasibility of emulating the profiles developed.

4.2 Activity1: Problem formulation

The problem faced in this thesis is that the offensive vulnerability assessment tool Lore lacks features to represent known threat actor groups, which serves as the foundation for building a framework. The use case of Lore is in automating attacks against virtual servers in a cyber range to assess operational security. To the best of our knowledge, there is no active support or research to link attack techniques from threat actors to software modules for attack execution.
While multiple frameworks collect incident information like VERIS and STIX, we have selected MITRE as our database of choice due to its industry support and the details about adversarial techniques. There is research [40] that has expressed it as a powerful resource for gathering threat intelligence and providing information on adversary behavior, malware families, and techniques. MITRE's knowledge base offers a concrete approach to connecting attack techniques with malicious actors, as demonstrated by companies such as AlienVault and D3 Security. These companies utilize this resource to provide comprehensive threat intelligence solutions.

4.2.1 Data Collection

This section provides a comprehensive overview of the steps taken to gather, select, and analyze data from the MITRE framework. For the sake of clarity, the section is divided into two parts: Data Analysis and Data Selection and Sampling Process.

4.2.1.1 Data Analysis

The MITRE database consists of more than 135 threat actors, including techniques and software they use for targeting in ATT&CK V12. Therefore, a systematic approach was used to process threat actor groups. The sampling was initially carried out with keywords to select profiles that include: *military*, *aerospace*, *defense*, *telecom*, and *government*. The decision was motivated by the fact that these types of threat actor groups specifically target critical infrastructure, making them highly relevant to simulate and model in cyber defense exercises. However, although we implemented a more selective approach, the process appeared to be complex, as the result of the total sample size was 71 threat actor groups given these keywords. There were certain overlaps between the groups, but the most observed were found in the government with 52 threat actor groups, followed by military 22, defense 12,

telecom 10, and aerospace 4. This large sample size indicates challenges for data analysis and highlights the need to implement a more fine-grained data selection and sampling process, which will be described in more detail below.

4.2.1.2 Data Selection and Sampling Process

Because of the vast amount of data collected in the initial phase, we determined that it was necessary to establish more strict boundaries within the scope of our study. An approach we used to establish these boundaries was to focus on threat actor groups that target military and defense organizations. Defense and military terms are often used interchangeably, but there is a slight difference between them. Military typically refers to the army, special forces, and any other branch of service responsible for defending the nation against external threats.
Defense, on the other hand, is a broader term that can refer to a wide range of activities related to national security, including military defense, but also encompassing research institutions, technology companies, and other intelligence services. We believe that by deriving attack techniques from these two keywords can provide valuable guidance in developing training sessions to mitigate potential attacks from these threat actor groups. As a result of our analysis, we reduced the number of identified threat actors (military, aerospace, defense, telecom, and government) from an initial count of 71 to a more manageable number of 30 (defense, military).
On the side, several observations under the analysis were that threat actor groups Andariel and Kimsuky strongly overlap with the Lazarus Group. Similarly, Deep Panda was observed to overlap with APT19. Because they are a subset of each other, we choose to group them as a single entity under the most common name. The remaining candidates from the 135 threat actor groups are visualized in Table 4.1 and processed in our investigation framework for further evaluation.

4.2.2 Data-Set Investigation and Analysis

An investigation framework is developed to analyse threat profiles from the data set. We describe the framework into two main components: threat actor analysis and threat actor group selection for artifact creation.

4.2.2.1 Threat Actor Analysis

The investigation framework is used to track the activities of threat actors in multiple instances of security incidents. For this purpose, we define this activity in the variable *NumOfTechniques* in our investigation framework and it is analyzed in the category *Technique Used* in MITRE. This is without considering sub-techniques. The variable is important to consider because if a threat actor has few techniques, it can be challenging to represent a complete profile for artifact evaluation. Based on this, we determined a threshold of 25 techniques to exclude threat actor groups. *Origin* refers to the suspected source of the threat actor group, such as Sweden, Iran, Russia, and China, based on reported incidents in the MITRE framework. This variable is essential to identify threat actors and avoid selecting actors of the same origin for artifact creation. *Included/Excluded* is used to indicate whether threat

Table 4.1: Dataset of threat profiles using keywords Military and Defense.

Name	Origin	Target
APT1	China	Military
APT10 (menupass)	China	Defense
APT17	China	Defense
APT19	China	Defense
APT28	Russia	Military
Axiom	China	Defense
Confucius	India	Military
Gallmaker	Unknown	Defense, Military
Gamaredon Group	Russian	Military
Ke3chang	China	Military
Lotus Blossom	Unknown	Military
Machete	Spain	Military
Mofang	China	Military
Naikon	China	Military
Sandworm Team	Russia	Military
Sidewinder	India	Military
The White Company	Unknown	Military
Turla	Russia	Military
WIRTE	Unknown	Military
Leviathan	China	Defense
Threat Group-3390	China	Defense
Elderwood	China	Defense
Fox Kitten	Iranian	Defense
Muddy Water	Iran	Defense
POLONIUM	Lebanon	Defense
Magic Hound	Iran	Military
Lazarus Group	North Korea	Military
Hafnium	China	Defense
Thrip	Unknown	Defense
Transparent Tribe	Pakistan	Defense

actors were included or excluded from the analysis. This variable is significant in indicating the integrity of the selection procedure. Finally, the result of the investigation framework is shown in Table 4.2.

Table 4.2: The table provides an overview of the investigation framework. The selection of these groups is based on the metric *NumOfTechniques*, which measures the number of techniques across the threat actor groups in the dataset. Note, however, that number of *NumOfTechniques* may vary when compared to the techniques in MITRE due to the continuous addition of new observations in the platform.

Name	Origin	NumOfTechniques	Included/Excluded
Lazarus Group	North Korea	79	Included
APT28	Russia	66	Included
Sandworm Team	Russia	58	Included
Turla	Russia	48	Included
Threat Group-3390	China	46	Included
Muddy Water	Iran	44	Included
Magic Hound	Iran	44	Included
Gamaredon Group	Russian	42	Included
APT10(menupass)	China	37	Included
Ke3chang	China	36	Included
Leviathan	China	33	Included
Fox Kitten	Iran	31	Included
Sidewinder	India	23	Excluded
APT1	China	20	Excluded
APT19	China	18	Excluded
Hafnium Team	China	18	Excluded
Confucius	India	16	Excluded
Axiom	China	15	Excluded
Naikon	China	13	Excluded
Transparent Tribe	Unknown	13	Excluded
WIRTE	Unknown	10	Excluded
Machete	Spain	7	Excluded
The White Company	Unknown	7	Excluded
POLONIUM	Lebanon	7	Excluded
Gallmaker	Unknown	6	Excluded
Elderwood	China	6	Excluded
Thrip	Unknown	4	Excluded
Mofang	China	3	Excluded
APT17	China	2	Excluded
Lotus Blossom	Unknown	0	Excluded

4.2.2.2 Threat Actor Group Selection for Artifact Creation

We will now move on to the final step of the sampling and selection process, which is to choose threat actor groups for artifact creation. In this instance, we randomly selected APT28, Lazarus Group, Leviathan, and Muddy Water, as shown in Table

4.3. The *NumOfTechniques* from these threat actor groups will serve as the basis for an attempt to link them with attack modules.

Table 4.3: Table presents the final candidates of the investigation process that will be used to link NumOfTechniques and attack modules in order to create Lore attack profiles (artifacts).

Name	Origin	NumOfTechniques
Lazarus Group	North Korea	79
APT28	Russia	66
Muddy Water	Iran	44
Leviathan	China	33

4.3 Activity2: Artifact design and creation

As mentioned previously, the first step is to establish the connection between the attack techniques (NumOfTechniques) used by the threat actors and the modules (Metasploit, Responder, THC Hydra, Responder) for attack execution. The artifact or Lore attack profile is created by combining all possible attack modules, which are then executed as a complete profile against a virtual system. The design for this process will be explained through the Mapping Framework Design and the Mapping Framework Flow Structure. We provide an explanation of each parameter and its purpose, followed by an illustrative example. Note that the terms software module and attack module might be used interchangeably.

