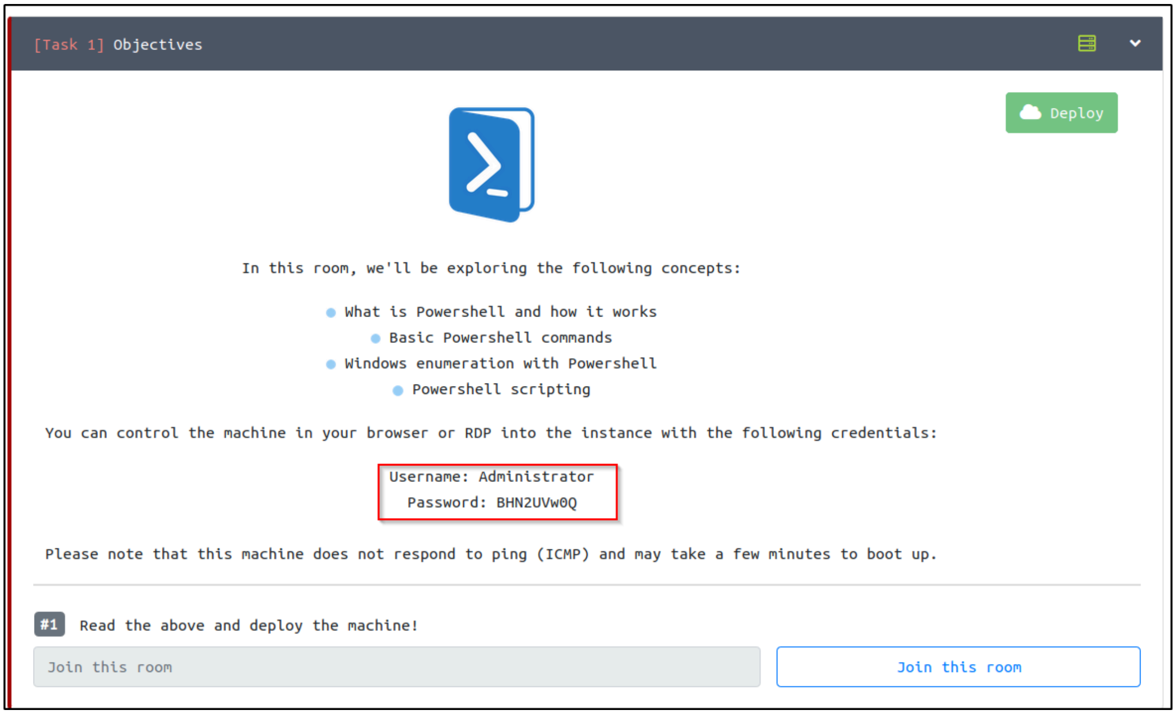
Hacking with powershell

☰ Tags

1. Objectives:

In This room, we'll be exploring the following Concepts

- What is Powershell and how it works.
- Basic of Powershell commands.
- Windows enumerations with powershell



[Task 1] Objectives

In this room, we'll be exploring the following concepts:

- What is Powershell and how it works
- Basic Powershell commands
- Windows enumeration with Powershell
- Powershell scripting

You can control the machine in your browser or RDP into the instance with the following credentials:

Username: Administrator
Password: BHN2UVw0Q

Please note that this machine does not respond to ping (ICMP) and may take a few minutes to boot up.

#1 Read the above and deploy the machine!

