FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of master's thesis

Title:	Analyzing Adversarial Lateral Movement Techniques on Windows Systems
Student:	Bc. Silvie Němcová
Supervisor:	Ing. Simona Fornůsek, Ph.D.
Study program:	Informatics
Branch / specialization:	Computer Security
Department:	Department of Information Security
Validity:	until the end of summer semester 2024/2025

Instructions

This thesis investigates existing lateral movement techniques outlined by MITRE ATT&CK framework specifically targeting Windows systems through the analysis of compromised systems. The primary aim of this research is to uncover evidence of adversarial lateral movement, which can be leveraged to develop detection mechanisms independent of contextual information from other hosts within the infected environment.

Within the thesis:
1. Study the common adversarial tactics and techniques used for lateral movement in cybersecurity and explain lateral movement tactics and its role in cyberattacks.
2. Propose, design and implement experimental setup (lab environment) for analyzing compromised Windows systems with focus on lateral movement.
3. Emulate and evaluate suitable and feasible detection rules for identified behavior. Examine compromised systems to identify evidence and artefacts of lateral movement, that can be further used for security detection and analysis.
4. Discuss the feasibility and effectiveness of detection methods in order to detect lateral movement based on endpoint artefacts and collected evidence.

Electronically approved by prof. Ing. Róbert Lórencz, CSc. on 15 February 2024 in Prague.

Master’s thesis

ANALYZING ADVERSARIAL LATERAL MOVEMENT TECHNIQUES ON WINDOWS SYSTEMS

Bc. Silvie Němcová

Faculty of Information Technology
Department of Information Security
Supervisor: Ing. Simona Fornůsek, Ph.D.
May 9, 2024

Citation of this thesis: Němcová Silvie. *Analyzing Adversarial Lateral Movement Techniques on Windows Systems*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2024.

Contents

List of Figures

List of Tables

List of Listings

I would like to express my deepest gratitude to all those who provided me with the support and guidance necessary to complete this thesis. First and foremost, I am immensely thankful to my partner, Tomas, for his unwavering support, patience, and encouragement throughout this challenging journey. Your presence has been a pillar of stability in times of stress and uncertainty. My sincere appreciation also goes to my parents, for their endless encouragement and for believing in me at every step. Additionally, I am deeply grateful to my thesis supervisor, Ing. Simona Fornůsek, Ph.D., whose insights and expertise were invaluable during the research and writing process.

Declaration

In Prague on May 9, 2024

Abstract

As sophisticated cyberattacks continue to evolve, cybersecurity researchers and experts are challenged to keep pace with an ever-changing threat landscape. Developing effective defense mechanisms remains a formidable task. This thesis employs a threat-informed approach to develop analytics for detecting lateral movement techniques, a critical phase in cyberattacks where adversaries expand their access and control within a compromised environment.

Typically, adversaries move laterally within their target network to reach their objectives, which are often located on more secured or isolated devices, following an initial compromise via less secured devices or phishing. This work provides an overview of lateral movement techniques, reviews and applies a threat-informed approach to cyber threat analytics development, and focuses on the in-depth analysis of one such technique, T1091 *Replication Through Removable Media*. Utilizing adjusted threat-informed cyber threat analytics development framework, this thesis analyzes lateral movement technique T1091, proposes specific detection strategies, implements them in a popular Security Information and Event Management (SIEM) system, and evaluates their performance and validity.

Ultimately, this thesis demonstrates the effectiveness of the threat-informed approach to developing and implementing cyber threat analytics and detection strategies, summarizing the process and evaluating the outcomes to enhance cybersecurity defenses.

Keywords Lateral Movement, Cyber Threat Analytics, APT, Replication Through Removable Media

Abstrakt

S pokračujícím nárustem počtu sofistikovaných kybernetických útoků rostou nároky na výzkumníky a experty v oblasti kybernetické bezpečnosti. Kybernetické hrozby se neustále vyvíjejí a mění, udržovat s nimi tempo je kritické pro vývoj efektivních bezpečnostních řešení. Vývoj obranných mechanismů je však náročný úkol. Tato práce využívá *threat-informed* přístup k vytváření analytik a vývoji detekčních mechanismů. *Threat-informed* přístup spočívá ve využívání dostupných informací získaných pomocí zpravodajství o kybernetických hrozbách (*Cyber Threat Intelligence (CTI)*), což může vést k výsledkům, které umožní implementaci a nasazení efektivnějších obranných mechanismů.

Útočníci typicky musí provést boční pohyb (*lateral movement*) v napadnutém prostředí aby dosáhli svých cílů. Obvykle je jejich prvotní přístup do cílového prostředí prostřednictvím zařízení s nižší ochranou nebo pomocí *phishingu* zatímco jejich cíl se většinou nachází na lépe zabezpečeném zařízení. Tato práce představí techniky používáné pro docílení bočního pohybu v cílovém prostředí, zkoumá a nasadí přístup k vývoji analytik a detekcí využívající zpravodajství o kybernetických hrozbách a zaměří se na hlubší analýzu jedné z těchto technik, šíření pomocí výměnných medií (například USB flash disky). Pomocí informací získaných ze zpravodajství o kybernetických hrozbách, tato práce analyzuje techniku šíření pomocí výměnných medií, navrhne metody detekce této techniky, implementuje a nasadí navžené detekce, a vyhodnotí jejich efektivitu a validitu.

Nakonec, tato práce ukazuje výhody využití informací získaných pomocí zpravodajství o kybernetických hrozbách při vývoji analytik a detekcí a disktuje kvalitu výsledků z tohoto procesu.

Klíčová slova Boční pohyb, analýza kybernetických hrozeb, APT, Šíření pomocí výměnných médií

Introduction

Motivation

Cybersecurity is confronting an increasingly complex and rapidly escalating threat landscape, dominated by sophisticated attacks. Among these, *Advanced Persistent Threat* (APT) campaigns stand out due to their complexity and persistence. APTs are well-organized efforts that manage to evade conventional security measures through the use of intricate techniques designed to operate undetected over extended periods [1]. The stealth and sophistication of these methods not only challenge existing defensive strategies but also highlight the urgent need for the development of more advanced and adaptive security measures. This evolving security environment demands a deeper understanding of APT tactics and an enhanced capability to detect and counteract them effectively.

Lateral movement, a critical post-exploitation tactic within APT campaigns, involves adversaries moving horizontally across compromised networks to extend their control and access multiple systems [2]. The significance of lateral movement in the lifecycle of a cyberattack is illustrated in Figure 1. This tactic typically occurs after the adversaries have gained initial access to a target environment, and before they inflict significant damage, such as exfiltration of valuable data typically stored on systems different from the initially compromised one. The *Unified Kill Chain* framework, which describes the common cyberattack lifecycle, categorizes lateral movement into the "Through" phase [3].

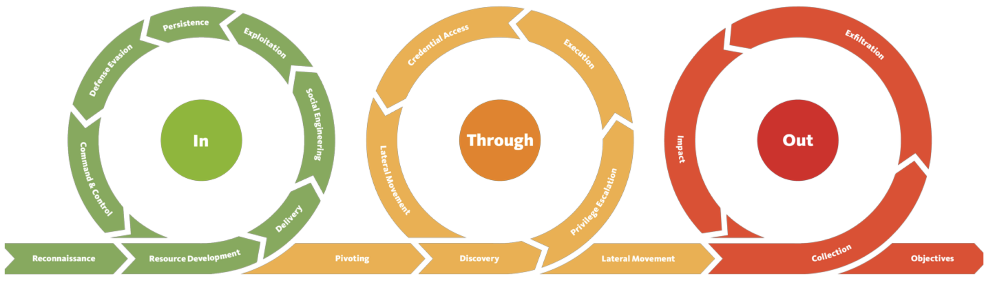


Figure 1 The Unified Kill Chain [3]. Lateral movement is classified as part of the "Through" phase of a cyberattack.

After gaining initial access, adversaries typically require an average of 62 minutes to commence lateral movement, according to recent data [4]. This period, known as the *breakout* or *dwell time*, has dramatically decreased from 101 days in 2017 [5] to just over an hour in recent years, illustrating the rapid evolution and increased efficiency of adversarial techniques. The breakout time represents a critical window for defenders to detect and mitigate threats. During this interval, adversaries execute preparatory steps necessary for lateral movement, which is often the final phase needed to solidify their presence within a victim's environment. Timely detection of lateral movement is crucial [6], as the detection can prevent the costs and damages associated with the attack.

However, detecting lateral movement presents a formidable challenge, as adversaries frequently use legitimate software and credentials to seamlessly blend their malicious activities with normal network operations [7, 8]. Traditional detection strategies often struggle to effectively identify these covert operations, highlighting the urgent need for more advanced and adaptable methodologies. The detection methodologies must be capable of distinguishing subtle anomalies from everyday activities, thus enabling defenders to respond swiftly and decisively to these stealthy maneuvers.

Thesis Overview

The theoretical foundation of this thesis is established through an introductory exploration of the lateral movement tactic as described in the MITRE ATT&CK framework [2] in Chapter 1, supplemented by a review of related work in Chapter 2. This foundational discussion sets the stage for a comprehensive examination of the methodologies employed in this research.

Chapter 3 provides an introduction to commonly used cyber threat detection approaches.

In Chapter 4, the proposed testing environment is described. This chapter includes testing environment design and steps taken to create the design. The testing environment consists of a target environment a monitoring system and a SIEM. Chapter 4 provides detailed description of the implementation and capabilities.

Chapter 3 provides an overview of detection approaches, discussing their individual drawbacks and advantages. Furthermore, Chapter 3 includes an analysis of TA0008, Lateral Movement, detection coverage by exploring existing detection rules from Sigma rule set [9] and Elastic Prebuilt Detection Rules [10].

Methods adopted and further developed to develop cyber threat analytics are detailed comprehensively in Chapter 5. This chapter formulates the analysis approach by providing detailed description of individual steps of cyber threat analytics development. This approach is based on TTP-Based Hunting [11], Finding Cyber Threats [12] and Active Defense Capability Set [13], scaled and optimized to fit individual researchers or small teams.

Finally, Chapter 6 presents the results of the lateral movement analysis conducted using the methodologies outlined in Chapter 5. This chapter specifically focuses on the examination of TA0008 Lateral Movement technique T1091 Replication Through Removable Media. Adhering to the cyber threat analytics approach, the analysis results in two novel detection rules developed in common format, Sigma. The proposed detection rules are deployed into Elastic SIEM, which allows the to be verified by emulating the adversary.

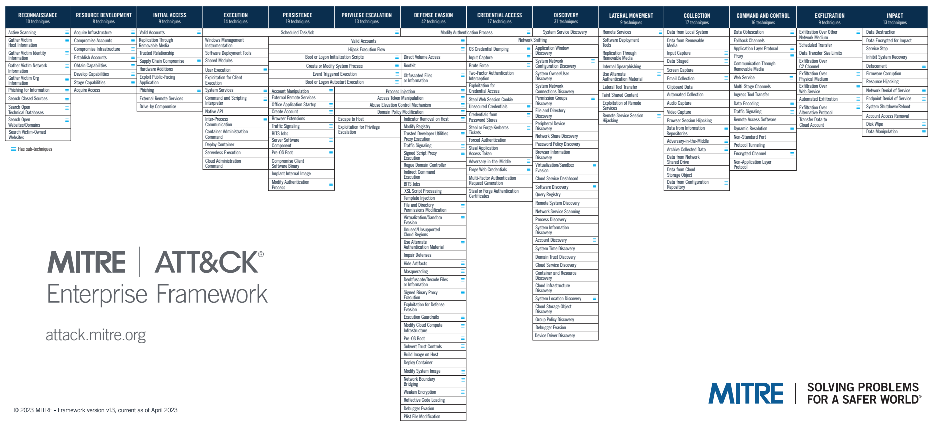
Conclusion in Chapter 7 synthesizes the contributions of this research and provides a summary of the findings. It reflects on the implications of the results for the field of cybersecurity and suggests directions for future research, underscoring the significance of this work in advancing the understanding of adversarial tactics and enhancing defensive strategies.

Chapter 1
Lateral Movement

Lateral movement refers to a post-exploitation strategy utilized by threat actors to expand their influence within a targeted environment [2]. Once adversaries breach perimeter defenses and secure initial access, lateral movement becomes a crucial phase in which they traverse across environmental segments. This cyberattack phase aims to broaden their presence and gain access to valuable resources and sensitive data [3]. Techniques used during lateral movement often exploit legitimate software, credentials, and protocols designed for remote system access [2], as detailed in the MITRE ATT&CK framework. This chapter provides an in-depth overview of the terminology, taxonomy, and techniques associated with lateral movement, delving into the tactics employed by adversaries to maneuver within and manipulate targeted environment.

1.1 Terminology and Taxonomy

The MITRE ATT&CK framework serves as an extensive knowledge base, cataloging various lateral movement techniques [14]. This section explores the terminology and taxonomy established by MITRE ATT&CK. The objective of this overview is to establish a common foundational language that facilitates a formal discussion of adversarial behaviors, enhancing the precision and clarity of the security discourse.



RECONNAISSANCE	RESOURCE DEVELOPMENT	INITIAL ACCESS	EXECUTION
10 techniques	8 techniques	9 techniques	14 techniques
Active Scanning ≡	Acquire Infrastructure ≡	Valid Accounts ≡	Scheduled Task/Job ≡
Gather Victim Host Information ≡	Compromise Accounts ≡	Replication Through Removable Media	Windows Management Instrumentation
Gather Victim Identity Information ≡	Compromise Infrastructure ≡	Trusted Relationship	Software Deployment Tools
Gather Victim Network Information ≡	Establish Accounts ≡	Supply Chain Compromise ≡	Shared Modules
Gather Victim Org Information ≡	Obtain Capabilities ≡	Hardware Additions	User Execution ≡
Phishing for Information ≡	Develop Capabilities ≡	Exploit Public-Facing Application	Exploitation for Client Execution
Search Closed Sources ≡	Stage Capabilities ≡	Phishing ≡	System Services ≡
Search Open Technical Databases ≡	Acquire Access	External Remote Services	Command and Scripting Interpreter ≡
Search Open Websites/Domains ≡		Drive-by Compromise	Native API
Search Victim-Owned Websites			Inter-Process Communication ≡
			Container Administration Command
			Deploy Container
			Serverless Execution
			Cloud Administration Command

PERSISTENCE	PRIVILEGE ESCALATION	DEFENSE EVASION
19 techniques	13 techniques	42 techniques
Scheduled Task/Job ≡	Scheduled Task/Job ≡	Modify Authentication Process ≡
Valid Accounts ≡	Valid Accounts ≡	Valid Accounts ≡
Hijack Execution Flow ≡	Hijack Execution Flow ≡	Hijack Execution Flow ≡
Boot or Logon Initialization Scripts ≡	Boot or Logon Initialization Scripts ≡	Direct Volume Access
Create or Modify System Process ≡	Create or Modify System Process ≡	Rootkit
Event Triggered Execution ≡	Event Triggered Execution ≡	Obfuscated Files or Information ≡
Boot or Logon Autostart Execution ≡	Boot or Logon Autostart Execution ≡	Process Injection ≡
Account Manipulation ≡	Process Injection ≡	Access Token Manipulation ≡
External Remote Services	Access Token Manipulation ≡	Abuse Elevation Control Mechanism ≡
Office Application Startup ≡	Abuse Elevation Control Mechanism ≡	Domain Policy Modification ≡
Create Account ≡	Domain Policy Modification ≡	Indicator Removal on Host ≡
Browser Extensions	Escape to Host	Modify Registry
Traffic Signaling ≡	Exploitation for Privilege Escalation	Trusted Developer Utilities Proxy Execution ≡
BITS Jobs		Traffic Signaling ≡
Server Software Component ≡		Signed Script Proxy Execution ≡
Pre-OS Boot ≡		Rogue Domain Controller
Compromise Client Software Binary		Indirect Command Execution
Implant Internal Image		BITS Jobs
Modify Authentication Process ≡		XSL Script Processing
		Template Injection
		File and Directory Permissions Modification ≡
		Virtualization/Sandbox Evasion ≡
		Unused/Unsupported Cloud Regions
		Use Alternate Authentication Material ≡
		Impair Defenses
		Hide Artifacts ≡
		Masquerading ≡
		Deobfuscate/Decode Files or Information ≡
		Signed Binary Proxy Execution ≡
		Exploitation for Defense Evasion
		Execution Guardrails ≡
		Modify Cloud Compute Infrastructure ≡
		Pre-OS Boot ≡
		Subvert Trust Controls ≡
		Build Image on Host
		Deploy Container
		Modify System Image ≡
		Network Boundary Bridging ≡
		Weaken Encryption ≡
		Reflective Code Loading
		Debugger Evasion
		Plist File Modification

CREDENTIAL ACCESS	DISCOVERY	LATERAL MOVEMENT
17 techniques	31 techniques	9 techniques
Modify Authentication Process ≡	System Service Discovery	Remote Services ≡
Network Sniffing	Network Sniffing	Software Deployment Tools
OS Credential Dumping ≡	Application Window Discovery	Replication Through Removable Media
Input Capture ≡	System Network Configuration Discovery ≡	Internal Spearphishing
Brute Force ≡	System Owner/User Discovery	Use Alternate Authentication Material ≡
Two-Factor Authentication Interception	System Network Connections Discovery	Lateral Tool Transfer
Exploitation for Credential Access	Permission Groups Discovery ≡	Taint Shared Content
Steal Web Session Cookie	File and Directory Discovery	Exploitation of Remote Services
Unsecured Credentials ≡	Peripheral Device Discovery	Remote Service Session Hijacking ≡
Credentials from Password Stores ≡	Network Share Discovery	
Steal or Forge Kerberos Tickets ≡	Password Policy Discovery	
Forced Authentication	Browser Information Discovery	
Steal Application Access Token	Virtualization/Sandbox Evasion ≡	
Adversary-in-the-Middle ≡	Cloud Service Dashboard	
Forge Web Credentials ≡	Software Discovery ≡	
Multi-Factor Authentication Request Generation	Query Registry	
Steal or Forge Authentication Certificates	Remote System Discovery	
	Network Service Scanning	
	Process Discovery	
	System Information Discovery	
	Account Discovery ≡	
	System Time Discovery	
	Domain Trust Discovery	
	Cloud Service Discovery	
	Container and Resource Discovery	
	Cloud Infrastructure Discovery	
	System Location Discovery ≡	
	Cloud Storage Object Discovery	
	Group Policy Discovery	
	Debugger Evasion	
	Device Driver Discovery	

COLLECTION	COMMAND AND CONTROL	EXFILTRATION	IMPACT
17 techniques	16 techniques	9 techniques	13 techniques
Data from Local System	Data Obfuscation ≡	Exfiltration Over Other Network Medium ≡	Data Destruction
Data from Removable Media	Fallback Channels	Scheduled Transfer	Data Encrypted for Impact
Input Capture ≡	Application Layer Protocol ≡	Data Transfer Size Limits	Service Stop
Data Staged ≡	Proxy ≡	Exfiltration Over C2 Channel	Inhibit System Recovery
Screen Capture	Communication Through Removable Media	Exfiltration Over Physical Medium ≡	Defacement ≡
Email Collection ≡	Web Service ≡	Exfiltration Over Web Service ≡	Firmware Corruption
Clipboard Data	Multi-Stage Channels	Automated Exfiltration ≡	Resource Hijacking
Automated Collection	Ingress Tool Transfer	Exfiltration Over Alternative Protocol ≡	Network Denial of Service ≡
Audio Capture	Data Encoding ≡	Transfer Data to Cloud Account	Endpoint Denial of Service ≡
Video Capture	Traffic Signaling ≡		System Shutdown/Reboot
Browser Session Hijacking	Remote Access Software		Account Access Removal
Data from Information Repositories ≡	Dynamic Resolution ≡		Disk Wipe ≡
Adversary-in-the-Middle ≡	Non-Standard Port		Data Manipulation ≡
Archive Collected Data ≡	Protocol Tunneling		
Data from Network Shared Drive	Encrypted Channel ≡		
Data from Cloud Storage Object	Non-Application Layer Protocol		
Data from Configuration Repository ≡			

≡ Has sub-techniques
MITRE | ATT&CK®
Enterprise Framework
attack.mitre.org
© 2023 MITRE • Framework version v13, current as of April 2023
MITRE | SOLVING PROBLEMS FOR A SAFER WORLD®

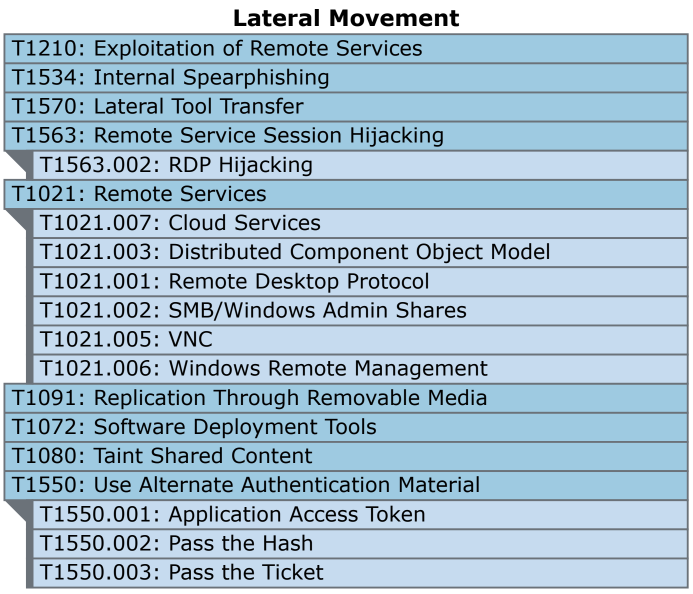
Figure 1.1 MITRE ATT&CK overall structure [15].

The ATT&CK framework models the behavior of adversaries as observed across various real-world adversary groups [14]. This model is crucial for understanding and anticipating the strategies employed by threat actors. The main components of the model are [14]:

- **Tactics**: Represent the *why*—the objective behind an adversary's actions within the context of their attack lifecycle.
- **Techniques**: Detail the *how*—the methods adversaries employ to achieve their tactical goals.
- **Sub-techniques**: Offer a more detailed or lower-level description of adversarial behavior, providing granularity to the techniques.
- **Procedures**: Describe the specific implementations used by adversaries to execute techniques or sub-techniques, often illustrated through real-world examples.

1.2 TA0008: Lateral Movement Techniques

Lateral movement techniques, categorized under the MITRE ATT&CK TA0008 tactic demonstrate how adversaries traverse through target environments to expand their foothold and escalate privileges within compromised systems.



