Challenge: Akira Lab

Platform: CyberDefenders

Category: Endpoint Forensics

Difficulty: Medium

Tools Used: Volatility 3, MemProcFS, EvtxECmd, Timeline Explorer, Strings, Notepad++

Summary: This lab involved investigating a memory image taken from a Windows machine that was infected with Akira ransomware. The primary tools used were Volatility 3, MemProcFS, EvtxECmd, and Timeline Explorer. I found this lab really enjoyable, and highly recommend it for those who enjoy memory forensics.

Scenario: As a member of the DFIR team, you're tasked with investigating a ransomware attack involving Akira ransomware that has impacted critical systems. You've been provided with a memory dump from one of the compromised machines. Your goal is to analyze the memory for indicators of compromise, trace the ransomware's entry point, and identify any malicious activity to assess the incident and guide the response strategy.

While analyzing the memory dump, identifying the compromised machine's network domain affiliation is a crucial step in understanding the attack's scope. What is the domain to which the infected machine is joined?

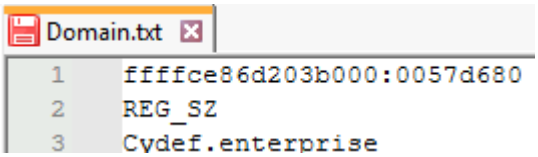
Before we dive into using Volatility 3 to find what domain the machine is joined to, I am going to process the memory dump usising MemProcFS, reason being that it takes a decent amount of time for MemProcFS to extract all the information from the memory dump. MemProcFS is a tool that enables you to view memory images as files in a virtual file system.

- `.\memprocfs.exe -f "C:\Users\Administrator\Desktop\Start Here\Artifacts\memory.dmp" -forensic 1`

This command mounts the output to a drive, in my case the drive letter assigned was M. Within this drive, you can find a bunch of important forensic information, including registry hives, processes, services, scheduled tasks, etc. To find the domain this system is joined to by using MemProcFS, navigate to the following location:

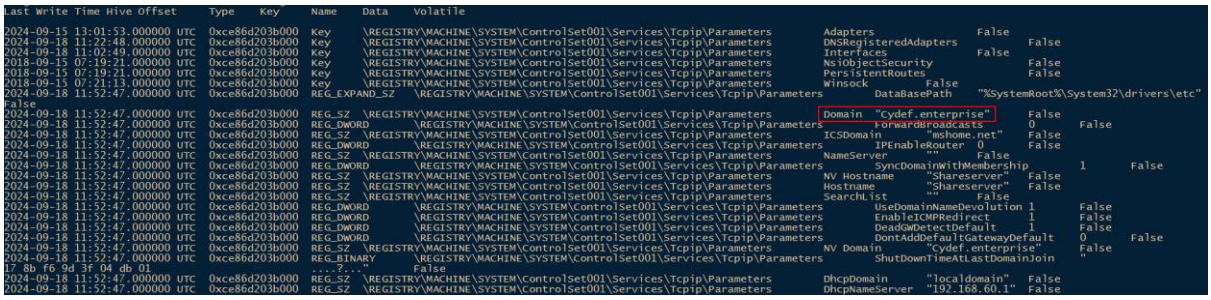
- `M:\registry\HKLM\SYSTEM\ControlSet001\Services\Tcpip\Parameters`

Within this folder, you will find a file called Domain.txt:



Alternatively, we can use Volatility 3 and the windows.registry.printkey plugin to print the Parameters key located at:

- HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
- python .\vol.py -f "C:\Users\Administrator\Desktop\Start Here\Artifacts\memory.dmp" windows.registry.printkey --offset 0xce86d203b000 --key ControlSet001\Services\Tcpip\Parameters



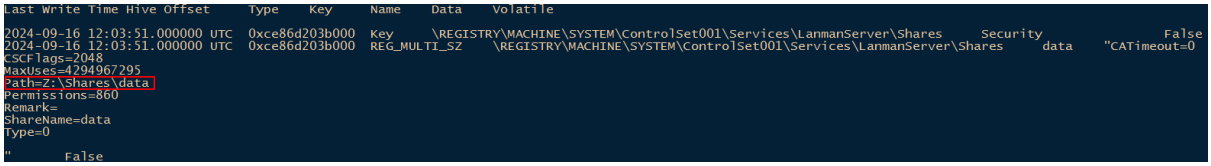
As you can see, both methods return Cydef.enterprise as the domain this machine is joined to.

Answer: Cydef.enterprise

Identifying the shared file path accessed by the attacker is crucial for understanding the scope of the breach and determining which files may have been compromised. What is the local path of the file that was shared on the file server?