4.3.1 Mapping Framework Design

The first parameter in the framework *Technique* refers to the specific procedure or method that a threat actor uses to carry out the attack in order to achieve their objective. This information is identified and gathered in the MITRE Technique section, which is used to provide descriptions of attack techniques. For instance, a large-scale scanning process to identify vulnerable servers in a network would fall under the Reconnaissance technique.
AttackID identifies a particular attack technique and can be found in the Technique section of MITRE described with the label ID. For instance, one action under AttackID T1589 involves gathering victim identity information for further exploitation. The use of an identifier such as AttackID offers several benefits to analysis with a consistent reference system for attack techniques.
Module refers to the attack modules used within SVED to carry out instructions during an attack. For instance, the module `post/windows/gather/enum_files` from Metasploit can perform a collection attack, which involves collecting information about files on a Windows system. Another example of an attack module is `exploit/windows/local/registry_persistence`, which enables persistence across system restarts by installing the payload in the Windows registry. These modules are just a few examples among many that can be utilized within SVED to

target a virtual system. This variable is not derived from the MITRE framework and represents a new addition to our analysis.

Mapping: identifies the relationship between an attack technique and an attack module, measured on a scale of low, medium, and high. It is used to determine the correlation between the attack techniques used by the threat actors and the attack modules used to carry out those instructions within SVED. This variable is not derived from the MITRE framework and represents a new addition to our analysis.

A *low* rating is assigned when there is a weak correlation between the attack technique used by a threat actor and the code specified in the attack modules. For example, the use of a custom malware called CHOPSTICK for command-and-control operations. This malware have tailored functionality that controls data transmission between the attacker's computer and the victim's machine. However, the complex structure of this malware makes it hard to identify appropriate attack modules while maintaining context and realism for cyber threat emulation. As a result, finding a suitable attack module is challenging that replicate similar functionality. Another scenario in which a low rating is assigned is when the attack technique focuses on creating and establishing resources. In such cases, the issue is related to the virtual environment and does not concern the use of attack modules. Another scenario where a low rating is assigned is when the incident description is not clear, overly complicated, or open to multiple interpretations of the attack. As a result, this makes it difficult to establish any form of link between the activities of a threat actor and functionality in an attack module. Besides, if the technique is identified as a module but does not exist in the Lore configuration, it is also evaluated as low. Finally, if an attack module covers only a fraction of the services specified in the incident description, such as one out of three, it is evaluated as low.

A *medium* rating indicates that the techniques used by a threat actor in their operations can be partially mapped to an attack module with some level of similarity. For instance, a threat actor group uses a modified version of the zip utility procedure for data exfiltration. This attack process might not be identical in implementation, but can be assigned to the `post/multi/manage/zip` module in the Metasploit framework. This module demonstrates similarities in the original execution with the attack technique. Another scenario in which a medium rating may be assigned is when the attack description is clear but there is not enough technical information available for attack replication. For instance, if a threat actor has engaged in activities related to the collection of generic files. This attack process can be assigned to the `gather/enum_files` module, but some implementation features are missing for the emulation to be assigned as high. Another scenario in which the mapping is assessed as medium occurs when the incident description mentions multiple attack sequences involved in a technique. For instance, a threat actor performs brute-force attacks against various services such as SMB, Microsoft Autodiscover and SQL. In this scenario, an attack module might be suitable for two of these services, but not necessarily for all of them. Therefore, while there is a degree of alignment, such as two out of three, the mapping may not be fully accurate due to the diversity of the targeted services. This implies that a mapping is feasible to some extent; however, the module does not target all the services specified in the attack scenario and misses details in execution to be evaluated as high.

A *high* rating indicates a strong correlation between an attack technique and an attack module without any significant loss in context or realism. For instance, if a threat actor uses a port-scanning technique to identify open ports or system services on a target system. This attack process is identified with the software tool *Nmap* due to its high degree of similarity in execution and context to the technique used in incident description. Another scenario in which a high rating is assigned is when the attack technique is associated with a CVE identifier and can be linked to an attack module. For instance, if a threat actor has exploited CVE-2015-1701 [41] that allows local users to gain privileges through a crafted application. This CVE identifier can be mapped to the `exploit/windows/local/ms15_051_client_copy_image` module in the Metasploit framework. As a result, the attack module yields an output that is closely aligned with the attack technique described in the incident description.

4.3.2 Mapping Framework Flow Structure

To link an attack technique and an attack module, activities 1-3 are used together in a flow chart diagram shown in Figure 4.1. We provide a brief summary of the activities described in this chapter to help understand the overall structure. *Activity1*: collect all relevant information about the incident and the technique that the threat actor uses to engage in their attack. *Activity2*: used to develop search terms to locate attack modules within the SVED or Metasploit framework where some typical searches are simplified in Table 4.4. *Activity3*: an artifact analysis is carried out with domain experts to evaluate and discuss the mapping framework.

4.4 Activity3: Artifact evaluation

Artifact evaluation is conducted in two stages. In the first stage, domain experts participate in a workshop to assess the mapping framework and the correlation between attack techniques and modules. In the second stage, the artifact is instantiated on virtual machines in a cyber range. However, for the sake of clarity, the instantiation process is shown separately in Figure 4.2. These two evaluation methods will be discussed in more detail below.

4.4.1 Workshop

In each workshop, a set of fourteen attacks was randomly chosen from the four groups of threat actors being studied. The selection process emphasize threat actions classified as low, medium or high value. Additionally, four assessments were included in the low category to identify pitfalls and provide a varied view of the assessment process. We have limited our evaluation steps to the above due to the time frame of this study.

4.4.1.1 Workshop approach

The workshop was conducted in the following structure. First, the general concept of the mapping framework was explained, along with the view and the criteria used to

Table 4.4: Some examples of search terms and modules identified using MSFConsole. Note that some modules, such as THCHydra, Responder, and MSFVenom, were not found directly through MSFConsole but required additional research to locate. To exemplify the functionality of some of the search terms; LDAP was used in combination with user credentials, dump data, and samba to identify and locate ldap_hashdump. Multiple searches were occasionally used to broaden the search, indicated with the OR operator.

Search terms	Modules
LDAP + usercredentials|dump data|samba	auxiliary/gather/ldap_hashdump
Data compression + 7Zip|7z.exe	post/multi/manage/zip
win32k.sys vuln + CVE:2015-1701|privilege escalation	ms15_051_client_copy_image
File enum|Gather files + searchfrom	post/windows/gather/enum_files
ARP Packet Scanner| ARP Discovery Module	auxiliary/scanner/discovery/arp_sweep
ReverseHttps + Stager|DLL injection	payload/windows/meterpreter/reverse_https
Gather credentials + User data|User tokens	post/windows/gather/credentials/credential_collector
Keylogger + recordscreen	post/windows/gather/screen_spy
SYN-flooding	auxiliary/dos/tcp/synflood
Bruteforce + Microsoft Autodiscover|SMB|LDAP|SQL	THCHydra
NTLMv.*|NBT-NS	Responder
WinRAR obfuscation|Payload reduction	MSFVenom