Lateral Movement

- T1210: Exploitation of Remote Services
- T1534: Internal Spearphishing
- T1570: Lateral Tool Transfer
- T1563: Remote Service Session Hijacking
  - T1563.002: RDP Hijacking
- T1021: Remote Services
  - T1021.007: Cloud Services
  - T1021.003: Distributed Component Object Model
  - T1021.001: Remote Desktop Protocol
  - T1021.002: SMB/Windows Admin Shares
  - T1021.005: VNC
  - T1021.006: Windows Remote Management
- T1091: Replication Through Removable Media
- T1072: Software Deployment Tools
- T1080: Taint Shared Content
- T1550: Use Alternate Authentication Material
  - T1550.001: Application Access Token
  - T1550.002: Pass the Hash
  - T1550.003: Pass the Ticket

Figure 1.2 Overview of TA0008 Lateral Movement techniques.

1.2.1 T1210: Exploitation of Remote Services

Adversaries often leverage remote services to facilitate lateral movement within a target network. Exploitation occurs when adversaries capitalize on an existing vulnerability within software [16]. A typical objective of exploiting remote services is to gain unauthorized access to a remote system.

For this technique to be successful, the target must possess vulnerabilities. To ascertain this, adversaries may attempt to identify vulnerable software deployed within the target network, absence of certain software patches, security software, network configurations, and other pertinent data. Numerous well-documented vulnerabilities exist in common services such as `SMB` and `RDP`, as well as applications commonly utilized within internal networks, such as `MySQL` and web server services [17].

1.2.2 T1534: Internal Spearphishing

Adversaries may employ internal spearphishing to obtain access to additional information or exploit other users within a target network. This technique entails a multi-staged campaign in which adversaries exploit a compromised user account to heighten the probability of deceiving a target into falling for the phishing attempt [18].

Adversaries may utilize T1566.001 Spearphishing Attachment [19] or T1566.002 Spearphishing Link [20] as components of internal spearphishing to accomplish their objectives [18].

1.2.3 T1570: Lateral Tool Transfer

Adversaries may transfer tools or other files across a target network using inherent file sharing protocols like file sharing over `SMB`, Windows Admin Shares or via `RDP` [21]. Furthermore, files can be transferred utilizing native or other available tools on the victim system, such as `scp`, `curl`, `sftp`, and `ftp` [21].

1.2.4 T1563: Remote Service Session Hijacking

Adversaries may exploit preexisting remote service sessions to move laterally within a target environment. Users may log into a remote service like `SSH` or `RDP` using valid credentials. Following the establishment of a remote service session, adversaries may hijack the session to conduct actions on remote systems [22].

1.2.5 T1021: Remote Services

Adversaries might leverage valid accounts to access remote services deployed in a target environment. This technique encompasses various sub-techniques, each elucidating the exploitation of a specific remote service or concept [16].

1.2.6 T1091: Replication Though Removable Media

Adversarial lateral movement within disconnected or air-gapped environments can be achieved by spreading through removable media [23]. Procedures related to technique T1091 include taking advantage of Autorun features [24, 25]. Adversaries may alter files on removable media or deceive a victim user into executing files stored on such media on a separate system. This method enables lateral movement in air-gapped environments [23].

1.2.7 T1072: Software Deployment Tools

Adversaries may abuse third-party administration, monitoring and deployment software deployed in a target environment [26].

1.2.8 T1080: Taint Shared Content

Delivering payloads to remote systems is possible by tainting shared content on shared storage locations like network drives or code repositories. Adversaries may insert malicious programs, scripts, or exploit code into otherwise valid files [27]. Once a victim accesses the tainted shared content, the malicious portion can be executed on a target system.

1.2.9 T1550: Use Alternate Authentication Material

Adversaries may utilize alternate authentication material such as password hashes, Kerberos tickets, or application access tokens to facilitate lateral movement. By employing alternate authentication, adversaries may bypass system access controls [28].

1.3 Data Sources Related to Lateral Movement

Lateral movement inherently leaves a digital footprint that reflects the progression of malicious actors within the targeted environment. This trail of activity forms a potential foundation for detection and mitigation strategies. However, the effectiveness of these strategies is heavily dependent on the quality and integrity of the data sources leveraged. It is crucial to recognize that the evidence left by lateral movements may not always be pristine; it can be fragmented or even deliberately obfuscated, complicating the detection process. This section explores the various data sources that can indicate lateral movement, discusses their potential limitations, and suggests methods to enhance their reliability and utility in cybersecurity defenses.

In this research, a cautious approach is adopted, assuming that the evidence collected remains unaltered, although it is acknowledged that there is an ongoing challenge in assessing its completeness. This assumption provides a solid foundation for exploring the landscape of lateral movement detection. Nonetheless, there is a conscientious awareness of the need for further inquiry into the nuances of data integrity and completeness, recognizing that these factors critically influence the effectiveness of detection strategies.

The MITRE ATT&CK framework provides invaluable insights into the myriad data sources essential for the effective detection of lateral movement. It identifies 15 distinct data components related to lateral movement, ranging from system logs and network traffic to authentication records and process monitoring, as illustrated in Figure 1.3. By comprehensively understanding and harnessing these data sources, cybersecurity practitioners can significantly bolster their defenses against the sophisticated lateral movement tactics employed by adversaries.

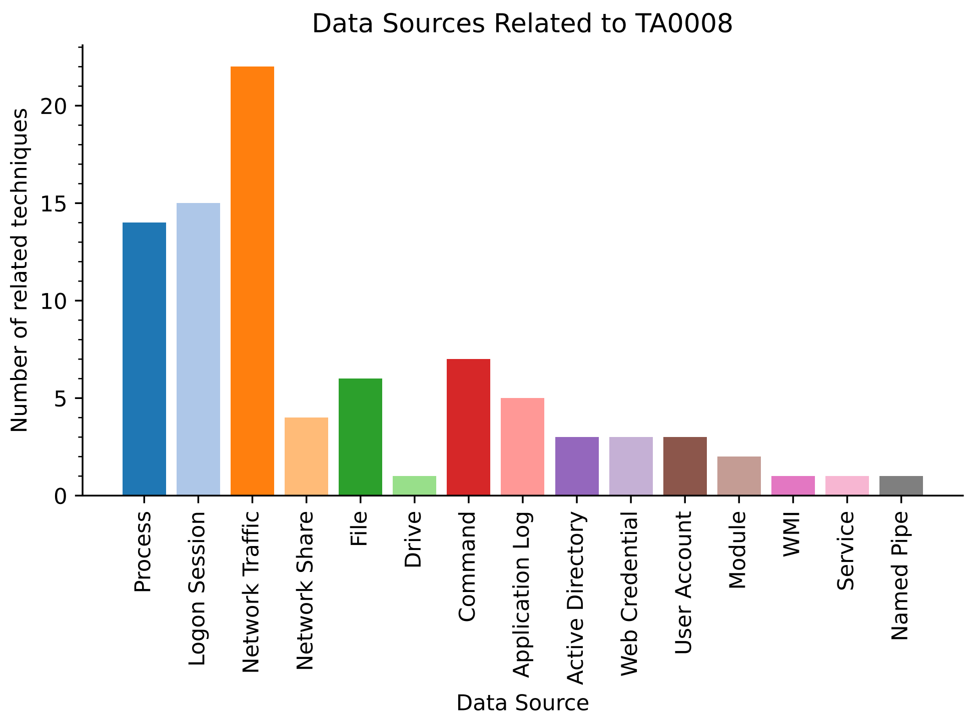


Figure 1.3 Data Sources related to TA0008 Lateral Movement.

The lateral movement techniques identified by the ATT&CK framework produce digital artifacts illustrated in Figure 1.3. Host-based data include application log [29], command events [30], file events [31], process events [32] and others. Nonetheless, analysis of network traffic data also plays an important role in detecting various malicious activities. These are, among others, the use of known malicious payloads [33], anomalous logon events [34] and events related to network shares [35]. Careful examination of network traffic patterns can reveal instances of abuse involving third-party administration and management software [33].

1.3.1 Network Data Sources

Network data sources are integral to cybersecurity, offering critical insights through the analysis of network traffic. Data from network interactions is preserved in analyzable formats, such as `pcap` files [33], facilitating a thorough examination for threat detection purposes. This capability allows security analysts to detect patterns, anomalies, and potential threats by meticulously analyzing the traffic flowing across the network.

The detection of lateral movement relies heavily on meticulous monitoring of network traffic [36, 37, 8] to identify known malicious activities, indicators of compromise (IoCs), and unusual system behaviors. The MITRE ATT&CK framework underscores the importance of monitoring connection creations, flows, and communication content as critical methods for effectively spotting adversarial maneuvers [17, 18, 21, 16, 27]. This targeted surveillance helps cybersecurity professionals pinpoint potential threats by observing deviations from normal network patterns and suspicious interactions within the network.

1.3.2 Endpoint Data Sources

The MITRE ATT&CK framework outlines numerous endpoint data sources critical for detecting lateral movement techniques, each providing valuable insights for comprehensive analysis and effective detection. Figure 1.3 reveals the importance of monitoring events related to logon, processes, files and command execution. Monitoring these data points enables a comprehensive view of both normal and potentially malicious activities within hosts, with command execution data often acting as a detailed subset of process creation information.

Monitoring the file system, or specific sections thereof, is a critical mechanism for identifying system compromises. This surveillance is particularly valuable in detecting signs of lateral movement, which often involves modifications to files and directories as adversaries attempt to navigate and gain control over the compromised environment. Vigilant monitoring of *file creation*, *modification*, and *metadata* alterations is crucial, providing key insights into unauthorized changes that may indicate an intrusion [21, 23, 27]. By closely examining these activities, security teams can track the spread of a threat across the network, pinpoint the techniques used, and implement effective countermeasures.

Files generated by remote service software and abnormal behavior from newly established mount points also serve as crucial cues, often indicating the exploitation of network services and systems to spread an attack within an organization [31, 38, 35].

Monitoring remote services within the network environment is crucial for detecting abnormal logon sessions, which often signal unauthorized access attempts [39, 34]. Additionally, application *crashes* may indicate potential service abuse or exploitation, serving as important red flags for security teams to investigate [29]. Similarly, the detection of uncommonly named pipes and unusual service metadata provides critical data points. These elements are particularly valuable for identifying lateral movement, as they often reveal subtle manipulations of system processes characteristic of advanced persistent threats [40, 41].

Chapter 2

Related Work

2.1 Cyber Threat Analytics

In the evolving landscape of cybersecurity, effective detection mechanisms are crucial for protecting systems and networks against emerging threats. To develop such, thorough cyber threat analytics are often necessary. This section discusses various approaches and methodologies within detection engineering, leveraging insights from recent research. Initially, *ATT&CK-based analytics* defined in *Finding Cyber Threats with ATT&CK-Based Analytics* [12] and *TTP-Based Hunting* are reviewed. Furthermore, it examines the application of the ATT&CK framework in detection and analytics, detailing the multi-tiered strategy proposed by [42]. The discussion then transitions to exploring active defense capabilities, as detailed in the Active Defense Capability Set technical manual [13], which outlines a structured process for identifying and mitigating malicious activities. Furthermore, this research investigates innovative methodologies for detecting adversaries through the analysis of Windows digital artifacts, as proposed in [43].

This synthesis aims to provide a comprehensive understanding of modern detection engineering methodologies and their role in enhancing cyber defense strategies.

ATT&CK-Based Analytics

Researchers at MITRE have developed a framework for describing and implementing behavioral intrusion detection analytics based on the ATT&CK model [12]. This approach has been rigorously tested through ”Cyber Games,” where Red Teams emulate adversaries to validate the effectiveness of the analytics developed by Blue Teams [12].

The *Threat-Based Security Approach* advocated by MITRE employs a behavioral methodology for detecting threats and introduces five guiding principles for devising effective, threat-based detection strategies [12]:

- **Post-Compromise Detection:** Focus on detecting threats that have bypassed initial defenses.
- **Behavioral Focus:** Use behavioral patterns for detection, avoiding reliance on easily outdated signatures.
- **Threat-Based Modeling:** Employ realistic and relevant threat models to steer detection strategies.
- **Iterative Design:** Continuously adapt security measures to address evolving adversarial tactics.

- **Realistic Testing Environments:** Develop and refine detection analytics in environments that simulate actual network conditions.

The ATT&CK-based analytics development process involves seven steps, validated during six Cyber Games [12]:

1. **Identify Behavior:** Prioritize adversary behaviors within the threat model for detection.
2. **Acquire Data:** Determine necessary data to detect these behaviors.
3. **Develop Analytics:** Construct analytics using the collected data to monitor identified behaviors.
4. **Develop Adversary Emulation Plan:** Create a plan based on ATT&CK that emulates identified behaviors.
5. **Emulate Adversary:** Execute the adversary emulation plan.
6. **Investigate Attack:** Formulate analytics based on insights gained from earlier steps.
7. **Evaluate Performance:** Assess the effectiveness of deployed analytics and sensors in detecting the behaviors identified.

The ”Cyber Games” highlighted several key recommendations for enhancing intrusion detection, including the widespread deployment of monitoring sensors, maintaining unified timelines, and generating lists of suspicious hosts and network graphs [12]. Additionally, detailed documentation of all actions taken during an attack investigation was noted as crucial.

TTP-Based Hunting

In [11], MITRE researchers introduce a *TTP-based* threat hunting approach aimed at proactive detection and investigation of malicious activity within a network. Their framework examines threats across three dimensions: time, terrain, and behavior. Each event within cyberspace is defined by a specific behavior occurring at a precise time and location within the cyber terrain, which encompasses a variety of network elements such as machines, processes, subnets, or other network segments.

The core of *TTP-based detection*, as outlined in [11], involves characterizing and searching for the techniques adversaries use to fulfill their objectives. Utilizing the MITRE ATT&CK model, this approach is instrumental in defensive operations, helping to identify new adversary behaviors, prioritize techniques for detection, and reveal gaps in visibility and defensive capabilities when paired with the Cyber Analytics Repository (CAR) data model [44].

The threat hunting methodology presented in [11] comprises two main components: *Characterization* of malicious activity and *Execution* of the hunt. For the purposes of this research, the emphasis is on the characterization component, where the focus is on collecting information that can be transformed into TTP-based analytics rather than relying solely on brittle indicators of compromise (IoCs). The authors advocate for the development of *hypotheses* and *abstract analytics* based on a deep understanding of adversarial invariant behavior. Effective hunting requires determining the necessary data specifications to ensure that the gathered data comprehensively captures adversary activity from relevant data sources.

Detection and Analytics with ATT&CK

Utilizing the ATT&CK framework, as described by [42], provides structured guidance for developing analytics that identify adversarial behavior. This method involves the systematic collection and analysis of log and event data to detect suspicious or malicious activities detailed within the ATT&CK framework.

At its core, ATT&CK-based detection features three progressive levels of sophistication [42]:

Level 1 focuses on understanding available data and search capabilities. Within the ATT&CK framework, each technique is associated with specific data sources. The initial goal is to identify relevant data sources, deploy sensors on monitored hosts to gather data, and ingest this data into an analysis platform. Preliminary analysis begins with running existing analytics on this data to identify initial detections and address any false positives.

Level 2 encourages analysts to create custom analytics tailored to specific characteristics of targeted attacks. Although this step is complex, as noted by [42], it deepens the understanding of attack methodologies and how they manifest within the data.

Level 3 emphasizes the emulation of real-world attacks to strengthen defensive strategies. This level involves purple teaming exercises, which consist of adversary emulation and subsequent evaluation of detection strategies. The purpose of purple teaming is to validate the effectiveness of the analytics and improve the overall defensive posture.

The ultimate goal at Level 3 is to build confidence in the analytics' effectiveness and enhance defense mechanisms against evolving threats.

Active Defense Capability Set

The *Active Defense Capability Set* technical manual provides a structured 7-step process designed to empower cyber hunt teams in identifying malicious activities. This process builds upon and extends the strategies proposed in [11], offering a comprehensive toolkit for proactive cyber defense.

Before initiating the cyber threat hunting process, it is essential to deploy and familiarize oneself with a robust set of tools [13]. The detailed steps of the 7-step process are as follows [13]:

1. **Develop Malicious Activity Model:** Construct models based on known or anticipated malicious behaviors.
2. **Develop Hypotheses and Abstract Analytics:** Formulate hypotheses about potential threats and create analytics to test these hypotheses.
3. **Determine Data Requirements:** Identify the types and sources of data necessary for the analytics.
4. **Filter the Sources of Data:** Select and prioritize data sources based on their relevance and reliability.
5. **Identify and Mitigate Data Collection Gaps:** Recognize deficiencies in data collection and implement measures to address these gaps.
6. **Implement and Test Analytics:** Apply the developed analytics and rigorously test their effectiveness.
7. **Detect Malicious Activity and Investigate:** Use the analytics to detect activities, and conduct thorough investigations following detections.

This 7-step process enhances cyber threat hunting capabilities, enabling teams to proactively address and mitigate cyber threats.

2.2 Lateral Movement Detection

The landscape of lateral movement detection has been a focal point for cybersecurity researchers and practitioners. Numerous efforts have been dedicated to devising effective methods to identify and mitigate the risks associated with lateral movement.

Detecting Adversary Using Windows Artifacts

The study *Detecting Adversaries using Windows Digital Artifacts* [43] investigates the potential of utilizing Windows digital artifacts to identify adversaries, especially in environments with limited logging capabilities.

The research focuses on scenarios where system observability is hindered—either by attackers actively hiding their tracks or due to failures in log collection mechanisms. By analyzing the dynamic behavior exhibited within a host and examining the characteristics of digital artifacts, the authors propose a prototype detection system capable of identifying adversarial activities.

A significant aspect of the study is the establishment of a chronological sequence of executed tasks on a host. This is achieved through the analysis of various artifacts, such as the Master File Table (MFT), Windows Prefetch files, Application Compatibility Cache (*ShimCache*), and Windows Event Log.

Moreover, the paper introduces innovative contributions like an algorithm for interval estimation of ShimCache events and a novel detection system named *XTEC*, which utilizes machine learning techniques. While these developments are not directly relevant to this thesis, they exemplify cutting-edge methods in adversary detection.

Detection Strategies Utilizing Log Data

Numerous studies have advocated for the effectiveness of log analysis in detecting lateral movements and other malicious activities. These include *Detecting Lateral Movement in Windows Infrastructure* [45], *Advanced Persistent Threats: Behind the Scenes* [1], and *Detecting Lateral Movement Through Tracking Event Logs* [6]. These publications highlight the critical role of implementing an appropriate logging policy to capture essential information effectively. They also provide a detailed examination of the usage and manipulation of built-in administrative tools and offensive security tools, which are frequently exploited by attackers.

Graph-Based Detection Systems

Graph theory has found novel applications in the realm of cybersecurity, particularly in detecting lateral movements within networks. For example, *the Hopper model* proposed in [36] constructs a graph based on login activity across internal machines, utilizing commonly available log data. This model effectively visualizes and tracks the flow of activity to identify unusual patterns that may indicate unauthorized movements.

Similarly, *the Latte system* presented in [37] employs a graph-based detection framework. In this system, computers and user accounts are represented as nodes, while logon activities form the edges connecting these nodes. This graphical representation allows for a more intuitive analysis of interactions and movements across the network, enhancing the ability to spot potential security breaches based on abnormal connectivity patterns.

These graph-based approaches provide a powerful tool for cybersecurity teams, offering enhanced visibility and a deeper understanding of network dynamics that traditional methods might overlook.

Real-Time Detection Systems

As discussed in *Real-Time Lateral Movement Detection Based on Evidence Reasoning Network for Edge Computing Environment* [46], real-time detection systems for lateral movement have become increasingly sophisticated, incorporating evidence reasoning networks to enhance their responsiveness. These systems are specifically designed to analyze network behavior as it occurs, allowing for the immediate identification and mitigation of lateral movement threats. The ability to react instantaneously provides a significant advantage in limiting the spread of an intrusion within a network.

Distributed Data Fusion

Distributed data fusion, proposed in *Lateral Movement Detection Using Distributed Data Fusion* [8], represents a progressive approach to improving the detection of lateral movements. By aggregating and integrating data from multiple sources, this technique aims to create a more comprehensive and accurate detection system. Such fusion enhances the detection capabilities by providing a broader perspective, which is critical in identifying complex patterns indicative of lateral movement.

Chapter 3

Adversarial Behavior Detection

This chapter delves into the foundational concepts of detecting adversarial behavior, exploring the specific features of prevalent detection approaches. It provides an analysis of various detection schemas, highlighting their strengths and weaknesses. The objective is to furnish a comprehensive overview of contemporary detection methods utilized in modern defense systems. Additionally, this chapter evaluates two open-source rule sets, Elastic Prebuilt Detection Rules [10] and Sigma [9], focusing on their coverage of the TA0008 Lateral Movement tactic. This evaluation aims to shed light on the current capabilities and limitations of these tools in detecting sophisticated cyber threats.

3.1 Detection Approaches

Detecting adversarial behavior presents a significant challenge within cybersecurity, necessitating a deep understanding of a wide range of detection methodologies to effectively address the growing spectrum of cyber threats. Malware is typically detected through its *signatures* or *behavior*, encompassing three main approaches: *signature-based*, *behavior-based*, and *heuristic-based* detection [47]. Figure 3.1 illustrates these approaches and their relationships with various digital artifacts, highlighting the unique strengths and challenges of each method. This section explores the diverse landscape of detection techniques used to protect digital environments, aiming to provide a comprehensive overview of these methods and enhance the reader's theoretical knowledge necessary for understanding complex cyber defense mechanisms.