Let's start by identifying the file shares on this computer, we can do so by using the windows.registry.printkey plugin in Volatility 3:

- python .\vol.py -f "C:\Users\Administrator\Desktop\Start Here\Artifacts\memory.dmp" windows.registry.printkey --offset 0xce86d203b000 --key ControlSet001\Services\Lanmanserver\Shares



The registry key located at HKLM\SYSTEM\CurrentControlSet\Services\Lanmanserver\Shares stores information about Windows shared folders. Within the LanmanServer\Shares subkey you can find the shares on the machine. In this case, we can see that Z:\Shares\data is the path on disk that is being shared.

Alternatively, you can find the same information in the MecProcFS output located at:

- M:\registry\HKLM\SYSTEM\ControlSet001\Services\LanmanServer\Shares

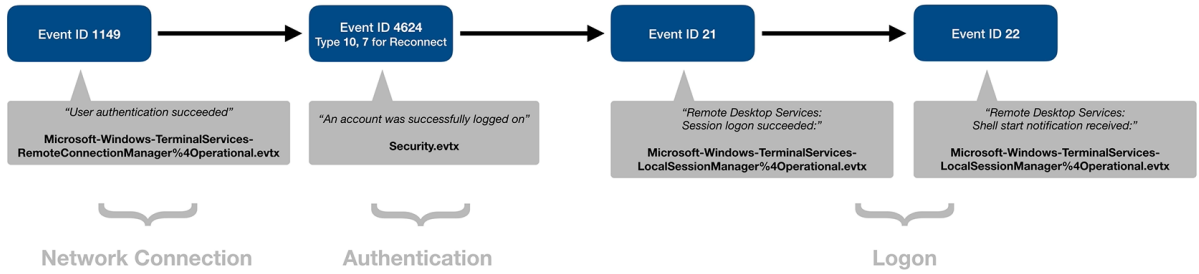
In this directory is a file called data.txt:

```
ffffce86d203b000:00522060
REG_MULTI_SZ
CATimeout=0
CSCFlags=2048
MaxUses=4294967295
Path=Z:\Shares\data
Permissions=860
Remark=
ShareName=data
Type=0
```

Answer: z:\Shares\data

Identifying the source of failed RDP connection attempts is crucial for tracing the compromised machine and analyzing the attacker's behavior. What is the IP address of the machine that attempted to connect to the file server?

To identify the source of failed RDP connection attempts, we can look through multiple Windows event log files to correlate events. A great chart created by 13Cubed details the key logs that are generated for RDP connections:



You can find the evtx files in the output of MemProcFS:

- `M:\misc\eventlog`

In my case, I am going to use three event logs: ffffde8561a391f0-Microsoft-Windows-TerminalServices-RemoteConnectionManager%4Operational.evtx, ffffde8561a38bb0-Microsoft-Windows-TerminalServices-LocalSessionManager%4Operational.evtx, and ffffde85619117d0-Security.evtx. To parse these event logs, I am going to use a tool called EvtxECmd:

- `.\EvtxECmd.exe -d . --csv . --csvf "rdp_events_out.csv"`
  - -d is used to recursively parse event logs in a directory, in this case the full stop specifies the current directory.
  - --csv specifies to output the result in csv format.
  - --csvf specifies the output filename.

We can then use a tool called Timeline Explorer to look through the CSV file. Let's start by filtering for Event ID 1149 (User authentication succeeded). This log is generated when someone successfully executes an RDP network connection to the target machine.

Map Description	User Name	Remote Host
RDP network connection established	CYDEF\Administrator	192.168.60.129

On September 18th, at 05:58 (2024-09-18 05:58:36), an RDP network connection was established from 192.168.60.129:

This machine also had Sysmon enabled, we can parse the Sysmon logs like as follows:

- `.\EvtxECmd.exe -f ".\ffffde8562d711d0-Microsoft-Windows-Sysmon%4Operational.evtx" --csv . --csvf sysmon_out.csv`

We can then filter for Event ID 3 (network connection) and focus on logs with the RuleName RDP:

Payload Data2	Payload Data3	Payload Data4	Payload Data5	Payload Data6
RDP				
RuleName: RDP	SourceHostname: -	SourceIp: 192.168.60.129	DestinationHostname: Shareserver.Cydef.enterprise	DestinationIp: 192.168.60.128
RuleName: RDP	SourceHostname: -	SourceIp: 192.168.60.129	DestinationHostname: Shareserver.Cydef.enterprise	DestinationIp: 192.168.60.128

At On September 18th, at 11:34 (2024-09-18 11:34:57) and 11:36:01, 192.168.60.129 was observed attempting to connect to the file server via RDP.

Answer: 192.168.60.129

Identifying the process name of the attacker's tool is key to tracking their actions. What is the process name of the tool used by the attacker to remotely execute commands and perform malicious activities on the compromised FileServer?

