Malware Detection in Security Operation Centres

Bushra Abdulrahman Alahmadi

Kellogg College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2019

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Prof. Ivan Martinovic for the continuous support of my DPhil study and related research, for the heart-full discussions, motivation, and immense knowledge. I could not have imagined having a better supervisor and mentor for my DPhil study.

Thank you to Professor Andrew Martin and Professor Christina Pöpper for examining this thesis. Thank you to Professor Kasper Rasmussen and Professor Cas Cremers for their valuable comments during my Transfer and Confirmation of Status interviews. I am also very grateful for the financial support provided for my studies by Kind Saud University.

During this research, I had the privilege of meeting several security practitioners who worked in security operation centres. Your participation had helped me learn a lot about this field and have an appreciation of its complexity. Thank you to my publications co-authors and for all who provided feedback or guidance during my doctoral studies.

Thank you to my husband and love of my life, Abdulsalam. I couldn't have done my doctorate without your love and support. You encouraged me to pursue my dreams, celebrating my successes and being there to comfort me in my failures. I will be forever grateful.

During the first year of my doctorate, we welcomed to the world a beautiful boy Abdulaziz. From that day Aziz, you stole my heart and have made my doctoral journey full of joy and adventure. Your smiles were all I needed to keep myself motivated, and I thank god for blessing me with you.

To mom and dad, thank you. Your love and encouragement since I was little have helped me reach this goal in my life. You are my role models in life and will be forever grateful. Thank you to my siblings Wael, Abdulaziz, Rakan, Saja, Rahaf, Ahmed, Mohammad, for your support and love and babysitting during the thesis writeup. Thank you, Khawla and aunt Aisha, for keeping me in your prayers and your love and support.

During my time at Oxford, I developed beautiful friendships with amazing people. Liz, Emma, Mariam, Hawra, Klaudia, Alina and the OxWoCS team—Thank you for everything.

Finally, thank you to the CDT in Cybersecurity for giving me this opportunity and teaching me to appreciate multidisciplinary research in cybersecurity. Special thanks to David and Maureen for always being there to listen and provide guidance with a smile. Thank you to all CDT students —especially CDT-13. I will cherish the time we spent during the first year and the valuable discussion.

Abstract

Malware has evolved from viruses attacking single victims to more sophisticated malware with disruptive purposes. For example, WannaCry ransomware attacks led to hundreds of disruption to NHS care in 2017. Although organizations might have invested in security technologies, their susceptibility to WannaCry hints that the problem goes beyond technology. Security Operations Centres (SOCs) are the first-line of defence in an organisation, providing 24/7 monitoring, detection, and response to security attacks. This thesis aims to explore the challenges in malware detection in Security Operation Centres (SOCs) providing recommendations for possible technological solutions.

We first start by investigating the workflow SOC practitioners follow. Through semi-structured interviews, we recognise the analysts' role in the SOC and their interactions with the technological solutions for malware monitoring, detection, investigation and response. Our results highlight the overwhelming reliance on analysts throughout the SOC operations, which might benefit from automation. We elicit the analysts analytical thinking when making decisions, identifying the influential factors that might impact their decision making.

Moreover, we investigate security practitioners' perspectives of the security monitoring tools deployed in SOCs and their perception of the high false-positive rates. By identifying the weaknesses and strengths in current SOC tools and challenges in deploying network-monitoring tools, we derive recommendations for future SOC tools development.

Understanding the type of malware is an essential step in determining the best response. Sometimes getting access to the infected host is not possible and analysts refer to the network traffic for analysis. Hence, we propose a system that classifies network flow sequences to a malware family. The proposed system is privacy-preserving and effective in classifying a binary to a malware family based on its network traffic, not requiring access to the malware binary itself.

Behavioural malware detection approaches are found to be the most reliable by analysts. We propose a behaviour-based malware detection system that improves over state-of-the-art by detecting new or unseen malware. The system uses behavioural high-level network features preserving the privacy of the monitored hosts. Using this system, malware's network activities are captured and modelled as a Markov Chain. Due to the modeling of general bot network behavior by the Markov Chains, the system can detect new malware that has not been seen before making it robust against malware evolution.

The novelty of this research is to provide a systematic study on SOCs processes, people, and technology; providing researchers with an understanding of the challenges and opportunities within; bridging that knowledge gap and thereby setting a better foundation for future research in the field.

Statement of Originality

This thesis is written in accordance with the regulations for the degree of Doctor of Philosophy. The thesis has been composed by myself and has not been submitted in any previous application for any degree. The work presented in this thesis has been undertaken by myself, except where otherwise stated in Chapter 1.5.

Parts of this thesis have been published previously as follows. The malware family classification and detection systems in Chapter 8 was published as a conference paper [1]. The malware detection system in Chapter 9 has been accepted at AsiaCCS 2020. The results of the quantitative and qualitative studies reported in Chapters 5, 6, and 7 are currently under review.

Contents

List of Figures

List of Tables

List of Abbreviations

SOC . . . . . Security Operation Centres.
NHS . . . . . National Health Service.
IT . . . . . Information Technology.
CVE . . . . . Common Vulnerabilities and Exposures (CVE) catalog.
AV . . . . . Anti-Virus.
MSSP . . . . . Managed Security Service Provider.
SIEM . . . . . Security Information and Event Management.
SDEE . . . . . Security Device Event Exchange.
ML . . . . . Machine Learning.
AI . . . . . Artificial Intelligence.
IDS . . . . . Intrusion Detection System.
NIDS . . . . . Network Intrusion Detection System.
DARPA . . . . Defense Advanced Research Projects Agency.
KDD . . . . . Knowledge discovery in databases.
SVM . . . . . Support Vector Machine.
C&C . . . . . Command and Control.
IRC . . . . . Internet Relay Chat.
HTTP . . . . . HyperText Transfer Protocol.
DNS . . . . . Domain Name System.
ICA . . . . . Independent Component Analysis.
CIDS . . . . . Collaborative IDS.
APT . . . . . Advanced Persistent Threats.
OSINT . . . . Open-source intelligence.
SOAR . . . . . Security Automation and Orchestration.
ITSM . . . . . Information Technology Security Management.
CTA . . . . . Cognitive Task Analysis.
SA . . . . . Situational Awareness.
MTTR . . . . Mean Time to Respond.
IOC . . . . . Indicator of Compromise.

IPS . . . . . Intrusion Prevention System.
SLA . . . . . Service Level Agreement.
ISP . . . . . Internet Service Providers.
DDoS . . . . . Distributed Denial of Service
RF . . . . . Random Forest
KNN . . . . . K-Nearest Neighbour.
ICMP . . . . . Internet Control Message Protocol.
IoT . . . . . Internet of Things.
DGA . . . . . Domain Generation Algorithm.
CTI . . . . . Cyber Threat Intelligence.
FP . . . . . False Positive.
FN . . . . . False Negative.
OS . . . . . Operating System.
GDPR . . . . . General Data Protection Regulation.

1 Introduction

Contents

1.1 Motivation

Malware has evolved from viruses attacking single victims to more sophisticated malware such as botnets, ransomware, or Advanced Persistent Threats (APTs) used by attackers for monetising or much more disruptive purposes. Malware is the source of many Internet threats such as information or credit card theft (e.g. Aurora [2]), network service disruption through DDoS (e.g. DDoS on Estonia [3]), email spam (e.g. Pushdo [4]), ClickFraud (e.g. ClickBot.A [5]), and spreading malware (e.g. Zeus [6]).

Such sophisticated attacks are challenging to detect as they employ multiple attack vectors to achieve their objectives by targeting an organisation for data exfiltration, impeding critical aspects, or positioning itself to carry out these objectives in the future. Large multinational organisations hit with destructive

malware can experience an average cost of $239 million and lose an average of 12,316 devices[1].

Building out an organisation's cybersecurity capabilities can be a daunting task. The number of advanced and emerging threats will continue to outpace the capabilities of security personnel within organisations. Security Operations Centres (SOCs), a centralised group of security personnel [2] responsible for the 24/7 monitoring, detection, and response to security attacks, are the first line of defence in an organisation. Previous research suggests that security practitioners may benefit from better tools for their job [7–9] and a need for a better understanding of the human side of security operation centres [10, 11]. For example, recent work highlighted the significant technical and human-centric issues SOCs face [12]. Oftentimes SOC analysts need to manually go through alarms performing monotonous tasks, wasting time on lower priority alerts while the more critical ones slip by.

To enable SOC practitioners to work more efficiently and focus their efforts on detecting more challenging threats, additional levels of intelligence need to be incorporated into SOC operations. As the level of automation in the SOC scales upward, analysts will be free to embrace a more effective *"hunter role"* detecting the more dangerous threats.

In 2013, Target was hit by the most prominent retail hack in U.S. history that managed to infiltrate Target's network installing malware designed to steal customer's credit cards. Target has installed a new malware detection technology by FireEye a few months prior to the breach that cost $1.6 million. This technology creates a virtual honeypot environment, executing each file received over the network to determine if it is malicious. The tool was able to detect the malware used in the breach, generating an alarm that was picked up by Target's SOC in Bangalore. However, when the alarm was escalated to the Minneapolis SOC, it was ignored [13].

The deployed technology could have been able to automatically respond by deleting the malicious file, a capability that had been disabled by Target's security personnel. This configuration is not unusual, as most SOCs would want to avoid any automated decision that could cause business disruption - a lack of trust caused by the prevalence of false positive alarms created by detection tools. This breach resulted in more than 90 lawsuits filed against Target by customers and banks for negligence, compensatory damages and significant financial loss of $61 million for the breach response alone [13].

[1] `https://www.ibm.com/downloads/cas/XZGZLRVD`

[2] We will refer to personnel that work in a SOC as security practitioners or SOC practitioners interchangeably.

The malware used in Target's breach was far from sophistication. Nevertheless, the malware was able to bypass a large and resourceful organisation's security controls and procedures, hinting that the problem goes beyond just a SOC's technological capabilities. What this case study highlights is that the challenges in detecting malware through a SOC are complex and cut across multiple processes and technologies. To understand both the importance and the challenges of the SOC it is worth exploring two areas in more detail: (1) difficulties related to the monitoring for malware in SOCs and the role of humans (analysts) in configuring and using technology, and (2) weaknesses in the malware detection technologies.

1.1.1 Monitoring for Malware in SOCs

Malware produces a series of events or breadcrumbs on their path of destruction that are recorded in logs. Organisations collect and store billions of logs generated by the various security controls on their network. One of the leading technologies used in SOCs, which pools such logs together are Security Information and Event Management systems (SIEMs). Alarms generated by SIEMs rely on human analysts to review, as most of these alarms are False Positives (FPs). The volume of alarms generated by security tools are growing exponentially, hence, SOC practitioners are struggling with: (1) The task of separating threats posing a serious risk to the organisation from noise; (2) Identify which require immediate attention to focus the limited incident response resources.

False positives can overload security practitioners [10]. Oftentimes SOC analysts are overburdened by the monotony of having to manually evaluate alarms. As with the Target attack, although the technology was able to detect the malware, the analysts did not have the time to filter that information from the noise. Triaging the overwhelming number of alarms not only causes analyst fatigue and burnout [11] but is prone to human error [14]. 76% of analysts reported feeling that not enough time is spent on searching for emerging threats within their organisations' SOC [15]. The faster the data breach can be identified and contained, the lower the costs [16].

Identifying the highly prominent and high-risk alarms from tens of thousands to millions of alerts and alarms generated daily in an organisation is an important task. Although prioritisation computations embedded in the functionalities of the SIEM can do much of the heavy lifting, SOC practitioners are still faced with the difficult tasks of figuring out which alarms are most dangerous. For example, if malware is the source of an alarm, threat intelligence can indicate if that malware is linked to previous cyber-crime or espionage activity. In addition, determining the type of malware (e.g. worm, ransomware) can assist security practitioners in

determining the threat and the best response. Analysts use sandboxes and online engines (e.g. VirusTotal[3]) to get information about a malicious binary. However, in Target's case, although the FireEye technology detecting the malware gave a high risk score, the security team may have been skeptical about the information [13], hence not prioritizing the alarm.

1.1.2 Malware Detection Tools

Current solutions for detecting and managing malware could be categorised based on location of deployment as network-based or host-based malware detection systems. There are a number of advantages in adopting a network monitoring approach to malware detection. For example, most malware uses some form of network communication to contact their C&C servers to receive further instructions or launch their attacks [17]. New types of malware are emerging, called Fileless malware, which are memory-based and do not generate files and leave no footprint [18]. Thus, detecting such malware would be challenging for host-based solutions. Hence, in this thesis, we will focus on network-based malware detection solutions and discuss its challenges.

Network-based malware detection approaches could be categorised as signature-based and behaviour-based. Network Signature-based detectors create signatures of known malware communications. However, as malware continues to grow in sophistication, employing new tactics to evade detection, signature-based approaches are susceptible to evasion. For example, malicious software kits (e.g. exploit kits) are available for purchase from online criminal markets, providing updates and ensuring evasion of existing security countermeasures [19]. Moreover, one of the drawbacks of signature-based approaches is the high false positives (i.e. 99% FPs [20]) and the lack of continuously adapting to changing malicious behaviours [21].

Behavioural detection approaches overcome these limitations by relying on behavioural malware features. Such systems are capable of detecting malware even when code evasions are employed. For example, the FireEye technology deployed by Target executes files in a virtual machine, observing its behaviour. However, behavioural malware detection approaches also have their challenges. For example, behavioural content inspection technologies can be bypassed by network traffic encryption.

In general, network-based monitoring technologies are built on the assumption that threats are observed as they enter the network in specific perimeter points at

[3] `virustotal.com`

the Internet edge. However, this assumption is no longer valid in modern networks. Well defined security perimeters no longer exist due to the heterogeneous nature of Internet connectivity in organisations and the adoption of technologies such as cloud computing, VPN, the borderless architecture of the IoT, and the increase of mobile devices and bring-your-own devices (BYOD).

Attackers have also grown in sophistication and regularly employ new methods to hide their activities and how they bypass these perimeter intrusion detection devices. They utilise stealthy malware that is very discrete, traversing in the network very slowly, taking days, weeks, or months to accomplish their objectives to avoid detection. Hence, as networks grow in size and speed, detecting such malware will be a difficult task.

1.2 Research Gaps

Designing practical tools requires us to understand the users who will be using the tool, to ensure that it meets their needs. Similarly, designing monitoring and detection tools without getting the SOC practitioners feedback make them less likely to be effective. Due to the sensitive nature of the work, gaining access to SOCs to understand its processes, technologies and the SOC practitioners' responsibilities is not trivial. Therefore, there is still a lack of systematic research and a rich understanding in this area, making it difficult for developers and researchers to improve SOC tools.

Previous research suggests that security practitioners may benefit from better tools for their jobs [8, 9]. However, understanding the analysts' cognition and the processes they follow when investigating an incident is not easy. Analysts themselves are unable to explain why and how they investigate an incident, relying on their experience in dealing with a similar event in the past and human intuition [22]. In addition, SOCs themselves are different, with different structures and focus [23], and providing a generalised solution is not feasible.

Security monitoring is a human-centred process with network detection tools to support the work of analysts, alarming on possible intrusions, and presenting the analysts with the information needed to make a decision about a potential threat. SOCs utilise SIEM solutions to monitor the organisations' activities. Although these commercial solutions are to some level helpful, they are mainly used as a data aggregation tool, presenting the aggregated information to the analysts who are left with putting together pieces to solve a puzzle. Such technologies don't address the operational needs, and lack comprehension of how analysts think and

work. Analysts (humans) are essential in a SOC, interacting and working together, using the information provided by the SIEM, and following specific workflows based on the incident at hand.

Today, more companies are outsourcing the security management and network monitoring to companies that offer SOC as a service, to escape significant investments and mitigate their lack of cybersecurity expertise [24]. Although these companies monitor the client's systems and networks for signs of malicious activity, they may not have direct access to the client's hosts for privacy or policy reasons. When a client's host is suspected of being infected by malware, gaining access to that host and retrieving the malware binary for analysis is a challenge. However, existing dynamic analysis tools for malware classification require SOC analysts to have access to these malicious binary executables. Even if access to hosts is possible, the time required to negotiate this access and retrieve the executable, before its analysis, is not efficient where speedy detection and response are critical. For example, in situations where the implications are severe, such as in ransomware attacks, classifying the attack as ransomware is critical, ensuring rapid response to avoid catastrophic implications.

Behavioural malware detection systems continue to dominate the most effective solutions for malware detection. One of the main limitations for many previously proposed network-based behavioural monitoring systems is their reliance on content-based features. Such technological solutions are privacy-invasive and not favourable in a real-world deployment, especially when monitoring is outsourced to a third party and with clients that require security clearance. Gartner estimated that by 2019 80% of organisations' web traffic would be encrypted (i.e. using HTTPS) [25]. Regulations such as General Data Protection Regulation (GDPR) require the enforcement of appropriate security measures for protecting personal data; hence, traffic encryption is favoured to ensure compliance. However, similarly attackers are increasingly employing HTTPS to conceal malware and evade detection, therefore, developing encryption resilient inspection measures is a requisite.

More than 317 million new variants of malware were observed in 2014 (i.e. malware automatically generated by modification of a previous malware binary that results in a new hash), an estimate of 1 million unique malware variants each day, increasing the total number of malware to approximately 1.7 billion [26]. Proposed detection solutions need to adapt to this growing threat landscape, developing systems that adapt to malware use of new attack tactics, and are resilient to malware evasion and obfuscation.

Detecting malware in a SOC is extremely challenging, with significant risks to those businesses that make mistakes. As such, it's important to enhance our knowledge around automation, reducing the cognitive burden on SOC analysts, and creating tools that produce fewer false positives and detect increasingly complex attacks. The novelty of this thesis is to bridge that gap and thereby set a better foundation for future research in the field.

1.3 Problem Statement and Research Aims

This thesis aims to explore the challenges in malware detection in SOCs as identified in Section 1.1, while considering the research gaps identified in Section 1.2. Hence, the aims of the research is twofold:

- Using quantitative and qualitative methods, we investigate the role of SOC dimensions (people, process, technology) on malware monitoring, detection and response.
- We design and evaluate novel malware classification and detection systems that improves over existing state-of-the-art tools.

This thesis is structured as four main research questions discussed in the following section.

RQ1: How do security practitioners monitor, detect and investigate security threats in a SOC?

In this contribution, we study the workflow that security practitioners in SOCs follow, identifying human-centric tasks that might benefit from automation. Through an online survey and semi-structured interviews with security practitioners, we recognise the role that humans play in the SOC operations and their interactions with the technological solutions. Our results highlight the overwhelming reliance on analysts throughout the SOC's operations, which might benefit from automation.

Through scenario-based questions, we elicit the analysts' analytical thinking when making decisions, identifying the influential factors that might impact their decision. Moreover, we revealed the existence of a number of contradictions with the SOC operations. We conclude with concrete recommendations for building automated systems for SOCs, to address human-centric tasks and opportunities for further research.

RQ2: What are the advantages and disadvantages that security practitioners perceive of network-monitoring tools?

In this contribution, we focus on providing an understanding of security practitioners' perspectives of the security monitoring tools deployed in SOCs and their perception of the high false positive rates. Through an online survey and semi-structured interviews, we identified common misconceptions of the definition of false positive, and a significant distinction between *false alarms* and *benign triggers*, which needs to be recognised when evaluating security tools in real-world use.

Our results highlight that, despite the high number of false positives, most are attributed to benign triggers — true alarms that can be explained by benign behavior within the organisation's environment. We identified analysts' perceptions of the strengths and weaknesses of SOC tools, challenges in network monitoring in SOCs, and how contextual knowledge can be incorporated into security-monitoring tools to reduce the number of false alarms. We conclude with concrete recommendations for addressing the main weaknesses of security tools identified by SOC security practitioners.

RQ3: To what extent can we classify malicious network traffic into a malware family, on-the-wire?

Understanding what malicious activities a malware is performing is an essential step in determining the best remediation mechanism. One way to understand malware behaviour is to classify it to a known malware family. By knowing the malware's family, public resources (e.g. VirusTotal [4]) can be utilised to learn about how to deal with this infection.

We propose a system that classifies network flows to a malware family. The proposed system utilises privacy preserving malware network behavioural characteristics for classification. It is effective in classifying a binary to a malware family based on its network traffic; hence, access to the malware binary and the infected host itself is not required.

RQ4: To what extent can we detect malicious network traffic generated by new or unseen malware?

Behavioural malware detection approaches are found to be the most reliable by SOC analysts. In contrast to traditional signature IDS, detecting malware based on behavioural features increases the probability of detecting unknown malware.

[4] `virustotal.com`

We propose a system that detects malware-infected hosts on a network by monitoring the network communications activity. Using Markov Chain Models, it identifies malware network communication patterns during the active propagation (e.g. scanning), C&C communication, and operational stage (e.g. DDoS, SPAM, ClickFraud). Due to its ability to detect various stages of malware network behaviour, it can then detect traffic originating from new or unseen malware binaries making it robust against malware evolution.

1.4 Thesis Organisation

This thesis is structured as follows to address the previously presented research questions. In Chapter 2 we present the background information relevant to this research: network-based monitoring solutions; SOCs and their main technologies; and malware network communications. In Chapter 3, we present a literature review, focusing on network-based malware classification and detection approaches and those related to the understanding of SOCs. The literature review highlights the research gaps we go on to address in later chapters. Chapter 4 represents the methodology chapter, complementing the research methods sections reported in each chapter.

In Chapters 5, 6, and 7 we address the problem of the lack of literature on SOC technologies and processes in monitoring, detecting and responding to malware. Specifically, Chapter 5 commences our study through a quantitative approach with an online questionnaire to extract the SOC practitioners' perspective on network-monitoring tools. Through this study, we identify the compelling research questions, further investigated through a qualitative approach. In Chapter 6, we address **RQ1** and provide an understanding of SOC processes, highlighting human-centric tasks that could benefit from automation and factors impacting these processes. We then, in Chapter 7, address **RQ2** and identify the strengths and weaknesses of network monitoring tools, representing recommendations for SOC tool developments.

In Chapter 8, we address **RQ3** and propose a behaviour-based malware family classification system that can determine the family of the malware running on a host by analysing its network traffic. Examining the network behaviour of this malware will assist analysts in determining the objective of the malicious executable, prioritising threats and determining the best response measure to employ.

In Chapter 9, we address **RQ4** and propose a behaviour-based bot detection system that improves over state-of-the-art by detecting new or unseen malware. The system uses behavioural high-level network features, preserving the privacy of the monitored hosts. Not only can analysts detect new, never-before-seen malware, but

they are also able to determine what malicious activities the malware is launching and choose the most effective response measure. Chapter 10 concludes the thesis and proposes future research.

1.5 Publications

The work described in this thesis has resulted in peer-reviewed publications, with additional papers still under review. The publications are derived from chapters, as illustrated in Figure 1.1, which also illustrates the relationship between research questions and the thesis structure.

In all publications, I was the first author and was responsible for the research reported and for writing the paper, with some writing contributions from the other authors. The survey and interviews reported in Chapters 3, 7, and 6 were part of a wider set of interviews carried out jointly with another author (Louise Axon); the questions relating to this thesis were devised, asked, and analysed by myself.

1. Alahmadi, B.A., Axon, L. , Webb H. and Martinovic, I., Down the Rabbit Hole: Understanding Security-Monitoring Processes in Security Operations Centres. (*Under Review S&P'21*)

2. Alahmadi, B.A., Axon, L. and Martinovic, I., 99% False Positives: On SOC Analysts' Love-Hate Relationship with Security Monitoring Tools. (*Under Review Usenix Security'21*)

3. AlAhmadi, B.A. and Martinovic, I., 2018, May. MalClassifier: Malware family classification using network flow sequence behaviour. In 2018 APWG Symposium on Electronic Crime Research (eCrime) (pp. 1-13). IEEE.

4. Alahmadi, B.A., Mariconti, E., Spolaor, R., Stringhini, G. and Martinovic, I., BOTection: Bot Detection by Building Markov Chain Models of Bots Network Behavior. (*AsiaCCS'20*)

I was also lead author and a co-author in other publications that are outside of the focus of this thesis.

1. Axon, L. Alahmadi B.A., Nurse J.RC , Goldsmith M., Creese S, Sonification in security operations centres: what do security practitioners think? (*Workshop on Usable Security (USEC) at the Network and Distributed System Security (NDSS) Symposium 2018*)—Best Paper Award

2. Axon, L. Alahmadi B.A., Nurse J.RC , Goldsmith M., Creese S, Data Presentation in Security Operations Centres: Exploring the Potential for Sonification to Enhance Existing Practice (*Journal of Cybersecurity 2020*)

3. Alahmadi, B.A., Legg P.A, Nurse J.RC , Using Internet Activity Profiling for Insider-threat Detection. (*12th Special Session on Security in Information Systems - WOSIS 2015*)

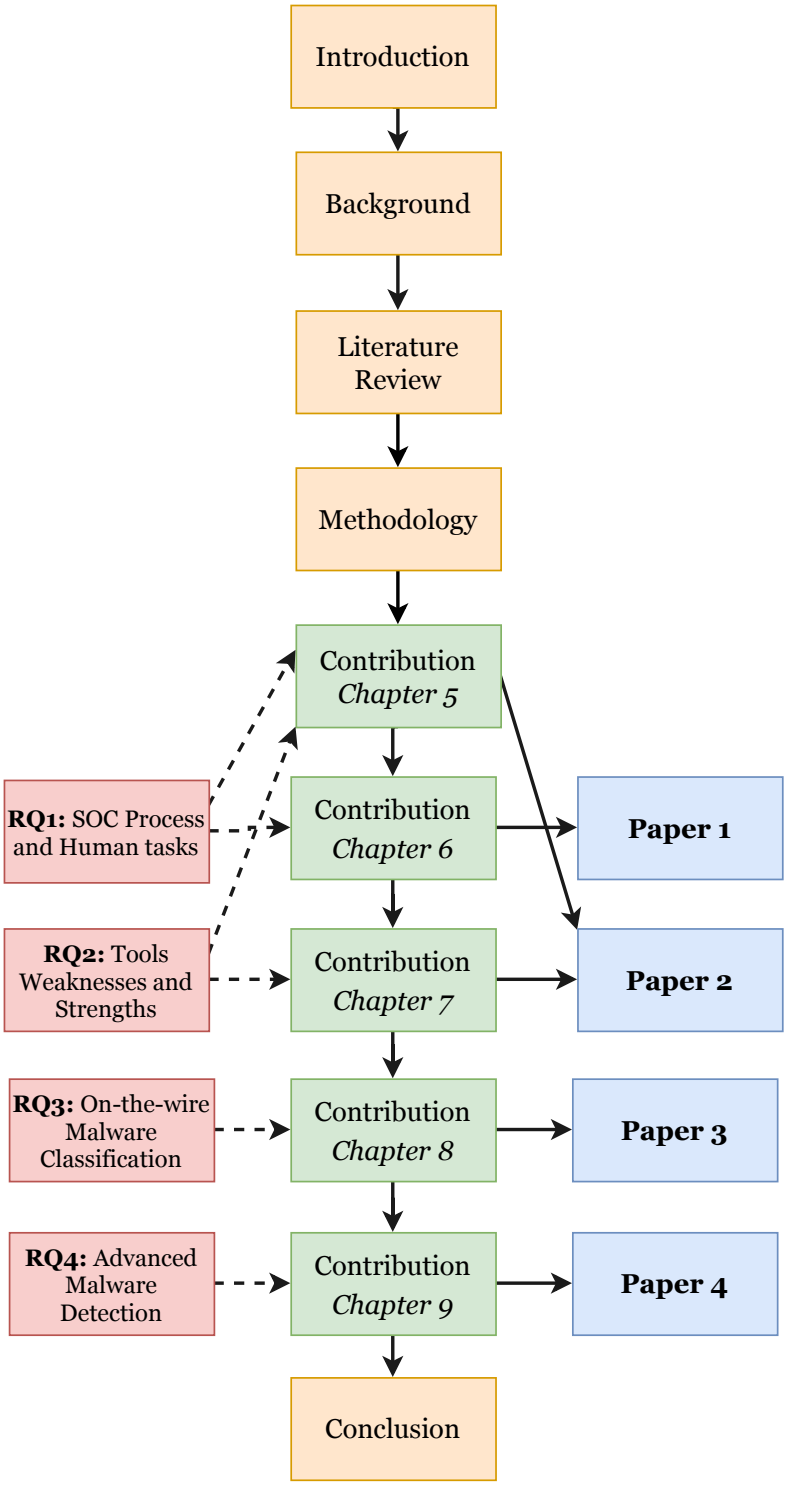

Figure 1.1: Relationship between research questions, thesis structure and publications.

2 Background

Contents

2.1 Introduction

In this chapter, we present a background overview to introduce the reader to the concepts and terminology used in subsequent chapters.

We start by describing the security operation centres (SOCs) landscape and the integration of people, process and technology. We focus on describing the primary technology used in SOCs —SIEMs. We present the SIEM architecture, and how existing monitoring technologies and security logs are integrated within. We then go on describing the botnet architecture and the various network communications it launches, which serve as a baseline of what malware detectors should detect.

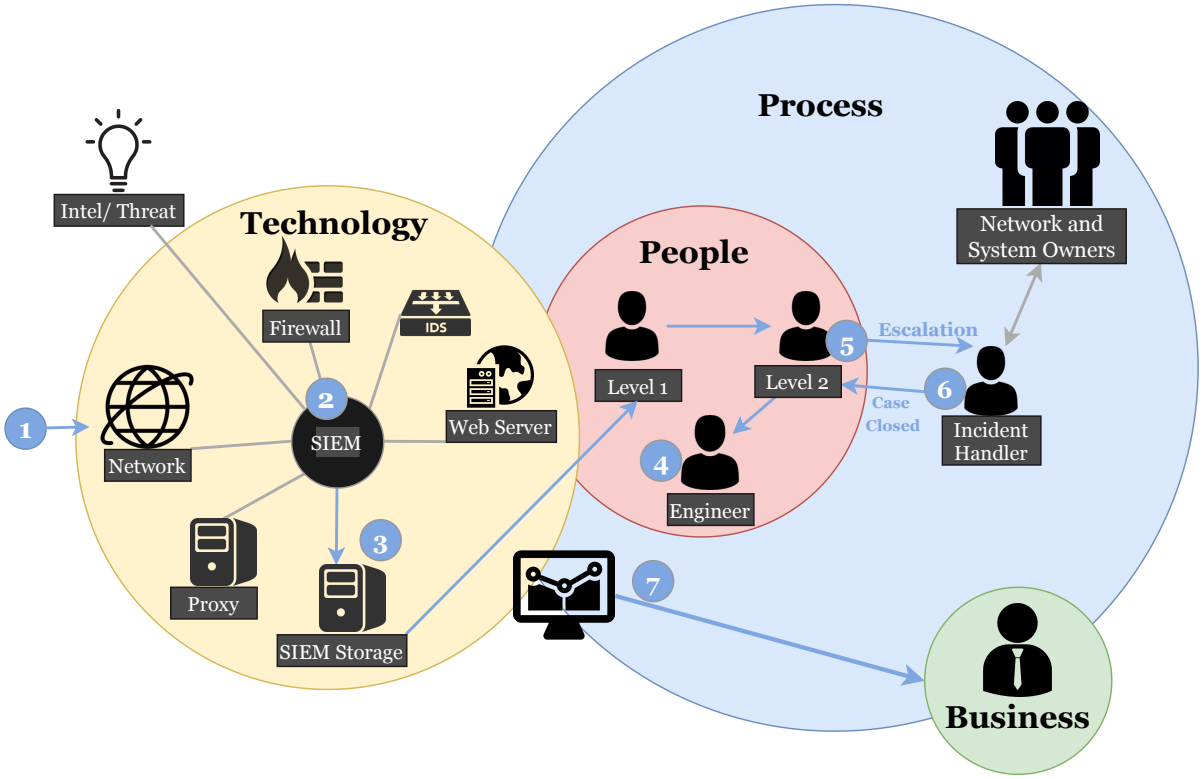

Figure 2.1: Example of a SOC structure

2.2 Security Operations Center (SOC)

Security Operations Centres (SOCs) are a centralised unit providing monitoring capabilities for the detection, escalation and recovery of security incidents on an organisational and technical level. Once a security incident is detected, the SOC aims to contain the attack as soon as possible, to limit the potential damage, saving the organisation money, data exfiltration or reputational damages.

There are different types of SOCs as discussed in [23], which could be classified by the services they provide, their capabilities or maturity [27]. Moreover, they can mainly be categorised as (1) in-house SOC, meaning the organisation builds and staffs the SOC for its organisation; (2) Managed Security Service Provider (MSSP), where an organisation hires a third party, outsourcing the threat monitoring, detection, and response. Some customers combine these two approaches, building their own SOC but also hire a MSSP to uplift their skill set by doing joined monitoring. There are several reasons why an organisation would use one over the other. For example, their budget, their lack of security expertise, or to avoid the setup costs of a SOC [24, 28].

SOCs consists of complex processes and technology and involves multiple people from within the SOC, organisation, customer and other third parties. We show

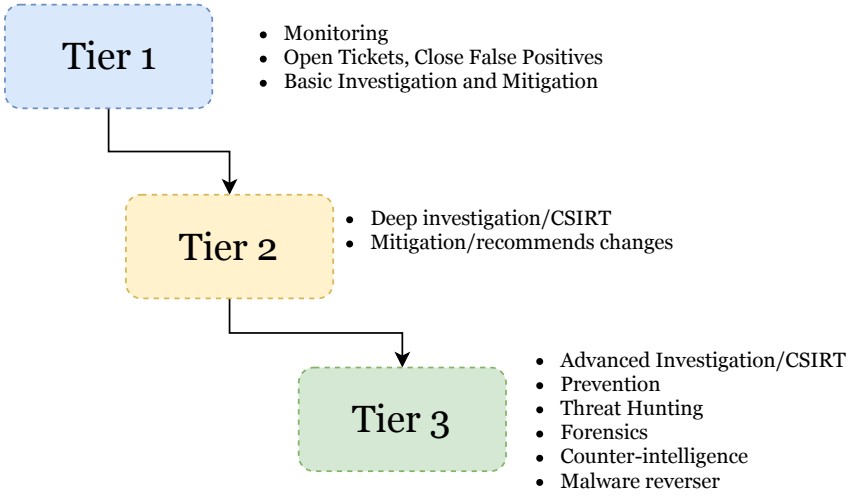

Figure 2.2: Example of a three-tier SOC and personnel responsibilities, adapted from [30]

in Figure 2.4 an example for a SOC structure and its three dimensions (people, process, technology). In such a structure, when a threat enters the network, security devices and logs detect this threat. These detection are collected by the SIEM that produces an alarm. The alarm is verified by level 1 analysts and is escalated to incident handlers to handle the incident by liaising with the system owners. Reports of organisations threats are communicated to the business management.

When discussing the challenges in threat monitoring in SOCS, it's essential to identify first the various Tools, Processes and People involved in the operation of the SOC.

2.2.1 People

The SOC team's goal is to detect, analyse, and respond to cybersecurity incidents using a combination of technology solutions and a strong set of processes. To do that, analysts have to maintain Situational Awareness (SA) of events from the systems and networks they monitor. Situational Awareness is defined as: "*Within a volume of time and space, the perception of an enterprise's security posture and its threat environment; the comprehension/meaning of both taken together (risk); and the projection of their status into the near future.*" [29].

The SOC team is typically staffed with security analysts, engineers, incident responders, hunters, contractors, as well as managers who oversee security operations. SOC engineers are responsible for providing and supporting the SOC with the required software (e.g. SIEM scripts or configurations). Incident responders handle

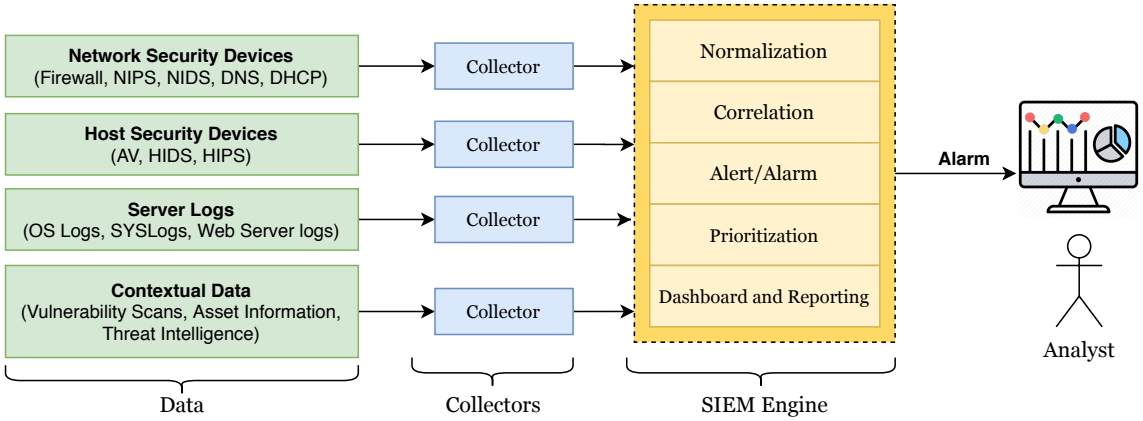

Figure 2.3: SIEM Architecture

events that are escalated by the analysts that need in-depth investigation and forensics. SOC practitioners' team structure and responsibilities varies [23]. For example, analysts can be categorised by their tasks to multiple levels, described in Figure 2.2.

2.2.2 Process

The SOC process involves the various workflows SOC security practitioners follow in their everyday tasks. For example, this includes process followed for SIEM monitoring and alarming, event management process, security incident ticket management, incident handling, reporting and escalation process. All these processes are documented in a Wiki Portal, share point or share drive for team reference.

There are standard guidelines that direct analysts in SOC operations. For example, The European Network and Information Security Agency (ENISA) [31], and the National Institute of Standards and Technology (NIST) [32] have published guidelines for incident response teams. To provide a rapid automated response for incident identification, detection, response to SOC team communications, procedures are documented in a *playbook* [33], a document prepared by experienced SOC analysts that details steps an analyst follow to deal with a security alarm.

2.2.3 Technology

SOCs deploy various technological solutions such as Asset Discovery, Vulnerability Assessment, Behavioural Monitoring, Signature-based Intrusion Detection Systems, and SIEMS. We discuss in the following the main platform deployed in SOCs.

SIEMs

One of the most frequently chosen tools in a SOC is Security Information and Event Management (SIEM)— e.g, ArcSight [1], AlienVault [2]. SIEMs are systems that combine SIM (security information management), and SEM (security event management) functions into one security management system. SEM deals with real-time monitoring, correlation of events, notifications and console views. SIMs provide long-term storage as well as analysis, manipulation and reporting of log data and security records of the type collated by SEM software. SIEMs replace the need for analysts to access traditional security tools directly. Instead, the SIEM aggregates the logs from the multiple data sources, and processes them to detect threats.

We show the SIEM architecture for monitoring and detection of alerts in Figure 2.3. The first step is to parse the data collected from the data sources and normalise it to a standard format produced as a security event. Multiple security events may be correlated to create a correlation rule or alarm. When the rule is triggered, an alarm is fired and prioritised. Then, it is up to the analyst to determine if the alarm is a false positive (FP) or it needs to be escalated. We discuss each step in further detail in the following.

Data Sources— The SIEM has data source plug-ins called collectors where data sources are fed into it. These data sources could be either raw logs or security events generated by security devices. Such data sources could be, for example, network-based security tools (e.g. firewalls, IDS), host-based security tools (e.g. Anti-Virus), logs (e.g. operating systems logs, web server logs). Contextual data provided by threat intelligence platforms and other processes (e.g. vulnerability scans) could also feed into the SIEM.

Ideally, these data sources are received from Security Device Event Exchange (SDEE) enabled devices/hosts. SDEE is a standard proposed by the International Computer Security Association (ICSA)[3] that specifies the format of messages and protocols used to communicate events generated by security devices. SDEE enables devices to collect logs in the device itself, and the SIEM retrieves these logs. If the raw logs match a specific criterion, then part of the message is inserted into the SIEM database as a security event. Other devices send the logs directly to the SIEM to store.

Normalisation— Logs/messages received from SDEE enabled devices are intrinsically suited to the SIEM. They do not require manipulation because they

[1] `https://www.microfocus.com/en-us/products/siem-security-information-event-management/overview`

[2] `https://cybersecurity.att.com/solutions/siem-log-management`

[3] `https://www.icsalabs.com/`

are in the right format. However, some applications/devices were never designed to generate logs. Therefore, they have to be heavily edited by scripts to produce a log that will fit the SIEM's requirements.

Correlation— Most alarms in a SIEM are directive alarms, also known as correlation rules. Using correlation rules, SOCs can identify potential security threats by detecting behaviour patterns in disparate yet related events. Directive alarms link these different events to generate an alarm that is more useful than any event seen in isolation. If the organisation has a particular threat use-case, then they can create their own directives.

Alarming and Prioritisation— Its worth noting that there is a distinction in the definition of alarm, alert, and event in a SOC operation. Security tools and networking devices produce alerts when they detect a threat. Similarly, these threats might be written in logs as an event. The alerts and events are then aggregated through the SIEM to produce an alarm.

When multiple alarms are flagged, a frequent scenario in a SOC, what alarm is looked at depends on what is monitored. For example, if an organisation is only concerned about where data is going, then network traffic and IDS alerts alone might be sufficient. However, receiving multiple alerts from different sources related to the same asset provides assurance of the validity of the alarm. How they choose the alarms they investigate depends on several factors, which could be defined using the prioritisation module in the SIEM engine.

Reporting and Visualisations— Alarms produced by the SIEM are presented to the analyst through visualisation (e.g. dashboards). In addition, the SIEM provides reporting capabilities, meaning analysts can auto-generate reports.

2.3 Malware Network Communications

Malware acts on commands received from one or more servers (*C&C servers*), all controlled and managed by a human controller (e.g. *botmaster*) to achieve a malicious purpose. Network communications are essential for the successful operation of the malware, as it produces extensive network traffic, communicating with the C&C server to receive, propagate, or execute commands.

Malware Detection is categorized by [34] as *Bot Detection*, *C&C Detection*, and *Botmaster Detection* (i.e. malware author). The main focus of this thesis is the *Bot Detection*, thus detection of malware infected machines (i.e. bots). We discuss in the following the network communications generated by bots during its different stages.

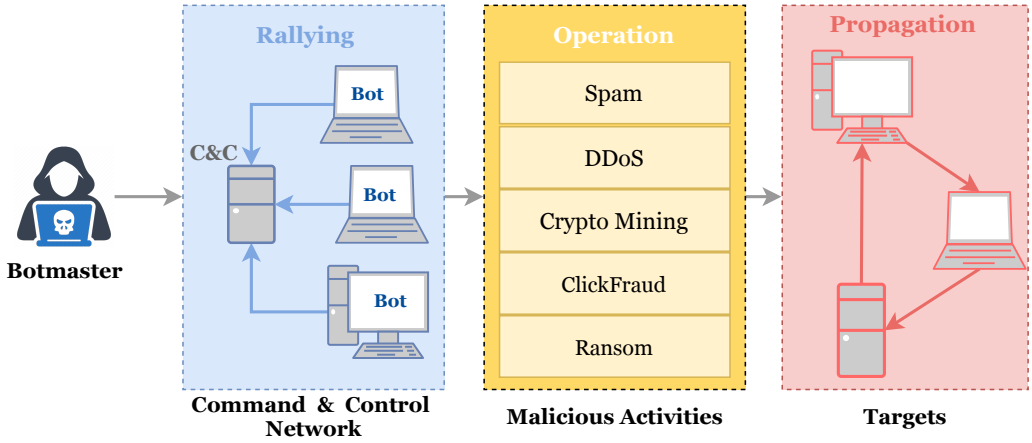

Figure 2.4: Botnet Command and Control Architecture

2.3.1 Propagation

There is strength in numbers, and the power of the botnet increases when the number of bots grows. Therefore, bots aim to increase the number of bots by recruiting new bots through propagation. The propagation process may initially start with a synchronisation stage. Bots need to synchronise/coordinate with each other through obtaining a seed input that is used by the bot Domain Generation Algorithm (DGA) [35]. This input could be a synchronized system time, for example, *Conficker-A* sends an empty HTTP request to a popular website such as *google.com* to get the current date/time, or *Torpig* [35] uses trending topics on Twitter or the Stock exchange. This synchronised token is then used as input to the Domain Generation Algorithm (DGA) to communicate with C&C server domain name. The bot then attempts to infect another machine either through active means (e.g. scanning) or passive propagation (e.g. phishing emails). Some bots employ multiple stages of infection, and once the victim is infected with the bot binary it downloads additional files from the C&C server. When the infection process is complete, the new bot registers with the C&C server. Propagation could be categorised as passive or active based on the recruitment mechanism used.

Active— Active propagation means that the bots exhibit worm-like behaviour, actively attempting to infect new victims, using existing vulnerabilities (e.g. Duqu 2.0). Bots leverage vulnerabilities that enable them to gain administrative privileges on the infected machine [34]. Some bots perform a network scan before the actual infection (e.g. Conficker), either by sending ICMP echo requests to detect reachable hosts or port scanning to detect vulnerable services (e.g. Neris) [36]. For example, Sality performed a scan of the entire IPv4 address space for 12 days [34]. Phatbot could either have hard-coded configuration or be instructed to scan a certain network

range [36]. Botnets may rely on scanning to propagate to reduce resources cost or as a way of obfuscation if the attack operation's communication protocol is used seldom on the victim network [36]. However, scanning is noisy and results in additional network traffic; thus stealthy bots may resort to a passive propagation approach.

Passive— Bots can propagate through other means such as phishing emails, drive-by-download or infected removable media, known as Passive propagation. For example, *Zeus* is distributed over phishing emails and drive-by-download. Although the underlying network communications resulting from passive approaches (e.g. receiving a phishing email, visiting a website) can be detected, correlating these activities with the actual bot infection is difficult using network-based detectors [36]. This is as a result of the passive propagation and the actual binary infection possibly occurring in different times [36].

2.3.2 Rallying

The first step a new bot does is registering with the C&C server, a process also known as rallying [34]. There are several techniques that bots use to contact the C&C server. The basic approach is to contact a domain hard-coded in the bot binary. However, this approach is vulnerable to domain takedowns, and when the domain becomes unreachable then the bot can no longer receive commands. Some bots use a *Domain Generation Algorithm* (DGA) (e.g. Zeus, Conficker), generating a different domain name from a changing seed input obtained in the coordination stage. This makes taking down the botnet domain a more complicated process; once a domain is taken down by the authorities, the bots can simply connect to the next domain generated by the DGA.

To add to the layer of resiliency, the bot can apply fast-fluxing where multiple bots register and de-register their IP addresses to the same DNS host record, mapping a single domain name to possibly thousands of bots. Thus instead of bots connecting to the C&C directly, they connect to one these active proxy bots that rally the messages to the C&C server. *Storm* uses Double-flux, a sophisticated fast-flux approach where these bots register their IP addresses as part of DNS Name Server record list for the DNS zone.

2.3.3 Operation

As part of the bot's operation, the bot can attempt to infect new victims through port scanning to detect vulnerable services, or ICMP scans to detect reachable hosts. Botnets could be used to carry out various types of attacks, each having different network flow communication structures.

For example, bots could be used to launch *Distributed Denial of Service Attacks* (DDoS), sending huge amounts of requests (*usually TCP/UDP HTTP requests*) to overload systems and seeking to make a machine or network resource unavailable. Bots could also be used to send unsolicited emails or SPAM, usually using Simple Mail Transport Protocol (SMTP). ClickFraud is another attack that uses bots to access advertising URLs (pay-per-click ads), forcing advertisers to pay for irrelevant clicks, leveraging HTTP and HTTPS protocols.

Botnets also utilise network communications for other operation tasks, such as scanning the network to collect network information (e.g. Duqu 2.0, Phatbot), login information harvesting (e.g. Storm), self-updating (e.g. Zeus, Phatbot), bitcoin mining (e.g. Miner), receiving C&C instructions (e.g. Stuxnet), or network traffic modification (e.g. Miner). Other protocols bots might leverage include Server Message Block (SMB) (e.g. Duqu 2.0) and Internet Relay Chat (IRC) (e.g. Rbot), all visible in a bots network traffic. More details on the bots communication patters could be found in [36].

2.4 Summary

We provided in this chapter an overview description of SOCs, focusing on describing its leading technological platform (i.e. SIEM) and the monitoring and detection of malware. SOCs have various other responsibilities and apply other tools that we do not address and are outside the scope of this thesis.

Malware requires network communication to interact with its C&C, execute its attack operations and propagate to recruit further bots. Network-based malware detection systems take advantage of malware's need for network connections, spotting malware by monitoring for malicious traffic. Substantial previous work has been proposed for malware network detection. We review the literature on network-based malware detection and its deployment in SOCs in the following chapter.

3
Literature Review

Contents

3.1 Introduction

In this chapter, we provide an overview of the existing literature on network-based malware detection systems used in SOCs. Such technologies can be divided into Network-based Intrusion Detection Systems (NIDS) and Malware-specific detection systems (i.e. techniques tailored to malware detection). NIDS can be signature-based or anomaly-based NIDS. Machine Learning (ML) can be applied to either type (i.e. ML-based NIDS). For malware-specific technologies, in addition to providing a review of network-based solutions, we also discuss host-based approaches for completeness.

These technologies can be deployed in a SOC to generate an alarm on its own or be fed into a SIEM to be combined with other alert sources as part of a correlation rule. Such an aggregation or *fusion* of multiple sources is a known functionality of a SIEM. Hence, in this chapter, we also discuss related work on multi-log and multi-source data fusion. We then conclude with a review of related

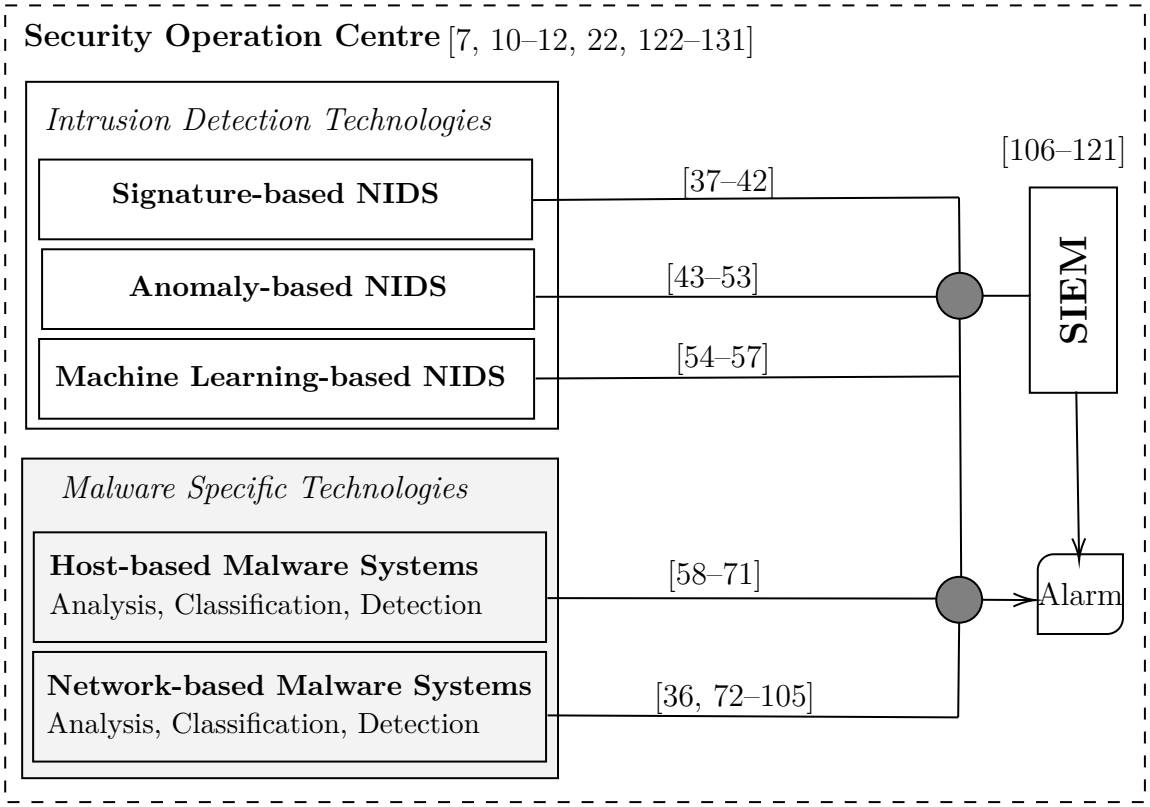

Figure 3.1: Literature Review Structure

work on Security Operations Centres (SOCs) and the challenges therein. We give an illustration of this structure in Figure 3.1.

3.2 Network Intrusion Detection Systems (NIDS)

Intrusion detection systems (IDS) were first proposed in 1987 by Denning [132], and since then, research on how to effectively detect malicious activities has grown. The primary purpose of IDS is the detection of any attempt to compromise confidentiality, integrity, availability, or simply to bypass the security mechanisms of a computer or network [133].

IDS can be deployed at the host or the network level, and a lot of work has been done in surveying and classifying IDS [109, 134–138]. Host-based IDS analyse system events and the behaviour of users on a single machine. Although this provides a more concentrated and detailed analysis, it needs to be deployed on all devices, causing a deployment and monitoring overhead. On the other hand, a network-based IDS protects all devices as it monitors only network traffic. We review herein network-based IDS and refer to a survey on host-based IDS [137].

NIDS monitors the network traffic, analysing the network, transport, and application protocols to identify suspicious activities. In NIDS monitored network packets are analysed for rule violations by a pattern recognition algorithm [139]. Depending on the model of detection, network-based IDs can be categorized as either signature-based or anomaly-based. A signature detection system identifies patterns of network traffic presumed to be malicious while anomaly detection systems compare network activities against a "normal" baseline. Garcia et al. [140] produced a survey that analyzes and compares the most important efforts carried out in a network-based detection area.

Signature-based IDS detect attacks by monitoring the network for pre-defined signatures of known attacks [37–42]. This approach is useful in detecting known attacks with a low false-positive rate. However, it is inadequate in detecting unknown attacks such as new or polymorphic malware. It requires a signature to be defined for all of the possible attacks that an attacker may launch against a network, thus requiring frequent rule-base updates and signature updates.

There have been several research papers on network-based anomaly detection (i.e. [43–52]). Anomaly-based IDS initially learns the normal network behaviour, then monitors for any deviations in this behaviour [138]. A survey on commercial anomaly-based IDS is provided in [53]. Compared to signature-based IDS, anomaly-based IDS could detect unknown intrusion attacks known as "zero-day" attacks. In addition, the network profiles of normal activity are unique to every network, making it very difficult for an attacker to know with certainty what actions it can carry out without being detected [139]. However, Anomaly-based IDS has its drawbacks. For example, they produce a high percentage of false positives, fail to scale to gigabit speeds, and have difficulty determining which specific event triggered an alarm [139]. The high rate of false positives in these alarms result in attacks or malicious activities on the network going undetected for periods of time. Since security analysts are looking for abnormal activity rather than a specific attack, they are prone to be overwhelmed by time-consuming false alarms, thus increasing the time until they detect the attack. We refer the reader to [141], a survey on anomaly detection.

Developing intelligent models for network-based intrusion detection systems is not trivial. NIDS deal with huge network volumes, highly imbalanced data distribution, and the challenge of recognising decision boundaries of normal and anomaly behaviour. In addition, NIDS needs to be adaptable to the constant change in networks without a lot of manual tuning [142]. A robust and effective NIDS needs to solve these challenges and be designed to adapt to different patterns of network traffic, to successfully detect unknown malicious attacks [21].

Goodall et al. [9] investigated security practitioners perspective on NIDS. The authors conducted semi-structured interviews with Intrusion Detection System (IDS) experts to understand how they use IDS. The study shows that IDS tasks require a combination of common knowledge (e.g. network and security) as well as situated knowledge (e.g. knowledge of normal network behaviour). However, the authors found that situated expertise might be challenging to transfer to new environments. Moreover, IDS tasks are collaboration driven, requiring cooperation from expertise within the organisation and community.

3.2.1 Machine Learning-Based NIDS

One of the drawbacks of intrusion detection techniques is the high false positive and false negative detection rates and a lack of continuous adaptation to changing malicious behaviours [21]. The research community applied Machine Learning (ML) to improve the accuracy of NIDS and reduce the false positives [21]. Such methods used ML algorithms such as decision trees [58, 59], Naive Bayes [60–62], rule induction [63, 64], Neural Network [65, 66], Support Vector Machine [67, 68], Genetic Algorithm [69], Fuzzy Logic [70], and Data Mining [71]. There have been several surveys in the application of machine learning and artificial intelligence in intrusion detection [143, 144], and we refer the reader to them for evaluation of their performance and detection accuracy. Moreover, the application of machine learning in intrusion detection has its challenges. Such challenges include, the high costs of errors, the variability of benign traffic, challenges in performing sound evaluation and its operation in an adversarial setting [145].

One challenge to evaluating ML-based NIDS is the lack or reliable real-world datasets. Many previous research rely on old datasets such as Knowledge Discovery in Databases (KDD) 1999 [146] Defense Advanced Research Projects Agency (DARPA) 1999 [147], which are more than a decade old, collected in the late 90s to evaluate its detection performance. As found by Azad [148], out of the 75 research papers reviewed, 46 used either the KDD or DARPA datasets while the other 29 chose something else. Wu et al. [21] summarize the most popular datasets used in intrusion detection system research.