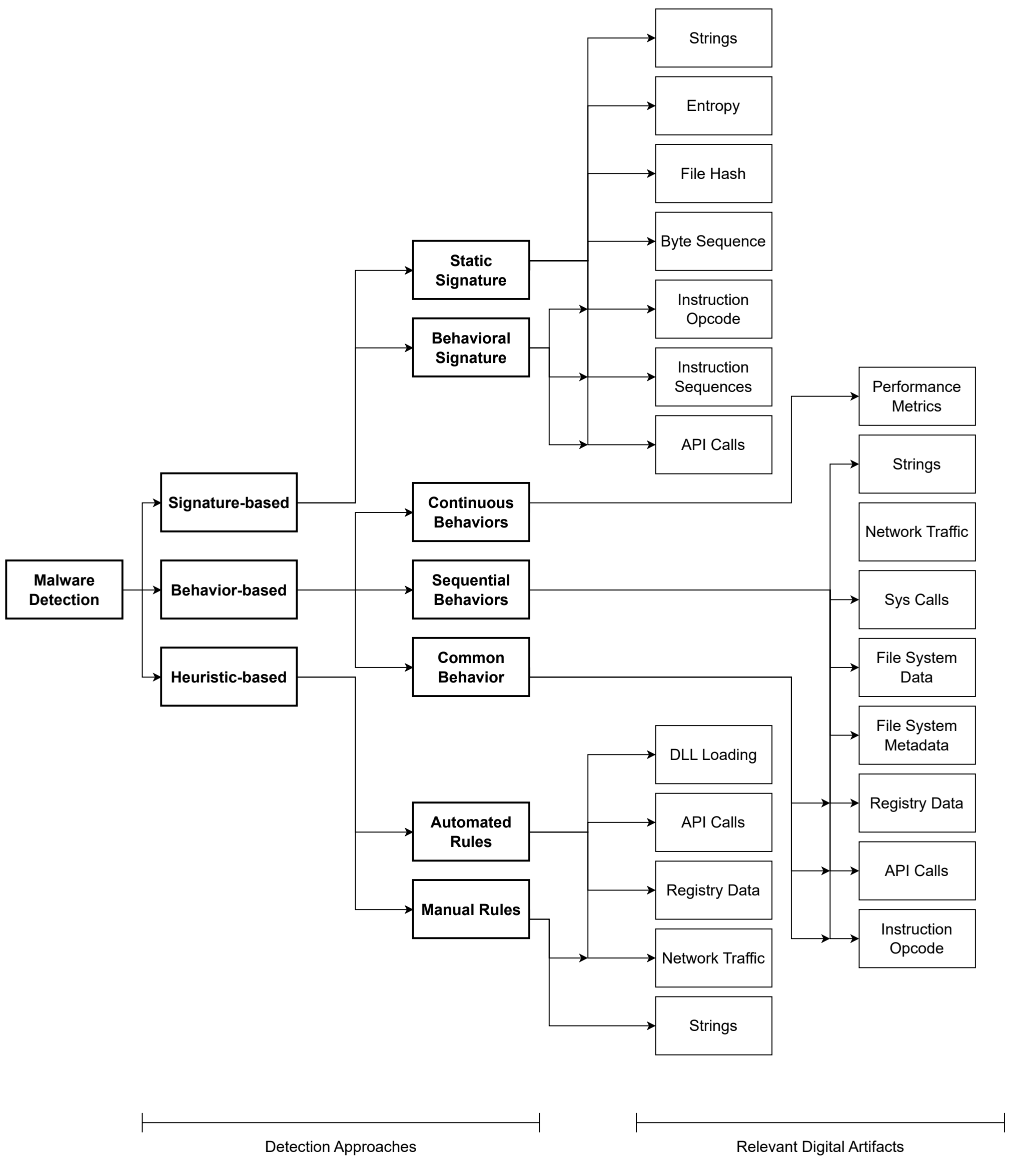


Figure 3.1 Malware detection taxonomy.

3.1.1 Signature-Based Detection

A signature in cybersecurity is a unique set of characteristics or a pattern that identifies a specific piece of malware, encapsulating its structural essence and serving as a distinct identifier. This detection approach is favored in commercial antivirus software for its speed and efficiency in recognizing known threats [47]. However, it struggles with unknown malware due to its reliance on pre-defined signatures. Furthermore, variants within the same malware family can often evade detection through obfuscation techniques, which modify their signatures without altering their malicious functionalities. Figure 3.2 provides a visual representation of this detection approach.

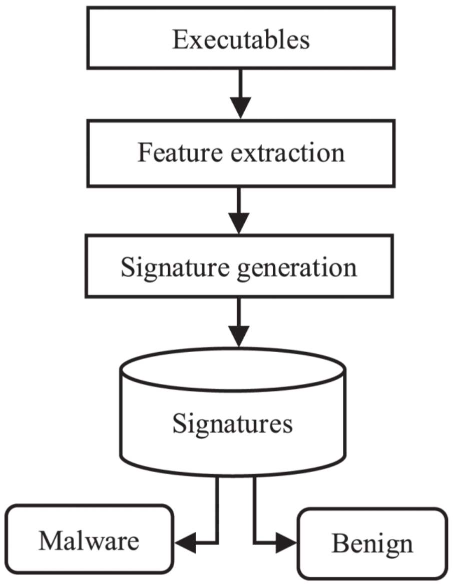


Figure 3.2 Signature-based detection schema [47].

Signature-based detection models operate by comparing the extracted signature of a file, obtained either statically or dynamically, with a database of known signatures to ascertain potential threats. Static extraction methods might analyze features such as API calls, instruction opcodes, byte-code series, byte sequences, and file hashes [48]. Dynamic extraction, on the other hand, assesses behavioral changes observed during the file's execution, including instruction sequences, network traffic patterns, and API call sequences [48]. While this method is highly effective against cataloged malware, its efficacy diminishes against new variants or extensively modified threats, underscoring the need for continuous updates to signature databases.

3.1.2 Behavior-Based Detection

Behavior-based detection focuses on monitoring executable files within an isolated environment to observe their runtime behaviors. This method evaluates the maliciousness of a file based on the behavioral patterns it exhibits during execution, effectively detecting malware regardless of code modifications that could mask adversarial actions. Figure 3.3 illustrates the layout of this detection approach, highlighting its capability to identify both known malware and novel variants.

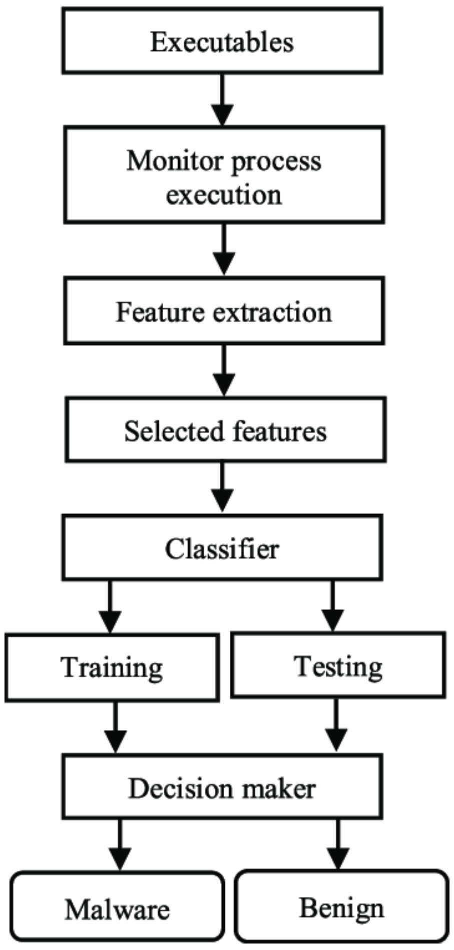


Figure 3.3 Behavior-based detection schema [47].

Behavior-based detection is categorized into three types, each targeting different aspects of behavioral analysis [48]:

- **Continuous behaviors:** Assessed through system performance metrics such as CPU usage, memory consumption, and network traffic. These metrics provide a continuous real-time overview of an executable's system impact, offering insights into potentially malicious activity.
- **Sequential behaviors:** Identified by analyzing patterns or sequences formed from API calls, system calls, and opcodes. This approach focuses on the order and context of operations executed by the software, which is crucial for distinguishing between benign and malicious processes.
- **Common behaviors:** Involves recognizing actions shared by both malware and benign applications. This identification helps classify the nature of a file, determining if its behavior aligns more with typical malware or harmless software.

Each category leverages different facets of executable behavior to increase the accuracy and effectiveness of malware detection, addressing the dynamic challenges posed by evolving cyber threats.

3.1.3 Heuristic-Based Detection

Heuristic-based detection combines experiential knowledge with various techniques, such as predefined rules and machine learning algorithms, to identify potential threats [49]. This method is particularly effective against zero-day malware, playing a critical role in adapting to the rapidly evolving cyber threat landscape [47]. By employing a wide array of heuristics, this detection approach can detect unusual or suspicious behaviors that lack known signatures, filling a crucial gap in traditional detection methods. Figure 3.4 provides a schematic representation of the heuristic-based detection approach, highlighting how these diverse elements collaborate to enhance the detection capabilities.

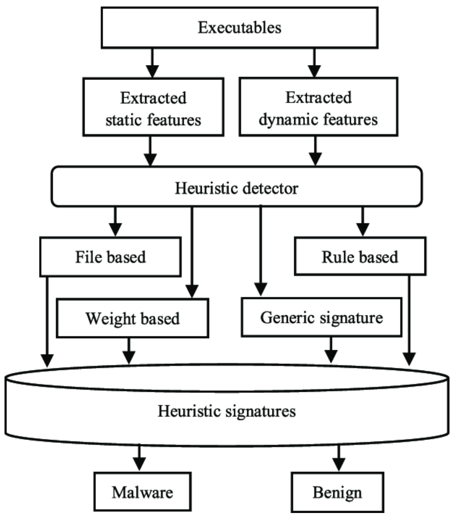
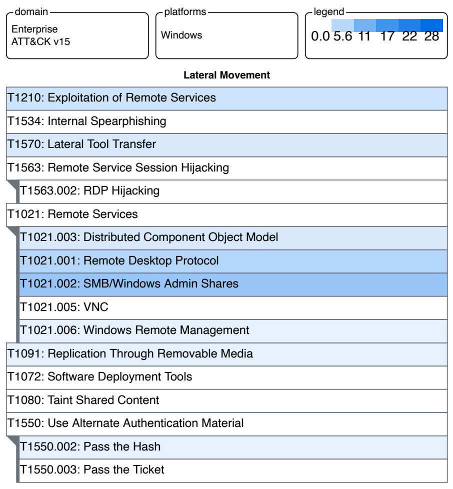


Figure 3.4 Heuristic-based detection schema.

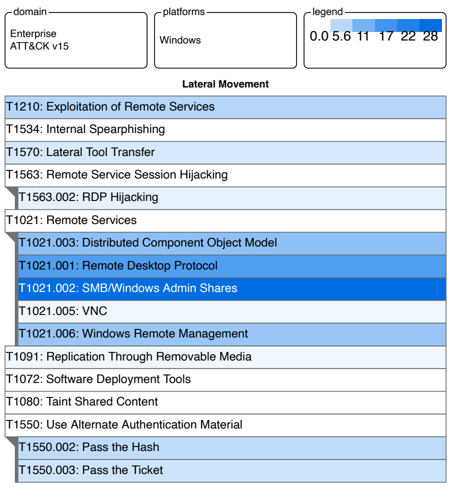
3.2 Lateral Movement Detection Coverage

The effectiveness of cybersecurity defenses is heavily influenced by the breadth and depth of detection coverage against an evolving threat landscape. Detection coverage, defined as the capability of a system or dataset to identify and alert on malicious activities, is a critical metric for evaluating the robustness of an organization's security posture [42]. This section evaluates key rule sets, such as the Elastic Detection Rules [50] and Sigma Rules [9], examining their contributions to improving lateral movement detection coverage. Additionally, this research considers the trade-offs, particularly the balance between achieving comprehensive coverage and minimizing the risk of false positives.

However, the development, verification, and validation of these rules pose significant challenges. A meticulous approach is essential to ensure that the rules maintain their effectiveness against evolving tactics and sophisticated cyber threats [13]. Each stage in the lifecycle of these rules, from initial development through to rigorous validation, requires careful attention to detail and extensive expertise in cybersecurity analysis and threat intelligence.



(a) Elastic Detection Rules lateral movement coverage.



(b) Sigma Rules lateral movement coverage.

Figure 3.5 TA0008 Lateral Movement detection coverage by Elastic and Sigma detection rulesets.

The development of the underlying analytics is a sophisticated and intensive process. It requires the collection, analysis, and synthesis of extensive volumes of threat data, behavioral patterns, and attack vectors [11, 12, 42]. The effort aims to uncover significant correlations and indicators of compromise. The creation of these analytical models necessitates expertise in data science, machine learning, and cybersecurity research to ensure they accurately reflect the dynamic nature of cyber threats. Despite the challenges this process presents, the resulting analytics typically deliver high-quality detection capabilities.

An illustrative example of a detection rule that follows the heuristic-based schema is *Incoming DCOM Lateral Movement with ShellBrowserWindow or ShellWindows* [51]. The core Event Query Language (EQL) query for this rule is shown in Listing 3.1. This rule specifically targets a sequence of events and examines certain features of these events to implement a heuristic-based detection approach, as depicted in Figure 3.4.

```
sequence by host.id with maxspan=5s
[
        network where host.os.type == "windows" and event.type == "start" and
         ↪  process.name : "explorer.exe" and
 network.direction : ("incoming", "ingress") and network.transport == "tcp" and
 source.port > 49151 and destination.port > 49151 and source.ip != "127.0.0.1"
  ↪  and source.ip != "::1"
] by process.entity_id

[
        process where host.os.type == "windows" and event.type == "start" and
 process.parent.name : "explorer.exe"
] by process.parent.entity_id
```

Listing 3.1 *Incoming DCOM Lateral Movement with ShellBrowserWindow or ShellWindows* detection rule EQL query [51].

Many of the rules within the Sigma rule set employ signatures to perform detection. Authors of the rules collect data, evaluate them, and extract signatures they think that may lead to a successful detection of given adversarial behavior. Rules from the Sigma rule set often include a variety of signatures that focus on specific file or process characteristics. Due to their specificity, rules using signatures generally exhibit a lower probability of generating false positives.

Examples of signature-based Sigma rules are demonstrated by the `win_alert_mimikatz_keywords` rule [52], displayed in Listing 3.2, and the `proc_creation_win_mmc_mmc20_lateral_movement` rule [53], shown in Listing 3.3. Both rules utilize known indicators of adversarial activities to trigger alerts.

The `win_alert_mimikatz_keywords` rule implements detection based on *keywords* [52], one of the simplest methods for identifying adversarial behavior [54]. The keyword-based detection searches log sources for a text that matches any of the specified keywords, effectively using a logical OR operator between each keyword [54]. The keywords list functions analogously to a file signature. The *signatures* in the context of detection rule 3.2, represented as keywords, are extracted from PowerShell commands that typically indicate the use of Mimikatz. This detection approach is akin to the signature-based detection schema illustrated in Figure 3.2.

```
...
logsource:
    product: windows
detection:
    keywords:
        - 'dpapi::masterkey'
        - 'eo.oe.kiwi'
        - 'event::clear'
        - 'event::drop'
        - 'gentilkiwi.com'
        - 'kerberos::golden'
        - 'kerberos::ptc'
        - 'kerberos::ptt'
        - 'kerberos::tgt'
        - 'Kiwi Legit Printer'
        - 'lsadump::'
        - 'mimidrv.sys'
        - '\mimilib.dll'
        - 'misc::printnightmare'
        - 'misc::shadowcopies'
        - 'misc::skeleton'
        - 'privilege::backup'
        - 'privilege::debug'
        - 'privilege::driver'
        - 'sekurlsa::'
    filter:
        EventID: 15  # Sysmon's FileStream Events (could cause false positives
        ↪  when Sigma rules get copied on/to a system)
    condition: keywords and not filter
falsepositives:
    - Naughty administrators
    - AV Signature updates
    - Files with Mimikatz in their filename
...
```

Listing 3.2 Sigma detection rule relying on specific keywords to detect Mimikatz [52].

Detection rule `proc_creation_win_mmc_mmc20_lateral_movement` implements detection by *Field* [55]. Rules implementing detection by field perform *field-value* searches [54]. Such rules generate an alert when a defined field-value pair is found in given log source. The `proc_creation_win_mmc_mmc20_lateral_movement` rule defines three field-value pairs which form a detection condition using AND logical operator. The fields defined in `proc_creation_win_mmc_mmc20_lateral_movement` are `ParentImage`, `Image` and `CommandLine` [55]. Each field in rule 3.3 use *Modifiers*, modifying the value of a field. Modifiers are discussed with greater detail in Section 5.5.1.

Lateral movement may be performed by abusing Distributed Component Object Model (DCOM), which is described in sub-technique T1021.003, Distributed Component Object Model [56]. One of procedures related to this technique is abusing the Microsoft Management Console (MMC) to spawn arbitrary processes through DCOM [56]. This behavior is captured in `proc_creation_win_mmc_mmc20_lateral_movement` detection rule by detecting typical MMC abuse process tree and `-Embedding` command line argument. Similarly to the previous detection rule 3.2, this rule utilizes features of a process creation event, serving as a *signature*. Detection approach employed by this rule is an analogy of signature-based detection schema, which is illustrated in 3.2.

```
...
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        ParentImage|endswith: '\svchost.exe'
        Image|endswith: '\mmc.exe'
        CommandLine|contains: '-Embedding'
    condition: selection
falsepositives:
    - Unlikely
level: high
```

Listing 3.3 Sigma detection rule utilizing process creation event data patterns related with MMC20 lateral movement [55]

Despite their initial efficacy, rules based on specific file or process characteristics may become less effective over time as adversaries adapt and modify their tactics to evade detection. This adaptability poses a significant challenge in maintaining the relevance and effectiveness of these rules in combating evolving cyber threats.

In contrast, other Sigma rules are constructed based on the typical behaviors exhibited by commonly used hacking tools and extensively analyzed malware specimens. These rules aim to detect malicious activities by identifying patterns and behaviors associated with known attack vectors and threat actor techniques. An example of this rule is `proc_creation_win_susp_copy_lateral_movement` [53]. This rule leverages the knowledge of common behavior patterns exhibited by adversarial lateral movement. However, the procedures that adversaries implement to achieve lateral movement may be confused with legitimate activities within the environment [2]. Without a baseline behavior within the environment it is very difficult to detect lateral movement without false positives. These rules often exhibit a broader scope, which may inadvertently flag legitimate software or behaviors, leading to false positives. This broader detection scope poses challenges for security solutions, as it may reduce the overall efficacy of threat detection and increase the workload for security analysts in distinguishing genuine threats from false alarms.

```
...
logsource:
    category: process_creation
    product: windows
detection:
    selection_target:
        CommandLine|contains:
            - '\\\\*$'
            - '\Sysvol\'
    selection_other_tools:
        - Image|endswith:
              - '\robocopy.exe'
              - '\xcopy.exe'
        - OriginalFileName:
              - 'robocopy.exe'
              - 'XCOPY.EXE'
    selection_cmd_img:
        - Image|endswith: '\cmd.exe'
        - OriginalFileName: 'Cmd.Exe'
    selection_cmd_cli:
        CommandLine|contains: 'copy'
    selection_pwsh_img:
        - Image|contains:
              - '\powershell.exe'
              - '\pwsh.exe'
        - OriginalFileName:
              - 'PowerShell.EXE'
              - 'pwsh.dll'
    selection_pwsh_cli:
        CommandLine|contains:
            - 'copy-item'
            - 'copy '
            - 'cpi '
            - ' cp '
            - 'move '
            - 'move-item'
            - ' mi '
            - ' mv '
    condition: selection_target and (selection_other_tools or all of
     ↪  selection_cmd_* or all of selection_pwsh_*)
falsepositives:
    - Administrative scripts
level: medium
```

Listing 3.4 Sigma detection rule `proc_creation_win_susp_copy_lateral_movement` [53].

The `proc_creation_win_susp_copy_lateral_movement` utilize commonly observed patterns related to technique T1021.002, Remote Services: SMB/Windows Admin Shares [57]. The detection approach of this rule follow behavior-based detection approach, summarized in Figure 3.3. By observing the execution of procedures implementing technique T1021.002, Remote Services: SMB/Windows Admin Shares, several features are collected. Combining them allows the detection rule `proc_creation_win_susp_copy_lateral_movement`, presented in 3.4, to detect behavior exhibited by T1021.002. However, this rule may incorrectly generate an alert when administrative scripts are run within the environment.

Chapter 4

Testing Environment

This chapter outlines the construction of a testing environment specifically designed to facilitate the development of cyber threat analytics and detection methodologies. The primary focus is on analyzing lateral movement techniques within a controlled setting that simulates a real-world network.

Given the prevalence of Windows operating systems in commercial and personal computing, the testing environment primarily utilizes Windows virtual machines. This choice is informed by the latest operating system market share statistics, which highlight Windows as the most widely used desktop operating system as of February 2024 [58], as depicted in Figure 4.1.

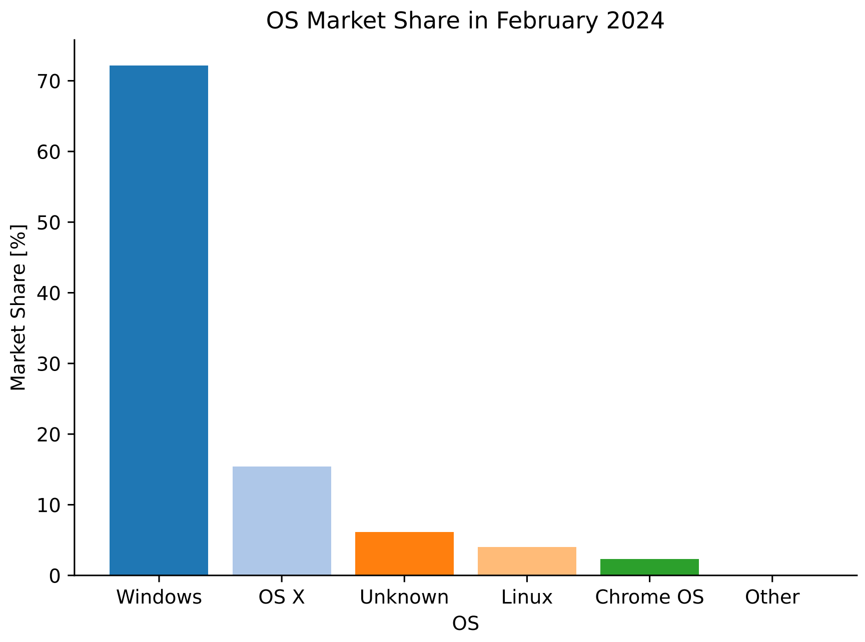


Figure 4.1 Operating system market share in February 2024, illustrating the dominance of Windows in the global market.

The subsequent sections detail the components of the testing environment, including the network architecture, security configurations, and the monitoring systems deployed to capture data for analysis. This environment is specifically structured to expose and analyze vulnerabilities associated with lateral movement techniques, thereby providing a robust platform for testing and refining detection rules.

4.1 Building the Lab

To ensure optimal performance and a smooth simulation experience, the lab environment should be established on a host machine that meets or exceeds the following specifications:

- **CPU**: At least 4 cores to efficiently manage multiple virtual machines.
- **RAM**: A minimum of 16GB to allow adequate resources for the host and guest operating systems.
- **Storage**: At least 256GB of available space to accommodate virtual machine files, snapshots, and logs.

The following software tools are essential for setting up and managing the virtual lab environment:

- **VMWare Workstation 16 or later**[1]: This virtualization software provides robust features for running multiple isolated operating systems on a single physical machine. It is known for its performance and reliability in a testing environment.
- **Packer by HashiCorp**[2]: Packer assists in building automated machine images. It is used here to create reproducible and consistent environments for testing.
- **Vagrant by HashiCorp**[3]: Vagrant is employed to manage and provision virtual machines through a simple and easy-to-use command-line interface. It works in conjunction with VMWare to streamline the development and testing of virtual environments.

For operating systems, Windows 10 Enterprise is chosen for its advanced security features suitable for testing, alongside Windows Server 2022, which provides a robust platform for simulating enterprise-level network scenarios.

This foundational setup is crucial for facilitating detailed cybersecurity research and development activities, ensuring that the environment closely mirrors a realistic network configuration. The choice of tools and operating systems maximizes compatibility and functionality, providing a stable and flexible testing ground for exploring lateral movement techniques and other cybersecurity threats.