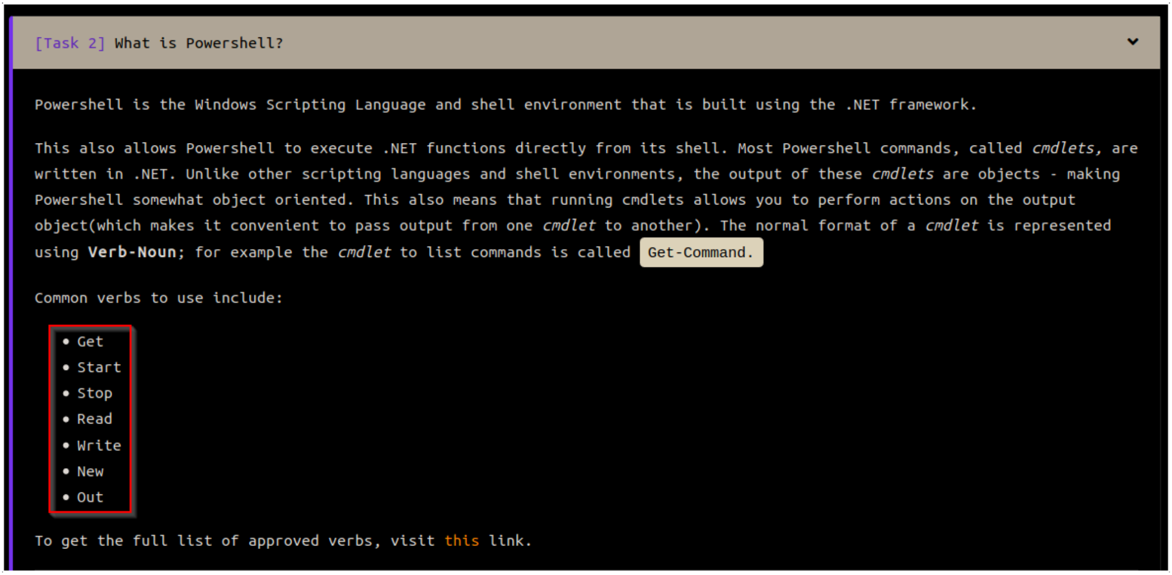
2. What is Powershell

Powershell is the Scripting language and shell environment that is built using the .NET framework.

This also allows Powershell to execute .NET function directly from its shell. Most Powershell commands called cmdlets, are written in .NET . Unlike other scripting language and shell environments, the output of these cmdlets are objects -

making Powershell somewhat Object Oriented. This also means that running cmdlets also means that allows you to perform actions on the output.

object (which makes it convenient to pass output from one cmdlets to another). The normal cmdlets is represents using Verb-noun. for example the cmdlets to list commands is called Get-Command.



[Task 2] What is Powershell?

Powershell is the Windows Scripting Language and shell environment that is built using the .NET framework.

This also allows Powershell to execute .NET functions directly from its shell. Most Powershell commands, called *cmdlets*, are written in .NET. Unlike other scripting languages and shell environments, the output of these *cmdlets* are objects - making Powershell somewhat object oriented. This also means that running cmdlets allows you to perform actions on the output object(which makes it convenient to pass output from one *cmdlet* to another). The normal format of a *cmdlet* is represented using **Verb-Noun**; for example the *cmdlet* to list commands is called `Get-Command.`

Common verbs to use include:

- Get
- Start
- Stop
- Read
- Write
- New
- Out

To get the full list of approved verbs, visit this link.

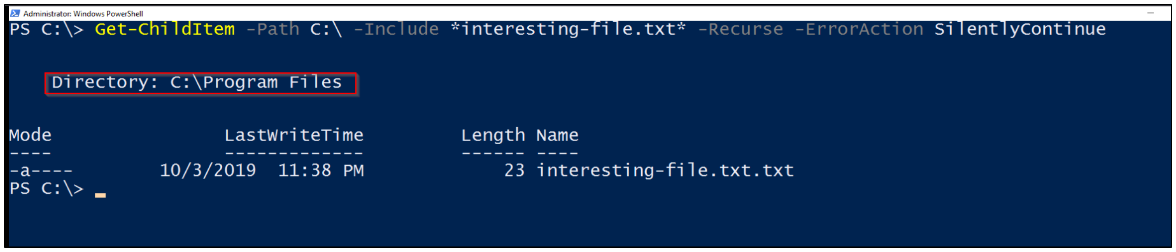
- command to get help about the particular cmdlets is

```
Get-Help
```