The cybersecurity landscape is changing rapidly, thus, these old datasets are no longer considered to reflect the current security threats [145, 149, 150]. In fact, the lack of good public real-world datasets available for evaluation makes proposing an effective IDS challenging. There are many reasons why obtaining such datasets is difficult. For example, network traffic contains confidential or sensitive data that introduces restrictions on its use. As a result, organisations are not keen to share

their network data with researchers. In addition, the anonymisation efforts of these data are not always effective to motivate the safe sharing of network data [151]. The mass volume of network traffic also presents obstacles in data storage and computation for researchers. Moreover, data of such volume are often unlabeled and lack ground truth needed for system validation. As a result of these challenges, researchers may settle by taking the proposed system to where the data is held, for example, through collaborations with industry partners which introduces challenges in Intellectual Property (IP) ownership. Due to the aforementioned challenges, researchers usually resolve to lab simulation, which may not produce realistic results.

We reviewed the literature related to NIDS that monitor the network for a specific attack signature (signature-based) or any deviation of the network's normal behaviour (anomaly-based). Although IDS aim to detect malicious activities in a network, it is difficult for IDS to detect a malware-infected host, as they do not consume a lot of bandwidth and so fly under the radar. In the following section, we review research contributions that focus on detecting specifically malware-oriented attacks.

3.3 Malware Analysis, Detection, and Classification

Malware is a constant cyber challenge for organisations, and research efforts, such as [152–155], have aimed for a better understanding of their behaviour. Most malware use network communication to contact their C&C servers [17] and launch their attacks. Hence, there have been various approaches and methodologies to its detection through network monitoring and analysis and using different characteristics of the network traffic. The research community was active in proposing techniques to collect [156], analyze [157], classify [158] and detect malware [159, 160].

Host-based Malware Analysis and Family Classification— Perhaps the main challenge for anti-malware vendors is the growing stream of incoming malware samples due to the use of polymorphic and metamorphic techniques, allowing the same malware to be modified avoiding detection. Although the binary classification of an unknown executable to either malicious or benign (*malware detection*) is important, accurately determining which family the malicious executable belongs to is also a much sought after application [100, 161].

Early malware family classification contributions focused on classifying malware based on the malware binary content or sequence of bytes in binary files. For example, the authors in [54] performed malware family classification using n-grams

of bytecode. However, such content-based approaches require dissembling the binary, which is a time-consuming process and vulnerable to malware obfuscation. Dynamic analysis was applied for malware classification to construct system calls behaviour graphs [55] and control flow graph signatures [162]. Bayer et al. [163] used dynamic analysis to generate malware behavioural profiles used as features in a clustering algorithm to group malware based on behaviour. Rieck et al. [56] applied dynamic analysis to identify the unique behavioural features of malware samples and use these features to build Support Vector Machine (SVM) classifiers that map unknown samples to known malware families.

Similarly, Rieck et al. [57] proposed a framework for malware behavioural analysis, using clustering (to identify novel malware with similar behaviour) and classification (to assign unknown malware to these clusters).

Although these approaches are effective in classifying malware by observing their host-level behaviour, they require access to the binary executable, which might not be possible in specific scenarios (i.e. MSSP SOC and fileless malware).

Network-based Malware Detection— Rule-based approaches [72] rely on defining rules by studying the behaviour of infected hosts in a controlled environment. The behaviour includes communication with the C&C server or the use of obfuscated protocols, which results in defining a rule-based signature of its behaviour. Researchers and anti-virus vendors usually use honeypots to capture malware in the wild and observe their behaviour in a controlled environment. For example, Anubis [73] and SWSandbox [74] provide insight into the dynamic network behaviour of malware. The main objective is to analyse binary samples of malware variants to create rule-based signatures of each malware family for the NIDS to apply.

Observing and analysing network traffic is another method. For example, Xue et al. [75] proposed a malware detection model that detects trojan network communications during three stages: connection establishment, operating control, and connection maintenance, using Bayes classification algorithm. It uses communication flows, thus is capable of detecting unknown or encrypted malware.

Gu et al. proposed *BotHunter* [82] that detects a bot's network activities during its multiple stages of the botnet lifecycle using infection dialogues. It monitors a network's incoming and outgoing communication flows by correlating the evidence trail of flow exchanges that matches a malware infection sequence pattern. The authors model malware communication flows during its infection life cycle as a sequence of events. Using this dialogue model, BotHunter recognizes successful bot infections by correlating the loosely ordered communication events; a technique called dialogue correlation. Unfortunately, this approach focused on

monitoring networks at the perimeter and did not include local host modification and internal network propagation.

BotHunter is only effective on unencrypted network traffic and requires multiple bot infections on the network to detect bot activity [36]. Similarly, *BotMiner* [81] requires multiple bot infections for detection and does not consider active bot propagation through scanning [36]. However, it does improve on *BotHunter* by not requiring unencrypted traffic. Ashfaq et al. [83] extended the infection dialog proposed by *BotHunter* [82] to include passive propagation mechanisms such as spam.

Gu et al. proposed BotSniffer [78], a network anomaly-based botnet detection system, to detect the malware's C&C communications with protocols such as IRC and HTTP. The approach utilizes spatial-temporal correlation and statistical algorithms to detect the synchronisation of hosts' network traffic. The authors hypothesised that botnet C&C communication has a certain pattern that is a result of pre-programmed activities. However, this approach requires multiple infected bots on the same network.

Gu et al. also proposed BotMiner [81], a network anomaly detection system that exploits the similarity of botnets C&C communications and malicious network patterns. Tegeler et al. proposed BotFinder [84], a bot detection system that monitors network traffic for C&C communication to detect bot-infected hosts. It builds a model of C&C network patterns of the different malware families; thus, traffic that matches the model is flagged as malicious. Compared to BotSniffer [78] and BotMiner [81], the approach detects individually infected hosts and therefore does not need to correlate network activities of multiple hosts. Moreover, it does not rely on payload information, but high-level network data such as NetFlow in the detection, making it applicable to encrypted botnet traffic. Abaid et al. [85] analyzed botnet network behaviours and identified those that are synchronised across multiple bots. However, they only focused on the detection of spam and port scanning.

C&C Detection— Since malware is normally distributed, managed, and coordinated through remote servers, research efforts have focused on monitoring, detecting, and blocking these malicious domains. Such malware communicates to C&C servers to receive commands using protocols such as IRC, or more commonly HTTP since web traffic is not usually blocked in most organisations.

For example, Botzilla [77] monitored malware network traffic, specifically a bot's communication to the C&C server, in a sandboxed environment to find patterns and generate network signatures. Bilge et al. proposed Disclosure [79] that extracts flow size-based features, client access patterns, and temporal behaviour features

from Netflow communication to detect botnet C&C communications. *ProVeX* [80] is a system that focused on the detection of C&C encrypted communications by focusing on network messages and extracting the C&C protocol semantics.

Other contributions such as [86–89] applied active probing to identify malware's underlying malicious C&C servers. Active probing is based on network fingerprinting these malicious servers by sending crafted packets to remote servers and determine if they are malicious, based on their response. For example, CYBERPROBE [87] utilises machine learning to produce network fingerprints from the network traces of malware families. However, this approach does not produce fingerprints for malware with replay protection or where C&C servers are not live [88]. Most malware uses a pull-based C&C protocol, where malware contacts the C&C server for further instructions instead of C&C servers' incoming probes being blocked by NAT gateways and firewalls.

AutoProbe [88] supports pull-based C&C protocols, extending CYBERPROBE [87] by leveraging dynamic malware analysis and active probing to generate fingerprints of malicious C&C servers. AutoProbe addresses some limitations of CYBERPROBE as it makes fingerprints even when C&C servers are not live during fingerprint generation. Although active probing has been efficient in detecting malicious domains, it requires a probe of the entire Internet, including benign hosts, thus disturbing legitimate hosts.

DNS Detection— A number of approaches for detecting malicious domains through DNS analysis have been attempted. EXPOSURE [90] uses a DNS analysis approach to detect malicious domains used by malware. It utilized 15 features to differentiate DNS names, how they are constructed, and how they are queried. NOTOS [91], a domain name reputation system assigns reputation scores to domain names whose maliciousness is yet to be determined.

Malware resort to evasion techniques by rapidly changing their domain names and the associated IP addresses (IP-fluxing and domain-fluxing [92]) to avoid detection by blacklists and IDS and increase resilience. For example, Hu et al. [93] analyses NetFlow traffic to detect redirection flux utilized by a type of malware (redirection botnets). However, the approach is not able to identify flux agents being used as transparent proxies, instead of redirection points [94]. Perdisci et al. [94] monitors DNS traffic in a network to capture queries to flux domain names. It takes a passive approach and does not interact with the malicious domain through, for example, active probing.

Similarly, Stalmans et al. [95] propose an approach deployed at the network edge to detect fast-flux queries. The active probing method was also applied to detect

malicious domains through analysing DNS Queries. The approach repeatedly issues DNS queries to a set of possible malicious domains and collects information about the resolved IP addresses, to classify each domain name into either fast-flux or non-fast-flux [96–98]. However, active probing may be detected by the adversary, which then avoids responding to probing queries to prevent unveiling further information.

Manadhata et al. [99] modelled the malicious domain detection problem as an inference problem on large host-domain graphs. A host-domain graph is constructed from the proxy log and a seed of minimal ground truth; they then apply belief propagation to estimate the marginal probability that the domain is malicious. Similarly, this approach could be applied to other logs, such as DNS queries.

Malware Family Classification— Rossow et al.[76] proposed a dynamic malware analysis environment by observing the network behaviour of 100,000 malware samples over long periods of time. The authors aimed to tackle the limitations of previously proposed malicious network traffic tools [73, 74], such as the short analysis period of a few minutes and the lack of detailed network behaviour analysis. They also provide insights on DNS and HTTP malware traffic trends. Rafique et al. [17] investigated the application of several machine learning algorithms for malware classification. They propose a framework to classify malware based on different network protocols (e.g. HTTP).

Malware behavioural clustering approaches aim to group malware behaviour to reveal similarities between malware samples that may not be captured using system-level malware analysis. Perdisci et al. in [103], proposed a network-level clustering system to derive similarities in HTTP-based malware behaviour for detection. FIRMA [104] used a similar approach to cluster and generate network signatures for malware network traffic. Previous research also extracted malware behavioural features as a sequence of values. For example, Santos et al. [105] used n-grams, which are sub-strings of a larger string with length n, to generate file signatures for malware detection.

Wressnegger et al. [102] studied the applicability of n-grams for anomaly detection and classification. Mohaisen et al. [100] observed the high-level network features of malware and applied n-gram document analysis for family classification. Similarly, Mekky et al. [101] used n-grams to encode the order of sub-sequences of malware network communication events to build malware classification models. The authors applied a similar approach to [100] by first isolating malware traffic from the benign traffic using Independent Component Analysis (ICA). However, their approach differs from [100] as they use coarse-grained groups of network events so that inbound, DNS, and port number are considered one network event. Similar to [100], they use a count-based approach for network flow identification.

3.4 Cyber Data Fusion

Early IDS were stand-alone devices that monitored the network for local attacks. However, as networks grew, IDS no longer scaled and a need for IDS that communicate with each other to detect more sophisticated and coordinated attacks that occur in different places at the same time. Collaborative IDSs (CIDSs) emerged, which contain several monitoring components that collect and exchange data [106]. The CIDS monitors produce alerts that are aggregated before performing analysis, thus providing a more holistic view of the network security status.

To protect their networks from adversaries, organisations deploy heterogeneous security devices such as IDS, firewalls, packet sniffers, SNMP traps, and honeypots. These tools generate events in logs resulting in billions of logs each day. Several studies proposed systems that correlate these log events to detect security incidents.

Data fusion in IDS, first proposed by [107], is a technique to aggregate data from these various heterogeneous sources with different data structures for better detection of intrusions. The author discussed the use of data fusion in near real-time, combining it with offline data mining of the resulting big data for more effective intrusion detection. Similarly, the authors in [108] utilized data fusion to aggregate data from heterogeneous sources using a data fusion technique called Dempster-Shafer (D-S) evidence theory to get a better sense of the data. The most-reported challenge in these studies is big data – an aggregation of billions of log alerts presented in heterogeneous data structures. Zuech et al. [109] present a good survey on this issue discussing the big data challenges and heterogeneous data aggregation contributions such as [110] and [111].

As Hall et al. [112] discussed, cyber fusion is a double-edged sword, as the performance of a data fusion system may be worse than the individual monitoring sensors. Therefore, cyber fusion systems need to be properly designed to avoid what the author described as “catastrophic fusion”.

Researchers have investigated how organisations could better deal with the big data challenge of their log data. Yen et al. [113] proposed Beehive that automatically performs “large-scale” analysis of billions of device logs of heterogeneous data structures for better detection of malicious activities in an organisation. It correlates log information to detect malicious activities that are a result of policy violations or malware and reports them as incidents. The incident response team then further analyzes these incidents to determine if they are malicious or benign. The system was deployed in a large enterprise EMC for two weeks, where 1.4 billion logs were generated on average per day. The main issue faced when implementing the system

was "Big Data", where the organisation devices generate billions of log alerts per average a day stored in a SIEM, and "efficient data-reduction algorithms and techniques" are required. Another challenge was "big velocity" as the organisation deploys various heterogeneous devices with a wide verity of data structures, making correlation of these events difficult. Moreover, they described some data as dirty, as devices may generate logs that are either incomplete or inconsistent (different time-stamps), making analysis an even more significant challenge. However, Beehive was quite successful in detecting 784 security incidents compared to the eight detected by the organisation's existing security system, reducing the alerts inspected by 74% by removing the white-listed target hosts. Hence, data log aggregation is found to be a promising approach for effective intrusion detection.

Bocchin et al. [114] used network connectivity graphs to extract host events relevant to a seed event to detect malicious network activities. The approach passively monitors network traffic at the edge and extracts and logs events, such as HTTP requests, DNS queries, and response or use of unknown protocols. When an intrusion detection system flags an event, this event is the seed that triggers the proposed system to correlate the logs correlated with the seed event. It then models network activities of hosts over time (different network traffic samples from the same host), space (connecting similar patterns across different hosts), and network layers (HTTP, DNS, etc.). The approach provides analysts with more visibility to what network traffic is associated with the IDS alert, thus more understanding of the underlying activities.

APT Detection— Oprea et al. [115] propose a graph-based approach based on belief propagation to detect malicious domains utilized by malware in the early stages of an Advanced Persistent Threats (APT). The method was evaluated using a large dataset of web proxy logs and utilizes supervised and unsupervised machine learning algorithms. Given a seed of a known malicious host or domain, the approach infers other possible domains or compromised hosts that belong to the same criminal network.

Milajerdi et al. proposed Holmes [164], a real-time APT detection system that correlates tactics, techniques, and procedures that might be used to carry out each APT stage. HOLMES generates a high-level graph that summarizes the attacker's steps in real-time. Pei et al. proposed Hercule [165], a multi-stage log-based APT attack detection. Inspired by graph analysis approaches, it builds a graph by correlating log entries across multiple logs, then discovers attacks through community detection algorithms. Similarly, Peng et al. [166] used a graph-based approach for malicious domain detection.

Milajerdi et al. proposed ProPatrol [167], an APT attack provenance graph, useful for APT forensic purposes. The approach utilizes malware communication structure — for example, bots in an enterprise contact the same C&C server. Liu et al. [168] propose an APT and insider threat detection approach. The proposed approach converts log entries into a heterogeneous graph, proposing a graph embedding to represent each log entry into a low-dimension vector. They also propose a detection algorithm capable of separating malicious and benign log entries into different clusters and identifying malicious ones.

Automation in SOCs — Recently, there has been an increased focus from the research community on developing security tools to automate some of the operations in SOCs such as data triage [117, 118], log aggregation [119], log mining [113], SIEM alert filtering [120], and threat hunting [121]. Similarly, commercial products such as “Security Automation and Orchestration (SOAR) platforms" [169, 170] provide solutions for automatically detecting, preventing, and recovering from cyber-attacks.

Najafi et al. [171] presented threat detection in SIEM environments as a large-scale graph inference problem. The knowledge graph models entities observed in proxy and DNS logs, enriched with related Open Source Intelligence (OSINT) and Cyber Threat Intelligence (CTI). In addition, they propose a graph-based inference algorithm designed to infer a node maliciousness score based on its associations to other entities presented in the knowledge graph,e.g. shared IP ranges or name servers.

Chau et al. [172] modelled malware detection as a large-scale graph mining and inference task where nodes represent users and files, with edges between them inferring a file present on a users’s machine. To identify malware, the proposed method locates files with low reputation using an adapted version of the Belief Propagation algorithm. Yen et al. proposed TAMD [116] to detect malware by aggregating traffic that shares the same characteristics, such as external destination, payload, and OS platforms.

3.5 Security Operation Centres

Probably the most notable research in understanding security practitioners’ role in SOCs is work arising from the HOT Admin Research project [7, 122, 124, 125, 127], as well as research by Sundaramurthy et al. [10, 11, 22].

The HOT Admin project was aimed at understanding the human, organisational, and technological factors (HOT) that impact security practitioners in Information Technology Security Management (ITSM) in their daily tasks.

Botta et al. [7] aim to provide a better understanding of how practitioners of Information Technology (IT) security skills and the use of their tools. They used a questionnaire and conducted semi-structured interviews with 14 practitioners, analyzing the data using open-coding and pre-designed themes in qualitative methods. Their results suggest that the workplace can be categorized based on the responsibilities of IT professionals. Specifically, the activities of professionals, goals of the activities, tasks they perform to achieve goals and skills needed. In addition, they discussed how the professionals feel about the tools they use in terms of hardware reliability, accessibility, integrability into work practices, and the ability to obtain training, consulting, and help with problems of usage in combination with existing systems.

Werlinger et al. [125] conducted interviews to research challenges that IT security practitioners faced when installing, configuring, and maintaining IDS, providing recommendations to address them. Jaferian et al. [173], identified guidelines and recommendations for designing ITSM tools from the literature as well as from prior studies from the HOT project. Werlinger et al. [123] studied the challenges that IT security practitioners face in across 17 organisations (academic, government, and private), particularly among human, operational, and technological factors. They identified 18 challenges that can affect IT security management within an organisation, building an integrated framework of security challenges. Werlinger et al. [124] then performed qualitative study activities that required interactions between IT security practitioners and other stakeholders. The research found that tools used by security practitioners to perform their security tasks provide insufficient support to the complex interaction required in their job. The authors then offer recommendations for addressing this complexity and improving IT security tools.

They extend this work in [122] by identifying tasks, skills, strategies, and tools that security practitioners use to diagnose security incidents, identifying opportunities for future research directions related to improving security tools. Botta et al. [126] analyzed the data using Busby's (2001) characterization of distributed cognition, focusing on cues and norms, and how ITSM challenges undermine their effective deployment.

Research by Sundaramurthy et al. [10, 11, 22, 128] takes an anthropological approach studying three SOCS and making several observations regarding the people, process, and technology. Specifically, in [22], they identified observations related to operational tools/teams, workflow, and how teams come together in solving security incidents. They identified factors that lead to analysts' burnout in a SOC, providing a model that explains the burnout phenomenon [11]. Using

Active Theory Model, they identified and list contradictions in SOCs that manifest as conflicts. They present a Pentagon model for improving the SOC operations by identifying tasks that can be automated and resolve conflicts [128]. Kokulu et al. [12] applied a qualitative approach to identify technological, human, and operational issues in SOCs across different sectors. The authors discussed limited issues arising from technology (e.g. malfunctioning of SOC tools). One of the most interesting findings was that analysts don't perceive false positives to be a major issue, despite academia's belief.

Akinrolabu et al. [174] found that current IDS are inadequate in detecting multi-stage attacks stealthy attacks. Although existing ML detection approaches exists, the authors discussed how they rely on statistical features derived from training sets limiting the models ability to detect new malware variants. The authors conducted interviews with analysts, eliciting 17 behavioural-based network features, capable of detecting novel attacks.

Paul et al. [175] studied how cybersecurity analysts establish and maintain Situation Awareness (SA) by conducting interviews, observations, and a card sorting activity. They identified questions analysts ask themselves when faced with a networking event, developing a taxonomy of the questions for event detection and orientation. D'Amico et al. [130] applied Cognitive Task Analysis (CTA) to understand the work processes, cognitive skills, and tools that analysts rely on to achieve SA. They also identified the cognitive challenges and obstacles that impede SA. Haney et al. [176] investigated the methods, tools, and challenges of Red and Blue teams, identifying examples of successful integration and opportunities to enhance SA. They also discuss design implications for tools that can facilitate SA among multiple cyber-defense teams by supporting data fusion, change detection, network mapping, and access tracking.

M'manga et al. [129] used Distributed Cognition and Grounded Theory, a qualitative approach, to understand how security analysts make decisions about risk. D'Amico et al. [131] studied the analysts' analytic goals, tasks, and the processes they follow, the decisions they make, and the data sources and tools used to make those decisions and challenges they face.

3.6 Summary

In this chapter, we reviewed the existing related literature identifying their limitations and the research gaps. Although previous work that considered malware network communication patterns exists [81–85], as identified by [36] these either

require unencrypted network traffic (e.g. [82]), require multiple malware (i.e. bot) infections on network (e.g. [78, 81, 82]), require active propagation through scanning (e.g. [81]), or don't consider local malware (i.e. bots) attacking local targets (e.g. [81]). Importantly, most previous work on malware detection using machine learning failed to evaluate the system in detecting unseen or new malware. This is crucial in determining the systems' performance over time in detecting new malware families.

Our review of existing literature studying SOCs revealed a shortage of systematic and comprehensive research in describing the SOC landscape and the technological and operational challenges therein. For example, research by Sundaramurthy et al. [10, 11, 22, 128] applied ethnography research methods to study educational SOCs, focusing only on their operational and personnel difficulties. Hence, the strengths and weaknesses of the deployed technological solutions were not discussed. To describe how these research gaps were addressed, we continue in the following chapter by outlining our methodology.

4
Methodology

Contents

4.1 Introduction

We discuss in this chapter the research methods applied to investigate the research aims identified in Chapter 1: (1) the role of SOC dimensions (people, process, technology) on malware monitoring, detection and response; (2) designing novel malware classification and detection systems that improves over existing state-of-the-art tools.

For the first aim, we applied quantitative and qualitative research methods reporting the findings in Chapters 5, 6, and 7. Ethical approval for this study

Table 4.1: Survey Participants —*(Expertise Level: Very High(H+), High(H), Medium(M), Low(L)), (Organisation Size: Large(L), Medium(M), Small(S))*

ID	Years of Exp.	Job Title	Expertise Level	Org. type	No. analysts in SOC	Org. Size
I1	0 - 3	Analyst	H	Security	1 - 9	S
I2	0 - 3	Analyst	L	Security	1 - 9	L
I3	-	Architect	H	Security	20 - 29	M
I4	0 - 3	Analyst	L	Security	20 - 29	M
I5	3 - 5	Manager	H	Security	10 - 19	L
I6	5 - 7	Engineer	H	Other	10 - 19	L
I7	7 - 10	Engineer	H+	Government	10 - 19	L
I8	10 - 15	Engineer	H	Government	10 - 19	L
I9	0 - 3	Analyst	L	Government	1 - 9	L
I10	0 - 3	Architect	H	Security	-	M
I11	0 - 3	Analyst	M	Security	10 - 19	L
I12	0 - 3	Analyst	M	Other	1 - 9	L
I13	3 - 5	Analyst	H	Security	1 - 9	L
I14	3 - 5	IR Manager	H	Security	200 +	L
I15	0 - 3	Manager	H	Security	200 +	L
I16	3 - 5	Analyst	M	Security	-	L
I17	0 - 3	Analyst	L	Security	1 - 9	L
I18	3 - 5	Analyst	M	Security	1 - 9	M
I19	0 - 3	Analyst	M	Security	10 - 19	S
I20	3 - 5	Analyst	H	Security	1 - 9	M

was granted by the Central University Research Ethics Committee, University of Oxford (Reference: R48822/RE001).

To address the second research goal, we considered the limitations of existing malware detection tools extracted in the literature review, and the requirements identified in the qualitative research. These findings motivated the design of malware classification and detection systems, evaluated through experimental methods reported in Chapters 8 and 9. As similar research methods were applied in these chapters, we describe them herein, complementing the methodology sections included in their relevant chapters.

4.2 Participant Recruitment and Demographics

We describe our recruitment of participants for both the quantitative and the qualitative parts of the study. Both studies targeted security practitioners who work in SOCs such as security analysts, engineers and managers. The nature of SOC work means that it often requires a high level of focus and prolonged attention to screens under considerable time pressure [22]. As such, recruiting SOC practitioners to participate in studies can be challenging: participation can be a burden on analysts, for example, taking time away from investigating tickets (which poses a particular challenge since some SOCs evaluate analysts based on the number of tickets processed and solved [11]). In addition, gaining the trust of security practitioners as study participants is essential to ensuring the validity of the data collected [11].

Table 4.2: Interview Participants —*(Expertise level: High(H), Medium(M), Low(L)), (Organisation Size: Large(L), Medium(M), Small(S))*

ID	Job Title	Expertise	SOC Type	Org. Size	Customer
P1	SOC Manager	-	In-house	M	Other
P2	Lead Analyst	H	MSSP	M	Other
P3	Engineer	-	In-house	M	Other
P4	Engineer	H	MSSP	M	Other
P5	Lead Analyst	H	MSSP	M	Other
P6	Engineer	-	MSSP	L	Government
P7	Analyst	M	In-house	L	Government
P8	SIEM Engineer	-	In-house	M	Other
P9	Manager	H	MSSP	M	Other
P10	Cybersecurity Monitoring Analyst	M	MSSP	M	Other
P11	Unix, UTM coding	L	In-house	M	Other
P12	Engineer	M	In-house	M	Other
P13	Engineer, L3 analyst	H	MSSP	M	Other
P14	Analyst	M	Internal	M	Other
P15	Incident Responder	H	MSSP	L	Other
P16	Engineer, L3 analyst	H	In-house	L	Government
P17	Lead Analyst	H	MSSP	M	Other
P18	Analyst	-	MSSP	L	Government
P19	CISO	H	In-house	L	Government
P20	SOC Manager	H	In-house	L	Government
P21	Analyst	L	In-house	L	Government

In order to overcome these issues, we leveraged the institutional relationships in our recruitment process. Leveraging such connections helped in establishing the level of trust needed between the researchers and the participating organisations. Thus, as the first stage in our study recruitment we contacted senior-level security professionals in organisations that have a SOC. Due to the aforementioned challenges in recruitment, our sampling was not random. We found that obtaining senior management engagement and access to participants was more important than a random sample which might have reduced bias in the results.

For the quantitative study, we asked these professionals to forward the online survey to security practitioners within their SOC. Some organisations required an opportunity to review the survey questions before forwarding them to their practitioners. Surveys were distributed in March 2017. For the qualitative study, we liaised with these senior-level professionals to arrange interview dates with a convenient sample of their security practitioners. Interviews were conducted between April 2017 and June 2017. Compensation for participation was not provided. Gaining the acceptance from senior-level security practitioners within the organisation helped in establishing a trust with the security analysts and encouraged their participation, despite their busy work day. While there was likely some overlap between participants in the quantitative and the qualitative studies, our anonymisation of the online-survey results prevents us from determining the extent of this.

Our participants work across seven different SOCs from organisations of various sizes serving government and non-government organisations. We show the demographics of our survey participants in Table 4.1 and interview participants in Table 4.2. In survey responses, participants provided their demographics at the beginning of the questionnaire. Hence, demographics such as *Expertise Level* indicated in Table 4.1 were based on the participants self-evaluation. On the other hand, the demographics of interview participants and their organisations were extracted from the participant's answer to the interview question: "*Please describe your position/job role/level*". Accordingly, the participants' expertise reported in Table 4.2 is based on the analyst level and number of years in the position. Hence, level-3 analysts or lead analysts are scored as high on expertise compared to level-1 analysts or those who are new to the job, who are scored low.

4.3 Quantitative Method: Online Questionnaire

The design of our quantitative method encompassed the development of a questionnaire, participant recruitment, and the analysis of the results.

4.3.1 Data Collection: Online Questionnaire

Our review of the literature revealed that SOCs have various organisational structures, each with its own challenges. The purpose of the survey was to gather information about working practice in our participating SOCs, and we anticipated that these initial findings would guide further data collection in the qualitative study. The full set of survey questions is provided in Appendix D. These are the questions relating to SOC working processes; more specifically, those that focused on identifying the SOC workflow and the security-monitoring approaches applied by security analysts in SOCs (including their use of security tools and attack indicators).

We drew on existing literature to design our online survey questions. We identified theories requiring validation and constructed questions on these aspects. Example of such literature includes work by Garcia et al. [138] and Goodall et al. [177] that explored challenges in intrusion detection systems such as the prevalence of false-positives. Theories identified from such literature were asked in an online survey of 20 security practitioners, in which participants were asked to indicate their level of agreement with such assertions.

We used an online survey to enable remote participation, improving our reach by enabling security practitioners located in other countries to participate, for example. The survey took approximately 15-20 minutes to complete. Prior to

starting the survey, participants were presented with information about the research goals, handling their data and the researchers' contact information. Participants then gave their consent for their participation. We offered participants the option to comment on the questions and sections in the survey by providing free-text spaces. In designing the survey questions, we incorporated feedback from subject-matter experts (a security analyst, senior security professional and a SOC manager).

In our survey, we presented the analysts with a set of assertions on the human-in-the-loop level in SOCs and the importance of maintaining awareness of the network. Participants were asked to indicate their level of agreement with each assertion using a Likert-Scale (1: "*Strongly Disagree*", 2: "*Disagree*", 3: "*Neutral*", 4: "*Agree*", 5: "*Strongly Agree*"). There is a discrepancy in the community as to how to handle Likert scale data [178]. Therefore, we calculated both the mode and median to analyse responses to the assertions. Mode or median values higher than 3 constituted overall agreement with an assertion as 3 was the neutral value. We also calculated comparison of non-neutral scores (CNNS), which represents the ratio of scores less than (1,2) and greater than (4,5) the neutral value (3). We present the full list of assertions in Appendix (Tables B.1 and B.2).

4.4 Qualitative Method: Semi-structured Interviews

The design of our qualitative method encompassed designing and conducting semi-structured interviews and the analysis of the results. We also recognise the limitations in applying qualitative research methods, detailed in Section 4.4.3.

4.4.1 Data Collection: Semi-Structured Interviews

Our design of the qualitative research questions was influenced by our findings from the quantitative online survey. We chose to conduct semi-structured interviews, intending to extend discussion based on the flow of conversation. The interview questions were structured as follows:

Interview Part 1— The first part of the interview was designed to encourage the practitioners to talk about their daily tasks, tools they used, and the challenges they face relating to tools and processes. In addition, we capture the security practitioners' perspectives on the importance of the human in the loop in SOC operations and the potential of automation. We present the interview questions in Appendix A.

Interview Part 2— To understand the practitioners' thought process when dealing with an incident, we used scenario-based questions. The practitioners were presented with a specific threat scenario and asked to talk through the tools they will use and steps they follow to address the problem. For example, we asked analysts to explain how they will go about investigating huge volumes of outgoing traffic from a server to an external IP address. This helped elicit the influencing factors on SOC operations that derive the practitioners' actions and decisions. We present the full description of the scenarios in Appendix A.

As with the online-survey questions, interview questions incorporated feedback from three subject-matter experts who worked, or had previously worked, in SOCs to ensure face validity [179]. We also ran a pilot interview with a security analyst, gathering feedback on the questions to ensure their clarity and suitability for the target audience.

The majority of the interviews were carried out face-to-face with participants at the organisations at which they worked, in rooms exterior to the SOC. Due to travel constraints, two participants were interviewed through a live video chat. The interviews were audio-recorded and lasted approximately one hour each; however, due to the nature of the job, one interview was interrupted multiple times for the analyst to deal with an incoming incident and therefore lasted longer. In addition, one organisation opted to have researchers conduct the interview with multiple security practitioners at once.

We started the interviews with a brief introduction of ourselves and the study objectives. Participants were presented with a participants information sheet detailing the objectives of the study, the handling and anonymisation of the data, and processes to withdraw from the study and the researchers contact information. Participants provided their consent by signing the consent form. The participants were allowed to withdraw from the study at any point, in which case their data would be deleted.

The semi-structured interview then took place. The interviews were conducted by two researchers in order to: ❶ enable consistency of the data collection process, ❷ mitigate the risk that an interviewer would bias participants by asking leading questions, ❸ obtain multiple researcher perspectives enabling peer reflection at a later date, and ❹ ensure that all questions were covered. However, not all questions were discussed in-depth, depending on the participants' position in the SOC.

Table 4.3: Apriori Themes defined and used in Template Analysis.

	Theme	Description
1	Human-centered Tasks	SOC tasks that rely on analysts (humans) intelligence and cognitive abilities for monitoring, detection and response.
2	Technology-focused Tasks	SOC functionalities that rely on technology/tools for monitoring, detection and response.
3	Factors influencing SOC Process.	Factors that influence SOC processes (e.g. type of SOC or type of client).
4	Importance of Knowing your Network	Tasks where analysts reported their reliance on knowledge of the monitored network and environment to analyse and incident and make decisions.
5	Network Traffic Collection	Procedures in place and granularity levels of the collected and stored network traffic.
6	Strengths of Network Security Monitoring Tools	Strengths analysts perceive of existing network security tools.
7	Weaknesses of Network Security Monitoring Tools	Weaknesses analysts perceive of existing network security tools.
8	Understanding SOC Workflow	Processes SOC analysts follow to prepare for, detect, investigate and respond to a security incident.
9	Automation and Machine Learning	Usage of machine learning technologies and opportunities for automation of tasks.
10	Internal Network Security Monitoring	Challenges in the monitoring of internal network traffic for security incidents.

4.4.2 Data Analysis

The audio-recorded interviews were transcribed, resulting in textual data of 105,523 words for analysis and interpretation. We ensured ethical handling of the data by preserving the anonymity of the participants and their organisations. All transcripts were anonymised prior to analysis and stored with appropriate security protections.

We applied Template Analysis [180], starting with an apriori set of themes we were interested in, and allowing the code to evolve with the addition of new arising themes. Themes are a group of repeating ideas that enable researchers to answer the study question [181]. Examples of themes identified in our study are shown in Table 4.3. This analysis approach that is both inductive and deductive. It is deductive as we built a hypothesis and developed theories by reading the relevant literature and conducting the interviews, and inductive as we also identified patterns in the data that build on the preconceived theories. Therefore, this analysis approach was chosen over grounded theory due to themes in Template Analysis being less prescriptive, providing the ability to identify and add new concepts if discovered while allowing us to have preconceived theories [180]. Moreover, recent closely related work have applied Template Analysis [128] to analyse qualitative

data of their study on security operation centres. As studies on security operation centres are still sparse, Template Analysis is useful due to our partial understanding of the concepts need to be identified in the data [128]. We followed the Template Analysis steps defined as follows:

- **Define the Apriori Themes**: We started with defining ten themes we were interested in after reviewing the literature and the results of our survey to guide our analysis, shown in Table 4.3.
- **Familiarize with Data**: Before we started with initial encoding, we read through the interview transcriptions.
- **Initial Coding:** We started by taking (5) interviews and identifying parts in the transcriptions that are relevant to the themes identified, assigning apriori theme code the them. When an interesting part is encountered that did not have a matching theme code, a new theme code is created or an existing themes is broadened. We coded the data using NVIVO [1], a well-known software for qualitative data analysis. One of the useful features provided by NVIVO is in highlighting the distribution of the codes within and across the template. This can help to draw attention to the aspects of data that require further investigation [180] and can also be used as a means of making connections within the text.
- **Produce Initial Template**: We use the codes that arise from the subset of the data to produce the initial template. The initial template may include further sub-codes, thus producing a template with a hierarchy of codes within each theme. We show a snapshot of the initial template in Appendix C.
- **Develop the Template:** This is an iterative process, where the initial template is applied o the rest of the interviews, modifying the codes as necessary until a final template is generated.
- **Interpretation**: Using the final template produced by coding all the interview data, we interpreted the data and wrote our findings.
- **Consistency and Quality:** Researchers consulted other research team members of the codes and findings, engaging in frequent reflections to maintain the quality of the analysis and make sure personal beliefs, the familiarity of the data, and biases do not affect our interpretation of the data.

[1] `https://www.qsrinternational.com/nvivo/nvivo-products`

4.4.3 Qualitative Research Limitations

As the method applied is qualitative; our sample size is small and does not allow us to draw quantitative, generalisable conclusions. An important consideration is that each SOC is very different, in their size, structure and analysts' skills. Some SOCs have an analyst "playbook", which defines how the analyst should analyse a potential threat. Additionally, our participants are based primarily in the UK, some working in SOCs serving UK-based public organisations. Therefore, an overall generalisation may not be possible because of these differences. We may, however, be able to identify trends that apply across all SOCs observed and use these to form the basis of theories.

Due to the decision to use semi-structured interviews as the approach to the qualitative study, the level of detail in which different participants discussed each question varied. Self-reported data also have limitations such as recall and observer bias. Furthermore, we report only those themes highlighted by the participants in this study. It is possible that other themes may emerge in conversation with other participants, and as such similar studies with further participants would be valuable in validating the findings of this study.

4.5 Experimental Design

The design of our experimental method encompassed understanding the weaknesses of existing malware detection and classification systems; extracting malware detection and classification system design goals; building a prototype of the system; and conducting experiments to evaluate the performance of the various system modules. We discuss in the following sections the ML algorithms used to build the system detection models and the datasets and metrics used for system evaluation.

4.5.1 Dataset

We used three datasets in evaluating our proposed systems.

ISCX Botnet Dataset— This dataset contains malicious and benign traffic and is split into training and testing sets with 5 and 13 bots respectively. The dataset contains traffic from 5283 and 9896 benign hosts for training and testing respectively with 27 bot-infected hosts. The benign traffic in the dataset contains activities such as SSH, HTTP, Web browsing, World of Warcraft gaming, bittorrent clients such as Azureus, web and email traffic and backup and streaming media, and SMTP mimicking users' behaviour. We refer the reader to [182] for a detailed description of the dataset.

Stratosphere IPS Project— This datatset provided by *The Stratosphere Research Laboratory* [2] contains over 300 malware samples of current malware that was first seen in the wild between years 2013-2018 such as *WannaCry*, *Emotet*, and *BitCoinMiner*.

Malware Capture Facility Project (MCFP), CTU-13 Dataset— The CTU-13 dataset contains 13 scenarios of botnet network traffic. The botnet families represented in the dataset employed various protocols (*e.g. IRC, P2P, HTTP* and various techniques (*e.g. sending SPAM, DDoS attacks, Fast-Fluxing*). Our main motivation for using this dataset is that real botnet attacks network traces were captured, providing reliable datasets for model building.

In addition to running all samples on the same network environment, precautions were taken to ensure that the full bot network flows were captured and outgoing traffic was not filtered or rate-limited. This ensures that the data is clear from artefacts that can affect the classifier performance, due to how the collection environment was set up. For more information about the nature of the dataset scenarios and how it was generated see [183].

4.5.2 Classifiers Design

We applied supervised ML algorithms, meaning they learn a function that maps an input to an output using labelled data. In our initial experiments, we evaluated the performance of four supervised machine learning algorithms: k-Nearest Neighbour (KNN), Random Forest (RF) [184], Neural Networks (NN) [185] and Support Vector Machines (SVM) [186]. Both k-Nearest Neighbour and Random Forest (RF) performed best in our initial results in terms of classifier performance and speed of classifier training. Hence, we only applied both algorithms in our system design which we describe in the following.

KNN– KNN is a non parametric lazy learning algorithm, meaning, it does not make assumptions about the data distribution. It can be applied for both classification and regression problems. We apply KNN to a classification problem, where the output would be a class of which the input belongs to, predicted by the majority vote of its k closest neighbours. It simply assumes that the classification of a sample is similar to other samples that are nearby in the vector space. Various distance measures can be applied to find the closest neighbours. We apply the most used method, Euclidean Distance [187].

One of the challenges in KNN is defining the value of k. The value of k plays a critical role in the accuracy of the classification. Smaller values lead to a low

[2] `https://stratosphereips.org/`

accuracy, especially when there is noise in the data, whilst larger values slows the classifier performance and can lead to overfitting. Overfitting is a modeling error that occurs when the algorithm function is too closely or exact fit to a particular set of data points, therefore, failing to classify additional data reliably.

Random Forest— Random Forest is an ensemble classifier that leverages multiple decision trees to produce a better prediction accuracy. It aggregates the decision trees individual predictions and combine into a final prediction, based on a majority voting of the individual predictions. The trees are trained using different subsets of the training set, thus overcoming over-fitting issues of an individual decision tree. However, random forests is an ensemble model. Hence, its predictions are inherently less interpretable than individual decision trees.

4.5.3 Evaluation Metrics

For evaluating the performance of the classifiers, the following evaluation metrics were used:

1. **Precision**— Precision is the number of True Positives (TP) divided by the number of True Positives (TP) and False Positives (FP), ($Precision = TP/(TP + FP)$). A low precision can indicate a large number of False Positives (FP).

2. **Recall**— Recall is the number of True Positives (TP) divided by the number of True Positives (TP) and the number of False Negatives (FN), ($Recall = TP/(TP + FN)$). A low recall indicates many False Negatives. Therefore, we seek an analysis setup that maximizes the precision and recall.

3. **F-measure (i.e. F1 score)**— F-measure is an aggregated performance score, that combines both precision and recall ($F1 = 2TP/(2TP + FP + FN)$). A perfect discovery of classes yields $F1 = 1$, where either a low precision or recall result in a low F-measure.

4.6 Summary

In this chapter, we discussed the methodology followed to carryout the research goals identified in Chapter 1. We applied quantitative methods in form of an online survey to gain an understanding of the cyber threats our participants face and the monitoring and detection tools they deploy. Qualitative method is then applied through the use of semi-structured interviews that allow us to gain a

rich understanding of the inner workings of the SOC, their tool use and the challenges therein. We also discussed the experimental design used to develop malware detection and classification systems, in designing the ML classifiers, and the datasets and metrics used for evaluation.

As these research methods are used consistently throughout the thesis, this chapter provides a reference to these methods, complementing additional methodology details provided in each chapter. In the next chapter, we present the findings of our online survey. These findings highlighted interesting research questions about the participating SOCs that we followup on using semi-structured interviews.

5

SOC Analysts' Perspective on Network Monitoring Tools

Contents

5.1 Introduction

We commence our study on understanding malware monitoring and detection in SOCs by conducting quantitative research through an online questionnaire. The goal of the survey was to understanding the operational landscape that SOC analysts are working in well enough to develop research questions. SOCs are diverse with distinctive setups and goals; thus, the people, technology, and processes will be unique as well. Hence, we followed an inductive approach and used a quantitative study as a starting point for our qualitative research. The results of the survey helped us to design our qualitative study, which we discuss in Chapters 6 and 7.

In the following, we explore the results provided by our survey: how security practitioners view tool detection capabilities, which tools they use, network features

and indicators of compromise valuable in the threat detection; and the practitioners agreement to assertions.

5.2 Security Tools Detection Capability

We asked our survey participants what security incidents their organisations face and whether they believe that the tools they have now are good at detecting/addressing these threats. Figure 5.1 shows the participant's responses based on their organisation size while Figure 5.2 shows responses based on type of client (government and non-government). The percentage annotated on the bar represents the percentage of participants per category, while the length of the bar represents the percentage in overall participants regardless of category.

Threats— We wanted to understand the top threats that our participating organisations defend against. We asked: *What are the main network security threats your organisation faces?* It is essential to highlight that these are the most frequent threats that the organisation faces. Thus, a low score for a threat does not indicate that the organisation does not mean the organisation is not susceptible to that threat; it just suggests that it is not the most frequent.

Most participants (85%) agreed that most threats they face are abnormal user activity, 80% Advanced Persistent Threats (APT), and 75% viruses, worms, trojans, and phishing emails. Interestingly, only 60% reported that ransomware, an emerging threat, was a threat their organisation faces. Specifically, only 33% of participants from government organisations think ransomware is a threat. However, as this study was conducted in 2017, this perception may have changed since then, especially with the rise of ransomware in following years. On the other hand, analysts from government SOCs all agreed that the main threats they face are APTs, abnormal network activity, abnormal user activity, and unauthorized access attempts. Analysts from non-government SOCs reported that the main threat they face is abnormal user activity and phishing emails (82%) and APT (76%).

Large organisations find the main threat to be APT (92%) and viruses, worms, and trojans (85%), and abnormal user activity (77%). On the other hand, medium organisations all agree that abnormal user activity is the main threat their organisation faces and 80% for ransomware and phishing emails. All analysts from small organisations agreed that abnormal user activity, scanning, viruses, trojans, worms, and phishing emails are the top threats.

Tools— Next, we wanted to understand which of the previous threats do their current security systems detect and which they would be better addressing if they

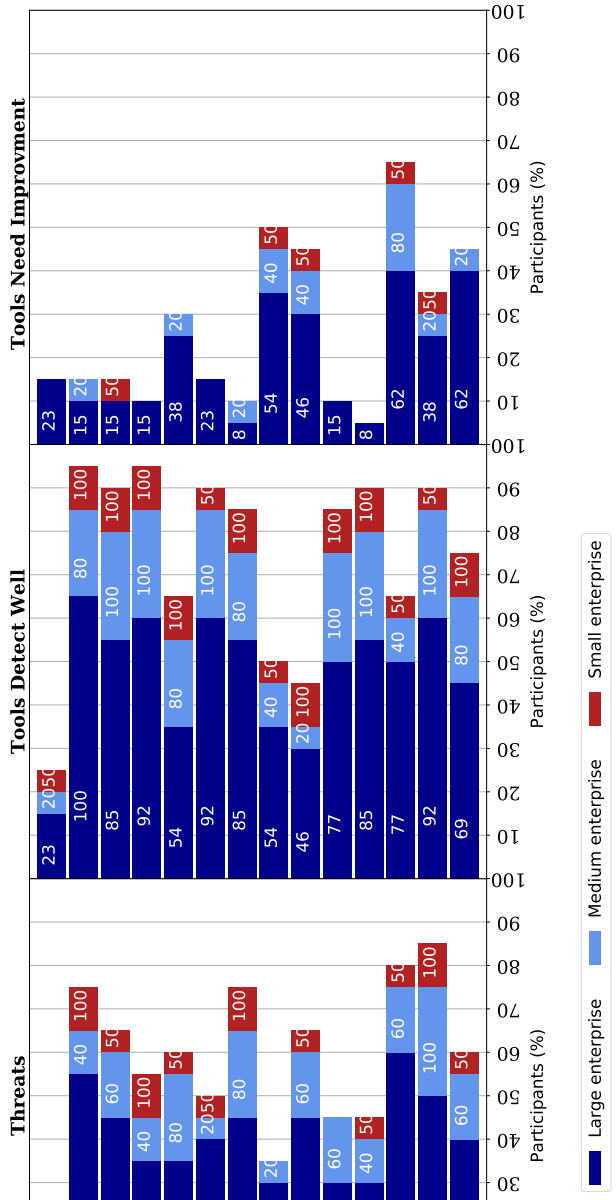

Figure 5.1: Threats organisations face, what tools detect them and what needs improvement based on participating organisation size.

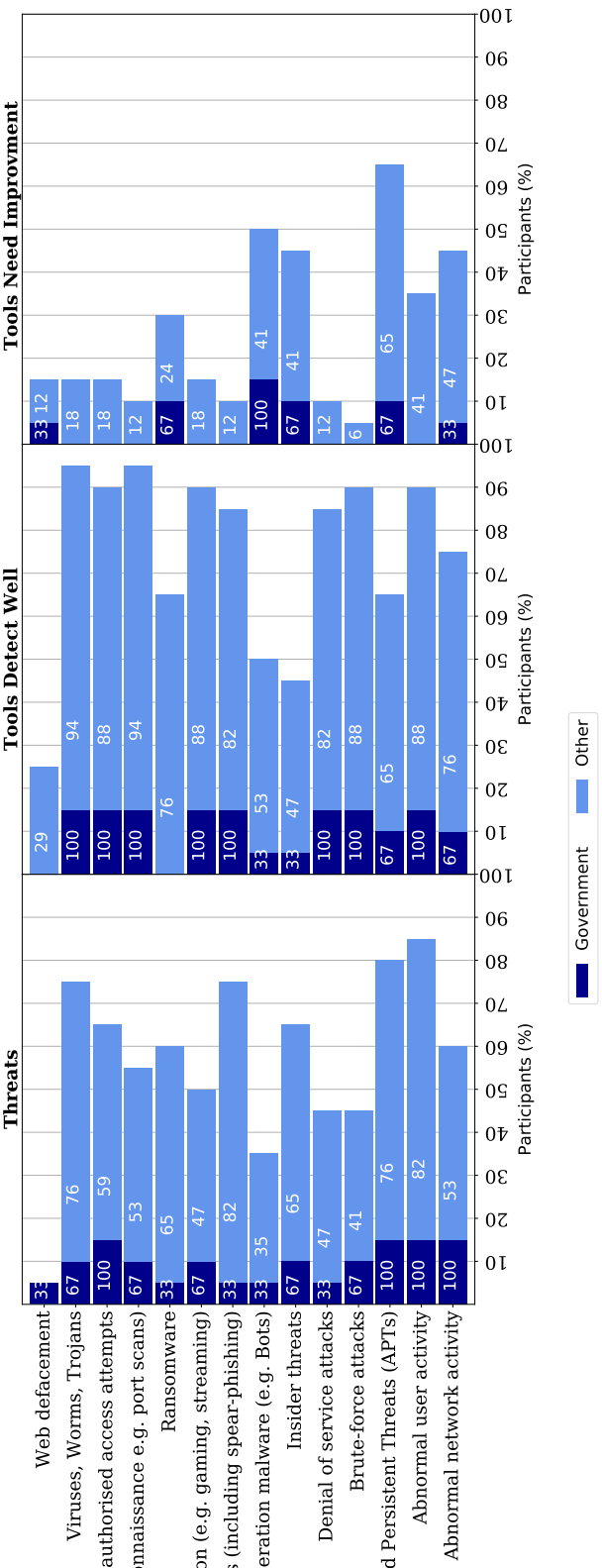

Figure 5.2: Threats organisations face, what tools detect them and what needs improvement based on of SOC client (government and non-government).

had the tools. We asked: *Which of these security incidents do your systems detect?*, and *Which of the following network security incidents do you believe you could do better at addressing if you had the tools?*

Most participants (90%) reported that their tools detect abnormal user activity, which is the main threat they face daily. Similarly, most analysts find that the tools they have are capable of detecting threats such as phishing emails (85%), unauthorized access (90%), scanning (95%), policy violations (90%), DoS (85%), brute force attacks (90%), and viruses, trojans, and worms (95%).

Although 80% of the participants reported that APT is a threat their organisation faces, only 65% of the participants believe that the tools they have can detect them. Still, 65% of the participants believe that they can do better if they had the tools. Similarly, 60% reported that ransomware is a threat, with 65% believing that current tools detect them. Participants that reported that their tools detect ransomware are from non-government organisations. Interestingly only 30% of the participants think they would be better in addressing ransomware if they had the tools.

Insider threats are also a challenge as 45% believe that current tools can detect insiders, while 45% think they can do better if they had the tools. Although 50% of participants reported that next-generation malware (more advanced malware such as bots) is a top threat they face, the participants were divided as 50% reported that their tools are capable of detecting it, whilst 50% think they can do better if they had the tools. Analysts from government SOCs all agreed that they can do better in detecting malware if they had the tools.

5.3 Security Monitoring Tools Use

We asked our participants: *Which of the following network monitoring tools do you use?* We show the results in Figure 5.3. 90% of the participants reported that they use Intrusion Detection Systems (IDS), 80% use Wireshark/Nmap, 80% Use a SIEM and 55% use Data/log Aggregation Tools (e.g. Splunk). However, 20% (4/20) of our participants reported that they do not use a SIEM. In analyzing the data, we found that one participant did not select any tools and commented "*Not at liberty to say.*" However, in further questions, the analyst did specify that they do monitoring using a SIEM. Another analyst reported they use a log aggregation tool (e.g. Splunk) and not a SIEM. The other two participants reported they use an IDS and Wireshark, with one mentioning Cisco ASA, an adaptive security appliance that combines firewall, antivirus, intrusion prevention, and Virtual Private Network (VPN) capabilities.

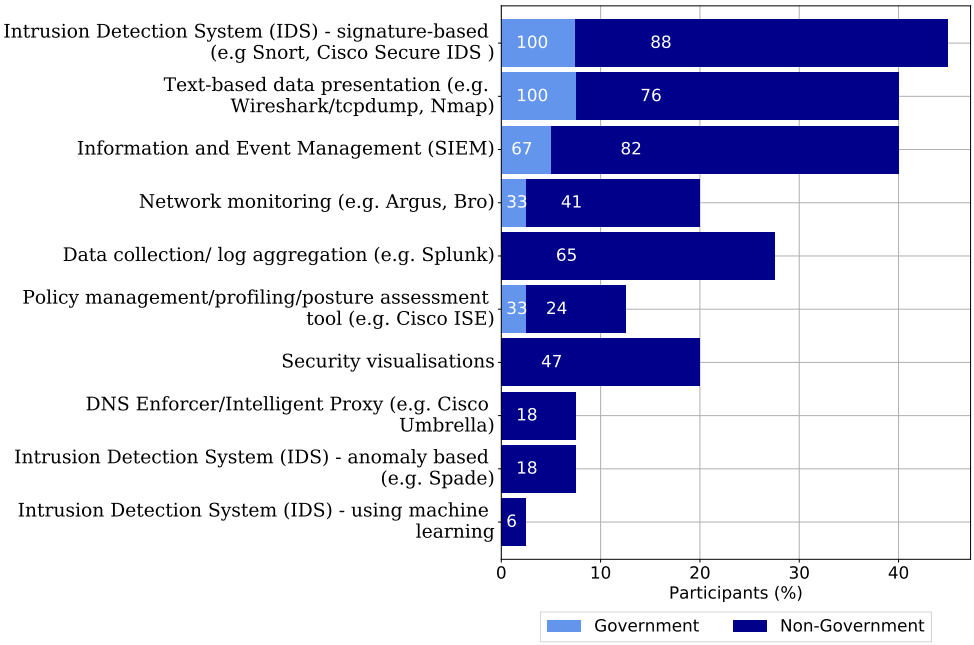

Figure 5.3: Which of the following network monitoring tools do you use?

Only one participant (medium enterprise, 1-9 analysts, non-government) mentioned that they use machine learning-based tools. However, they did not indicate which tool they use. Participants that reported they use anomaly detection tools (15%) are from large non-government organisations.

To determine what features SOCs look for in a security tool, we asked: *How important are the following features in choosing an ideal network monitoring system?* We show the Figure 5.4 that 55% of the participants find that detection accuracy is a very important feature. False Positives (FP) and False Negatives (FN) are also important features— 55% participants think FN is a very important feature. The cost of the tool is also an important factor (50%), as well as cross-vendor API and required processing power (40%). Interestingly, although the cost was a significant factor, only 20% of the participants thought that open source was an important feature. One participant commented: "*Business requirements shift the relative importance of each factor.*"

SIEM— As we discussed in the previous section, 80% analysts indicated they use a SIEM. We wanted to determine which data sources are monitored and whether they are being monitored through a SIEM or individually. We asked: *Which network security data sources do you monitor in your work? For each source you monitor, please indicate whether you monitor the source using a Security Incident and Event Management (SIEM) tool or individually?*

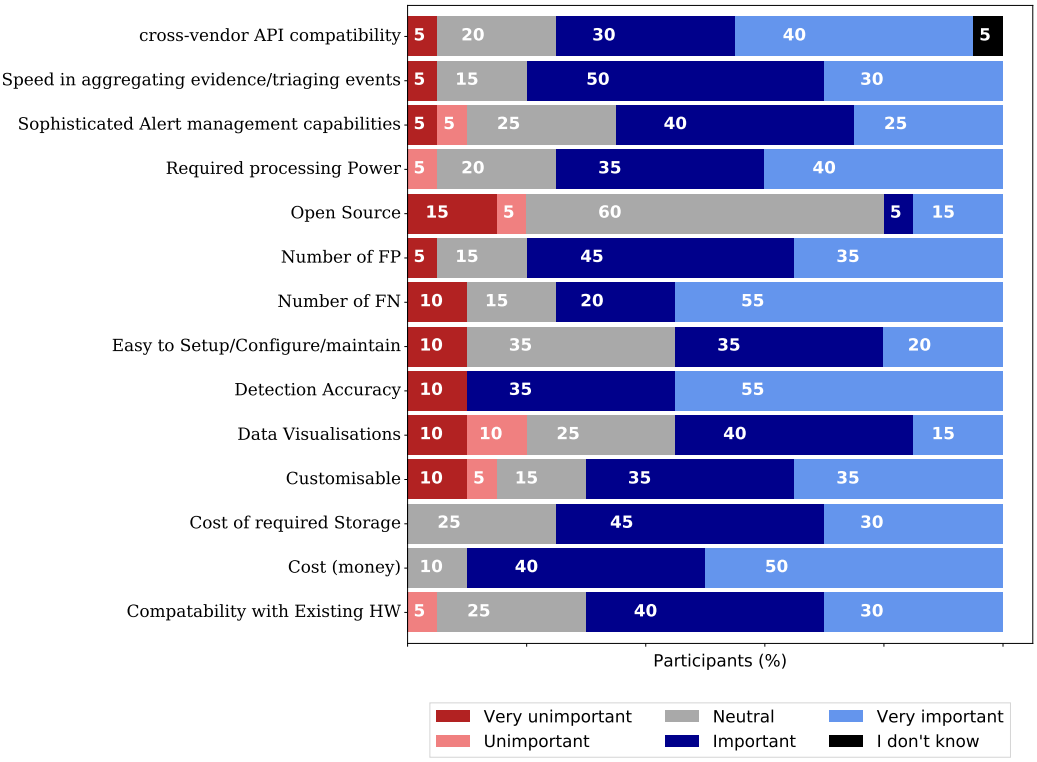

Figure 5.4: How important are the following features in choosing an ideal network monitoring system?