Tip: Check both active and terminated or hidden processes in the memory capture.

To identify the process name of the threat actors' tool, we can use the psscan plugin in Volatility 3. This plugin can identify processes, even hidden ones, that were present at the time of the memory capture.

- `python .\vol.py -f "C:\Users\Administrator\Desktop\Start Here\Artifacts\memory.dmp" windows.psscan`

```
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

4       0       System  0xde855c463200  196     -       N/A     False   2024-09-18 10:59:41.000000 UTC  N/A     Disabled
1104    676     PSEXESVC.exe    0xde855c482080  0       -       0       False   2024-09-18 12:01:30.000000 UTC  2024-09-18 12:01:36.000000 UTC  Disabled
88      4       Registry        0xde855c50a080  4       -       N/A     False   2024-09-18 10:59:36.000000 UTC  N/A     Disabled
2288    676     spoolsv.exe     0xde855c5a4080  10      -       0       False   2024-09-18 10:59:44.000000 UTC  N/A     Disabled
4068    676     msdtc.exe       0xde855c77c240  9       -       0       False   2024-09-18 10:59:46.000000 UTC  N/A     Disabled
320     4       smss.exe        0xde855fd920c0  2       -       N/A     False   2024-09-18 10:59:41.000000 UTC  N/A     Disabled
428     420     csrss.exe       0xde855feeb140  11      -       0       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
924     676     svchost.exe     0xde8560ed3080  10      -       0       False   2024-09-18 10:59:43.000000 UTC  N/A     Disabled
532     420     wininit.exe     0xde8560eda080  1       -       0       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
676     532     services.exe    0xde8560edd140  8       -       0       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
540     524     csrss.exe       0xde8560edf140  11      -       1       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
4104    804     unsecapp.exe    0xde8560f10080  2       -       0       False   2024-09-18 11:07:38.000000 UTC  N/A     Disabled
4720    804     backgroundTask  0xde8560f11080  7       -       1       False   2024-09-18 12:01:42.000000 UTC  N/A     Disabled
600     524     winlogon.exe    0xde8560f19080  4       -       1       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
684     532     lsass.exe       0xde8560f43140  8       -       0       False   2024-09-18 10:59:42.000000 UTC  N/A     Disabled
804     676     svchost.exe     0xde8560f80080  17      -       0       False   2024-09-18 10:59:43.000000 UTC  N/A     Disabled
832     532     fontdrvhost.ex  0xde8561620140  5       -       0       False   2024-09-18 10:59:43.000000 UTC  N/A     Disabled
836     600     fontdrvhost.ex  0xde8561622140  5       -       1       False   2024-09-18 10:59:43.000000 UTC  N/A     Disabled
1016    600     dwm.exe 0xde85616a2080  13      -       1       False   2024-09-18 10:59:43.000000 UTC  N/A     Disabled
```

Immediately I notice PSEXESVC.exe. PSEXESVC.exe is a core component of the PsExec tool, used for remotely executing processes on computers. Whilst this tool has legitimate uses, it is often leveraged by threat actors to remotely execute programs.

Answer: PSEXESVC.exe

Identifying the attacker's initial commands reveals their intentions and the level of access they gained. What was the first command executed remotely to begin system enumeration?

Navigating back to the parsed Sysmon logs, we can filter for event ID 1 (process create) and look for logs where PSEXESVC.exe is the parent process:

Payload Data4	Payload Data5	Payload Data6	Executable Info
PSEXESVC.exe			
ParentProcess: C:\Windows\PSEXESVC.exe	ParentProcessID: 4484, ParentProcessGUID: 8eef59…	ParentCommandLine: C:\Windows\PSEXESVC.exe	"tasklist"

On September 18th, at 11:36 (2024-09-18 11:36:40), the threat actor was observed using PsExec to execute the tasklist command. Tasklist is used to display a list of all currently running processes on the computer.

Answer: tasklist

Understanding how the attacker disabled security measures is key to assessing how they gained persistence and weakened the system's defenses. The attacker used a remote execution tool, which generates a different Process ID (PID) for each command executed. What is the Process ID (PID) of the first command used to turn off Windows Defender?

Continuing with exploring the process creation Sysmon logs, we can see that on September 18th, at 11:40 (2024-09-18 11:40:31) the threat actor used PowerShell to disable Windows Defender real time monitoring:

```
ParentCommandLine: C:\Windows\PSEXESVC.exe "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" Set-MpPreference -DisableRealtimeMonitoring 1
```

You can find the process ID (PID) under the Payload Data1 column:

```
ProcessID: 5344
```

Answer: 5344

Identifying changes to the system's registry is essential for understanding how the attacker disabled security features, allowing malicious actions to proceed undetected. In an attempt to disable Windows Defender, the attacker modified a specific registry value. What is the name of the registry value that was added or modified under HKLM\SOFTWARE\Policies\Microsoft\Windows Defender?