3. Basic Powershell command

3.1 Location of file "interesting-file.txt"

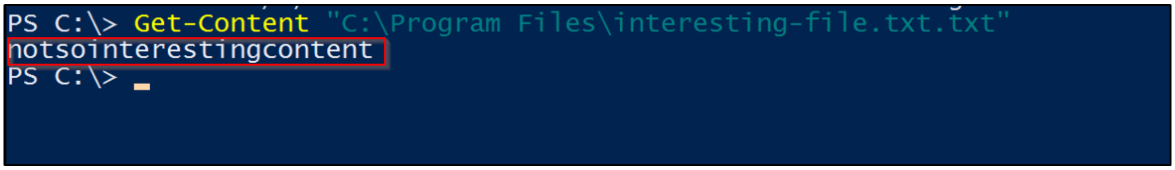
```
PS C:\> Get-ChildItem -Path C:\ *interesting-file.txt* -Recurse -ErrorAction SilentlyContinue
```



```
PS C:\> Get-ChildItem -Path C:\ -Include *interesting-file.txt* -Recurse -ErrorAction SilentlyContinue

    Directory: C:\Program Files

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/3/2019  11:38 PM             23 interesting-file.txt.txt
PS C:\> _
```

3.2 Specify the content of the given file

```
PS C:\> Get-Content "C:\Program Files\interesting-file.txt.txt"
```



```
PS C:\> Get-Content "C:\Program Files\interesting-file.txt.txt"
notsointerestingcontent
PS C:\> _
```

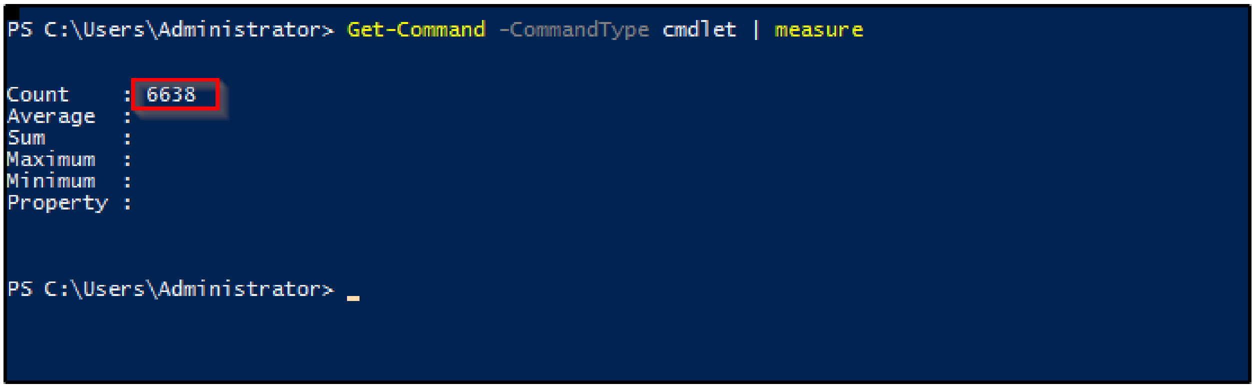
3.3 How many cmdlets are installed

@Prasant Adhikari Dai, i can't get this answer right. don't know where am i missing. please guide me.

@ankit Karn `get-command -commandtype cmdlet`

actually the command you are running also gives me the same value. looks like somehow you have a few extra commands installed. it might just help to run this on a freshly rebooted machine.