4.1.1 Network Setup

The network configuration for the lab is critical to ensure controlled communication between the virtual machines and the host, while restricting external access. For this purpose, a ”Host-only” network is utilized [59]. This network configuration allows all virtual machines within the lab to interact with each other and the host machine, yet prevents any external network access, enhancing the security and isolation of the test environment.

If internet connectivity is required for updates or additional configurations, it can be facilitated through the addition of a second network adapter configured to connect to a Network Address Translating (NAT) network [60]. This setup allows controlled internet access without compromising the isolation of the primary host-only network.

[1] https://www.vmware.com/products/workstation-pro.html
[2] https://www.packer.io/
[3] https://www.vagrantup.com/

Network Adapter	Host-only
Internet Access	No (isolated environment)
Subnet	192.168.56.0 (example)
Subnet Mask	`255.255.255.0`
DHCP	Enabled

Table 4.1 Testing environment network configuration.

This table outlines the general settings applied to the network adapters in the lab environment. The DHCP-enabled host-only network ensures that all machines are automatically assigned IP addresses within the specified subnet, simplifying network management and device communication within the lab. The dual-network adapter setup allows flexibility and control, aligning with specific testing needs and security requirements.

4.1.2 Base Boxes Build

Base boxes serve as foundational templates for virtual machines in the testing environment, ensuring consistency and repeatability in deployments. The construction of these base boxes is automated using Packer by HashiCorp, a tool designed to create identical machine images for multiple platforms from a single source configuration.

The process involves the automatic installation and provisioning of operating systems, utilizing *Answer Files* [61] to pre-configure settings during the OS setup phase. These XML files are used to automate Windows installations by specifying various system settings including disk configurations, administrative credentials, and network settings [61, 62].

- **Disk Configuration:** Sets up partitions and storage settings.
- **Administrator Account:** Configures the default administrator account with predefined credentials.
- **Remote Access:** Enables and configures remote desktop protocols to allow remote connections.

During the base box provisioning phase, several crucial adjustments are made:

- **Network Classification:** Networks are set to ”Private” to reduce complications with firewall rules that may block communication.
- **Remote Desktop:** Remote desktop access is enabled, allowing remote management and testing.
- **Virtualization Tools:** Necessary helper tools from the virtualization software are installed to enhance performance and compatibility.
- **Network Prompts:** Automated responses to network prompts are configured to prevent interruptions in the automation process.

This meticulous setup results in the creation of two primary base boxes: one configured for Windows Server 2022 and another for Windows 10 Enterprise. These base boxes streamline the deployment of new test environments, significantly reducing the time and effort required to initiate testing scenarios. The reproducibility offered by these pre-configured templates ensures that the testing environment can be rapidly replicated or scaled as needed.

4.1.3 Virtual Machines Deployment

Virtual machines deployment process utilizes Vagrant by HashiCorp to automate the instantiation of virtual hosts from the base boxes. Vagrant simplifies the process of virtual machine management and provisioning, allowing for the efficient setup of a controlled testing environment.

Upon initiating Vagrant, the tool automatically creates virtual machines based on predefined configurations in the Vagrantfile. This file specifies the virtual hardware settings, networking configurations, and base box images to use. Once the virtual machines are instantiated, Vagrant remotely connects to each machine to perform the following provisioning tasks:

1. **Software Installation:** Essential software tools and utilities used for monitoring and managing the testing environment are downloaded and installed. This includes security monitoring tools like Sysmon, which provides detailed information about process creations, network connections, and file changes.

2. **Network Configuration:** Network adapters are configured to ensure proper communication within the lab environment, and the hosts file is updated to facilitate name resolution among the virtual machines.

3. **Active Directory Setup:** For environments requiring domain management, Vagrant automates the installation and configuration of Active Directory Domain Services. A new Active Directory forest is created, or the virtual machines are joined to an existing domain, establishing a controlled domain environment for testing.

4. **Policy and Security Settings:** Group Policy Objects (GPOs) are defined and applied to enforce specific security policies and settings across the domain, enhancing the security posture of the testing environment.

The deployment process concludes with the virtual machines fully configured and integrated into the lab's network, ready for subsequent phases of testing and analysis. The use of Vagrant ensures that each component of the testing environment can be reliably reproduced and efficiently managed, providing a scalable and flexible platform for cybersecurity research.

4.2 Monitoring

Effective monitoring is pivotal for detecting and understanding adversarial behavior within the network. This section details the deployment and configuration of specialized monitoring tools designed to capture a comprehensive spectrum of system and network activities.

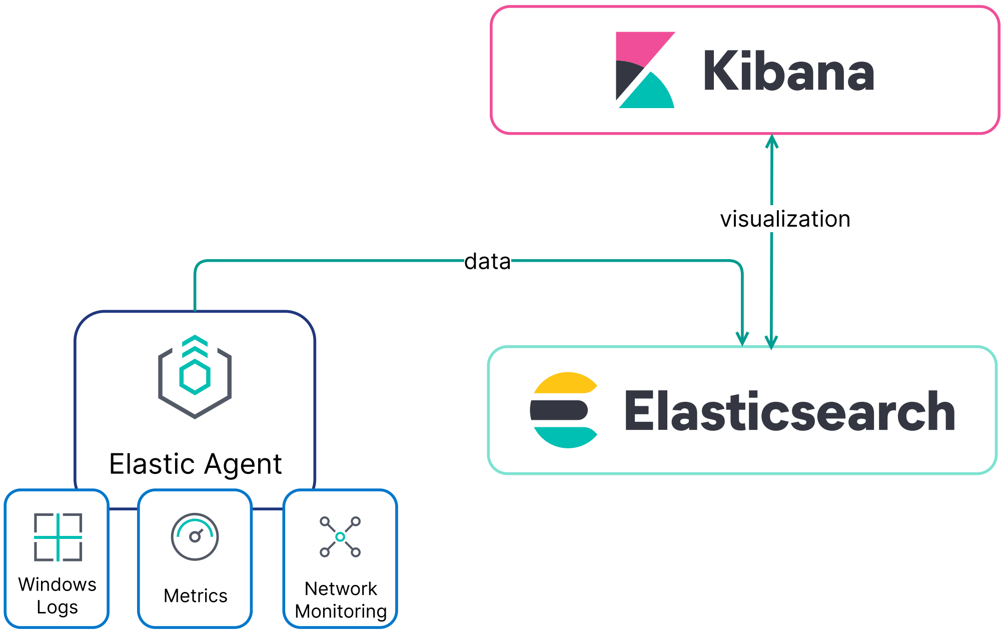


Figure 4.2 Overview of the monitoring setup in the testing environment.

Figure 4.2 provides a visual overview of the entire monitoring setup, illustrating how each component of the system contributes to the overarching goal of capturing and analyzing adversarial behavior effectively.

By deploying these sophisticated host-based monitoring tools and configurations, the testing environment is equipped to capture detailed logs that are essential for analyzing and understanding the tactics, techniques, and procedures (TTPs) employed by adversaries. This comprehensive monitoring setup not only supports the detection of malicious activities but also enhances the research's ability to develop effective cybersecurity strategies and defenses.

Furthermore, comprehensive network monitoring and host activity logging are facilitated through the deployment of *Elastic Agent* [63]. This agent plays a pivotal role by ingesting logs from the monitored systems and standardizing their format into the *Elastic Common Schema* (ECS) [63, 64]. The use of the Elastic Common Schema ensures consistency across data types, making it easier to analyze and correlate data from various sources. This uniform data structure allows for more efficient data processing and integration into *Kibana*, where advanced data analysis and visualization tools can be utilized to detect anomalies and potential security threats more effectively.

System Monitoring with Sysmon

Sysmon (System Monitor) is a critical component of the monitoring framework within the lab environment. As a Windows system service and device driver, Sysmon provides detailed logging of system activity to the Windows event log, making it an invaluable tool for security analysis [65].

Sysmon is specifically tailored to record intricate details about process creations, network connections, file creation timestamps, and much more. This level of detail includes the hashing of process images, comprehensive network information associated with each process, and the context of process execution, such as user-initiated actions.

The robustness of Sysmon lies in its ability to operate as a *protected process*, which significantly limits the ability of user-mode processes to interact with it [65]. This protection ensures that even if an adversary has compromised the system, altering or disabling Sysmon without elevated privileges is highly challenging.

Sysmon Configuration for In-depth Monitoring

For this research environment, Sysmon is configured to capture data exhaustively using a configuration file developed from Olaf Hartong's `sysmon-modular` project. This project provides a flexible framework for Sysmon configurations, allowing for detailed logging tailored to specific research needs [66].

The chosen configuration, `sysmonconfig-research.xml`, is set to an extremely verbose logging level to ensure that all pertinent events are captured. This configuration is specifically designed for environments where detailed analysis and comprehensive data capture are required, such as in a testing or research setting where understanding the full scope of adversarial actions is crucial [66].

However, it's important to note that the verbose logging by Sysmon, especially with the `sysmonconfig-research.xml` file, can lead to substantial consumption of system resources. Thus, it is recommended that this level of detailed monitoring be used judiciously, balancing between the need for comprehensive data and the overall performance of the system.

Elastic Agent Integrations

The *System* integration for Elastic Agent is strategically deployed to aggregate comprehensive system logs and metrics. This integration collects a wide array of logs, including application, system, and security events from each monitored host [67]. Additionally, it captures vital system metrics such as CPU usage, load statistics, memory usage, and detailed information on network behavior [67]. These data points are crucial for a thorough analysis of the system's health and security posture, enabling proactive identification and mitigation of potential threats.

The *Windows* integration enables the Elastic Agent to comprehensively collect data specific to Windows operating systems, services, and applications [68]. This integration is designed to capture detailed metrics, including service details and performance counters. These metrics provide insights into the operational health and performance of Windows services and applications, such as service start times, stop times, and the status of various performance counters that monitor CPU, memory usage, and disk activity. The data collected is vital for monitoring system performance and ensuring the stability and security of Windows environments.

Network Monitoring Overview

The *Network Packet Capture* Elastic Agent integration is essential for capturing network traffic and enabling thorough analysis of communications within the network. This integration supports several key protocols, which are crucial for understanding network interactions [69]:

- `ICMP` (v4 and v6), for diagnostic and error messages within the network
- `DHCP` (v4), for monitoring dynamic IP address assignments
- `DNS`, for observing domain name resolution traffic
- `TLS`, for tracking encrypted communications
- `NFS`, for analyzing file system access over the network

Additionally, this integration captures *network flows*, which provide a contextual overview of network connections on a host. A network flow represents a series of packets transmitted during a given time period that share common properties such as source and destination addresses and the protocol used [69].

The *Elastic Defend* integration for Elastic Agent, equipped with *Prebuilt Security Detection Rules*, plays a crucial role in the analysis of security events [70, 71]. This powerful set of tools is designed to automatically identify and alert on potential security threats by analyzing the patterns and anomalies in the logged data. The prebuilt rules are based on known attack patterns and behaviors, which significantly accelerates the detection process, allowing security teams to respond swiftly to mitigate risks [10]. This integration enhances the proactive security measures by leveraging advanced algorithms to sift through vast amounts of data for signs of compromise.

Chapter 5

Proposed Analytics and Detection Methods

To develop robust and reliable analytics for detecting adversarial behavior, several established processes have been adopted and adjusted. These include methodologies from *TTP-based Threat Hunting* [11], *Finding Cyber Threats with ATT&CK-Based Analytics* [12], and *Active Defense Capability Set* [13]. Each of these sources provides a framework for systematically identifying and responding to cyber threats by focusing on tactics, techniques, and procedures (TTPs) commonly used by adversaries. This approach ensures a comprehensive and proactive defense strategy.

The cyber threat analysis in this research adheres to the five principles of a threat-based security approach as defined in *ATT&CK-Based Analytics* by Strom et al. [12]. These principles are:

1. Include post-compromise detection
2. Focus on behavior
3. Use a threat-based model
4. Iterate by design
5. Develop and test in a realistic environment

To implement Principle 1, the analysis specifically targets lateral movement, which is recognized as a post-compromise tactic, ensuring that detection mechanisms are in place even after an initial breach has occurred. In accordance with Principle 2, the analytics and detection methodologies developed in this work eschew traditional signature-based approaches, instead focusing on anomalous behavior patterns that indicate malicious activity. This behavior-based focus aids in catching sophisticated attacks that might otherwise evade signature-based detectors.

For Principle 3, each behavior analyzed is encapsulated within an accurate and well-defined threat model. This model ensures that detection methodologies are finely tuned to identify realistic adversary behaviors, enhancing the overall effectiveness of the security measures. Adhering to Principle 4, the development of analytics and detection methodologies is inherently iterative, allowing for continual refinement and adaptation in response to the evolving cyber threat landscape.

Finally, Principle 5 is fulfilled by testing the detection methodologies in a controlled environment that simulates adversarial behavior. This testing confirms the efficacy of the security measures under realistic conditions, ensuring that they are capable of defending against actual cyber threats.

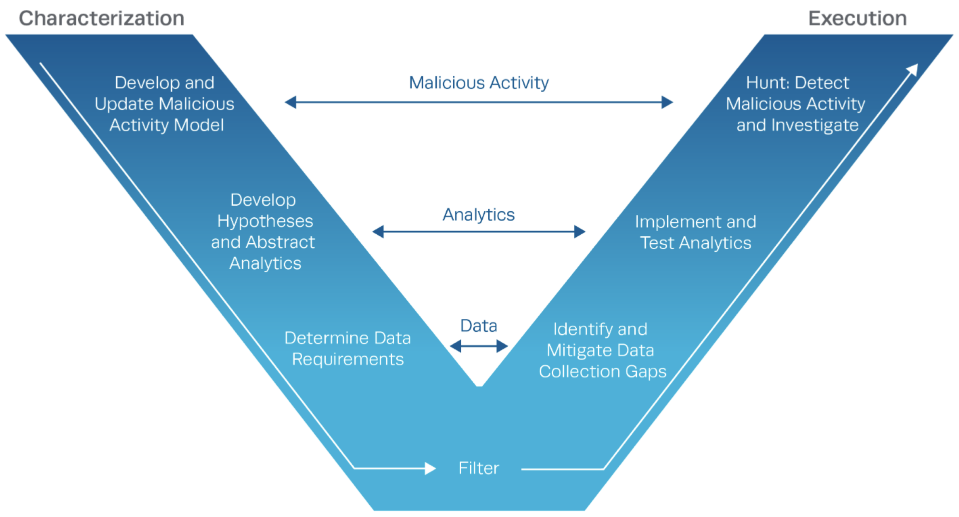

Characterization
Execution
Develop and Update Malicious Activity Model
Malicious Activity
Hunt: Detect Malicious Activity and Investigate
Develop Hypotheses and Abstract Analytics
Analytics
Implement and Test Analytics
Determine Data Requirements
Data
Identify and Mitigate Data Collection Gaps
Filter

Figure 5.1 TTP-based hunting methodology ”V” diagram [11].

With the aforementioned five principles serving as a foundation, the analytics developed in this research employ a slightly modified version of the seven-step ATT&CK-based analytics development method, outlined in *Finding Cyber Threats with ATT&CK-Based Analytics* by Strom et al. [12]. These modifications include complementing incorporating elements from an analytics development method version detailed in the *Active Defense Capability Set* by Gloor [13] and enriching the method with detailed information from [11].

The proposed analytics development method follows steps:

1. **Model Adversarial Behavior** – Identify adversarial behavior and formally describe it.
2. **Acquire Data** – Model data required to detect the behavior.
3. **Develop Hypothesis** – Create a hypothesis based on an understanding of the behavior and its data dependencies.
4. **Develop Adversary Emulation Plan** – Develop a detailed plan to emulate the adversarial behavior realistically.
5. **Emulate Threat** — Execute the emulation plan to simulate the threat environment.
6. **Develop Analytics** – Analyze the evidence collected during emulation to refine detection strategies or propose new ones.
7. **Verify Analytics** – Test the effectiveness of the developed analytics and proposed detection methodologies using the emulation.

These steps represent a rigorous approach to developing and validating threat detection analytics, ensuring that each phase contributes to a robust defense against cyber adversaries.

5.1 Model Adversarial Behavior

The first step to effectively identify and prioritize adversarial behavior, is to develop a malicious activity model through extensive collection and analysis of cyber threat intelligence. During this initial analysis, transient aspects of adversarial behavior are identified, with a particular focus on those elements that are challenging for adversaries to alter. According to the *Pyramid of Pain* [72], these aspects are categorized into classes based on the difficulty of changing each aspect. This research emphasizes those elements at the top of the Pyramid of Pain, which relate directly to TTP-based analytics, thereby targeting the most complex behaviors for adversaries to modify.

However, relying solely on these high-complexity aspects could potentially lead to the development of inefficient detection methodologies. Therefore, a second crucial parameter of adversarial behavior – its occurrence in current cyber threats – is also considered. This approach ensures a balanced focus on both the changeability and prevalence of adversarial tactics, enhancing the effectiveness and relevance of the detection strategies developed.

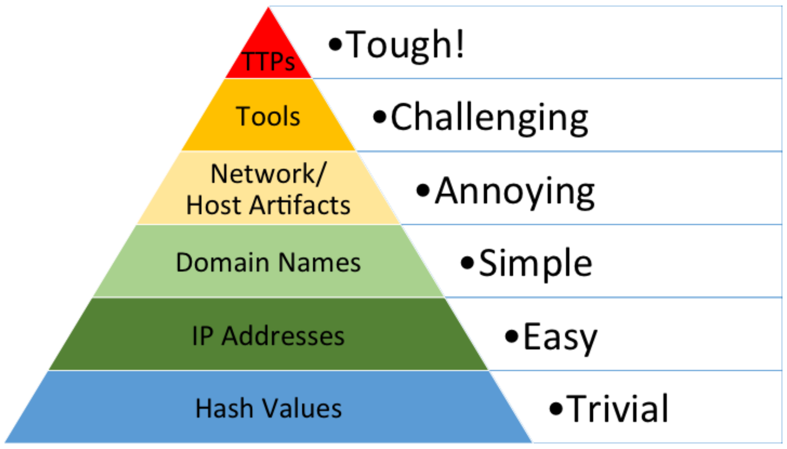


Figure 5.2 Pyramid of Pain [72].

To accurately model adversarial behaviors, two principal methodologies are utilized: *data flow diagrams* (DFDs) and *attack trees*.

A Data Flow Diagram (DFD) is a graphical tool that illustrates the flow of data within a system [73]. DFDs are instrumental in mapping out the sequence of processes and the circulation of information, depicted in relatively abstract terms. This modeling tool helps in identifying core processes, systems, and activities without delving into technological specifics, providing a high-level overview of system interactions [73].

An attack tree is a comprehensive model used to describe potential adversary attacks in cybersecurity [74]. It organizes the methods of attack within a hierarchical, tree-structured framework. The primary objective or the main threat is represented at the root, while the various strategies to achieve this threat are mapped out as leaf nodes [74]. Nodes within the tree can be classified as AND or OR types: OR nodes denote alternative methods to achieve a goal, whereas AND nodes describe sequential steps required to reach a particular objective. The construction of an attack tree begins by identifying the main goals of potential attacks. Each primary goal spawns a separate tree, although different trees may share common sub-goals and branches. The process continues by branching out possible attack methods under each goal, systematically expanding the tree.

5.2 Model Data Requirements

To effectively capture adversarial activity and bolster the detection methodology, precise data requirements are specified. This strategic approach is directly derived from the principles of *TTP-based threat hunting*, as proposed in [11]. This methodology emphasizes the importance of aligning data collection with known TTPs used by adversaries. By doing so, it ensures that the data gathered is not only relevant but also structured in a way that optimally supports the detection and analysis of potential security threats. This targeted data modeling is crucial for the development of robust analytics that can accurately identify and mitigate adversarial actions. The risk of developing analytics without modeling the data first is that the analytics could be too specific to a certain environment making it harder to re-apply [11].

To establish effective data collection requirements, a list of essential data sources is compiled, drawing directly from the malicious activity model. This process results in a comprehensive set of data requirements designed specifically to monitor lateral movement, aggregating necessary analytics for each technique under investigation. Sensor and data source selection are critical components of this framework, as they must provide a balance between the depth of contextual information and the volume of data produced [13]. While it is generally true that more detailed context leads to increased data volume, a well-informed understanding of adversarial behavior and abstract analytics can optimize the data collection strategy. By tailoring the approach to focus on the most relevant data, it is possible to reduce the overall load of data collection without compromising the effectiveness of the threat detection system [11].

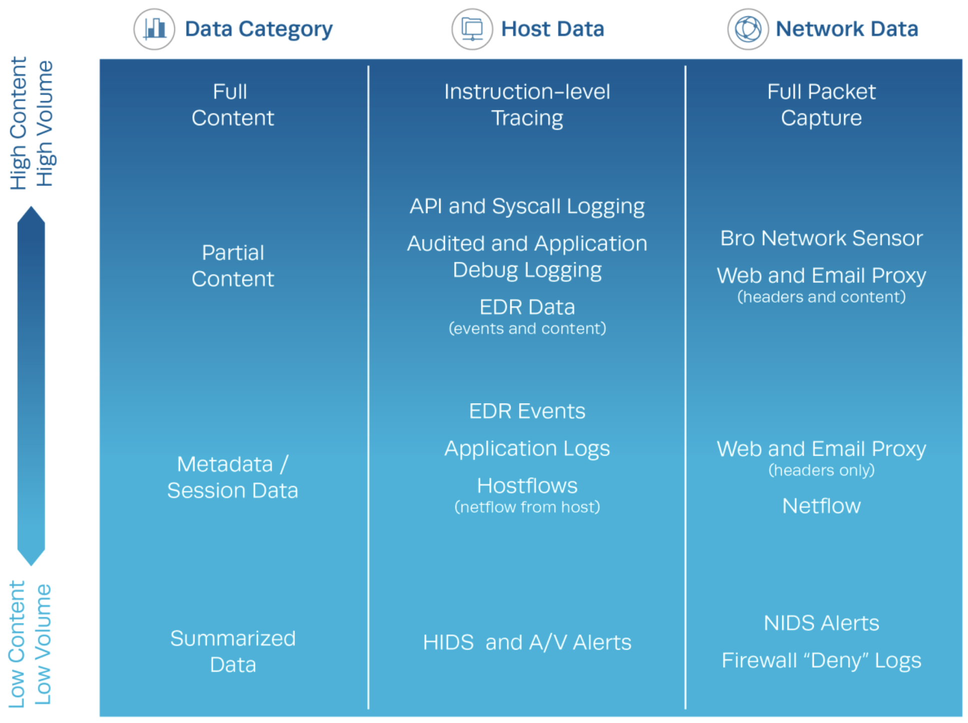


Figure 5.3 Context vs. volume of host and network data [11].

Sensors that offer continuous observability of the network and systems are preferred over those that operate solely on signatures and alert-based mechanisms. This preference stems from the need for a more dynamic and comprehensive monitoring approach that can capture a broader spectrum of potential threats in real-time. However, while continuous monitoring is prioritized, alerts generated by existing defense systems are not disregarded; instead, they are utilized as crucial anchors in the subsequent analysis of adversary behavior. These alerts can provide immediate indicators of potential security incidents, serving as valuable starting points for deeper investigations into malicious activities.