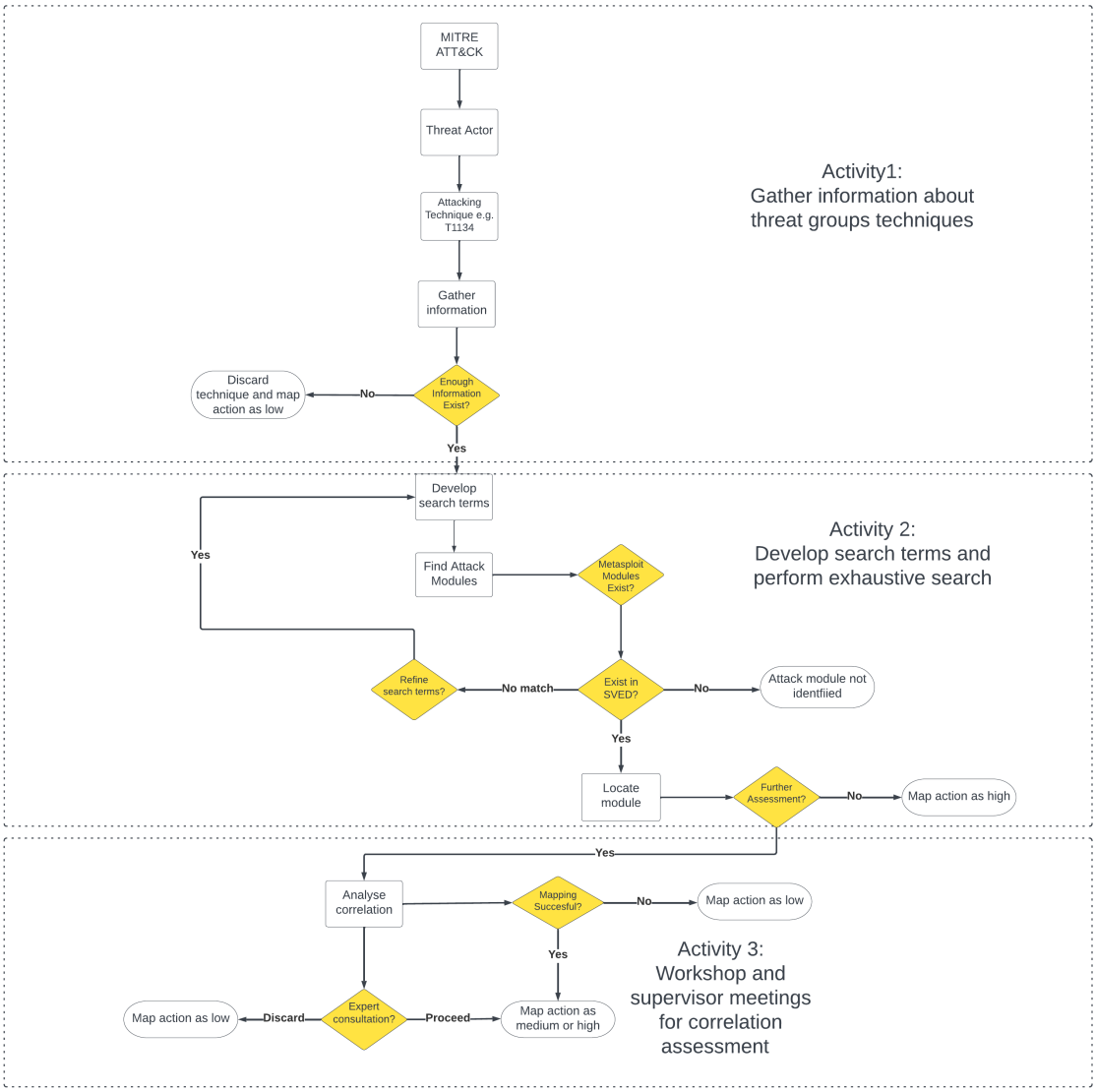

Figure 4.1: Representation of the steps involved in linking attack techniques and modules is provided using the activities. The process involves analysing attack techniques and develop search terms to find attack modules and evaluation with domain experts to assess this correlation when needed.

Table 4.5: Mapping framework that link actions from threat actor groups with software modules utilized in SVED.

Actions	Name	Technique	AttackID	Module	Mapping
Action1	APT28	Reconnaissance	T1595.002	Nmap	High
Action2	APT28	Discovery	T1120	-	Low
Action3	Leviathan	Resource Development	T1583.001	-	Low
Action4	Leviathan	Collection	T1560	post/multi/manage/zip	High
Action5	MuddyWater	Persistence	T1547	exploit/windows/local/registry_persistence	High
Action6	MuddyWater	Execution	T1203	exploit/windows/fileformat/office word_hta	High
Action7	Lazarus group	Privilege Escalation	T1055	post/windows/manage/reflective_dll_inject	High
Action8	Lazarus group	Credential Access	T1595	THC Hydra	High
Action9	Leviathan	Defense evasion	T1027	MSFVenom	High
Action 10	APT28	Discovery	T1040	Responder	High
Action 11	APT28	Defense evasion	T1027	-	Low
Action 12	Leviathan	Exfiltration	T1567	-	Low
Action 13	MuddyWater	Resource Development	T1588	-	Low
Action 14	Lazarus group	Privilege Escalation	T1134	-	Low

link attack techniques and attack modules. Second, the list of the fourteen actions was presented with a justification for why they were placed in the low, medium, or high mapping categories. Third, domain experts were given the opportunity to iterate through this list of actions and make their own assessments. Finally, general thoughts and concluding questions were discussed. For more details of the workshop structure, please refer to the Appendix.

4.4.1.2 Workshop advantages

This thesis motivates several advantages in evaluating the artifacts using a workshop, including:

- **Validation of the artifact**: The workshop serves as a platform for analyzing and validating the artifact created in this study. We believe that by engaging domain experts in the cyber security field, we can assess the quality and effectiveness of the artifact through their insights and feedback.
- **Expert input and perspective**: Engaging domain experts in activities such as workshops allows us to tap into their vast expertise and experience in the domain. Their input provides valuable insights and perspectives to improve the process of linking attack techniques and attack modules.
- **Increased credibility**: Input and evaluation from domain experts add weight to findings and conclusions to enhance the overall validity of the artifact. Expert feedback can highlight potential gaps to make necessary adjustments and improvements.

4.4.1.3 Workshop challenges

However, besides advantages, it is important to recognize the challenges that arise during this evaluation process. The workshop faces a set of potential threats that should be highlighted with the approach, including:

- **Two domain experts**: Having only two experts can be misleading, as different sets of experts may interpret the questions differently and offer alternative assessments based on the description of the incident.
- **Two workshops**: Having only two workshops may be considered insufficient for a comprehensive analysis. A diversity of workshops can provide greater insight and reveal additional problems and pitfalls associated with the research approach.
- **One set of attacks**: Having only one set of attacks can be misleading, as different sets of attacks can impact the outcome of the assessment process.

4.4.2 Instantiating threat profiles against an organization in CRATE

The result of linking attack techniques with attack modules is shown in Table 4.6. The attack modules from these threat actor groups will be executed as a complete profile to target a virtual system in CRATE. We will first describe the scenario used in this study and then explain the process of instantiating threat profiles.

Table 4.6: The table presents the mapping of attack techniques to attack modules for execution on virtual systems in CRATE. These mappings represent the Lore attack profiles developed in this study, which resulted from the previous step of linking attack techniques with their corresponding attack modules.

Name	Origin	Attack modules
APT28	Russia	45
Lazarus Group	North Korea	39
Muddy Water	Iran	29
Leviathan	China	18

4.4.2.1 Scenario

In the context of this paper, a scenario refers to a collection of technical activities executed by a red team against one or more virtual systems [6]. These activities comprise a range of adversary techniques, including execution, reconnaissance, lateral movement, exfiltration, and others.

4.4.2.2 Baseline for executing attacks

The scenario for all attacks in this study is a simulated attack where a user inserts a malicious USB drive into a computer. This initial step is used to establish communication with the virtual machines using a reverse TCP connection. In short, the connection enables us to create a meterpreter shell session on the compromised machine, granting us privileged access to Metasploit modules and other actions that are not usually available through a regular command shell.

4.4.2.3 Virtual environment for executing attacks

The experiment is being conducted in the cyber range CRATE. This allows us to expose virtual environments to our artifacts while maintaining strict control over the conditions. In this thesis, we use virtual machines that cover a wide range of services, applications, and operating systems for artifact evaluation.