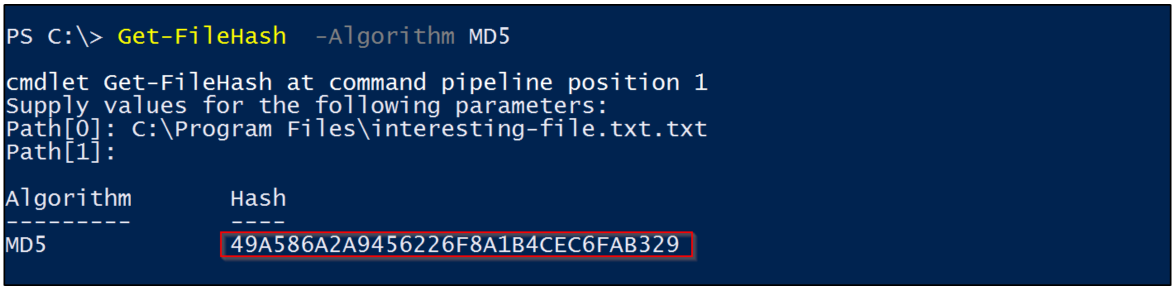
```
PS C:\> Get-Command -CommandType cmdlet | measure
```



```
PS C:\Users\Administrator> Get-Command -CommandType cmdlet | measure

Count    : 6638
Average  :
Sum      :
Maximum  :
Minimum  :
Property :


PS C:\Users\Administrator> _
```

3.4 md5 hash of interesting-file.txt

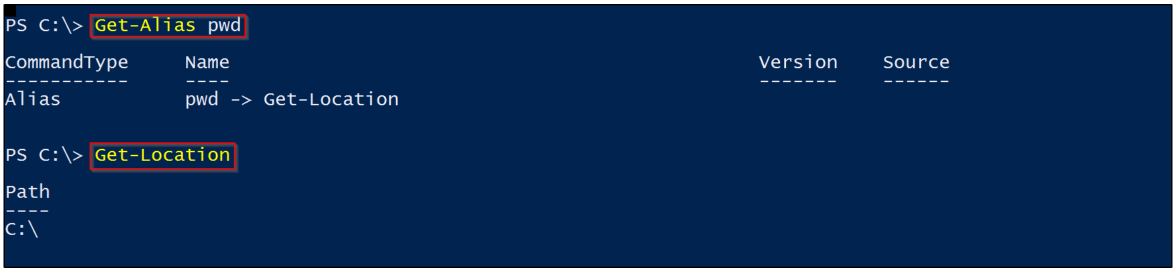
```
PS C:\ Get-FileHash -Algorithm MD5
Path[0]: C:\Program Files\intersting-file.txt.txt
```



```
PS C:\> Get-FileHash  -Algorithm MD5

cmdlet Get-FileHash at command pipeline position 1
Supply values for the following parameters:
Path[0]: C:\Program Files\interesting-file.txt.txt
Path[1]:

Algorithm       Hash
---------       ----
MD5             49A586A2A9456226F8A1B4CEC6FAB329
```

3.5 Command to get current working directory

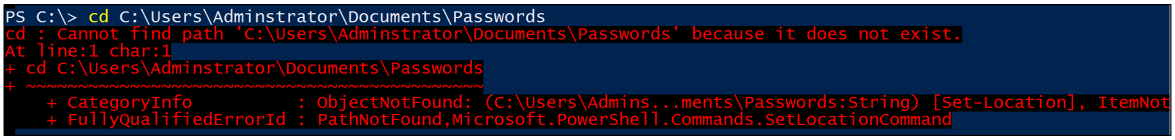
```
PS C:\> Get-Alias pwd
PS C:\>Get-Location
```



```
PS C:\> Get-Alias pwd

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Alias           pwd -> Get-Location


PS C:\> Get-Location

Path
----
C:\
```

3.6 Does the path "C:\Users\Administrator\Documents\Passwords" exist

```
ps C:\> cd C:\Users\Administrator\Documents\Password
```