To feasibly collect, aggregate, and analyze data, the *Elastic Common Schema* [64] is leveraged to standardize and unify the captured data. Utilizing a common data schema facilitates the integration of diverse data sources, allowing for seamless correlation of captured data with related events. This standardized approach aids significantly in the analysis of adversarial behavior by ensuring that all data elements are consistently formatted and easily accessible. This uniformity not only streamlines the analytical process but also enhances the ability to detect and respond to threats by providing a clearer view of how events are interconnected.

The ATT&CK framework utilizes *Data Sources*, which describe and categorize information that can be collected by sensors [14]. By leveraging ATT&CK data sources and Elastic Common Scheme, it is possible to closely align the telemetry collected with adversary activity, making these sources one of the most crucial elements in the development of rules for detecting adversary actions.

Understanding lateral movement techniques necessitates a robust data model that captures all relevant information for effective detection. Designed data model leverages the ATT&CK data sources [14] and the Elastic Common Schema [64] to provide a comprehensive representation of lateral movement traces across compromised hosts and infected networks.

Data sources in ATT&CK are modeled as a pair, which consists of *Data Source* and *Data Component* [14].

Data Source	Data Component
Categories of information	Specific property or value of a data source

Table 5.1 MITRE ATT&CK data model [14].

The monitoring setup detailed in Section 4.2 allows collecting rich telemetry data, which is crucial for developing detection methodologies that are both effective and efficient. The integration of these advanced logging capabilities with the proposed sophisticated data model ensures that the captured dataset is detailed and actionable, facilitating the identification and analysis of lateral movement activities.

While lateral movement often blends seamlessly with benign network traffic, utilizing legitimate software and credentials, the subtle traces it leaves behind are invaluable for behavioral-based detection methodologies. These traces, though often slight, provide critical indicators that can differentiate malicious activities from normal operations. By focusing on these nuanced differences, the proposed detection methodologies are able to identify and respond to lateral movements more effectively, even amidst the complexity of legitimate network usage.

5.3 Develop Hypothesis

A hypothesis in cyber threat hunting represents an informed assumption that an adversary will exhibit specific behaviors during an attack [13]. This educated belief is grounded in a thorough understanding of known adversarial behaviors and tactics. Based on this knowledge, a hypothesis is formulated to detect such behaviors through abstract analytics.

The proposed hypothesis should avoid being overly specific; instead, it should focus on behavioral invariants that are less likely to change even as tactics evolve [11]. By centering on these invariant aspects of adversarial behavior, the hypothesis becomes a powerful tool in the threat hunter's arsenal, enabling them to anticipate and detect actions indicative of malicious intent effectively.

5.4 Emulate Adversary

Emulating adversary behavior is a crucial step in the cyber threat analysis process. This emulation involves small-scale, repeated engagements designed to improve and test defenses by introducing a variety of malicious behaviors into the environment. In organizational settings, adversary emulation is often conducted through collaborative efforts between the *red team*, which simulates attacks, and the *blue team*, which defends against them, a practice commonly referred to as *purple teaming* [12]. For this research, purple teaming procedures are adopted to align with the specific scale and scope of this research, ensuring that they are both applicable and effective within the context of this research.

As the cybersecurity community develops and deploys novel detection methods, both security researchers and adversaries persistently adapt to the changing landscape. Security research is increasingly focused on identifying potential ways that these new methods can be circumvented, while adversaries continually evolve their strategies to bypass or neutralize these defenses. In this dynamic environment, the design of adversary emulation scenarios becomes critical. These scenarios should concentrate specifically on understanding and replicating the strategies adversaries employ to achieve their objectives, ensuring that the defenses can withstand not only current but also emerging threats [12]. This approach helps in maintaining a proactive stance in cybersecurity operations, ready to adapt and respond to the ever-evolving tactics of adversaries.

Adversary emulation typically follows a structured four-step process to ensure comprehensive and effective simulation of adversarial tactics [75]:

1. **CTI Research**: Conduct in-depth research to identify an adversary that represents a relevant and significant or growing threat. This includes a deep analysis of the selected adversaries to fully understand the scope, sophistication, and potential impact of each emulation plan.

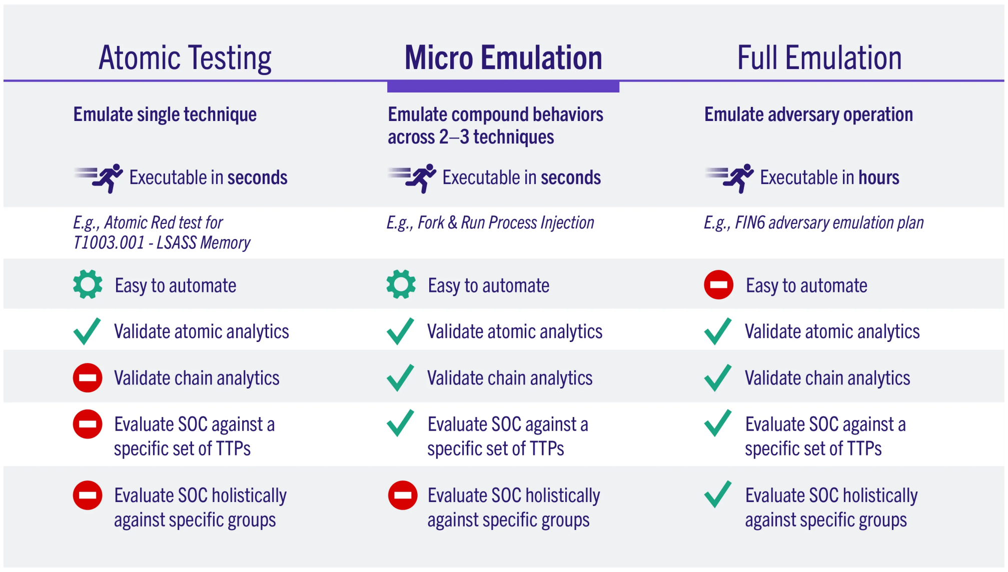
2. **Technique Selection**: Choose techniques from a broad array of tactics extracted from CTI reports, and organize these techniques into a coherent emulation scenario.

3. **Offensive Procedures Development**: Develop offensive procedures to accurately emulate the selected scenario.

4. **Emulation Execution**: Execute the developed emulation plan to test and evaluate defense mechanisms.

While this workflow is instrumental in producing threat-informed assessments, its broad scope can be more expansive than necessary for research that does not focus on a single adversary but rather on general adversarial tactics. To address this, MITRE researchers advocate for the use of *micro emulation plans* [75]. These plans concentrate on specific aspects of adversarial behavior [75], allowing for more targeted and manageable emulation efforts that align closely with the research objectives, especially when exploring particular tactics rather than broad adversary profiles.

Micro emulation plans adhere to the same foundational four steps as full adversary emulation plans, albeit executed in a more streamlined and focused manner. The micro plans are designed to facilitate quick and efficient validation of defense systems through the construction of smaller-scale, fully automated adversary emulation scenarios [75]. By concentrating on specific aspects of adversarial behavior, micro emulation plans allow for rapid testing and refinement of security measures, providing a practical approach to continuously enhance defensive capabilities without the extensive resource commitment typically required for larger-scale emulations.



Atomic Testing	Micro Emulation	Full Emulation
Emulate single technique	Emulate compound behaviors across 2–3 techniques	Emulate adversary operation
Executable in **seconds**	Executable in **seconds**	Executable in **hours**
E.g., Atomic Red test for T1003.001 - LSASS Memory	*E.g., Fork & Run Process Injection*	*E.g., FIN6 adversary emulation plan*
⚙ Easy to automate	⚙ Easy to automate	⛔ Easy to automate
✓ Validate atomic analytics	✓ Validate atomic analytics	✓ Validate atomic analytics
⛔ Validate chain analytics	✓ Validate chain analytics	✓ Validate chain analytics
⛔ Evaluate SOC against a specific set of TTPs	✓ Evaluate SOC against a specific set of TTPs	✓ Evaluate SOC against a specific set of TTPs
⛔ Evaluate SOC holistically against specific groups	⛔ Evaluate SOC holistically against specific groups	✓ Evaluate SOC holistically against specific groups

Figure 5.4 Comparison of various approaches to adversary emulation [75].

Finally, the execution of micro emulation plans is designed to be rapid, typically completing within seconds [75]. This swift execution allows for frequent and iterative testing, making it possible to quickly assess and enhance the effectiveness of defense systems against specific adversarial tactics.

5.4.1 Emulation Plan

Developing an adversary emulation scenario for the purpose of testing detection methods necessitates a comprehensive, high-level plan. This plan should be meticulously crafted using a deep understanding of the target environment, including knowledge of existing defenses, monitoring systems, and any gaps in analytic detection capabilities [76]. It is essential that the plan is detailed enough to facilitate thorough verification of defense systems, yet remains flexible to allow for extensions and adaptations as new insights or requirements emerge [12].

The adversary emulation plan should include several key components to ensure a robust and effective test of the defensive capabilities [12]:

1. **Sensor or Analytic and Defensive Capabilities**: Identification of the specific sensors, analytics, and defensive mechanisms that will be evaluated.
2. **Common Adversary Behavior**: Selection of adversary behaviors that are common and relevant to the scenario, ensuring that the emulation is realistic and applicable.
3. **Plan of Action**: A structured outline detailing the sequences of actions that will be undertaken to test and verify the defensive capabilities. This plan should be precise but adaptable, providing clear guidance while allowing for necessary adjustments.
4. **Required Resources**: Specification of all necessary systems, networks, and other resources that are critical for conducting the cyber test. This includes any tools, access permissions, and infrastructural elements needed to execute the emulation plan effectively.

This structured approach not only ensures that the emulation scenarios are relevant and comprehensive but also that they provide actionable insights that can directly inform and enhance the effectiveness of the organization's cybersecurity measures.

5.4.2 Emulation Execution

With an emulation plan in place, the adversary emulation is executed while logging all activities performed during the engagement to ease the analytics development.

Before initiating the adversary emulation, a detailed log template is prepared. The log contains the day, in `YYYY-MM-DD` format, of the engagement, starting time in `HH:MM` format, time difference from start for each emulation step, emulation step description, results if any. The adversary emulation engagement report contains short summary about the emulation execution instance, written after the engagement finished.

5.5 Develop Analytics

The analytic should specifically identify the adversarial behavior noted in the behavior model and leverage the data model as much as possible [11].

The penultimate step in the developed analysis approach involves the development of analytics tailored to detect adversarial behavior. This critical phase begins by identifying any gaps in data collection and proposing appropriate mitigation to ensure comprehensive coverage. Subsequently, the analytics themselves are carefully implemented, followed by a rigorous verification and evaluation process. This systematic approach ensures that the analytics are not only robust and capable of detecting adversarial activities but are also fine-tuned to respond to the specific dynamics and challenges presented by the observed threat landscape.

Before embarking on the development of analytics, it is crucial to ensure that all existing data sources are operational and generating valid data, and that the search platform is fully functional. Should any discrepancies or errors be detected, they must be promptly addressed. This can involve investigating and resolving issues, modifying configurations, or deploying additional sensors to cover any identified gaps in data collection. In situations where deploying new sensors is not feasible—whether due to availability constraints or other limitations—it is acceptable to merely acknowledge these gaps. This acknowledgment allows for an adjustment of the analytics development process, ensuring that the objectives of data collection still provide sufficient observability despite the limitations. By incorporating these checks and balances, the analytics development can proceed with a clear understanding of the system's capabilities and limitations.

Utilizing the behavior model, specified data requirements, and the data actually collected, the final analytics are developed. The development of these analytics is inherently iterative, necessitating continual re-evaluation to refine their accuracy and effectiveness. This process ensures that the analytics remain responsive to evolving adversarial tactics and the dynamic nature of network environments, ultimately enhancing the capability to detect and mitigate threats.

Finally, the proposed detection methodology, based on the analytics developed in this research, is operationalized by developing Elastic detection rules. These rules are then deployed within Elasticsearch environments, utilizing *Event Query Language* (EQL) [77] to effectively query and analyze security events. This deployment allows for the real-time application of the developed analytics, enabling the monitoring systems to detect malicious activities promptly and accurately. By integrating these advanced detection capabilities, the research enhances the responsiveness and efficacy of cybersecurity defenses in identifying and mitigating threats.

5.5.1 Develop Detection

This section outlines the structure and features of the proposed detection rules that are pivotal in identifying cybersecurity threats effectively.

Detection rules are initially crafted in the Sigma format, a generic and open signature format that enables researchers to describe detection-relevant events comprehensively [9]. Created to facilitate the sharing of detection methods in a structured manner, the Sigma project allows researchers to disseminate their findings broadly, enhancing detection capabilities across various security platforms [9]. This format ensures that the results of cyber threat analytics can be universally applied, independent of the specific defense systems in use.

The proposed Sigma detection rules are then converted into Kibana SIEM detection rules in `ndjson` format. Kibana SIEM detection rules follow the Elastic detection rules specification. Elastic detection rules offer a more rich feature set than the Sigma detection rules format and can be categorized accordingly. The rules generated by converting from the Sigma format are subsequently enriched with additional details, stored as metadata, to enhance their functionality and applicability within the Elastic ecosystem.

Detection Rules Specification

Each Sigma detection rule is crafted in YAML format, adhering to the Sigma specification to ensure consistency and interoperability within the Sigma ecosystem [78, 79]. The rules must comply with conventions defined in the Sigma specification to facilitate seamless integration and functionality across various platforms. The complete Sigma detection rule schema is outlined in the Rx YAML schema D.1.

Sigma Rules: Detection Component

The detection component is the cornerstone of any Sigma rule, specifying the criteria that the rule searches for across relevant logs [54]. This component is crucial for identifying specific events or patterns within the data.

Detection criteria are organized into selection groups to enhance readability and facilitate filtering [79]. Each group specifies necessary data fields and their corresponding values required for detection. These values can be refined using modifiers, appended after the field name with a pipe character `|`, allowing for complex operations to be performed on the data within Sigma rules [79]. Modifiers can be applied to both single values and lists and can be chained to increase their effectiveness. Furthermore, data fields can contain two special values—`null` and an empty string `''`. These special values are used to create negative filters or to search for any existing value in a data field, enhancing the flexibility of rule conditions [79].

The condition section within the detection component is pivotal, structuring the logic of the Sigma rule. It supports logical operators such as *and*, *or*, and *not*, and features like `1 of them`/`all of them`, implementing logical *or*/*and* across selected search groups [79].

Detection Rules Deployment

To deploy the proposed detection rules, those in Sigma format are converted into Elastic detection rules.

Elastic detection rules operate periodically, scanning for events that match defined criteria. When a rule's criteria are met, a detection alert is generated [71]. Elastic also provides a suite of pre-defined detection rules [10] and allows users to craft custom detection rules.

Users can create the following types of rules [71]:

- **Custom query**: A search query-based rule that scans defined data indices and triggers an alert when events match the rule's query.

- **Machine learning**: Utilizes machine learning to identify anomalies, requiring a paid license [80].
- **Threshold**: Triggers an alert when a defined number of events contain a specified field value.
- **Event correlation**: Scans selected data indices and triggers an alert when results match an Event Query Language (EQL) query.
- **Indicator match**: Triggers an alert when fields in the Elastic Security index match values defined in the specified indicator index pattern.
- **New terms**: Alerts for each new term or term combination detected in the collected events within a specified time frame.
- **ES|QL**: Scans selected data indices and triggers an alert when results match an Elasticsearch Query Language (ES|QL) query.

To configure a detection rule, users must specify Elastic index patterns or select a data view field. Rules of type custom query, machine learning, event correlation, and indicator match can include *Exceptions* to prevent alert generation even when the criteria are met, useful for reducing false positives [71].

The rules developed during this research are of the *Event correlation* type, designed to resist frequent changes in easily altered features, represented by the base levels of the Pyramid of Pain as shown in Figure 5.2. Behavior-based and heuristic-based detections, more resilient to changes in the cyber threat landscape, follow the schemas illustrated in Figures 3.3 and 3.4 respectively. Event correlation rules define relationships between collected events to trigger an alert for matching events or sequences of events, leveraging the capabilities of Event Query Language (EQL) to also detect missing events in a sequence [80, 77].

5.6 Verify Analytics

This section outlines the processes and metrics employed to verify the effectiveness and accuracy of the analytics developed in this research. It includes detailed evaluations of detection performance using both static and dynamic methods.

5.6.1 Static Analysis

Detection rules written in the Sigma format undergo validation through the *Sigma Rules Validator* [81], a GitHub Action that triggers each time rules are modified. This validator employs a `JSON` schema D.2 to ensure the syntax of the proposed detection rules is correct. Regular validation of detection rules via this tool provides rapid and reliable feedback, significantly enhancing the robustness and quality of the detection rules. This systematic validation process is crucial for maintaining high standards in rule development and ensuring that the rules perform as expected without syntax errors or logical inconsistencies.

5.6.2 Dynamic Analysis

The developed analytics should also be tested for functionality to ensure they accurately represent the logic of the hypothesis. This verification involves *dynamic analysis*, which tests the correctness of the system under test (SUT) by executing the SUT and recording the outputs.

To conduct dynamic analysis, detection rules that need verification are deployed in a system capable of running them. Elastic Security is utilized for this purpose due to its robust capabilities. Once the detection rules are operational, adversary emulation is executed on a monitored system.

The detection rules are then applied to data collected during this emulation, providing a practical test of their effectiveness.

Comparing detection results with adversary emulation report and activity log allows for a comprehensive evaluation of the analytics' accuracy [12]. *The precision* of the analytics is measured by the rate of false positives, which is determined by running the analytics over a prolonged period across the entire test environment and counting occurrences of false alerts [13]. Analytics with a high rate of false positives may be disregarded by analysts, diminishing their utility. *The recall* of the analytics, defined as the ratio of correctly detected adversarial behaviors within the test environment, is also evaluated [13].

The results from analytics verification can then be used to refine and improve the analytics. If necessary, new sensors may be developed or deployed to enhance detection capabilities.

Chapter 6
Lateral Movement Analysis

This chapter delves into the analysis of technique T1091, Replication Through Removable Media, as defined in the MITRE ATT&CK framework [23]. This technique involves adversaries exploiting removable media devices, such as USB drives, to move laterally across systems and deploy malware [23]. By leveraging such common tools, attackers can bypass network defenses and execute malicious payloads directly on target hosts. This analysis follows a structured approach as previously defined in Chapter 5, encompassing several key phases: modeling adversarial behavior, acquiring necessary data, developing hypotheses, creating and executing adversary emulation plans, developing analytics, and finally, verifying the effectiveness of proposed detection methodologies. Each phase is crucial for comprehensively understanding and mitigating the threats posed by this technique, with detailed results discussed in the following subsections.

6.1 Adversarial Behavior Model

This section examines the adversarial behavior associated with technique T1091, Replication Through Removable Media, as depicted in the attack tree illustrated in Figure 6.1. Malware exploiting this technique utilizes removable media to propagate, following a series of defined procedures to spread and ensure persistence within the target environment [23]. Recognizing and understanding these procedures are crucial for devising effective countermeasures and enhancing detection capabilities.

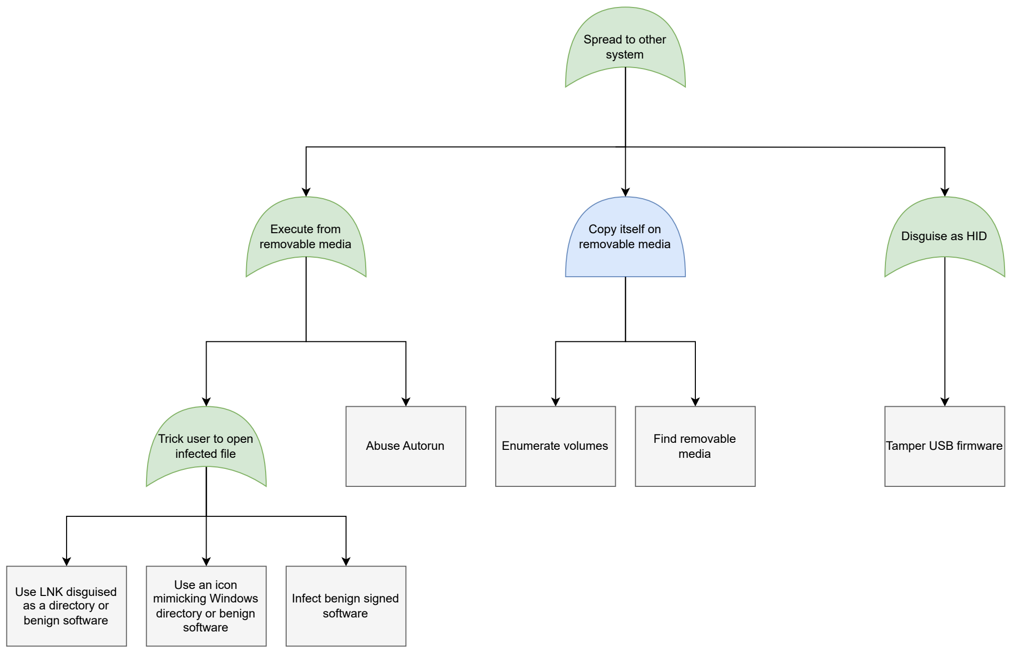

Spread to other system
Execute from removable media
Copy itself on removable media
Disguise as HID
Trick user to open infected file
Abuse Autorun
Enumerate volumes
Find removable media
Tamper USB firmware
Use LNK disguised as a directory or benign software
Use an icon mimicking Windows directory or benign software
Infect benign signed software

Figure 6.1 T1091 Replication Through Removable Media attack tree.

Literature review and CTI research helped to identify typical characteristics of adversarial behavior exhibiting technique T1091:

- **Searching for Removable Media:** Malware systematically scans the system to detect connected removable devices such as USB drives, which are potential vectors for spreading the infection [82].
- **Writing Files on Removable Media:** Upon identifying a suitable device, the malware writes copies of itself or other malicious files onto the removable media, preparing it to infect additional systems [82].
- **Modifying Files on Removable Media:** Malware may modify existing files on the device, integrating malicious code into legitimate files to evade detection and facilitate propagation [24].
- **Side-loading Malicious DLLs:** Malicious dynamic link libraries (DLLs) are placed in the path of legitimate software on the removable media, allowing the malware to execute when the legitimate software is run [24].
- **Hiding Malicious Content and Masking it Using `LNK` Files:** Malware often disguises its presence using shortcut (`LNK`) files that appear harmless but execute malicious operations when activated [82, 83].

- **Copying Itself from Removable Media:** When infected removable media is connected to another host, the malware replicates itself onto the new host machine, furthering its spread [25].
- **Modifying Firmware to Emulate HID (Human Interface Device):** In more advanced attacks, malware can alter the firmware of removable devices to mimic input devices like keyboards, enabling it to execute commands directly on the host computer [84].