4.4.2.4 Instantiation of attack profiles

Figure 4.2 shows the process of instantiating attack profiles. First, an operator selects a threat actor from the MITRE ATT&CK framework for investigation. Second, the operator analyzes whether the attack technique can be identified as an attack module in the Lore configuration. Third, if the technique can be identified as an attack module, the operator specifies the attack module in the profile by including the particular action in a whitelist. Fourth, if a technique is not identified as an attack module in Lore, it is excluded from the process, as illustrated in Figure 4.2 with the *Discard Technique* step. Finally, the last step involves executing the profile against the virtual environment to observe its overall success.

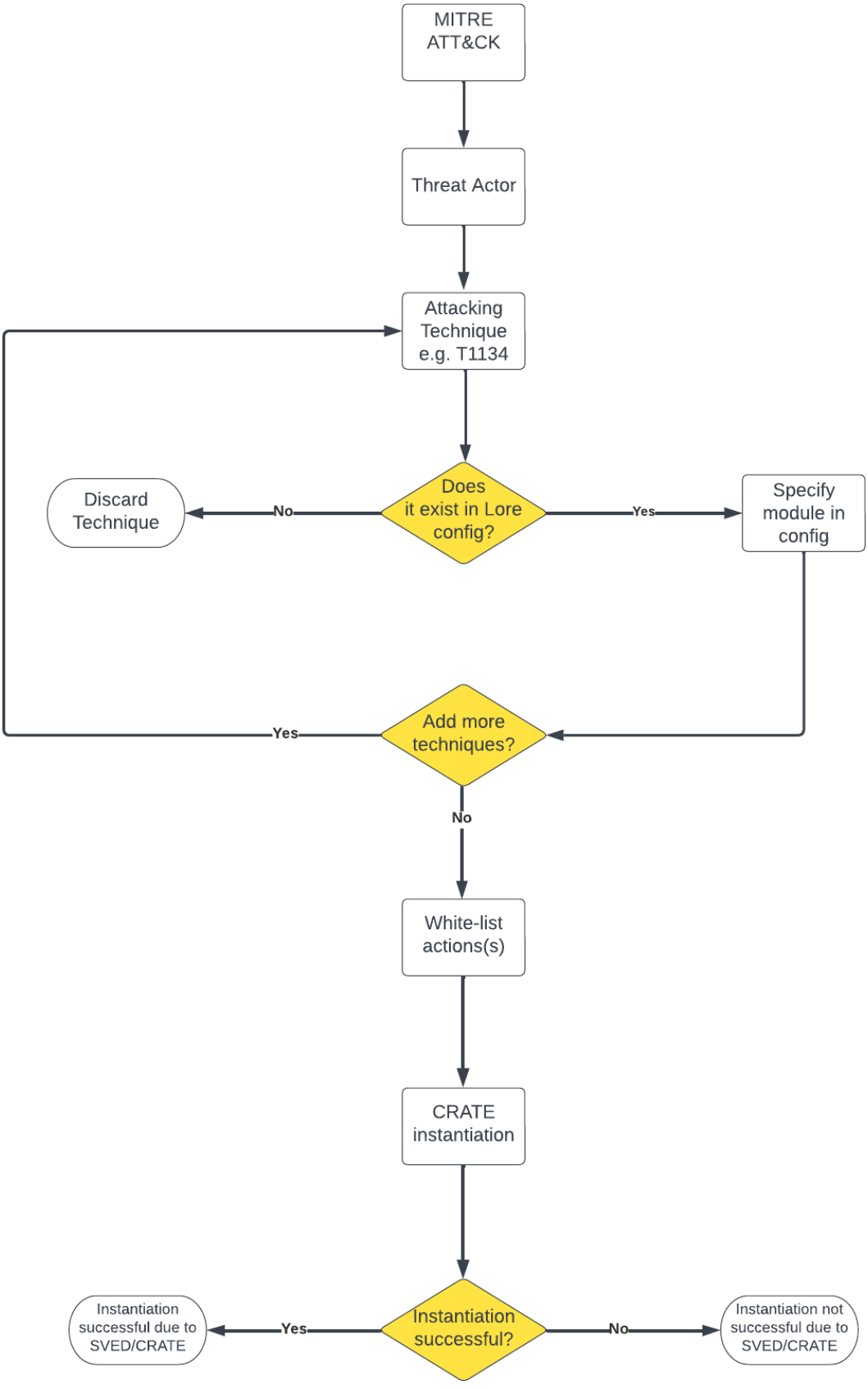

Figure 4.2: Simplified instantiation process of executing attack modules with Lore.

5
Results

In this section, we present the final results of the mapping process together with the evaluation of the workshop and the instantiation of the threat actor profiles.

5.1 Implementing a representation of known threat actor groups

The result of linking attack techniques and modules for the four types of threat actors is shown in Table 5.1. This table shows the attack techniques and modules that have been analyzed and assigned a high, medium, or low mapping. The findings indicate the presence of attack modules for 45 techniques at medium and high intervals for APT28, resulting in an accuracy of 68%. Similarly, for Muddy Water, attack modules were identified for 26 techniques at high and medium intervals, resulting in an accuracy of 59%. In the case of Leviathan, modules for 18 techniques were identified at high and medium intervals, resulting in an accuracy of 55%. Finally, for the Lazarus Group, attack modules were identified for 39 techniques in high and medium intervals, achieving an accuracy of 49%. Table 5.2 presents the variety of techniques these threat actor groups use to achieve their objectives.

Table 5.1: Table presents the results of the mapping framework to represent known threat actor groups from public sources sorted by *Accuracy*.

Name	NumOfTechniques	High	Medium	Low	Accuracy(Medium+High)
APT28	66	30	15	21	68%
Muddy Water	44	17	9	18	59%
Leviathan	33	8	10	15	55%
Lazarus Group	79	29	10	40	49%

5.2 Artifact evaluation

The evaluation of the artifact is presented through two separate steps: the workshop evaluation and the instantiation of threat profiles.

Table 5.2: The variety of attack techniques used by the four threat actors groups.

Attack techniques	Lazarus Group	APT28	Muddy Water	Leviathan
Defense Evasion	21.74%	16.46%	14.58%	18.42%
Collection	4.35%	12.66%	6.25%	2.63%
Persistence	7.61%	11.36%	6.25%	7.89%
Command & Control	10.87%	8.86%	16.67%	10.53%
Initial Access	4.35%	8.86%	4.17%	10.53%
Resource Development	5.43%	3.8%	4.17%	10.53%
Credential Access	2.17%	5.06%	6.25%	2.63%
Execution	5.43%	6.33%	12.5%	13.16%
Reconnaissance	3.26%	3.8%	2.08%	2.63%
Exfiltration	3.26%	3.8%	2.08%	5.26%
Privilege Escalation	6.52%	7.59%	6.25%	10.53%
Lateral movement	4.35%	5.06%	2.08%	5.26%
Discovery	15.22%	5.06%	16.67%	0%
Impact	5.43%	1.27%	0%	0%

5.2.1 Workshop evaluation

During the workshop, fourteen questions were discussed and eleven of them received unanimous agreement. The three questions that did not achieve consensus were thoroughly discussed, leading to new insights and perspectives to improve the mapping process. To visualize both similarities and differences in the assessments, we used a notation system called *Mapping*, with *D1* and *D2* denoting the low, medium, and high action assessments made by domain experts.

5.2.1.1 Workshop discussion

The first point of discussion (Action 2) concerns the use of the `CopyToUSBFromVM` module, which was suggested by this study as a means of receiving notifications whenever a USB mass storage device is connected to a target system. The domain experts assessed the selection of the attack module as incorrect based on the information provided in the incident description. However, both proposed using the *ListUsb* system command in meterpreter as an alternative solution.

The second topic of discussion (Action11) concerns how to assess the attack technique. One domain expert suggested that there might be more suitable modules to achieve the same goal. However, due to time constraints during the workshop, it was decided that further evaluation were needed to confirm this assessment.

The third point of discussion was the last attack in the workshop. One domain expert suggested using the `token manipulator` command and the `keylogger` functionality within the meterpreter rather than specifying no module and assigning a low rating. We looked into all these ideas and concluded that some of the functionality is not related to any attack module but is instead executed with system commands in meterpreter. As a result, it cannot be directly associated with a module in the library of actions, and thus a low assessment score is more suitable in this context.