We show the results in Figure 5.5. Most data sources are monitored through a SIEM. However, 30% of the analysts reported that they monitor network packet captures individually, with 10% indicating they do not monitor packet captures at all. Similarly, 20% of the analysts stated they do not monitor *Netflow* data, and 15% monitor it individually. 25% indicated they do not monitor website logs. This is not surprising, as only one analyst reported that web defacement is one of the threats their organisation faces, as shown in Figure 5.1.

We want to determine which logs are considered important in detecting malicious activities on a network. We ask: *Q24: How important are the following data sources in detecting malicious activity on the network?* We show the results in Figure 5.6. Overall, analysts find these data sources important. Other data sources that analysts reported in the comments are threat intelligence and user reporting. One analyst commented, "User reporting is very important and also mail server logs are essential unless you are running PCAP device across your mail traffic."

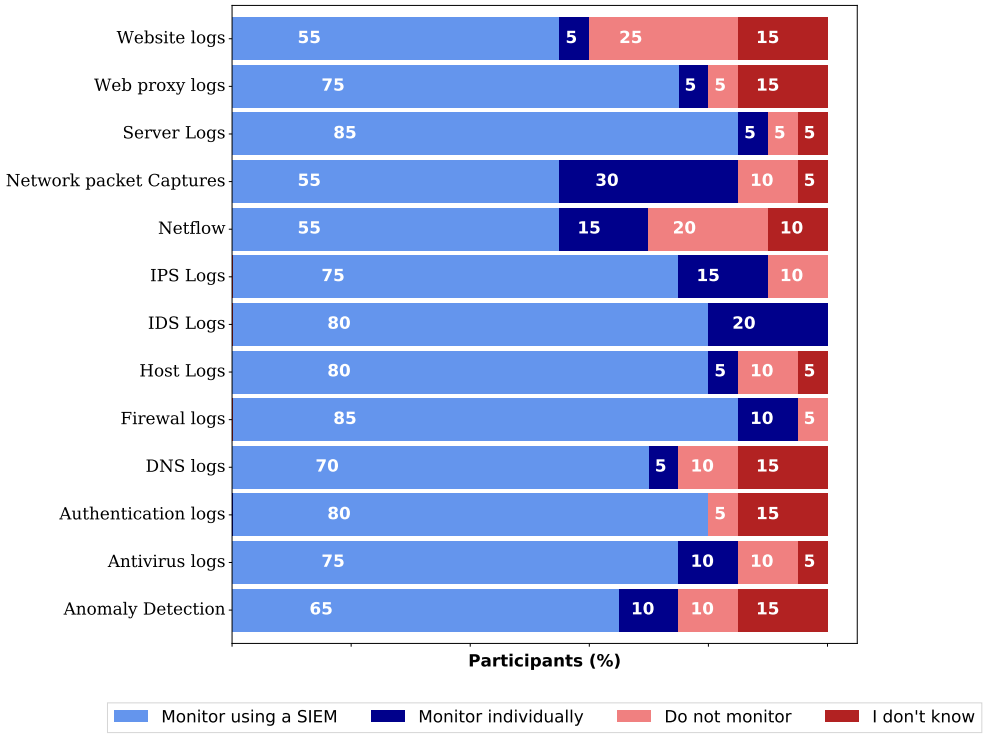

Figure 5.5: Which network security data sources do you monitor in your work? For each source you monitor, please indicate whether you monitor the source using a Security Incident and Event Management (SIEM) tool or individually?

5.4 Network Features & Indicators of Compromise (IOC)

Figure 5.7 shows the analysts opinion on the importance of network traffic features. Analysts found that source and destination IP addresses are an important feature. Equally as important is *Packet Content*, although, it is an issue when traffic is encrypted. However, as pointed out by one participant, the importance of these features depends on the nature of the attack in hand.

When investigating a possible incident, analysts look for Indicator of Compromise (IOC). Figure 5.8 shows the analysts' opinions of the importance of IOC. 70% found that suspicious registry or system file changes and anomalies in the account activity of a privileged user are very important features, both which are host-based activities. However, one participant noted that they look at a combination of IOCs rather than one specific IOC, as threat feeds generally group IOCs.

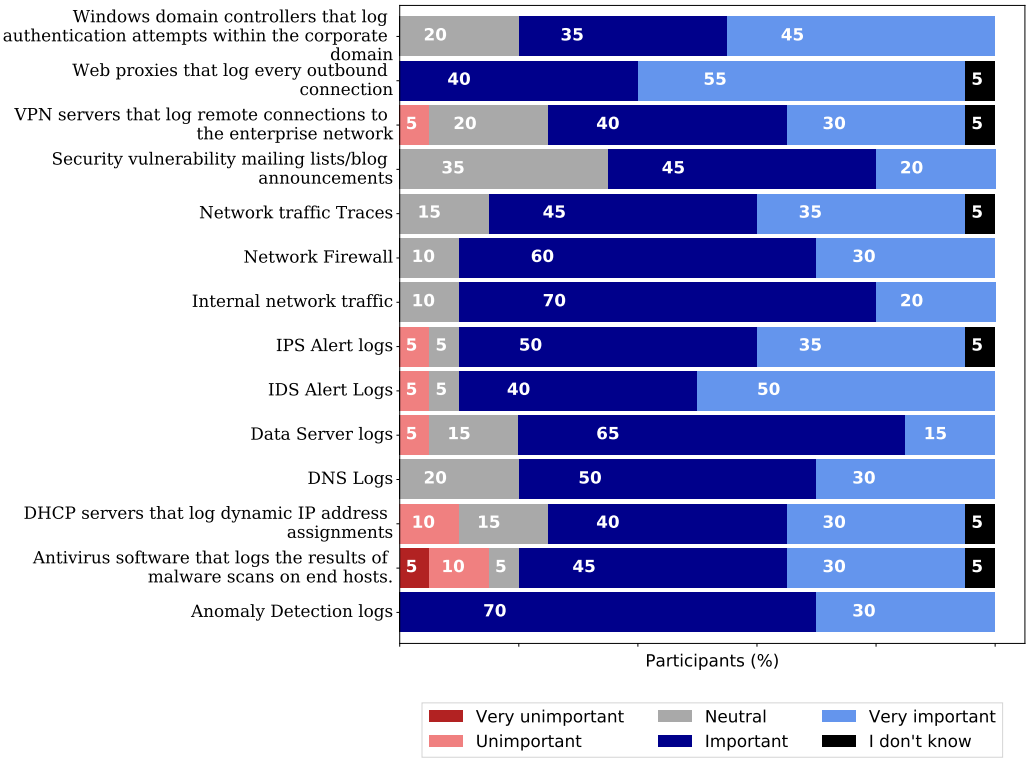

Figure 5.6: How important are the following data sources in detecting malicious activity on the network?

5.5 Network Monitoring Efforts

In our survey, we wanted to understand the network monitoring focus of the participating SOCs. We found that remediation and prevention are mostly done by large organisations (in addition to detection), while medium/small organisations are focusing more on detection. We asked our participants on their responsibilities in the SOC, showing their responses in Figure 5.9. 90% of the participants indicated that their job entails monitoring IDS, 85% on incident handling, and 75% on monitoring network and system logs. 60% reported that their role requires producing technical documentation.

SOCs are known to be overwhelmed with the number of alarms. Hence, we asked our participants to indicate the number of security alarms their organisation receives daily. We show the results in Figure 5.10. 45% reported that they receive less than 5K alarms daily. Participants receiving over 100K alarms are from large enterprises, 40% serving government customers.

From these alarms, some analysts shared the number of alarms they process daily and the amount that they found to be legitimate. For example, one participant indicated that 1 out of 100 alarms they investigate is an actual alarm (i.e. genuine

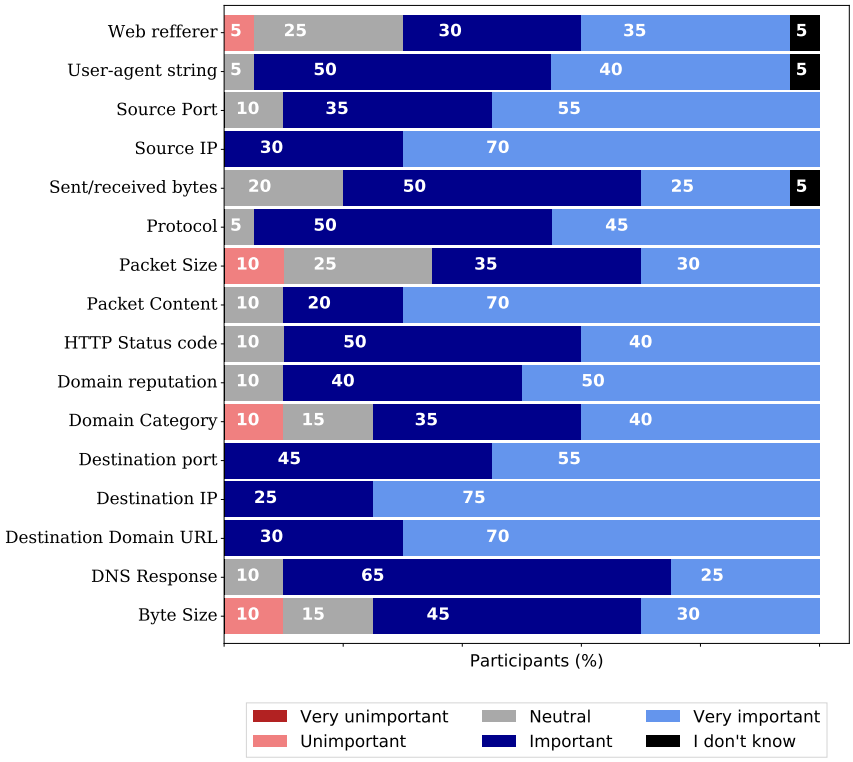

Figure 5.7: How important are the following network traffic features in detecting malicious activity on the network?

threat). Another participant stated a 200:50 ratio. Both participants are not from a large organisation. Participants from small/medium organisations have reported that they investigate more alarms than participants from large organisations, thus encountering more false positives. The sophisticated capabilities of the tools deployed by large organisations, due to having more resources, may lead to producing more reliable alarms. Hence, the overwhelming number of FPs produced in small/medium participating organisation requires them to focus their resources on detection.

5.6 Assertions

In order to obtain the analysts opinion on various areas of the SOC operation, we asked analysts to indicate their level of agreement with each statement. We used Likert-Scale (1: Strongly Disagree, 2: Disagree, 3: Neutral, 4: Agree, 5: Strongly Agree). We present the full results of the assertions in Tables B.1 and B.2.

Human in the Loop— We wanted to measure how much analysts rely on their experience in analysing and detecting malicious activity as opposed to relying on security monitoring system alerts. We show the results in Figure 5.11. 55% analysts indicated that they rely on their experience over 51% of the time. Overall,

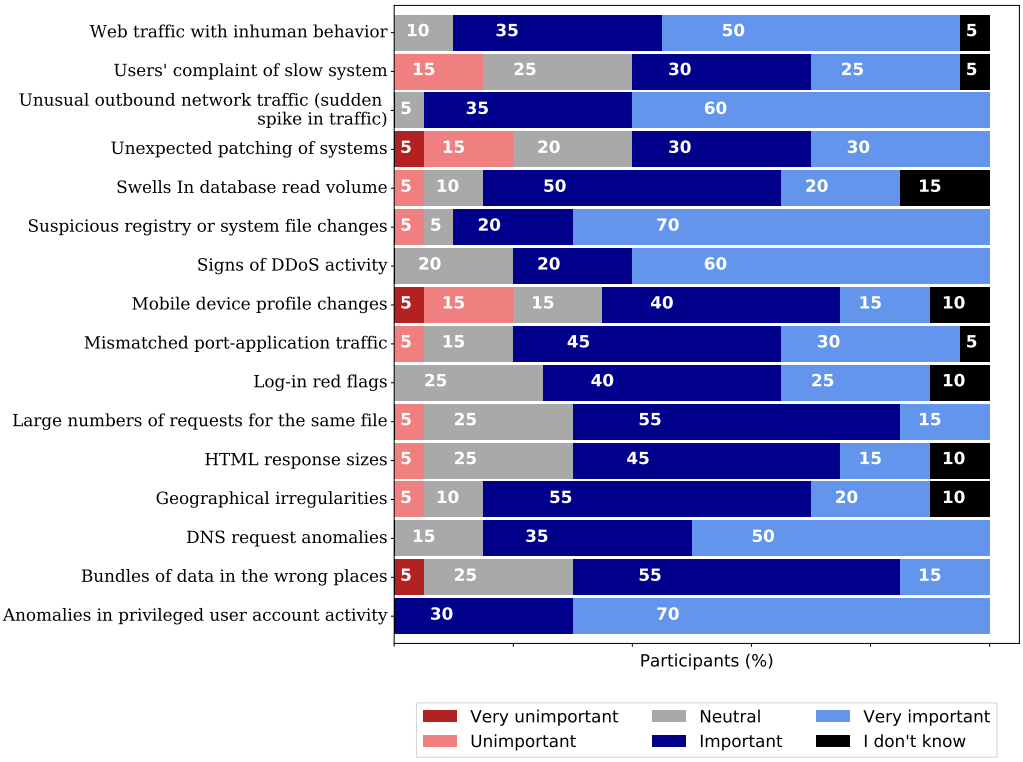

Figure 5.8: How important are the following Indicators of Compromise in detecting malicious activity on the network?

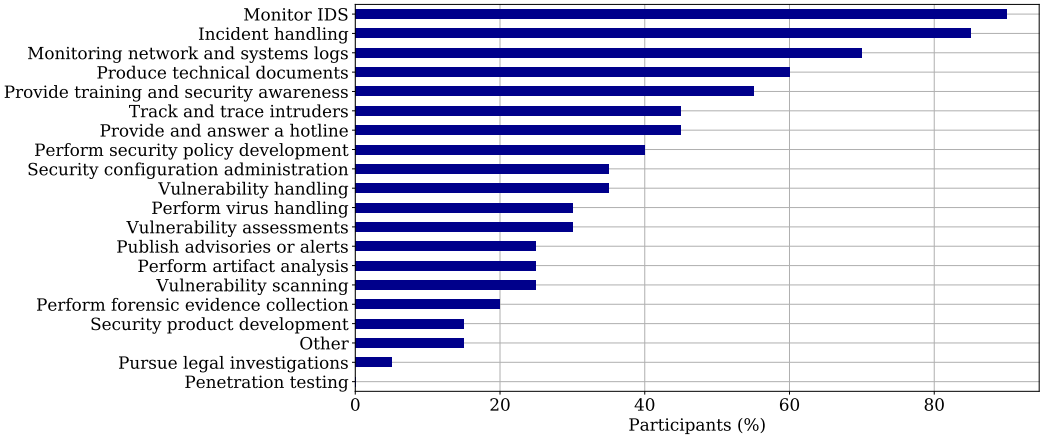

Figure 5.9: Which tasks are you required to carry out in your role?

as the experience of the analyst gets higher the more reliant they are on their experience and intuition. That is expected, as experienced analysts are assigned more complex tasks such as *hunting*. In addition, in Table B.2 — Human in the loop analysts agree on the importance of the human role in detection, filtering false positives, and preliminary analysis of events.

Data Fusion— We wanted to determine whether the aggregation of multiple

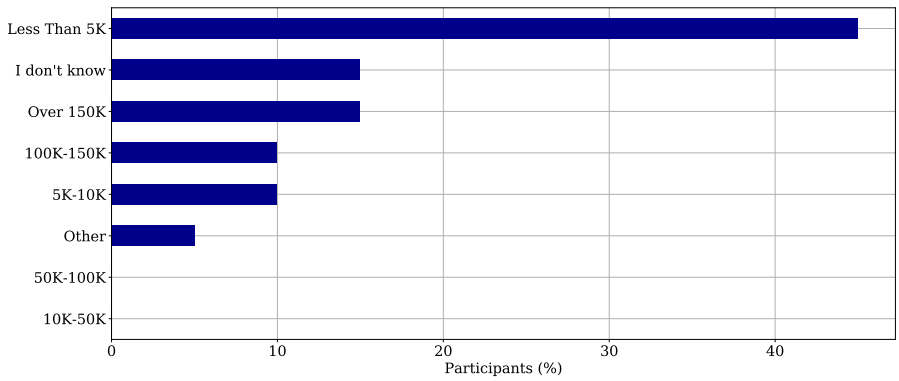

Figure 5.10: How many network security alarms does your organisation receive on average daily?

data sources are helpful for the analyst to perform their job. The analysts agree that they are required to simultaneously monitor information from multiple data sources ($Median = 4, Mode = 4$), as well as monitor the state of multiple network devices($Median = 4, Mode = 4$). Overall, the analysts believe that the tools they have are adequate technologies for aggregation and correlation from multiple sources ($Median = 4, Mode = 4$). However, the analysts reported that the heterogeneous data structures or incomplete/inconsistent logs are still a challenge ($Median = 4, Mode = 4$).

Network Monitoring Tools— We asked analysts how satisfied they are with the network monitoring tools. Specifically, we asked them about the FPs, FNs, and their use of scripts for the detection of threats and aggregation. Most analysts agreed that current network monitoring tools frequently produce false positive alarms ($Median = 4, Mode = 4$). However, the analysts' response on whether the tools deployed in their organisations produce false-negatives was inconclusive ($Median = 3, Mode = 3$).

Intrusion Detection Systems— As we discussed in Section 5.2, 90% of analysts reported they used IDS. We wanted to determine analysts' opinions of on IDS capabilities. Overall, the analysts' responses were quite split between agree and disagree and thus inconclusive. However, analysts do agree that they find that the number of alerts generated by most IDS is overwhelming ($Median = 4, Mode = 4$).

Knowing your Network— We investigate how important analysts believe knowing the network is in analyzing incoming threats. Overall, the analysts agree that it is important to keep up with changes in the configuration of the network and knowing the network environment ($Median = 4, Mode = 4$). Most analysts agreed that they keep track of the network configurations using a database

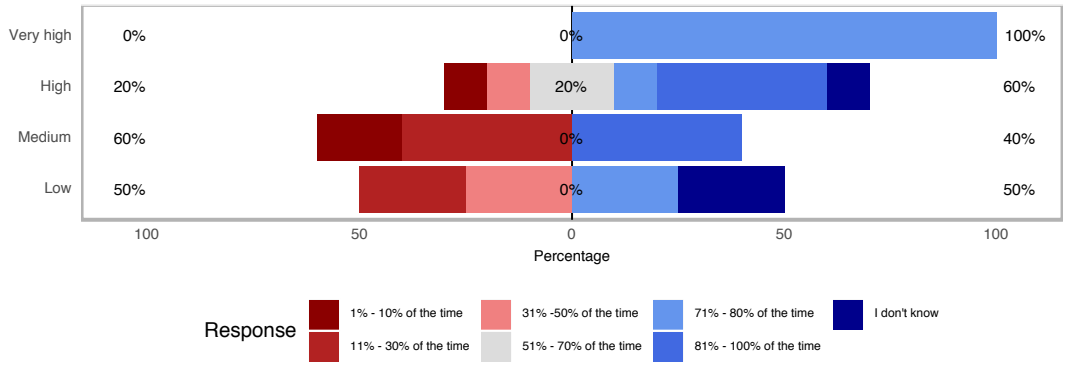

Figure 5.11: How much do you rely on your experience in analysing and aggregating data sources to detect malicious activity as opposed to relying on security monitoring system alerts?

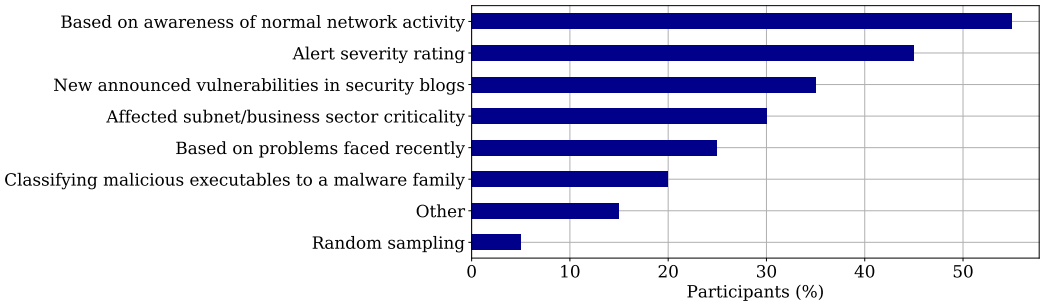

Figure 5.12: How do you choose the security alerts you process?

($Median = 4, Mode = 4$), whilst when asked if they track these configurations using their memory, results were inconclusive ($Median = 3, Mode = 3$). Knowledge of the network is also important in detecting anomalies on the network. Analysts were asked: *How do you choose the security alerts you process?* As shown in Figure 5.12, 57.9% of the analysts indicated that they chose the alerts based on awareness of normal network activity, and 47.4% said based on the alert severity rating.

5.7 Summary of Findings

The quantitative study gave us valuable insights into the participating SOCs that we explore further using qualitative research methods in following chapters. These insights are as followed.

Importance of Human in the Loop— We found that security practitioners believe that their role (the role of the human) is important in the detection and that sometimes they rely on their experience to detect anomalous activity. We investigate further the balance between technology and humans in SOC operations in Chapter 6.

Influencing Factors— Our survey results imply that customers from public sector handle threats and tools differently than other sectors. For example, tools detecting ransomware were seen mostly used in non-government organisations. Moreover, a survey participant noted (in one of the free-text comments boxes included in the survey interface) that answers to questions on SOC operations may vary based on whether the SOC is in-house or an MSSP: "*You should consider the model of an MSSP in SOC monitoring practices.[..] The model an MSSP will follow is everything is put in one queue rather than an analyst being devoted to a particular sensor or customer*". We therefore aim to investigate further the way that factor (in-house or MSSP) influences SOC processes and identify further factors in Chapter 6.

Threat Investigation— In our survey, we asked the analysts to rate the importance of a list of network traffic features and indicators of compromise (IOC) in detecting malicious activity on the network. However, as raised by one of the participants—"*Basically depends on the nature of the attack; the approach is different"*. We explore this further in Chapter 6 to study how such indicators are thought of by analysts when investigating a possible incident.

Importance of Knowing your Network— We found that practitioners believe that their knowledge of the network is essential in detecting anomalous behaviour on the network. Furthermore, analysts reported that they track the network configurations as it is necessary to provide context to diagnose an alert. We explore this further in Chapter 7, identifying how such context is vital to the analysts' job.

Network Traffic Collection— Only 30% of the analysts reported that they monitor network packets captures individually. We explore this finding further in Chapter 7. In particular, when in the incident monitoring, detection and response process are packet captures collected and examined. In addition, we will also explore whether SOCs collect and store network traffic and at what granularity (e.g. Netflow, PCAPs).

SOC Tools— We gained initial insight into the tools analysts use. In Chapter 7, we investigate such technologies further. In particular, how these tools are used and analysts' opinions on their weaknesses and strengths. In addition, we plan to explore their use of machine learning-based tools.

6

Malware Detection Processes in Security Operations Centres

Contents

6.1 Introduction

In this chapter, we describe the SOC operations, focusing on the previously understudied role that humans play. We explore the workflows security practitioners follow to prepare, detect, and investigate a threat — including the kinds of activities security practitioners engage in daily, the tools they use, and the skills required.

As we identified Chapter 5, security practitioners believe that their experience plays a significant role in their everyday tasks. We investigate this further and determine the current state of balance between the human and technology, understand why some malware attacks slip through and identify opportunities for automation of operational bottlenecks. SOC operations are impacted by various factors that we elicit during our interviews, such as the type of customer and SOC.

Drawing on the findings from the quantitative study (discussed in Chapter 5), we designed interview questions to address the following research questions:

- Which of the processes involved in the work of security practitioners could be automated, and which require human input?
- What factors (internal or external) influence SOC operations?

6.2 SOC Operations Overview

We believe that understanding the inner workings of a SOC is essential for researchers to propose effective threat detection. This section brings together our findings about the various processes that practitioners follow to monitor, detect, and respond to security threats in a SOC.

Our interview analysis revealed a common workflow across the participating SOCs. The workflow consists of four main steps: Preparation, Monitoring & Detection, Investigation, and Response, illustrated in Figure 6.1. In describing these processes, we highlight the tasks that are *human-centric*, meaning tasks that require human interactions, collaborations, reasoning, and human intelligence, which represent our first findings. To ensure clarity in our discussion, we will refer to the organisation a SOC is protecting as "*customer*", for both in-house and MSSP SOCs.

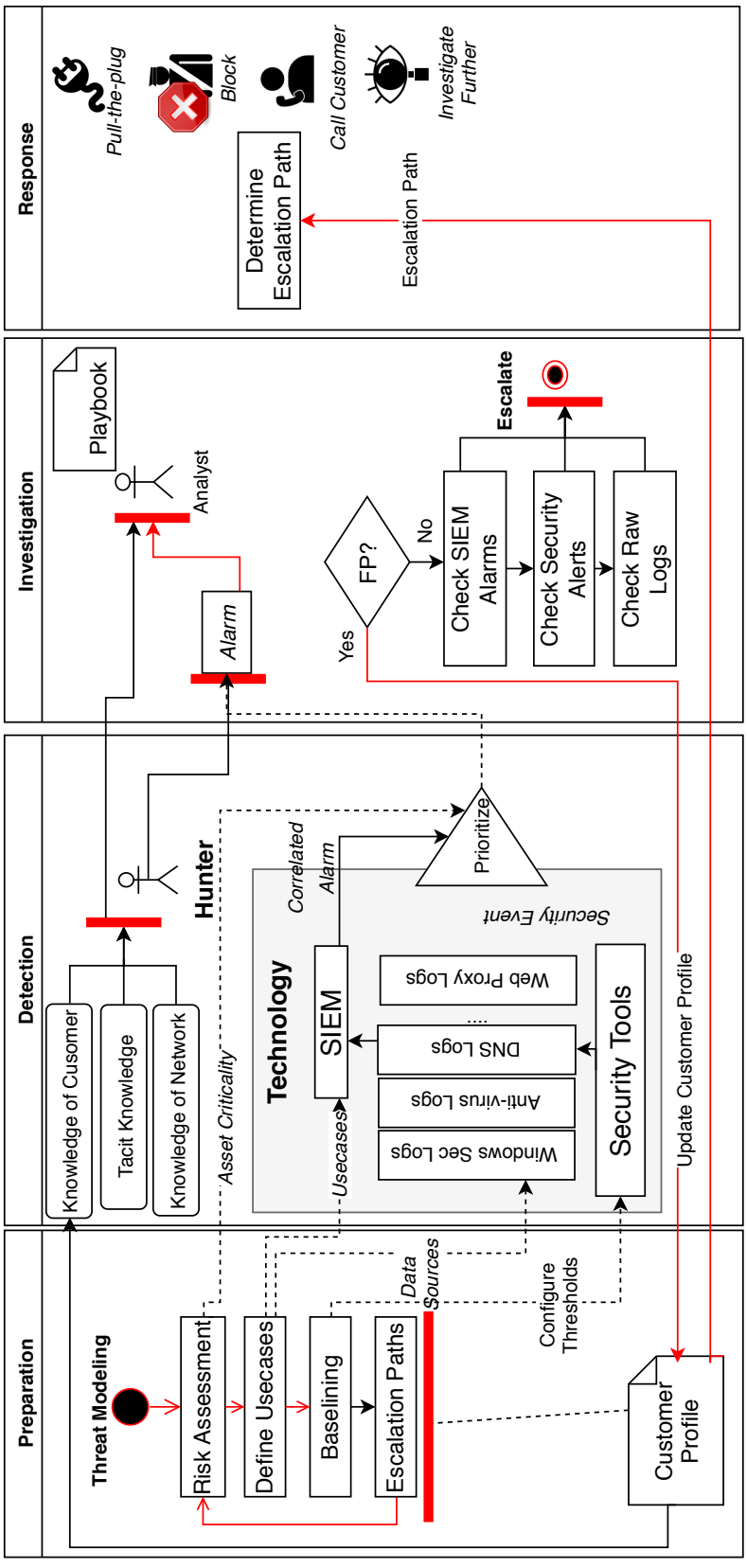

Figure 6.1: SOC Operations Workflow: Preparation, Detection, Investigation, Response

6.2.1 Preparation

Participants across four SOCs (P17, P2, P9, P5), have referred to the "*Threat Modelling*" or "*Customer On-boarding*". During this process, the SOC aims to answer these main questions: *What threats does the organisation care about? What does a threat look like?* and *How should the SOC respond to the threat?* The answers to these questions guide the operations and decision making in a SOC. An in-house SOC goes through this process for their organisation only. In contrast, MSSP SOCs need to do this process for each customer. Hence, in this section we will focus on MSSP SOCs participants reflection on how they liaise with their customers during the *Preparation* phase.

What threats does the organisation care about? How analysts monitor, detect, and respond to incidents depends on the customer's risk appetite. *Risk Appetite* can be defined as *"The amount and type of risk that an organisation is prepared to pursue, retain or take"* [188]. The risk appetite for a customer depends on their Risk Assessment and influences SOC operations. For example, it impacts the *prioritization* of alarms generated by SIEMs and how analysts respond to incidents, ensuring practitioners focus their efforts on detecting threats that target important business operations. *Asset Criticality* (an output of the Risk Assessment) also is considered in the prioritization. SOC practitioners engage with customers to understand business risks and identify the most important assets. The SOC works together with the customer to identify such information and document it.

> *P2: We will bring them [customers] on board and do a two-day session, which is threat modeling, basically saying, "Right, what you are using could be important, not just within your own environment." They will say, "This is our most important system" and then we will work them out from there.*

What does a threat look like? Even if technology provides full automation, humans are still required to for configuration. When asked about how they were able to detect a particular incident, P5 explained that the analysts were able while the technology failed to detect. He attributed that to technology not being configured by the human to detect such a threat. Hence, human are needed not only to detect things that technology missed but to design scenarios or use cases of what should be detected (what a threat looks like) and configure the technology accordingly.

> *P5: So I think that was a case of humans spotting something that a machine didn't spot, but then, that's arguably because the machine had been – well, not even arguably, that was because the machine hadn't been configured to spot it.*

Define Use-cases— An example of a technology that needs to be configured by a human is the SIEM. As part of the preparation, analysts need to design SIEM use-cases to detect particular scenarios the customer is interested in. A use-case is defined as "*Specific condition or event (usually related to a specific threat) to be detected or reported by the security tool*" [189]. P18 explained how SIEMs require humans to configure it on what to look for:

> *P18: If you look at the way SIEM solutions are set up. They're designed with that in mind to try and reduce the amount that the human has to do, but still the kind of rules, "what is it I'm looking for?" is still initially has to be set up by a human, so there's definitely room for the human still.*

When P20 described designing the use-case, he referred to *imagination*, a human ability.

> *P20: All of these things are easily doable, you just have to sit down and write the use-case, and the limit of your imagination is the limit to what you can more-than-likely do with most SIEM tools.*

Define Data Sources— In addition to configuring the technology, SOC practitioners, through discussions with the customer, attempt to understand how the organisation uses its assets to determine the data sources they would need from the monitoring. As explained by P2:

> *P2: They [customer] will say, "This is how our users access this system" and we will say [SOC], "These are the sort of logs we need to actually be able to monitor this for you.".*

The SOC will request the basic data sources needed for threat monitoring (e.g. Anti-Virus (AV) logs). Then, more explicit use cases are discussed with the customer. As P17 remarks:

> *P17: Whenever we take a customer on board, we basically aim for a baseline. We want your DNS logs, we want your DHCP logs, we want your AV logs, we want firewalls, we want...you know, all the core stuff.*

It is important to identify the data sources required for each use-case, collecting only the bare minimum needed. As P20 explained.

> *P20: You have to choose those logs appropriately to meet your use-cases rather than just say "I'll collect everything" because then your network becomes more about sending logs than it does about doing anything else. So, it's choosing the minimum number of logs to meet the use-case, rather than - so yes it can, a SIEM can do absolutely anything you want it to, as long as you can collect the logs.*

Alarm Tuning & Baselining— Alarms generated as a result of these use-cases are periodically reviewed with the customer to tune the defined alarms further, eliminate false positives, or alarms the customer does not need to be informed about (i.e. benign triggers). P9 explained how SOC practitioners review these alarms with the customer to check that the alarms are within the customer's tolerance.

> *P9: We'll then dig into those and look for, is there noise in there we will look at the tickets that were generated for the customers, and how many of those will come back from the customers as "well we sort of want to know about this but we sort of don't".*

Each customer is different, and consequently, their network communications and system usage is unique. Therefore, one important step is identifying the "*normal*" baseline. What is considered a normal network communication or system usage for a customer? As P9 remarks:

> *P9: When taking in a new customer, the SOC provider in the first months attempts to understand what's normal within the customer's network.*

Through understanding the customer's normal environment use, the SOC can adjust the security tools' parameters accordingly. For example, when deploying an anomaly detection device, you need to configure the device to alert you when the network traffic goes above a certain threshold. If the threshold is configured too low in an environment where the traffic could be high, then this will generate multiple false positives. So it is down to the analyst to tune the parameters.

> *P6: A human will have to adjust those parameters [...] or else you'd be getting high, high, high and it would turn out to be nothing.*

The process of base-lining the new asset might take up to a month, depending on the type of the asset. For example, a webserver will be generating many types of alarms that need to be filtered out to the point both the customer and SOC are satisfied. As P9 noted *"So then we will track that through on a ticket until ourselves, and the customer are happy that the output that they're getting from that alarming for that box are now appropriate to the new asset type."*

How does the SOC respond to a threat? What happens when an alarm is triggered needs to be determined in advance with the customer. In particular, how the SOC contacts the customer (*Escalation Paths*) and how the SOC handles the incident. The latter is determined when defining the use-cases and documented in the *Customer Profile*.

Escalation Paths— One of the main objectives of the preparation stage is defining the escalation paths for each customer. Specifically, when and whom the SOC can contact from the customer or upper management to escalate an incident. In addition, the customer needs to know their point of contact in the SOC. Escalation paths are defined for both in-house and MSSP alike. These pre-defined communications are defined in advance to help avoid disorientation when an incident occurs. As P9 remarks on escalation paths:

> *P9: So that when we log a ticket on the customer during an event, they know what's going to happen, it's not suddenly 'ooh blimey something's happening', everybody panicking, they've already got a pre-defined communication, that they'll understand how we're going to engage with them.*

Customer Profile— Information learned about the customer is documented in a database or a wiki per customer, which analysts refer to as the *customer profile*. This is another way for indirect communication between analysts to ensure any information that may impact SOC operations is communicated to all SOC practitioners involved.

For example, such profiles include information about potential false positives (i.e. benign triggers) the analysts should ignore. In addition, the profile contains information about the customer's assets such as (1) Details about hosts and servers in the network, their IP addresses, DNS names, and Operating System (OS) building a profile per host. (2) The criticality and priority of the asset based on the Risk Assessment. (3) Information about asset change from customer's asset management process. Information about the nature of the customer's business is also included in the profile.

As P5 explained, although they attempt to gather such information from discussions with the customers themselves, they may refer to open-intelligence tools (e.g. Shodan [1]) or search engines to gather other information about the business (e.g. work hours).

> *P5: We will get as much information as we can from each customer, some are a lot more responsive than others, [...], but we will do a level of open-source investigations, and then we will periodically, as things arise, we will try and review that with the customer, and keep up-to-date with it really.*

[1] `https://www.shodan.io/`

The preparation process is a critical process for SOC operations. It influences how and what the SOC should monitor as the customer grows the business, and their risk appetite changes or new assets are introduced. Therefore, analysts spend time collecting such information from the customer.

> *P17: Then that's [Threat Modelling] collected on the day, and over a period of time, it takes quite a while to get host information and network information from customers because usually they're all still writing it down..*".

Threat Modelling is an iterative process that should be done periodically, especially when new assets are introduced. As P2 noted:

> *P2: So we are doing that, I believe once a quarter with each customer to actually continuously improve the log sources and the visibility we have got.*

6.2.2 Detection

The previous step prepares the SOC to monitor the customer's network and host environments for threats. In particular, (1) identifying the assets and their criticality for alarm prioritization, (2) Configure SIEM alarms from use-cases and thresholds for security tools, (3) Obtain the required logs and deploy proprietary sensors, and (4) document escalation path and customer's business and technical details.

The SOC is now ready to start the monitoring and detection of potential threats. Detecting a threat may occur through reports from end-users or other teams in the organisation, through hunting performed by analysts on an ad-hoc basis, or it may be accomplished by using technology.

In this section, we will discuss how the SOC analysts detect potential threats through the use of detection tools (*Technology*) and through *Hunting*.

Technology— SOCs rely on various technologies for monitoring and detection of forensics. For example, as we discussed in Section 6.2.1, SOC analysts need to get a baseline of the customer's normal network behaviour for anomaly detection tools. As we found in Chapter 5, one of the most frequently used tools was SIEMs. SOC practitioners (humans) configure the SIEM to monitor and generate alarms based on the use-cases defined in Section 6.2.1. These use-cases are usually thought of during the Threat modelling process, from what the scenarios are to what data sources (logs) are needed. The SIEM then monitors for these use-cases and generates an alarm when a use-case was flagged.

For example, P20 believes that a SIEM can be configured to do what the analysts needs, as long as you can boil that down to basic logic, and where you're going to get that information from. He provided an example of incorporating physical access security system in the SIEM tool so that if somebody logged on and they were supposed to be offsite, the nearest camera to their desk can be activated, and the feed can be observed.

Once the SIEM raises an alarm, it is prioritized depending on the use-case triggered, the asset criticality (obtained from customer profile), and the predefined agreement logged in the *customer profile*.

> *P17: If we could see that there's a lot of traffic going out to a host, say, in a suspicious geographic region, for example, and it looks like they're transmitting a lot of data outbound and, say, it was a critical server internally, domain controller or file server or something like that then that would shoot right up there in our high priorities to get the communications blocked, especially if it's something that we haven't seen before.*

As P17 described, the detection process usually involves looking at multiple observations and deriving for patterns and correlations, which may be obvious to humans, but not to the available tools. These alarms are reviewed by the analysts to decide whether it is an incident or a false positive. Nevertheless, technology can be more effective than humans for detection at scale.

> *P9: So, there are instances where humans will spot things that machines haven't, but I think in terms of large volumes of things, machines can do it a lot more efficiently than we can.*

However, in certain cases, an alarm might not be generated by the SIEM at all. In other cases, the attack is stealthy (e.g. APTs), and their activities are spread out across months, hence, not captured by technology.

> *P15: An alert may never be produced, but just because you have a hunting capability that can make a more complex human-based estimation of why something looks abnormal over a 3-4 month period, which the big data now is trying to address as well, so you are able to query much faster, much larger datasets.*

Therefore, "Hunting", where analysts (humans) look at the data provided by technology to spot missed threats, is a critical part of the SOC operation.

Hunting— Although analysts still rely on alarms generated by the SIEM, humans have the ability to proactively look at the data or visualizations and spot if 'something's weird' or 'something that doesn't look right'. As P15 remarks:

> *P15: It's the hunting human capability, I haven't at least seen a big data, autonomous solution that actually is able to give you security insights on its own.*

Hunting is the process of actively going through data sources to find any abnormal or "weird" activity. As P15 and P19 noted:

> *P15: The work of a hunter never starts with there is an alert that has been correlated.*

> *P19: The words I don't want to hear out of his mouth is "I'm bored". Because "I'm bored" means he's going to go and look for something, which means I'm going to spend the next ten days running around.*

Similarly, P2 and P17 explained how they conduct a hunting session monthly.

> *P2: What we try to do once a month is a hunt day. So we will basically go on a deep dive. Just, "Right, let's see what we can find." And just start doing investigations into everything.*

> *P17: Once a month we'll have a threat hunting session where we'd get in more analysts for the same day, so that would be a bit more active team building and sifting through the logs trying to find anomalies.*

Analysts seek things that they do not expect to produce security events, yet they believe are uncommon or not normal. For example, an analyst mentioned an anomalous activity could be a discrepancy between certain applications and protocols and usage, or data and usage. In addition, hunting involves looking for larger or "big" picture discrepancies that do not necessarily produce an alert for technical reasons. P15 noted:

> *P15: So the assumption is, you seek for something that won't correlate, that appears very normal, very casual, and won't trigger anything, won't trigger a network-monitoring event, won't trigger a SIEM event afterwards, won't do any of that.*

6.2.3 Investigation

Once a SIEM reports an alarm or a hunter detects unusual activity, the analyst starts investigating the event. Although technology might detect and generate the alarm, it is still the analyst who needs to evaluate that alarm to determine if its a false positive, benign trigger, and how to act and respond to the alarm. In doing so, analysts rely on their human cognitive abilities (e.g. pattern matching, association, reasoning, and computation).

> *P6: And I think that that's where the human element still remains, because even when you get an alert, the alert will have to be sent to a human to make that intelligent decision.*

To understand the analyst's thought process during the investigation, we asked our participating practitioners to discuss how they will investigate a particular incident to elicit their reasoning. For that, we described a scenario (e.g. an increase in outbound network traffic from a server). We asked them to discuss the process they will follow, the tools they used, data sources looked at, and the analytical questions they will think about throughout the incident. Our participants described that they would first determine if the alarm is a false positive. If not a false positive, then the analysts investigate the alarm further.

Investigating Suspicious Network Activity— Once the analysts eliminate the possibility of a false positive or a known activity, they start to investigate further. When explaining their analytical thinking, analysts would begin by looking at high-level SIEM security events going in deeper into raw logs as they progress into the investigation. For example, if an analyst is unable to determine the reason for an alarm from the SIEM security events and logs, they may start looking at PCAP files, if available, or start collecting PCAPs for in-depth investigation.

When investigating an incident, analysts may follow a predefined set of procedures and processes on handling a particular alarm documented in a "*Playbook*". However, although *Playbooks* exist in most SOCs, analysts do not necessarily follow them. As P2 remarks:

> *P2: I know everyone says ideally there are processes in place we should follow. You should do this that and the other. I am not great at that. I tend to go dig for the interesting stuff first and fill in the blanks afterwards.*

Although the procedures for handling an alarm are defined, analysts fall back to their human cognitive abilities to investigate an incident. When our participants described their thought process and actions, we found that although tools such as SIEMs may assist them in correlating alarms, in most cases, analysts are more capable in connecting these patterns to detect threats than SIEMs.

> *P5: Some of the correlation, obviously it isn't done purely by Splunk or by one of these tools, you've got to correlate the things in your brain, and think "why would - this has happened, what could result in this happening?" and there's stuff like that that, arguably at some point, you could maybe get a machine to do for you, but I think currently it's better in the hands of humans.*

For example, when investigating suspicious network activity, analysts mentioned the following analytical questions that are are a result of their own reasoning, using the existing technologies (e.g. SIEM) to query for answers. Is the activity still ongoing? What processes were running on the server? What is the server connecting to, where it is going and where it is coming from? What communications are they using? Which user was logged into the server at that time? Answers to such questions help analysts to determine what they are dealing with. Their ability to look at the data and use their analytical skills to find patterns and make decisions is not easily replicated in automated systems.

> *P2: Looking for multiple things like chaining together and stuff like that. Whether it will manage it, I think it is just a bit too much intelligence to put into an automated tool, and I think you always need that little bit of human interaction just to say, "That's not normal".*

Similarly, as P17 described, the detection process usually involves looking at multiple observations and deriving patterns and correlations.

> *P17: If we could see that there's a lot of traffic going out to a host, say, in a suspicious geographic region, for example, and it looks like they're transmitting a lot of data outbound and, say, it was a critical server internally, domain controller or file server or something like that then that would shoot right up there in our high priorities to get the communications blocked, especially if it's something that we haven't seen before.*

Investigating Malware— Investigating malware infections relies on whether the detected Indicator of Compromise (IOC) is that of known malware. We discussed in Chapter 5, section 5.4 the importance of IOC in detection malicious activity on the network. If the IOC is of a known malware, the SIEM would have generated an alert with a known IOC such as an MD5, a URL, or match a particular string. If that is the case, then the analysts simply create and deploy signatures for these IOCs.

The main challenge is when the malware has never been seen before or is unknown. In such cases, malware is detected due to an IOC or an alert firing in the SIEM that was investigated and turned out to be a result of malware. These IOC could be host-based or network-based. For example, a network-based IOC could be abnormal or "weird" traffic.

Examples analysts provided of what they consider unusual traffic includes: (1) traffic from a misconfigured device, (2) email traffic coming from a server that isn't an email server (e.g. domain controllers emailing people), (3) sudden spike in the outgoing traffic of a server. A host-based IOC are indicators found in the host, such

as a file name, mutex, loading or side-loading of a file. It also includes log entries such as failed log-ins for a local account, privilege escalation, or creation of a new service on the host that are found in the Operating System (OS) security events.

The unusual network traffic detected by the SIEM may be a result of a malware-infected host. So when analysts cannot determine the source of the traffic, they may resolve to remove the host from the network and perform on-demand AV scans using the latest AV. If it is a definite hit and its definitely a malware, then the analysts need first to determine what the malware does.

What is the malware? The analysts need to determine the type of malware. Is it going to attempt to spread, or is it going to try and stay local and hide itself? Some analysts noted that they would try to determine what the malware is before isolating the network.

> *P7: Different malware work in different ways, if it is spreadable it could affect other machines we have to know that. Does it download further malware, we have to know the characteristics of what we are actually dealing with at the time. That is possibly one of the first things we look at, what can this thing do.*

If it is a known malware detected, for example, by a security device, then you have its name and so have immediate access to the description (IOC). Based on that information, the analysts may decide to notify the customer to pull the plug. As P9 noted, *"I would see the indicator of compromise, do a little bit of research as quick as possible in terms of the capabilities of said malware, and then based upon that is how I would make my decision."*

Unfortunately, establishing the type of malware directly from the security devices may not always be straightforward. The analysts themselves would then need to research the malware name by looking for the IOC.

> *P2: If you are looking at abnormal activity, you can say ' There is something on that machine, I am not sure what but it is doing really crazy stuff.' So you are looking yourself, looking for indicators, You are looking for changes. So you don't always know straight away.*

Analysts often refer to public sites such as *VirusTotal* or AV sites, to get information on a detected executable. Getting information directly from such sites allows analysts to start incident handling quicker.

> *P2: However, it's more important to limit the spread first (So as a Forensic step). So if I can grab the name of it (from Snort), jump on one of the many blogs which has already profiled it This is how it spread. You are looking for port 4898 traffic' then I can start monitoring for it a lot quicker. Sandbox, 10-15 minutes to get results half the time people have done it for you already.*

However, such public sites, although useful, may provide different classification to the same malware, and it is up to the analysts to determine which one to rely on.

> *P7: Its bit difficult sometimes, a lot of AV sites and information, each have a different intake on what a piece of malware does, so yes it is difficult to pick which one to use. We do go through forums and such and if they all say the same thing, yes this thing does do this, we act accordingly.*

Analysts are also faced with a dilemma when using public websites to get answers as quickly as possible as this risks the malware author knowing their detection, specifically in more targeted or APT malware. However, P19 remarks: *"We also use the malware site, if it's publicly available information, obviously we can't use internet tools for most of the stuff that we do onsite here."* Similarly, P9 remarks: *"So if you have seen something that could be very targeted, you wouldn't go to any of that stuff, because that could indicate to your attacker that you're aware of it, and then they can change their methods."*

As a last resort, the analyst may run the malware executable (if they can get access to it) in a sand-boxed environment to determine what it does. However, this is usually done in the forensics phase as it takes time to get the knowledge you want.

> *P2: We do have sandboxes set up. [..] But yeah if you do explode it you have got a massive 50 odd page report of everything that is going on. It is going to take you a while to get what you need out of it.*

Investigation Process— For investigating the malware incident, analysts primarily referred to three methods. (1) They first utilize the SIEM by looking at the security events, and logs. (2) They may then look at IOC provided by other security devices such as firewalls. (3) The analyst may also perform hunting to look for other IOCs or similar infections.

When looking at the SIEM, analysts start by looking at the interactions from the infected host. They monitor what is leaving the host, looking for any interaction with any known compromised hosts. They also look at what that infected host talked to since the point where the compromise was identified. In addition, they look at what processes the host is running.

> *P2: If you logged under the workstation, then set a tail for the username, IP, hostname, basically monitor everything your host is doing. Stick that on one screen. Have a summary of events running down, basically saying 'This is what that machine is doing.*

Analysts reported that they usually interact with the SIEM platform to determine these interactions. Therefore, it is vital during the Threat Modelling Process that they capture all of the log types, that would answer questions about such communications without requiring to go to the customer onsite.

In addition to looking at the SIEM, the analysts may look at other security devices that can see any other known indicators of compromise, e.g. firewall. One participant mentioned that they would start looking at technologies that they have the "*highest respect for or has proved to be effective for a customer*". Usually, for malware, one analyst argues that the "*first and optimal choice*" is to have APT-based available technology to determine if it can say something about the detected malware.

> *P15: So, the technology that is most contemporary and that has proven in real-life battles is likely, that we know is available [..]. So, that is the way we would start looking at it.*

Other sources of information analysts may look at for malware is the built-in technologies, like the Windows advanced security log, or the system log of Windows. One useful host-based indicator found by analysts for investigating APT is Windows PowerShell. Although the host user may use PowerShell for automation, obfuscating the usage of PowerShell gives an indication of something more suspicious. A P15 remarks: *"Power shell usage, very sort of contemporary and favoured by the threat actors, APT-based threat actors."*

Investigating malware depends on what type of malware was detected. Certainly, in more advanced malware (e.g. APT), analysts have to rely on offline tools to determine what that malware is capable of. For example, when looking at worms, analysts need to limit the infection spread by identifying what other hosts it attempted to connect to. As P9 noted on investigating a worm:

> *P9: So we would work from the first point of compromise, and look at where that had talked to next, look at those for any indication of compromise, isolate and remove those from the network if there's a relative transaction, and then work out where they have spoken out to.*

Participant P15 noted that interacting directly with the host through remote desktop tools for hands-on live response is possible but used as as a last resort and not recommended. It is only possible when the SOC provides live response service to the customer.

6.2.4 Response

At some point during the investigation, the SOC practitioners need to make a decision on how to respond to a threat. For example, the analysts may need to *Pull-the-Plug*, and segregate the infected hosts from the network. Determining whether a SOC should respond by pulling the plug depends on what the encountered threat.

> *P9: So say I found, say there's a particular IP address or a particular domain that malware is known to communicate with. If I saw something of that nature, I would ring whoever was a responsible person near this machine, or whoever is responsible for it, and say "if possible, I would like you to remove that machine from the network".*

However, pulling the plug is not always the ideal solution to handle an incident. Specifically for APT, where disconnecting the host means revealing the detection to the adversary.

> *P15: So, if it's APT, you don't want to tip of the threat actor, right? You wouldn't want to disconnect, unless this disconnect - you don't want to move about it really quick, in a way that you may reveal yourself to the malicious threat actor right."*

When analysts determine that what they are seeing is, in fact, an APT, they remediate as soon as possible.

> *P15: When do we go for remediation right away? When we decide to actually call them and say, ok this is not the daily, common type of attack we see here, this is something more, this is way bigger, this is likely targeted"*

Pulling the plug is always a business decision that the customer needs to make themselves. The response decision depends on its impact on the business, and the pre-agreed escalation paths defined with the customer during the customer threat modelling. The MSSP SOC is responsible for monitoring and informing the customer of detected threats and the potential impact it has on the business. Hence, this decision is not determined by the SOC practitioner alone but through communications between SOC practitioner and customer decision-makers. It is then the customer's responsibility to make the decision.

> *P17: From our perspective, we're really monitoring, alerting and notifying customers. We don't have the authority to shut down communications or do anything interaction on the network.*

> *P19: We're the security team. We don't have the business decision to say, "stop". That's the business's choice. I can make a very strong representation to the business owner, to the senior risk owner and say, "if you do not do this, this is what's going to happen, choice is yours". It would be an interesting discussion.*

Pulling the plug sometimes has a major impact on the business. Therefore, this decision differs based on the customer and its tolerance to risk. Some device actions are more performance-costly than others. For example, as analyst P15 noted, blocking a communication costs more than permitting it. So costly decisions such as enabling packet capture are likely to be crucial decisions that one needs to think about again upfront. Determining the best action (response) depends on various variables, requiring human intelligence to make a decision.

Incident Response & Forensics— Finally, once the infected machines are pulled from the network, depending on the SOC setup, the investigation will be passed to the incident response or Forensics team, which is a team that might not be necessarily part of the SOC.

> *P6: Whilst we would not do maybe forensic investigation of a device, we have other teams, you know that we could pass that device to do forensic analysis on that device to ensure that it's not been compromised.*

The incident team will advise the customer on how to handle the incident and possible course of action.

> *P9: So it would then be, depending on the type of event, we would escalate out to our incident team, who can put somebody onsite, and we would reach out to the customer and say, "at the moment, we have advised you on this course of action, and these possible repercussions..*

Then based on the agreement with the customer, the forensics team may need to go onsite to capture evidence, isolate hosts, or take snapshots of machines for after-event analysis to determine the level of compromise that customers may need for insurance or evidence for prosecution.

> *P9: Once we have identified the potential impact to the customer, so that could be quite expensive, they may need somebody to go onsite and actually capture evidence of what has happened for their insurance, for any legal action that may come off the back of that.*

It is the incident response team's job to work with the customer on the possible actions they should take to remediate the threat. As P9 noted: "*so what we will tend to do when we take a customer on is build an incident plan that's specific for their*

industry or environment, so that we can give them the quickest course to isolating a problem with the minimum disruption to other aspects of the business." For example, if the detected threat was a worm, they inform the customer how it spreads, whether it is a known worm or a variant of a known worm, and whether it has spread in the customer's network. One possible action the incident team may recommend is isolating sections of the network that are likely to be infected. However, there are other possible actions that the customer can take that do not disrupt the business.

> *P9: So if we know it spreads by shares then we can make all the shares read-only for instance, so that people can still work, but they can't compromise the shares if they get infected.*

If an APT was detected, the incident response team needs to determine how far it is in the cyber kill chain, a term used to describe the series of stages of a cyber attack from the early reconnaissance stages to the exfiltration of data.

> *P15: And then we go ahead and say, well this is what we suggest needs to be done to evaluate if this is a single case, if this is just starting, or has much stronger foothold into your environment, and this is just being called now while doing a site investigation. This is very common that you investigate something and you see something else in the meantime that you need to follow up, or later.*

Some customers' immediate action in case of APT-compromise is to "restage". Therefore, each time they encounter an APT, they need to spend millions to restage the whole environment (e.g. server environment that might consist of several data centres). Therefore, it is best practice for MSSPs to advise their customers when making business decisions whether to proceed with deep forensic analysis, or whether to proceed directly to a restage. It is important for customers to estimate the magnitude of the attack's impact and the potential exposure of the compromised service before they make any decision on how to remediate. It depends on the customers' security maturity level.

> *P15: We have seen customers that get APT-based infection or compromise on a worldwide level, every year, and the business decision is "go ahead, restage right away", single individual alerts or threats or compromised hosts, without thinking about the bigger picture, which means that exercise needs to be repeated, that always leads to exfiltration of data, until that company is virtually ruined.*

Table 6.1: Factors Influencing SOC Operations that participants mentioned during their interview.

Influencing Factor	Sub-factor	Participants
Budget (Cost)		P2, P7, P16, P18, P21
Clearance	*Recruitment*	P18, P19
	Tool Use	P18
	Information Access	P6
	Incident Handling	P20
Size Organisation		P18, P9, P6
Change Management		P9, P6, P18, P19
Risk Appetite		P2, P3, P4, P11, P15
Maturity of Customer		P15, P2
Type of SOC	*Making Decisions*	P17, P18, P4
	Stress & fines	P6, P18, P4, p19
	Playbook	P5, P9, P19
	Access to logs	P2, P17, P6, P4
	Maintaining Awareness	P16
Communication	*Customer*	P9, P18
	Third parties	P6, P10, P18
	SOC members	P2, P6, P12, P18, P19
SOC Maturity	*Monitoring internal network traffic*	P2
	APT in Risk Assessment	P1
	Reliance on automation over humans	P1, P15, P20

6.3 Factors Influencing SOC Operations

In our interview analysis, we found that there are factors that influence the SOC operations, as summarised in Table 6.1. Here we discuss these factors, providing examples of how they may affect SOC practitioners' decisions and actions.

6.3.1 Security Tools Budget

Participating SOCs that serve public/government customers reported that the budget is a limitation, and thus their *cybersecurity capabilities* and the *SOC maturity* depends on the allocated budget. This introduces challenges in acquiring the latest technologies or changing existing, expensive monitoring solutions. P7 commented when asked about the main challenge his organisation (public) faces: "How much money have you got? Cost. IT security isn't cheap."

P21 also agreed: "*The cost of training to the tools and stuff, it all adds up and you don't get it.*" Similarly, Participant P18 remarks when asked about why they do not deploy a SIEM: *"We are a Public sector body who move very, very slowly, so when new developments and technology come along they don't necessarily get deployed straightaway."* Thus, analysts in that SOC need to access the tool

logs directly rather than have a SIEM to aggregate the logs, slowing down their detection and hunting abilities.