Each of these behaviors exploits the unique capabilities of removable media to spread malware, highlighting their importance as critical points for developing targeted detection and prevention strategies.

6.2 Data Model

The data model for analyzing Technique T1091, Replication Through Removable Media, is structured around specific data sources identified by the MITRE ATT&CK framework. This model concentrates on capturing the digital traces left by malware as it propagates through removable media. A detailed summary of these data sources and their components, specifically targeted in this research to effectively monitor and analyze the spread of malware via removable devices, is presented in Table 6.1.

Data Source	Data Component
Drive	Drive Creation
File	File Access File Creation
Process	Process Creation

Table 6.1 T1091 data model.

To effectively capture the data outlined in Table 6.1, multiple sensors were strategically deployed throughout the testing environment, as detailed in Section 4.2. Alongside common sensors, *Group Policy Objects* (GPOs) were utilized to configure host machine settings to enhance data collection capabilities. Critical settings, including **Turn on PowerShell Script Block Logging**, **Turn on Module Logging** [85], and **Turn on PowerShell Transcription** [86], were activated through Administrative Templates for PowerShell Core within Group Policy. These settings are vital for capturing comprehensive logs of PowerShell activities, which are often exploited in malicious operations via removable media.

Data collected, including logs, host machine metrics, and network traffic, are ingested by the Elastic Agent [87] and processed into the Elastic Common Schema (ECS) [64] by the Elastic Agent Integrations [88]. This standardization of data processing enhances efficiency in data analysis within Kibana, enabling streamlined examination and rapid response to potential security incidents.

The deployment strategy was meticulously designed to ensure comprehensive monitoring of all potential entry points for removable media within the testing environment. Each sensor was precisely configured to capture detailed information on process execution paths, file access times, and drive connections. This detailed logging is essential for a thorough analysis of potential malicious activities related to removable media.

This comprehensive approach to data collection and processing not only allowed for the direct observation of behaviors indicative of technique T1091 but also ensured that the collected data was both relevant and sufficient for robust analysis. The precise configuration of sensors and the integration of advanced data processing tools were crucial in minimizing data noise and enhancing the system's detection capabilities. This meticulous setup yielded a high-fidelity dataset that was instrumental in testing and verifying the effectiveness of the proposed detection methodologies.

6.3 Hypothesis

In the context of technique T1091, Replication Through Removable Media, malicious actors employ a variety of methods to propagate malware across systems. A common procedure involves exploiting the Autorun feature of removable media [23], which can automatically execute a malicious payload when the media is connected to a host. Additionally, attackers may deceive users into launching malicious payloads disguised on removable media [82, 83], directly transfer malware to these devices [25], or modify the firmware of the device to emulate a Human Interface Device (HID) [84], allowing them to execute commands directly on the host machine.

The hypothesis, derived from a detailed examination of behaviors associated with T1091, suggests several potential indicators of compromise, including:

- Detection of actions searching for connected removable media.
- Observations of suspicious file-writing activities on removable media.
- Modifications to files stored on removable media.
- Identification of suspicious content present on removable media.
- Use of configurations exploiting the Autorun feature.
- Firmware alterations on removable media indicative of HID emulation.

Suspicious content may include hidden files and folders, `.lnk` shortcuts camouflaged as benign items through icon modifications [82], or `.lnk` files targeting potentially malicious executables such as `cmd.exe`. These deceptive tactics or configurations are designed to either trick users into initiating malicious actions or to automate the execution of malware, complicating detection and mitigation efforts significantly.

6.4 Emulation Plan

This section details the strategy for accurately emulating technique T1091, Replication Through Removable Media, using selected procedures associated with USB malware propagation. The emulation exercises will be conducted using PowerShell scripts and a physical USB drive to closely mimic real-world attack scenarios.

The initial step involves reviewing existing T1091 emulation procedures from the Atomic Red Team [89]. These procedures typically include scanning for attached removable drives and creating new text files on them, as demonstrated in the referenced PowerShell script C.1. This basic emulation serves as a preliminary test to evaluate the system's response to new file creation on removable media. Additionally, a cleanup command is employed to remove any test files from the removable media, ensuring a clean state post-emulation.

Further emulation exercises are designed to simulate more sophisticated behaviors associated with T1091, particularly focusing on techniques that malware might use to evade detection. These include:

- Hiding files using the `attrib.exe` utility.
- Overriding file attributes to hide files and folders.

These behaviors are emulated by creating both a single hidden file and a hidden folder containing a file on the removable media. The PowerShell scripts C.4, C.5 and C.6, C.7 provide detailed commands used to modify file system visibility attributes effectively.

The T1091 emulation plan consists of the following steps:

1. Connect removable media to the target machine.
2. Execute the Atomic T1091 emulation as detailed in PowerShell script C.1.
3. Implement file and folder hiding using the `attrib.exe` utility, demonstrated by PowerShell scripts C.4 and C.5.
4. Implement file and folder hiding by overriding their attributes, demonstrated by PowerShell scripts C.6 and C.7.
5. Execute cleanup commands to remove all test artifacts using the cleanup PowerShell script C.2.

This structured approach not only replicates behaviors associated with the T1091 technique but also assesses the system's capability to detect and respond to various methods used by malware to disguise its presence on removable media. The outcomes of this emulation are crucial for developing the T1091 analytics.

6.5 Emulation Execution

The emulation plan, as outlined in Section 6.4, was implemented on a Windows Server 2022 virtual machine, using a physical USB drive and adversary emulation PowerShell scripts detailed in Appendix C. This setup aimed to simulate the deployment and propagation of malware via removable media, focusing on a specific attack path visualized in Figure 6.2 from the T1091 attack tree, which is illustrated in Figure 6.1.

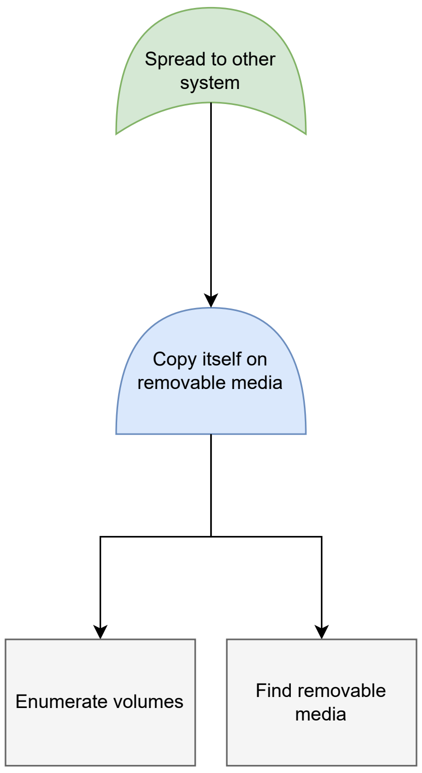


Figure 6.2 Detail of the emulation execution scenario visualized by attack tree.

The initial step involved running the `Get-RemovableMedia.ps1` script, referenced in PowerShell script C.3, which enumerated all attached removable media. This script successfully identified the connected USB devices without including any internal drives, confirming the effectiveness of the media detection mechanisms.

Following this, scripts designed to conceal files on the removable media were activated. The `New-HiddenFileOnRemovableMediaAttrbExe.ps1` and `New-HiddenFolderOnRemovableMedia-AttrbExe.ps1` scripts, detailed in PowerShell scripts C.4 and C.5, were executed. These scripts successfully created hidden objects on the removable media, mimicking malware propagation and concealment tactics.

Subsequently, scripts for hiding files by altering their attributes, `New-HiddenFileOnRemovable-MediaAttributesOverride.ps1` and `New-HiddenFolderOnRemovableMediaAttributesOverri-de.ps1`, scripts, detailed in PowerShell scripts C.4 and C.5, referenced in PowerShell scripts C.6 and C.7, were run. These also performed as intended, with the hidden files and folders being properly established, showcasing vulnerabilities associated with file attribute manipulation.

The emulation concluded with the cleanup process executed via the `Remove-TestFiles-FromRemovableMedia.ps1` script, referenced in PowerShell script C.2, which efficiently removed all test files from the removable media. This final step was essential to return the emulation environment to its original state and prevent residual data from impacting subsequent analyses.

The successful execution of this comprehensive emulation plan not only demonstrated the system's reactions to various file-hiding techniques but also generated crucial data that will be used to refine detection strategies. This empirical evidence is vital for developing analytics that can effectively detect and counter T1091 behavior, thus enhancing the robustness of defenses against such adversarial tactics.

6.6 Analytics

This section explores the development of detection analytics aimed at enhancing the capability to monitor and respond to incidents involving removable media, a prevalent vector in cybersecurity breaches. While current Elastic detection rules effectively address scenarios where processes are directly executed from removable media [90], this research seeks to broaden these capabilities by introducing novel detection methodologies. The focus is on scenarios not fully covered by existing frameworks, particularly the creation and manipulation of files on removable devices. Additionally, the Sigma ruleset includes a rule designed to detect first-time-seen removable devices connected to a monitored host machine [91].

Informed by the emulation results of technique T1091, this research proposes two new detection rules aimed at capturing subtle adversarial behavior exhibited by T1091:

1. **Detection of New File Creation on Removable Media**: This rule is specifically designed to target the creation of new files on removable devices, a common method employed by malware to propagate. By focusing on the moment of file creation, the rule aims to catch malware at a critical point in its dissemination process.

2. **Detection of Hidden Files via System Attributes Manipulation**: This rule focuses on identifying files whose attributes have been altered to conceal their presence. Manipulating system attributes to hide malicious files is a technique often utilized by adversaries to evade basic file system scans, making this rule essential for uncovering stealthy malware operations.

Both rules are designed to integrate seamlessly with existing security systems and anti-malware solutions. By enhancing detection capabilities in this manner, the proposed rules help close gaps in current defensive measures and contribute to a more robust cybersecurity posture.

Analytics Development

The analytics development begins with preprocessing data collected during the adversary emulation execution. Detailed logs from the T1091 adversary emulation, along with pre-defined detection rules provided by Elastic, were crucial in focusing the data analysis effort. Specifically, the rules *First Time Seen Removable Device* [92] and *Execution from a Removable Media with Network Connection* [90] helped filter the data into more manageable subsets, targeting specific suspicious activities related to removable media.

The initial dataset comprised 15,215 events, predominantly classified under the `"process"` category, confirming alignment with the data model discussed in Section 6.2. To focus on file-related activities, the data was filtered to include only those events categorized under `event.category: "file"` and either `event.type: "creation"` or `event.type: "access"`, reducing the dataset to 153 events.

A notable portion of these events originated from `elastic-endpoint.exe`, a component of the monitoring solution used in this research. To ensure the integrity and relevance of the data, events associated with this process were excluded from further analysis using the filter `NOT process.name: "elastic-endpoint.exe"`.

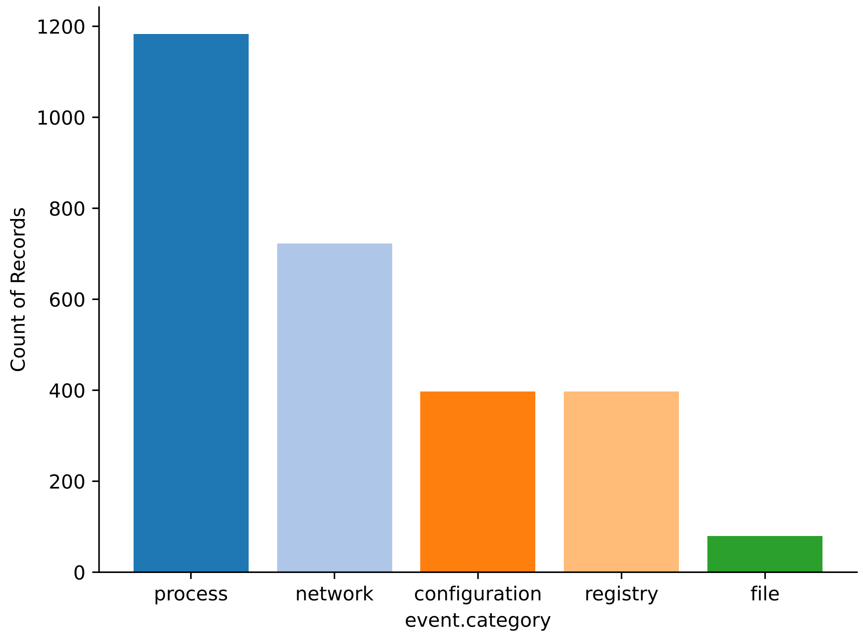


Figure 6.3 Top five values of `event.category` ECS field in raw data collected during T1091 emulation.

The visualization of the top event categories, as shown in Figure 6.3, underscores the predominance of process-related data. This finding provides a clear direction for subsequent analysis phases, helping to streamline the investigation process by focusing on data points most likely to reveal malicious activities associated with the T1091 technique.

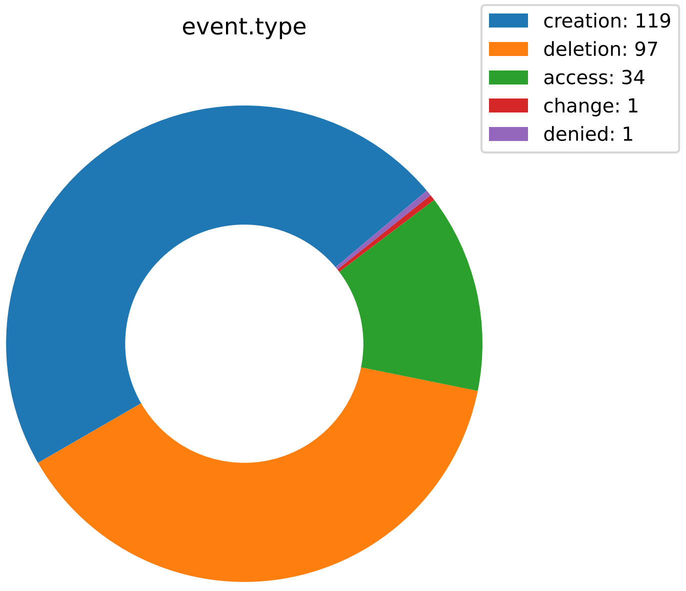


Figure 6.4 Top five values of `event.type` ECS field in raw `event.category: "file"` data collected during T1091 emulation.

Further refinement was carried out by examining the `file.path` field. Events where `file.path` did not reference the system drive (`C:`) were considered particularly relevant, as the focus was on removable media. This targeted analysis led to the identification of two pivotal file creation events. These two events are critical as they directly correspond to the typical behavior exhibited by malware leveraging removable media for propagation.

These findings informed the development of a new generic detection rule aimed at identifying new files on removable media. This rule is designed to be highly customizable, allowing security teams to tailor it to their specific needs by adjusting criteria such as drive letter and file characteristics that may suggest malicious activity.

```yaml
...
detection:
    selection_file_creation:
        EventId: 11
    filter:
        TargetFilename|startswith:
            - "C:"  # First partition/System
    condition: selection_file_creation and not filter
...
```

Listing 6.1 Detection component of proposed rule detecting new files on volumes with drive letter other than `C:`

The detection rule specified above has been translated into an EQL query for implementation within Kibana SIEM, facilitating real-time monitoring and detection:

```
file where winlog.event_id : "11" and not stringContains~(file.path, "C:")
```

Listing 6.2 EQL query to detect new files on removable media.

This rule exemplifies the nuanced approach necessary for detecting and analyzing adversarial actions involving removable media, highlighting the continuous adaptation required in cybersecurity defenses. By focusing on removable media not mapped to the standard system drive letter, it ensures a targeted and effective response to potential threats propagated via such devices.

Following the analysis of `event.category: "file"`, attention shifted to `event.category: "process"`, where most events were classified as `access`. However, the primary focus of this analysis, as outlined in the data model shown in Table 6.1, is on `event.type: "start"` (Process Creation) events. This event category represents only a small fraction (0.68%) of all collected `event.category: "process"` events, as illustrated in Figure 6.5.

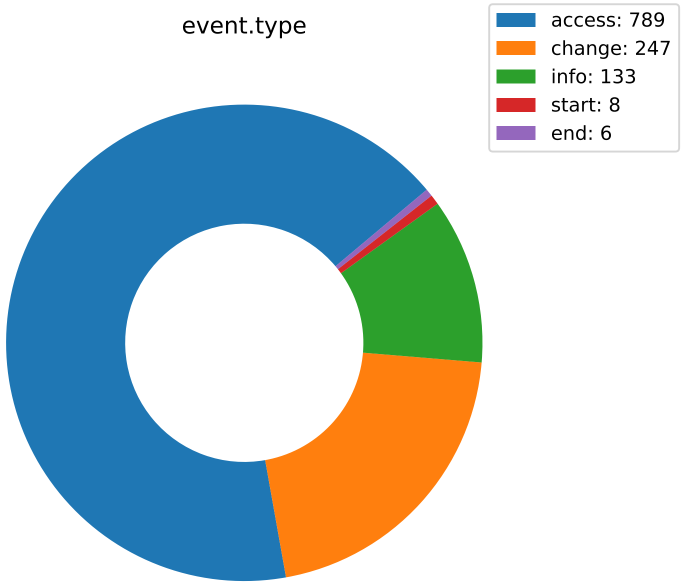


Figure 6.5 Top five values of `event.type` ECS field in raw `event.category: "process"` data collected during T1091 emulation.

Filtering for `event.category: "process" and event.type: "start"` narrows down the data. The focus is on records where the `process.args` field includes commands like `+h` or paths located on removable media, or where the `process.name` field is `attrib.exe`, to detect attempts to hide files via the `attrib.exe` utility. Excluding events related to the monitoring process `elastic-endpoint.exe` leaves 17 relevant events.

Based on these findings, a second rule is proposed to specifically target hidden file system objects on removable media. This rule, like the first, is adaptable to meet specific defense needs by including conditions or filters.

```
...
detection:
    selection_hide_file:
        CommandLine|contains:
            - 'attrib.exe'
            - '+h'
            - '[System.IO.FileAttributes]::Hidden'
    selection_system_drive_letter:
        TargetFilename|startswith: 'C:'
    condition: (selection_hide_file and not selection_system_drive_letter)
...
```

Listing 6.3 Detection component of proposed detection rule detecting hiding files on volumes with drive letter other than `C:`.

This rule is translated into an EQL query for deployment within the Kibana SIEM system:

```
process where (process.name == "attrib.exe" or ?process.pe.original_file_name
↪   == "ATTRIB.EXE") and stringContains(process.args, "+h") and not
↪   stringContains~(file.path, "C:")
```

Listing 6.4 Hide file on removable media.

Conclusion

The two proposed detection rules may help in safeguarding against the proliferation of malware via removable media. The proposed rules can be deployed to generate real-time alerts, providing quick recognition of adversarial lateral movement, which is a desired feature of rules detecting lateral movement [6].

For optimal effectiveness, it is recommended that these rules undergo further customization and rigorous testing within live environments. Such an iterative testing and refinement process will ensure that the rules are finely tuned to the specific needs of the organization and robust enough to handle the nuances of the prevailing threat landscape.

Integrating these rules into existing security frameworks may enhance the defensive capabilities and may provide a more resilient and agile response mechanism to combat the diverse and evolving threats associated with removable media.

6.7 Detection Verification

The effectiveness of the newly developed detection rules was evaluated within a controlled testing environment, proposed in Chapter 4, designed to precisely assess their capability to identify malicious activities involving removable media. The adversary emulation plan, as outlined in Section 6.4, was executed twice to verify the validity and performance of the proposed detection rules.

During the first testing run, a total of nine alerts were generated by the rule designed to detect new file creation on removable media. An example of such an alert is provided in Listing 6.5. These alerts corresponded to different scenarios involving removable media, including:

- Files created directly in the root directory of the removable media.
- Files created within a new folder on the removable media.
- Files created within nested folder structures on the removable media.

```
{
        ...
        "fields": {
                ...
                "event.category": [
                "file"
              ],
              "event.type": [
                      "creation"
              ],
                "kibana.alert.reason": [
                        "file event with process powershell.exe, file
                        ↪  T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride.md, by
                        ↪  Administrator on tgt created low alert New File Created Not On C:
                        ↪  Drive."
                ],
                "file.path": [
                        "E:\\T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride.md"
                ],
                "message": [
                "File created:\nRuleName: -\nUtcTime: 2024-04-26 16:43:36.916\nProcessGuid:
                ↪  {47d33b01-b6ea-662b-9d00-000000000500}\nProcessId: 1448\nImage:
                ↪  C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe\nTargetFilename:
                ↪  E:\\T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride.md\nCreationUtcTime:
                ↪  2024-04-26 16:43:36.916\nUser: TGT\\Administrator"
              ],
              "winlog.event_id": [
                      "11"
              ],
        }
        ...
}
```

Listing 6.5 Selected parts of an alerted generated by emulating T1091 behavior.

Three alerts were generated by the rule designed to detect hidden files on removable media using the `attrib.exe` utility, detailed in E.2. One such alert is detailed in Listing 6.6. These alerts reflected various scenarios, including:

- A hidden file located in the root directory of the removable media.
- A hidden folder, created using the `attrib.exe` utility.
- An unintended alert triggered by an incorrect `attrib +h` command in the emulation script, which was not part of the planned testing scenarios.

This mix of expected and unexpected alerts highlights the sensitivity of the detection rule to changes in file attributes, demonstrating its effectiveness in capturing even unintended modifications. The false alert also emphasizes the importance of precise scripting and parameter setting in emulation environments to avoid skewing test results.

```
{
        ...
        "fields": {
                ...
            "event.category": [
                    "process"
            ],
            "event.type": [
                    "start"
            ],
            "kibana.alert.reason": [
                      "process event with process attrib.exe, parent process powershell.exe, by
                      ↪  Administrator on tgt created low alert Hide File On Removable Media via
                      ↪  attrib.exe."
            ],
            "process.command_line": [
                    "\"C:\\Windows\\system32\\attrib.exe\" +h
                    ↪  E:/T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md"
            ],
                ...
        }
        ...
}
```

Listing 6.6 Selected parts of an alert generated by emulating T1091 behavior.

The second testing run was highly successful, generating four alerts, which matched the four expected outcomes based on the predefined scenarios. This includes one alert for ”File Created,” as detailed in Listing 6.7, and another for ”Hide File,” illustrated in Listing 6.8.