Table 5.3: Result of the workshop discussion. D1 = Domain Expert 1, D2 = Domain Expert 2.

Actions	Threat Actor	Mapping	D1	D2	Comment
Action1	APT28	High	High	High	Executes a variation of TCP/UDP flags
Action2	APT28	Medium	Low	Low	Modules used in the wrong context. Meterpreter runs ListUSB
Action3	Leviathan	High	High	High	Can be done with one command in CRATE
Action4	Leviathan	High	High	High	Target a directory/file in the VM and zip it. Identical.
Action5	MuddyWater	High	High	High	Both local/remote exist to add currentuser/run in registry.
Action6	MuddyWater	High	High	High	CVE-indicator specifies the identical software module
Action7	Lazarus group	High	High	High	Identical to what adversaries use for reflective dll
Action8	Lazarus group	High	High	High	Hydra are correct due to the technical context.
Action9	Leviathan	High	High	High	Encoder works, but this is possible with one command in Linux.
Action10	APT28	High	High	High	Responder modules target the identical thing.
Action11	APT28	Low	Medium	Medium	Interpretation question. Can be achieved.
Action12	Leviathan	Low	Low	Low	Logic missing in SVED
Action13	MuddyWater	Low	Low	Low	Logic missing in SVED
Action14	Lazarus Group	Low	Low	Medium	Add token manipulation models, adduser and keylogger module

5.2.1.2 General feedback and closing questions

In terms of general feedback and closing questions, the participants raised several questions about the context and view in which the mapping framework is developed. We discussed that the view is for cyber defense exercises and the importance of striking a balance at the context level when analyzing attack techniques for module mapping. This discussion was important in determining how much context about the attack should be integrated into the assessment. To clarify this, we had two types of scenarios: a low-context and a high-context scenario. A low-context scenario refers to a situation where the focus is solely on the execution of a specific task, such as packing a file for data exfiltration. In this situation, the technique is taken out of the context which allows multiple methods to achieve the same result. For example, emulation can involve a variety of open methods or concealed methods for copying, transferring, or retrieving data. In contrast, a high-context scenario involves additional variables, such as the specific operating system, exfiltration technique, file type, and tool that were used to establish communication. Like using Ubuntu operating system, 7zip utility, encrypted_file.txt file, and Secure Shell (SSH).

5.2.2 Instantiating threat profiles

For the sake of clarity, we illustrate the result of the instantiation in three distinct stages: negative, positive, and neutral results.

5.2.2.1 Negative outcome

Negative outcomes occur when the profile fails to execute the intended attacks and exploit vulnerabilities effectively. These results are due to multiple factors, such as the absence of suitable vulnerabilities, session disruptions, missing module parameters, or protective software that prevent the attack process from being executed. This result is shown in Table 5.4.

5.2.2.2 Positive outcome

Positive outcomes are instances where the modules in the profiles successfully execute and exploit vulnerabilities within the scenario that result in a realistic attack. These examples illustrate how Lore was able to identify the attack module and take advantage of system vulnerabilities. This result is shown in Table 5.5.

5.2.2.3 Neutral outcome

Neutral outcomes are situations where the execution of the attack did not occur due to missing information applied to Lore or action exhaustion within the scenario. This information was observed with the state of the code injection process in the log file. However, the module `local_exploit_suggester` in the Metasploit framework was useful for checking a system for local vulnerabilities and for giving Lore instructions to further exploit the victim machine.

Table 5.4: Sample of failed attack modules executions.

Failure	Attack module
Did not receive an HTTP 200 OK response	exploit/windows/http/sharepoint_workflows_xoml
USB malware run failed	payload/windows/meterpreter/reverse_tcp
Failed to get admin logon	post/windows/escalate/getsystem
Failed to detect the application	exploit/windows/http/ssrs_navcorrector_viewstat
Session read failed	DisableAV
Cannot create a file when that file already exists	exploit/multi/handler
Failed to send SSRF request	exploit/linux/http/vmware_vrops_mgr_ssrf_rce
Target did not respond to check	exploit/linux/http/f5_bigip_tmui_rce
Failed to test OGNL injection	exploit/multi/http/atlassian_confluence_webwork_ognl_injection
Failed to open LenovoDiagnosticsDriver device	post/multi/recon/local_exploit_suggester
Failed to get system via administrator	post/windows/escalate/getsystem
Job completed without session	exploit/windows/http/exchange_ecp_viewstate
Error for MetasploitServerExploit	exploit/windows/http/exchange_chainedserializationbinder_denylist_typo_rce
The service is running, but could not be validated	post/multi/recon/local_exploit_suggester
Module missing required parameter	exploit/solaris/ssh/pam_username_bof
Failed to generate token	exploit/linux/http/f5_icontrol_rest_ssrf_rce
Failed authentication	exploit/unix/ssh/array_vxag_vapv_privkey_privesc

Table 5.5: Sample of successful attack module executions.

Success	Attack module
All shellcode commands run	BloodHound
All shellcode commands run	MetasploitShellcode
Add routes succesfully	post/multi/manage/autoroute
Post module completed	post/windows/escalate/getsystem
Post module completed	post/multi/recon/local_exploit_suggester
All shellcode commands run	DisableAV
scanning users at 192.168.1.0/24	auxiliary/scanner/ssh/ssh_enumusers
Reset Proxy	Cleanup completed
ping sweep for IP range 192.168.1.0/24	post/multi/gather/ping_sweep
All basic commands run	BasicCommands
All shell scripts run	RawShellCommand
Evasion module completed	evasion/windows/process_herpaderping
USB malware run using the command F:/admin_hotfix.exe	MountUSBandRunFile
Bypass antivirus successful	Veil completed
nmap -v -sn 192.168.1.0/24	Nmap
Started reverse TCP handler on 192.168.1.1:1013	payload/linux/x86/meterpreter/reverse_tcp
DefaultUser=admin, password = 123	post/windows/gather/credentials/windows_autologin

6
Discussion

6.1 Result

In this section, the discussion of the result and method is presented.

6.1.1 Represent threat actor profiles with offensive vulnerability tools

The development of a framework that map attack techniques with modules has been shown to be an initial step in organizing data to represent known threat actors from public sources. The findings indicate that attack scripting tools, such as SVED, have the capability to execute attack techniques used by threat actors in real-world cyber attacks, although more development is indicated. While comparing the different groups of threat actors, APT28 stands out with higher accuracy, mainly due to the increased prevalence of CVE identifiers related to their attack techniques.
To explore the possibility of modeling threat actor profiles and representing them with offensive vulnerability tools, we will divide the discussion into two parts: challenges and effective parameters for representing threat actor profiles.

6.1.1.1 Challenges for representing threat actors

This study indicates that the process of representing threat profiles can be limited based on the requirements involved in replicating an attack technique. In particular, resource development is an example where techniques often exceed the capabilities of attack modules and instead involving adversaries' strategic planning, scheduling, and resource allocation to enhance targeting efforts. For example; actions related to developing and staging capabilities, establishing accounts, and infrastructure.
We also observe challenges in identifying attack modules that concern defense evasion and persistence techniques. The challenges arise from the actors using multiple different code techniques to obfuscate their command and control operations. The techniques used were a subset of http(s) methods, but they involved special invocations, such as using a shorter timeframe for communication, steganography, or the use of junk data within protocol traffic. This type of custom-crafted code is intricate and does not align easily with existing attack modules.
Another challenge was the possibility that a single technique was associated with multiple attack modules. This complexity made it occasionally challenging to determine the most appropriate module that aligned best with the incident description. For example, denial-of-service attacks that focus on disrupting the normal function