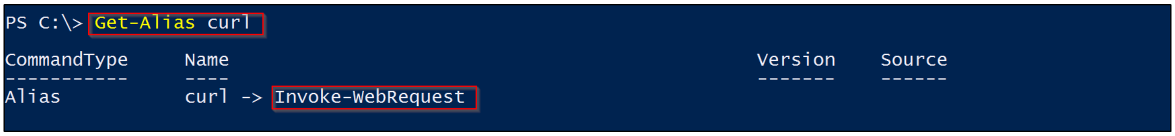


```
PS C:\> cd C:\Users\Adminstrator\Documents\Passwords
cd : Cannot find path 'C:\Users\Adminstrator\Documents\Passwords' because it does not exist.
At line:1 char:1
+ cd C:\Users\Adminstrator\Documents\Passwords
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\Users\Admins...ments\Passwords:String) [Set-Location], ItemNot
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand
```

> Since the command cannot execute, so the path doesn't exist.

3.7 Command used to make a request to a web server

```
PS C:\> Get-Alias curl
PS C:\> Invoke-WebRequest
```

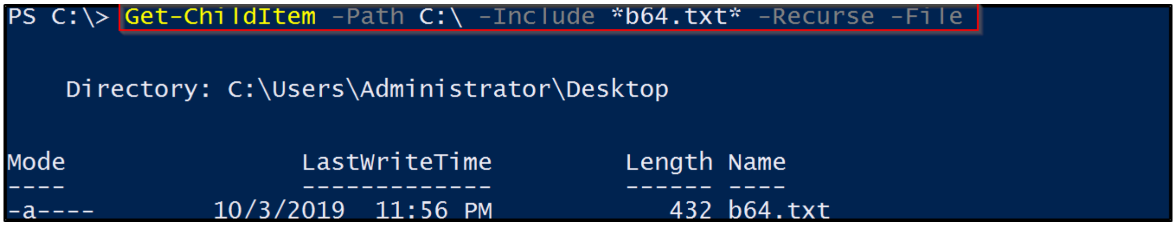


```
PS C:\> Get-Alias curl

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Alias           curl -> Invoke-WebRequest
```

3.8 Base64 decode the file b64.txt on windows

First we need to find out the location of the file b64.txt

```
PS C:\> Get-ChildItem -Path C:\ -Include *b64.txt* -Recurse -File
```



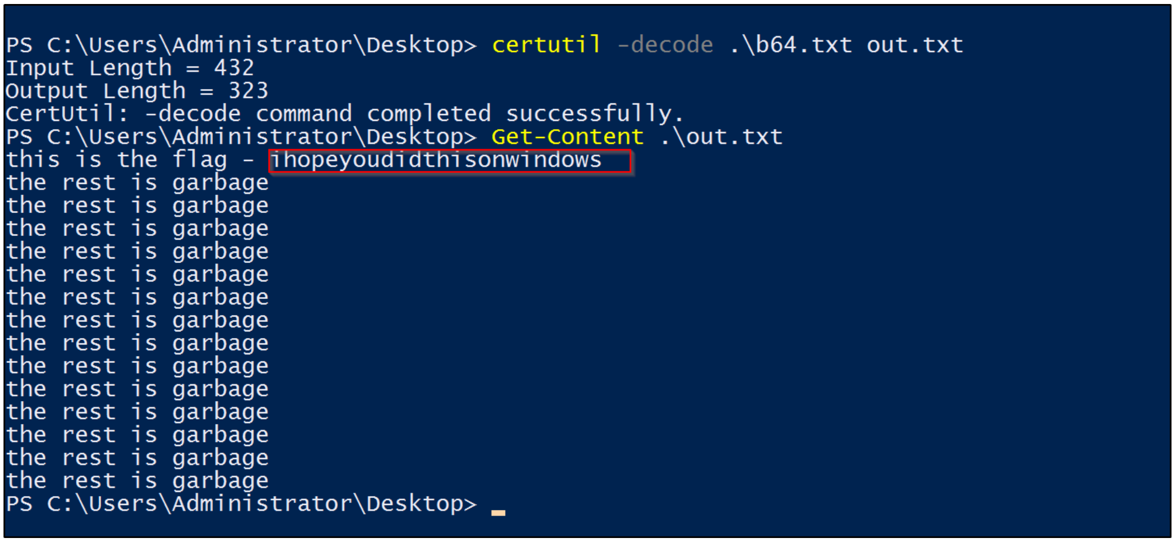
```
PS C:\> Get-ChildItem -Path C:\ -Include *b64.txt* -Recurse -File

    Directory: C:\Users\Administrator\Desktop

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        10/3/2019  11:56 PM            432 b64.txt
```