Not long after the threat actor disabled real time monitoring, on September 18th, at 11:42 (2024-09-18 11:42:01) the reg add command was used to disable the Windows Defender AntiSpyware feature:

```
"reg" add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender" /v DisableAntiSpyware /t REG_DWORD /d 1 /f
```

Answer: DisableAntiSpyware

Understanding how the attacker leveraged specific system files is crucial, as it can reveal their methods for accessing sensitive data and escalating privileges. What DLL file did the attacker use in the PowerShell command to dump the targeted process for further exploitation?

On September 18th, at 11:45 (2024-09-18 11:45:06), the threat actor was seen executing the MiniDump function within comsvcs.dll to dump LSASS memory.

```
"powershell.exe" -command "rundll32.exe c:\Windows\System32\comsvcs.dll, MiniDump ((Get-Process lsass).Id) C:\windows\temp\lsass.dmp full"
```

For a full rundown, this command invokes the MiniDump function from comsvcs.dll using rundll32.exe. PowerShell then retrieves the PID of LSASS (Local Security Authority Subsystem Service), which handles Windows authentication and holds credentials in memory. The output is saved to lsass.dmp and was likely used to dump credentials.

Answer: comsvcs.dll

Investigating the creation of new accounts is crucial for identifying how the attacker maintains unauthorized access to the system. To establish persistent access, the attacker created a new user account on the compromised system. What is the name of the account that the attacker created?

Each time a user is created on Windows, Event ID 4720 is generated within the Security logs. If you filter for this event ID, we can see that on September 18th, at 11:51 (2024-09-18 11:51:41), the Administrator account (which we know to be compromised) was used to create a user called ITadmin_2:

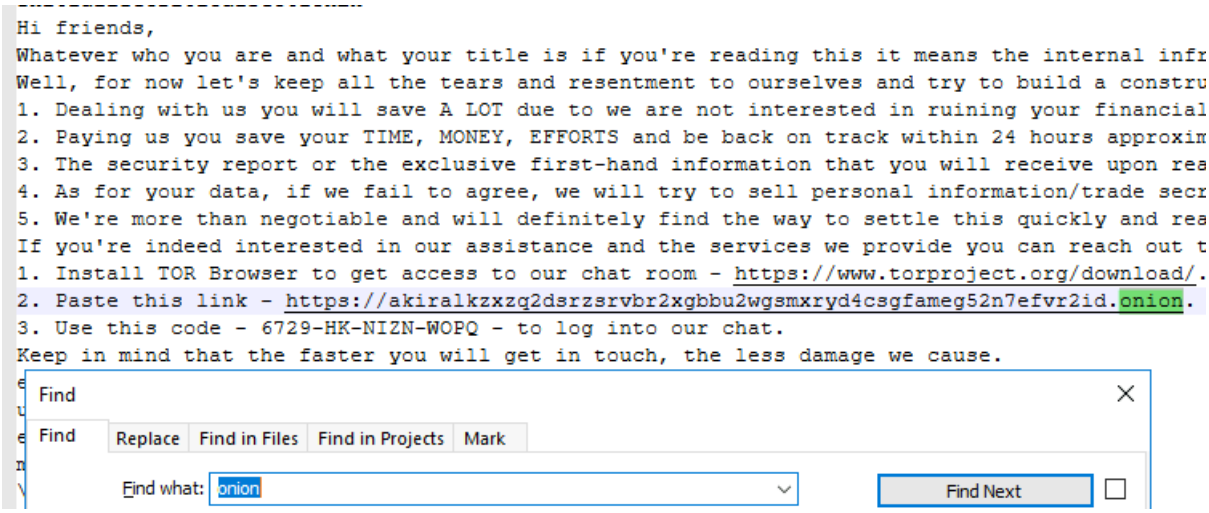
```
CYDEF\Administrator (S-1-5-21-2547355392-3774477586-307…    Target: SHARESERVER\ITadmin_2 (S-1-5-21-94776327-2305441286-1715799293-1000)
```

Answer: ITadmin_2

Identifying the URL in the ransom note is vital for understanding the attacker's communication and data exposure threats. The attacker included a link to their blog where stolen data would be published if negotiations fail. What is the URL provided for communication and accessing the attacker's chat?

Typically threat actors, especially ransomware groups, provide a .onion link in their ransom note. To find this URL, we can use the strings command against the memory dump and search for the .onion domain:

- `strings.exe .\memory.dmp > strings.txt`



In the above image, you can clearly see the entire Akira ransom note, including the .onion chat link.

Answer: https://akiralkzxzq2dsrzsrvbr2xgbbu2wgsmxryd4csgfameg52n7efvr2id.onion