of a service can be performed in multiple ways with Nmap or modules in Metasploit (`auxiliary/dos`). It was also observed that some incident descriptions were complex to understand, and extracting techniques in the correct context was challenging. To be able to do so effectively, malware analysis skills is essential. The knowledge is needed to distinguish actions, focus on the relevant technique for the intended tactic, and avoid inflicting other operations that are not needed for modeling purposes. This implies that the accuracy of this mapping process depends, to some extent, on the practitioner's knowledge.
Malware and exploit kits with complex operational structures that feature multiple callers and stages of execution present a challenge. These stages, while interconnected, might correspond to a subset of modules available within the library of actions that can complicate the mapping process. Our perspective is that if a technique is greatly disconnected from its execution context, the value of cyber emulation is likely to decrease and may not generate the intended output. This, however, is a complicated problem and depends on setting the balance between realism and the context of execution while providing value in a cyber defense exercise. We also believe that development hours required to create modules in order to replicate attacks is something to take into consideration. This concerns understanding the incident description, reverse engineering the code, and transforming it into a module in the library of actions. Some actions classified in low and medium intervals in this study would require significant changes in existing attack modules and the virtual environment to successfully replicate an attack. On the side, some techniques used by the threat actors could be mapped to red team tools like LaZange, Kodiac, and Mimikatz, but not identified as a module in Lore, which indicates that more tools are available to be integrated in the library of actions.
Finally, as a last remark, the hierarchical structures within the MITRE platform are sometimes inconsistent. Techniques often overlap because they are applied in multiple tactics and phases of an attack. Hence, assigning techniques exclusively to a specific attack module based on the incident description is shown to be challenging. This complexity in classification makes it difficult to define clear boundaries between techniques in order to map actions to a module.

6.1.1.2 Successes in representing threat actors

Multiple effective parameters can be used to represent known threat actor groups. The prevalence of CVE entries in incident descriptions has been shown to be a key factor in linking attack techniques and modules. When such entries exist, no further assessment is required. This made, for example, APT28 more easier to represent because of the large number of CVE-identifiers linked with their attacks. We believe the reason is that APT28 often leverages known vulnerabilities for initial access. They do this by exploiting common vulnerabilities for which security patches or updates may not have been applied by the targeted organization.
Incident reports that are descriptive and tailored to a specific attack process improve the representation of threat actor groups. This information is used to develop search terms and locate attack modules. To what extent this is achieved is based on the MITRE analysts who gather threat intelligence from the authors that investigate security incidents and write technical reports. Another parameter to consider are the

documentation or comments within the attack module that the developer provides. This information plays an important role in understanding the purpose of the attack module and its intended execution. More importantly, comments provided by the developer are used to identify functionality by using search terms derived from the incident reports. Techniques related to initial access, collection, and reconnaissance were comparatively easier to identify and formulate search terms. The reason why they are simple to recognize and assess is believed to be a result of their fundamental role in the attack cycle and the clear information specified in the incident description. For example, phishing for initial access to harvest user credentials and collect data from network shared drives. Additionally, the techniques has been widely used together to collect victim information such as IP addresses and domain names. Techniques involving persistence; for example, targeting registry run keys or the startup folder were frequently observed among the threat actor groups. This is used to maintain access over a longer period of time through a specific command or script that is run every time the system is rebooted. In addition, DLL injection was often used to execute code within the address space of a system process. We also identified multiple persistence techniques among the threat actor groups that involve uploading an executable to a remote machine and configuring it as a persistence service. The Metasploit framework has proven to feature suitable modules for these above actions, simplifying the process of representing this type of technique.

6.1.1.3 Instantiating known threat actor profiles in CRATE

The success or failure of the instantiation of threat profiles against a virtual organization in CRATE is observed to be influenced by a variety of factors.
First, the exploitability of the target system plays a significant role. If the target system does not possess the required vulnerability version or service, it will not be compatible with the executed attack module.
Second, the establishment of sessions can be disrupted during the initiation phase. These issues can be caused by unexpected interruptions or disconnections between the attacker and the target system.
Third, in certain instances, attacks were marked as completed, but a session could not be successfully established. This observation indicates that the attack process had halted because all the actions specified in the scenario had been exhausted.
Fourth, some of the attack modules within the profiles were found to not be fully activated. This implies that Lore is not provided with enough instructions on what types of actions to execute and in what order based on the targeted system.
Finally, rejection of the payload by the victim machine can also lead to failure. If the virtual machine unexpectedly responds to the payload that contains malicious code or instructions, the module will fail, and an error message will be generated.

6.2 Method

Investigating threat actor groups with offensive vulnerability tools is still a relatively scarce field and the existing research on this topic is limited. This indicates that interpreting the findings related to representing threat actors using offensive vulner-

ability tools can be challenging. To ensure the feasibility and validity of the project, several important decisions were made throughout the research process that could influence the final results. This section aims to highlight some of these decisions and their potential impact on overall results, while Chapter 9 will provide a more comprehensive analysis of validity threats.

6.2.1 Dataset

As discussed in Chapter 6, the dataset was composed of 135 threat actor groups. Therefore, we had to adopt a systematic approach to select and sample threat actor groups. This was done to ensure a detailed and focused analysis, considering the time constraints of this thesis. Although other databases could be used to generate a dataset from security incidents, they provide a different level of detail in describing techniques from known threat actor groups than the one used in this study.

6.2.1.1 Selection and Sampling Threat Actors

The selection and sampling process was used to determine which groups of threat actors should be included or excluded when developing the artifacts. This facilitated a more focused analysis that aligned with the context of our study's objectives. However, an alternative approach might have entailed systematically selecting threat actor groups for campaigns tailored to specific objectives, such as antivirus evasion.

6.2.1.2 Investigation framework

As discussed in Chapter 6, the data-set investigation framework was used to observe the activity of threat actors in multiple instances of security incidents. We gathered the information and stored it in *NumberOfTechniques*, which was then used to demonstrate the number of attack techniques used by the threat actor groups. Gaining insight into the different types of threat actors enabled us to implement our exclusion criteria more effectively. Based on this investigation process, we ensured that the threat actor groups had enough techniques to be represented.

6.2.2 Mapping framework

As discussed in Chapter 6, the mapping framework was implemented to represent the attack techniques described in public sources with attack modules.

6.2.2.1 Map techniques and modules

We believe that this framework has the capability to organize data from threat actor groups and can be used to generate realistic simulations of cyber threats in a systematic manner. By utilizing this framework, operators can identify potential information gaps and develop training sessions to increase the excellence of security practitioners based on real-world observations.

6.2.2.2 Future improvements

To further enrich the framework, we suggest the addition of a *CRATE* parameter to improve its effectiveness. This parameter would determine the software components that must be present in the cyber range before instantiating attack profiles. Finally, we suggest a parameter *Custom solution* within the framework. The parameter can be used to specify the commands or instructions that threat actors use during their attack, so that it can be further developed into a module in the library of actions.

7

Threats to Validity

In this section we will examine the validity threats encountered in this study and we group them into three parts: Construct, External, and Conclusion Validity.

7.1 Threats to Construct Validity

Construct validity is an essential aspect in research that concerns the degree to which a measurement accurately captures the intended concept or construct. In the context of this study, multiple frameworks were used to select and construct threat profiles and finally develop the mapping framework. We believe that it is important to identify that each of these frameworks has its own unique strengths and weaknesses in relation to the measurements.

Because the mapping framework was specifically developed for this study, there is a potential risk of imprecision and the lack of following a standardized definition. To enhance the validity of the study, two steps were taken. First, the study followed the principles of design science research, which provided a structured and repeatable approach for developing the Lore attack profiles. Second, efforts were made to validate the framework through various means, such as expert consultations with workshops, where feedback and insights from domain experts were gathered, and finally, the instantiation of threat actor profiles against a virtual organization in CRATE.

By using these above measures, the study aimed to strengthen the construct validity of the framework, ensuring that it accurately represented the intended constructs in the context of connecting attack techniques and software modules.

7.2 Threats to External Validity

External validity in a study refers to the extent to which the findings and conclusions can be generalized and applied to other contexts or populations beyond the specific scope of the research. In the context of this study, the reliance on the offensive vulnerability assessment tool Lore introduces a potential threat to its external validity. If researchers or practitioners cannot access Lore or a similar tool with comparable capabilities, it becomes difficult to reproduce the study and achieve similar results. Thus, the availability and accessibility of the tool become important when considering the external validity of this research.