Then we use a certutil function to decode the file

```
PS C:\> certutil -decode b64.txt out.txt
```

And read the content of the output file I.e. out.txt

```
PS C:\> Get-Content out.txt
```

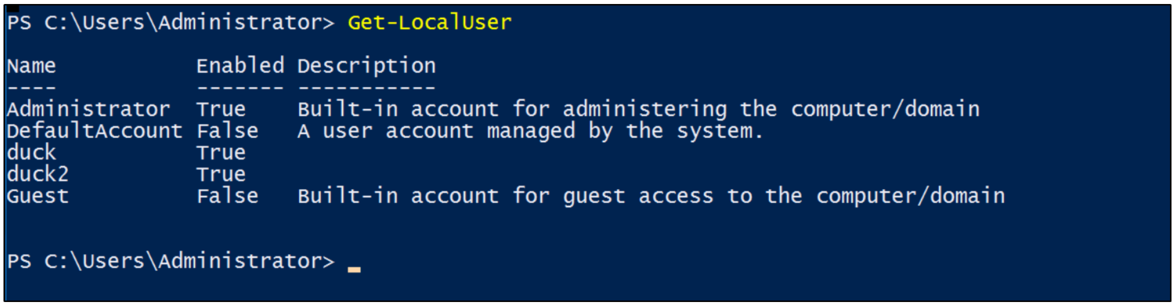


```
PS C:\Users\Administrator\Desktop> certutil -decode .\b64.txt out.txt
Input Length = 432
Output Length = 323
CertUtil: -decode command completed successfully.
PS C:\Users\Administrator\Desktop> Get-Content .\out.txt
this is the flag - ihopeyoudidthisonwindows
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
the rest is garbage
PS C:\Users\Administrator\Desktop>
```

Enumeration

4.1 How many users are there on Machine

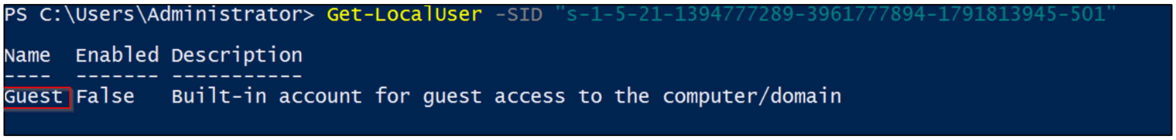
```
PS C:\> Get-LocalUser
```



```
PS C:\Users\Administrator> Get-LocalUser

Name           Enabled Description
----           ------- -----------
Administrator  True    Built-in account for administering the computer/domain
DefaultAccount False   A user account managed by the system.
duck           True
duck2          True
Guest          False   Built-in account for guest access to the computer/domain

PS C:\Users\Administrator>
```