6.3.2 Security Clearance

Due to the sensitive nature of some customers, working in a SOC that serves such customer requires a *Security clearance*. This introduces many challenges to SOC operations, particularly in skill recruitment, tools used, information access, and how incidents are reported and handled.

Recruitment— Cybersecurity skills recruitment is a challenge [190], and requiring a security clearance makes it even more difficult.

> *P19: The biggest problems we've got, is a lack of people who are suitable qualified to use it. We just can't find enough. I have posts that have been vacant for 9 months, and I can't find people. We're constrained because everybody has to have a top-secret clearance. And there aren't that many people who have a top-secret clearance and are good at Arcsight.*

To solve the recruitment challenge, SOCs may train internal employees in cybersecurity, As P19 continues: *"but we're training people internally to do stuff, because I just can't find people. I think that's a universal problem. So, we're starting to train our own, which is the only way we can do things."* . Alternatively, they may hire contractors with security clearance as P19 mentioned: *"75% of my SOC are actually contractors, because I can't get permanents, so they're temporary staff, including my SOC manager."*

Tools used— Customers with vetting and security requirements affect which tools can be used on the network, such as *machine learning* tools. For example, P18 explained how the tools need to be accredited to be used on the customer's network.

> *P18: So, one of the problems we face is getting things accredited for use on our networks because of the security levels involved. So, I mean especially things like machine learning which are really cutting edge still at the moment but they're really the realms of academic stuff, so getting something like that approved for use on the network is probably several years away.*

Information Access— Obtaining access to a customer's network diagrams, which may be considered '*classified*' adds to the complexity of the practitioner's job.

> *P6: It depends on the security classification as well, because we're working in a security tier system [..], you could be an analyst but they are setting their diagrams, you will never ever see it, you will never ever see it. So, that adds complexity into them.*

Incident handling— Where customers require clearance levels, the amount of detail an analyst can document in the incident ticketing tool is affected. P20 described how all their incident investigations are treated as though there may be a criminal prosecution. This makes the incident handling process rigid. Consequently, analysts may need to write the incident details in a physical notebook for physical evidence in case of prosecution.

> *P20: We have a [...] fairly rigid incident process because of who we work for. [..], and there's set times for reporting and what sort of level you would report at, etc.*

Security clearance also affects how security practitioners learn from the reported incidents. Once they report an incident, they sometimes do not know the outcome as it is considered "*classified*".

6.3.3 Size of Organisation

The size of the monitored organisation influences threat monitoring. Monitoring an organisation that spans multiple geographic regions is a challenge. As P18 remarks:

> *P18: On our particular network you're talking about a global network. It would be impossible to try and keep track of what's going on everywhere all of the time. So, yeah, I think scale would be difficult on that front.*

Another challenge introduced as a result is storage. The bigger the size of the customer's network, the more logs it generates that need to be looked at.

> *P9: But it's quite a mammoth task, especially for bigger networks, it's a lot of data that you're going to have to store, and you might not review it for a week, it's a long time to store a lot of data.*

6.3.4 Change Management Process

Customers need to have adequate *Change Management* processes in place assist the SOC in filtering out false positives. As participant P9 remarks on reviewing the *benign triggers*:

> *P9: We may go back to the customer and go "well, a year ago you told us this was a fixed environment, but we're now seeing indicators that that traffic has changed, have you changed your environment and not followed through with your change process and let us know that you have had those changes."*

On the other hand, some customers are not usually aware of the change in their network or systems until the MSSP SOC asks them, as P9 remarks:

> *P9: So usually we will highlight something to the customer saying "this server has gone quiet, have you changed it?" And they will go "let's go and check" because we will engage with the security side of the business, and then they will come back on our ticket and go "yes, we've decommissioned that server", or "that server has now changed its function"*

Changes in the network need to be communicated to the SOC, as it effects, for example, the baselining of security tools and tuning thresholds, as we discussed in Section 6.2.1.

> *P9: Once the customer has confirmed why and what they have changed on there, and then once they have identified the type of asset, then we can remove any filters that may be applied, and then restart base-lining it with the customer.*

Similarly, P6 remarks:

> *P6: So, for instance, if you're going to deploy a new server, you have to have chain where different teams who should be aware that this work is going to happen. [..] ideally, things need to be done within the chain window or the organisational chain process, well documented, everybody is aware of what's happening.*

However, the main challenge in baselining is capturing the normal behaviour that may occur throughout the year, not only for a few days. As P9 noted,"*There are certain events that maybe don't happen every day, they maybe happen once a month or something, so you can't quite get them in the baselining period*". In addition, networks frequently change, such as the addition or removal of a server. Hence, it is important that the SOC is aware of the customers' *Change Management* process to obtain an updated baseline when a change occurs. As P9 noted on a customer deploying a new asset:

> *P9: Once the customer has confirmed why and what they have changed on there, and then once they have identified the type of asset, then we can remove any filters that may be applied, and then restart base-lining it with the customer.*

6.3.5 Customer's Risk Appetite

What assets the customer cares about, and the risk assessment is also a factor that impacts SOC operations, as discussed in Section 6.2.1. For example, one SOC emphasized the importance of maintaining the confidentiality of their data, and that it is their number one priority. In the case of ransomware, they have data backups, and the "*Availability*" is not a concern. However, if that SOC detects a possible data exfiltration, then analysts would immediately stop that by *pulling the plug*.

> *P15: So that data to us is really really important. So knowing what's going out every day, and knowing it's only going to places we let it go to, it critical. Once we start seeing it go somewhere we haven't permitted it to go to, then Arcsight will come into its own when it will be, "right, pull the plug, now".*

6.3.6 Maturity of the Customer

Deciding how to respond to a threat is a business decision. Taking the information provided by the SOC on the threat level and the business impact and making an informed decision depends on the customer's maturity level, as P15 remarks:

> *P15: So that is usually an ad-hoc business decision every time, and we are trying to get our customers to a mature level where this is a business decision, which means we give them the intelligence, the security intelligence, and our understanding of the threat level, our understanding of the risk level, of the risk factor, the risk assessment associated to the level of threat we perceive.*

The maturity of the customer's network topology and network (e.g. how the customer segregates the network) also has an impact on SOC operations. As P2 remarks:

> *P2: We have some customers who are absolutely brilliant, who segregate their users pretty well and everything goes over a proper router, so we will actually have logs. Some people have thousands of users sat in one subnet and you are not going to get any logs.*

In addition, whether the internal network is monitored or not affects the detection of threats that emerge internally, such as detecting worm propagation or insider threats. As P2 remarks:

> *P2: We have limited network taps on the internal, between the important segments of the network. It depends on the maturity of the business.*

6.3.7 Type of SOC

The type of SOC (internal or MSSP) affect how analysts make decisions, communicate with the customer, and whether they use a playbook when investigating an incident.

Making Decisions— MSSP's actions in handling an incident rely heavily on the customer's indicated preference during the "*Threat Modelling*" stage discussed in Section 6.2.1. As P17 explained, MSSPs monitor and alert the customer while it is the customer's job to decide how to handle the incident.

> *P17: From our perspective we're really monitoring, alerting and notifying customers. We don't have the authority to shut down communications or do anything interaction on the network.*

It is then up to the customer to make the decision on how they should respond. This is a challenge when faced with threats such as a worm that propagates in the network and requires a prompt response. Analysts need to contact the customer to pull a host from the network. In the event they are unable to get in touch with the customer, then they can only hope the worm does not spread. As participant P18 mentioned:

> *P18: There are divisions of responsibility, so you can make a recommendation as an analyst but it's actually up to somebody else to make the final decision and maybe even to rectify it if it's not something that's within our sphere of influence to actually correct.*

Monitoring for a customer adds pressure on the analysts that they need to raise every suspicious alarm to the customer. However, this introduces a dilemma about whether to raise every suspicion to the customer or only raise those that they are sure about, so that the provider does not look incompetent. As P4 remarks:

> *P4: Being as analysts, most of them are afraid to not raise them to the customer. They do tend to raise quite a lot of alarms to the customers, obviously to be on the safe side.*

Making a decision to inform the customer is also magnified when not providing a certain level of service results in fines. As P18 remarks:

> *P18: The security teams there are there to maintain availability above all else because it's when it's not available to the [customer] that you start seeing fines and things like that put in place. So, from a commercial side, availability is the most important".*

Using Playbook— Most SOCs have a manual called a "*playbook*", that details steps an analyst follow to deal with a security alarm. In addition to having the playbook, the MSSP SOC refers to the customer profile to determine how the customer wants them to respond to the threat. When monitoring multiple customers, MSSP SOCs rely on such documentation as the method of communication of each customer's requirements..

> *P9: So we have a playbook for internal events just like we have a playbook for external events, and the people that we'll interact with are primed in the same way, so they know what we will be telling them during an event, and the same for escalation paths as well.*

However, in in-house SOCs, such documentation may not be as critical, as the SOC analysts are all monitoring one network/customer. Hence, when the analysts detect an attack, they communicate with each other directly and their senior members on how to handle an incident. As noted by P19, this is much faster than looking it up in a playbook. However, P19 believes that a playbook is more needed in an MSSP SOC where they monitor for multiple customers.

> *P19: But it's much better, quicker, because our SOC is only the size of that bit of the room. [showed us - approx 8x6m] Tom [analyst] turns around and John[SOC manager] sat 3 foot away, and says "John, what do I do about this?"*

Access to Logs— The SIEM aggregates the logs from the multiple *data sources*, and processes them to detect threats. Although obtaining the data sources from an organisation for an in-house SOC might be feasible, an MSSP is limited by the data that the customer is willing to share. As a P2 noted:

> *P2: I think the limitation for us is always going to be we are relying on what the customer wants to send us. Because we are a managed security provider, we can only do so much. Trying to get people to give us data is hard.*

Similarly, P17 explained:

> *P17: So, for the most part it's about understanding the network topology, but we do try to get involved in their change processes so we can see when a new server has been added or if a server has been taken offline, as well as understanding what's configured on security enforcing devices such as firewalls or switches. We need to get a bit of information around that. It depends on how much the customer wants to share but the more they share the better a job we can do.*

Likewise, for patching, knowing which assets are vulnerable to which vulnerability assists in SOC operations.

> *P2: Vulnerability scans are great for us because it tells us unpatched systems. It will tell us if they are actually corporate compliant, so has somebody taken their laptop off and connected into a system they are not meant to? [..] But by having the patching logs we know straight away what software they are running and if they are vulnerable. It makes it quicker.*

Obtaining access to patching information or logs might be straightforward in internal SOCs. They may communicate directly to the patching team or have access to the patching logs. As participant P6 remarks:

> *P6: We have a team that keep on top of patching. So, if you're an analyst in something and you're unsure, you can easily speak to that team and they can give you an answer. Not just that, we also have tools,[..], you can log-in onto a system and you can identify what patch this device has or the device hasn't got".*

However, for MSSP SOCs, the patching logs are only available if the customer has performed vulnerability scans. As P4 remarks:

> *P4: If it's a vulnerability type alarm [..], the first thing that the analyst will do is it will go and check the asset to see if it's vulnerable to that. Now, the analyst will only know that if there's been a vulnerability scan performed against that asset.*

Thus, the SOC relies on the customers' patching team to provide it with this information. P4 continues

> *P4: So as a company we tend to ask customers to do a vulnerability scan at least once a week as a standard, because of WannaCry they actually want to do it every fortnight, every week, every night".*

If the detected security event is aimed at the vulnerable system, then the SOC categorises this as a Priority 1, raising the alarm to the customer.

Maintaining Awareness— The MSSP SOC analysts' ability to monitor multiple customer networks and systems has its advantages. For example, if a threat hit one customer, that threat knowledge is shared across other customer's environments to ensure the same threat is not present. Unfortunately, it also has drawbacks as analysts struggle to keep track of multiple customers, as P16 explained.

> *P16: When I used to do more of the analytics directly I just keep a view on what is going on roughly, so I skim through the alerts of each customer and figure out what sort of things were going on the last few days. When I was going off lunch after that point it got a bit more hairy, it got hard keeping track of that many customers at one go.*

6.3.8 Communication with the Customer

In case an analyst needs to get a "second opinion", they can brainstorm ideas and collaborate with other analysts. As explained by P9, the human ability of communication and collaboration need to be Incorporated in proposed SOC automation tools.

> *P9: The other beauty is a person can go "ah Bill, that looks odd doesn't it", and there's other people in there they can bounce those ideas off. So I think the Machine Learning, personally from that perspective, will be collaborative, people-driven quite a lot, and I guess over a period of time it will learn from the people that interact with it.*

Establishing effective communication channels with the customer is critical to SOC operations, whether it is an internal or an outsourced SOC. Not only do communication channels between the SOC or security team members need to be established, but also other departments in the organisation such as networking, IT, patching etc. As we discussed in Section 6.2.4, analysts rely on their knowledge of the monitored network to spot any anomalies. Thus, any changes in the network should be documented and communicated to all relevant parties. Participant P18 provided an example of how not communicating network changes led to wasting SOC team efforts.

> *P18: We were seeing a huge amount of ICMP traffic that we don't normally see and we have no idea why it suddenly started when it did and, actually, the networking team sit right next to us so you'd think they would be all over anything to do with ICMP, they have no idea. It turned out that they deployed a new security tool into the network and nobody had told us.*

For MSSP SOCs, some SOCs assign each customer an analyst (i.e. Technical Account Manager) as a point of contact. That analyst will be responsible for understanding the customers' network and assets. On the other hand, some SOCs find that having the customer exposed to all analysts provides ease of continuity and consistency for the customers. As P9 remarks:

> *P9: For a lot of our customers, it's quite a stressful experience when they're told that there's something you need to investigate that looks suspicious. By having exposure to all of the engineers, there's no sudden shock of dealing with somebody you don't know.*

Maintaining a record of the customer (customer profile) and updating it helps in sharing information of the customer to all MSSP SOC analysts. In case an alarm was raised for the customer, any analyst can login to that record, determine if it is a false positive or benign trigger and investigate the incident.

6.3.9 Third parties

Establishing proper communication channels with Internet Service Providers (ISPs) and other third parties ensures that the SOC has a point of contact not only to assist in blocking attacks but to be informed of any changes that may affect its operations. For example, when a customer is faced with a Distributed Denial of Service (DDoS) attack, the SOC might contact the customer's ISP to block the DDoS sources. As P10 discussed:

> *P10: It has to be on to the customer then to get in contact with our ISP and say look we need to block these particular IPs.*

6.3.10 SOC Maturity

The maturity of the SOC is another factor that influences the SOC operations [191]. The SOC maturity assessment determines the SOC's capability in detecting and responding to advanced threats.

> *P1: It comes to a maturity of the SOC. If you have APTs in your risk assessment or not, if you are not that mature to keep on, Advanced persistent threat like type of attacks, then you haven't probably got the capability or the tools or the human inputs to actively investigate those.*

In our interview analysis, security practitioners referred to SOC maturity when talking about the balance between technology and human. For example, threat hunting is found to be a critical human-centric functionality in SOCs. However, its capability across our participating SOCs ranges from a single analyst looking through logs to spot things, to having threat hunting exercises or even a dedicated team (i.e. purple team).

> *P20: Unfortunately I think the general maturity level of all SOCs in the world in still quite low, in that they rely heavily on skilled personnel spotting things that are odd rather than having a very good SIEM tool.*

Similarly, participants measure the SOC maturity based on the level of "*automation*". P18 referred to the SIEM they use as *very rudimentary*; thus they look directly at log events rather than taking it all into the SIEM and filtering it through there. Participant P1 remarks:

> *P1: It depends on the maturity of the SOC as well. I've seen a lot of SOCs that would depend completely on just tools giving them alerts and then digging through it. We've progressed from there, that we not only wait for the alert to generate, 30% of time is saved on automation that the systems have already done for us, we save that and use that 30% on active sort of threat hunting.*

6.4 Discussion

The aim of this study was to to provide the research community a better understanding of the inner workings of SOCs and the challenges that impact its operations. We summarise in Figure 6.1 the overall SOC workflow, and summarise the human-centric tasks in Figure 6.2.

Our findings highlight several tensions and contradictions that influence how SOC analysts prepare, detect, investigate, and respond to threats using technology. We summarise these findings below, per SOC process module, as broader lessons that researchers and security tool vendors can consider for future contributions in this field.

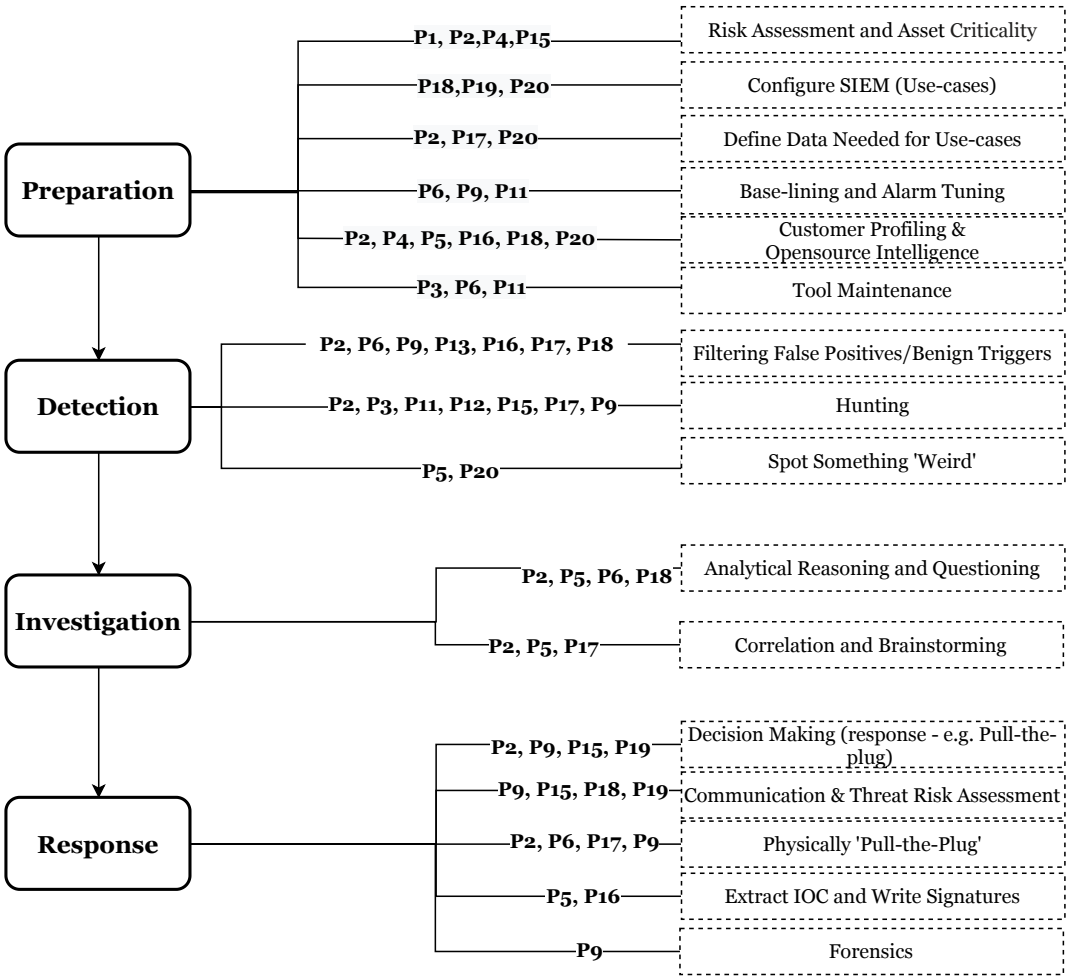

Figure 6.2: Human-centric processes in SOC Operations.

6.4.1 Preparation

Data Sources— As we discussed in Section 6.2.1, the outcomes of the Preparation process influences how the SOC monitors for and detects threats. Our analysis revealed that the outcome of this stage has an impact on the technology as well. For example, the data sources identified (e.g. logs) are fed into the technology, specifically the SIEM. Based on that data, SIEM correlation rules are written, and alarms are defined.

Identifying these data sources is reliant on understanding the customer's network and systems. Therefore, access to documentation of the customer's network topology, systems design, and usage is crucial. However, as reported by the analysts, customers are still documenting this information. In the absence of such data, analysts need to find other ways to obtain information required for analysis. P2 explained how patching reports could, for example, detail the operating system of a host. In

the absence of such reports, the analysts need to inspect the network traffic to get such information manually.

MSSP SOC practitioners hinted that they would want their customers to share more data. However, SOCs charge customers based on the amount of data they have; hence, customers may be reluctant to share much. However, this adds burden to SOC analysts to get that information through other means. As with the patching logs example, analysts may obtain such data through indirect methods (e.g. network traffic). Such practice delays the investigation and exhausts resources. This causes tensions as analysts may be evaluated by the number of tickets closed daily, and SOC bill customers by the number of triaged alarms.

Configuring Technology— Knowledge analysts gather during the Preparation stage is the first step of configuring security monitoring tools (e.g. SIEM/IDS). For example, the knowledge of the customer's network and "normal" traffic helps the analyst in tuning security monitoring tools parameters. Similarly, defining the SIEM use cases requires analysts to communicate with the customer to think of the possible scenarios. Analysts (humans) determine what threats to look for, through defining use-cases. Although technology is more effective than humans in detection scale, still, it requires humans to configure it. However, this dependency makes configuration prone to human error [14].

Customer Profile— During the preparation stage, SOC analysts document business information about the customer in the customer profile. Such information has proved very valuable for analysts, especially in prioritisation of alarms and determining the best remediation to deploy. On the other hand, other information that may be equally useful may not be included, such as change management process reports or patching logs. Such customer information, although important during the SOC process, is looked at adhoc and on request based on the incident in hand.

Escalation paths are documented in the customer profile, an essential practice to determine how the SOC and customer communicate when faced with an incident, preplanned to establish trust. In the case of outsourced SOCs, some SOCs assign a lead analyst to each customer. On the other hand, this may introduce a problem when the lead analyst is not present (e.g. on vacation). Hence, some MSSP SOCs instead choose to have all SOC analysts in communication with the customer to ensure all analysts are knowledgeable of the customer and its environment. Although such an arrangement is useful to avoid customer disturbance, it requires MSSP analysts to be knowledgeable of all customers' environments, causing pressure to keep such knowledge up to date and stay informed.

6.4.2 Detection

Technology vs. Human— One of the valuable characteristics of the deployed technology in SOCs is its ability to monitor and detect "known attacks" or "'known anomalies" at scale. However, although the technology may identify a possible threat, analysts need to review the alarms to determine if they are valid or a false positive. In general, analysts give priority to alarms produced by the SIEM that are either (1) built using a use case (designed by the analysts based on a specific scenario) or (2) is an alarm that is a correlation of multiple alerts from multiple security devices. As described by P15, analysts prioritize technologies that they respect and which have proven effective for a customer from previous experience. Nevertheless, reviewing such alarms to filter out false positives is still one of the analysts' most troublesome tasks. In doing so, analysts rely on their tacit knowledge and their knowledge of the monitored environment, built over experience and documentation of the customer profile.

The aforementioned detection processes are only dependable for known attacks. Still, the detection of more sophisticated or new attacks relies heavily on humans' capabilities. Indeed, our analysis revealed that technology is currently deployed in the SOC as a method to collect the billions of logs, identifying points of interest to narrow the threat search space. Possible threats are then conveyed to the analysts as alarms or visualizations. Then it depends on the humans (analysts) and their ability to correlate and link patterns observed over long periods — "chaining things together"(P9, P2). This is essential in detecting more stealthy attacks (e.g. APTs), where activities may be observed in a month's time.

For example, for hunting, technology is there as a way to organize logs and data in a structured format (e.g. Elastic Search) so that analysts using their analytical thinking to query such tools to make connections. Therefore, technology here is a means for humans to sort through data quickly to detect zero-days or APTs that may not trigger a network monitoring alarm.

Human-Technology Balance— Overall, the human-technology balance depends on the SOC maturity. The more established the SOC is, the more reliant they are on technology for triaging and filtering out false positives, having analysts' expertise in responding to incidents, and hunting for more advanced threats (e.g. APTs). Therefore, the first objective of automation in SOCs is to automate level-1 SOC analysts' tasks. On the other hand, in more advanced incidents, there is still a reliance on human intelligence that some analysts feel is difficult to replace (P9, P2, P15). As P15 described, "there is still no technology that can give security insights on its own".

Lack of Skills— There are many mentions of the lack of cybersecurity expertise [190]. Cybersecurity job descriptions require candidates of a technical background. Indeed, our SOC participants showed that a technical education (e.g. networking) is needed in their jobs. However, our analysis has revealed a contradiction in such assumptions. As described in Section 9.6, there is a reliance on humans to spot things that "don't look right" or "weird". However, such abilities are not necessarily present in "technical" analysts.

Interestingly, one participant who was a new hire and was able to detect an incident during their first weeks used to be an accountant. The event was detected by spotting deviating patterns in visualization. Another alarm was attributed to a misconfigured IP address. Both were missed by technical analysts and was spotted by the new hire. The new hire's "eye for detail", as described by their manager, was a valued ability in the SOC, especially level-one analysts. In fact, the manager was reluctant to provide technical training to the new hire, as such technical information might distract the analyst, effecting their valued ability. Nevertheless, technical knowledge is valuable in investigating triaged incidents (e.g. discrepancies in protocol usage). However, other skills, such as the ability to spot irregularities, are also valued for level-1 analysts.

6.4.3 Investigation

When describing the triaged incident investigation process, analysts revealed how they use technology to respond to queries emerging from their analytical thinking. Again, technology is a method to support humans in an investigation and making decisions rather than automation. Our analysis revealed a contradiction in whether analysts should follow their tacit knowledge or follow a set of guidelines defined in the playbook when investigating the incident at hand.

The investigation is a cognitive process, requiring humans to observe multiple data sources and make connections using their expertise and knowledge of the environment. Each person has their unique cognitive abilities of imagination, association, and pattern matching, all abilities needed in such investigative tasks. Therefore, enforcing analysts to follow procedures defined in a playbook may do more harm than good by limiting their intuitive ability. As reported by P2, such procedures represent a burden for them, and therefore, they rely on their own methods.

Malware Classification— In certain scenarios where the incident involves malware, determining the type of malware impacts the decision (P7, P9). For example, ransomware and worms spread in the network requiring immediate

disconnection. On the other hand, in the case of APTs, disconnecting an asset may signal APT perpetrators. Hence, knowing the type of malware is crucial in such investigations. However, depending on the Service Level Agreement (SLA), SOCs may not have access to their customers' machines and rely only on the data at hand. Obtaining the malware executable to run malware sandbox may not be feasible. If at all feasible, then getting approval and access to the asset take time, delaying the rapid response such attacks require.

6.4.4 Response

Reporting an Incident— Once an incident is verified, SOC analysts need to report this detection to the customer. Our analysis revealed a tension in MSSP SOC analysts reporting. To avoid appearing incompetent, analysts are under pressure to ensure that the incident reported is indeed an incident and not a FP. For example, one analyst reported that they spend time investigating a possible malware infection before notifying the customer. However, delays in dealing with malware might have huge repercussions, especially if its a ransomware or a worm. On the other hand, MSSP SOCs may be susceptible to fines if an incident is not handled on time under the SLA.

Response Decision— Once an incident is triaged, the SOC needs to decide on the most suitable response. In such decisions, the type of SOC profoundly impacts who makes the judgment. For MSSP SOCs, the response is not determined by the analysts themselves but is based on the pre-agreed response documented in the customer's profile. As the analysts describe it, they are only there to monitor and detect threats, and the response is a business decision. The SOC will communicate the threat and its risks to the customer, and they make the decision. However, not all customers are at a maturity level to make such a decision. Therefore, some MSSP SOCs might advise their customers to get them to a business maturity to make such decisions. In addition, specific response actions (e.g. Pull-the-Plug) might have a higher cost than others, and therefore, decisions need to be scrutinized based upon how it will impact the business. Similarly, in internal SOCs, decisions are made by people up the hierarchy.

Response Automation— Once a decision is made (e.g. Pull-the-Plug), the type of SOC impacts how the response action is administered. Analysts in internal SOC inform whoever is in charge of the impacted asset to disconnect the asset from the network. On the other hand, MSSP analysts advise the customer of the best course of action, which is then executed by the customer's security team. This is mainly a manual process and involves communication between the SOC

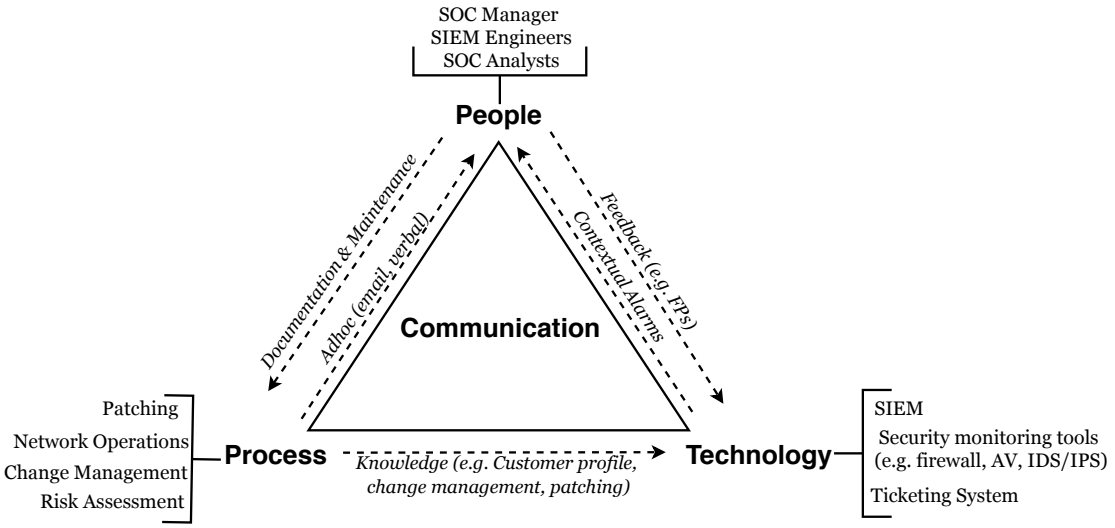

Figure 6.3: Communication and Contextual Knowledge Flow in SOCs

members and other teams (internal) or SOC and the customer's team (MSSP). One participant noted that automation of such a task was seen performed by one customer (described as scripts) to respond to the incident, but that is still deemed rare. However, an automated decision may not be as effective as human involvement. For example, when deciding on the response action for a malware that spreads, its implication on business disruption needs to be considered. Hence, as described by P9, an action might be to make file shares read-only to ensure that data necessary for business continuity is available but still prevent malware from spreading. Such an orchestrated response currently needs to be thought out by humans.

6.5 Summary

Research on SOCs and their operation are still untapped due to the sensitive nature of the job. This is a problem for the security research community, as most security technologies are used mainly by analysts in the SOC. In this chapter, we aimed to address this gap by describing the SOC workflow, obtained by interviewing a wide variety of SOCs. From our study, we found that the SOC is still heavily human-oriented, and they would benefit from automation tools that assist the analysts in their decision making. We hope that by understanding the inner working of the SOC, researchers can propose solutions that automate aspects of the monitoring, detection, and response processes.

Technology has proven to be more useful in detecting threats at scale with the human direction of what is a threat (e.g. use-cases, signatures). Human involvement is then centered more on filtering out the FPs. FP filtering requires the presence of

knowledge accumulated in the customer profile and existing processes (e.g. change management) that are currently not incorporated into technology. This transparency in knowledge is challenged in environments that require security clearance and acquire confidentiality levels. Our findings revealed a precondition that needs to be dealt with before attempting automation. We found that communication and transparency are critical in SOCs operations.

Previous work highlighted the need for communication between analysts and managers as well between analysts and other organisation teams [12]. However, communication and transparency not only need to be between people but also embedded between all SOC elements: People, Process, and Technology, as we summarise in Figure 6.3, and discuss in the following.

Process-Technology— For example, the process of change management and patching needed to communicate its outcome to technology (e.g. SIEM) as input to be considered when designing SIEM use-cases to reduce FPs. Similarly, embedding patching reports to technology will provide transparency on which asset is vulnerable, to identify possible immediate threats. Incorporation of the output of the various customer's processes (e.g. risk assessments, change management, patching, HR records etc.) into the technology can result in generating more context-aware alarms.

People-Process— Of course, such process outcomes are not useful unless people keep it up to date, communicating changes into the process (e.g. change management process). As we showed in Figure 6.2, customer profiling is a very human-intensive task; removing such a load from analysts using automation can release them to work on threat hunting. Currently, such communication between process and people (e.g. the result of vulnerability scans) is adhoc, done verbally or via email [129].

People-Technology— One of the most human-centric tasks in the SOC is FP filtering. Analysts providing feedback to the technology on the reported alarms and whether they are false positives can facilitate future FP filtering automation.

As we summarised in Figure 6.2, analysts are overwhelmed with mundane tasks. Automation is needed to automate such tasks to provide analysts with time to work on more interesting and career rewarding tasks (e.g. hunting). Moreover, we found that contextual knowledge is important and needs to be incorporated in the produced alarms. In the following chapter, we continue our qualitative study by focusing on tool usage by analysts, identifying advantages and disadvantages analysts perceive from these tools, discussing how contextual knowledge can be incorporated in these tools and provide recommendations for future tool development.

7

Malware Detection Tools in Security Operations Centres

Contents

7.1 Introduction

As we discussed in the previous chapter, SOC analysts use several tools in their day-to-day tasks. For example, one of the leading tools used in SOCs are SIEMs. Alarms generated by SIEMs rely on humans to review them, as most of these alarms are FPs. For example, NIDS have been found to produce 99% FPs [20]).

Previous research found that false positives can overload security practitioners [10]. Consequently, academia has been focusing on improving network-monitoring systems, specifically addressing the challenge of false-positives (e.g. [192–194]). However, Kokulu et al. [12] have found that SOC security practitioners do not consider false-positives in automatic malicious activity detection as a significant issue in SOC operations. Such findings contradict academia's current understanding.

Similarly, in Chapter 5, we found that although analysts find the number of alerts generated by most IDS to be overwhelming, the results were neutral on the IDS's inadequacy in detecting threats. We aim to explore this further to determine analysts' perspectives on the strengths and weaknesses of these tools and the reasons why such tools that produce high false positives are not seen as flawed by analysts.

Our survey results show that knowledge of the network is vital in detecting anomalous behaviour on the network. In fact, analysts choose the alerts they process based on that awareness. We explore this further to investigate how such contextual knowledge can be incorporated into tools to reduce the number of false positives. To summarise, in this chapter, we address the following research questions.

- What do analysts' perceive to be false positives, and how can we establish a clearer definition?
- What are the advantages and disadvantages that security practitioners perceive of network-monitoring tools?
- How can we design better network-monitoring tools to improve the process of filtering out false positives?

7.2 The Perception of False-Positives

To understand the analysts' definition of a *false positive*, we asked our participating practitioners to discuss how they will investigate a particular incident to elicit their reasoning. In their response, they indicated that the first step when receiving an alarm is to determine if the alarm is a which they will need to filter out.

Filtering out False Positives— Once an alarm is picked up by an analyst, the analyst needs to determine if it needs to be escalated for investigation. However, a huge amount of alarms is triggered in a SOC daily, where the majority of these alarms are considered FPs.

> *P2: The SIEM in itself is too noisy. We know 99% of the alarms we generate false positives, but we still have to look at them.*

Interestingly, when we asked our participants about false positives during the interviews, P9, P4, P15 asked us to clarify if we mean a *false alarm* or a *benign trigger/noise*. In fact, one analyst emphasized the importance of distinguishing between *benign trigger* and *false positives* when evaluating a security monitoring tool performance and its accuracy. False alarms are used to describe an alarm being generated without a true security-related event (*the boy who cried wolf*). In contrast, benign triggers are true alarms; meaning they match an existing signature (e.g. vulnerable java version) but the organisation chooses to ignore it (e.g. due to legacy systems). P9 provided an example of a benign trigger due to outdated *Java* versions.

> *P9: There's Snort signatures, we have particularly noisy — well I say they're noisy, they're always doing exactly what they should do, they're always identifying vulnerable versions of Java, but a lot of companies have a lot of vulnerable versions of Java, so we get a massive influx of it.*

Similarly, when we asked P4 if he finds the detections reported by network-based security tools to be "accurate", he explained:

> *P4: So they are very accurate I believe, but whether they're accurate and a false positive, now, it depends on how you perceive a false positive. So if we raise an alarm to the customer, and they're aware of it, they will just call out, we are aware of it, but it's a false positive in that sense*

Hence, an IDS detecting benign triggers does not mean it is not accurate. It is accurate because as it is the same trigger a malicious actor can utilise. Hence, labelling such benign triggers as *false positives* gives the impression that the technology itself is flawed when it is detecting what it was designed to detect.

> *P15: So, you have a very high number of events right, and it doesn't mean they're a false positive, but it means many enough are again triggered by benign triggers. Which means the condition is perfectly matched, and the filter as such works, but the circumstances are completely legitimate. This is benign, because the purpose is business-justified, and it's not malicious, it's just manifests in the same way as particular malware would, right.*

A high percentage of the alarms generated by security tools are found to be benign triggers. Although analysts may have reported a high number of false positives in our survey, our qualitative study revealed that many of these false positives are attributed to benign triggers – hence tools are doing what they are designed to do. This explains why analysts reported, as discussed in Chapter 5, that they are overwhelmed by alarms but were neutral about the adequacy of IDS.

> *P15: So the benign trigger probability is usually very high, from hands-on experience. So is the volume itself, across the IPS/IDS vendors.*

7.3 Intrusion Detection Systems

Signature-based intrusion detection tools have their challenges, yet, they are still widely used in SOCs — as reported by 90% of our participants (Chapter 5-Figure 5.3). For example, participant P2 noted: *"IDS traffic, quite important to us, so we rely quite heavily on IDS alerts."* In order to understand this contradiction, we obtained the analysts' perspectives on the strengths and weaknesses of the traditional network monitoring tools (e.g. NIDS) in the SOC, discussed in the following.

7.3.1 Strengths

Good first indicator— Some analysts found IDS Alerts to be a good first indicator for an intrusion. As P4 explained "Most of the anomalies are basically network traffic because that's the first point of call for any sort of intrusion". Similarly, P2 reported:

> *P2: So IDS analysis and network traffic for us performs very well. It is a very good early indicator that something is going on.*

Deal with high volumes— One of the reasons why IDS is very popular is its capability to handle and cope with high data rates (high volume of traffic), as we discussed in Chapter 5.

> *P15: If we look at the volume itself, the top talkers are without any doubt the IPS/IDS appliances. And that is expected, given the relative detection uncertainty that they apply to implement detection as such, like regular expressions and anomaly-based detection right.*

Well Maintained— Having the tool be well maintained is also a sought after tool property. For example, updating the tool with signatures of the latest threats.

> *P4: In terms of the good parts of the tool, I believe Suricata or Snort [signature-based NIDS], because it's pretty well maintained and there are new rules every day added to it, I think it's probably one of the best tools for network intrusion detection systems.*

Easy to Write Signatures— There are three types of signatures deployed in a NIDS: (1) signatures released by the product vendor, (2) signatures shared by the community or other organisations, (3) user-defined or use-case-based signatures (e.g. admin logins, TOR). Some analysts think that one of the good features of intrusion detection systems is the capability of defining custom signatures — User Defined (UD) signatures. Every organisation is different in its networks, systems, users, and its use. Defining custom configurations that work for that environment is a sought after trait.

> *P9: In terms of the good features they have capabilities for you to deploy your own customised signatures, you know, UD, you know, user-defined ones.*

Creating custom signatures also assists analysts in providing a quick fix to a new threat. For example, in case of new malware, analysts can write signatures to detect it based on the malware known Indicators of compromise (IOC). As P9 pointed out in the WannaCry case:

> *P9: So like WannaCry, instantly, we wrote as many signatures for that as possible, for everything we could, and then put it in place. There were already signatures there, but we just wanted to be as thorough as possible.*

Work Across Protocols— P18 and P6 also commented that the ability for the signature-based IDS to work across protocols is an advantage.

> *P18: I suppose the good thing about the IPS is that it can work across protocols.*

7.3.2 Weaknesses

Mostly False Alarms– A high number of alarms generated in the SOC are attributed to NIDS [20]. As P4 explained: *"We do tend to get a lot of alarms based on the network traffic, and 90% of the time they are based on the Suricata rules."* Similarly P6 explained: *"I think that most of the tools will generate a higher volume of false positives, especially when it comes to a signature-based event."*

However, some analysts find that these alerts are mostly benign triggers rather than attributing to a malicious threat.

> *P15: We had many detections that are accurate, as much as the filter mechanics is concerned, but I can't recall to be honest, a genuine malicious detection, attributable to TippingPoint or any other IPS as a detecting device. And this is concerning, given that at the same time they represent most of the volume of the SOC, when you come to think of it.*

Some analysts perceive alerts generated by IDS/IPS tools to be unreliable on their own. They use it as an additional data source that detected the same intrusion, correlating it with alerts generated by other platforms. Although P2 indicated that IDS is a good first indicator, he later explained that it is not good on its own—*"So I think network monitoring tools on their own aren't the greatest of indicators."*

Unreliable Signatures— The IDS accuracy is highly dependent on how the filters/rules/signatures are written. For example, in the case of malware signatures, signatures that incorporate IP addresses as a filter are not reliable, as malware is prone to changing its domain.

> *P2: A lot of the network-based stuff I don't find very reliable, just because a lot of the IOCs are based on IP addresses which change. They are based on domains which get shut down.*

Likewise, P18 explained that a poorly written signature will result in many alarms, that analysts can not review, rendering the signature not "useful".

> *P18: Yeah, it's kind of an internal battle of having a signature for the latest threats and having a useful signature because if it's just constantly firing, nobody's got the time to review all of them, so it has to be well written and they're not always.*

Black Box— One of the disadvantages of IDS/IPS is that they are closed systems. Meaning that the analyst receives an alert but does not know the reason for the detection. The analysts themselves need to lookup more information about the alarm using, for example, the filter name to understand what the alarm is for.

> *P15: You know the filter name and you have the filter description, and this may or may not be tied to a CVE number, for a particular vulnerability, which you can then read up, see what operating system, software, etc., it is related to.*

Likewise, P4 explained that alarms produced are not clear and require analysts to search further for the source of the alarm.

> *P4: I think the fact, I think your last question about the anomaly-based, although it gives us some sort of indication, it's not very clear and It does require us to dig in deeper and find out where the source was.*

However, such ambiguity not only requires more effort for the analyst to investigate, as P15 explained, they also lose their trust in the validity of the alarm.

> *P15: So that's one issue of traditional IPS/IDS, that we also don't really have high respect for, because if you don't tell me the reason you fire, you're not ready to open up the reason you produce an event for me, and I have only a very high level of description that may not be usable, it's not usually usable, let alone actionable, then I can't have really high respect on that.*

Loosely Written— Some analysts reported that signatures are "loosely written", meaning that it is written to be general to capture a variety of activities, thus resulting in a high volume of false positives.

> *P6: I think that most of the tools will generate a higher volume of false positives [..], especially when it comes to a signature-based event. We have IPSs and we get signatures from the vendor. Some of the signatures are written very loosely in order to allow it to capture a wide range of activities and those are some of the downfalls of using it*

This is also true for signatures that are designed to detect malware. As the malware evolves and generates various attack behaviours, signatures can be written to be too broad to capture all these possible behaviours.

> *P18: Malware can change so quickly now that, yeah, a lot of their signatures can be very loose and very broad to try and capture every possibility.*

Similarly, when creating malware signatures, they are written to capture part of the malware activities. Although, the alarm is triggered, the analysts are left confused about which kind of malware triggered it. P4 explained:

> *P4: But what happens is, sometimes when they write the rules, they only write to capture a part of the malware and not all, not other pieces of it. So what happens when the rule triggers and we have an alarm, it may give a false indication that that alarm is for this particular malware, when in reality, it's for a different one. And that could be, that is an issue because as malware perform different, sort of, actions on a system, we don't really know what sort of remediation advice to give to a customer. Whether it's for that malware or whether it's for this.*

One analyst provided an example of how such loose signatures results in the tool generating a false alarm. Such an alarm was determined as a false alarm on examining the network traffic.

> *P9: But some of the signatures are very poor, so they will look for the words "select" and "from", in clear text, in a packet, but it could be "from" and "select" in a packet, there's no context applied, or "drop tables" a classic one, so we have a customer who sells a [Drop Table], I'm assuming it's a fold-down side table, every time they sell one of them it fires an alert, but it fires it on the SQL injection alarm, so you can't turn that signature off, as poor as it is, because that's one of the SQL injection alarms that's part of that product set for.*

However, such a manual inspection of the packet might not be possible with encrypted traffic, as P9 continues.

> *P9: which is quite time-wasting really, but then if the limitation is, so if that traffic is then encrypted, you might get the alarm but you can't see the raw text.*

Lack Context— Some analysts remarked that one of the reasons that IDS alerts are not reliable is that it does not perform *contextual analysis*. Meaning, that the alert lacks context that might help the analysts determine if its a true alarm.

> *P15: It's a bit of a paradox there, you have quite a bit of an output, at the same time, when the detection is accurate, and the regular expression is correct and everything is fine, it doesn't really mean it's malicious, and the reason is, it doesn't do usually contextual analysis, it doesn't add more event sources into it, which is what then the SIEM does, and the analysis of the human element does, of the human actor.*

Therefore, providing some context to the generated alarms assist analysts in determining whether the alarm was generated due to an actual threat. P17 explained how current signatures do not provide enough context to determine if it was triggered due to an actual threat or a benign trigger.

> *P17: Because without it you're pretty blind because if you see the content that a signature wants to match on plus a few bytes after, that's not always the best way of actually, well, you can't really confirm whether something bad has happened or not.*

For example, some network intrusion detection systems do not capture the packets that triggered the filter. This may be due to storage limitations.

> *P9: Some of the devices will be configured to give you the trigger packet, so the one packet that contains the string that triggered the alarm. In order to kind of contextualise some of these trigger packets, it would be useful to have the packets before it, the packets after it, so you can understand what was going on in the bigger picture. Where we do have that, analysis is just made loads easier. Where we don't have that, you've got to start making assumptions, which is where a weakness could lie.*

However, some analysts find that more products are now adding more context to their alarms, hence are *getting better* as P9 continues to explain.

> *P9: So, things are getting better with that, Cisco have sort of got rid of a lot of that technology and moved on with the SourceFire and FirePower stuff, so that's more contextualised but yeah it can be a real challenge.*

Detecting New Threats— In general, most IDS systems used by SOCs are signature-based, as P16 remarks:

> *P6: From the network monitoring its all still signature-based, so naturally if there is no signature it will not cause an alarm.*

Therfore, if there is no signature for a threat, a signature-based IDS would not be able to detect it. This is an issue when several malware variants are being introduced each day. Moreover, in more sophisticated attacks such as APTs, signature-based tools will not generate alerts.

> *P2: Yeah, so with a lot of the APT malware you don't notice it on the way in. Snort signatures aren't written for it, just because it is not very well known. So a lot of the time it will get through.*

Slow in Deploying Signatures— One of signature-based solution's advantages is that it allows analysts to create signatures quickly to detect new threats. However, deploying these signatures to all IDS/IPS on the network takes time.

> *P18: So, with something like WannaCry when it first hit on the Friday afternoon, we didn't have a signature for it. So, that was something we needed to develop as quickly as possible and then, of course, again on a huge network it takes time to deploy all this stuff.*

7.4 SIEMs

As we discussed in Chapter 5, 80% of our participants use SIEMs. SIEMs replace the need for analysts to access the traditional security tools, sometimes called eye-on-glass, directly.

> *P15: This is where it is getting interesting, because these days, 2017, very few I guess security vendors use directly a network monitoring tool. Most of them feed into SIEMs, like Arcsight, QRadar, LogRhythm, you name it.*

Similar to traditional network-monitoring tools, SIEMs also have their strengths and weaknesses, which we identify in the following sections.

7.4.1 Strengths

Visibility — SIEMs are capable of handling large numbers of security events from multiple sources [195].

> *P2: So any one source on its own, I don't find that useful. That is why most SOCS now, the core bit of technology is a SIEM, because you can tie all the different technologies in together.*

Hence, the SIEM can see all logs and alerts generated by security devices you connect to it, as P2 explained: "Yeah, so we have a SIEM platform that ingests logs from all sorts of stuff." Everything goes into this one central repository, so all the logs and alerts are in one place. In addition, the data sources are monitored continuously to make sure they are working. Thus, if the log data source becomes silent for some time, the SOC receives a warning saying, for example, "You haven't received an IDS alert for 20 minutes." This functionality allows analysts to have visibility of the monitored network and systems.

Analysts having access to host and network logs is attractive, as the more information, they can view the better their situational awareness [195].

> *P12: The good thing is we have recordings of all the networks and all the emails and everything, so all the files and directories that any system user will have so if there's any malware detected or any malicious mail received by them we all have the same tool, all logs are collected via SIEM, a so in that way it's good that we have visibility to all the systems via SIEM.*

Custom use-cases — Each environment customer's risk assessment and assets are different. Customers might want to write SIEM alarms for specific assets that match a certain threat scenario. The functionalities provided by the SIEM allow analysts to write such use-cases, tailoring alarms to their environments. As P19 explained:

> *P19: So they will tell arcsight, "I want to see how many people have logged onto so-and-so in the last 24 hours", and it goes "dzzzz" back.*

For example, defining a use-case and the required logs to monitor that use-case will reduce the size of the acquired logs to the bare minimum. This makes storing them more feasible as P2 noted: "*You can store it that bit longer because you don't get too many of them.*"

Moreover, defining use-cases and the bare-minimum logs needed ensures that analysts are not swamped with far too much information. P20 noted:

> *P20: I mean a SIEM tool can do pretty much anything you want it to, as long as you can see how you can boil that down to basic logic, and where you're going to get that information from. As long as you can do that, then the SIEM tool can do absolutely anything.*

P20 explained how an analyst could write a use-case that integrates the physical security system into the SIEM. In such a use-case, the SOC can detect log-ins originating from users who are currently offsite. P20: "*a log-on by someone who has left the organisation is a particularly good example of how it can work well.*" However, such detection is only possible if the SIEM data (e.g. employee leavers) is up-to-date and maintained.

Cross-event, Cross-Platform Correlation — One of the main advantages of SIEMs over traditional IDS is its capability of correlating multiple alerts and log events generated by heterogeneous platforms.

> *P16: With the network threat it all goes Scericata (or snort of whatever), where I think if the, it cant do like comparing 2 events at the same time, is just an IDS really, it can't put the 2 together. If it was going to the SIEM platform that would be easy, because that is what it does.*

For example, as described by P2, a correlation rule could be "*If you have seen an IP doing any one of these activities, followed by authentication success or a large amount of data transfer to it, generate an alarm.*"

Correlation Rules/Directive alarms could be alarms that monitor the network for "anomalies". For example, one SOC had correlation rules for noticing spikes in traffic. These rules learn normal traffic patterns over two weeks by looking at host traffic, creating upper thresholds. Thus when traffic for a host goes beyond that threshold (*threshold determined during the baselining phase in preparation*), an alarm is generated.

> *P2: It's a bit of both. You signature it, you write the rule and then you baseline it from there, so it's a learning rule you would call it. So you are basically saying learn what activity is there for several days and then monitor what changes for the next however many hours.*

SOCs usually have a fairly extensive set of correlated rules, which the analyst is expected to react to immediately. The analyst find these rules effective: P2 explained "*So yeah we do correlation rules like that. It proved quite effective, just because you are generating that event yourself.*"

> *P9: So, bit of traffic bouncing off your firewall is not super-interesting, a host sending a bit more traffic than it normally does, you know peaks and troughs in traffic do change, but join those two together within your SIEM, now that's suddenly become more interesting because that's been persistently hit externally and now suddenly more things are going out, so that's a type of anomaly that that kind of kit would spot.*

There are several reasons why a SOC may define directive alarms. Defining correlation rules or directive alarms allows the SOC to store these alarms for an extended period so that correlating these alarms over time might allow capturing more stealthy attacks such as Advanced Persistent Threats (APTs).

> *P6: The analysts that you've got run in 24-hour shifts, session, so if something is missed during the day, they still have to run historical, maybe do trend analysis, you know, where you analyse, maybe, a 24-hour period log against the previous 24 hours. So, chances are, you'll be able to pick something that was missed by whoever was on shift before you came on and rectified and those logs will still be there.*

Categorization and Data organisation— Logs/messages received from SDEE enabled devices are intrinsically suited to go into the SIEM. They do not require manipulation because they are in the right format. However, some applications/devices were never designed to generate logs. and their output has to be edited before its import to the SIEM, a process called Normalization. The SIEM's ability to take such unstructured data and organize it in a structured manner makes it easier for analysts to query the SIEM, defining filters to retrieve meaningful results for the investigation in hand.

> *P9: I think how the SIEM platform is configured to receive that information and then deal with it accordingly helps, so it's a lot easier for manual analysis, when things are structured properly or categorised properly..*

Prioritization — Analysts receive multiple alarms at once. How they choose the alarms they investigate first depends on several factors. As we found in our quantitative study reported in Chapter 5, 57.9% of the analysts indicated that they chose the alerts based on awareness of normal network activity, and 47.4% said based on the alert severity rating. As P2, explained, the priority score assigned to an alarm will depend on the criticality of the alert, and the impacted host and its location in the network.

> *P2: So the SIEM platform itself will assign a risk based priority score to the alarm that it generates. These scores are based on the criticality of the host, where it sits in the network, the criticality of the alert. So if it is a strong signature which is only seen as information it assigns a lower score. Basically the computation in the background assigns a score to it, and obviously the higher ones we jump on as quick as possible.*

Fast MTTR— One of the metrics used to measure a SOC's maturity is the time needed to take action and neutralize the threat is Mean Time to Respond (MTTR). SOCs aim to lower the MTTR, and the SIEM's ability to aggregate datasource helps to achieve that. However, some customers may not have a SIEM, and analysts have reported that they would need to go onsite and look at the host. Specifically, they look for historical evidence of how the infected machine interacted with the network. To determine that, they would examine the file shares that can access it, local logs stored on the infected host, switch logs in switches, and firewalls that it may have passed through.

Nevertheless, the best option is always to have such information feed into a SIEM. As a participant noted, going onsite takes time and delays the investigation.

> *P9: Even with the best will in the world, you might take an hour, two hours to get somebody to somebody's site, if those logs are available for inspection within the SIEM, you're going to be investigating within minutes.*

7.4.2 Weaknesses

Overwhelm Analysts— The SIEM's ability to plug-in data sources, although attractive for comprehensive monitoring, might result in a huge collection of data and alarms that overwhelm the analysts. Hence, when an analyst wants to find particular information, it might not be a straightforward process.

> *P17: Downsides may be sometimes too much information. If you just want to quickly jump to something there's usually no shortcut to just finding the answer, you have to go through certain steps to find what you're looking for.*

Therefore, it is best practice when deploying SIEMs is to properly design the use-cases, and collect data needed only for these cases, as noted by P20, P11, and P3.

> *P20: It's just a case of getting the bare minimum that you need to meet your security use-case.*

Use of structure databases— The volume of data becomes even more problematic when the deployed SIEM uses structured databases that take time to retrieve queries.

> *P13: When handling large amounts of data in terms of alarms, it's not got quite as much resource to handle it, cause at the moment the platform manager uses the Microsoft SQL, so when handling big data, structure database doesn't really work for it. In terms of just taking in data, it can handle it cause it uses a ElasticSearch and backhand, but the actual alarms are still structured.*

False Alarms and benign triggers filtering— Similar to traditional security monitoring tools (e.g. IDS), SIEMS may also generate noise or many false positives, especially when first deployed. The SIEM is only as good as its configuration. Therefore, it also needs to go through a baselining period to reduce noise and benign triggers.

> *P9: I think it would be good to get some of the noise away that might be generated within there, although one of the big tasks that come through the SOC is the cost of baselining, taking the noise out of the equation.*

> *P2: The SIEM in itself is too noisy. We know 99 percent of the alarms we generate false positives, but we still have to look at them.*

Cost— One of the main challenges of SIEMs is cost. A SIEM platform with hundreds of connectors and archival databases may require $3 to $5 million of upfront hardware cost as well as a yearly maintenance cost [195]. Moreover, adding additional functionalities as "plug-ins" will accumulate an additional cost.

> *P12: Pattern discovery, yeah, so you have to buy this stuff separately and it will do all your IOCs together, its so smart but in other tools like Splunk we don't have that pattern discovery. But we have to do it manually.*

Configuration hassle and lack of expertise— SIEMs require a lot of configurations from security practitioners, such as defining thresholds and use-cases, as we discussed in Chapter 6. In addition, the lack of expertise to use and configure the SIEM has an impact on using the SIEM to its full potential.

> *P19: The biggest problems we've got, is a lack of people who are suitable qualified to use it.*

Detection of Zero-Day attacks— Although a SIEM provides additional functionalities over traditional monitoring tools, it still relies on humans to be able to detect more advanced attacks.

> *P20: They [SIEM] are good at detecting anomalies, but if you're clever enough to make sure that you don't look like an anomaly then the SIEM is not going to do you any good, then you're reliant on an analyst saying – that's fairly normal but it just looks a little bit odd. And then you're reliant on the human being making a connection.*

7.5 Machine Learning-based Tools

Recently, the security community have proposed technologies that apply machine learning for network malware detection [36, 72–105]. In our interviews, we investigated the state of machine learning tool deployment in SOCs.

For example, P15 explained how, in addition to SIEMs, analysts also have respect for APT-based tools (i.e. tools that use machine learning).