7.3 Conclusion Validity

Conclusion validity refers to the degree to which the conclusions drawn from a study are accurate and supported by the data collected. The measurement assesses whether the observed relationships or effects are valid with the investigated variables. In the context of this paper, threats to the validity of the conclusion are primarily the assessment process to connect the attack techniques described in public sources and the software modules for attack execution. The inability to ask domain experts in the cyber security field and the choice of context when assessing threat profiles might decrease the possibility of achieving similar results or conclusions.

8

Conclusion and Future Work

In this chapter, we present a conclusion that answers to what extent attack techniques can be translated into modules for attack execution. We also answer the question of the extent to which it was possible to emulate attack techniques from known threat actor groups. Finally, future work is addressed.

8.1 Mapping attack techniques and modules

The alarming rise in cyber attacks is a matter of great concern that needs to be addressed and dealt with proactively. As threat actor groups continuously refine their strategies, tactics, and procedures, it becomes essential for organizations to stay informed and ready by training with realistic cyber threats.
In this paper, we investigated how attack techniques used by known threat actor groups can be represented with attack modules. To challenge this problem, we have developed a mapping framework that serves as a bridge between these two components. Our findings indicate that offensive vulnerability assessment tools such as Lore have the capability to simulate techniques used by known threat actor groups such as Lazarus Group, APT28, Muddy Water, and Leviathan. The findings were supported by identifying key parameters and challenges in representing techniques by known threat actor groups with attack modules.

8.2 Instantation of known threat actor profiles

The instantiation of the profiles indicates both challenges and success in emulating attack techniques of known threat actors described in public sources. However, the extent to which it is possible to emulate these techniques is shown to be influenced by several factors. The factors include receiving invalid responses from the victim machine, payload incompatibility, and lack of instructions applied to Lore for vulnerability exploitation. Clearly, further research is indicated to explore how the modules in the attack profiles can be used more effectively.
We also believe that addressing the widespread use of defense evasion techniques observed among the threat actor groups in this study is essential for future work. This observation highlights the need for further exploration and research on antivirus evasion methods. The identification of this trend highlights further investigation into threat actor groups that focus on tools and techniques with the purpose of obfuscating malicious code.

A

Appendix 1

Modeling threat actor groups from public sources

This workshop is part of a Master's thesis conducted at Chalmers University under the following title: Cyber Threat Emulation: Exploring the capability of implementing a framework to represent known threat actor groups from public sources with offensive vulnerability assessment tools.

Olof Magnusson would like your feedback to (1) improve his threat actor modeling process, (2) evaluate the mapping between software modules and threat actions, and (3) identify any gaps or deficiencies. Anonymity is ensured by sharing feedback only in an aggregated form that cannot be traced back to the source of the statement. However, responses to free-text questions marked with ▸ will be reported literally. If you prefer not to write it directly in the form, we would still be very much interested in your feedback and would be glad to receive an email from you at olofmagn@chalmers.se or teodor.sommestad@foi.se.

We are conducting research on how and to what extent threat actors described in public sources can be represented with offensive vulnerability assessments tools. The purpose of this research is to understand how comparable attack techniques in the wild are to software modules for attack execution. We would like to have your opinion on this modeling process and therefore devised a questionnaire specifically for this purpose.

The questionnaire includes ten types of attacks, each followed by a brief description of scenarios that have been assigned to an attack module that can be executed by SVED. Attacks are chosen randomly from four distinct threat actor groups, with each attack having an equal chance of being chosen. The range of values in the assessment to indicate agreement or disagreement is from 1 (strongly disagree) to 5 (agree completely). You can find the link to the threat intelligence platform below and a longer description of the attack scenario if necessary by inspecting the specified links. This workshop has been scheduled to have a duration of one hour.

https://attack.mitre.org/

The following criteria are used to analyze the correlation of attack techniques and modules.

- High: indicates a strong relationship between an attack technique and a software module without any significant loss of realism and execution. Mapping evaluates to High.
- Medium: indicates that the technique can be compared to a software module with some level of similarity in realism and execution. Mapping evaluates to Medium.
- Low: indicates a weak relationship between the attack technique and the software module with a significant loss in realism and execution. Mapping evaluates to Low.

10 actions randomly chosen from a pool of four different threat actors.

Action1
Description: APT28 has performed large-scale scans in an attempt to find vulnerable servers.
Sources: Pawn storm
Attack technique: Reconnaissance
AttackID: T1595.002
Suggested Module: Nmap
Mapping: High (scanning a variety of common ports) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action2
Description: APT28 has used a module to receive a notification every time a USB mass storage device is inserted into a victim.
Sources: Microsoft
Attack technique: Discovery
AttackID: T1120
Suggested Module: CopyToUSBfromVM
Mapping: Medium (Logging function is similar) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action3
Description: Leviathan has established domains that impersonate legitimate entities to use for targeting efforts.
Sources: Cisa
Attack technique: Resource Development
AttackID: T1583.001
Custom solution: Set up DNS-services in CRATE
Mapping: Low (CRATE issue rather than SVED) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action4
Description: Leviathan has archived victim's data prior to exfiltration.
Sources: Cisa
Attack technique: Collection
AttackID: T1560
Suggested module: post/multi/manage/zip
Mapping: High (Zip a file or directory on a target VM) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action5
Description: MuddyWater has added Registry Run key KCU to establish persistence
Sources: Mandiant SecureList Earth Vetala Talosintelligenc
Attack technique: Persistence
AttackID: T1547
Suggested module: exploit/windows/local/registry_persistence
Mapping: High (Point to the same root key in the registry) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action6
Description: MuddyWater has exploited the Office vulnerability CVE-2017-0199 for execution.
Sources: Clearskysec
Attack technique: Execution
AttackID: T1203
Suggested module: exploit/windows/fileformat/office_word_hta
Mapping: High (CVE matches to a specific module) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action7
Description Lazarus group malware sample performs reflective DLL injection.
Sources: McAfee Welivesecurity
Attack technique: Defense Evasion, Privilege Escalation
AttackID: T1055
Suggested module: post/windows/manage/reflective_dll_inject
Mapping: High (does initially do the same thing) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action8
Description: Lazarus Group has performed brute force attacks against administrator accounts.
Sources: Welivesecurity
Attack technique: Credential Access
AttackID: T1110
Suggested module: Hydra
Mapping: High (Cracking administrators accounts) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action9
Description Leviathan has obfuscated code using base64 and gzip compression
Sources: Proofpoint
Attack technique: Defense Evasion
AttackID: T1027
Suggested module: msfvenom Veil
Mapping: High (Specify encoder and linux package) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action10
Description APT28 has deployed the open source tool Responder to conduct NetBIOS Name Service poisoning **Sources**: Fireeye
Attack technique: Credential Access, Discovery
AttackID: T1040
Suggested module: Responder
Mapping: High .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

4 actions selected randomly based on low mapping assessment.

Action11
Description APT28 encrypted a .dll payload using RTL and a custom encryption algorithm.
Sources: Paltoaltonetworks
Attack technique: Defense Evasion
AttackID: T1027
Suggested module: None
Mapping: Low (Code and platform specific) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action12
Description Leviathan has used an uploader known as LUNCHMONEY that can exfiltrate files to Dropbox.
Sources: Proofpoint
Attack technique: Exfiltration
AttackID: T1567
Suggested module: None
Mapping: Low (Missing code and services) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action13
Description MuddyWater has made use of legitimate tools ConnectWise and Remote Utilities to gain access to target environment.
Sources: Anomali
Attack technique: Resource Development
AttackID: T1588
Suggested module: None
Mapping: Low (Tools and utilities are missing) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

Action14
Description Lazarus Group keylogger KiloAlfa obtains user tokens from interactive sessions to execute itself with API call CreateProcessAsUserA under that user's context
Sources: Web.achieve
Attack technique: Defense Evasion, Privilege Escalation
AttackID: T1134
Suggested module: None
Mapping: Low (Link broken in the technical report) .....

agree completely →
← strongly disagree
☐ ☐ ☐ ☐ ☐

▸General thoughts:

Closing questions

Please share ideas how you would approach things differently, e.g., describe your analysis procedure, discuss any challenges with this modeling process, and feel free to provide any additional insight or considerations that you consider important.