This consistency in the second run confirms the reliability and effectiveness of the detection rules under conditions that closely mimic potential real-world attacks. The alerts generated corroborate the rules’ capability to accurately identify both the creation and concealment of files on removable media, crucial for preemptive cybersecurity defenses.

```
{
        ...
        "fields": {
                ...
            "event.category": [
                      "file"
            ],
            "event.type": [
                    "creation"
            ],
            "kibana.alert.reason": [
                    "file event with process powershell.exe, file
                    ↪   T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md, by Administrator on
                    ↪   tgt created low alert New File Created Not On C: Drive."
            ],
            "file.path": [
                    "E:\\T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md"
            ],
            "message": [
                    "File created:\nRuleName: -\nUtcTime: 2024-04-30 22:06:19.381\nProcessGuid:
                    ↪   {47d33b01-6b25-6631-de00-000000000700}\nProcessId: 3580\nImage:
                    ↪   C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe\nTargetFilename:
                    ↪   E:\\T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md\nCreationUtcTime:
                    ↪   2024-04-30 22:06:19.381\nUser: TGT\\Administrator"
            ],
            "winlog.event_id": [
                    "11"
            ],
                ...
        }
        ...
}
```

Listing 6.7 Selected parts of an alert generated by emulating T1091 behavior.

```
{
        ...
        "fields": {
                ...
            "event.category": [
                    "process"
            ],
            "event.type": [
                    "start"
            ],
            "kibana.alert.reason": [
                    "process event with process attrib.exe, parent process powershell.exe, by
                    ↪   Administrator on tgt created low alert Hide File On Removable Media via
                    ↪   attrib.exe."
            ],
            "process.command_line": [
                    "\"C:\\Windows\\system32\\attrib.exe\" +h
                    ↪   E:/T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md"
            ],
                ...
        }
        ...
}
```

Listing 6.8 Selected parts of an alert generated by emulating T1091 behavior.

Even though no false positives were recorded during the tests, two potential issues were identified that might lead to false positives in the future.

The first issue affects both proposed rules; the rules currently filter collected events by the target filename drive letter. In their non-modified form, these rules could generate false positives if a monitored system has more than one internal volume attached. Therefore, it is crucial that the rules are personalized to better fit the specific environment in which they are deployed.

The second issue pertains only to the *Hide File on Removable Media* rule. Examination of a selected alert (see Listing 6.6) reveals that there is no `TargetFilename` field present. This may pose a challenge; however, this event is a *rule* event, generated by running the rule on collected data. The original process creation event does contain the `TargetFilename` field.

Despite these challenges, the rules proved to be effective in detecting behavior exhibited by the lateral movement technique T1091, Replication Through Removable Media. The proposed detection rules cover behaviors not previously addressed by available Prebuilt Elastic Detection rules [10] and Sigma rules [9], including the creation of new files and the concealment of files on removable media.

Given these results, continuous refinement and ongoing validation of these rules are recommended to ensure they remain effective against evolving adversarial tactics and strategies. This proactive approach will help maintain a robust defense posture in the face of changing cyber threats.

Chapter 7
Conclusion

This thesis has undertaken a detailed analysis of adversarial lateral movement, with a specific focus on detection methodologies within targeted environments. It explores the techniques cataloged under TA0008, Lateral Movement, in the MITRE ATT&CK framework, highlighting the pivotal role of lateral movement in the cyberattack lifecycle.

The primary aim of this research was to enhance detection capabilities by proposing novel detection rules tailored to identify key techniques that facilitate lateral movement. This was achieved by designing and deploying a modular, fully documented testing environment. The configuration of this environment, which included two host machines, an Active Directory domain, and a sophisticated monitoring system reporting to a well-known SIEM solution, is thoroughly described in Chapter 4.

Furthermore, a scaled analytics development method informed by threat intelligence was proposed in Chapter 5. This method is particularly suited for individual researchers or small teams and incorporates a detailed adversary emulation process.

The practical application of this methodology was demonstrated through the emulation of Technique T1091, Replication Through Removable Media. This emulation facilitated a deep dive analysis, culminating in the proposal of two new detection rules.

The efficacy of these rules was tested and evaluated, with the results discussed assessing the feasibility and effectiveness of the detection methods aimed at identifying lateral movement.

Contributions

This research yielded several key contributions:

- **Testing Environment Design and Deployment:** A custom-designed testing environment was developed to simulate real-world IT infrastructures at risk of adversarial lateral movements.
- **Monitoring System Design and Deployment:** A sophisticated monitoring system was conceptualized and implemented to effectively capture and analyze adversarial activities.
- **Analytical Development:** Detailed behavioral and data models were constructed to predict and understand adversarial behaviors.
- **Adversary Emulation Plan and Execution:** Emulation plans were crafted and executed to replicate adversarial tactics, providing essential insights into their operational techniques.
- **Analysis and Detection:** Intensive analysis was conducted on the data collected during adversary emulation, leading to the development of new detection rules.

- **Verification:** The proposed analytics and detection strategies were thoroughly verified to confirm their effectiveness and accuracy.

Implications and Future Work

The insights garnered from this study significantly contribute to the cybersecurity domain, particularly in enhancing the understanding and mitigation of lateral movement strategies employed by adversaries. The methodologies and infrastructures developed not only bolster current security postures but also pave the way for future research initiatives.

Future studies may build on this work by incorporating additional adversarial techniques and further refining the detection and analytical processes.

In summary, this thesis marks a significant advancement in combating cyber threats, offering a comprehensive framework for the analysis, detection, and mitigation of adversarial lateral movement. The strategies developed herein are designed to address current security challenges while remaining adaptable to the evolving landscape of cyber threats, thus ensuring sustained resilience in cybersecurity defenses.

Contents of the Attached Archive

```
detection-rules/
├── T1091 ........................................ proposed detection rules
│   └── converted/ ........................ rules converted from Sigma format
FIThesis-LaTeX/
├── ctufit-thesis.pdf ................................... this thesis
├── img ....................................... images used in this thesis
└── text ..................................... source code of this thesis
README.md
resources/ ...................................... data and other resources
├── analysis/
│   └── T1091/ ................ data generated or collected during T1091 analysis
│       └── generated-alerts/ ................ proposed detection rules results
└── cti/ ..................................... data used during CTI research
src/
├── adv-emu/
│   └── T1091/ .......................................... T1091 emulation
├── cti/ ............................. CTI data processing and visualization
└── test-env/
    ├── active-directory-lab/ ........ simulated environment build and deployment
    └── monitoring/ .................... monitoring deployment and configuration
```

Appendix B
Testing Environment Configuration

This chapter contains configuration used to simulate target environment.

Sysmon Configuration

```xml
<!-- Author: Olaf Hartong -->
<Sysmon schemaversion="4.83">
  <HashAlgorithms>*</HashAlgorithms>
  <CheckRevocation />
  <DnsLookup>False</DnsLookup>
  <ArchiveDirectory>Research</ArchiveDirectory>
  <EventFiltering>
      <ProcessCreate onmatch="exclude">
                <Image condition="begin with">C:\Program Files\SplunkUniversalForwarder\bin\</Image>
                <Image condition="begin with">C:\WindowsAzure\GuestAgent</Image>
          </ProcessCreate>
      <FileCreateTime onmatch="exclude"/>
      <NetworkConnect onmatch="exclude"/>
      <ProcessTerminate onmatch="exclude">
                <Image condition="begin with">C:\Program Files\SplunkUniversalForwarder\bin\</Image>
                <Image condition="begin with">C:\WindowsAzure\GuestAgent</Image>
          </ProcessTerminate>
      <DriverLoad onmatch="exclude" />
      <ImageLoad onmatch="exclude">
                <Image condition="is">C:\Tools\Sysinternals\Sysmon64.exe</Image>
                <Image condition="begin with">C:\Program Files\SplunkUniversalForwarder\bin\</Image>
                <Image condition="begin with">C:\WindowsAzure\GuestAgent</Image>
          </ImageLoad>
      <CreateRemoteThread onmatch="exclude"/>
      <RawAccessRead onmatch="exclude" />
      <ProcessAccess onmatch="exclude">
                <SourceImage
                ↪   condition="is">C:\windows\system32\csrss.exe</SourceImage>
                <SourceImage
                ↪   condition="is">C:\windows\system32\lsass.exe</SourceImage>
                <SourceImage condition="is">C:\Program Files\Microsoft
                ↪   Monitoring Agent\Agent\HealthService.exe</SourceImage>
                <SourceImage condition="begin with">C:\Program
                ↪   Files\SplunkUniversalForwarder\bin\</SourceImage>
          </ProcessAccess>
      <FileCreate onmatch="exclude"/>
      <RegistryEvent onmatch="exclude">
                <Image condition="begin with">C:\Program
                ↪   Files\SplunkUniversalForwarder\bin\</Image>
                <Image condition="is">C:\Program Files\Microsoft Monitoring
                ↪   Agent\Agent\HealthService.exe</Image>
                <Image
                ↪   condition="is">C:\tools\sysinternals\Sysmon64.exe</Image>
                <Image condition="is">⌟
                ↪   C:\windows\System32\WindowsPowerShell\v1.0\powershell.exe⌟
                ↪   </Image>
                <Image condition="is">C:\windows\Sysmon64.exe</Image>
                <Image condition="begin
                ↪   with">C:\WindowsAzure\GuestAgent</Image>
          </RegistryEvent>
      <FileCreateStreamHash onmatch="exclude"/>
      <PipeEvent onmatch="exclude">
                <Image condition="begin with">C:\Program
                ↪   Files\SplunkUniversalForwarder\bin\</Image>
                <Image condition="begin with">C:\Program Files\Microsoft
                ↪   Monitoring Agent\Agent\</Image>
                <Image condition="begin
                ↪   with">C:\WindowsAzure\GuestAgent</Image>
          </PipeEvent>
      <WmiEvent onmatch="exclude"/>
      <DnsQuery onmatch="exclude"/>
      <FileDelete onmatch="include"/>
          <ClipboardChange onmatch="exclude"/>
          <ProcessTampering onmatch="exclude"/>
          <FileDeleteDetected onmatch="exclude"/>
          <FileBlockExecutable onmatch="exclude"/>
          <FileBlockShredding onmatch="exclude"/>
  </EventFiltering>
</Sysmon>
```

Listing B.1 `olafhartong/sysmon-modular/sysmonconfig-research.xml` Sysmon configuration [66].

Appendix C
Adversary Emulation

This chapter contains adversary emulation implementation.

T1091: Replication Through Removable Media

This section contains code emulating T1091, Replication Through Removable Media, behavior.

T1091 Atomic Emulation

```yaml
attack_technique: T1091
display_name: "Replication Through Removable Media"
atomic_tests:
- name: USB Malware Spread Simulation
  auto_generated_guid: d44b7297-622c-4be8-ad88-ec40d7563c75
  description: |
    Simulates an adversary copying malware to all connected removable drives.
  supported_platforms:
    - windows
  executor:
    name: powershell
    command: |
      $RemovableDrives=@()
      $RemovableDrives = Get-WmiObject -Class Win32_LogicalDisk -filter "drivetype=2" | select-object -expandproperty DeviceID
      ForEach ($Drive in $RemovableDrives)
      {
      write-host "Removable Drive Found:" $Drive
      New-Item -Path $Drive/T1091Test1.txt -ItemType "file" -Force -Value "T1091 Test 1 has created this file to simulate malware spread to removable drives."
      }
    cleanup_command: |
      $RemovableDrives = Get-WmiObject -Class Win32_LogicalDisk -filter "drivetype=2" | select-object -expandproperty DeviceID
      ForEach ($Drive in $RemovableDrives)
      {
      Remove-Item -Path $Drive\T1091Test1.txt -Force -ErrorAction Ignore
      }
```

Listing C.1 T1091 Atomic emulation from Atomic Red Team [89].

T1091 Emulation: Cleanup

```
$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

ForEach ($DriveLetter in $RemovableMedia) {
    attrib -h -r -s /s /d $DriveLetter\T1091*.*  # unhide files

    $testFiles = Get-ChildItem $DriveLetter\*  -Include T1091*

    ForEach ($File in $testFiles) {
        Remove-Item -Path $File.FullName -Force -ErrorAction Ignore
    }
}

```

Listing C.2 T1091 Emulation cleanup.

T1091 Emulation: Get Attached Removable Media

```
$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -Filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

foreach ($Drive in $RemovableMedia) {
    Write-Host "Removable media found: " $Drive
}
```

Listing C.3 Enumerate attached removable drives.

T1091 Emulation: Create File On Removable Media and Hide It via `attrib.exe`

```
$TestFilename = "T1091_Test_NewHiddenFileOnRemovableMediaAttrbExe.md"
$TestFileValue = "T1091 Detection Test: New Hidden File On Removable Media
↪  attrib.exe"

$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

foreach ($DriveLetter in $RemovableMedia) {
    Write-Host "Removable media found: " $DriveLetter
    New-Item -Path $DriveLetter/$TestFilename -ItemType File -Force -Value
    ↪  $TestFileValue
    attrib +h $DriveLetter/$TestFilename
}
```

Listing C.4 Create a file on removable media and hide it via `attrib.exe`.

```
$TestDirName = "T1091_Test_NewHiddenFolderOnRemovableMediaAttrbExe"
$TestFilename = "T1091_Test_NewHiddenFolderOnRemovableMediaAttrbExe.md"
$TestFileValue = "T1091 Detection Test: New Hidden Folder On Removable Media
↪  attrib.exe"

$RemovableMedia = @()
$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -Filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

foreach ($DriveLetter in $RemovableMedia) {
        Write-Host "Removable media found: " $DriveLetter

        New-Item -Path $DriveLetter/$TestDirName -ItemType Directory -Force
        attrib +h $DriveLetter/$TestDirName

        New-Item -Path $DriveLetter/$TestDirName/$TestFilename -ItemType File
        ↪  -Value $TestFileValue -Force
}
```

Listing C.5 Create a folder on removable media and hide it via `attrib.exe`.

T1091 Emulation: Create File On Removable Media and Hide It By Overriding File Attributes

```
$TestFilename =
↪  "T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride.md"
$TestFileValue = "T1091 Detection Test: New Hidden File On Removable Media
↪  Attributes Override"

$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -Filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

foreach ($DriveLetter in $RemovableMedia) {
    Write-Host "Removable media found: " $DriveLetter

New-Item -Path $DriveLetter/$TestFilename -ItemType File -Value $TestFileValue
↪  -Force

Get-Item $DriveLetter/$TestFilename -Force | foreach { $_.Attributes =
↪  $_.Attributes -bor "Hidden" }
}
```

Listing C.6 Create file on removable media and hide it by overriding its attributes.

```
$TestDirName = "T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride"
$TestFilename =
↪  "T1091_Test_NewHiddenFolderOnRemovableMediaAttributesOverride.md"
$TestFileValue = "T1091 Detection Test: New Hidden Folder On Removable Media
↪  Attributest Override"

$RemovableMedia = Get-WmiObject -Class Win32_LogicalDisk -Filter "drivetype=2"
↪  | Select-Object -ExpandProperty DeviceID

foreach ($DriveLetter in $RemovableMedia) {
    Write-Host "Removable media found: " $DriveLetter

    New-Item -Path $DriveLetter/$TestDirName -ItemType Directory -Force
    Get-Item $DriveLetter/$TestDirName -Force |  foreach { $_.Attributes =
    ↪  $_.Attributes -bor "Hidden" }
    attrib +h $DriveLetter/$TestDirName

    New-Item -Path $DriveLetter/$TestDirName/$TestFilename -ItemType File
    ↪  -Value $TestFileValue -Force
}
```

Listing C.7 Create folder on removable media and hide it by overriding its attributes.

Appendix D
Detection Rules Specification

Sigma Detection Rules

```
    type: //rec
required:
    title:
        type: //str
        length:
            min: 1
            max: 256
    logsource:
        type: //rec
        optional:
            category: //str
            product: //str
            service: //str
            definition: //str
    detection:
        type: //rec
        required:
            condition:
                type: //any
                of:
                    - type: //str
                    - type: //arr
                      contents: //str
                      length:
                          min: 2
        rest:
            type: //any
            of:
                - type: //arr
                  of:
                      - type: //str
                      - type: //map
                        values:
                            type: //any
                            of:
                                - type: //str
                                - type: //arr
                                  contents: //str
                                  length:
                                    min: 2
                - type: //map
                  values:
```

```yaml
                      type: //any
                      of:
                          - type: //str
                          - type: //arr
                            contents: //str
                            length:
                                min: 2
optional:
    status:
        type: //any
        of:
            - type: //str
              value: stable
            - type: //str
              value: test
            - type: //str
              value: experimental
            - type: //str
              value: deprecated
            - type: //str
              value: unsupported
    description: //str
    references:
        type: //arr
        contents: //str
    author: //str
    date: //str
    modified: //str
    fields:
        type: //arr
        contents: //str
    falsepositives:
        type: //any
        of:
            - type: //str
            - type: //arr
              contents: //str
              length:
                  min: 2
    level:
        type: //any
        of:
            - type: //str
              value: informational
            - type: //str
              value: low
            - type: //str
              value: medium
            - type: //str
              value: high
            - type: //str
              value: critical
rest: //any
```

Listing D.1 Sigma rules `Rx` `YAML` schema [79].

```
  {
"$schema": "http://json-schema.org/draft-07/schema#",
"title": "Sigma rule specification V1.0.4 (2023/06/29)",
"type": "object",
"required": ["title", "logsource", "detection"],
"properties": {
  "title": {
    "type": "string",
    "maxLength": 256,
    "description": "A brief title for the rule that should contain what the rules is supposed to detect"
  },
  "id": {
    "type": "string",
    "description": "A globally unique identifier for the Sigma rule. This is recommended to be a UUID v4, but not mandatory.",
    "format": "uuid"
  },
  "related": {
    "type": "array",
    "description": "A list of related Sigma rules to keep track of the relationships between detections. This can be used to indicate that a rule is derived from another rule, or that a rule has been obsoleted by another rule.",
    "items": {
      "type": "object",
      "required": ["id", "type"],
      "properties": {
        "id": {
          "type": "string",
          "description": "A globally unique identifier for the Sigma rule. This is recommended to be a UUID v4, but not mandatory.",
          "format": "uuid"
        },
        "type": {
          "type": "string",
          "oneOf": [
            {
              "const": "derived",
              "description": "The rule was derived from the referred rule or rules, which may remain active"
            },
            {
              "const": "obsoletes",
              "description": "The rule obsoletes the referred rule or rules, which aren't used anymore"
            },
            {
              "const": "merged",
              "description": "The rule was merged from the referred rules. The rules may be still existing and in use"
            },
            {
              "const": "renamed",
              "description": "The rule had previously the referred identifier or identifiers but was renamed for whatever reason, e.g. from a private naming scheme to UUIDs, to resolve collisions etc. It's not expected that a rule with this id exists anymore"
            },
            {
              "const": "similar",
              "description": "Use to relate similar rules to each other (e.g. same detection content applied to different log sources, rule that is a modified version of another rule with a different level)"
            }
          ]
        }
      }
    }
  },
  "status": {
    "type": "string",
    "oneOf": [
      {
        "const": "stable",
        "description": "The rule didn't produce any obvious false positives in multiple
        ↪  environments over a long period of time"
      },
      {
        "const": "test",
        "description": "The rule doesn't show any obvious false positives on a limited set of
        ↪  test systems"
      },
      {
        "const": "experimental",
        "description": "A new rule that hasn't been tested outside of lab environments and
        ↪  could lead to many false positives"
      },
      {
        "const": "deprecated",
        "description": "The rule was replaced or is now covered by another one. The link
        ↪  between both rules is made via the `related` field"
      },
      {
        "const": "unsupported",
        "description": "The rule can not be used in its current state (special correlation log,
        ↪  home-made fields, etc.)"
      }
    ]
  },
  "description": {
    "type": "string",
    "description": "A short description of the rule and the malicious activity that can be
    ↪  detected",
    "maxLength": 65535
  },
  "license": {
    "type": "string",
    "description": "License of the rule according the SPDX ID specification
    ↪  (https://spdx.dev/ids/)"
  },
  "author": {
    "type": "string",
    "description": "Creator of the rule. (can be a name, nickname, twitter handle, etc.)"
  },
  "references": {
    "type": "array",
    "description": "References to the source that the rule was derived from. These could be
    ↪  blog articles, technical papers, presentations or even tweets",
    "uniqueItems": true,
    "items": {
      "type": "string"
    }
  },
  "date": {
    "type": "string",
    "description": "Creation date of the rule. Use the format YYYY/MM/DD",
    "pattern": "^\\d{4}/(0[1-9]|1[012])/(0[1-9]|[12][0-9]|3[01])$"
  },
  "modified": {
    "type": "string",
    "description": "Last modification date of the rule. Use the format YYYY/MM/DD",
    "pattern": "^\\d{4}/(0[1-9]|1[012])/(0[1-9]|[12][0-9]|3[01])$"
  },
  "logsource": {
    "type": "object",
    "description": "The log source that the rule is supposed to detect malicious activity in.",
    "properties": {
      "category": {
        "description": "Group of products, like firewall or process_creation",
        "type": "string"
      },
      "product": {
        "description": "A certain product, like windows",
        "type": "string"
      },
      "service": {
        "description": "A subset of a product's logs, like sshd",
        "type": "string"
      }
    }
  },
  "detection": {
    "type": "object",
    "required": ["condition"],
    "description": "A set of search-identifiers that represent properties of searches on log
    ↪  data",
    "additionalProperties": {
      "description": "A Search Identifier: A definition that can consist of two different data
      ↪  structures - lists and maps.",
      "anyOf": [
        {
          "type": "array",
          "items": {
            "anyOf": [
              {
                "type": "string"
              },
              {
                "type": "integer"
              },
              {
                "type": "object",
                "items": {
                  "type": "string"
                }
              }
            ]
          }
        },
        {
          "type": "object",
          "items": {
            "type": "string"
          }
        }
      ]
    },
    "properties": {
      "condition": {
        "type": "string",
        "description": "The relationship between the search identifiers to create the detection
        ↪  logic. Ex: selection1 or selection2"
      }
    }
  },
  "fields": {
    "type": "array",
    "description": "A list of log fields that could be interesting in further analysis of the
    ↪  event and should be displayed to the analyst",
    "uniqueItems": true,
    "items": {
      "type": "string"
    }
  },
  "falsepositives": {
    "description": "A list of known false positives that may occur",
    "uniqueItems": true,
    "anyOf": [
      {
        "type": "string",
        "minLength": 2
      },
      {
        "type": "array",
        "items": {
          "type": "string",
          "minLength": 2
        }
      }
    ]
  },
  "level": {
    "type": "string",
    "description": "The criticality of a triggered rule",
    "oneOf": [
      {
        "const": "informational",
        "description": "Rule is intended for enrichment of events, e.g. by tagging them. No
            ↪  case or alerting should be triggered by such rules because it is expected that a
            ↪  huge amount of events will match these rules"
      },
      {
        "const": "low",
        "description": "Notable event but rarely an incident. Low rated events can be relevant
            ↪  in high numbers or combination with others. Immediate reaction shouldn't be
            ↪  necessary, but a regular review is recommended"
      },
      {
        "const": "medium",
        "description": "Relevant event that should be reviewed manually on a more frequent
        ↪  basis"
      },
      {
        "const": "high",
        "description": "Relevant event that should trigger an internal alert and requires a
        ↪  prompt review"
      },
      {
        "const": "critical",
        "description": "Highly relevant event that indicates an incident. Critical events
            ↪  should be reviewed immediately. It is used only for cases in which probability
            ↪  borders certainty"
      }
    ]
  },
  "tags": {
    "description": "Tags to categorize a Sigma rule.",
    "type": "array",
    "uniqueItems": true,
    "items": {
      "type": "string",
      "pattern": "^[a-z0-9_-]+\\.[a-z0-9._-]+$"
    }
    }
  }
}
```

Listing D.2 Sigma rules `JSON` schema [79].