> *P15: So, it's about cross-platform, cross-event source correlation, that we can actually have high respect for, not any technology as such, unless of course it's APT-based technology. Contemporary APT-based technologies are a family of its own that have by default very high respect for.*

As P15 continued to explain, compared to traditional IPS/IDS, such APT tools (e.g. Carbon Black, FireEye) do behavioural analysis of files and artifacts of network traffic.

> *P15: So that's the sort of behaviour-based approach that you don't have a-priori built-in conception of what to look for, but you see what it does, machine learning,*

Furthermore, P1 explained that they are planning to deploy a tool that applies ML to network analysis in the next year.

> *P1: When it comes to the network side, we are in the process to sort of look for machine learning activities in there. But right now it's just based on signatures.*

P1 continued to explain that ML could be applied to detect file manipulation on the host to automatically block it.

> *P1: I really like to, you know, it's, in a perfect world it will be, all the processes would be known by my machine learning tool or the AI tool.*

P18 explained how machine learning is being used to automate FP filtering, one of the most human-centric tasks as we found in Chapter 6. Such automation functionality is often referred to as "orchestration" [196].

> *P18: Well, I was at InfoSec yesterday and it seems that a lot of the big vendors think it can be. They were very heavily pushing their tools that do away with all of the false positives for you, or they're supposed to, to the point where it was even kind of raising a ticket for you and things like this. I can't remember what they called it. Orchestration.*

P6 explained that although they do not deploy machine-learning products, they employ an Defence in Depth approach.

> *P6: We do have a bit of defence in depth approach, so not just using one particular tool to do all our monitoring. [..] So, coming back to your question, yes, we may not have machine learning, but we have a range of technology in place to mitigate a number of threats.*

P17 explained that they attend conferences and vendor demos to keep up to date on the latest technologies. However, they have not seen commercial products that stood out, and they were skeptical of the tools' application of machine learning.

> *P17: When I've looked at machine learning in the past it seems like it's not entirely machine learning and they say they've got sensors deployed which have got algorithms that get updated by the vendor, but it really sounds like they're updating some form of signatures on there.*

P9 also mentioned the absence of machine learning-based technologies in their SOC. However, they believe that it would be good to use ML to for the human intensive task of baselining and so reduce noise, as we found in Chapter 6.

> *P9: I think it would be good to get some of the noise away that might be generated within there, although one of the big tasks that comes through the SOC is the cost of baselining, taking the noise out of the equation.*

As we found in Chapter 6, the type of customer (e.g. government) is a factor that impacts SOC operations. When the customer is a public sector, it might be a challenge to deploy machine-learning technologies as it requires approval for deployment, which takes time.

> *P18: So, one of the problems we face is getting things accredited for use on our networks because of the security levels involved. So, I mean especially things like machine learning which are really cutting edge still at the moment but they're really the realms of academic stuff, so getting something like that approved for use on the network is probably several years away.*

7.6 Challenges in Network-based Monitoring Tools

Our analysis revealed challenges in the adoption of network-based monitoring related to encryption and the position of the tools on the perimeter of the network.

7.6.1 Encrypted Network Communications

One of the findings of our survey (Chapter 5) is that 70% of our participants find packet content to be a very important feature for threat monitoring. However, organisations have moved to encrypt their network traffic to maintain its confidentiality. Similarly, threat actors are employing obfuscation through traffic encryption, posing a challenge for network monitoring solutions.

> *P1: There are quite a lot of things that you just can't tell from network. You just can't rely on the networking data because of encryption, SSL, HTTPS. We are going, we are moving, I think, in time or a period where we want everything to be encrypted. But there again, attackers can do the same.*

Consequently, due to network encryption, security practitioners have moved to adopt host-based monitoring tools. As P15 explained:

> *P15: And truth told, the battleground of security has moved from the network to the host, and the reason is simple – the level of obfuscation usually, and the fact that the traffic is ciphered, the traffic is encrypted by the malicious actors, by the threat actors.*

When the deployed tools perform content inspection, encryption renders them obsolete, as P9 explained.

> *P9: If the traffic is encrypted, we won't be able to do that [content inspection], unfortunately, but provided it's not, yes it will take full packet captures and then it will run the Snort signature base through all the packet captures, and see what it can match on. So yes, I would call it like content inspection, yes.*

In fact, analysts find that host-based tools provide more useful information and are more accurate. As P17 explained, host-based solutions provide insight into all of the processes and activities occurring on the end host, while maintaining the host user's privacy. Similarly, P2 explained:

> *P2: The most accurate ones are normally the host agent ones, so we are looking for a sequence of things. So a change to a registry key followed by an application starting with a parent process which is normal, followed by network traffic out to a suspicious network block and stuff like that.*

7.6.2 Internal Network Traffic Monitoring

One main question we were interested in is whether SOCs monitor the internal network traffic. Most SOCs agreed that they only monitor network traffic on the organisation's perimeter, and rarely collect internal network traffic. Unfortunately, the lack of monitoring of the internal network is a security weakness - it is easy to infect the network using a malware-infected USB or through phishing emails. Once the malware is in the network, it can propagate internally, causing damage.

> *P9: I know from what you hear in the industry, a lot of people put a very very large focus on protecting themselves around the perimeter, but then, say you do manage to get through that, a lot of networks they don't spend half as much time securing the innards of it all, so once you're actually in, a lot of the time it's very easy to just propagate throughout these networks and gain a foothold*

SOC practitioners agree on the importance of monitoring internal network traffic, especially with the rise of insiders and malware such as WannaCry. Moreover, monitoring workstation's internal network traffic is also important during forensics to determine how malware got into the network. For example, in the case of Wannacry, although it infiltrated the perimeter, the means used (i.e. email or drive-by-download) is not known.

> *P9: I think that's where people start wising up, because things like the WannaCry, it got into people's networks, but it also entered at the perimeter, but what people are struggling to find at the moment is they can't find whether they got it via an email, and somebody clicked on something they shouldn't, or whether they had a public-facing share and that's how it got into their network. And there's quite a lot of companies that just don't know yet [...] But because people aren't monitoring at that level on the inside, that bit is a little bit of a grey area for a lot of companies at the moment.*

Likewise, in cases such as detecting insider threats, monitoring the internal network is crucial. When we asked a participant what they would need to detect an insider, he mentioned that they would need network traffic either as a full capture or in NetFlow format.

> *P17: So, you'd either need full packet archive or NetFlow and knowledge of what the internal subnets look like. So, we collect the info on what internal subnets are used for what. So, a workstation would be LAN specific department network and where the servers live and we could use NetFlow to detect how much traffic is being transmitted between two.*

There are several reasons why they do not monitor internal traffic. For example, P4 mentioned that it is hard to monitor internal traffic because of geographically how the network topology has been configured. So if the SOC monitors the internal part of the customer's firewall, then when host A tries to connect to host B, and they are on the same network, that traffic will not be captured because it will never hit the firewall.

Another main challenge is storage. Capturing and storing full packet captures of the ingress and egress points to the network alone is costly. Thus, to additionally collect internal network traffic in even greater volume is a challenge.

> *P9: And again, that's where you get into vast amounts of storage, if you're going to capture all the packets of all of that internal interaction, at the perimeter you've got a few devices, and then it's whoever lands at the perimeter, talking outside, but there's lots and lots of traffic happening all the time on the inside, that if you were trying to capture that's quite a vast amount of data to sort of look at.*

Similarly, an MSSP SOC, collecting PCAPs depends on how much the customer is willing to pay for storage. Although they cover the customer perimeter network connections, monitoring internal traffic is still costly for the customer.

> *P2: We cover all ingress and egress into networks, but it is how much the customer wants to pay for, for us to cover within the network. Most of it at the end of the day comes down to storage and how many logs we can process a second for them. There is always a cost and they are always trying to save money, and it's normal, "Well as long as we've got the internal ... in and out covered then internal shouldn't matter." It is a very alternate way of thinking. We don't like it but ... yeah.*

Internal network traffic may still be visible in the SIEM, but that depends on whether use-cases that the customer defined rely on part of the internal network communications as a data source. For example, the customer might be interested in monitoring the internal network traffic of specific servers. As such, the SOC defines use-cases that monitor that part of the network. As P2 mentioned:

> *P2: So we will see, well all Windows servers, so SMB anywhere. So we will see any connection via the windows system, security log on that one. So we should see, "Oh that's the authentication."*

Some SOCs may attempt to monitor the internal network behaviour through the host logs. However, when defining such use cases, baselining what is considered a normal activity for an user asset is a challenge.

> *P17: At the moment we don't have any alerts on that kind of traffic because it's hard to really determine for that side what's normal really because sometimes people log on to servers, SharePoint or something, download a lot of files because of working on some big documents, so that would be quite challenging to baseline, but we definitely use that during an investigation. So, then whilst we're looking through the traffic for a host we'll see the logs related to that host so if they've connect to a bunch of servers which they don't usually connect to then we can flag that as well.*

PCAPs are useful in understanding "*what they are dealing with*", but are usually only examined after a SIEM alarm detected malicious activity. Some SOCs collect the PCAPs but only look at it during the investigation process.

> *P9: So you can go through, review the traffic manually, it's an immense job, you wouldn't want to review all of the traffic manually, without having some sort of intelligence to point you to a particular thing. But that's one way, and I suppose it works the same in Splunk or any kind of big data platform - you can manually go and review it, but with the quantities of data and stuff in there, it's going to be very difficult unless you have got something to point you in the right direction*

Some SOCs monitor the internal network through IPS. However, the size and scale of the network make it difficult to push signatures to all IPS. Such as in the WannaCry attack, as P18 discussed in Section 7.3.2.

To overcome storage challenges, some SOCs are attempting to collect internal traffic communications in alternative formats such as network flow (e.g. NetFlow). When asked to explain the reasons they do not collect internal network traffic, P17 replied:

> *P17: Storage, I think. We're looking at getting alternatives in the, not full PCAP data but to get something like NetFlow and we captured that from some of the switches on our customer's estates, so at least we can see what hosts we're communicating with, what and how much traffic was transmitted as an alternative to getting full blown, full packet capture at that point.*

NetFlow vs. PCAP

One of the main questions we were interested in is at what stage of the process is network traffic analysis useful. Network traffic can be captured as a packet capture (PCAP) or in a NetFlow format. As we have shown in Chapter 5, only 55% or our participants monitor network packet captures and NetFlow through the SIEM. However, some analysts find NetFlow to be only helpful in forensics

and so these are not actively monitored (20%). In addition, 80% use Wireshark to analyse network traffic.

> *P15: NetFlow is not so much popular, although it is to be found for some forensics purposes, so for ad-hoc specific tasks, but this doesn't usually feed into security-event management systems.*

In contrast, some analysts have configured the SIEM to collect PCAPs of the network when it receives an alarm from a network-based IDS. Although the intrusion was picked up by the NID, investigating the alarm and determining why that alarm was fired was done though the PCAP.

> *P4: If it's a NID-based alarm so through Suricata or Snort, then AlienVault automatically performs a capture, a PCAP capture, so we can download the PCAP and look at why that alarm was triggered, and then looking at the PCAP we can understand what is because it matches certain criteria which that that rule was looking for ..*

As we discussed in Section 7.3, one of the weaknesses of signature-based IDS is the lack of context in the generated alarms. However, as P9 explained, PCAPs can help provide context to a received alert, providing explanation to why the alarm was fired.

> *P9: Whereas there's other bits of kit, and I suppose it's not down to the vendor, it's more down to the configuration, some of them aren't configured to take full packet captures, and in certain instances you really struggle to apply context to the alert you have seen. So yes, I would say it varies, but I think they do their job ultimately.*

Although some SOCs might attempt to collect the customers' network traffic, they may keep them for a day and a half to three days. After that, they will store the metadata, which will go back a couple of months enabling them to perform retrospective searching if required. Similar to what we discussed in Chapter 6, some SOCs only start collecting the PCAP after they start investigating an incident. So they will take a PCAP of the traffic for 120 sec, then wait for a couple of minutes before collecting another one.

7.7 Towards Contextual Alarms

One of the main findings of our survey, reported in Chapter 5, is the importance of knowledge of the network in filtering out false alarms. Such knowledge provides *context* that is helpful to analysts in investigating an encountered alarm. Knowledge of the customer and its network allows SOC practitioners to use their *tacit knowledge* and experience in spotting threats.

> *P15: OK so getting to know the customer, customizing this experience over months and years, that bit of abstraction that actually also allows you as tacit knowledge to make judgment calls, why something looks a bit off from what it should be.*

For example, one of the main weaknesses reported by the analysts in the tools they use, discussed in Section 7.3.2, is the lack of context in the signature description, leaving them to determine the cause of the alarm, adding more effort to a very tiresome task. We discuss in the following how adding context to alarms can help analysts quickly determine its genuineness.

To be able to determine if a network activity is "normal", the analyst needs to know the monitored network. This knowledge spans, for example, the network topology, network devices, what these devices are used for, location, and the owner of these devices.

> *P18: I suppose it comes down to knowing your network again, is as much about knowing who looks after what and what's actually going on in the organisation at the time.*

For example, the analyst might need to know the connection patterns of each asset—i.e. what services are running on this asset and what are their frequent connections.

> *P4: Again, the topology of where the server is based is important, if it was a server based in the DMZ then I would expect some sort of traffic, you know? If this was a server on the internal side of the network, then I'd be more cautious.*

Before analysts start looking at logs, they need first to have an awareness of what they are looking at. So they go back to the customer profile to determine, for example, *what is the server for the customer? Is it a domain control or a web server? Is there a new server being tested?* The abnormal activity can be an indication that the server changed or a different IP address was reassigned to the server. Such changes, as well as benign triggers, asset usage, and other information about the network, are recorded in the customer profile (i.e. wiki). The analyst may also contact the server administrator to validate abnormal activities or check historically in the logs to see if there was any similar connection seen previously and at what times.

Knowledge of the asset and network helps practitioners eliminate false positives and determine the path of investigation. For example, knowing an asset is a windows machine will assist when receiving an alarm for a Linux signature for that machine.

> *P6: To be able to make a decision as to whether an event or an incident is a false positive or not, it comes down to understanding environment, so predominately if there is an event that alerted in a certain IP and that device is maybe a Windows machine, but a signature relates to maybe a Linux that, you know, you'd clearly say, "Oh, hang on, this is not right. This is a windows machine, but this is alerting us."*

SIEM use-cases need to be carefully designed by analysts. Analysts' knowledge of the customer network and environment assist in defining SIEM use-cases and security tool thresholds.

> *P20: The SIEM tool is only as good as you program it, so you still need that knowledge only on, because most of the correlated events are going to be specific to your environment.*

Not only do analysts need to know the network they are defending, they also need to be aware of the customer's business. Knowledge of the customer is critical in many aspects in the SOC operations, from ruling out false positives to making a decision on how to respond to a threat.

> *P4: So it's not just about what you see in the security events, or in the tools, it's about what you know of the customer and the customer's nature, you know, business nature.*

Similarly, *"Hunting"* requires the analyst to have knowledge of the network as well as the customer. When analysts hunt objectively, they may spot patterns that look abnormal to them based on their knowledge of the network and the customer that might not be detected by security tools. For example, seeing a policy on a customer's firewall that accepts everything but blocks connections to port 80 and port 443 might be odd. However, knowing that the customer has a third-party trust relationship with someone else clarifies the policy.

> *P15: So you hunt this week and you hunt next month, and you see if anything changed that you believe it shouldn't have, and there are aspects of knowing the network, of knowing the company afterwards.*

Knowledge of the customer and the nature of its business helps analysts in investigating an incident. One participant provided an example of how valuable it could be for determining a false positive. On detecting large outgoing connection from the customer to another company, after spending time investigating, they researched if there was any business between these two companies. They found out that their customer acquired that company recently. Similarly, P16 remarked on how knowing the customers' working hours helped in filtering a false positive.

> *P16: I use open-source intelligence to build up a view of the customer first, [..]. So I'll figure out like if they are in Dubai or wherever that they might not be doing the same working hours [..], because we have been caught out by that one before, there is no traffic on a Friday. Oh yeah.*

Analysts might acquire knowledge from third parties, such as ISP, security vendors, and the security community to investigate threats. For example, the ISP might inform the customer of any changes in the network they need to be aware of. However, that information might not reach the SOC, especially an outsourced SOC.

> *P6: You've got [ISP] and sometimes you've got to talk to each other, but it becomes difficult because, for instance, [ISP] would be doing some configuration change on setting a limit for the network. They might tell somebody within [customer], but that information may not necessarily come to you.*

Security vendors are responsible for maintaining their products and providing signatures for new threats. P6 discussed how they rely on security vendors to share that knowledge, pushing out any new signatures and any new indicators of compromise (IOC).

> *P6: Something like WannaCry, for example, the vendor also has the responsibility because they have to maintain activity of their system so when it happened, straightaway McAfee, for instance, was on the case developing signatures ready for it to be deployed.*

The security community can also assist the SOC process through intelligence sharing, writing blogs about how a threat behave, and spreads. As we found in our survey, 35% of our participants triage alarms according to newly announced vulnerabilities in security blogs. The SOC practitioners search online for information about a threat they encounter to determine the best countermeasure to deploy.

> *P2: So WannaCry was great, because straight away people were publishing vlogs, this is how it spreads. This is what you need to look for, and you could go and look for it before there were any signatures for it. We were finding things quite quickly.*

7.8 Discussion: Lessons Learned

Security tools deployed in SOCs have their weaknesses and strengths, as summarised in Figure 7.1. We discuss the lessons learned from this study providing recommendations for further research.

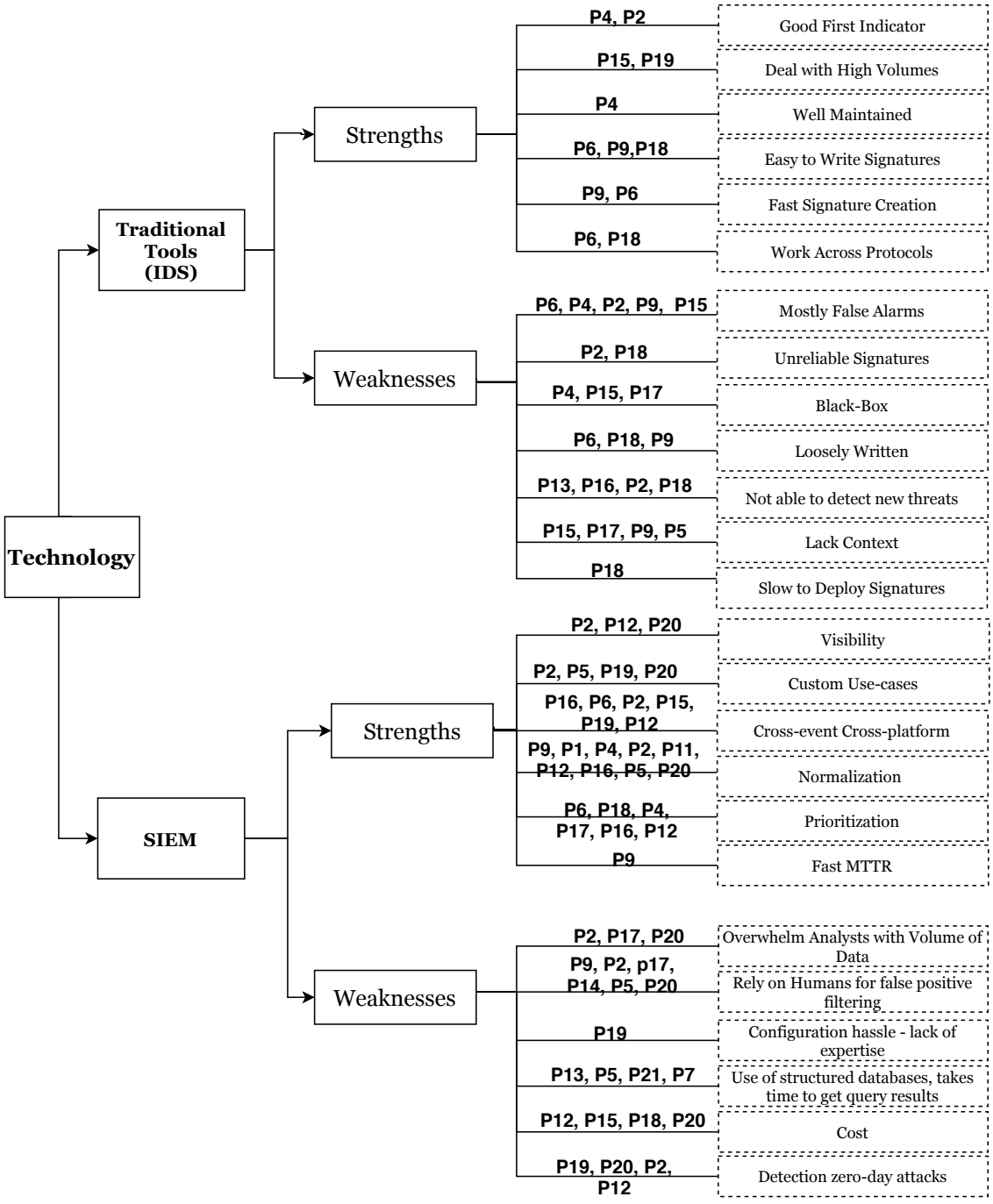

Figure 7.1: Summary of SOC tools weaknesses and strengths, and the participants agreeing with each factor.

7.8.1 Human intelligence vs. Automation

As we found in Chapter 6, many of the SOC operations are human-centric requiring humans' to either configure tools, identify patterns of behaviour or investigate alarms. One of the weaknesses perceived by our participants for both IDS and SIEMs is its reliance on humans for false-positive filtering and configuration, as reported in

Section 7.3.2 and Section 7.4.2. This is due to humans' ability to recognise patterns, imagination, as well as the social ability of communication as discussed in Chapter 6.

Research is still far from integrating human cognitive and social abilities into SOC solutions. One emerging field to attempt to tackle this challenge is "Cognitive Security". Cognitive security leverages multiple forms of AI, including machine-learning and deep-learning networks to uncover the human cognitive ability. Probably one of the leading research in this area is IBM Watson[1]. Further research is needed in developing human-machine cognition solutions in SOCs where humans and machines work together rather than using full automation.

7.8.2 Contextual Alarms

There is no doubt that the knowledge obtained by experience as well as knowing the customer and its network helps analysts perform their job, as discussed in Section 7.7. For example, a SOC tool that monitors for abnormal network activity should consider the customer's work hours to eliminate benign triggers such as no traffic on a Friday (Friday is a weekend in some Middle-Eastern countries). Such context is currently incorporated into technology to some extent. However, as identified in Section 7.4.2 challenges such as the use of structured storage, where the volume of data results in delays in query retrieval, limit its implementation.

To overcome such limitations, one research opportunity is in investigating how logs and other context data sources can be represented as knowledge graphs. In such graphs *knowledge* about the customer, network, and knowledge from external entities can be incorporated to provide more context to alarms. Hence, contextual data can be incorporated in the graph to produce contextual alarms. Moreover, such a structure can be used by analysts during the false alarm filtering process or even hunting.

Representing host logs as graphs to benefit SOC operations is a research area that has recently emerged, such as [164, 197]. One of the main strengths of signature-based IDS is its capability of creating signatures for new threats in a short time. A compelling use-case for future research is to study whether signatures of such systems can be represented as graphs to allow better maintenance, fast retrieval, and search. Such graphs can also be contextualised by incorporating the customer's benign triggers to produce more reliable alarms.

[1] https://www.ibm.com/watson

7.8.3 Internal Network Monitoring

Network-monitoring is built on the assumption that threat is observed at the network perimeter; an assumption is no longer valid in modern networks. In Section 7.6.2, we found that organisations still do not monitor the internal network traffic, relying only on the defined SIEM use-cases or decreasing traffic collection to network flow formats (e.g. NetFlow).

However, as the analysts agreed, this is a weakness, especially with more sophisticated malware that uses zero-days (e.g. Duqu 2.0, WannaCry). However, one of the main obstacles to internal network monitoring is cost. As the scale of networks grows, the cost of monitoring and collecting the logs and traffic grows exponentially. The SOC is already overwhelmed with the amount of data generating alarms as it is that monitoring the internal traffic is not possible. Instead of deploying solutions to find *the needle in the haystack*, alternatively, we can reduce the haystack.

The SIEM is currently set up to collect all possible data, feed it into the SIEM, and rely on the SIEM engine to produce the alarms. As P20 pointed, it is critical to collect only the data that is needed for the use-cases and not just collect everything. In such a setup, only use-case required data is fed into the SIEM to produce the alarms. Once an alarm is generated, then other logs or traffic needed by the analyst are collected. In fact, this is an opportunity for automation; when the SIEM fires an alarm, it automatically collects the "evidence" that the analyst will need for the investigation. Recently, commercial security organisations have looked into incorporating similar concepts, called orchestration [196] —known as Security Orchestration, Automation, and Response (SOAR) commercially.

Moreover, most SOCs have a playbook that details how the SOC should respond to an alarm, usually written by knowledge experts. This provides an opportunity for researchers to understand these processes and evaluate the possibility of automating these procedures. In addition, these playbook responses can be combined with the customer profile to investigate how the response to an incident varies based on the customer business and risk appetite.

7.8.4 Customised Machine Learning Models

One of the strengths of current SOC tools reported by analysts is customisation. Specifically, user-defined signatures in IDS and use-cases in SIEMs. Such customisation capabilities allow analysts to create signatures and alarms to fit the monitored environment. One of the main observations arising in this study is that organisations' network traffic and asset usage are different, as are the networks

and systems governing them. We identified in Chapter 6 the factors that impact SOC operations, and thus tool usage. Accordingly, although SOCs may deploy the same technologies (e.g. SIEM), these technologies are configured and used differently and should not be developed as *one size fits all*.

Machine Learning (ML) tools, although currently not used by our participants, will be deployed in the near future. The success of such ML technologies relies on their capacity to adapt the model to the monitored environment. For example, our analysts stressed the importance of adding context to the alarms. Such contextual knowledge could be incorporated in the ML models to produce more effective alarms, reducing the number of benign triggers. In addition, incorporating the analysts "feedback" on the generated alarms to the model itself can strengthen its future predictions and its alarm prioritisation. In case of benign triggers, the model can then learn that the prediction made, although true, is not suitable for the organisation in hand, adjusting the model accordingly.

7.8.5 Technology Evaluation Metrics

One of the main findings reported in Section 7.2, is that both vendors and the research community should use *false alarms* or *benign triggers* as an alternative metric to false positives, which is more general and vague. Specifically, when evaluating the performance of a system deployed in a real-world setting, using the term *false positive* gives the impression that the technology itself is fundamentally flawed. Hence, when analysts' report a 99% false positive, this may not be due to the performance of the tool itself but due to the customer's system and network configurations generating benign triggers that analysts ignore.

7.8.6 Compatibility with Existing Technologies

The main factors impacting a SOC's adoption of technology is cost, hassle of technology configurations, and lack of expertise, as we discussed in Section 7.4.2. Security technologies are expensive, and it takes an organisation time to acquire the latest technologies and develop the SOC's maturity. Hence, any newly proposed techniques should be compatible with existing tools to reduce any additional costs, configuration effort and the analysts' learning curve.

7.8.7 Malware Detection Systems

Encryption — As organisations and malware (e.g. zbot, Trickbot) encrypt their network connection, SOCs have relied more on trusted host-based monitoring solutions, as discussed in Section 7.6.1. To foster real-world adoption, network-traffic monitoring needs to be content-agnostic using header-only features to detect malware while preserving the organisation's network communication privacy. In fact, previous research efforts have proposed malware network monitoring solutions that are encryption resilient, such as BotFinder [84], Disclosure [79], and BotMiner [81].

Malware Generic Behaviour — Malware usually has the same structure; it communicates with a C&C and performs the same malicious activities (e.g. DDoS), as we described in Chapter 2.

As we identified in Section 7.3.2 and Section 7.4.2, one of the weaknesses of existing technology is its inability to detect unknown malware or malware that uses zero-days. This is because the detection functionality is tailored to identify specific malware types. Instead, malware detection methods can focus on detecting generic malware behaviours, making it capable of discovering new malware. However, as P4 explained, signatures, rules, or behavioural features used for detection should not be "loosley written". Hence, in addition to detecting the malware activity (e.g. DDoS), it should be able to reliably attribute that to a malware family.

7.9 Summary

In this chapter, we focused on understanding how SOC practitioners define false positives, a term that is vague and general. We found that a clear distinction needs to be made between false alarms and benign triggers when evaluating a SOC tool.

We also investigated the tools used by analysts, identifying their strengths and weaknesses. We discovered that these tools produce alarms that lack the context needed by analysts to filter false alarms from genuine ones. We identified the knowledge sources that can support such contextual alarms and challenges in deploying them.

Our interviews revealed research opportunities for improving SOC technologies, which we summarised as lessons learned. One of the lessons derived are requirements for network-based malware detection systems. Firstly, due to encryption, network-based technologies need to be content-agnostic, relying on network header-level features for detection. Moreover, such tools should be able to detect generic malware activities and adapt to malware evolution. Furthermore, malware generated traffic

should also be attributed to a specific malware family, as found in Chapter 6, to assist analysts in deploying the most effective remediation.

Based on the aforementioned requirements identified from our qualitative study, we propose malware detection and malware classification systems, outlined in the subsequent chapters.

8

Malware Family Classification Using Network Flow Sequence Behaviour

Contents

8.1 Introduction

We discussed in Chapter 6 the process analysts follow when investigating malware. Although MSSP SOCs monitor the client's systems and networks for signs of malicious activity, they may not have direct access to the client's hosts due to privacy or policy reasons. When malware-infected hosts are the source of the detected malicious traffic, the analysts need to negotiate access to the infected host with the client. Even if access is possible, the time required to negotiate this and running the executable in a sand-box for classification is not efficient, particularly in situations where rapid detection and response is critical.

As identified in Chapter 6.4.3, the incident response procedure for ransomware is different from a APT, and being able to determine the family can speed up the remediation process. Therefore, analysts need *on-the-wire* malware classification systems capable of matching detected malicious network traffic to a malware family, thus not requiring to access the infected host. Moreover, as we found in Chapter 7, with the increase in the use of encryption by malware for obfuscation and by hosts for privacy (e.g. HTTPS), network-based monitoring has become challenging.

Considering the aforementioned requirements in mind, we propose *MalClassifier*, a system for the automatic extraction of network behavioural characteristics for malware family classification. As malware use some form of network communication to propagate or contact their command and control (C&C) servers [17], *MalClassifier* derives the network behavioural characteristics for a malware family, abstracting the malware family's network behaviour to a flow sequence profile. *MalClassifier* is effective in classifying malicious network flow sequences to a malware family on-the-wire, thus negating the need for access and sand-box execution of the malware binary.

MalClassifier uses non-identifiable network traffic features derived from network connection logs for training the classifiers. Such features are visible even when network connections are encrypted, making it resilient to adversary obfuscation through encryption. This also makes the data required to train the classifiers accessible and easier to share than full PCAP traces due to privacy concerns. In designing *MalClassifier*, we make our approach IP-agnostic, as sophisticated malware, such as exploit kits, apply dynamic DNS and Domain Generation Algorithms (DGA) to change their communications destination [198].

Text mining approaches such as n-grams have been successfully applied to malware family classification [100, 101]. Similar to [101], we apply n-grams to sort network flows into groups of n consecutive flows (i.e. n-flows). Such a representation provides granularity to the extracted malware behaviour. For example, a single

failed SMTP flow might be benign, but multiple consecutive ones exhibit the behaviour of a bot sending spam. *MalClassifier* mines n-flows that are distinctive for each malware family. Such distinctive n-flows are used as features to train a supervised model capable of classifying unseen n-flows to the malware family. The models learn re-occurring network flow patterns that capture the characteristics of a malware family's network behaviour.

The contributions of this chapter are three-fold:

- We propose a novel fuzzy flow sequence similarity measure, that calculates the *value* similarity of two flow sequences.
- We propose an order sequence similarity measure robust against malware evasion through flow sequence manipulation.
- We develop a prototype of *MalClassifier*, and evaluate its performance using a dataset of malicious network traffic, achieving more than 95% F-measure for malware family classification.

8.2 MalClassifier System Design

Malware exhibit diverse and complex network traffic behaviour. Yet, malware variants of the same family have been known to have common behavioural patterns reflecting their origin and purpose [56, 82]. In our system, we aim to exploit these shared patterns, particularly their network behaviour, for malware family classification.

MalClassifier can be deployed by security analysts to understand and classify the behaviour of a malicious executable by observing its network flows when access to that executable itself is not possible. It maps malicious network flow sequences (n-flows) to a malware family by comparing these sequences to previously observed malware activity. In general, *MalClassifier* operates in three phases as illustrated in Figure 8.1.

1. *Pre-processing and Sub-Sequence Extraction:* Network traffic of malware variants of malware family y is input into Zeek for network flow reassembly. The assembled flows are then encoded to a *textual sequence*. The sequence is then divided to sub-sequences (n-flows) of length n.

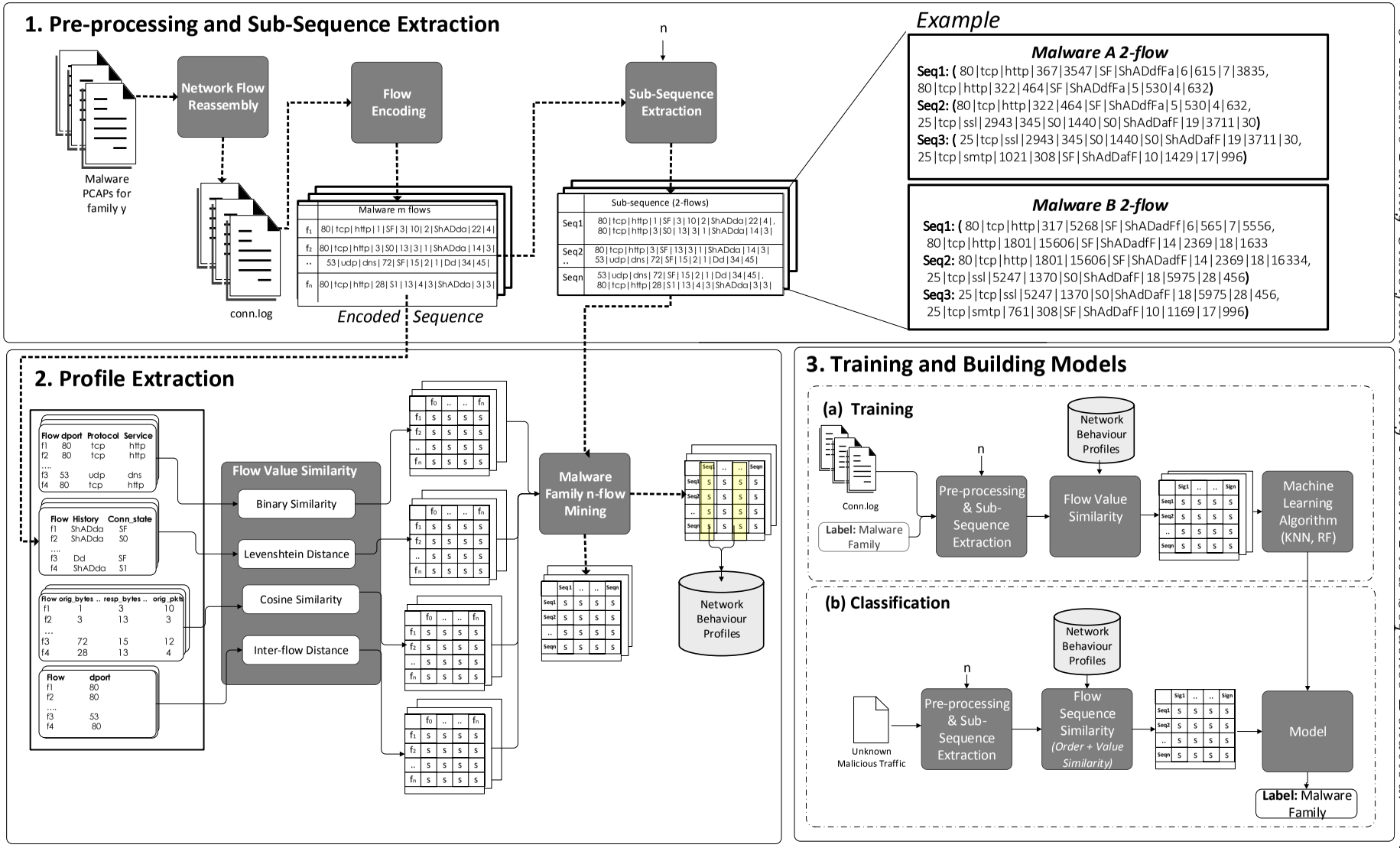

Figure 8.1: MalClassifier System.

2. *Malware Family Profile Extraction:* The *Value Similarity* of each individual flow in the encoded sequence to all other flows is computed, in turn the sub-sequence similarity is determined. The most distinctive flow sub-sequences (n-flows) are selected as profiles for malware family y.

3. *Training and Building Models:* Using the profiles, the supervised machine learning model is trained for classifying unseen n-flows to a malware family.

8.2.1 Design Goals and Requirements

The main limitation of existing malware family classification approaches is the need to obtain the malware executable, run it in a sandbox to classify it to a malware family. Unfortunately, this timely process hinders its adoption in real-world application where access to the infected host is not possible or where a rapid analysis and response is crucial. To foster its real-world use, we consider this limitation and the following requirements when designing *MalClassifier* to ensure that our system is resilient to malware evasion and classifies n-flows with a high accuracy.

IP-agnostic— The system must not use destination IP as a feature. Malware rapidly changes its C&C and deploy sophisticated domain generation algorithms and domain shadowing of legitimate domains to evade reputation filtering.

Non-privacy invasive— MSSP clients require privacy preserving monitoring and systems that observe the payload are privacy invasive. Therefore, the system must not rely on network traffic payload to extract features. In addition to reducing the storage space, this makes the system resilient to encryption which sophisticated malware and benign hosts (*e.g.* HTTPS) use. Moreover, the non-identifiable network data required for training the models are accessible, which is critical for supervised classifier training and for the potential adoption and acceptance of the system at scale.

Automatically identify distinctive malware network behaviour— The system must be able to automatically identify and extract distinctive network behaviour (i.e. profiles) of each malware family.

On-the-wire classification— Obtaining the malicious executable and running it in a sandbox should not be required. Instead, it should be able to classify malicious network traffic on-the-wire to a malware family.

Resilient to malware evasion attacks and adaptability to malware behaviour changes— The system must be adaptive to malware evolution and behavioural changes. Meaning, if the malware network flow behaviour changes slightly (*e.g.* change protocol UDP to TCP or increase/decrease in size), then the

system model should still be able to classify to the correct malware family. In addition, the system should be robust against malware evasion through flow field manipulation by using tamper-resistant features [199]. Malware may attempt to change the order sequence of the flows to avoid detection. Therefore, the system should consider flow order deception, and be able to still classify manipulated sequences with high accuracy.

High classification accuracy— The classifier must aim to provide acceptable classification accuracy using only sub-sequences of network traffic, thus not requiring a malware's full packet captures.

Considering the aforementioned design goals, we discuss the design of each module of the *MalClassifier* system in detail in the following sections.

8.2.2 Pre-processing

In order to convert the network flows into a format that can be applied to sequence mining methods, we first need to pre-process the data by reassembling and encoding the network flows.

Network Flow Reassembly

Organisations deploy network monitoring systems such as Zeek Network Analysis Framework [1](previously known as Bro), which generates statistical and behavioural logs about the network communications, the application level protocols, and exchanged payload of each network flow. Maintaining full network traces (PCAPs) requires a huge amount of storage, making it challenging for organisations to keep full network traces for more than a couple of days. On the other hand, to investigate security breaches, whose effects might show up much later, logs may be stored for a longer period and reduce storage costs. We envision *MalClassifier* to be deployed with network monitoring systems for on-the-wire malware classification.

We note that the *Network Flow Reassembly* module is only needed when converting network PCAP traces to Zeek *conn* logs, and therefore is not needed if the Zeek logs are available or when Zeek is applied in the network for network monitoring.

To extract behavioural features from the malware PCAP traces, we use Zeek to reassemble the network flows. A flow is a sequence of packets from a source host and port to a destination host/port that is part of a unique TCP/UDP session. Packets in a flow are either going to (or coming from) the same destination IP address and port. As input, Zeek takes the captured malware PCAP network traces and

[1]https://www.zeek.org

Table 8.1: Description of the fields in conn.log generated by Zeek network monitoring framework and used in *MalClassifier*.

Field	Description
resp_port	Destination port
proto	Transport layer protocol (TCP, UDP)
service	Application protocol being sent over the connection.
orig_bytes	Number of payload bytes the originator sent
resp_bytes	Number of payload bytes the responder sent
conn_state	State of the Connection. 13 different states (*e.g.* connection attempt rejected)
history	State history of connections as a string of letters.
orig_pkts	Number of packets that the originator sent
orig_ip_bytes	Number of IP level bytes that the originator sent
resp_pkts	Number of packets that the responder sent
resp_ip_bytes	Number of IP level bytes that the responder sent

generates a number of logs for each malware sample. These logs include information that is useful in understanding malware behaviour, such as C&C communication statistics, DNS queries and fast fluxing, unusual communications (*e.g.* unknown protocols) and port-host scanning.

In *MalClassifier*, we leverage the Zeek *conn.log* file that shows non-identifiable network flow header information of TCP/UDP/ICMP connections. Each row in the log represents an individual flow f and is described by 20 attributes representing the column fields. We use 11 of the attributes derived from the *conn.log* and described in Table 8.1.

Flow Encoding

For each log $x \in X$, where X is the set of all Zeek *conn.log* logs for samples of a malware family y, we encode the log x to a long *sequence* of network flows, $E(x) = f_1 \rightarrow f_2 \rightarrow .. \rightarrow f_i$, where f_i represents a single flow in the log x. Formally, we define a flow f_i as:

$f_i :=< resp\_port, proto, service, orig\_bytes, resp\_bytes, conn\_state, history,$
$orig\_pkts, orig\_ip\_bytes, resp\_pkts, resp\_ip\_bytes >$

As we show in Figure 8.1, the result of the Flow Encoding module is a textual sequence representation of the flows in the *conn.log* of malware samples of a malware y.

Flow Sub-Sequence Extraction

We represent the behaviour of malware as a sub-sequence of flows. This provides higher granularity of the captured malware network behaviour. For example, a single *icmp* flow does not map to a particular behaviour, but a sequence of multiple *icmp* flows represent a malware performing an *icmp* scan. To capture the malware flow-level behaviour, we consider *sub-sequences* of the malware network flows of length n, called n-flows as shown in Figure 8.1. Therefore, when $n = 1$, the 1-flow represents a single network flow, and when $n = 2$, the bi-flow represents two consecutive network flows, and so on. This results in a group of consecutive network flows of length n, that reflect flow-level behavioural patterns. Such an approach allows us to capture the network sequence behaviour and extract the unique sub-sequence (i.e n-flow) profiles.

8.2.3 Malware Family Profile Extraction

MalClassifier extracts the malware family's distinctive network flow sequence behaviour, abstracting that behaviour as a *sequence profile*. We discuss below how these profiles are generated for each malware family. To clarify the different steps involved in generating the behavioural profiles, we use a *running example*. Thus, we show in Figure 8.1 an example of the extracted sub-sequences or n-flows (when $n = 2$) of two malware samples (Malware A and Malware B).

Flow Sequence Similarity

Similarity measures are defined as "functions that quantify the extent to which objects resemble each other, taking as an argument object pair and return numerical values that are higher as the objects are alike" [200]. Choosing the similarity measure relies on the data nature itself, whether binary, numerical or structured data (*e.g.* sequences, trees, graphs).

Similarity measures for binary data are used to determine the presence or absence of characteristics in the object pair (*i.e. pair of flows*), thus take a value 1 if the flow possesses the characteristic, and 0 otherwise. For example, the authors in [100] applied a binary similarity approach, by counting the number of times the exact n-gram occurs. We argue that network flows are structured data, and thus binary similarity are not suitable as they do not capture the malware underlying network flow semantic.

Instead, *MalClassifier* applies a *fuzzy* Value Similarity measure. Thus, instead of determining *is a sub-sequence for malware A and a sub-sequence for malware*

B the same?, fuzzy similarity determines how similar a sub-sequence in malware A to a sub-sequence in malware B by computing the degree of similarity. Value Similarity computes the similarity of each flow f_{iA} in a sub-sequence for Malware A to its corresponding flow f_{iB} in the sub-sequence for Malware B.

A single flow f is composed of a number of attributes as we defined in Table 8.1. Each attribute is semantically different in data type and value distribution, and thus when choosing a similarity measure the underlying differences between the attributes need to be considered. For example, *history* is represented as a string of characters, where each character has a meaning and the order of the characters also has a meaning that should be considered. In contrast, numeric attributes such as (*orig_bytes, resp_bytes, orig_pkts, orig_ip_bytes, resp_pkts, resp_ip_bytes*), are represented as numeric vectors, thus numeric similarity measures such as Cosine Similarity should be applied.

Although *resp_port* is a numeric, the underlying meaning differs from other numeric fields. For example, although the values of $orig\_bytes = 100$ or $orig\_bytes = 99$ should be considered similar, a small difference in *resp_port* does not indicate a similarity (port 80 for HTTP could be considered related to port 443 for HTTPS, whilst port 23 being closer to port 80 is semantically different). Therefore, applying one similarity measure to all attributes is not sufficient.

We propose a hybrid value similarity measure to determine the similarity of two flow sequences. Our hybrid approach takes into account the semantic differences of the flow attributes and applies a similarity measure suitable to each attribute. In general, we apply four similarity measures, depending on the flow attributes.

We list in the following each similarity measure providing an example of how the similarity of Seq_1 of Malware A and Seq_2 of Malware B shown in Figure 8.1 is calculated.

Binary similarity—(*resp_port*, *protocol*, *service*): The similarity is 1 if the attribute values are the same, otherwise 0. Thus, the Binary Similarity in our example of ($f_{0A} \in Seq_1 : 80|tcp|http, f_{0B} \in Seq_2 : 80|tcp|http$) and ($f_{1A} \in Seq_1 : 80|tcp|http, f_{1B} \in Seq_2 : 25|tcp|ssl$) is (1, 0.33) respectively, resulting in an average Binary Similarity of 0.665.

Levenshtein Distance [201]— (*history*,*conn_state*): Levenshtein Distance is a fuzzy string similarity measure that measures the minimum number of modifications required (insertions, deletions, and substitutions) to change one string into the other, divided by the maximum length of the same two strings. It also takes into consideration the order of the characters in the string.

Assuming the cost of insertion, deletion, and modification is the same (= 1), then the Levenshtein Distance of making $ShADdFa$ into $ShADadR$ is 3. The trivial implementation has a runtime and space complexity of $O(nm)$. The distance value ranges from [0,100], where 0 indicates a low distance thus higher similarity and 100 indicates a low similarity. We scale the distance value to be in the range [0,1] instead of [0,100].

Using our example, the Levenshtein Distance of ($f_{0A} \in Seq_1 : SF|ShADdFa, f_{0B} \in Seq_2 : SF|ShADadfF$) and ($f_{1A} \in Seq_1 : SF|ShADdFa,\ f_{1B} \in Seq_2 : S0|ShAdDafF$) is (2, 4) respectively, resulting in an average Levenshtein Distance of 3, scaled to 0.03.

Cosine Similarity— The numeric attributes of the two flows are represented as two vectors. Cosine Similarity measures the similarity of two non-zero vectors by calculating the cosine of the angle between them. Given the vectors x and y of length $n = 6$ (*number of numeric attributes*), the cosine similarity is represented as:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{xy}}{\|\mathbf{x}\|\|\mathbf{y}\|} = \frac{\sum_{i=1}^{n} \mathbf{x}_i \mathbf{y}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{x}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{y}_i)^2}} \tag{8.1}$$

The Cosine Similarity of $x = [367, 3547, 6, 615, 7, 3835] \in f_{0A},\ y = [1801, 15606, 14, 2369, 18, 16334] \in f_{0B}$ is 0.00025.

Similarly, the Cosine Similarity of $x = [322, 464, 5, 530, 4, 632] \in f_{1A},\ y = [5274, 1370, 18, 5975, 28, 456] \in f_{1B}$ is 0.284. Thus, the average Cosine Similarity for Seq_1 and Seq_2 is 0.142.

Inter-flow Distance— (*resp_port*): Inter-flow distance calculates the distance between the *resp_port* in every two consecutive flows of a sub-sequence. This helps identify malware network behavioural attributes such as performing a port scan (*e.g. when the difference of resp_port of two consecutive flows is 1*). To calculate the inter-flow similarity, we first calculate the distance of *resp_port* in each consecutive flows in a sub-sequence. For example, the *resp_port* distance between $f_{0A}, f_{1A} \in Seq_1$ is 0, as the *resp_port* in both flows is the same. However, the *resp_port* distance between $f_{0B}, f_{1B} \in Seq_1$ is 55, which is the difference between port 80 and port 25. The inter-flow similarity of Seq_1 and Seq_2 is the distance of (0,55), thus is 55.

The Inter-flow distance is normalised to obtain a value in $[0, 1]$ to define a dissimilarity. The simplest and most common normalisation method uses a linear transformation, known as feature scaling where $\eta(d) = \frac{d - d\_min}{d\_max - d\_min}$. However, the drawback of applying this method is that it is sensitive to outliers [200]. Therefore, to normalise the distance value, we apply another normalisation method that overcomes this drawback by defining parameter $Z = (m, M)$, where m and

M are user-defined values and are interpreted as a tolerance threshold, where $\eta_Z(d) = min(max(\frac{d-m}{M-m}, 0), 1)$ [200]. Any distance value d less than min means that there is zero dissimilarity, thus distinct points (that have a non-zero distance) could be determined identical. Consequently, distance values higher than max are considered totally dissimilar [200]. In our example, we set $Z = (10, 100)$, resulting in a dissimilarity of 0.5.

We define the hybrid similarity approach in Algorithm 1: *Value Similarity*. The Value Similarity function takes as input two flow sub-sequences, X and Y. For each flow f in sub-sequence X and Y, we extract the *port_protocol_service*, *history_state*, and *numeric* attributes, and calculate the Binary similarity, Levenshtein Distance, and Cosine Similarity, and Inter-flow Distance consequently. The similarity of the pair of flows is the sum of the four similarities, each multiplied by a weight w. The weight w is defined as the number of attributes that were given to the similarity measure, *e.g.* $flow\_sim = 3\ binary + 2\ ld + 6\ cosine\_sim + 1\ inter\_flow$. The highest possible similarity score using this approach is 12. The weights w could also be used to give *importance* to a similarity measure over another.

Using our example, X would be Seq_1 and Y is Seq_2. The similarity of Seq_1 and Seq_2 is calculated as follows:
$sim = 3(0.665) + 2(1 - 0.03) + 6(1 - 0.142) + 1(1 - .5) = 9.5$

As Levenshtein Distance, Cosine Similarity, and Inter-flow Distance are distance measures, we derive the similarity measure through decreasing functions as $S(x, y) = 1 - LevenshteinDistance(x, y)$ and $S(x, y) = 1 - CosineDistance(x, y)$. Thus the similarity measure is the complement to 1 of dissimilarity [200].

Malware Family n-flow Mining

To extract the network flow sequence behaviour for a malware family y, we derive the set S of n-flows of all the network flows in the log x that belong to the malware family y. We calculate the similarity $sim(i, s)$ of each n-flow i in a malware sample m_j to each n-flow s in the set S. To calculate the value $sim(i, s)$, we apply the value similarity measure defined in Section 8.2.3.

To reduce memory and processing complexity, we pre-compute the similarity of each pair of flows in S and store as a key-value store. Consequently, when determining the similarity of two n-flows, the similarity of each flow is pre-computed and is fetched from the key-value store. Therefore, only unique flow pairs are stored and only the similarity of new flow pairs are computed.

We are interested in mining the malware family's network traffic for the highly frequent n-flows, i.e. flows that occur in all or the majority of the samples of

Algorithm 1 Value Similarity

```
procedure ValueSimilarity(X,Y)
    X ← [x_1, x_2, ..., x_i], where x_i = [f_0, f_2, ..., f_10]
    Y ← [y_1, y_2, ..., y_i], where y_i = [f_0, f_2, ..., f_10]
    attributes ← [], attributes of flow in conn.log
    Sim ← []
    for i in range (0,n) do
        x_i ← X[i]
        y_i ← Y[i]
        x_{i+1} ← X[i + 1]
        y_{i+1} ← Y[i + 1]
        port_prot_ser ← (x_i[0 : 2], y_i[0 : 2])
        history_state ← (x_i[3 : 4], y_i[3 : 4])
        numeric ← (x_i[5 :], y_i[5 :])
        if port_prot_ser[0] == port_prot_ser[1] then
            binary = 1
        else
            binary = 0
        ld = 1−LevenshteinDistance(history_state)
        cosine = 1−CosineDistance(numeric)
        interflow = abs((x_i[0] − x_{i+1}[0]) − (y_i[0] − y_{i+1}[0]))
        s = w_0 binary + w_1 ld + w_2 cosine + w_3 interflow
        Sim.append(s)
    return Average(Sim)
```

Algorithm 2 Order Similarity

```
procedure OrderSimilarity(X,Y,n)
    X ← [x_1, x_2, ..., x_i], where x_i = [f_0, f_2, ..., f_10]
    Y ← [y_1, y_2, ..., y_i], where y_i = [f_0, f_2, ..., f_10]
    n ← sequence length
    if length(X) == 1 then
        return ValueSimilarity(X,Y) * 1/n
    else
        s_0 = ValueSimilarity(X,Y) * length(X)/n
        s_1 = OrderSimilarity(X[1 :],Y[1 :],n)
        s_2 = OrderSimilarity(X[: −1],Y[: −1],n)
        return Max(s_0,s_1,s_2)
```

a malware family. Therefore, we represent the n-flow i as a vector in a high-dimensional space, where the elements (*features*) in the n-flow vector corresponds to all n-flows of a malware family y.

Profile Selection

In order to extract the malware family profiles, we select the most relevant n-flows (profiles) that represent that malware family's network behaviour. For each malware family, we mine the network flows for all samples of that malware family, then select n-flows that are significant. We apply a modified version of class-wise feature selection approach as proposed in [202]. The class-wise document frequency is the number of network flows in a malware family y that contain an n-flow s. We assign each n-flow a class-dependent weight according to its coverage in the malware family network traffic (*tendency*). As we apply a fuzzy similarity approach, we multiply

the average similarity of an n-flow s in a malware family y to its *tendency*. We then select the top k n-flows as profiles for that malware family.

8.2.4 Building Malware Family Classification Models

To avoid misclassifying malware binaries as a result of not having access to its full packet capture, we classify the network n-flows. This ensures that the malicious binary can be classified based on a sub-set of its network flow communication, thus not requiring the malware's whole network packet capture.

Training— We show in Figure 8.1 how the classifier is trained to produce the model used for classification of future unseen n-flows. To train the classifier, a collection of malicious n-flows of malware samples $M = m_1, m_2, ..., m_l$ and their corresponding malware families $Y = y_1, y_2, ..., y_k$ are required. Hence, we train a multi-class supervised classifier with the aim of determining if an n-flow i of a malware sample m belongs to a malware family y, where the selected profiles S represent the features in our classifier.

Classification— The trained model is then deployed to classify unseen n-flows to a family. One of the main design goals for *MalClassifier* is resiliency to malware evasion. Malware authors may try to evade the sequence detection by changing the order of the flows or injecting noise flows. Therefore, when deploying the trained model, in addition to calculating the *Value Similarity* we also introduce the *Order Similarity* defined as Algorithm 2. The Order Similarity algorithm computes the highest Value Similarity score of all possible sub-sequence in X and Y.

Using our example, the possible sub-sequences for X are $(f_{0A}), (f_{0B}, f_{1B}), (f_{1A})$ and for Y are $(f_{0B}), (f_{0B}, f_{1B}), (f_{1B})$, where $length(subsequence) <= n$. The Order Similarity computes the Value Similarity of all possible sub-sequence orders, thus $S(f_{0A}, f_{0B}), S((f_{0B}, f_{1B}), (f_{0B}, f_{1B})), S(f_{1A}), (f_{1B}))$. Then, the Value Similarity is multiplied by $length(X)/n$. Thus, the similarity of the sub-sequence is at its highest when the length of sub-sequence for X = n, and at its lowest when the length of sub-sequence X = 1, thus is a single flow.

8.3 Evaluation

To evaluate our approach, we implement the system as a multi-threaded Python application. To accelerate the analysis, the application uses Python Multiprocessing with separate threads to calculate the similarity scores for each malware sample. The experiments were conducted on a 40-core processor with 126GB RAM, Centos OS.

Table 8.2: Description of datasets.

Dataset	Family	# Samples	# Flows	# Unique Flows
CTU-13	Murlo	1	37019	422
	Rbot	4	46,184,716	1697
	Virut	2	358,378	4088
	Neris	3	839,077	24895
	Menti	1	291,677	160
	NSIS.ay	1	30,063	2591
Stratosphere IPS Project	Miuref	7	1867273	17257
	Sality	1	6,073,775	307,355
	WannaCry	7	291,677	330
	Conficker	2	323,238	6,300
	Notpetya	4	5,424	174
Total		33	56,302,317	365,289

8.3.1 Dataset

Botnets are known to follow a certain infection life-cycle, sharing similarities in network flow characteristics in each infection phase [82]. Thus, we evaluate the effectiveness of our system in determining the unique network flow behaviour of each botnet family in our dataset, and its accuracy in classifying botnet network traffic to its family despite the botnets' network behaviour similarities. To train our classifier we used (1) the CTU-13 botnet traffic dataset [183] provided by the Malware Capture Facility project and (2) current botnets and ransomware (*Miuref, WannaCry, Conficker, Sality, Notpetya*) provided by Stratosphere IPS Project[2], both described in Chapter 4.