Bibliography

[1] J. C. Wong, "Uber concealed massive hack that exposed data of 57m users and drivers," *The Guardian*, vol. 22, 2017.

[2] S. Mohurle and M. Patil, "A brief study of wannacry threat: Ransomware attack 2017," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, pp. 1938–1940, 2017.

[3] E. Mikalauskas, "Rockyou2021: largest password compilation of all time leaked online with 8.4 billion entries," 2021.

[4] H. Holm, K. Shahzad, M. Buschle, and M. Ekstedt, "p^2 cysemol: Predictive, probabilistic cyber security modeling language," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 6, pp. 626–639, 2014.

[5] X. Ou, S. Govindavajhala, A. W. Appel, *et al.*, "Mulval: A logic-based network security analyzer.," in *USENIX security symposium*, vol. 8, pp. 113–128, Baltimore, MD, 2005.

[6] H. Holm, "Lore a red team emulation tool," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[7] P. Zilberman, R. Puzis, S. Bruskin, S. Shwarz, and Y. Elovici, "Sok: A survey of open-source threat emulators," *arXiv preprint arXiv:2003.01518*, 2020.

[8] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre att&ck: Design and philosophy," in *Technical report*, The MITRE Corporation, 2018.

[9] S. Y. Enoch, Z. Huang, C. Y. Moon, D. Lee, M. K. Ahn, and D. S. Kim, "Harmer: Cyber-attacks automation and evaluation," *IEEE Access*, vol. 8, pp. 129397–129414, 2020.

[10] H. Holm and T. Sommestad, "Sved: Scanning, vulnerabilities, exploits and detection," in *MILCOM 2016-2016 IEEE Military Communications Conference*, pp. 976–981, IEEE, 2016.

[11] T. Gustafsson and J. Almroth, "Cyber range automation overview with a case study of crate," in *Secure IT Systems: 25th Nordic Conference, NordSec 2020, Virtual Event, November 23–24, 2020, Proceedings*, pp. 192–209, Springer, 2021.

[12] M. Zenko, *Red Team: How to succeed by thinking like the enemy*. Basic Books, 2015.

[13] "Locked shields." `https://ccdcoe.org/exercises/locked-shields/`. Accessed: 2023-02-19.

[14] B. Ferguson, A. Tall, and D. Olsen, "National cyber range overview," in *2014 IEEE Military Communications Conference*, pp. 123–128, IEEE, 2014.

[15] H. Kavak, J. J. Padilla, D. Vernon-Bido, R. Gore, and S. Diallo, "A characterization of cybersecurity simulation scenarios.," in *SpringSim (CNS)*, p. 3, 2016.

[16] J. Vykopal, R. Ošlejšek, P. Čeleda, M. Vizvary, and D. Tovarňák, "Kypo cyber range: Design and use cases," 2017.

[17] "Realistic global cyber environment." `https://jyvsectec.fi/wp-content/uploads/2018/10/JYVSECTEC-cyber-range.pdf`. Accessed: 2023-02-20.

[18] "Crate – sveriges nationella cyberanläggning för totalförsvaret." `https://www.foi.se/forskning/informationssakerhet/crate---sveriges-nationella-cyberanlaggning-for-totalforsvaret.html`. Accessed: 2023-01-20.

[19] T. Sommestad, "Informationselement i incidentbeskrivningar: Framtagning och utvärdering under övning, ipilot," in *Technical report*, pp. FOI–R–4501–SE, 2017.

[20] M. M. Yamin, B. Katt, and V. Gkioulos, "Cyber ranges and security testbeds: Scenarios, functions, tools and architecture," *Computers & Security*, vol. 88, p. 101636, 2020.

[21] R. Alford, D. Lawrence, and M. Kouremetis, "Caldera: A red-blue cyber operations automation platform," 2022.

[22] "Simulate, validate, and mitigate with the infection monkey." `https://www.akamai.com/infectionmonkey`. Accessed: 2023-01-22.

[23] "Infection monkey." `https://github.com/guardicore/monkey`. Accessed: 2023-02-10.

[24] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *Mitre Corporation*, vol. 11, pp. 1–22, 2012.

[25] A. V. Uzunov and E. B. Fernandez, "An extensible pattern-based library and taxonomy of security threats for distributed systems," *Computer Standards & Interfaces*, vol. 36, no. 4, pp. 734–747, 2014.

[26] W. Xiong and R. Lagerström, "Threat modeling–a systematic literature review," *Computers & security*, vol. 84, pp. 53–69, 2019.

[27] S. Al-Fedaghi and S. Moein, "Modeling attacks," *International journal of safety and security engineering*, vol. 4, no. 2, pp. 97–115, 2014.

[28] M. Frydman, G. Ruiz, E. Heymann, E. César, and B. P. Miller, "Automating risk analysis of software design models," *The Scientific World Journal*, vol. 2014, 2014.

[29] S. MUSMAN and A. J. Turner, "A game oriented approach to minimizing cybersecurity risk," *Safety and Security Studies*, p. 27, 2018.

[30] D. S. Lavrova and A. I. Pechenkin, "Adaptive reflexivity threat protection," *Automatic control and computer sciences*, vol. 49, no. 8, pp. 727–734, 2015.

[31] J. Meszaros and A. Buchalcevova, "Introducing ossf: A framework for online service cybersecurity risk management," *computers & security*, vol. 65, pp. 300–313, 2017.

[32] L. Wang, S. Jajodia, A. Singhal, P. Cheng, and S. Noel, "k-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities,"

IEEE Transactions on Dependable and Secure Computing, vol. 11, no. 1, pp. 30–44, 2013.

[33] T. Sommestad, M. Ekstedt, and H. Holm, "The cyber security modeling language: A tool for assessing the vulnerability of enterprise system architectures," *IEEE Systems Journal*, vol. 7, no. 3, pp. 363–373, 2012.

[34] M. Chu, K. Ingols, R. Lippmann, S. Webster, and S. Boyer, "Visualizing attack graphs, reachability, and trust relationships with navigator," in *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, pp. 22–33, 2010.

[35] N. Ghosh, I. Chokshi, M. Sarkar, S. K. Ghosh, A. K. Kaushik, and S. K. Das, "Netsecuritas: An integrated attack graph-based security assessment tool for enterprise networks," in *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, pp. 1–10, 2015.

[36] P. Johnson, A. Vernotte, M. Ekstedt, and R. Lagerström, "pwnpr3d: an attack-graph-driven probabilistic threat-modeling approach," in *2016 11th International Conference on Availability, Reliability and Security (ARES)*, pp. 278–283, IEEE, 2016.

[37] A. Applebaum, D. Miller, B. Strom, H. Foster, and C. Thomas, "Analysis of automated adversary emulation techniques," in *Proceedings of the Summer Simulation Multi-Conference*, pp. 1–12, 2017.

[38] W. Xiong, E. Legrand, O. Åberg, and R. Lagerström, "Cyber security threat modeling based on the mitre enterprise att&ck matrix," *Software and Systems Modeling*, vol. 21, no. 1, pp. 157–177, 2022.

[39] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of management information systems*, vol. 24, no. 3, pp. 45–77, 2007.

[40] A. Villalón-Huerta, H. Marco-Gisbert, and I. Ripoll-Ripoll, "A taxonomy for threat actors' persistence techniques," *Computers & Security*, vol. 121, p. 102855, 2022.

[41] "Cve-2015-1701." `https://nvd.nist.gov/vuln/detail/CVE-2015-1701`. Accessed: 2023-10-27.