Appendix E
Analytics

T1091 Analysis: Proposed Detection Rules

```yaml
title: File Created On Removable Media
id: 6320ea94-5c93-43e3-8004-a3cc79c97868
status: experimental
description: Detects newly constructed file on removable media
references:
    - https://attack.mitre.org/techniques/T1091/
    - https://d3fend.mitre.org/offensive-technique/attack/T1091/
tags:
    - attack.lateral_movement
    - attack.t1091
author: Silvie Nemcova
date: 2024/04/19
logsource:
    service: sysmon
    product: windows
detection:
    selection_file_creation:
        EventId: 11
    filter:
        TargetFilename|startswith:
            - "C:"  # First partition/System
    condition: selection_file_creation and not filter
falsepositives:
    - Legitimate file creation on removable media
level: low
```

■ **Listing E.1** Proposed detection rule detecting file creationg on a volume with drive letter other than `C:`.

```
title: File On Removable Media Hidden
id: 48341ce3-453a-4cf7-8b97-50e75f4f3f7d
status: experimental
description: 'Detects hiding files on drives with different drive letter than
↪  C:'
references:
    - https://www.majorgeeks.com/content/page/how_to_hide_files_or_folders_usi⌟
    ↪  ng_command_prompt_or_powershell.html
    - https://learn.microsoft.com/en-us/windows-server/administration/windows-⌟
    ↪  commands/attrib
tags:
    - attack.lateral_movement
    - attack.t1091
author: Silvie Nemcova
date: 2024/04/26
logsource:
    product: windows
    category: process_creation
detection:
    selection_hide_file:
        CommandLine|contains:
            - 'attrib.exe'
            - '+h'
            - '[System.IO.FileAttributes]::Hidden'
    selection_system_drive_letter:
        TargetFilename|startswith: 'C:'
    condition: (selection_hide_file and not selection_system_drive_letter)
level: low
```

Listing E.2 Proposed detection rule detecting hiding files on a volume with drive letter other than `C:`.

Appendix F

Acronyms and Abbreviations

APT	Advanced Persistent Threat
CTI	Cyber Threat Intelligence
DCOM	Distributed Component Object Model
DFD	Data Flow Diagram
ECS	Elastic Common Schema
EQL	Event Query Language
HID	Human Interface Device
MFT	Master File Table
MMC	Microsoft Management Console
SIEM	Security Information and Event Management
SUT	System Under Test
TTP	Tactics, Techniques, Procedures

Bibliography

1. USSATH, Martin; JAEGER, David; FENG CHENG; MEINEL, Christoph. Advanced Persistent Threats: Behind the Scenes. In: *2016 Annual Conference on Information Science and Systems (CISS)* [online]. Princeton, NJ, USA: IEEE, 2016, pp. 181–186 [visited on 2023-04-19]. ISBN 978-1-4673-9457-4. Available from DOI: `10.1109/CISS.2016.7460498`.
2. *Lateral Movement, Tactic TA0008 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-12-28]. Available from: `https://attack.mitre.org/tactics/TA0008/`.
3. POLS, Paul; DOMÍNGUEZ, Francisco. The Unified Kill Chain. In: 2021. Available also from: `https://api.semanticscholar.org/CorpusID:235741204`.
4. CROWDSTRIKE. *Global Threat Report 2024*. 2024.
5. MANDIANT. *M-Trends 2018*. 2018.
6. JPCERT/CC. Detecting Lateral Movement through Tracking Event Logs (Version 2). 2017. Available also from: `https://www.jpcert.or.jp/english/pub/sr/20170612ac-ir_research_en.pdf`.
7. OVRUTSKY, Anton. *The Lowdown on Lateral Movement* [online]. Lares, 2022-02-23. [visited on 2024-01-01]. Available from: `https://www.lares.com/blog/the-lowdown-on-lateral-movement/`.
8. FAWAZ, Ahmed; BOHARA, Atul; CHEH, Carmen; SANDERS, William H. Lateral Movement Detection Using Distributed Data Fusion. In: *2016 IEEE 35th Symposium on Reliable Distributed Systems (SRDS)* [online]. Budapest, Hungary: IEEE, 2016, pp. 21–30 [visited on 2023-12-27]. ISBN 978-1-5090-3513-7. Available from DOI: `10.1109/SRDS.2016.014`.
9. *SigmaHQ/Sigma: Main Sigma Rule Repository* [online]. [N.d.]. [visited on 2024-04-24]. Available from: `https://github.com/SigmaHQ/sigma`.
10. ELASTIC. *Prebuilt Security Detection Rules — Documentation* [online]. [visited on 2024-04-02]. Available from: `https://docs-elastic-tute5nwws-elastic-dev.vercel.app/integrations/security_detection_engine`.
11. TTP-Based Hunting. [N.d.].
12. STROM, Blake; BATTAGLIA, Joseph A; KEMMERER, Michael S; MILLER, Douglas P; WAMPLER, Craig; WHITLEY, Sean M; WOLF, Ross D. Finding Cyber Threats with ATT&CK-Based Analytics. [N.d.], no. 16.
13. GLOOR, Travis; HAZARD, Eric; MERCADO, Ronald; OLSEN, Denise. ACTIVE DEFENSE CAPABILITY SET. [N.d.].
14. BLAKE E. STROM; ANDY APPLEBAUM; DOUGH MILLER; KATHRYN C. NICKELS; ADAM G. PENNINGTON; CODY THOMAS. MITRE ATT&CK®: Design and Philosophy. 2020.

15. MITRE. *MITRE ATT&CK Matrix Poster*. 2023. Available also from: `https://attack.mitre.org/docs/attack_matrix_poster_2023_april.pdf`.
16. DAN BORGES. *Remote Services, Technique T1021 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1021/`.
17. EXTRAHOP. *Exploitation of Remote Services, Technique T1210 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1210/`.
18. SWETHA PRABAKARAN; TIM MALCOMVETTER. *Internal Spearphishing, Technique T1534 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1534/`.
19. PHILIP WINTHER. *Phishing: Spearphishing Attachment, Sub-technique T1566.001 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2024-05-01]. Available from: `https://attack.mitre.org/techniques/T1566/001/`.
20. JEFF SAKOWICZ; KOBI HAIMOVICH; MARK WEE; MENACHEM GOLDSTEIN; PHILIP WINTHER; SAISHA AGRAWAL; SHAILESH TIWARY. *Phishing: Spearphishing Link, Sub-technique T1566.002 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2024-05-01]. Available from: `https://attack.mitre.org/techniques/T1566/002/`.
21. *Lateral Tool Transfer, Technique T1570 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1570/`.
22. *Remote Service Session Hijacking, Technique T1563 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1563/`.
23. JOAS ANTONIO DOS SANTOS. *Replication Through Removable Media, Technique T1091 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1091/`.
24. *Always Another Secret: Lifting the Haze on China-nexus Espionage in Southeast Asia* [online]. Mandiant. [visited on 2024-03-27]. Available from: `https://www.mandiant.com/resources/blog/china-nexus-espionage-southeast-asia`.
25. CHEN, Joey. Tropic Trooper's Back: USBferry Attack Targets Air-gapped Environments. 2020. Available also from: `https://documents.trendmicro.com/assets/Tech-Brief-Tropic-Trooper-s-Back-USBferry-Attack-Targets-Air-gapped-Environments.pdf`.
26. SHANE TULLY. *Software Deployment Tools, Technique T1072 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1072/`.
27. DAVID ROUTIN; MICHAL DIDA. *Taint Shared Content, Technique T1080 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/techniques/T1080/`.
28. *Use Alternate Authentication Material, Technique T1550 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2023-08-07]. Available from: `https://attack.mitre.org/techniques/T1550/`.
29. *Application Log, Data Source DS0015 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0015/`.
30. CENTER FOR THREAT-INFORMED DEFENSE; AUSTIN CLARK. *Command, Data Source DS0017 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0017/`.

31. CENTER FOR THREAT-INFORMED DEFENSE. *File, Data Source DS0022 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0022/`.

32. CENTER FOR THREAT-INFORMED DEFENSE. *Process, Data Source DS0009 — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/datasources/DS0009/`.

33. CENTER FOR THREAT-INFORMED DEFENSE; EXTRAHOP. *Network Traffic, Data Source DS0029 — MITRE ATT&CK®* [online]. [visited on 2023-08-18]. Available from: `https://attack.mitre.org/datasources/DS0029/`.

34. CENTER FOR THREAT-INFORMED DEFENSE. *Logon Session, Data Source DS0028 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0028/`.

35. CENTER FOR THREAT-INFORMED DEFENSE. *Network Share, Data Source DS0033 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0033/`.

36. HO, Grant; DHIMAN, Mayank; AKHAWE, Devdatta; PAXSON, Vern; SAVAGE, Stefan; VOELKER, Geoffrey M; WAGNER, David. Hopper: Modeling and Detecting Lateral Movement. [N.d.].

37. LIU, Qingyun; STOKES, Jack W.; MEAD, Rob; BURRELL, Tim; HELLEN, Ian; LAMBERT, John; MAROCHKO, Andrey; CUI, Weidong. Latte: Large-Scale Lateral Movement Detection. In: *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)* [online]. Los Angeles, CA: IEEE, 2018, pp. 1–6 [visited on 2023-12-27]. ISBN 978-1-5386-7185-6. Available from DOI: `10.1109/MILCOM.2018.8599748`.

38. CENTER FOR THREAT-INFORMED DEFENSE. *Drive, Data Source DS0016 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0016/`.

39. CENTER FOR THREAT-INFORMED DEFENSE. *Active Directory, Data Source DS0026 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0026/`.

40. CENTER FOR THREAT-INFORMED DEFENSE. *Service, Data Source DS0019 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0019/`.

41. CENTER FOR THREAT-INFORMED DEFENSE. *Named Pipe, Data Source DS0023 — MITRE ATT&CK®* [online]. [N.d.]. [visited on 2023-08-21]. Available from: `https://attack.mitre.org/datasources/DS0023/`.

42. WUNDER, John. *Getting Started with ATT&CK: Detection and Analytics* [online]. MITRE ATT&CK®, 2019-09-18. [visited on 2024-02-29]. Available from: `https://medium.com/mitre-attack/getting-started-with-attack-detection-a8e49e4960d0`.

43. LIEW, Seng Pei; IKEDA, Satoshi. Detecting Adversary Using Windows Digital Artifacts. In: *2019 IEEE International Conference on Big Data (Big Data)* [online]. Los Angeles, CA, USA: IEEE, 2019, pp. 3210–3215 [visited on 2023-09-01]. ISBN 978-1-72810-858-2. Available from DOI: `10.1109/BigData47090.2019.9006552`.

44. MITRE. *CAR Data Model.* [N.d.].

45. SORIA-MACHADO, M; ABOLINS, D; BOLDEA, C; SOCHA, K. Detecting Lateral Movements in Windows Infrastructure. [N.d.].

46. TIAN, Zhihong; SHI, Wei; WANG, Yuhang; ZHU, Chunsheng; DU, Xiaojiang; SU, Shen; SUN, Yanbin; GUIZANI, Nadra. Real-Time Lateral Movement Detection Based on Evidence Reasoning Network for Edge Computing Environment. *IEEE Transactions on Industrial Informatics* [online]. 2019, vol. 15, no. 7, pp. 4285–4294 [visited on 2023-12-27]. ISSN 1551-3203, ISSN 1941-0050. Available from DOI: `10.1109/TII.2019.2907754`.

47. ASLAN, Omer; SAMET, Refik. A Comprehensive Review on Malware Detection Approaches. *IEEE Access* [online]. 2020, vol. 8, pp. 6249–6271 [visited on 2023-07-20]. ISSN 2169-3536. Available from DOI: `10.1109/ACCESS.2019.2963724`.

48. ABOAOJA, Faitouri A.; ZAINAL, Anazida; GHALEB, Fuad A.; AL-RIMY, Bander Ali Saleh; EISA, Taiseer Abdalla Elfadil; ELNOUR, Asma Abbas Hassan. Malware Detection Issues, Challenges, and Future Directions: A Survey. *Applied Sciences* [online]. 2022, vol. 12, no. 17, p. 8482 [visited on 2023-07-20]. ISSN 2076-3417. Available from DOI: `10.3390/app12178482`.

49. KMA ALZAROONI. *Malware Variant Detection*. 2012.

50. ELASTIC. *About Detection Rules — Elastic Security Solution [8.13] — Elastic* [online]. [visited on 2024-05-01]. Available from: `https://www.elastic.co/guide/en/security/current/about-rules.html`.

51. ELASTIC. *Detection-Rules/Rules/Windows/Lateral_movement_dcom_shellwindow_shellbrowserwindow.Toml at Main · Elastic/Detection-Rules* [online]. [N.d.]. [visited on 2024-05-08]. Available from: `https://github.com/elastic/detection-rules/blob/main/rules/windows/lateral_movement_dcom_shellwindow_shellbrowserwindow.toml`.

52. FLORIAN ROTH; DAVID ANDRE. *Sigma/Rules/Windows/Builtin/Win_alert_mimikatz_keywords.Yml at Master · SigmaHQ/Sigma* [online]. 2017. [visited on 2024-05-08]. Available from: `https://github.com/SigmaHQ/sigma/blob/master/rules/windows/builtin/win_alert_mimikatz_keywords.yml`.

53. FLORIAN ROTH; OSCD.COMMUNITY; TEYMUR KHEIRKHABAROV; ZACH STANFORD; NASREDDINE BENCHERCHALI. *Sigma/Rules/Windows/Process_creation/Proc_creation_win_susp_co at Master · SigmaHQ/Sigma* [online]. 2019. [visited on 2024-05-08]. Available from: `https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process_creation/proc_creation_win_susp_copy_lateral_movement.yml`.

54. SIGMAHQ. *Explore Sigma* [online]. 2024-05-05. [visited on 2024-05-08]. Available from: `https://sigmahq.io/`.

55. @2XXEFORMYSHIRT; TEYMUR KHEIRKHABAROV. *Sigma/Rules/Windows/Process_creation/Proc_creatio at Master · SigmaHQ/Sigma* [online]. 2020. [visited on 2024-05-08]. Available from: `https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process_creation/proc_creation_win_mmc_mmc20_lateral_movement.yml`.

56. *Remote Services: Distributed Component Object Model, Sub-technique T1021.003 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2024-05-08]. Available from: `https://attack.mitre.org/techniques/T1021/003/`.

57. *Remote Services: SMB/Windows Admin Shares, Sub-technique T1021.002 - Enterprise — MITRE ATT&CK®* [online]. [visited on 2024-05-08]. Available from: `https://attack.mitre.org/techniques/T1021/002/`.

58. *Desktop Operating System Market Share Worldwide* [online]. StatCounter Global Stats. [visited on 2024-02-24]. Available from: `https://gs.statcounter.com/os-market-share/desktop/worldwide/`.

59. *Configuring Host-Only Networking* [online]. [visited on 2024-05-03]. Available from: `https://docs.vmware.com/en/VMware-Workstation-Pro/17/com.vmware.ws.using.doc/GUID-93BDF7F1-D2E4-42CE-80EA-4E305337D2FC.html`.

60. *Configuring Network Address Translation* [online]. [visited on 2024-05-03]. Available from: `https://docs.vmware.com/en/VMware-Workstation-Pro/17/com.vmware.ws.using.doc/GUID-89311E3D-CCA9-4ECC-AF5C-C52BE6A89A95.html`.

61. Windows-driver-CONTENT; DCTHEGEEK; TEDHUDEK; ALHOPPER-MSFT; JOSHBAX-MSFT. *Answer Files Overview* [online]. 2020-11-05. [visited on 2024-05-03]. Available from: `https://learn.microsoft.com/en-us/windows-hardware/customize/desktop/wsim/answer-files-overview`.

62. Windows-driver-CONTENT; THEMAR-MSFT; ELLIOTSEATTLE; DCTHEGEEK; TED-HUDEK; JOSHBAX-MSFT. *Automate Windows Setup* [online]. 2021-10-29. [visited on 2024-05-03]. Available from: `https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/automate-windows-setup?view=windows-11`.

63. ELASTIC. *Elastic Agent* [online]. Elastic. [visited on 2024-04-02]. Available from: `https://www.elastic.co/elastic-agent`.

64. ELASTIC. *Elastic Common Schema*. [N.d.]. Available also from: `https://www.elastic.co/guide/en/ecs/current/index.html`.

65. SYSINTERNALS. *Sysmon*. [N.d.]. Available also from: `https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon`.

66. OLAF HARTONG. *Sysmon Modular*. [N.d.]. Available also from: `https://github.com/olafhartong/sysmon-modular/tree/master`.

67. ELASTIC. *System — Documentation* [online]. [visited on 2024-03-03]. Available from: `https://docs-elastic-78h6lt7us-elastic-dev.vercel.app/integrations/system`.

68. ELASTIC. *Windows — Documentation* [online]. [visited on 2024-03-03]. Available from: `https://docs-elastic-78h6lt7us-elastic-dev.vercel.app/integrations/windows`.

69. ELASTIC. *Network Packet Capture — Documentation* [online]. [visited on 2024-03-03]. Available from: `https://docs-elastic-78h6lt7us-elastic-dev.vercel.app/integrations/network_traffic`.

70. ELASTIC. *Elastic Defend — Documentation* [online]. [visited on 2024-04-02]. Available from: `https://docs.elastic.co/integrations/endpoint`.

71. ELASTIC. *Elast Detection Rules*. [N.d.]. Available also from: `https://github.com/elastic/detection-rules`.

72. DAVIDJBIANCO. *Enterprise Detection & Response: The Pyramid of Pain* [online]. 2013-03-01. [visited on 2023-06-12]. Available from: `http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html`.

73. GANE, Chris; SARSON, Trish. *Structured Systems Analysis: Tools and Techniques*. McDonnell Douglas Systems Integration Company, 1977. ISBN 0930196007.

74. SCHNEIER, Bruce. Attack trees. *Dr. Dobb's journal*. 1999, vol. 24, no. 12, pp. 21–29.

75. SKOOG, Ingrid. *Ahhh, This Emulation Is Just Right: Introducing Micro Emulation Plans* [online]. MITRE-Engenuity, 2022-09-15. [visited on 2024-02-29]. Available from: `https://medium.com/mitre-engenuity/ahhh-this-emulation-is-just-right-introducing-micro-emulation-plans-7bf4c26451d3`.

76. CAT, Self; KATE, Esprit. *Becoming a Dark Knight*. [N.d.]. No. 22. Available also from: `https://www.blackhat.com/us-23/briefings/schedule/#becoming-a-dark-knight-adversary-emulation-demonstration-for-attck-evaluations-33209`.

77. ELASTIC. *EQL Search — Elasticsearch Guide [8.13] — Elastic* [online]. [visited on 2024-05-01]. Available from: `https://www.elastic.co/guide/en/elasticsearch/reference/8.13/eql.html`.

78. *YAML Ain't Markup Language (YAML™) Revision 1.2.2* [online]. [visited on 2024-04-24]. Available from: `https://yaml.org/spec/1.2.2/`.
79. SIGMA. *Sigma-Specification/Sigma_specification.Md at Main · SigmaHQ/Sigma-Specification* [online]. [visited on 2024-04-24]. Available from: `https://github.com/SigmaHQ/sigma-specification/blob/main/Sigma_specification.md`.
80. ELASTIC. *Create a Detection Rule — Elastic Security Solution [8.13] — Elastic* [online]. [visited on 2024-05-01]. Available from: `https://www.elastic.co/guide/en/security/current/rules-ui-create.html`.
81. MOSTAFA MORADIAN; NASREDDINE BENCHERCHALI. *SigmaHQ/Sigma-Rules-Validator: Validates Sigma Rules Using the JSON Schema* [online]. [N.d.]. [visited on 2024-05-07]. Available from: `https://github.com/SigmaHQ/sigma-rules-validator`.
82. AVIRA. *New Wave of PlugX Targets Hong Kong* [online]. Avira Blog, 2020-01-31. [visited on 2024-03-27]. Available from: `https://www.avira.com/en/blog/new-wave-of-plugx-targets-hong-kong`.
83. SEITZ, Anna. *Raspberry Robin Gets the Worm Early* [online]. Red Canary. [visited on 2024-03-27]. Available from: `https://redcanary.com/blog/raspberry-robin/`.
84. LU, Hongyi; WU, Yechang; LI, Shuqing; LIN, You; ZHANG, Chaozu; ZHANG, Fengwei. BADUSB-C: Revisiting BadUSB with Type-C. [N.d.].
85. SDWHEELER. *About Logging Windows - PowerShell* [online]. 2024-01-03. [visited on 2024-04-28]. Available from: `https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_windows?view=powershell-7.4`.
86. SDWHEELER. *About Group Policy Settings - PowerShell* [online]. 2023-08-17. [visited on 2024-04-28]. Available from: `https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_group_policy_settings?view=powershell-7.4`.
87. *Fleet and Elastic Agent Overview — Fleet and Elastic Agent Guide [8.13] — Elastic* [online]. [visited on 2024-04-28]. Available from: `https://www.elastic.co/guide/en/fleet/8.13/fleet-overview.html`.
88. *Elastic Integrations — Documentation* [online]. [visited on 2024-04-28]. Available from: `https://docs-elastic-mpu84froj-elastic-dev.vercel.app/integrations/`.
89. *Atomic-Red-Team/Atomics/T1091/T1091.Md at Master · Redcanaryco/Atomic-Red-Team* [online]. GitHub. [visited on 2024-04-13]. Available from: `https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1091/T1091.md`.
90. ELASTIC. *Detection-Rules Initial_access_execution_from_removable_media.Toml* [online]. [N.d.]. [visited on 2024-04-30]. Available from: `https://github.com/elastic/detection-rules/blob/rules/windows/initial_access_execution_from_removable_media.toml`.
91. KEITH WRIGHT. *Sigma/Rules/Windows/Builtin/Security/Win_security_external_device.Yml at 39db80478e36599be3b25d9cdbd2c168815c4ea3 · SigmaHQ/Sigma* [online]. [N.d.]. [visited on 2024-04-30]. Available from: `https://github.com/SigmaHQ/sigma/blob/39db80478e36599be3b25d9cdbd2c168815c4ea3/rules/windows/builtin/security/win_security_external_device.yml#L11`.
92. ELASTIC. *Detection-Rules Initial_access_exfiltration_first_time_seen_usb.Toml* [online]. [N.d.]. [visited on 2024-04-30]. Available from: `https://github.com/elastic/detection-rules/blob/rules/windows/initial_access_exfiltration_first_time_seen_usb.toml`.