We applied network flows of scenarios 1−13 to train and evaluate the classification models. Scenario 7 was excluded as it did not contain a sufficient number of flows. We applied only the malicious flows (C&C and botnet flows) to train the classifier for malware family classification. We provide an overview of the malware families in our datasets in Table 8.2.

8.3.2 Experiments

The aim of the experiments is to (1) determine how accurately we can classify an n-flow to its malware family using the extracted profiles as features; and (2) determine the classifiers robustness to malware evasion. In particular, we plan to explore the following:

[2]https://www.stratosphereips.org/

Impact of Flow Sequence Similarity— We measure the effect of applying each similarity approach (*Levenshtein Distance, Cosine Similarity, Binary Similarity, and Inter-flow Distance*), on the classifier performance. This will help determine which similarity measure has the highest positive influence on the classification and thus will be assigned a higher weight w, as discussed in Section 8.2.3.

Impact of n-flows— We evaluate the impact of using an n-flow approach for malware family classification. We separate the network flows in our dataset into n-flows of length 1 (single flow) to 7 consecutive flows. The aim is to determine if the malware family classification accuracy improves when a sequence of flows approach such as n-flows is applied as opposed to a single flow.

Robustness to Malware Evasion— Malware authors could attempt to evade detection by changing the malware network flow behaviour, either by injecting noise packets and changing the flow sequence order, affecting the numeric attributes in a flow *e.g.* sent/received packets. To account for this, we explore the robustness of the classifiers' to evasion by evaluating the classifier's performance in classifying n-flows when we randomly shuffle the order of the flows in a sequence.

8.3.3 Classifier Design

We build multi-class classifiers that map an unknown network n-flow input as belonging to one malware family.

Classifiers— We apply two supervised machine learning classifiers, K-Nearest Neighbour (KNN) and Random Forest (RF). These classifiers were chosen due to their popularity in text classification using a vectorial representation of features. KNN is a non-parametric lazy learning algorithm. It simply assumes that the classification of a sample is similar to other samples that are nearby in the vector space. Random Forest is an ensemble classifier that leverages multiple decision trees, that are trained using different subsets of the training set, thus overcoming over-fitting issues of an individual decision tree.

Classifier Features— The extracted flow sequence profiles for each malware family represent the features in our classifiers. We only use 20% of the n-flows for each malware family to generate and select ($k = 20$) behaviour profiles. This is to ensure that the profile selection does not bias the classifier estimates, known as *feature subset selection bias* or *selection bias* [203].

Dimensionality Reduction— Due to the sparsity of the dataset (20 profiles * 11 (Number of classes) = 220 features), we apply Principle Component Analysis (PCA) [204]. PCA reduces the dimensionality of the dataset while retaining the variation present in the dataset, up to the maximum extent. Given the set of

Table 8.3: Example of the selected 2-flows for each malware family in our dataset.

Family	Example 2-flow
Murlo	*135.0-tcp-0-0-0-S0-S-2.0-96.0-0.0-0.0* *135.0-tcp-0-0-0-S0-S-2.0-96.0-0.0-0.0*
Rbot	*14.0-icmp-0-0-0-OTH-0-0.0-0.0-0.0-0.0* *227.0-icmp-0-0-0-OTH-0-0.0-0.0-0.0-0.0*
Virut	*80.0-tcp-0-0-0-RSTO-^hR-2.0-80.0-1.0-44.0* *443.0-tcp-0-0-0-REJ-Sr-2.0-96.0-1.0-40.0*
Neris	*80.0-tcp-http-318-368-RSTO-ShADadR-10.0-1052.0-3.0-496.0* *25.0-tcp-0-0-0-S0-S-2.0-96.0-0.0-0.0*
Menti	*25.0-tcp-0-0-0-S0-S-4.0-192.0-0.0-0.0* *25.0-tcp-0-0-0-S0-S-2.0-96.0-0.0-0.0*
NSIS.ay	*32234.0-udp-0-103-596-SF-Dd-1.0-131.0-2.0-652.0* *31037.0-udp-0-103-596-SF-Dd-1.0-131.0-2.0-652.0*
Miuref	*5353.0-udp-dns-1599-0-S0-D-36.0-2607.0-0.0-0.0* *136.0-icmp-0-0-0-OTH-0-1.0-72.0-0.0-0.0*
Sality	*80.0-tcp-0-0-0-S0-S-2.0-96.0-0.0-0.0* *53.0-udp-dns-30-46-SF-Dd-1.0-58.0-1.0-74.0*
WannaCry	*445.0-tcp-0-0-0-REJ-Sr-1.0-52.0-1.0-40.0* *445.0-tcp-0-0-0-S0-S-1.0-52.0-0.0-0.0*
Conficker	*3824.0-tcp-0-0-0-RSTOS0-R-2.0-80.0-0.0-0.0* *3821.0-tcp-0-0-0-RSTOS0-R-2.0-80.0-0.0-0.0*
Notpetya	*130.0-icmp-0-0-0-OTH-0-1.0-72.0-0.0-0.0* *130.0-icmp-0-0-0-OTH-0-1.0-72.0-0.0-0.0*

transition probabilities of all the samples, PCA evaluates the variance of each feature to linearly transform the feature space into a new one where the information (evaluated as a percentage of variance) is contained in as few components as possible. Thus, the 220 features are transformed to a new set of features, known as the principal components. By reducing the number of features (dimensionality) used in the classifier, the computational cost is reduced.

We set $PCA = 10$, thus reducing the features space to 10 principle components. We evaluated different values for PCA, and the classifier performance when we apply $PCA = 10$ performs almost as well as when we use all features. The advantage of using PCA over using all features is the low memory complexity required to run the machine learning algorithms due to reducing the dimensions of the features space used by the classifier.

Classifier Performance Measures— To assess the performance of the classifiers, we apply 10-fold cross-validation. Cross-validation removes any bias in the data while maximizing the number of score computations from a given dataset. As

the CTU-13 datasets consist of only a few PCAPs (executions) per family, when performing cross-validation we compare the different executions of the same family.

We employ evaluation measures such as *Precision*, *Recall* and *F-measure* to evaluate the classifiers' performance. A low precision can indicate a large number of False Positives (FP). A low recall indicates many False Negatives. Therefore, we seek an analysis setup that maximizes the precision and recall. We also consider *F-measure (*$F1$*)* an aggregated performance score, that combines both precision and recall. A perfect discovery of classes yields $F1 = 1$, where either a low precision or recall result in a low F-measure.

Although these metrics are defined for a binary classifier, they can be extended for multi-class problems. Each class is represented by a binary classifier *(e.g. One-vs-rest approach*), and we average the binary metric across the set of classes. We also apply a macro-averaged Precision-Recall curve as an evaluation metric, which gives equal weight to the classification of each label compared to micro-averaging which gives equal weight to each per-family classification decision.

8.4 Results

We discuss in the following our results from the experiments outlined in Section 8.3.2 and the impact of the various system configurations and approaches on the classification performance.

8.4.1 Malware Family Profile Extraction

Our results show that malware of a single family exhibit network flow sequence regularities that can be used for malware family classification. We select 20 n-flows for each malware family using the method described in Section 8.2.3. We illustrate in Table 8.3 an example of the selected 2-flows for each malware family.

Murlo traffic contained sub-sequences of TCP flows to destination port 135 (*Messenger Services*). The flows have a connection state $S0$, meaning there was a connection attempt, but no reply. Therefore, the duration of the flow is 0 and there was no payload.

Similarly, *Menti* performed a multiple flow connections to destination port 25 (SMTP) and port 21 (FTP). The connection state set to either RSTO, meaning the connection attempt was rejected by the destination or S0. However, there was some successful TCP flow with the exchanged payload. The high number of outgoing flows to an SMTP port means that the botnet is sending email spam, a behaviour linked to *Menti* that is known to send pharmaceutical and stock-based

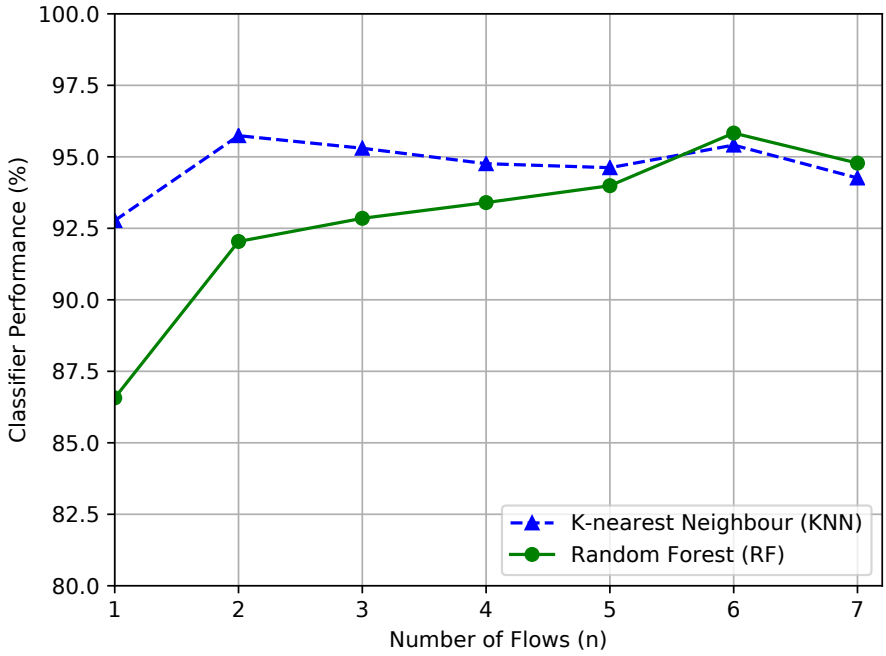

Figure 8.2: Average F-measure for Random Forest and KNN classifiers, $n = 1 - 7$ for n-flows.

email spam. Although, *Neris* 2-flows show HTTP flows with connection state set to RST0. This means that the connection was established but the originator aborted. However, we do see other successful HTTP connections with connection state SF, meaning normal establishment and termination.

WannaCry is a ransomware known for sending SMB flows to other victim hosts on the network. This is captured by the selected 2-flows of TCP flows with a destination port of 445. *Miuref's* 2−flows sequences were mostly HTTPS flows (port = 443), followed by DSN requests or DSN requests followed by an ICMP flow.

8.4.2 Classifier Performance

Impact of the Value of n in n-flows— We illustrate in Figure 8.2 the models' performance for each value of n. Overall, the accuracy of the classifiers is at its best using 2-grams with the KNN classifier and 6-grams with RF classifier. The classifier accuracy was better when flows were represented as a sequence ($n > 1$) rather than individual flows ($n = 1$). This shows that malware behaviour is best captured when we look at sequences of flows. For example, an individual SMTP flow might not infer a malicious behaviour, or capture the behaviour of a particular malware class (*e.g. Menti*), whilst a sequence of rejected SMTP messages gives confidence of a maliciousness of the flow sequence.

When choosing the value of n, the speed of classification needs to be considered. Although a longer sub-sequence (higher n) results in a higher classification accuracy, it delays the classification of the network behaviour to a malware family. For example, choosing 7-grams requires waiting for 7 consecutive malware network communications to classify and understand its behaviour, increasing the duration of the malware execution damage and delaying active remediation. In situations where the attack implications are severe such as in ransomware attacks, determining the malware class as soon as possible is critical for a quick reaction before it reaches the encryption stage.

Impact of Classification Algorithm— To determine the most accurate classifier, we measured the performance based on the machine learning algorithm used. KNN ($n = 2$) performed best (F-Measure = 95.74%) with shorter flow sequences, while RF classifier's accuracy increased as the number of flows in the sequence (n-flow) increases, reaching 95.83% when $n = 6$.

To compare the precision and recall trade-off of the Random Forest classifier, we present the macro-average Precision-Recall (PR) curve when $n = 5$ in Figure 8.3. We employ macro-average that measures the overall precision and recall of all the classes to produce the macro-averaged ROC curves. We build multi-class models, therefore, averaging the evaluation measures can provide us with a view of the general classifier performance. The model has an AUC of 94%, indicating high recall and high precision. Overall, the classification of all malware families performs well. However, *Notpetya* had an AUC of 39%, with a high number of miss-classifications (42%). We will discuss the reasons for this miss-classifications and how to improve the classification accuracy in Section 9.5.

Impact of Flow Sequence Similarity— We measure the effect of the four similarity measures (*Binary Similarity, Cosine Similarity, Levenshtein Distance, and Inter-flow Distance*) and their associated flow attributes on the classifier accuracy. Specifically, we train four Random Forest (RF) classifiers, each using a similarity measure and a subset of the flow attributes. For example, we train a classifier using only $resp\_port, protocol, service$, applying only the Binary Similarity measure. We show the F-measure of each of the four classifiers ($n = 5$) per malware class in Figure 8.4.

The highest performance resulted from the Cosine Similarity classifier, with a 90% F-measure over most malware classes. Binary Similarity classifier was best for representing *Murlo* behaviour with 97.46% F-measure, *Miuref* with 95.86%, and *Rbot* with 99.26%. However, the Binary Similarity for *WannaCry* had a low performance (11.78%) showing that the flows' attributes ($resp\_port, protocol, service$) that are

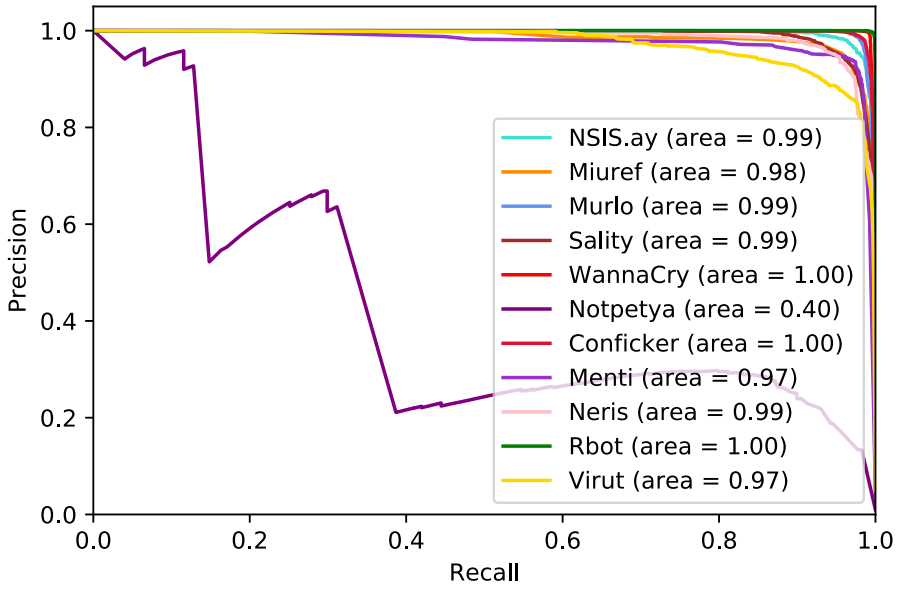

Figure 8.3: Macro-averaged Precision Recall Curves for each malware family (*Random Forest classifier,* $n = 5$).

measured using this similarity approach might not be a unique representative of its behaviour.

However, *WannaCry*'s Inter-flow Distance classifier had the highest performance (93.59%). Thus, although the exact matching (*i.e. Binary Similarity*) of the *res_port* did not perform well, the average difference of the *res_port* between flows in the sequence (*i.e, Inter-flow Distance*) was able to capture that malware family's flow sequence behaviour. Levenshtein Distance had the lowest performance overall, with only having high accuracy with Rbot 97.52%.

8.4.3 Robustness to Evasion

Malware authors can attempt to evade sequence detection by changing the order of the communication flows or even injecting noise flows. We evaluate the use of the Order Similarity, introduced in Section 8.2.3. We randomly shuffle the order of the flows in the $5 - flows$ of our malware families and test the model's classification performance on the shuffled $n-$flows. Change in the order of flows did not affect the classifier accuracy when applying Order Similarity, retaining an F-measure of 95.36% for Random Forest Classifier ($n = 5$).

8.5 Discussion

We discuss how *MalClassifier* meets our design goals and identify potential limitations, suggesting approaches to address them.

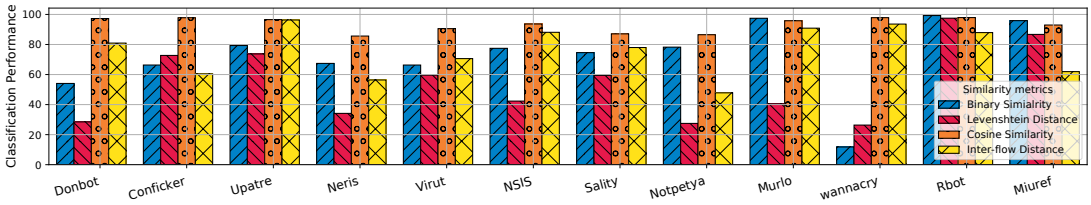

Figure 8.4: The malware familys' classification F-measure of the four Random Forest Classifiers ($n = 5$), each using one of the four similarity measures.

8.5.1 Meeting Design Goals

MalClassifier utilises non-privacy invasive features to train and build its models, relying on packet header information and not requiring deep packet inspection (*i.e. content-agnostic*). In addition, all identifiable header fields such as IP addresses are removed in the network flow encoding module (*i.e. IP-agnostic*). As malware is known to change its behaviour in order to evade detection, *MalClassifier* applies a *fuzzy* approach to flow sequence similarity to ensure that slight deviations in flow attribute values are detected. The main challenge in malware classification is obtaining the required model training datasets. Therefore, *MalClassifier* uses only non-identifiable packet headers features making datasets needed for training and building the models accessible.

MalClassifier achieved a high accuracy for malware family classification (F-measure $\approx 95.5\%$), demonstrating the effectiveness of the system in identifying distinctive network n-flows for each malware family. It is worth noticing that despite the accuracy of our *MalClassifier* not improving significantly over the state-of-the-art, it still provides a high accuracy while preserving communications privacy and being robust against encryption. In addition, the classifier performance can be improved by modifying the profile selection module, as we will discuss in the next section. *MalClassifier* classifies n-flows to a malware family, meaning it only requires a subset of flows instead of obtaining the full packet trace of the malicious binary for classification.

8.5.2 Understanding Classifier Errors

We represent the confusion matrix for the Random Forest Classifier ($n = 5$) in Figure 8.5. Each row in the confusion matrix represents the instances of the actual class (*i.e. True Label*) while each column represents the instances of the predicted class (*i.e. Predicted Label*). The main observation is that 26% of *Notpetya* 5-flows where incorrectly classified to *Miuref*. We identified the miss-classified *Notpetya*

flows to a sequence of 5 flows of *445-tcp-0-0-0-S0-S-4-192-0-0*. However, such a sequence was not selected as a profile for *Notpetya*, whilst a similar 5-flow *443-tcp-0-0-0-S0-S-1-48-0-0* was selected as a profile for *Miuref*. Therefore, the classifier was trained to assign such an n-flow to *Miuref*. To improve the classification, n-flows that are shared by more than one malware family should be identified and not selected beforehand. This could be done using clustering approaches, which we consider for future work.

8.5.3 Lessons Learned

The performance of the classifier relies on the quality of the profiles selected for each malware family. We introduced a novel method for profile selection that selects n-flows for each malware family using two metrics: (1) average similarity score for that sequence; (2) tendency, the number of times a sequence occurred. In our initial experiments, we noticed that selected flows for a malware family were all similar, as they all have a high score and similar flow attribute values except the destination port. Thus, the profile selection was not capturing the various distinctive behaviour. Accordingly, we amended the selection process to not include the destination port field, selecting a distinctive set of profiles for each malware family. This increased the accuracy of the classifier by 10%, as the profiles selected represented various stages of a malware family network behaviour.

We note that for extracting the sequence profiles we only looked at 20% of the traffic of each malware family. Using these profiles, we evaluated the classifier performance in classifying the other 80% of traffic. However, in application, the profile selection process should consider the various malware network behaviour stages, to ensure that the selected profiles capture the malware behaviour in each infection stage.

We measured the classifier performance using each similarity measure used in the *Value Similarity*. The effect of the similarity measure on each malware family differs. Although Cosine Similarity had a positive effect on the classification accuracy of each malware family, some families were also highly influenced by the Binary Similarity (*e.g. Murlo)*, and Inter-flow Distance (*e.g. WannaCry*). This provides insight on the similarity measures that have the most positive influence on the classifier performance, thus can be assigned a higher weight w as defined in Section 8.2.3. Based on the results, we can assign Cosine Similarity the highest weight, followed by Inter-flow Distance, Binary Similarity, and finally Levenshtein Distance.

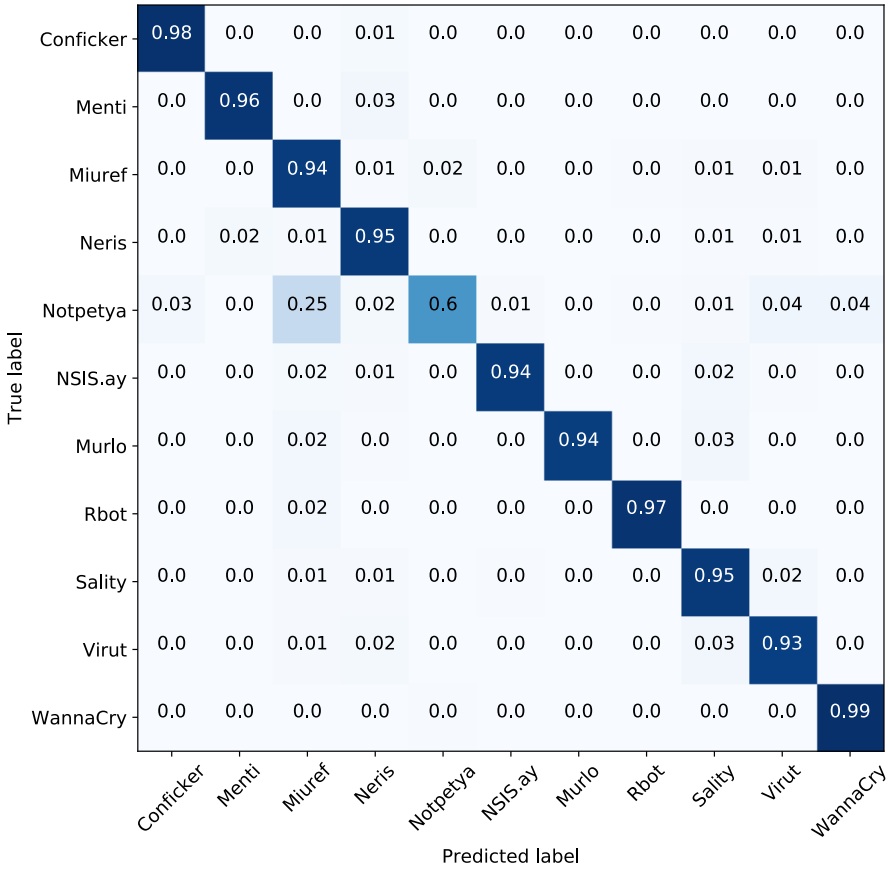

Figure 8.5: Normalised confusion matrix showing actual classes vs. predicted classes for the Random Forest Classifier (n = 5).

8.5.4 Evasion and Limitations

The main challenge in most behavioural-based malware analysis approaches is malware behaviour obfuscation and manipulation, known as *noise-injection* attacks [205]. Although malware evasion by altering the binary itself might be feasible due to available obfuscation tools, we believe that the network behaviour is more troublesome to tamper with. We determine the feasibility of such an evasion in respect of two associated costs: implementation complexity and effect on malware utility. The evasion complexity is based on the ease which the malware author can modify the code to include the evasion tactic which may result in affecting it's utility [206].

Malware classification systems that apply supervised machine learning approach require continuous training of new malware variants to adapt to behavioural changes. However, applying a fuzzy similarity measure allows a degree of flexibility in malware behavioural change, thus only needs training on samples of a new malware family.

In addition, *MalClassifier* adapts to flow sequence order manipulation by applying the Order Similarity approach.

We identified that the reason for the classifier misclassifications was due to the feature selection not considering similarities of n-flows between families. Although we discussed how we ensure the selection of distinctive flows for a malware family, these flows should also not be similar to selected flows for other families. For example, *Neris*, *Virut* and *Sality* are all bots that send email spam, and identifying network flow sequences distinctive for each family can avoid n-flow misclassifications. Thus, to improve the profile selection process, K-means clustering can be applied to identify flows that are similar in more than one family, to avoid using these flows as profiles. Moreover, identifying the frequent n-flows in *benign* traffic can help reduce the false positives.

8.6 Comparison to Related Work

The most directly related work to ours is *CHATTER* [100]. Such system considers the order of high-level network events as features and applies n-gram document analysis to produce the classifier achieving 90% accuracy. The malware network behaviour profile is represented as fine-grained events, thus each attribute of a single packet is considered an event. For example, an inbound packet using TCP on port 80 is represented as $A1\_A3\_A6$, where A1 refers to the inbound connection event, A3 refers to the TCP protocol usage event, and A6 refers to the usage of port number 80 event. Instead, *MalClassifier* uses a similar approach to [101], using coarse-grained groups to represent a network behaviour, i.e. $A1A3A6$ is considered a single event. Therefore, our system is able to capture behaviours such as SMTP flow followed by another SMTP flow that might indicate an email spam.

Moreover, the numeric features are mapped into one of the four quartiles that are pre-determined based on the training data. However, there is a risk of new malware variants falling out of the quartiles ranges leading to inaccurate family classifications. In addition, this abstraction may result in loss of underlying distinctive behaviour that could have resulted in more accurate classifications.

Instead, *MalClassifier* uses *Cosine Similarity* to compare the similarity of the numeric features, thus not requiring pre-determined ranges or thresholds. The main limitation of previous work is their use of payload features, meaning they are vulnerable to malware obfuscation through encryption and that they are not privacy-preserving.

Table 8.4: Comparison of related work on malware family classification and MalClassifier against our design goals.

	Data	Content-agnostic	IP-agnostic	Evasion Resiliency	Use n-grams	Use sand-box	Accuracy
Rieck et al. [56]	Behavioural reports	N	N	N	N	Y	80.7%
Rieck et al. [57]	Behavioural reports	N	N	N	N	Y	95%
Perdisci et al. [103]	HTTP traffic	N	Y	N	N	N	N/A
FIRMA [104]	Network traffic	N	N	N	N	N	98.8%
CHATTER [100]	Network traffic	N	Y	N	Y	Y	90%
MalClassifier	Zeek *conn* logs	Y	Y	Y	Y	N	96%

In Table 9.6, we compare the beforehand related work to *MalClassifier*. Specifically, we identify the data used to train the classifier, whether the approach is IP and content agnostic, uses n-grams, and if the approach requires running the sample in a sandboxed environment. We also identify whether related work addressed malware evasion through noise injection in their system design.

Previous approaches such as [100, 103, 104] proposed systems for malware classification using network traffic. Perhaps the most related work to ours is [100] that observes the high-level network features of malware and applied n-gram document analysis for family classification. Our work improves on [100], by (1) using non-payload features, making our system privacy aware and robust against encryption; (2) adapting to malware behaviour changes; (3) not requiring the execution of the malware in a sand-box, thus performing classification on-the-wire. Similarly, previous approaches are either not content agnostic, relying on features from payload (e.g. [103, 104]) or protocol dependent (*e.g. cover only HTTP traffic*) such as [103].

8.7 Summary

In this chapter, we presented a novel approach for analysing and classifying network traffic of malware variants based on their network flow sequence behaviour. Considering the limitations of existing approaches, we proposed a system that is privacy-preserving, time efficient, and resilient to malware evasion. We showed

MalClassifier's effectiveness in identifying frequent malware network n-flows and its robustness against malware evasion by flow order alteration. *MalClassifier* eliminates the need to have access to the malicious binary. This allows SOC analysts to classify malicious network flow sequences on-the-wire, reducing the time and effort required in other dynamic analysis approaches while maintaining a high classification accuracy.

Representing malware network behaviour as a sequence proved reliable in discriminating between malware families. In the following chapter, we investigate how we can model malware's flow sequence behaviour for building effective malware detection systems.

9

Bot Detection by Building Markov Chain Models of Bots Network Behavior

Contents

9.1 Introduction

In Chapter 8, we applied malware network behavioural features for malware family classification. In our results, we found that most of the 2-flows selected for the malware families are a sequence of two identical flows. For example, *Notpetya* traffic contained a sequence of 5 flows of *445-tcp-0-0-0-S0-S-4-192-0-0*. Moreover, we found that the *conn_state* to be a lightweight and useful feature in understanding the outcome of this connection. Mariconti et al. [207, 208] showed the effectiveness of building Markov Chains from the sequence of Android API calls for malware detection that can maintain its detection capabilities over time. In this chapter, we explore if Markov Chains can be applied to a sequence of connection states (*conn_state*) to provide insights on bot behaviour.

Malware tends to launch their attacks in bursts [209, 210], meaning they send multiple network connections in a short amount of time. Hence, we first explore the bursty nature of malware, to determine how frequently bots send network traffic. We then investigate the discrepancies between malware network connections and those produced by benign applications. Malware network traffic is then modelled using Markov Chains to explore its strength in capturing bots' network communication behaviour during its C&C interactions, propagation stage, and attack operations.

To evaluate the effectiveness of applying Markov Chains for malware network connections, we propose *BOTection*, a novel system that detects malware-infected hosts (bots) on a network and classifies it to its malware family by monitoring their network communication. We consider all types of malware-infected hosts that communicate with a command and control server (e.g. ransomware).

In Chapter 7.8.7, the analysts reported that one of the weaknesses in existing tools is that signatures are "loosely written". Meaning, signatures are written to capture a general malware behaviour (e.g. network scanning) but are incapable of then identifying its malware family. This poses a challenge to analysts that need to determine the cause of that activity to determine the best course of action. Hence, we explore Markov Chains capability in modelling general malware behaviour capable of detecting unseen malware. We then evaluate the system's performance in attributing that activity to a particular malware family.

9.2 BOTection System Design

BOTection system first detects a bot's network activity, then classifies such activity to a particular bot family. In designing *BOTection*, we consider its use in the

real world. For example, organisations may deploy middle-boxes that intercept their network communications, thus allowing the deployment of content-based detection mechanisms. Alternatively, some organisations outsource their security monitoring to MSSPs, due to lack of cybersecurity expertise or to avoid high-security investments [24]. As we identified in Chapter 6, one of the factors that impacts SOC operations, specifically outsourced SOCs, is security clearance. Some organisations operate under restricted security permissions, requiring SOC analysts to have certain level of clearance to view network traffic payload. For example, maintaining the confidentiality of email communications is critical in such restricted organisations. However, as we found also in Chapter 6, recruitment of SOC analysts that have security clearance is a challenge. Hence, content inspection technologies are discouraged and using non-privacy invasive features is crucial to foster *BOTection's* adoption in scale.

Considering the limitations of existing approaches and aforementioned monitoring restrictions, we identify five main design goals that the *BOTection* system has to fulfill: ❶ resilience to obfuscation/encryption by avoiding deep packet inspection (Section 9.2.1), ❷ privacy preservation (Section 9.2.2), ❸ independence of network topology and protocol (Section 9.2.2), ❹ high detection and family classification accuracy (Section 9.4.2, 9.4.1), and ❺ capability of detecting unseen bots by capturing general bot network behavior (Section 9.4.2).

BOTection operates in four phases as illustrated in Figure 9.1.

1. *Network Flow Reassembly:* The malicious and benign network traces are converted to logs using Zeek. These logs are then split to sub-logs of n flows.

2. *Connection State Extraction:* The features are (e.g. *conn_state*) from the logs and produce a key-value store of state transitions and their frequency.

3. *Markov Chain Modelling:* The state transition frequency are used to build Markov Chain models and produce a feature vector for each sub-log.

4. *Training and Building Models:* Malware detection and classification models are built to detect malicious n-flows then classify it to a bot family.

We show the *BOTection* system in Figure 9.1, describing in this section its modules and how design goals were considered.

9.2.1 Network Flow Reassembly

Previous work (e.g. [84, 211]) used a high-level representation of the network (NetFlow) that are easier to obtain than full network dumps (for privacy concerns) and ensuring resiliency to encryption. Similarly, *BOTection* reassembles the network communications of the bot traces to *flows*, extracting content-agnostic features — fulfilling design goal ❶. Hence, data required for supervised machine learning training are accessible, which is critical for the adoption of the system at scale.

Similar to *MalClassifier*, we use *Zeek* to perform the network flow reassembly, utilising features provided in *conn.log*, one of the logs generated by Zeek. Before building the Markov Chains, we split the traffic to windows of flows of length n. Specifically, in network security applications, the length of the flow window determines how often network flows are sampled for detection. Usually, in such systems, the traffic is separated to *time* windows, meaning the system reports detections in a fixed amount of time (e.g. 5 min). Instead we split the *conn.log* to sub-logs each containing n number of flows (bursts of n-flows). Bots are known to send traffic in bursts [209, 210]. For example, bots used to launch DDoS attacks send huge amounts of requests to overload systems and networks, seeking to make a system or network resource unavailable. Similarly, bots tend to send spam emails in bursts. For such attacks to be successful, sending network bursts is inevitable. We will explore the burstiness property of malware network traffic in our dataset and discuss evasion challenges in Section 9.5.3.

9.2.2 Connection States Extraction

To build a Markov Chain, we need first to identify its states. Hence, we use as states the behavioural features provided by Zeek in the *conn.log*, some which were previously used for malware family classification [1]. We build two types of Markov Chain models: one using conn_state as states and one that is a combination of conn_state, service, and protocol. It is worth noting that we avoid using statistical features that may be susceptible to malware evolution [212]. We describe the two type of states used in the following.

Conn_state — Zeek represents the network flow connection state (*conn_state*) using 13 different states, described in Table 9.1. *Conn_state* depicts the status of a bot's communication flows. For example, if a bot sends a TCP scan that was rejected by the receiver, the connection state will be *REJ*.

Protocol, Service, and *Conn_state* (PSC) — PSC is a composite of the *conn_state* attribute, connection network protocol (e.g. TCP/UDP/ICMP)

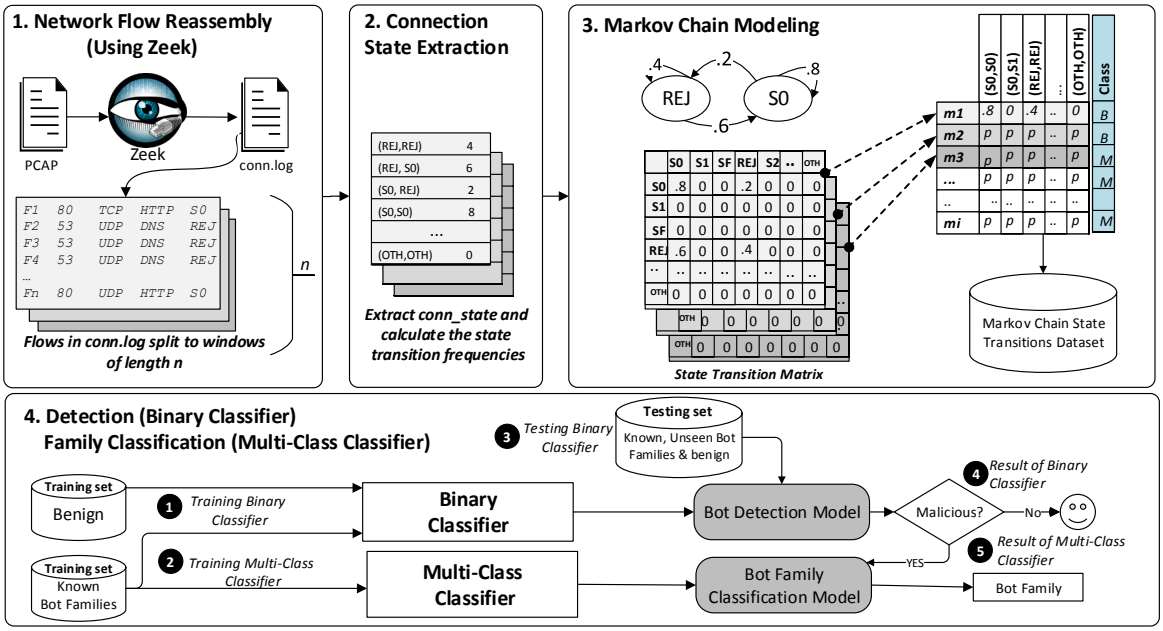

Figure 9.1: BOTection System.

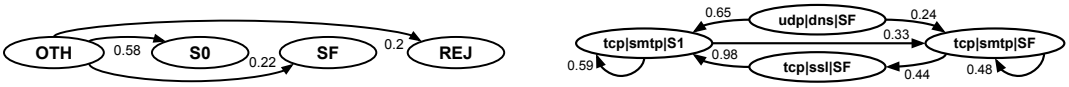

Figure 9.2: Markov Chain for Neris ICMP port scanning (left) and Zeus C&C (right)

and application protocol or service (e.g. DNS). An example of such a state is $(udp|dns|S0)$.

Conn_state provides a more abstracted representation of the flow, while PSC provides more granularity to the flow behaviour. Although the *PSC* feature might provide a more detailed representation of the network flow behaviour, we need to consider the number of states for the Markov Chain model. *Conn_state* could have 13 different values, resulting in $13^2 = 169$ Markov Chain state transitions used as features. In contrast, PSC may have various combinations of letters, resulting in over 14641 state transitions.

For each type, we extract the state transitions and calculate their frequency (number of times the state transition occurred within a window n). A state transition represents the connection states of two consecutive network flows in the log. As shown in Figure 9.1, the connection state transition from f_1 to f_2 is $S0 \rightarrow REJ$ (for *conn_state*) and $tcp|http|S0 \rightarrow udp|dns|REJ$ (for *PSC*). We store these state transitions for each sub-log as a key-value pair (i.e. key = (S0, REJ) and the value is the frequency = 2). These state transitions are extracted from flow headers and are thus resilient to encryption and privacy preserving — fulfilling design goals ❷, ❸.

9.2.3 Markov Chain Modeling

Markov Chains model sequences of events named states $s \in S$, where S is the set of all possible states. Markov Chain models are represented as a set of nodes (states) and edges between the nodes labelled with the corresponding transition probabilities. Markov Chains are memoryless, meaning the transition from state s_i to state s_{i+1} is dependent only on the current state s_i. The transition probabilities are calculated by observing the frequencies of each transition from each state $s_l \in S$ to state $s_j \in S$, normalised by the total number of transitions that start from node s_l. The sum of all the transition probabilities of the edges starting from any node (e.g. s_l), is one. The number of transition probabilities is the square of the number of states, as each state can be connected to all other states, including itself.

As an example, in Figure 9.2 we illustrate the Markov Chain of network communication for the *Neris* bot family using *conn_state* as the feature type. *Neris* performs port scanning; thus the sequences start from an OTH packet to get to either S0 (attempt to establish a connection, no reply), or REJ (connection rejected), or SF, where the connection is established. The probabilities of transition are given by the normalisation of transition occurrences between OTH and the other three states. The graph demonstrates the port scanning behaviour of this family, where connections were attempted and sometimes was successful ($P = 0.2$), sometimes rejected (with similar probability for this sample) and in all the other cases were ignored ($P = 0.57$).

BOTection uses Markov Chains to generate the feature vectors used for classification. Using the state transition frequencies (key-value store), we compute the transition probabilities. This is represented as the state transition matrix for each sub-log. We then represent each sub-log as a vector with all possible state transitions (i.e. keys from the key-value store) representing the features (i.e. columns). The feature vectors of all sub-logs represent our dataset. For example, if the feature type is *conn_state*, then the states in the Markov Chain model are those defined in Table 9.1. The 13 different *conn_state* states result in 169 transitions. Hence, for each sample sub-log, the probability of transition from one state to the other in the Markov Chain represents the feature vector for that sample.

9.2.4 Classification

Using feature vectors of Markov Chain state transitions belonging to bot and benign samples, we train supervised machine learning models. Particularly, we use an ensemble classifier known to overcome over-fitting issues. Since our system carries

Table 9.1: Description of the flow connection state (*conn_state*) feature obtained from Zeek *conn.log* logs.

State	Description
S0	Connection attempt seen, no reply.
S1	Connection established, not terminated.
SF	Normal establishment and termination.
REJ	Connection attempt rejected.
S2	Connection established and close attempt by originator seen (but no reply from responder).
S3	Connection established and close attempt by responder seen (but no reply from originator).
RSTO	Connection established, originator aborted (sent a RST).
RSTR	Responder sent a RST.
RSTOS0	Originator sent a SYN followed by a RST, but did not receive a SYN-ACK from the responder.
RSTRH	Responder sent a SYN ACK followed by a RST, we never saw a SYN from the originator.
SH	Originator sent a SYN followed by a FIN, we never saw a SYN ACK from the responder.
SHR	Responder sent a SYN ACK followed by a FIN, we never saw a SYN from the originator.
OTH	No SYN seen, just midstream traffic.

out two subsequent tasks (i.e. bot detection and bot family classification), we adopt two different classification approaches discussed in the following.

Binary Classifier (Bot Detection Model) — Binary classification aims at partitioning the samples in the feature space in two distinct and complementary sub-spaces. The partition is done in such a way that a sub-space includes samples of a class while the complementary sub-space includes the samples of the other class. In our system, the classifier is trained using n-flow samples of known malware families and benign traffic. The trained model is then used to classify an n-flow into either the bot class (i.e. Malicious) or the benign class (i.e. not Malicious).

Multi-Class Classifier (Bot Family Classification Model) — Since our family classification task aims at discriminating between malicious flows of different bot families, we rely on a classification approach that partitions the feature space in multiple non-overlapping sub-spaces. The multi-class classifier in our system is trained using malicious n-flows belonging to multiple families. Thus, once an n-flow is classified as malicious by the *Bot Detection Model*, it is then attributed to a malware family by the *Bot Family Classification Model*.

9.3 Evaluation

We implement the system as a Python 2.7 application using the *Scikit-learn* library. [1] The experiments were conducted on a 40-core processor with 128GB RAM and 20G disk space, Centos OS. We open-sourced the implementation of our system as a contribution. [2]

9.3.1 Experiments

We design the following experiments to determine BOTection's capability in modelling bot network communication, that can be used for bot detection and family classification.

Bot and Benign Communication Patterns (results in Section 9.4.1) — We model the bots' and benign communications as a Markov Chain to identify bots' network behaviour and explore the discrepancies in bot and benign state transitions.

Bot Detection (results in Section 9.4.2 and Section 9.4.3) — We build and evaluate two bot detection classifiers to: ❶ detect known bots and ❷ detect previously unseen bots. The objective of the unseen bot classifier is to assess the classifiers' generalisability in detecting communication for bot families that were *not* considered in classifier training. We also attempt to answer the question '*What if a few wrong flows end up within the window n?*' Hence, we study the effect of injecting benign flows into the malicious n-flows on the system detection.

Bot Family Classification (results in Section 9.4.4) — We identify network communications that are unique to each bot family. Thus, once an n-flow is identified as malicious by the bot detection classifier, we evaluate the multi-class classifier's accuracy in classifying these *malicious* n-flows to a bot family.

9.3.2 Datasets

We use two datasets to evaluate the binary classifier (bot detection) and multi-class classifier (bot family classification). In Table 9.2, we show the number of malicious flows (both datasets) and benign flows (ISCX only) and the percentage of flows belonging to each family. In Table 9.3, we also show the bot families attack network communications in the dataset used to train the multi-class classifier. These datasets were described in more detail in Chapter 4.

ISCX Botnet Dataset — for Binary Classifier— This dataset contains malicious and benign traffic and is split into training and testing sets with 5 and 13

[1] Scikit-learn: machine learning in Python - https://scikit-learn.org/
[2] https://github.com/balahmadi-Ox/botection

Table 9.2: Datasets used for evaluation - *ISCX contains bot and benign flows used in binary classifier, MCFP contains bot traffic only for multi-class classifier.*

Dataset	#Flows	Bot Families
ISCX Botnet Dataset	Training - 6,925,812 flows - 34.97% Malicious Testing - 5,040,068 flows - 44.97% Malicious	***Training:*** Neris (12%), Rbot (22%), Virut (0.94%), Zeus (0.01%), Zeus C&C (0.01%) ***Testing:*** *Known Bots* — Neris (5.67%), Rbot (0.018%), Virut (12.80%), Zeus (0.109%), Zeus C&C (0.006%) *Unseen Bots* — Menti (0.62%), Sogou (0.019%), Murlo (0.16%), Tbot (0.283%), Zero Access (0.221%), Weasel (9.25%), Smoke Bot (0.017%), ISCX IRC Bot(0.387%)
MCFP	7,283,060 Bot flows	Miuref (2.8%), Zeus (3.5%), Geodo (3.4%) NSIS (0.3%), Bunitu (2.7%), WannaCry (0.9%),Conficker (0.8%), Zeus C&C (8.6%), Neris (2.8%), Rbot (4.5%), Virut (1.23%), Sality (20.7%), TrickBot (0.5%), Stlrat (5.3%), Donbot (0.07%), Papras (41.8%), Qvod (0.1%)

bots respectively, where eight bot families in the testing set were not used to train the classifier (*unseen bots*). We use this dataset to train and evaluate the binary classifier *using malicious and benign network traces*, in detecting unseen bots. This is due to the testing set containing bot families that were not present in the training set.

Stratosphere IPS Project — for Binary Classifier— This dataset is used to verify the binary classifier's performance over time, thus determining the date the samples were first in the wild was essential. The samples date of "*first seen in the wild*" was verified using VirusTotal[3] reports.

Malware Capture Facility Project (MCFP) and Stratosphere IPS Project — for Multi-class Classifier— To train our multi-class bot family classifier, we used the CTU-13 botnet traffic dataset provided by the Malware Capture Facility Project [183]. In addition, we use the samples of newer bots (e.g. *Trickbot - 2016*) and ransomware (e.g. *WannaCry - 2017*) collected by *The Stratosphere IPS Project*. The multi-class classifier classifies detected malicious traffic to a family; therefore, only *malicious* traffic (C&C communication and bot communication) is used to train and evaluate the classifier. The botnet families represented in the dataset employed various protocols (e.g. *IRC, P2P, HTTP*) and various techniques (e.g. *spam, DDoS*), as shown in Table 9.3.

[3] `www.virustotal.com`

Table 9.3: Bot Network communication used to train the multi-class classifier *(PS: Port Scanning, NS: Network Scanning, FF: Fast-Fluxing, CF: ClickFraud).*

Family	Propagation		C&C			Operational		
	PS	*NS*	*DGA*	*FF*	*CC*	*DDoS*	*Spam*	*CF*
Miuref								✓
Zeus			✓					
Geodo							✓	
Bunitu			✓	✓	✓			
WannaCry		✓						
Conficker		✓			✓			
Zeus C&C			✓		✓			
Neris	✓						✓	✓
Rbot	✓					✓		
Virut	✓						✓	
Sality		✓	✓				✓	
Stlrat						✓		
Donbot	✓							
Papras			✓			✓		
Qvod	✓							

9.3.3 Classifier Design

For bot detection, we use a binary classifier which classifies an observation as malicious or benign. For bot family classification, we use a Multi-Class classifier, meaning an observation is classified into one of multiple bot family classes. For both approaches we use Random Forest classifiers, an ensemble method that builds a strong classifier relying on the consensus among classification results from several weak learners (i.e. estimators), thus overcoming over-fitting issues of an individual learner. In our work, we consider a Random Forest composed of 101 decision trees as weak learners.

To address the class imbalance in the MCFP dataset used for bot family classification, we apply a method that balances the training set by associating a weight to samples of a specific class according to the number of samples that class includes. Hence, it associates a higher weight to samples of under-represented classes. In the Scikit-learn library, Random Forest classifier handles imbalanced classification problems by setting the parameter $class\_weight = balanced$.

Window Flow Size— The window flow size represents the number of flows in each sample in our dataset. For each sample *conn.log* in our dataset, we split the log into several sub-logs of size n. Therefore when the window size $n = 15$, that means each observation in our final dataset contains bursts of 15 flows (15-flows for short). We show the cumulative number of flows generated by a bot-infected host per bot family during the first 7 minutes in Figure 9.3. All bots generate 15 or more flows within less than 170 seconds, then continuously send over 15 flows

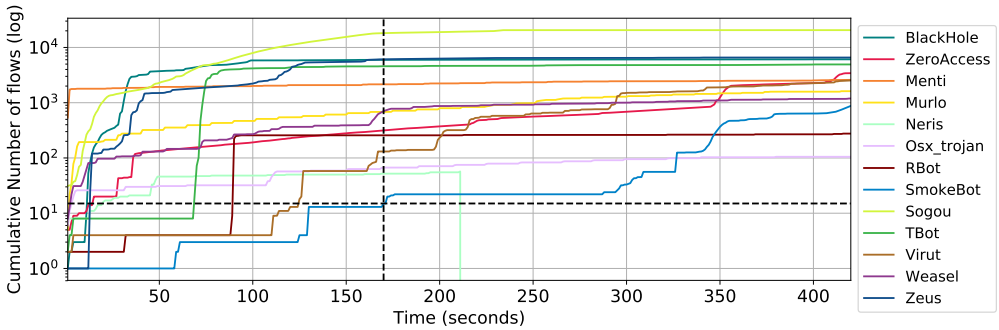

Figure 9.3: Cumulative number of flows (log scale) generated by each bot family every second over a time frame of 7 minutes.

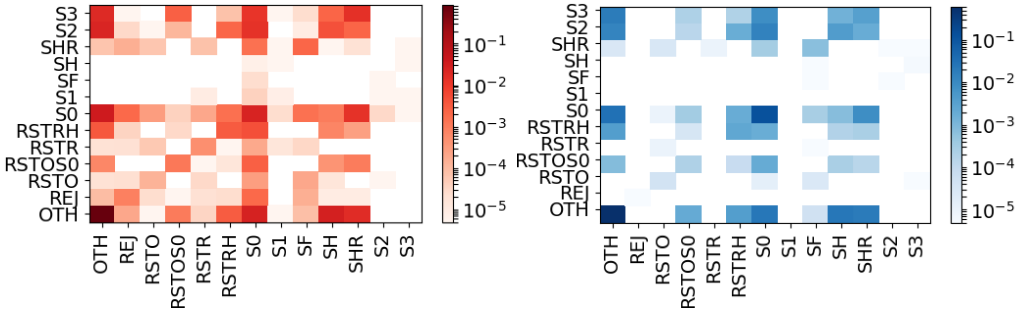

Figure 9.4: Average connection state transitions for network communications of bot (red-left) and benign (blue-right) samples in ISCX dataset.

each second. This confirms our system design hypothesis that bots send traffic in bursts. To explore the ideal window size, we experiment with various configurations for the window n, where $n = 10, 15, 20, 25, 30, 35$.

Classifier Performance— To assess the performance of the known bots binary classifier and multi-class classifier, we apply stratified 10-fold cross-validation. For the unseen bot classifier, we use the training set to train the classifier and evaluate using the testing set containing samples of bot families not considered in training. We employ the evaluation measures of *Precision*, *Recall* and *F-measure* to evaluate classifiers' performance. For the binary classifier, we also measure the *False Positive Rate* ($FPR = FP/(FP + TN)$) and the *False Negative Rate* ($FNR = FN/(FN + TP)$).

9.4 Results

We present in the following sections the results for the experiments identified in Section 9.3.

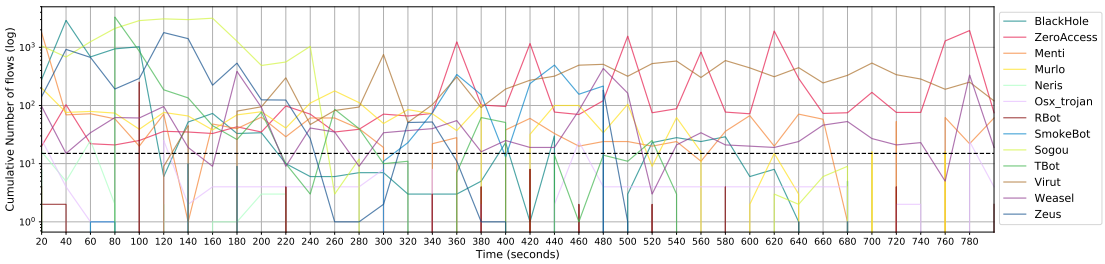

Figure 9.5: Number of flows generated by each malware family every 20 seconds.

9.4.1 Bot and Benign Communication Patterns

To understand the discrepancies of benign and bot network flows, we visualise the average Markov Chain *conn_state* state transitions in Figure 9.4 for bot and benign samples. Benign state transitions are concentrated on a few state transitions compared to bots, which are more diverse. There are similarities in both, which is due to how the Internet works and similarities in the traffic. For example, although both have a high transition probability for $OTH \rightarrow OTH$, the diversity of the states in a bot's n-flow makes it detectable. Bots generate more unique connection state sequence behaviour that benign traffic does not attempt.

Bot Traffic Over Time— We show the overall amount of traffic by each bot family in Table 9.2 over time in Figure 9.5. In particular, the y-axis has a logarithmic scale, and each data-point consists of the traffic in an interval of 20 seconds. We can notice that the majority of the bot families constantly produce a significant amount of network flows. This means that a bot is more likely to be detected in different phases of its malicious operations since our method has available an abundant quantity of n-flows for the classification.

To understand the various communications a bot performs during propagation, C&C communication, and other attack operations, we modelled the communication as a Markov Chain. We discuss the most prominent bot network communication found in the MCFP dataset. To simplify the Markov Chain for visualisation, we show only the state transitions with probability $p > 0.1$.

C&C Communication— Communication with the C&C is the most important communication for the bot operation, as its how they receive commands. To show how that can be captured as a Markov Chain, we show Zeus's C&C communication in Figure 9.2. Zeus is known to initiate the communication to its C&C by sending a *GET* message, which then the C&C server replies with the encrypted configurations which Zeus will use to send to encrypted stolen information through a *POST*

Table 9.4: False Positive Rate (FPR), False Negative Rate (FNR) of the known and unseen classifiers for each value n.

Classifier	**n**	**conn_state**		**PSC**	
		FPR	*FNR*	*FPR*	*FNR*
Unseen Bots	10	0.004	0.111	0.353	0.256
	15	0.060	0.096	0.386	0.055
	20	0.100	0.074	0.407	0.056
	25	0.093	0.073	0.175	0.043
	30	0.079	0.065	0.142	0.030
	35	0.097	0.067	0.142	0.026
Known Bots	10	0.003	0.000	0.002	0.003
	15	0.003	0.001	0.001	0.002
	20	0.004	0.001	0.001	0.008
	25	0.008	0.001	0.003	0.001
	30	0.004	0.001	0.000	0.000
	35	0.001	0.000	0.002	0.009

message [213]. This can be seen in the Markov Chain with the HTTP connection with state SF followed by an SSL connection with $p = 0.44$. Following the SSL state, the bot establishes a connection with the C&C ($p = 0.98$) that is terminated by the bot.

SMTP Spam— Multiple bots in our dataset send email SPAM, specifically *Neris*, *Sality* and *Geodo*. For brevity, we show the average of all the Markov chains for the Geodo bot performing email SPAM in Figure 9.12. We found 29,575 Simple Mail Transfer Protocol (SMTP) flows in our dataset for Geodo bot. Overall, there are 2 types of SMTP flows: (1) f = conn_state:S1, history:ShAdDa, (2) f = conn_state:SF, history:ShAdDafF. Meaning, the first was not terminated, whilst the second was terminated by a FIN bit sent by both ends. These two types were captured in the Markov Chain model, showing the next state after a completed SMTP connection is a DNS connection ($P = 1$), whilst an unterminated connection will (1) either remain in the same state ($P = 0.41$), (2) receive a SYN Ack followed by a FIN from destination ($tcp \mid - \mid SHR$)($P = 0.27$), (3) make a DNS request ($P = 0.32$). Other bot families had similar state transitions, although *Neris* had some SMTP connection with the conn_state set to RSTR and history *ShAdDaTfrr*, indicating the destination sent a RST.

Port Scanning— We were also able to identify ICMP port scanning (Figure 9.2), where an ICMP connection flow with *conn_state* OTH is followed by a UDP flow with states $S0$, or a TCP connection with *conn_state* SF or REJ. Bot families *Rbot*, *Neris*, *Virut*, *Qvod*, and *Donbot* all perform port scanning and have shown to have similar state transitions.

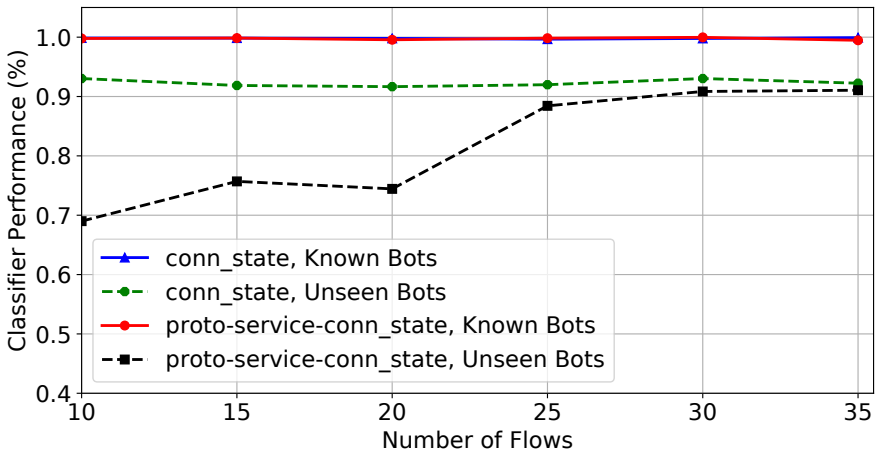

Figure 9.6: Binary classifier F-measure for detecting known/unseen bots for each state type.

DDoS— We are also able to observe two types of DDoS attack, TCP and ICMP DDoS in *Rbot*. DDoS was observed as multiple connection requests sent to a single host, where these requests could be an ICMP request or TCP request. For ICMP DDoS, the bot establishes an ICMP request which has a *conn_state* OTH, whilst TCP DDoS connections have *conn_state* REJ.

9.4.2 Performance of Binary Classifier

Classifier Performance— We show the performance of the binary classifiers in detecting known and unseen bots in Figure 9.6, with the flows' *conn_state* and *PSC* as Markov Chain states. When analyzing the known bots, the flow window had no effect on the classifier performance, and the classifier performed well (99% F-measure). However, when we try to identify unseen bots the detection performance varies based on the window size: larger window sizes allow a better identification in the *PSC* while smaller ones work slightly better for *conn_state*.

Overall, *conn_state* works best for the goal of detecting network connections belonging to unseen bots. Specifically, for detecting unseen bots, *conn_state* had the best performance with a detection rate of 93.03%, when $n = 15$. In contrast, using PSC as a feature yielded a better accuracy (91.06%) when ($n = 35$). We also report the FPR and FNR for each classifier in Table 9.4. The Unseen Bot Classifier performed best when $n = 10, n = 15$ with 0.4% and 5.87% FPR respectively. The FPs originated from only a small number of hosts and for specific cases. We discuss this in more detail in Section 9.5.1.

Classifier Performance Based on the Type of Feature— Clearly, *conn_state* performance was best in detecting known (F-measure= 99.78%), and unseen bots

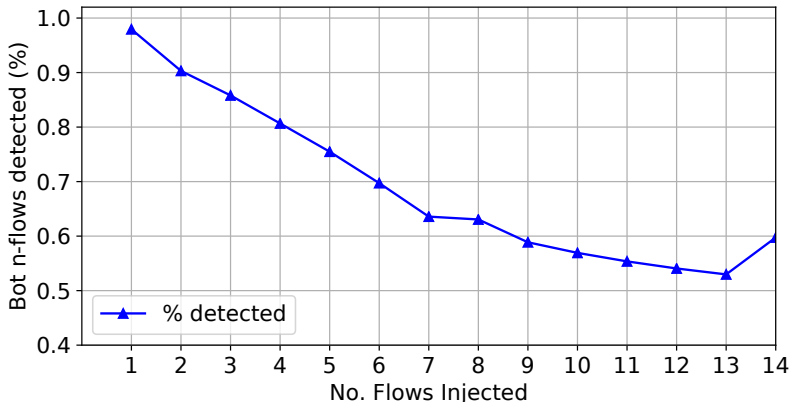

Figure 9.7: Percentage of bot 15-flows detected when we inject b benign flows in the sequence.

(F-measure= 93.03%) compared to PSC where detection efficiency was high for unseen bots only when the burst window size was above 20.

Classifier Performance When Injecting Benign Flows— We explore the effect of injecting random benign flows b into the bot 15-flow ($n = 15$) at random locations on the detection performance. In practice, this means that we inject states that are randomly selected among the most frequent *conn_states* in benign traffic, as reported in Section 9.4.1 and shown in Figure 9.4. For example, *REJ conn_state* was not injected in this experiment as it rarely occurs in benign traffic.

In this experiment, we injected benign flows (i.e. b random benign *conn_states*), in the malicious 15-flows. For example, for 15-flow of malicious flows, we can inject 2 benign flows ($b = 2$), meaning that the other 13 flows are malicious ($m = 13$), where ($n = b + m$). We conducted experiments with various configurations of b and m. For each configuration, we repeated the experiment 40 times and obtained the average of the number of detected malicious flows by the *known bot classifier*.

We show the results of this experiment in Figure 9.7. The x-axis represents the number of benign flows injected b. For example, when less than four benign flow was injected ($b = 4$, $m = 11$), the classifier was capable of detecting 80% of the 15-flows even after injecting a random state into the sequence. This may be due to the diversity of the states in a bot n-flow, as discussed in 9.4.1.

9.4.3 Classifier Performance Over Time

Machine learning detectors need to maintain their performance over time to facilitate real-world application. This is crucial to prevent the costly re-training of classifiers and to ensure its detection capability to new malware. As such, we conduct an experiment using current malware from the Stratosphere IPS dataset.

Table 9.5: Binary Classifier Performance: F1 (F-measure), Precision (P) and Recall (R) over time with *Temporal Training Consistency*. We train the classifier using bot samples first seen in the years precedent of the samples in the testing set.

Training set	**Testing set**											
	2015			*2016*			*2017*			*2018*		
	F1	P	R	F1	P	R	F1	P	R	F1	P	R
Pre 2015	.86	.89	.87	.87	.89	.87	.86	.89	.87	.86	.89	.87
Pre 2016	-	-	-	.987	.987	.987	.983	.983	.983	.982	.982	.982
Pre 2017	-	-	-	-	-	-	.981	.982	.981	.981	.981	.981
Pre 2018	-	-	-	-	-	-	-	-	-	.989	.989	.989

In training and testing the classifier, we take into consideration the *Temporal Training Consistency*, where all samples in the training are temporally precedent to samples in the testing set [214]. Thus we split the test set, depending on the year the malware was "*first seen*", which was verified from *VirusTotal* reports.

We show the results of training the binary classifier ($n = 15$) in Table 9.5. Interestingly, the classifier's performance over time is 86% when trained with samples appearing before 2015. However, the classifier performance increases to 98% when trained with samples appearing after 2015. These results show that the binary classifier generalises well enough not to be affected by the temporal bias problem defined in [214]. This means that our classifier (i) can detect traffic from previously unseen bot families, and (ii) is robust over time; thus, it does not need to be frequently re-trained.

9.4.4 Performance of Multi-Class Classifier

Markov Chain Models— We show the Markov Chain state transitions for the *conn_state* feature in Figure 9.9 for some bot families. Highlighting only the most important state transitions for each bot family, we show in the figure only state transitions with probability > 0.6. We notice that connection transition $SF \rightarrow SF$ and $S0 \rightarrow S0$ are seen in most bot connections. Having an $SF \rightarrow SF$ is expected, as bots will typically have normal connections.

Classifier Performance— We show in Figure 9.8 the performance of the multi-class bot classifier for each of our connection state types. When the flow window size is at least 20, the scores are stable and close to 100. Our classifiers performs well, reaching 98% F-Measure score for bot family classification when $n = 20$, reaching its maximum 99.09% when $n = 35$. The *conn_state* feature performs very well independently from the window size value, while the other feature sets had these performances with $n \geqslant 20$. We show the classifier's performance in more detail per

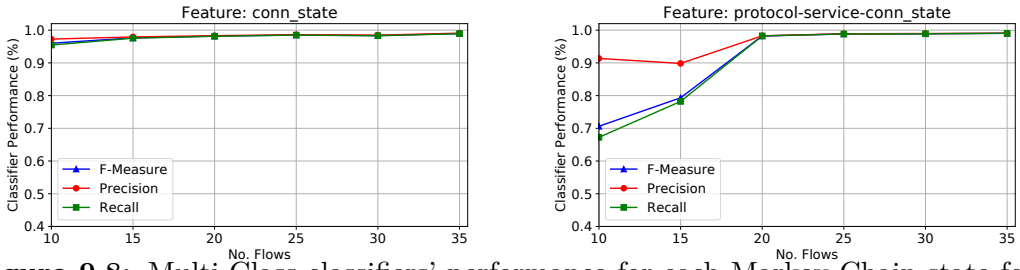

Figure 9.8: Multi-Class classifiers' performance for each Markov Chain state feature type.

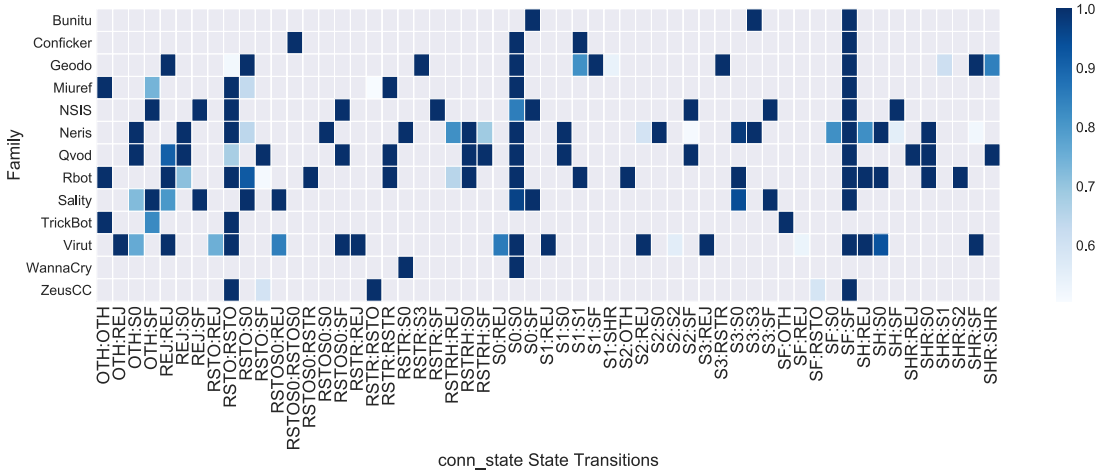

Figure 9.9: Markov Chain Model Transition Probabilities of the *conn_state* feature for each botnet family. For brevity, we show only transition probabilities greater than 0.6. For example, Virut had a high transition probability between states S1 and REJ.

bot family when $n = 20$ in Figure 9.10. As discussed in 9.3.3, we take precautions to ensure that the classifier is not impacted by the imbalanced training set. This is shown in Figure 9.10, as the classifier maintained a high precision per bot family. We observe that the *Geodo* bot is the only family presenting an F-Measure score less than 90%, with some connections misclassified to either *Bunitu* or *Stlrat*.

Classifier Performance Based on Type of Feature— *Conn_state* performed the best across the various window sizes; however, we find that PSC might provide a more complete representation of a bot's behaviour. For example, we observed for *Virut* that state transitions from *S0* to *S0* for TCP flows were of a spam attack. However, when the bot used a UDP connection for DNS services ($udp|dns|S0 \rightarrow udp|dns|S0$), it indicated that it is sending DNS queries. While these differences are transparent to the *conn_state* features, they are shown by the PSC. The email spam was confirmed when we uncovered that the destination port was port 25. Hence, in certain attacks, destination port, service, and protocol are crucial to distinguish types of attacks with similar state transitions.

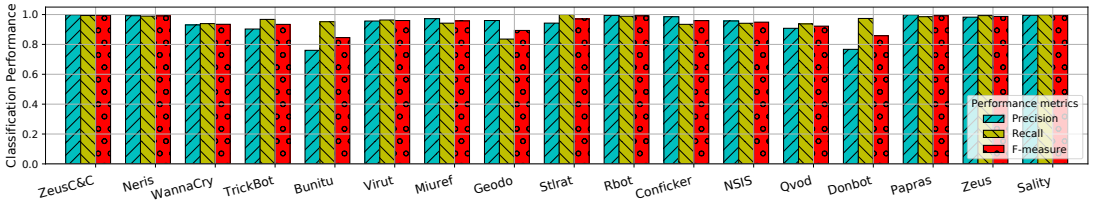

Figure 9.10: Multi-Class classifiers' performance per bot family (n=15).

9.5 Discussion

We discuss in the following sections the reasons behind classifiers' misclassifications, lessons learnt from our experiments, potential methods malware can deploy to evade detection and system limitations.

9.5.1 Understanding Classifiers' Errors

Binary Classifier: False Positives— We focus our discussion here on the performance of the binary classifier (for unseen bots). The classifier misclassified 20561 (5.87%) benign n-flows of the test set as malicious. In a real-world setting, a defender (e.g. analyst) needs to know which are the infected hosts, to remove them from the network. The process of investigating FPs is known to be a tedious task for analysts [209]. However, we found that the falsely-labelled flows originated from only a small number of hosts ($FPR > 0.1$) — 36 out of 1164 benign hosts, as shown in Figure 9.11.

For example, benign host IP = 10.0.0.254 had the highest percentage of FPs, where 87% of its benign traffic was flagged as malicious. The misclassified n-flows had either only *conn_state* OTH or S0 — a state transition that we found to be high in both benign and malicious (Figure 9.4). The FPR could be improved by white-listing connection states that are highly present in benign and malicious flows or n-flows containing only one kind of state (e.g. only OTH). Therefore, although the FPR might look high, investigating these false positives is a rather reasonable load for a security defender.

Binary Classifier: False Negatives— We attempt to understand the misclassification of bot flows to benign of the *unseen* binary classifier. In Table 9.6, we show the number of false negatives for each bot family, where $n = 15$. Over the $136,288$ bot 15-flows in our testing dataset, 8124 were misclassified as benign. Interestingly, *Virut* bot produced the highest number of False Negatives, although the bot family was considered in the training set. 99.87% of the misclassification for *Virut* are 15-flows with three connection state transitions (4 $RSTRH \rightarrow S0$, 4 $S0 \rightarrow RSTRH$, and 7 $S0 \rightarrow S0$).

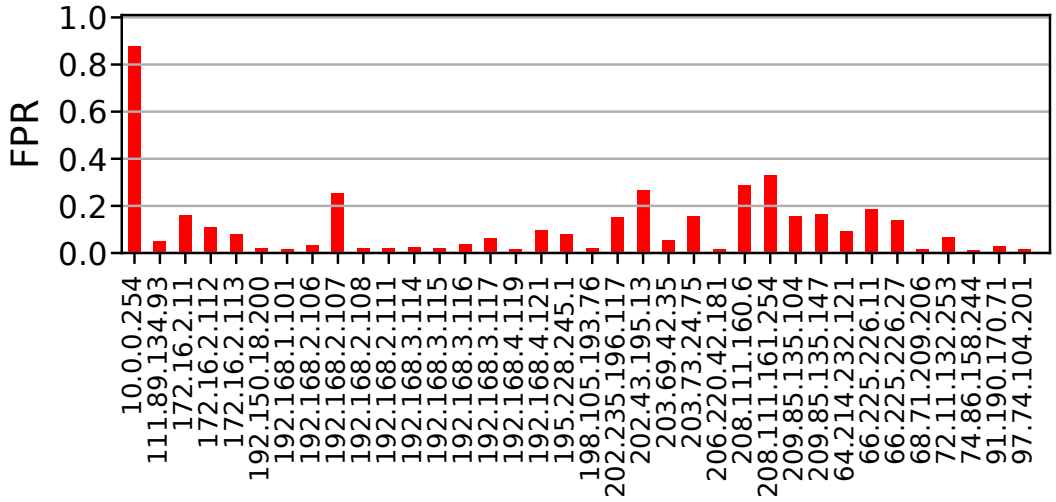

Figure 9.11: Host IPs that produced the highest FPR for the binary classifier (*FPR* > *0.1%*).

Table 9.6: Number of flows, 15 window length samples, and number of bots per family in the ISCX testing dataset, and the number of False Negatives (FN) for each bot family and benign traffic.

Family	**Total # Flows**	**# 15-flow window**	**# Hosts**	**# FN**
IRC Attack	773370	51558	8	107
Menti	13024	1628	1	8
Murlo	81135	5409	1	68
Neris	402000	2680	1	28
Osx_trojan	375	25	1	0
Rbot	116505	7767	1	10
SmokeBot	1755	117	1	0
Sogou	20625	1375	1	1
TBot	5205	347	4	5
Virut	437550	29170	1	7726
Weasel	508080	33872	2	165
ZeroAccess	19440	1296	2	9
Zeus	15660	1044	4	9
Benign	**2,948,550**	**196570**	**1164**	**16100**

These misclassified connection states represent *Virut's* IRC communication and spam behaviour. *Virut* sends IRC communication to port 6667 with $conn\_state = RSTRH$ followed by sending spam on port 25 with $conn\_state = S0$. Such state transitions were not seen in the training set for *Virut* and other bot families, while present in 57% of the testing set explaining the misclassification.

On examining these misclassified flows, we found that in our training set, Virut attempts to establish an IRC connection by sending packets to a server that either rejects the connection (REJ or $S0$) or accepts the connection (SF). In contrast,

the testing set contains unsuccessful IRC connections ($S0$) to the server followed by a $RSTRH$ response from the server (meaning the connection was timed out). Hence, this is not a result of Virut changing its behaviour but was a result of the server delay in its reply and connection timing out.

Multi-Class Classifier— We attempt to investigate the misclassification (when $n = 20$) per bot family. *Geodo* had the lowest F-Measure score of 84%, with some 20-flows classified to *Bunitu* or *Sltrat*. Most flows misclassified to *Bunitu* are connections with only state transition $SF \rightarrow SF$. This explains the misclassification, as the classifier learned that 20 consecutive SF flows is a behavior of *Bunitu*, resulting in the low classification accuracy for *Geodo*. However, as $SF \rightarrow SF$ state transition represents a normal network connection, n-flows with only connection state SF should be white-listed. Similarly, *Geodo* flows with only $S0 \rightarrow S0$ transitions are classified as *Sltrat*, which has the same flows.

9.5.2 Lessons Learned

Emerging bots are stealthy and may seek a passive propagation approach (e.g. *spear phishing)*, avoiding noisy active propagation (e.g. *scanning*) that some previous bot life-cycle detection approaches (e.g. [82, 85, 215, 216]) expect. In Figures 9.2 and 9.12 we showed the Markov Chain for various bot communications such as C&C communications, ICMP scanning and email spam respectively. Each bot attack resulted in a different Markov Chain representing that attack. These models are not only valuable in understanding bot behaviour but may be used to determine bot attack stages and whether bots do actually follow the *bot life-cycle*.

Figure 9.12: Markov Chain for Sality (*right*) and Geodo (*left*) spam behavior (only state transition probabilities, where $p > 0.1$).

Although bots share similarities in their Markov Chains, there are still differences that make it possible to distinguish between the network traffic of different bot families. *BOTection* was able to classify a n-flow to its bot family with 99% accuracy — fulfilling design goal ❹. Each bot family had a set of uniquely identifiable Markov Chain state transitions (Figure 9.9). In fact, bots may operate the same attack in different ways. For example, Figure 9.12 shows the behavior of two spam families, *Geodo* and *Sality*. They both successfully use the TCP SMTP protocol ($tcp|smtp|SF$)

and perform DNS queries through UDP ($udp|dns|SF$), but they use them in different ways. When *Geodo* successfully sends an email, it always goes back to the DNS queries ($P = 1$). However, *Sality* is often sending many emails in a row as state $tcp|smtp|SF$ has 0.64 probability of returning to itself in the next step of the chain.

Another example is the Markov Chains for *Papras* that shows TCP DNS requests, while normally DNS is sent over UDP. This may be due to the DNS response or request data size exceeding the size of a single packet. *Papras* flows in our dataset contain DGA communication; this unique DNS request behaviour is peculiar of *Papras* DGA behaviour that is captured by our system.

We explored two types of states used in the Markov Chain models. Both *conn_state* and PSC are effective in detecting bot communications, but PSC provides a higher level of granularity. For example, a flow with *conn_state SF* might be sent using either UDP or TCP, and might be an SSL communication, a DNS request, or an HTTP request, each representing a different flow behavior. However, PSC has 404 state transitions compared to *conn_state* ($13^2 = 169$) increasing the dimensionality thus the computation needed for classifier training.

We highlight the importance of evaluating the system performance in detecting bots not used when training the classifier. Our system was able to detect known bots with an F-measure of 99% and new bots with an F-measure of 93% — fulfilling design goals ❹ and ❺.

Bot Markov Chain models are different from benign ones as bots have more diverse state transitions, as seen in Figure 9.4. Hence, *BOTection* was able to capture bot behaviour and recognise it in flows of families not included in the training set. Markov Chain models are robust to bot evolution due to the abstraction of network communication to finite states, thus less susceptible to the introduction of new states. Applying *conn_state* in the Markov Chain models ensures that no additional states could be introduced.

We evaluated our binary classifier's performance over time. Following recommendations provided in [214] to overcome the temporal bias problem, we trained the classifier with samples first seen in the years preceding the test samples, complying with the *temporal training consistency*. We also ensured that samples in the test sets had been seen for the first time in the same year. As we reported in Table 9.5, our method maintained its performance over time with a 98% F-measure. This ensures that a classifier can detect new bots without frequent and time-consuming re-training.

For binary classification, we used the ISCX dataset that contains network flows labelled (malicious/benign) according to the host generating them. We studied the

effect of mixed traffic on *BOTection's* performance by injecting a benign flow in the malicious 15-flow. *BOTection* was able to still detect 80% of the malicious n-flow even when four benign flows were injected into the bots 15-flow.

As shown in Figure 9.3, bots have a "bursty attacks" nature that can be seen in network traces [209, 210]. We observed that bots in our dataset generated 15 or more flows within less than 3 minutes from their first execution. Such bots also sent a significant amount of flows every second during their various attack stages (Figure 9.5). Hence, *BOTection* is capable of detecting a bot-infected host during its early communications as detecting a single burst can lead to its discovery.

9.5.3 Evasion and Limitations

Bots may evade detection by reducing the number of network communications or increasing the time interval between them. Our detection method relies on a bot's bursts of network flows; thus, the detection of a stealthier bot may be a challenge. Moreover, fewer flows may affect the reliability of the Markov Chain, as the statistical models can change.

Nevertheless, we argue that most bot operational communications (e.g. *scanning, spam, DDoS*) inevitably generate communication bursts. In particular, a spam bot sends a high number of SMTP packets in a few seconds, while a DDoS attack misuses specific packets.

Those operations result in network communication behaviours that are different from benign user activities. Thus, a bot would not be able to carry out its operational communications while attempting to evade detection at the same time. Bots can use covert channels [217] (e.g. encoding the data in TCP/IP header bits). Another evasion technique is imitating patterns of benign communications. These patterns can be replicated only to some extent (e.g. sending flows with *conn_state* = SF) and for specific particular activities (e.g. a spam bot mimicking normal traffic results in sending fewer emails). Although some samples already implement these evasion techniques in the wild, their evasion efforts might introduce constraints on their behaviour, affecting their full functionality [206].

BOTection uses states that are protocol dependent, thus an attacker might attempt to evade the detection by changing the protocol (e.g. from TCP to UDP). However, changing the protocol limits bot capabilities. A bot would not be able to carry out DDoS attacks with a specific method protocol different from the original one (e.g. ICMP ping of death, UDP Flood, SYN packets on TCP handshake). ClickFraud uses HTTP; thus, it is dependent on the TCP protocol. Despite SMTP

Table 9.7: Comparison of previous Bot Detection Approaches and proposed system BOTection

Approach	Type of Detector	Encryption Resiliency	Single Bot Detection	Unseen Bots
BotSniffer [78]	C&C	N	N	N
BotFinder [84]	C&C	Y	Y	N
DISCLOSURE [79]	C&C	Y	Y	N
BotHunter [82]	Operation	N	N	N
BotMiner [81]	Operation	Y	N	Y
Abaid et. al [85]	C&C	N	N	Y
Abaid et. al [216]	Operation	N	Y	N
BOTection	**C&C Operation**	Y	Y	Y

using TCP, UDP is also used in rare benign applications so can be easily detected by a rule-based system. Bots are known to have similar attack objectives, thus attempting to change its network behaviour will be detected as its new behaviour may be similar to another family's attack behaviour.

In evaluating our system, we considered its performance in detecting traffic generated by bot families that were not included in the training set. Our system was able to detect such traffic with a 93% F-measure due to these bot families exhibiting similar attacks (e.g. DDoS) to families that are present in the training set. Hence, unseen bots that use novel techniques (i.e. new types of attacks that use protocols in a unique way) may not be detectable in our system (e.g. zero-days).

As with other machine learning-based approaches, a malware might try to evade the proposed systems via adversarial learning techniques [218]. It might also try to mimic the patterns of multiple families to impact the classification. However, the main objective of malware family classification is to identify the one that exhibits the most similar network behaviour to assist the defender in deploying proper mitigation. In the case of a new malware family, one solution is to classify such malicious traffic to a "New Family" class using the confidence of multi-class Random Forest classifiers applied previously in [219]. Thus, classifications with a confidence level below a threshold will not be attributed to a known family but as "New Family".

9.6 Comparison to Related Work

In Table 9.7, we provide a comparative analysis on *BOTection* and previous work on bot detection. Specifically, we compare based on the type of detector, its resiliency to encryption, single bot detection capability and its evaluation of the detection performance of unseen bots. *BotFinder* [84] is a botnet detection system that monitors network traffic for C&C communication to detect bot-infected hosts.

Compared to *BotSniffer* [78], the approach detects individually infected hosts and therefore does not require correlating network activities of multiple hosts. Moreover, it does not rely on payload information but high-level network data such as *NetFlow* in the detection, thus applicable also for encrypted botnet traffic. However, BotFinder uses statistical time-related features (e.g. *time interval)* of the network flows, assuming that bots use constant time intervals between C&C communications. However such time-related features are affected by the network quality (i.e. speed) and may vary in future unseen malware variants [212].

In contrast, *BOTection* uses behavioural features that are resistant against such variations, as proven by its effectiveness in detecting unseen bots. To ensure a fair and systematic evaluation, we provide only an empirical comparison of the detection results to previous work that use the same dataset. For example, Zao et al. [220] used the ISCX dataset for bot detection yielding a true positive rate of over 90% and a false positive rate under 5%. In addition, they evaluated their system in detecting unseen bots, which, although yielding a high detection rate, had a FPR (*specifically for Weasel*) of 82%. A high FP increases the burden on analysts to filter out these false positives, thus is not adoptable in real-world settings.

In contrast, *BOTection* was able to achieve a high detection accuracy whilst having a low FPR generated from only a few number of hosts. Previous work [82, 83, 85] on botnet detection focused on detecting bot activities by modelling the bot activities using bot life-cycle, mostly using the life-cycle proposed in [82]. In the future, we plan on using the *BOTection* Markov Chain models of the various bot attack stages to evaluate whether the previously proposed bot life-cycle is still valid nowadays.

9.7 Summary

In this chapter, we presented *BOTection*, a novel topology and protocol-independent system capable of detecting and classifying flows to a bot family with a 99% accuracy. Importantly, *BOTection* models capture similar network behaviour, thus are capable of detecting unseen bot network behaviour with 93% accuracy without re-training the classifier. We evaluated our approach using the network communication of various bots performing propagation, C&C communication and attacks. Our results demonstrate the effectiveness of using Markov Chains to understand bot network behaviour for better detection.

Our research on analysing malware behaviour revealed that malware utilise similar techniques to launch their attacks. We capture this phenomenon when

analysing the behavioural features extracted from their network traffic. Hence, modelling malware common behaviour allows detection systems to identify new malware variants that use similar attack techniques. As detecting the malware is essential, attributing that activity to a malware family is also critical in SOCs to determine the best remediation. It is crucial when designing network-based monitoring tools to consider real-world deployment, where content-inspection is no longer possible due to privacy and malware obfuscation through encryption. Our proposed systems recognise these design goals, improving over existing state-of-the-art malware detection tools. To foster further research and real-world deployment evaluation, we open-source our system.[4]

In the following chapter, we will conclude by providing an overview of this thesis contributions and identifying opportunities for future research directions.

[4]https://github.com/balahmadi-Ox/botection

10 Conclusions and Future Work

Contents

In this chapter, we conclude this thesis. We start by stating our research contributions and how the research goals, identified in Chapter 1, were addressed. We then discuss the research validity, suggest future research directions and finish with final remarks.

10.1 Research Contributions

We addressed the problem of the lack of literature on SOC's technologies and processes in monitoring, detecting, and responding to malware in Chapters 5, 6, and 7. In particular, in Chapter 5, we presented the results of an online questionnaire extracting the SOC practitioners' perspectives on network-monitoring tools. The results of the questionnaire highlighted interesting research questions, further investigated using qualitative research methods.

In Chapter 6, we addressed **RQ1: How do security practitioners monitor, detect, and investigate security threats in a SOC?** We extracted the main workflow that SOCs follow, noting whether this was in an internal or an outsourced SOC. One of the main findings is the extensive reliance on humans across all SOC

workflows. We identified these human-centric tasks, discussing opportunities for automation. In addition, we identified factors that impact the SOC process, thus affecting the SOCs ability to monitor and detect threats to the organisation.

In Chapter 7, we address **RQ2: What are the advantages and disadvantages that security practitioners perceive of network-monitoring tools?** We discussed the strengths and weaknesses of network monitoring tools, specifically NIDS and SIEMs. For example, we discovered that existing network monitoring tools are inefficient when monitoring internal network traffic; produce false positives which led to analysts losing respect for their tools and ignoring alarms; and produce vast amounts of logs, making it computationally challenging to correlate data.

Moreover, alarms triggered in behaviour-based technologies are more *respected* by analysts than traditional rule-based systems. Another requirement identified is to determine the type of malicious binary running on a host that triggered an alarm, not by accessing the host, but through network traffic analysis. Most organisations hire a third-party SOC to monitor their network but restrict their direct access to the hosts. Thus, the analyst is only able to observe the network traffic and not able to get access to the binary on the infected host. Based on the results of this study, we extract requirements for a malware detection system, providing recommendations for future tool development.

In Chapter 8, we addressed **RQ3: To what extent can we classify malicious network traffic into a malware family, on-the-wire?** We proposed and evaluated a behaviour-based malware family classification system that can determine the family of the malware running on a host by analysing its network traffic. Examining the network behaviour of this malware can assist analysts in determining the objective of the malicious executable. In situations where the attack implications are severe, such as in ransomware attacks, it is not only the detection of the threat that is critical — but classifying to a malware family is equally as important. This ensures that analysts can quickly react to a ransomware attack to avoid catastrophic implications. Malware's classification to a malware family is invaluable in prioritising threats and determine the best deterrence measure to employ.

In Chapter 9, we addressed **RQ4: To what extent can we detect malicious network traffic generated by new or unseen malware?** and proposed a behaviour-based bot detection system that is capable of detecting previously unseen malware. The system uses behavioural features of high-level network attributes, thus preserving the privacy of the monitored hosts. Malware's network activities (e.g. spam, DDoS, ClickFraud) are captured and modelled as a Markov Chain. Thus, not only can analysts detect new malware that has not been seen before, but they are also able to determine which malicious activities the bots are launching to choose the most effective deterrence measure.

10.2 Validation of Research Results

We employed a qualitative research method in our SOC study, and as a result, our sample size is small (n = 21). SOCs are distinctive, in their size, structure, operations and personnel and our study was only involved seven SOCs. Most of these SOCs and their analysts were based primarily in the UK, with some serving UK-based public and private sectors. Moreover, due to our non-random method of recruitment that leveraged institutional relationships to ensure participation and engagement, the type of participating SOCs were mostly in the security sector. As we report only themes highlighted by participants in this study, other themes may emerge in conversation with other SOC analysts. Hence, the aforementioned limitations prevent extracting generalisable conclusions, and therefore, require further validation of the study findings. In future work, we plan on validating the findings emerging from this study with further SOCs and participants residing in other countries or serving organisations of various sector types. Our data analysis focused on deriving recurring themes in both in-house and out-sourced SOCs. However, our study findings highlighted a clear distinction between in-house and outsourced SOCs operations, people, and technology, therefore, a more focused study targeting each type individually might result in deriving further themes and contradictions that were missed in this study.

Evaluating machine learning systems highly relies on the quality of the data used to train them. Consequently, malware research requires the use of real-world data that reflect current security threats. However, the absence of good quality real-world datasets of malware and benign network traffic continues to be the biggest challenge for malware security researchers. Moreover, executing malware samples and collecting its network traffic, while taking precautions to ensure malware does not cause harm and maintaining the validity of its network behaviour is a challenging and troublesome process. As a result, in addition to the scarcity of available datasets, those available tend to contain samples of old malware that does not represent current advanced threats. In our malware detection and classification research, we used three datasets to validate system effectiveness and performance of all datasets collected by researchers in a controlled lab environment. To the best of our ability, we ensured that datasets used were collected according to best practice in maintaining the authenticity of the generated malware traffic and included malware samples of more recent threats such as Emotet and WannaCry. Nevertheless, proper validation of such systems still requires real-world datasets to have a more realistic evaluation of a system. To foster further research and validation of proposed systems, we open-sourced our system implementation for real-world deployment and evaluation.

One of the main findings of our SOC study is the importance of designing systems that are human-machine collaborative, where technology works together with SOC practitioners for the detection of threats. The SOC operations are highly human reliant, where technology is there to present the information for the practitioners for them to make the intelligent decision. Moreover, most technologies in SOCs rely on humans to configure and tune false positives or benign triggers according to the networks and systems ecosystem being monitored. Hence, when designing systems to be used in SOC environments it is important to not only validate its performance standalone but evaluate its performance when integrated into a SOCs ecosystem of various technologies, people and processes. In this work, we proposed a malware detection system based on requirements identified in our qualitative research. In future work, we need to validate the applicability of the proposed system in SOCs and its integration to existing technological methods and operations. Such research can be in the form of a user study, where SOCs integrate the proposed system in their SOC technological pipeline and analysts evaluate its effectiveness in assisting them to recognise and detect possible malware attacks. Similarly, our SOC research has highlighted possible directions of designing technological solutions for malware detection in SOCs, that should also be validated in a SOC setting. We discuss future work relating to some of these technological solutions in the following section.

10.3 Future Work

Owing to our findings that analysts can be overwhelmed by the information current tools present them with; that there is a lack of automation, particularly in identifying false positives; and that the need to prioritise ingress and egress points leaves companies exposed to threats originating from inside their networks, we suggest future work be carried out in the following areas.

Malware Network Behaviour Knowledge Graph

Graph-based approaches have been used to organise cybersecurity information for fast retrieval. Graphs provide the ability to attach contextual information to the data, providing a more grounded inference. For example, Shu et al. [121] presented a threat intelligence graph model that could be queried by analysts for threat hunting. Similarly, Holmes [164], organised logs as a graph that could be used by analysts to map the various stages of an APT campaign. These methods promote human-machine decision making, allowing the human to query these graphs and use the information to make decisions about the maliciousness of an

alarm. This supports the fast processing of alarms, allowing analysts to focus their efforts on more serious threats.

To facilitate malware human-machine monitoring and detection in SOCs, malware network threat discovery can be modeled as a graph problem. Network traffic logs, in particular, have unique properties, that make them different to other logs. For example, the sequence of network traffic, as we found in Chapter 8 and Chapter 9, provides insights on the malware behaviour. Therefore, representing the sequence in the graph is useful to infer information about a malware's network communication behaviour. Although such modeling of malware network behaviour proved effective for detection, still, real-world deployment requires incorporating contextual knowledge. As we found in Chapter 7, machine learning models need to incorporate contextual features in model training and include analysts feedback on the predictions to reduce the number of benign triggers. Hence, to facilitate a more accurate inference, contextual information about the malware (e.g. malware family name, stage of attack, CVE) and context about the infected host (e.g. OS) and subnet (e.g. business sector) can be incorporated in the graph to be used by analysts to contextualise an alarm.

SDN-based SIEM

In Chapter 7, our participants noted they are not able to monitor the internal network traffic due to storage and computational constraints. Instead, to understand network activity, security analysts turn to a semi-manual investigation of network data (e.g. firewall logs, packet capture data, network flows, system logs). As we identified in Chapter 6 a host's network connections are only looked at in the later stages of an investigation, missing out on meaningful earlier network connections that may have been stronger Indicators of Compromise (IOC).

As future work, we propose the idea of reducing the logs collected by the SIEM, hence decreasing the hay in the "haystack". Security analysts in the SOC define the SIEM alarms (e.g. use-cases) that would initiate a further investigation. Once that alarm is flagged, they then manually look at the logs to determine if the alarm is a legitimate threat or a false alarm. Since the raw data is only viewed by analysts when a SIEM alarm is generated, why not initiate network collection for a host or subnet when an alarm is triggered for that host/subnet. Hence, we propose collecting network traffic at the early stages of the investigation with the level of granularity needed to make a decision of its maliciousness.

Software-Defined Networking (SDN) is a new network paradigm that allows greater control over network entities by decoupling the control plane and the

forwarding plane, enabling the network control to become directly programmable. The network visibility and centralised reconfiguration of devices provided by SDN assist in a rapid reaction to security threats and facilitates dynamic security policy deployment.

Using the programmability capabilities of an SDN, the amount of storage needed for logs is reduced by collecting them dynamically (*evidence collection*). Thus, collect logs of hosts only where an alarm was received. This ensures that all the evidence needed by the analyst to determine the maliciousness of a triggered alarm is available without needing to run computationally intensive and slow queries. Such a system reduces the workflow overhead on analysts and allows them to focus on what they do best, threat hunting.

10.4 Final Remarks

As we found in the Target breach case study discussed in Chapter 1, the reasons behind the susceptibility of Target to the breach goes beyond the limitations of technology. Although the technology detected the threat, no further action by the SOC was warranted. There are various reasons why such errors may occur such as; analysts scepticism of the technology; the overwhelming number of alarms that lead to threats being missed or not prioritised, and issues in the communication between various teams and across locations and organisations.

Security monitoring and incident response have been around since the 1990s, still, SOCs struggle with issues related to technology people and processes and how all three SOC dimensions can work together. Our study highlighted that effective malware detection requires human-technology collaboration to identify those challenging threats. Detection of advanced threats has been attributed to analysts themselves hunting and spotting deviating patterns using the data provided by technology. Mature SOCs are more reliant on technology for triaging, with analysts having more time for hunting.

To achieve the appropriate level of automation in a SOC, technology should be sophisticated enough to identify false alarms from benign triggers—to establish whether further remediation action is warranted. Such automation will reduce analysts' monotonous tasks. Technology should also be collaborative driven, where technology produces contextual alarms that allow analysts to make informed decisions. Moreover, analysts need to be able to provide feedback on the technology threat predictions directly to the technology instead of documenting in the wiki. Such feedback can help technology "learn" through ML and AI and improve its

future predictions. Developed tools should also consider real-world requirements and challenges such as low hardware/software cost, privacy-preserving, low false alarms, security clearance obstacles. Moreover, the capability of these tools to adapt its detection capabilities to future threats and malware is critical.

The novelty of the research in this thesis is in providing a systematic study on SOCs processes, people, and technology; providing researchers with an understanding of the challenges and opportunities within; bridging that knowledge gap and thereby set a better foundation for future research in the field.

References

[1] Bushra A AlAhmadi and Ivan Martinovic. "MalClassifier: Malware family classification using network flow sequence behaviour". In: *Proc. of IEEE eCrime*. 2018.

[2] M Antonakakis et al. "The Command Structure of the Operation Aurora Botnet: History, Patterns, and Findings". In: *Atlanta, GA: Damballa, Inc* (2010).

[3] J Nazario. "Political DDoS: Estonia and beyond". In: *Proc. of USENIX security*. 2008.

[4] Alice Decker et al. "Pushdo/cutwail botnet: A study of the pushdo/cutwail botnet". In: *TrendMicro Labs* (2009).

[5] Neil Daswani and Michael Stoppelman. "The anatomy of Clickbot. A". In: *USENIX HotBots*. 2007.

[6] Hamad Binsalleeh et al. "On the analysis of the Zeus botnet crimeware toolkit". In: *IEEE PST*. 2010.

[7] David Botta et al. "Towards understanding IT security professionals and their tools". In: *Proceedings of the 3rd symposium on Usable privacy and security*. ACM. 2007, pp. 100–111.

[8] Eser Kandogan and Eben M Haber. "Security administration tools and practices". In: *Security and Usability: Designing Secure Systems that People Can Use* (2005), pp. 357–378.

[9] John R Goodall, Wayne G Lutters, and Anita Komlodi. "I know my network: collaboration and expertise in intrusion detection". In: *Proceedings of the 2004 ACM conference on Computer supported cooperative work*. ACM. 2004, pp. 342–345.

[10] Sathya Chandran Sundaramurthy et al. "Humans are dynamic-our tools should be too". In: *IEEE Internet Computing* 21.3 (2017), pp. 40–46.

[11] Sathya Chandran et al. "A Human Capital Model for Mitigating Security Analyst Burnout". In: *SOUPS 2015* (2015), p. 347.

[12] Faris Bugra Kokulu et al. "Matched and Mismatched SOCs: A Qualitative Study on Security Operations Center Issues". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2019, pp. 1955–1970.

[13] Michael Riley et al. "Missed Alarms and 40 Million Stolen Credit Card Numbers: How Target Blew It." In: *Bloomberg.com* (2014), p. 1. URL: `http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=94957499&site=ehost-live&authtype=ip,uid`.

[14] Constanze Dietrich et al. "Investigating System Operators' Perspective on Security Misconfigurations". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2018, pp. 1272–1289.

[15] Holger Schulze. *Threat Hunting Report*. CyberSecurity Insiders, 2018.

[16] Ponemon-Institute. *2017 Cost of Data Breach Study*. 2017.

[17] M Zubair Rafique et al. "Evolutionary algorithms for classification of malware families through different network behaviors". In: *Proceedings of the 2014 conference on Genetic and evolutionary computation*. ACM. 2014, pp. 1167–1174.

[18] John Mccane and Chaz Yonez. *Fileless attacks against enterprise networks*. URL: `https://securelist.com/fileless-attacks-against-enterprise-networks/77403/`.

[19] Mamoun Alazab et al. "Cybercrime: the case of obfuscated malware". In: *Global Security, Safety and Sustainability & e-Democracy* (2012), pp. 204–211.

[20] Klaus Julisch. "Clustering intrusion detection alarms to support root cause analysis". In: *ACM transactions on information and system security (TISSEC)* 6.4 (2003), pp. 443–471.

[21] Shelly Xiaonan Wu and Wolfgang Banzhaf. "The use of computational intelligence in intrusion detection systems: A review". In: *Applied Soft Computing* 10.1 (2010), pp. 1–35.

[22] Sathya Chandran et al. "A Tale of Three Security Operation Centers". In: (2014).

[23] Carson Zimmerman. "Ten strategies of a world-class cybersecurity operations center". In: *MITRE corporate communications and public affairs. Appendices* (2014).

[24] Jon Oltsik. *SOC-as-a-service for Midmarket and Small Enterprise Organizations*. Tech. rep. The Enterprise Strategy Group, Mar. 2015.

[25] *Next-Generation Firewalls: NGFW: FortiGate*. URL: `https://www.fortinet.com/products/next-generation-firewall.html`.

[26] Symantec. *Internet Security Threat Report*. 2015. URL: `http://www.symantec.com/security%5C_response/publications/threatreport.jsp`.

[27] P. Jacobs, A. Arnab, and B. Irwin. "Classification of Security Operation Centers". In: *2013 Information Security for South Africa*. Aug. 2013, pp. 1–7.

[28] LM Axon et al. "Sonification in security operations centres: what do security practitioners think?" In: Internet Society, 2018.

[29] CW Dukes. *Committee on national security systems (CNSS) glossary*. Tech. rep. Technical report CNSSI, 2015.

[30] *Building A SOC with Splunk*. URL: `https://www.splunk.com/en_us/cyber-security/security-operations-automation/building-a-soc-with-splunk.html`.

[31] Miroslav Maj, Roeland Reijers, and Don Stikvoort. "Good practice guide for incident management". In: *European network and information security agency (ENISA)* (2010).

[32] Paul Cichonski et al. “Computer security incident handling guide”. In: *NIST Special Publication* 800.61 (2012), pp. 1–147.

[33] Jeff Bollinger, Brandon Enright, and Matthew Valites. *Crafting the InfoSec Playbook: Security Monitoring and Incident Response Master Plan.* " O’Reilly Media, Inc.", 2015.

[34] Sheharbano Khattak et al. “A taxonomy of botnet behavior, detection, and defense”. In: *IEEE COMST* 16.2 (2014), pp. 898–924.

[35] Sandeep Yadav et al. “Detecting Algorithmically Generated Malicious Domain Names”. In: *ACM IMC*. 2010.

[36] Gernot Vormayr, Tanja Zseby, and Joachim Fabini. “Botnet Communication Patterns”. In: *IEEE COMST* 19.4 (2017), pp. 2768–2796.

[37] GD Kurundkar, NA Naik, and SD Khamitkar. “Network intrusion detection using Snort”. In: *International Journal of Engineering Research and Applications* 2.2 (2012), pp. 1288–1296.

[38] Suman Rani and Vikram Singh. “SNORT: an open source network security tool for intrusion detection in campus network environment”. In: *International Journal of Computer Technology and Electronics Engineering* 2.1 (2012), pp. 137–142.

[39] Farzaneh Izak Shiri, Bharanidharan Shanmugam, and Norbik Bashah Idris. “A parallel technique for improving the performance of signature-based network intrusion detection system”. In: *2011 IEEE 3rd International Conference on Communication Software and Networks*. IEEE. 2011, pp. 692–696.

[40] Yong Tang and Shigang Chen. “Defending against internet worms: A signature-based approach”. In: *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.* Vol. 2. IEEE. 2005, pp. 1384–1394.

[41] Nattawat Khamphakdee, Nunnapus Benjamas, and Saiyan Saiyod. “Improving intrusion detection system based on snort rules for network probe attack detection”. In: *2014 2nd International Conference on Information and Communication Technology (ICoICT)*. IEEE. 2014, pp. 69–74.

[42] Eugene Albin and Neil C Rowe. “A realistic experimental comparison of the Suricata and Snort intrusion-detection systems”. In: *2012 26th International Conference on Advanced Information Networking and Applications Workshops*. IEEE. 2012, pp. 122–127.

[43] Prasanta Gogoi, Bhogeswar Borah, and Dhruba K Bhattacharyya. “Anomaly detection analysis of intrusion data using supervised & unsupervised approach”. In: *Journal of Convergence Information Technology* 5.1 (2010), pp. 95–110.

[44] Dhruba Kumar Bhattacharyya and Jugal Kumar Kalita. *Network anomaly detection: A machine learning perspective*. Chapman and Hall/CRC, 2013.

[45] Levent Koc, Thomas A Mazzuchi, and Shahram Sarkani. “A network intrusion detection system based on a Hidden Naıve Bayes multiclass classifier”. In: *Expert Systems with Applications* 39.18 (2012), pp. 13492–13500.

[46] Yan Qiao et al. “Anomaly intrusion detection method based on HMM”. In: *Electronics letters* 38.13 (2002), pp. 663–664.

[47] VVRPV Jyothsna, VV Rama Prasad, and K Munivara Prasad. "A review of anomaly based intrusion detection systems". In: *International Journal of Computer Applications* 28.7 (2011), pp. 26–35.

[48] Jiong Zhang and Mohammad Zulkernine. "Anomaly based network intrusion detection with unsupervised outlier detection". In: *2006 IEEE International Conference on Communications*. Vol. 5. IEEE. 2006, pp. 2388–2393.

[49] Zheng Zhang et al. "HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification". In: *Proc. IEEE Workshop on Information Assurance and Security*. 2001, pp. 85–90.

[50] Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. "Survey of network-based defense mechanisms countering the DoS and DDoS problems". In: *ACM Computing Surveys (CSUR)* 39.1 (2007), p. 3.

[51] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. "Network anomaly detection: methods, systems and tools". In: *Ieee communications surveys & tutorials* 16.1 (2013), pp. 303–336.

[52] Constantine Manikopoulos and Symeon Papavassiliou. "Network intrusion and fault detection: a statistical anomaly approach". In: *IEEE Communications Magazine* 40.10 (2002), pp. 76–82.

[53] P. Garcia-Teodoro et al. "Anomaly-based network intrusion detection: Techniques, systems and challenges". In: *Computers & Security* 28.1 (2009), pp. 18–28. URL: `http://www.sciencedirect.com/science/article/pii/S0167404808000692`.

[54] Jeremy Z Kolter and Marcus A Maloof. "Learning to detect malicious executables in the wild". In: *Proc. of ACM SIGKDD*. 2004.

[55] Younghee Park et al. "Fast Malware Classification by Automated Behavioral Graph Matching". In: *Proc. of ACM CSIIRW*. 2010.

[56] Konrad Rieck et al. "DIMVA". In: 2008. Chap. Learning and Classification of Malware Behavior.

[57] Konrad Rieck et al. "Automatic analysis of malware behavior using machine learning". In: *Journal of Computer Security* 19.4 (2011), pp. 639–668.

[58] Christopher Kruegel and Thomas Toth. "Using decision trees to improve signature-based intrusion detection". In: *Recent Advances in Intrusion Detection*. Springer. 2003, pp. 173–191.

[59] Gary Stein et al. "Decision tree classifier for network intrusion detection with GA-based feature selection". In: *Proceedings of the 43rd annual Southeast regional conference-Volume 2*. ACM. 2005, pp. 136–141.

[60] Mrutyunjaya Panda and Manas Ranjan Patra. "Network intrusion detection using naive bayes". In: *International journal of computer science and network security* 7.12 (2007), pp. 258–263.

[61] Nahla Ben Amor, Salem Benferhat, and Zied Elouedi. "Naive bayes vs decision trees in intrusion detection systems". In: *Proceedings of the 2004 ACM symposium on Applied computing*. ACM. 2004, pp. 420–424.

[62] Dewan Md Farid, Nouria Harbi, and Mohammad Zahidur Rahman. "Combining naive bayes and decision tree for adaptive intrusion detection". In: *arXiv preprint arXiv:1005.4496* (2010).

[63] Martin Roesch et al. “Snort: Lightweight Intrusion Detection for Networks.” In: *LISA*. Vol. 99. 1. 1999, pp. 229–238.

[64] Koral Ilgun, Richard Kemmerer, Phillip Porras, et al. “State transition analysis: A rule-based intrusion detection approach”. In: *Software Engineering, IEEE Transactions on* 21.3 (1995), pp. 181–199.

[65] Jake Ryan, Meng-Jang Lin, and Risto Miikkulainen. “Intrusion detection with neural networks”. In: *Advances in neural information processing systems* (1998), pp. 943–949.

[66] James Cannady. “Artificial neural networks for misuse detection”. In: *National information systems security conference*. 1998, pp. 368–81.

[67] Majid Ghonji Feshki, Omid Sojoodi, and Minoo Deljavan Anvary. “Managing Intrusion Detection Alerts Using Support Vector Machines”. In: *International Journal of Computer Science and Security (IJCSS)* 9.5 (2015), p. 266.

[68] Shi-Jinn Horng et al. “A novel intrusion detection system based on hierarchical clustering and support vector machines”. In: *Expert systems with Applications* 38.1 (2011), pp. 306–313.

[69] Mohammad Sazzadul Hoque et al. “An implementation of intrusion detection system using genetic algorithm”. In: *arXiv preprint arXiv:1204.1336* (2012).

[70] Arman Tajbakhsh, Mohammad Rahmati, and Abdolreza Mirzaei. “Intrusion detection using fuzzy association rules”. In: *Applied Soft Computing* 9.2 (2009), pp. 462–469.

[71] Wenke Lee and Salvatore J Stolfo. “Data mining approaches for intrusion detection”. In: *Usenix Security*. 1998.

[72] M Zubair Shafiq, S Momina Tabish, and Muddassar Farooq. “Are evolutionary rule learning algorithms appropriate for malware detection?” In: *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*. ACM. 2009, pp. 1915–1916.

[73] Ulrich Bayer et al. “A view on current malware behaviors”. In: *USENIX workshop on large-scale exploits and emergent threats (LEET)*. 2009.

[74] Carsten Willems, Thorsten Holz, and Felix Freiling. “Toward automated dynamic malware analysis using cwsandbox”. In: *IEEE Security & Privacy* 2 (2007), pp. 32–39.

[75] L Xue and G Sun. “Design and implementation of a malware detection system based on network behavior”. In: *Security and Communication Networks* 8.3 (2015), pp. 459–470.

[76] Christian Rossow et al. “Sandnet: Network traffic analysis of malicious software”. In: *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*. ACM. 2011, pp. 78–88.

[77] Konrad Rieck et al. “Botzilla: Detecting the phoning home of malicious software”. In: *proceedings of the 2010 ACM Symposium on Applied Computing*. ACM. 2010, pp. 1978–1984.

[78] Guofei Gu, Junjie Zhang, and Wenke Lee. “BotSniffer: Detecting botnet command and control channels in network traffic”. In: (2008).

[79] Leyla Bilge et al. “Disclosure: detecting botnet command and control servers through large-scale netflow analysis”. In: *Proc. of ACM ACSAC*. 2012.

[80] Christian Rossow and Christian J Dietrich. “Provex: Detecting botnets with encrypted command and control channels”. In: *DIMVA*. 2013, pp. 21–40.

[81] Guofei Gu et al. “BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection.” In: *USENIX Security Symposium*. Vol. 5. 2. 2008, pp. 139–154.

[82] Guofei Gu et al. “BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation.” In: *Usenix Security*. Vol. 7. 2007, pp. 1–16.

[83] Ayesha Binte Ashfaq et al. “Diagnosing bot infections using Bayesian inference”. In: *Journal of Computer Virology and Hacking Techniques* (2016).

[84] Florian Tegeler et al. “BotFinder: Finding Bots in Network Traffic Without Deep Packet Inspection”. In: *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. CoNEXT ’12. Nice, France: ACM, 2012, pp. 349–360. URL: http://doi.acm.org/10.1145/2413176.2413217.

[85] Zainab Abaid, Mohamed Ali Kaafar, and Sanjay Jha. “Early Detection of In-the-Wild Botnet Attacks by Exploiting Network Communication Uniformity: An Empirical Study”. In: *IFIP Networking* (2017).

[86] Zhaoyan Xu et al. “PeerPress: utilizing enemies’ P2P strength against them”. In: *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM. 2012, pp. 581–592.

[87] Antonio Nappa et al. “Cyberprobe: Towards internet-scale active detection of malicious servers”. In: 2014.

[88] Zhaoyan Xu et al. “AUTOPROBE: Towards Automatic Active Malicious Server Probing Using Dynamic Binary Analysis”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. CCS ’14. Scottsdale, Arizona, USA: ACM, 2014, pp. 179–190. URL: http://doi.acm.org/10.1145/2660267.2660352.

[89] Guofei Gu et al. “Active botnet probing to identify obscure command and control channels”. In: *Computer Security Applications Conference, 2009. ACSAC’09. Annual*. IEEE. 2009, pp. 241–253.

[90] Leyla Bilge et al. “EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis.” In: *NDSS*. 2011.

[91] Manos Antonakakis et al. “Building a Dynamic Reputation System for DNS.” In: *USENIX security symposium*. 2010, pp. 273–290.

[92] Lei Zhang et al. “A survey on latest botnet attack and defense”. In: *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*. IEEE. 2011, pp. 53–60.

[93] Xin Hu, Matthew Knysz, and Kang G Shin. “RB-Seeker: Auto-detection of Redirection Botnets.” In: *NDSS*. 2009.

[94] Roberto Perdisci et al. “Detecting malicious flux service networks through passive analysis of recursive dns traces”. In: *Computer Security Applications Conference, 2009. ACSAC’09. Annual*. IEEE. 2009, pp. 311–320.

[95] Etienne Stalmans and Barry Irwin. "A framework for DNS based detection and mitigation of malware infections on a network". In: *Information Security South Africa (ISSA), 2011*. IEEE. 2011, pp. 1–8.

[96] Thorsten Holz et al. "Measuring and Detecting Fast-Flux Service Networks." In: *NDSS*. 2008.

[97] Emanuele Passerini et al. "Fluxor: Detecting and monitoring fast-flux service networks". In: *Detection of intrusions and malware, and vulnerability assessment*. Springer, 2008, pp. 186–206.

[98] Jose Nazario and Thorsten Holz. "As the net churns: Fast-flux botnet observations". In: *Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on*. IEEE. 2008, pp. 24–31.

[99] Pratyusa K Manadhata et al. "Detecting malicious domains via graph inference". In: *Computer Security-ESORICS 2014*. Springer, 2014, pp. 1–18.

[100] Aziz Mohaisen et al. "Chatter: Classifying malware families using system event ordering". In: *Proc. of IEEE CNS*. 2014.

[101] Hesham Mekky, Aziz Mohaisen, and Zhi-Li Zhang. "Separation of benign and malicious network events for accurate malware family classification". In: *IEEE CNS*. 2015.

[102] Christian Wressnegger et al. "A close look on n-grams in intrusion detection: anomaly detection vs. classification". In: *ACM AISec*. 2013.

[103] Roberto Perdisci, Wenke Lee, and Nick Feamster. "Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces." In: *Proc. of USENIX NSDI*. 2010.

[104] M Zubair Rafique and Juan Caballero. "Firma: Malware clustering and network signature generation with mixed network behaviors". In: *Proc. of RAID*. 2013.

[105] Igor Santos et al. "N-grams-based File Signatures for Malware Detection." In: 2009.

[106] Emmanouil Vasilomanolakis et al. "Taxonomy and Survey of Collaborative Intrusion Detection". In: *ACM Computing Surveys (CSUR)* 47.4 (2015), p. 55.

[107] Tim Bass. "Intrusion detection systems and multisensor data fusion". In: *Communications of the ACM* 43.4 (2000), pp. 99–105.

[108] Fang Lan, Wang Chunlei, and Ma Guoqing. "A framework for network security situation awareness based on knowledge discovery". In: *Computer Engineering and Technology (ICCET), 2010 2nd international conference on*. Vol. 1. IEEE. 2010, pp. V1–226.

[109] Richard Zuech, Taghi M Khoshgoftaar, and Randall Wald. "Intrusion detection and Big Heterogeneous Data: a Survey". In: *Journal of Big Data* 2.1 (2015), pp. 1–41.

[110] BA Fessi et al. "Data collection for information security system". In: *Engineering Systems Management and Its Applications (ICESMA), 2010 second international conference on*. IEEE. 2010, pp. 1–8.

[111] Abdoul Karim Ganame et al. “A global security architecture for intrusion detection on computer networks”. In: *Computers and Security* 27.1-2 (2008), pp. 30–47.

[112] David L Hall and James Llinas. “An introduction to multisensor data fusion”. In: *Proceedings of the IEEE* 85.1 (1997), pp. 6–23.

[113] Ting-Fang Yen et al. “Beehive: Largescale log analysis for detecting suspicious activity in enterprise networks”. In: *Proceedings of the 29th Annual Computer Security Applications Conference*. ACM. 2013, pp. 199–208.

[114] E. Bocchi et al. “Network Connectivity Graph for Malicious Traffic Dissection”. In: *Computer Communication and Networks (ICCCN), 2015 24th International Conference on*. Aug. 2015, pp. 1–9.

[115] Alina Oprea et al. “Detection of early-stage enterprise infection by mining large-scale log data”. In: *arXiv preprint arXiv:1411.5005* (2014).

[116] Ting-Fang Yen and Michael K Reiter. “Traffic aggregation for malware detection”. In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2008, pp. 207–227.

[117] C. Zhong et al. “Automate Cybersecurity Data Triage by Leveraging Human Analysts’ Cognitive Process”. In: *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*. Apr. 2016, pp. 357–363.

[118] Chen Zhong et al. “A cyber security data triage operation retrieval system”. In: *Computers & Security* 76 (2018), pp. 12–31.

[119] Elias Raftopoulos, Matthias Egli, and Xenofontas Dimitropoulos. “Shedding light on log correlation in network forensics analysis”. In: *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer. 2012, pp. 232–241.

[120] Antonio Pecchia et al. “Filtering security alerts for the analysis of a production saas cloud”. In: *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*. IEEE. 2014, pp. 233–241.

[121] Xiaokui Shu et al. “Threat Intelligence Computing”. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. CCS ’18. Toronto, Canada: ACM, 2018, pp. 1883–1898. URL: http://doi.acm.org/10.1145/3243734.3243829.

[122] Rodrigo Werlinger et al. “Preparation, detection, and analysis: the diagnostic work of IT security incident response”. In: *Information Management & Computer Security* 18.1 (2010), pp. 26–42.

[123] Rodrigo Werlinger, Kirstie Hawkey, and Konstantin Beznosov. “An integrated view of human, organizational, and technological challenges of IT security management”. In: *Information Management & Computer Security* 17.1 (2009), pp. 4–19.

[124] Rodrigo Werlinger et al. “Security practitioners in context: Their activities and interactions with other stakeholders within organizations”. In: *International Journal of Human-Computer Studies* 67.7 (2009), pp. 584–606.

[125] Rodrigo Werlinger et al. “The challenges of using an intrusion detection system: is it worth the effort?” In: *Proceedings of the 4th symposium on Usable privacy and security*. ACM. 2008, pp. 107–118.

[126] David Botta et al. “Toward understanding distributed cognition in IT security management: the role of cues and norms”. In: *Cognition, Technology & Work* 13.2 (2011), pp. 121–134.

[127] Kirstie Hawkey et al. “Human, organizational, and technological factors of IT security”. In: *CHI’08 extended abstracts on Human factors in computing systems*. ACM. 2008, pp. 3639–3644.

[128] Sathya Chandran Sundaramurthy et al. “Turning contradictions into innovations or: How we learned to stop whining and improve security operations”. In: *Proc. 12th Symp. Usable Privacy and Security*. 2016.

[129] Andrew M’manga et al. “Folk risk analysis: Factors influencing security analysts’ interpretation of risk”. In: *Proc. of the 13th Symposium on Usable Privacy and Security, ser. SOUPS*. Vol. 17. 2017.

[130] Anita D’Amico et al. “Achieving Cyber Defense Situational Awareness: A Cognitive Task Analysis of Information Assurance Analysts”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 49.3 (2005), pp. 229–233.

[131] Anita D’Amico and Kirsten Whitley. “The real work of computer network defense analysts”. In: *VizSEC 2007*. Springer, 2008, pp. 19–37.

[132] Dorothy E Denning. “An intrusion-detection model”. In: *Software Engineering, IEEE Transactions on* 2 (1987), pp. 222–232.

[133] Bazara IA Barry and H Anthony Chan. “Intrusion detection systems”. In: *Handbook of Information and Communication Security*. Springer, 2010, pp. 193–205.

[134] Zhi-cai SHI and Yong-xiang XIA. “Survey on intrusion detection techniques for high-speed networks”. In: *Application Research of Computers* 5 (2010), p. 004.

[135] Chenfeng Vincent Zhou, Christopher Leckie, and Shanika Karunasekera. “A survey of coordinated attacks and collaborative intrusion detection”. In: *Computers & Security* 29.1 (2010), pp. 124–140.

[136] Ahmed Patel, Qais Qassim, and Christopher Wills. “A survey of intrusion detection and prevention systems”. In: *Information Management & Computer Security* 18.4 (2010), pp. 277–290.

[137] Niva Das and Tanmoy Sarkar. “Survey on Host and Network Based Intrusion Detection System”. In: *Int. J. Advanced Networking and Applications* 6.2 (2014), pp. 2266–2269.

[138] Pedro Garcia-Teodoro et al. “Anomaly-based network intrusion detection: Techniques, systems and challenges”. In: *computers & security* 28.1 (2009), pp. 18–28.

[139] Animesh Patcha and Jung-Min Park. “An overview of anomaly detection techniques: Existing solutions and latest technological trends”. In: *Computer Networks: The International Journal of Computer and Telecommunications Networking* 51.12 (2007), pp. 3448–3470.

[140] Sebastián Garcıa, Alejandro Zunino, and Marcelo Campo. "Survey on network-based botnet detection methods". In: *Security and Communication Networks* 7.5 (2014), pp. 878–903.

[141] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* 41.3 (2009), p. 15.

[142] Chih-Fong Tsai et al. "Intrusion detection by machine learning: A review". In: *expert systems with applications* 36.10 (2009), pp. 11994–12000.

[143] Chakchai So-In et al. "An evaluation of data mining classification models for network intrusion detection". In: *Digital Information and Communication Technology and it's Applications (DICTAP), 2014 Fourth International Conference on*. IEEE. 2014, pp. 90–94.

[144] Roshani Gaidhane, C Vaidya, and M Raghuwanshi. "Survey: Learning Techniques for Intrusion Detection System (IDS)". In: (2014).

[145] Robin Sommer and Vern Paxson. "Outside the closed world: On using machine learning for network intrusion detection". In: *2010 IEEE symposium on security and privacy*. IEEE. 2010, pp. 305–316.

[146] KDD Cup. "Dataset". In: *available at the following website http://kdd. ics. uci. edu/databases/kddcup99/kddcup99. html* (1999).

[147] MIT Lincoln Laboratory. *DARPA Intrusion Detection Dataset*. 1999. URL: `http://www.ll.mit.edu/ideval/data/`.

[148] Chandrashekhar Azad and Vijay Kumar Jha. "Data mining in intrusion detection: a comparative study of methods, types and data sets". In: *International Journal of Information Technology and Computer Science (IJITCS)* 5.8 (2013), p. 75.

[149] John McHugh. "Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory". In: *ACM transactions on Information and system Security* 3.4 (2000), pp. 262–294.

[150] Ciza Thomas, Vishwas Sharma, and N Balakrishnan. "Usefulness of DARPA dataset for intrusion detection system evaluation". In: *SPIE Defense and Security Symposium*. International Society for Optics and Photonics. 2008, 69730G–69730G.

[151] Scott E Coull et al. "Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces." In: *NDSS*. Vol. 7. 2007, pp. 35–47.

[152] Hossein Rouhani Zeidanloo et al. "A taxonomy of botnet detection techniques". In: *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*. Vol. 2. IEEE. 2010, pp. 158–162.

[153] Moheeb Abu Rajab et al. "A Multifaceted Approach to Understanding the Botnet Phenomenon". In: *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*. IMC '06. Rio de Janeriro, Brazil: ACM, 2006, pp. 41–52. URL: `http://doi.acm.org/10.1145/1177080.1177086`.

[154] Grégoire Jacob, Hervé Debar, and Eric Filiol. "Behavioral detection of malware: from a survey towards an established taxonomy". In: *Journal in computer Virology* 4.3 (2008), pp. 251–266.

[155] Joanna Rutkowska. “Introducing stealth malware taxonomy”. In: *COSEINC Advanced Malware Labs* (2006), pp. 1–9.

[156] Antonio Nappa, M Zubair Rafique, and Juan Caballero. “The MALICIA dataset: identification and analysis of drive-by download operations”. In: *International Journal of Information Security* 14.1 (2015), pp. 15–33.

[157] Manuel Egele et al. “A survey on automated dynamic malware-analysis techniques and tools”. In: *ACM Computing Surveys (CSUR)* 44.2 (2012), p. 6.

[158] Ekta Gandotra, Divya Bansal, and Sanjeev Sofat. “Malware analysis and classification: A survey”. In: *Journal of Information Security* 2014 (2014).

[159] Nwokedi Idika and Aditya P Mathur. “A survey of malware detection techniques”. In: *Purdue University* 48 (2007).

[160] P Vinod et al. “Survey on malware detection methods”. In: *Proceedings of the 3rd Hackers Workshop on Computer and Internet Security (IITKHACK09)*. 2009, pp. 74–79.

[161] Guillermo Suarez-Tangil et al. “Dendroid: A text mining approach to analyzing and classifying code structures in android malware families”. In: *Expert Systems with Applications* 41.4 (2014), pp. 1104–1117.

[162] Silvio Cesare and Yang Xiang. “Classification of Malware Using Structured Control Flow”. In: *Proc. of AusPDC*. 2010.

[163] Ulrich Bayer et al. “Scalable, Behavior-Based Malware Clustering.” In: *Proc. of NDSS*. 2009.

[164] Sadegh M Milajerdi et al. “Holmes: real-time apt detection through correlation of suspicious information flows”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2019, pp. 1137–1152.

[165] Kexin Pei et al. “Hercule: Attack story reconstruction via community discovery on correlated log graph”. In: *Proceedings of the 32Nd Annual Conference on Computer Security Applications*. ACM. 2016, pp. 583–595.

[166] Chengwei Peng et al. “Discovering malicious domains through alias-canonical graph”. In: *2017 IEEE Trustcom/BigDataSE/ICESS*. IEEE. 2017, pp. 225–232.

[167] Sadegh M Milajerdi et al. “ProPatrol: Attack Investigation via Extracted High-Level Tasks”. In: *International Conference on Information Systems Security*. Springer. 2018, pp. 107–126.

[168] Fucheng Liu et al. “Log2vec: A Heterogeneous Graph Embedding Based Approach for Detecting Cyber Threats within Enterprise”. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2019, pp. 1777–1794.

[169] *Cyberbit: Security Orchestration Automation and Response (SOAR)*. `https://www.cyberbit.com/solutions/security-operations-automation-orchestration/`.

[170] *Demisto: Automated Incident Response and Security Orchestration*. `https://www.demisto.com`.

[171] Pejman Najafi et al. “MalRank: a measure of maliciousness in SIEM-based knowledge graphs”. In: *Proceedings of the 35th Annual Computer Security Applications Conference*. ACM. 2019, pp. 417–429.

[172] Duen Horng “Polo” Chau et al. “Polonium: Tera-scale graph mining and inference for malware detection”. In: *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM. 2011, pp. 131–142.

[173] Pooya Jaferian et al. “Guidelines for designing IT security management tools”. In: *Proceedings of the 2nd ACM Symposium on Computer Human interaction For Management of information Technology*. ACM. 2008, p. 7.

[174] Olusola Akinrolabu, Ioannis Agrafiotis, and Arnau Erola. “The challenge of detecting sophisticated attacks: Insights from SOC Analysts”. In: *Proceedings of the 13th International Conference on Availability, Reliability and Security*. ACM. 2018, p. 55.

[175] Celeste Lyn Paul and Kirsten Whitley. “A taxonomy of cyber awareness questions for the user-centered design of cyber situation awareness”. In: *International Conference on Human Aspects of Information Security, Privacy, and Trust*. Springer. 2013, pp. 145–154.

[176] Julie M Haney and Celeste Paul. “Toward Integrated Tactical Operations for Red/Blue Cyber Defense Teams”. In: Aug. 2018.

[177] John Goodall, Wayne Lutters, and Anita Komlodi. “The work of intrusion detection: rethinking the role of security analysts”. In: *AMCIS 2004 Proceedings* (2004), p. 179.

[178] Judy Robertson. “Likert-type scales, statistical methods, and effect sizes.” In: *Communications of the ACM* 55.5 (2012), pp. 6–7.

[179] Baruch Nevo. “Face validity revisited”. In: *Journal of Educational Measurement* 22.4 (1985), pp. 287–293.

[180] Nigel King. “Doing template analysis”. In: *Qualitative organizational research: Core methods and current challenges* 426 (2012).

[181] Gery W Ryan and H Russell Bernard. “Techniques to identify themes”. In: *Field methods* 15.1 (2003), pp. 85–109.

[182] Elaheh Biglar Beigi et al. “Towards effective feature selection in machine learning-based botnet detection approaches”. In: *Proc. of IEEE CNS*. 2014.

[183] Sebastian Garcia et al. “An empirical comparison of botnet detection methods”. In: *Computers & Security* 45 (2014), pp. 100–123.

[184] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.

[185] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[186] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[187] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. “Distance metric learning for large margin nearest neighbor classification”. In: *Advances in neural information processing systems*. 2006, pp. 1473–1480.

[188] ISO Guide. "73: 2009". In: *Risk management—Vocabulary* 551 (2009), p. 49.

[189] Anton Chuvakin and Augusto Barros. *How to Develop and Maintain Security Monitoring Use Cases*. URL: https://www.gartner.com/en/documents/3844970.

[190] Calvin Nobles. "The Cyber Talent Gap and Cybersecurity Professionalizing". In: *International Journal of Hyperconnectivity and the Internet of Things* 2.1 (2018), pp. 42–51.

[191] Rob Van Os. "SOC-CMM: Designing and Evaluating a Tool for Measurement of Capability Maturity in Security Operations Centers". MA thesis. Luleå University of Technology, Computer Science, 2016, p. 74.

[192] Christopher Kruegel and William Robertson. "Alert verification determining the success of intrusion attempts". In: *DIMVA 2004, July 6-7, Dortmund, Germany* (2004).

[193] Fredrik Valeur et al. "Comprehensive approach to intrusion detection alert correlation". In: *IEEE Transactions on dependable and secure computing* 1.3 (2004), pp. 146–169.

[194] André Årnes et al. "Using hidden markov models to evaluate the risks of intrusions". In: *International Workshop on Recent Advances in Intrusion Detection*. Springer. 2006, pp. 145–164.

[195] Sandeep Bhatt, Pratyusa K Manadhata, and Loai Zomlot. "The operational role of security information and event management systems". In: *IEEE security & Privacy* 12.5 (2014), pp. 35–41.

[196] Chadni Islam, Muhammad Ali Babar, and Surya Nepal. "A Multi-Vocal Review of Security Orchestration". In: *ACM Computing Surveys (CSUR)* 52.2 (2019), p. 37.

[197] Xiaokui Shu et al. "Threat intelligence computing". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2018, pp. 1883–1898.

[198] Fraser Howard. *A closer look at the Angler exploit kit*. July 2015.

[199] Z Berkay Celik et al. "Malware traffic detection using tamper resistant features". In: *IEEE MILCOM*. 2015.

[200] Marie-Jeanne Lesot, Maria Rifqi, and Hamid Benhadda. "Similarity Measures for Binary and Numerical Data: a Survey". In: *Int. J. Knowl. Eng. Soft Data Paradigm.* 1.1 (Dec. 2009), pp. 63–84.

[201] Vladimir I Levenshtein. "Binary codes capable of correcting deletions, insertions and reversals". In: *Soviet physics doklady*. Vol. 10. 1966, p. 707.

[202] D Krishna Sandeep Reddy and Arun K Pujari. "N-gram analysis for computer virus detection". In: *Journal in Computer Virology* 2.3 (2006), pp. 231–239.

[203] Surendra K Singhi and Huan Liu. "Feature subset selection bias for classification learning". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 849–856.

[204] Svante Wold, Kim Esbensen, and Paul Geladi. "Principal component analysis". In: *Chemometrics and intelligent laboratory systems* 2.1-3 (1987), pp. 37–52.

[205] Roberto Perdisci et al. “Misleading worm signature generators using deliberate noise injection”. In: *2006 IEEE Symposium on Security and Privacy (S&P’06)*. IEEE. 2006, 15–pp.

[206] Elizabeth Stinson and John C Mitchell. “Towards Systematic Evaluation of the Evadability of Bot/Botnet Detection Methods.” In: *Proc. of USENIX WOOT* (2008).

[207] Enrico Mariconti et al. “MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models”. In: *Proc. of NDSS*. 2017.

[208] Lucky Onwuzurike et al. “MaMaDroid: Detecting android malware by building markov chains of behavioral models (extended version)”. In: *ACM TOPS* 22.2 (2019), p. 14.

[209] Elias Raftopoulos and Xenofontas Dimitropoulos. “Detecting, validating and characterizing computer infections in the wild”. In: *ACM IMC*. 2011.

[210] Paul Dokas et al. “Data mining for network intrusion detection”. In: *Proc. of NSF NGDM*. 2002.

[211] Michal Piskozub, Riccardo Spolaor, and Ivan Martinovic. “MalAlert: Detecting Malware in Large-Scale Network Traffic Using Statistical Features”. In: *ACM SIGMETRICS Performance Evaluation Review* 46.3 (2019), pp. 151–154.

[212] Z Berkay Celik et al. “Malware traffic detection using tamper resistant features”. In: *MILCOM 2015-2015 IEEE Military Communications Conference*. IEEE. 2015, pp. 330–335.

[213] Hamad Binsalleeh et al. “On the analysis of the Zeus botnet crimeware toolkit”. In: *Proc. of ACM PST*. 2010.

[214] Feargus Pendlebury et al. “TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time”. In: *Proc. of USENIX Security*. 2019.

[215] Moheeb Abu Rajab et al. “A Multifaceted Approach to Understanding the Botnet Phenomenon”. In: *Proc. of ACM IMC*. 2006.

[216] Zainab Abaid et al. “The Early Bird Gets the Botnet: A Markov Chain Based Early Warning System for Botnet Attacks”. In: *Proc. of IEEE LCN*. 2016.

[217] Sebastian Zander, Grenville Armitage, and Philip Branch. “A survey of covert channels and countermeasures in computer network protocols”. In: *IEEE COMST* 9.3 (2007), pp. 44–57.

[218] Daniel Lowd and Christopher Meek. “Adversarial learning”. In: *Proc. of ACM SIGKDD*. 2005.

[219] Vincent F Taylor et al. “Robust smartphone app identification via encrypted network traffic analysis”. In: *IEEE TIFS* 13.1 (2018), pp. 63–78.

[220] David Zhao et al. “Botnet detection based on traffic behavior analysis and flow intervals”. In: *Computers & Security* 39 (2013), pp. 2–16.

Appendices

A

Semi-Structured Interview Questions

A.1 Interview Part 1

The goal for this interview is to capture the analysts views on the current network security tools used in the SOC.

1. Please describe your position/job role/level.

2. Which network monitoring tools do you use in your monitoring work?

3. Looking back on past events, have there been times during your use of network-monitoring systems when they performed particularly well?

4. Can you describe an incident that was detected well?

5. Looking back on past events, have there been times during your use of network-monitoring systems when they could have performed better? Can you describe an incident that was not detected as well as it should have been?

6. What is your view on the strengths and weaknesses of the network-monitoring systems you use?

7. What is your view on the accuracy of the network-monitoring systems you use?

   (a) Are there events they do not detect that they should be detecting (false negatives)? If so, is this a frequent occurrence?

   (b) Do they detect false positive events? If so, what proportion of events detected are false positives?

   (c) Can you comment on the balance between false positives and false negatives in your systems? Which are there more of?

8. What is your view on the usability of the network-monitoring systems you use?

9. We are interested in the balance between attack detection by automated systems, and work performed manually by analysts. Can you describe this balance? (e.g. are there times when you as an analyst use your own experience to explore the data and make decisions, rather than or alongside using automated system alerts, and how do you do this?)

10. Do you feel that you as an analyst monitoring the network are capable of detecting attacks that might be missed by automated systems?

    (a) How does your existing monitoring setup enable this?

    (b) Can you give an example?

    (c) How do you maintain situational awareness of your network?

11. Do you feel the level of Situational Awareness you are able to achieve currently is sufficient?

    (a) Can you comment on the usability of the systems through which you maintain SA? How easy is it to stay "in the loop"?

    (b) Is it necessary for you to maintain SA while performing other tasks, and how do you do so?

12. How do you decide which events to prioritise?

13. Would you like to share any more views on the network-monitoring systems you use that have not been covered by the questions so far?

A.2 Interview Part 2

During the interview, the participant was presented with a problem scenario and was asked to talk through the steps they will use to address the problem. We presented the analysts' with a number of scenarios in a semi-structured format, criticizing and inspecting scenarios in walk-throughs. Hence, we asked: For the following network scenarios, could you describe step-by-step how you would investigate?

1. Processes, roles, workflow between SOC team members. Which are the first tasks you would carry out?

2. Which tools would you use?

3. Which data would you gather to help in your investigation? What are the first information sources you would check, and what indicators would you look for?

4. How would you prioritise and assess severity?

5. Which tasks would you automate, and which data sources would you explore manually?

A.2.1 Scenarios

1. You are monitoring the organisation's network traffic, and observe an increase in the traffic from a server to an outside entity.

2. You are monitoring the organisation's network traffic, and observe an increase in the traffic from a server to another device on the network.

3. You are monitoring the organisation's network traffic, and observe an increase in the network outbound traffic (how does the analyst find the malicious internal host?).
4. You receive an alert from the SIEM of unauthorised privilege escalation attempt.
5. Scanning the network, you find an IOC such as a malware MD5 hash.
6. Your web server is receiving SYN flood requests.

B

Online Questionnaire - Assertions Results

Table B.1: Online Survey Results: Responses to Assertions (Resp, Ordered from "Strongly Disagree"(=1) - "Strongly Agree"(=5)) Mode, Median, and Comparison of non-neutral scores - Disagree (1-2):Agree (4-5)(CNNS: D:A)

Assertion	Resp	Mode	Median	CCN
Data Fusion				
- I am required to simultaneously monitor information from multiple monitoring sources (eg. network traffic, IDS logs, firewall logs, etc)	0,3,2,9,6	4	4	3:15
- I am required to simultaneously monitor the state of multiple network elements (e.g. individual machines, database server, web server, etc.)	0,3,6,9,2	4	4	3:11
- Monitoring the status of the network involves interacting with an IDS and other monitoring tools in addition to following external information resources for vulnerabilities	0,0,6,10,4	4	4	0:14
- SIEMS require collecting and storing billions of logs every day, we have limited resources and are not able to keep these logs for long periods of time	2,6,4,6,2	4	3	8:8
- Aggregation of billions of log alerts a day presented in heterogeneous data structures makes analysis a challenge	0,3,5,8,4	4	4	3:12
- Devices may generate logs that are either incomplete or inconsistent (e.g. different time-stamps) making analysis an even bigger challenge	0,3,5,8,4	4	4	3:12
- The tools I use are effective at supporting data fusion and correlation across incidents and data sources	1,1,4,12,2	4	4	2:14
Network Monitoring Tools				
- The monitoring tools I use frequently produce false positive results (they detect a security event when there was not actually a security event)	0,2,4,11,3	4	4	2:14
- The monitoring tools I use frequently produce false negative results (they fail to detect a security event that occurs)	0,6,9,3,2	3	3	6:5
- I rely on my custom scripts to aggregate the data I need to analyse an incident	2,5,8,3,2	3	3	7:5
- To detect malicious activity on the network I deploy custom created scripts	1,5,7,5,2	3	3	6:7
Intrusion Detection Systems				
- I believe that current IDS are inadequate in detecting attacks	2,5,7,5,1	3	3	7:6
- Current IDS solutions are incapable of coping with the high data rates	3,6,8,2,1	3	3	9:3
- Current IDS solutions lack in effective and speedy threat detection and response	4,5,5,5,1	2	3	9:6
- Current IDS solutions are built on the assumption that threats are observed as they enter the network in specific perimeter points at the Internet edge	2,3,6,8,1	4	3	5:9
- The number of alerts generated by most IDS are overwhelming	0,4,4,8,4	4	4	4:12
- To reduce the number of false positives, I limit the IDS signature set to focus on important attacks	2,4,6,6,2	3	3	6:8
- I pick IDS alerts of interest based on problems we have been having lately	1,10,4,4,1	2	2	11:5

Table B.2: Online Survey Results: Responses to Assertions (Resp, Ordered from "Strongly Disagree"(=1) - "Strongly Agree"(=5)) Mode, Median, and Comparison of non-neutral scores - Disagree (1-2):Agree (4-5)(CNNS: D:A)

Assertion	Resp	Mode	Median	CCN
Human in the Loop				
A1: It is important to have a human in the loop for the detection and preliminary analysis of potential security events this process cannot be carried out by automated systems alone	0,2,0,7,11	5	5	2:18
A2: Human analysts monitoring the network are capable of detecting network anomalies that are missed by automated systems	0,1,5,10,4	4	4	1:14
A3: I am often required to make decisions on the accuracy of the alerts produced by automated systems	0,0,5,8,7	4	4	0:15
A4: I sometimes rely on my experience and intuition to detect attacks rather than monitoring system alerts	0,2,7,7,4	3	4	2:11
Knowing your Network				
A5: Maintaining awareness of the network security state is important in enabling me to make decisions on the accuracy of alerts produced by automated monitoring systems	0,0,3,13,4	4	4	0:17
A6: Keeping up with changing configurations in the network is difficult, but necessary to provide the context needed to analyse and diagnose an alert	0,1,4,9,6	4	4	1:15
A7: Keeping track and knowing the network environment is important to detect attacks	0,0,1,12,7	4	4	0:19
A8: I track network configurations using personal memory	1,4,8,4,3	3	3	5:7
A9: I track network configurations by keeping an updated database of network devices and configuration changes	0,4,5,8,3	4	4	4:11

C

Template Analysis: Template

We detail in the following the template derived using template Analysis methodology and used to code the semi-structured interviews.

1. **People**
   - (a) Recruitment
   - (b) Responsibilities an Positions
   - (c) Skills and Education
   - (d) Teams and Groups
   - (e) Challenges
     - i. Challenges in Recruitment
     - ii. Communication between SOC members and customer
     - iii. Division of Responsibilities
       - A. Pressure in making decisions
       - B. Who monitors what
     - iv. Multitasking and Overwhelming analysts
2. **Process**
   - (a) Workflow Stages
     - i. Preparation
       - A. Risk assessment and asset criticality
       - B. Configuring SIEM use-cases
       - C. Identifying datasources
       - D. Base-lining and Alarm tuning
       - E. Customer Profile documentation
       - F. open-source intelligence
       - G. tool maintenance and updates
     - ii. Detection
       - A. Detection through Tools
       - B. Detection through Hunting

C. Filtering False positives/benign Triggers/noise
D. Prioritization of triaged alarms

iii. Investigation
A. Filtering false positives
B. Knowledge and context needed (tacit, network, customer)
C. Usage of tools
D. Analytical questions and reasoning

iv. Response
A. Response decision based on type of threat (e.g. APT)
B. Communicating Risk to customer
C. Forensics and onsite investigations

(b) Knowledge and context needed (tacit, network, customer)

(c) Use of playbook and procedures

(d) Management of SOC

(e) Influencing factors

i. Government vs. non-government organisations
ii. Type of SOC (internal vs. MSSP SOC)
iii. Cost of technology and skills
iv. Organisation (size, business)
v. Change management and patching processes
vi. Risk appetite
vii. Communication channels
viii. SOC maturity

3. **Technology**

(a) Traditional Technologies

i. Traditional technologies types (signature/anomaly)
ii. Strengths
A. Good first indicator
B. Deal with high volume of traffic
C. well maintained
D. Easy to write signatures
E. Fast signature creation
F. Work across protocols

iii. Weaknesses
A. Mostly false alarms
B. Unreliable signatures
C. Black-box or generic filter
D. Loosely written signatures
E. Not able to detect new threats
F. Lack context
G. Slow to deploy signature

(b) SIEM

(c) Strengths

i. Visibility of logs and data sources of an organisation

   ii. Custom use-cases
   iii. Cross-event and Cross-platform correlation
   iv. Normalization
   v. Prioritization
   vi. Fast response to threats

(d) Weaknesses

   i. Overwhelm analysts with volume of data
   ii. Rely on humans for false positive filtering
   iii. Configuration hassle and lack of expertise
   iv. Use of structured databases, query takes too long
   v. Cost (hardware, software, maintenance)
   vi. Detection of zero-day attacks.

(e) Machine Learning

(f) Challenges in Network monitoring

   i. Encrypted network traffic (customer, malware)
   ii. Monitoring of internal network traffic
      A. Cost of storage
      B. PCAP vs. Netflow format

4. **Communication**

(a) Communication between SOC members

(b) Communication between SOC members and Customer (or other teams)

(c) Communication between SOC and third parties

(d) Communication between the SOC analysts and technology.

D
Online Questionnaire

Security Operations Centre (SOC) Monitoring Practice

Page 1: Introduction

We appreciate your interest in participating in our study on Security Operations Centre Monitoring Tools. This survey is intended for security analysts that work in a Security Operations Centre (SOC). The survey should take approximately 15-20 minutes to complete. Please read through these terms before agreeing to participate by continuing to the next page.

The study has three aims:

1. To capture requirements for improving security monitoring tools by gathering information on the strengths and weaknesses of existing tools.

2. To identify the thought processes of analysts in security monitoring, and data features to be included in security tool development.

3. To compare working practice and security tool use across the SOCs of different organisation types.

We will gather information for the above three parts through questions on SOC working practice, security tool use, attack indicators, and security monitoring and detection techniques applied by security analysts in SOCs in organisations.

The researchers are Louise Axon and Bushra Alahmadi, who are doctoral students in the Centre for Doctoral Training in Cybersecurity. The research is being conducted under the supervision of Professor Sadie Creese, Professor Michael Goldsmith and Professor Ivan Martinovic.

1. Why have I been invited to take part?

You have been invited to take part because of your experience working as a security analyst in a SOC.

2. Do I have to take part?

You can ask questions about the study before deciding whether to participate. You can choose whether you participate and, if you agree to participate, you may withdraw yourself and your data from the study without penalty at any time, and without giving a reason, by advising the researchers of this decision.

3. Are there any potential risks in taking part?

There are no known risks or disadvantages of taking part and no specific preparatory requirements, as we strive to protect your confidentiality, unless you explicitly agree that the type or nature of your company can be mentioned in publications arising from the research.

4. What happens to the research data provided?

The survey responses are recorded anonymously, and all participants in this study will remain anonymous. The data will be stored in a password-protected file on the researcher's computer, and will be destroyed at the end of the project. Only the researchers will have access to the personal data collected in the survey, and this data will be kept separately from the interview responses, if you later take part in an interview.

5. Will the research be published?

The results of this study may be published in conference proceedings, peer-reviewed journals and online in University archives. The results of the study will also contribute towards the DPhil theses of the two researchers, which will be published online. No personal data from this study will be published in any of the above; all study participants will remain anonymous and the data collected will be stored in a password-protected file on the researchers' computers, accessible only by the researchers.

The University of Oxford is committed to the dissemination of its research for the benefit of society and the economy and, in support of this commitment, has established an online archive of research materials. This archive includes digital copies of student theses successfully submitted as part of a University of Oxford postgraduate degree programme. Holding the archive online gives easy access for researchers to the full text of freely available theses, thereby increasing the likely impact and use of that research.

If you agree to participate in this project, the research will be written up as a thesis. On successful submission of the thesis, it will be deposited both in print and online in the University archives, to facilitate its use in future research. The thesis will be published with open access, meaning it will be available to every internet user.

6. Who has reviewed this project?

This project has been reviewed by, and received ethics clearance through, the University of Oxford Central University Research Ethics Committee.

This research is funded by the Engineering and Physical Sciences Research Council (EPSRC). The results of the study may be published in academic conferences or journals, and will be published in the researchers' theses.

7. Who do I contact if I have a concern about the study or I wish to complain?

If you have a concern about any aspect of this project, please speak to the researchers, Louise Axon and Bushra Alahmadi (louise.axon@cs.ox.ac.uk, bushra.alahmadi@cs.ox.ac.uk) or their supervisors, Professor Sadie Creese (sadie.creese@cs.ox.ac.uk), Professor Michael Goldsmith (michael.goldsmith@cs.ox.ac.uk), and Professor Ivan Martinovic (ivan.martinovic@cs.ox.ac.uk) who will do their best to answer your query. The researcher should acknowledge your concern within 10 working days and give you an indication of how he/she intends to deal with it. If you remain unhappy or wish to make a formal complaint, please contact the chair of the Research Ethics Committee at the University of Oxford (using the contact details below) who will seek to resolve the matter in a reasonably expeditious manner:

Chair, **Social Sciences & Humanities Inter-Divisional Research Ethics Committee**; Email: ethics@socsci.ox.ac.uk; Address: Research Services, University of Oxford, Wellington Square, Oxford OX1 2JD

Please note that you may only participate in this survey if you are 18 years of age. If you agree to participate and have read the terms above, please continue to the next page to get started.

Page 2: Preliminary Questions

What is your job title? ✱ *Required*

How many years' experience do you have in your role? ✱ *Required*

- 0 - 3
- 3 - 5
- 5 - 7
- 7 - 10
- 10 - 15
- 15 +
- I prefer not to say

How would you rate your level of expertise in network-security monitoring?

- Very low
- Low
- Medium
- High
- Very high

What is the type of organisation you work in? ✱ *Required*

- Government
- Financial Services
- Healthcare Providers

- Higher Education
- K-12
- Technology (Non-security focus)
- Technology (security focus)
- Manufacturing
- Media and Entertainment
- Travel, Hospitality, and Transportation
- Retail
- Other

If you selected Other, please specify:

How would you describe the size of your organisation ✱ *Required*

- Small enterprise
- Medium enterprise
- Large enterprise
- I don't know
- Other

If you selected Other, please specify:

How many security analysts work in your organisation's SOC? ✱ *Required*

- 1 - 9
- 10 - 19
- 20 - 29
- 30 - 39
- 40 - 49
- 50 - 99
- 100 - 199
- 200 +
- I prefer not to say

How many network monitoring devices (e.g. a single IDS, firewall) are you personally responsible for monitoring in the SOC you work in? ✱ *Required*

- 1
- 2 - 4
- 5 - 6
- 7 - 8
- 9 - 10
- 11 - 12
- 13- 14
- 15 - 16
- 17 - 18
- 19 - 20
- 21+
- I don't know

How many network elements does a single monitoring device observe? *Optional*

In which country is the SOC you work in located? ✱ *Required*

Which tasks are you required to carry out in your role? ✱ *Required*

- [ ] Monitor IDS
- [ ] Monitoring network and systems logs
- [ ] Track and trace intruders
- [ ] Perform forensic evidence collection
- [ ] Produce technical documents
- [ ] Perform artifact analysis
- [ ] Incident handling
- [ ] Perform security policy development
- [ ] Publish advisories or alerts
- [ ] Perform virus handling
- [ ] Provide and answer a hotline
- [ ] Provide training and security awareness
- [ ] Pursue legal investigations
- [ ] Vulnerability handling
- [ ] Security product development
- [ ] Vulnerability scanning
- [ ] Vulnerability assessments
- [ ] Security configuration administration
- [ ] Penetration testing
- [ ] Other

If you selected Other, please specify:

How many network security alerts does your organisation **receive** on average daily? ✱ *Required*

- Less Than 5K
- 5K–10K
- 10K–50K
- 50K–100K
- 100K–150K
- Over 150K
- I don't know
- Other

If you selected Other, please specify:

How many network security alerts does your organisation **investigate** on average daily? *Optional*

Please enter a whole number (integer).

Describe your usual SOC responsibilities ✱ *Required*

Approximately how many legitimate network security alerts does your organisation remediate on average daily? *Optional*

Please enter a whole number (integer).

Which of the following does your SOC mostly focus on? You may select more than one if necessary. ✱ *Required*

- [ ] Threat prevention
- [ ] Threat detection
- [ ] Threat remediation
- [ ] Recovery
- [ ] Other

If you selected Other, please specify:

Page 3: Network Monitoring

What are the main network security threats your organisation faces? ✱ *Required*

- [ ] Denial of service attacks
- [ ] Abnormal user activity
- [ ] Scanning (reconnaissance e.g. port scans)
- [ ] Unauthorised access attempts
- [ ] Viruses, Worms, Trojans
- [ ] Abnormal network activity
- [ ] Ransomware
- [ ] Advanced Persistent Threats (APTs)
- [ ] Next-generation malware (e.g. Bots)
- [ ] Phishing emails (including spear-phishing)
- [ ] Brute-force attacks
- [ ] Insider threats
- [ ] Web defacement
- [ ] Policy violation (e.g. gaming, streaming)
- [ ] Other

If you selected Other, please specify:

Which of these network security incidents do your systems detect? ✱ *Required*

- [ ] Denial of service attacks
- [ ] Abnormal user activity
- [ ] Scanning (reconnaissance e.g. port scans)
- [ ] Unauthorised access attempts

- [ ] Viruses, Worms, Trojans
- [ ] Abnormal network activity
- [ ] Ransomware
- [ ] Advanced Persistent Threats (APTs)
- [ ] Next-generation malware (e.g. Bots)
- [ ] Phishing emails (including spear-phishing)
- [ ] Brute-force attacks
- [ ] Insider threats
- [ ] Web defacement
- [ ] Policy violation (e.g. gaming, streaming)
- [ ] Other

If you selected Other, please specify:

Which of the following network security incidents do you believe you could do better at addressing if you had the tools? ✱ *Required*

- [ ] Denial of service attacks
- [ ] Abnormal user activity
- [ ] Scanning (reconnaissance e.g. port scans)
- [ ] Unauthorised access attempts
- [ ] Viruses, Worms, Trojans
- [ ] Abnormal network activity
- [ ] Ransomware
- [ ] Advanced Persistent Threats (APTs)
- [ ] Next-generation malware (e.g. Bots)
- [ ] Phishing emails (including spear-phishing)
- [ ] Brute-force attacks

- [ ] Insider threats
- [ ] Web defacement
- [ ] Policy violation (e.g. gaming, streaming)
- [ ] Other

If you selected Other, please specify:

Which of the following network monitoring tools do you use? ✱ *Required*

- [ ] Intrusion Detection System (IDS) - signature-based (e.g Snort, Cisco Secure IDS)
- [ ] Intrusion Detection System (IDS) - anomaly based (e.g. Spade)
- [ ] Intrusion Detection System (IDS) - using machine learning
- [ ] Data collection/ log aggregation (e.g. Splunk)
- [ ] Security visualisations
- [ ] Network monitoring (e.g. Argus, Bro)
- [ ] Text-based data presentation (e.g. Wireshark/tcpdump, Nmap)
- [ ] Information and Event Management (SIEM)
- [ ] Policy management/profiling/posture assessment tool (e.g. Cisco ISE)
- [ ] DNS Enforcer/Intelligent Proxy (e.g. Cisco Umbrella)
- [ ] Other

If you selected Other, please specify:

If you selected Intrusion Detection System (IDS) - using machine learning or security

visualisations in the previous question, please specify the tools used.

Which network security data sources do you monitor in your work? For each source you monitor, please indicate whether you monitor the source using a Security Incident and Event Management (SIEM) tool or individually. ✱ *Required*

	Monitor using a SIEM	Monitor individually	Do not monitor	I don't know
Firewall logs	☐	☐	☐	☐
Network packet captures	☐	☐	☐	☐
Netflow	☐	☐	☐	☐
Host logs	☐	☐	☐	☐
Server logs	☐	☐	☐	☐
IDS logs	☐	☐	☐	☐
IPS logs	☐	☐	☐	☐
Web proxy logs	☐	☐	☐	☐
Website logs	☐	☐	☐	☐
Antivirus logs	☐	☐	☐	☐
Anomaly detection alerts	☐	☐	☐	☐
Domain Name System (DNS) logs	☐	☐	☐	☐
Authentication logs	☐	☐	☐	☐

Are there any other network security data sources you monitor in your work that were not addressed in the question above? If so, please specify.

How important are the following features in choosing an ideal network monitoring system? ✱
Required

	Very unimportant	Unimportant	Neutral	Important	Very important	I don't know
Cost of required storage	☐	☐	☐	☐	☐	☐
Cost (money)	☐	☐	☐	☐	☐	☐
Required processing power	☐	☐	☐	☐	☐	☐
Number of false positives	☐	☐	☐	☐	☐	☐
Number of false negatives	☐	☐	☐	☐	☐	☐
Detection accuracy	☐	☐	☐	☐	☐	☐
Cross-correlation of logs from different vendor sources (cross-vendor API compatibility)	☐	☐	☐	☐	☐	☐
Data Visualisations	☐	☐	☐	☐	☐	☐
Easy to setup/configure/maintain	☐	☐	☐	☐	☐	☐
Customisable	☐	☐	☐	☐	☐	☐
Speed in aggregating evidence/triaging events	☐	☐	☐	☐	☐	☐
Open-source	☐	☐	☐	☐	☐	☐
Compatibility with existing hardware	☐	☐	☐	☐	☐	☐

Sophisticated alert-management capabilities	☐	☐	☐	☐	☐	☐

Do you have any further comments on any of the topics covered on this page?

Page 4: Network Monitoring Features

How important are the following data sources in detecting malicious activity on the network? ✱
Required

	Very unimportant	Unimportant	Neutral	Important	Very important	I don't know
Network traffic traces	☐	☐	☐	☐	☐	☐
Internal network traffic	☐	☐	☐	☐	☐	☐
IDS alert logs	☐	☐	☐	☐	☐	☐
IPS alert logs	☐	☐	☐	☐	☐	☐
Web proxies that log every outbound connection	☐	☐	☐	☐	☐	☐
DHCP servers that log dynamic IP address assignments	☐	☐	☐	☐	☐	☐
VPN servers that log remote connections to the enterprise network	☐	☐	☐	☐	☐	☐
Windows domain controllers that log authentication attempts within the corporate domain	☐	☐	☐	☐	☐	☐

Antivirus software that logs the results of malware scans on end hosts.	☐	☐	☐	☐	☐	☐
Network firewall	☐	☐	☐	☐	☐	☐
Data server logs	☐	☐	☐	☐	☐	☐
Anomaly detection logs	☐	☐	☐	☐	☐	☐
DNS logs	☐	☐	☐	☐	☐	☐
Security vulnerability mailing lists/blog announcements	☐	☐	☐	☐	☐	☐

Do you rely on other data sources we have missed? If yes, please specify them below.

How important are the following network traffic features in detecting malicious activity on the network? ✱ *Required*

	Very unimportant	Unimportant	Neutral	Important	Very important	I don't know
Source IP	☐	☐	☐	☐	☐	☐
Destination IP	☐	☐	☐	☐	☐	☐
Source port	☐	☐	☐	☐	☐	☐
Destination port	☐	☐	☐	☐	☐	☐
Protocol	☐	☐	☐	☐	☐	☐
Packet size	☐	☐	☐	☐	☐	☐

Byte size	☐	☐	☐	☐	☐	☐
Packet content	☐	☐	☐	☐	☐	☐
Destination domain URL	☐	☐	☐	☐	☐	☐
HTTP status code	☐	☐	☐	☐	☐	☐
Web referrer	☐	☐	☐	☐	☐	☐
Domain reputation	☐	☐	☐	☐	☐	☐
Domain category	☐	☐	☐	☐	☐	☐
User-agent string	☐	☐	☐	☐	☐	☐
Sent and received bytes	☐	☐	☐	☐	☐	☐
DNS response	☐	☐	☐	☐	☐	☐

Do you rely on other network features we have missed? If yes, please specify them below.

How important are the following Indicators of Compromise in detecting malicious activity on the network? ✱ *Required*

	Very unimportant	Unimportant	Neutral	Important	Very important	I don't know
Unusual outbound network traffic (sudden spike in traffic)	☐	☐	☐	☐	☐	☐

Anomalies in privileged user account activity	☐	☐	☐	☐	☐	☐
Geographical irregularities	☐	☐	☐	☐	☐	☐
Log-in red flags	☐	☐	☐	☐	☐	☐
Swells In database read volume	☐	☐	☐	☐	☐	☐
HTML response sizes	☐	☐	☐	☐	☐	☐
Large numbers of requests for the same file	☐	☐	☐	☐	☐	☐
Mismatched port-application traffic	☐	☐	☐	☐	☐	☐
Suspicious registry or system file changes	☐	☐	☐	☐	☐	☐
DNS request anomalies	☐	☐	☐	☐	☐	☐
Unexpected patching of systems	☐	☐	☐	☐	☐	☐
Mobile device profile changes	☐	☐	☐	☐	☐	☐
Bundles of data in the wrong places	☐	☐	☐	☐	☐	☐
Web traffic with inhuman behavior	☐	☐	☐	☐	☐	☐
Signs of DDoS activity	☐	☐	☐	☐	☐	☐

Users' complaint of slow system	☐	☐	☐	☐	☐	☐

Do you rely on other Indicators of Compromise we have missed? If yes, please specify them below.

How much do you rely on your experience in analysing and aggregating data sources to detect malicious activity as apposed to relying on security monitoring system alerts? ✱ *Required*

- ☐ 1% - 10% of the time
- ☐ 11% - 30% of the time
- ☐ 31% -50% of the time
- ☐ 51% - 70% of the time
- ☐ 71% - 80% of the time
- ☐ 81% - 100% of the time
- ☐ I don't know

In what ways (if any) does your experience aid in network security monitoring tasks?

How do you choose the security alerts you process?

- ☐ Affected subnet/business sector criticality
- ☐ Random sampling
- ☐ Based on problems faced recently

- [ ] Based on awareness of normal network activity
- [ ] New announced vulnerabilities in security blogs
- [ ] Alert severity rating
- [ ] Classifying malicious executables to a malware family
- [ ] Other

If you selected Other, please specify:

Do you have any further comments on any of the topics covered on this page?

Page 5: Assertions: Accuracy and Usability of Network Monitoring Tools

Network Monitoring Tools: Please indicate your level of agreement with the following assertions. ✱ *Required*

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The monitoring tools I use frequently produce false positive results (they detect a security event when there was not actually a security event)	☐	☐	☐	☐	☐
The monitoring tools I use frequently produce false negative results (they fail to detect a security event that occurs)	☐	☐	☐	☐	☐
I sometimes rely on my experience and intuition to detect attacks rather than monitoring system alerts	☐	☐	☐	☐	☐
I rely on my custom scripts to aggregate the data I need to analyse an incident	☐	☐	☐	☐	☐
To detect malicious activity on the network I deploy custom created scripts	☐	☐	☐	☐	☐

Intrusion Detection Systems: Please indicate your level of agreement with the following assertions. ✱ *Required*

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I believe that current IDS are inadequate in detecting attacks	☐	☐	☐	☐	☐
Current IDS solutions are incapable of coping with the high data rates	☐	☐	☐	☐	☐
Current IDS solutions lack in effective and speedy threat detection and response	☐	☐	☐	☐	☐
Current IDS solutions are built on the assumption that threats are observed as they enter the network in specific perimeter points at the Internet edge	☐	☐	☐	☐	☐
The number of alerts generated by most IDS are overwhelming	☐	☐	☐	☐	☐
To reduce the number of false positives, I limit the IDS signature set to focus on important attacks	☐	☐	☐	☐	☐
I pick IDS alerts of interest based on problems we have been having lately	☐	☐	☐	☐	☐

Data Presentation Tools: Please indicate your level of agreement with the following assertions. ✱ *Required*

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Data presentation tools, such as visualisations, are important in my work	☐	☐	☐	☐	☐
Data presentation tools such as visualisations can help me to detect incidents that are missed by the automated systems, or that do not fit the automated attack detection profile	☐	☐	☐	☐	☐
Visualisations are useful for finding interesting patterns in raw network packet data	☐	☐	☐	☐	☐
Visual distractions sometimes mean I miss important information that is conveyed visually	☐	☐	☐	☐	☐

Do you have any further comments on any of the topics covered on this page?

Page 6: Assertions: Working Practice in SOCs

The "Human in the Loop": Please indicate your level of agreement with the following assertions. ✱ *Required*

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
For my monitoring work, it is important that I maintain a continuous awareness of the network security state	☐	☐	☐	☐	☐
It is important to have a "human in the loop" for the detection and preliminary analysis of potential security events - this process cannot be carried out by automated systems alone	☐	☐	☐	☐	☐
Human analysts monitoring the network are capable of detecting network anomalies that are missed by automated systems	☐	☐	☐	☐	☐
The monitoring setup I use enables me to detect network anomalies that are missed by automated systems	☐	☐	☐	☐	☐
I am often required to make decisions on the accuracy of the alerts produced by automated systems	☐	☐	☐	☐	☐

Maintaining awareness of the network security state is important in enabling me to make decisions on the accuracy of alerts produced by automated monitoring systems	☐	☐	☐	☐	☐

Data Fusion: Please indicate your level of agreement with the following assertions. ✱
Required

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
In monitoring , I am required to watch multiple monitors depicting different data at one time	☐	☐	☐	☐	☐
I am required to watch multiple dashboards on the same monitor depicting different data at one time	☐	☐	☐	☐	☐
I am required to simultaneously monitor information from multiple monitoring sources (eg. network traffic, IDS logs, firewall logs, etc)	☐	☐	☐	☐	☐

Monitoring the status of the network involves interacting with an IDS and other monitoring tools in addition to following external information resources for vulnerabilities	☐	☐	☐	☐	☐
I am required to simultaneously monitor the state of multiple network elements (e.g. individual machines, database server, web server, etc.)	☐	☐	☐	☐	☐
I am required to monitor the network, while carrying out other tasks simultaneously (e.g. responding to emails, carrying out incident response)	☐	☐	☐	☐	☐
SIEMS require collecting and storing billions of logs every day, we have limited resources and are not able to keep these logs for long periods of time	☐	☐	☐	☐	☐
Aggregation of billions of log alerts a day presented in heterogeneous data structures makes analysis a challenge	☐	☐	☐	☐	☐

Devices may generate logs that are either incomplete or inconsistent (e.g. different time-stamps) making analysis an even bigger challenge	☐	☐	☐	☐	☐
The tools I use are effective at supporting data fusion and correlation across incidents and data sources	☐	☐	☐	☐	☐

Network Configurations: Please indicate your level of agreement with the following assertions.
✱ *Required*

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Keeping up with changing configurations in the network is difficult, but necessary to provide the context needed to analyse and diagnose an alert	☐	☐	☐	☐	☐
Keeping track and knowing the network environment is important to detect attacks	☐	☐	☐	☐	☐
I track network configurations using personal memory	☐	☐	☐	☐	☐

I track network configurations by keeping an updated database of network devices and configuration changes	☐	☐	☐	☐	☐

Do you have any further comments on any of the topics covered on this page?

Page 7: Any further comments?

Do you have any further comments you would like to make about aspects of network security monitoring in SOCs that have not been covered in this survey?

Page 8: Thank you for participating.

Thank you for completing this survey. Your answers have been recorded. If you have a concern about any aspect of this project, please feel free to contact the researchers, Louise Axon and Bushra Alahmadi (louise.axon@cs.ox.ac.uk, bushra.alahmadi@cs.ox.ac.uk).

Key for selection options

9 - In which country is the SOC you work in located?

I prefer not to say
AD - Andorra
AE - United Arab Emirates
AF - Afghanistan
AG - Antigua and Barbuda
AI - Anguilla
AL - Albania
AM - Armenia
AO - Angola
AQ - Antarctica
AR - Argentina
AS - American Samoa
AT - Austria
AU - Australia
AW - Aruba
AZ - Azerbaijan
BA - Bosnia and Herzegovina
BB - Barbados
BD - Bangladesh
BE - Belgium
BF - Burkina Faso
BG - Bulgaria
BH - Bahrain
BI - Burundi
BJ - Benin
BL - Saint Barthelemy
BM - Bermuda
BN - Brunei
BO - Bolivia
BR - Brazil
BS - Bahamas, The

BT - Bhutan
BV - Bouvet Island
BW - Botswana
BY - Belarus
BZ - Belize
CA - Canada
CC - Cocos (Keeling) Islands
CD - Congo, Democratic Republic of the
CF - Central African Republic
CG - Congo, Republic of the
CH - Switzerland
CI - Cote d'Ivoire
CK - Cook Islands
CL - Chile
CM - Cameroon
CN - China
CO - Colombia
CR - Costa Rica
CU - Cuba
CV - Cape Verde
CW - Curacao
CX - Christmas Island
CY - Cyprus
CZ - Czech Republic
DE - Germany
DJ - Djibouti
DK - Denmark
DM - Dominica
DO - Dominican Republic
DZ - Algeria
EC - Ecuador
EE - Estonia
EG - Egypt
EH - Western Sahara
ER - Eritrea
ES - Spain
ET - Ethiopia
FI - Finland
FJ - Fiji
FK - Falkland Islands (Islas Malvinas)
FM - Micronesia, Federated States of
FO - Faroe Islands

FR - France
FX - France, Metropolitan
GA - Gabon
GB - United Kingdom
GD - Grenada
GE - Georgia
GF - French Guiana
GG - Guernsey
GH - Ghana
GI - Gibraltar
GL - Greenland
GM - Gambia, The
GN - Guinea
GP - Guadeloupe
GQ - Equatorial Guinea
GR - Greece
GS - South Georgia and the Islands
GT - Guatemala
GU - Guam
GW - Guinea-Bissau
GY - Guyana
HK - Hong Kong
HM - Heard Island and McDonald Islands
HN - Honduras
HR - Croatia
HT - Haiti
HU - Hungary
ID - Indonesia
IE - Ireland
IL - Israel
IM - Isle of Man
IN - India
IO - British Indian Ocean Territory
IQ - Iraq
IR - Iran
IS - Iceland
IT - Italy
JE - Jersey
JM - Jamaica
JO - Jordan
JP - Japan
KE - Kenya

KG - Kyrgyzstan
KH - Cambodia
KI - Kiribati
KM - Comoros
KN - Saint Kitts and Nevis
KP - Korea, North
KR - Korea, South
KW - Kuwait
KY - Cayman Islands
KZ - Kazakhstan
LA - Laos
LB - Lebanon
LC - Saint Lucia
LI - Liechtenstein
LK - Sri Lanka
LR - Liberia
LS - Lesotho
LT - Lithuania
LU - Luxembourg
LV - Latvia
LY - Libya
MA - Morocco
MC - Monaco
MD - Moldova
ME - Montenegro
MF - Saint Martin
MG - Madagascar
MH - Marshall Islands
MK - Macedonia
ML - Mali
MM - Burma
MN - Mongolia
MO - Macau
MP - Northern Mariana Islands
MQ - Martinique
MR - Mauritania
MS - Montserrat
MT - Malta
MU - Mauritius
MV - Maldives
MW - Malawi
MX - Mexico

MY - Malaysia
MZ - Mozambique
NA - Namibia
NC - New Caledonia
NE - Niger
NF - Norfolk Island
NG - Nigeria
NI - Nicaragua
NL - Netherlands
NO - Norway
NP - Nepal
NR - Nauru
NU - Niue
NZ - New Zealand
OM - Oman
PA - Panama
PE - Peru
PF - French Polynesia
PG - Papua New Guinea
PH - Philippines
PK - Pakistan
PL - Poland
PM - Saint Pierre and Miquelon
PN - Pitcairn Islands
PR - Puerto Rico
PS - Gaza Strip
PS - West Bank
PT - Portugal
PW - Palau
PY - Paraguay
QA - Qatar
RE - Reunion
RO - Romania
RS - Serbia
RU - Russia
RW - Rwanda
SA - Saudi Arabia
SB - Solomon Islands
SC - Seychelles
SD - Sudan
SE - Sweden
SG - Singapore

SH - Saint Helena, Ascension, and Tristan da Cunha
SI - Slovenia
SJ - Svalbard
SK - Slovakia
SL - Sierra Leone
SM - San Marino
SN - Senegal
SO - Somalia
SR - Suriname
SS - South Sudan
ST - Sao Tome and Principe
SV - El Salvador
SX - Sint Maarten
SY - Syria
SZ - Swaziland
TC - Turks and Caicos Islands
TD - Chad
TF - French Southern and Antarctic Lands
TG - Togo
TH - Thailand
TJ - Tajikistan
TK - Tokelau
TL - Timor-Leste
TM - Turkmenistan
TN - Tunisia
TO - Tonga
TR - Turkey
TT - Trinidad and Tobago
TV - Tuvalu
TW - Taiwan
TZ - Tanzania
UA - Ukraine
UG - Uganda
UM - United States Minor Outlying Islands
US - United States
UY - Uruguay
UZ - Uzbekistan
VA - Holy See (Vatican City)
VC - Saint Vincent and the Grenadines
VE - Venezuela
VG - British Virgin Islands
VI - Virgin Islands

VN - Vietnam
VU - Vanuatu
WF - Wallis and Futuna
WS - Samoa
XK - Kosovo
YE - Yemen
YT - Mayotte
ZA - South Africa
ZM - Zambia
ZW - Zimbabwe