4.2 Which local user does this SID (S-1-5-21-1394777289-3961777894-1791813945-501) belong to?

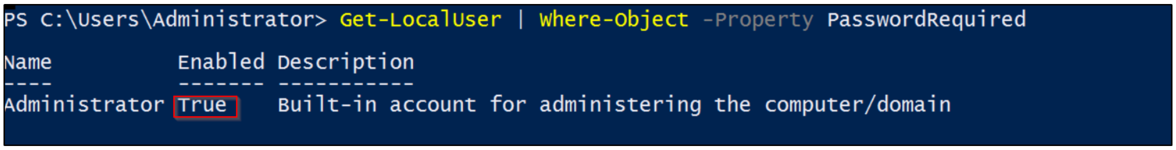
```
PS C:\> Get-LocalUser -SID "S-1-5-21-1394777289-3961777894-1791813945-501"
```



```
PS C:\Users\Administrator> Get-LocalUser -SID "s-1-5-21-1394777289-3961777894-1791813945-501"

Name  Enabled Description
----  ------- -----------
Guest False   Built-in account for guest access to the computer/domain
```

4.3 How many users have their Password required to false

```
PS C:\> Get-LocalUser | where-Object -Property PasswordRequired
```

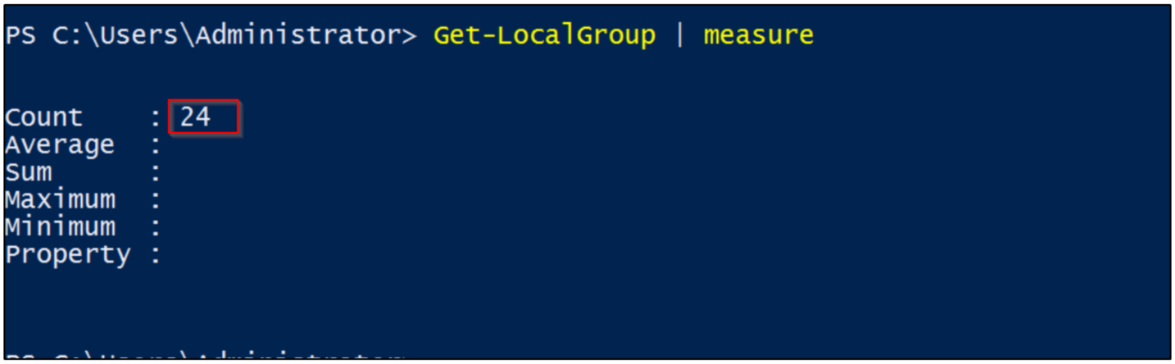


```
PS C:\Users\Administrator> Get-LocalUser | Where-Object -Property PasswordRequired

Name          Enabled Description
----          ------- -----------
Administrator True    Built-in account for administering the computer/domain
```

Here Out of 5 Accounts Only one Account is set for Password required True:

4.4 How many Local Group Exist

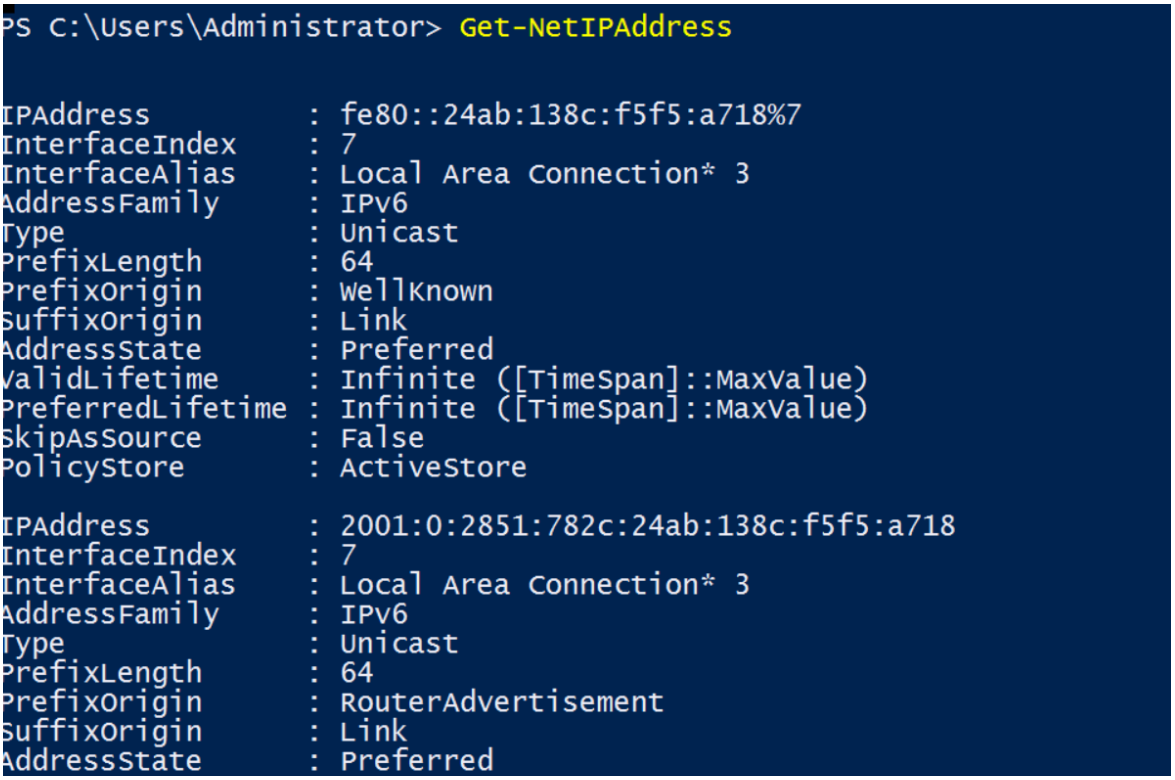
```
PS C:\ Get-LocalGroup | measure
```



```
PS C:\Users\Administrator> Get-LocalGroup | measure


Count    : 24
Average  :
Sum      :
Maximum  :
Minimum  :
Property :



PS C:\Users\Administrator>
```

4.5 Command to get the IP address info

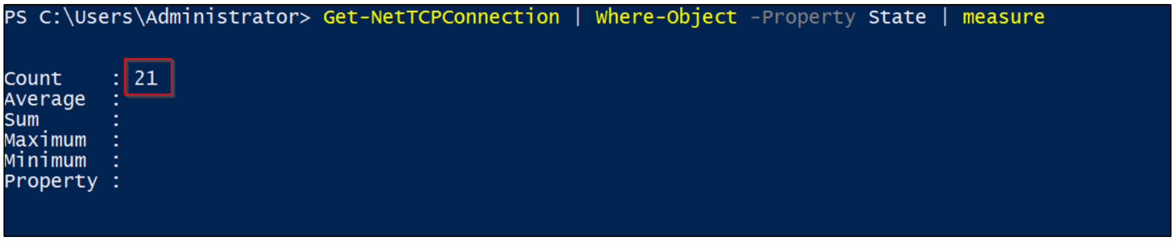
```
PS C:\> Get-NetIPAddress
```



```
PS C:\Users\Administrator> Get-NetIPAddress


IPAddress         : fe80::24ab:138c:f5f5:a718%7
InterfaceIndex    : 7
InterfaceAlias    : Local Area Connection* 3
AddressFamily     : IPv6
Type              : Unicast
PrefixLength      : 64
PrefixOrigin      : WellKnown
SuffixOrigin      : Link
AddressState      : Preferred
ValidLifetime     : Infinite ([TimeSpan]::MaxValue)
PreferredLifetime : Infinite ([TimeSpan]::MaxValue)
SkipAsSource      : False
PolicyStore       : ActiveStore

IPAddress         : 2001:0:2851:782c:24ab:138c:f5f5:a718
InterfaceIndex    : 7
InterfaceAlias    : Local Area Connection* 3
AddressFamily     : IPv6
Type              : Unicast
PrefixLength      : 64
PrefixOrigin      : RouterAdvertisement
SuffixOrigin      : Link
AddressState      : Preferred
```

4.6 How many Ports are listed as listening

```
PS C:\> Get-NetTCPConnection | where-Object -Property State |measure
```

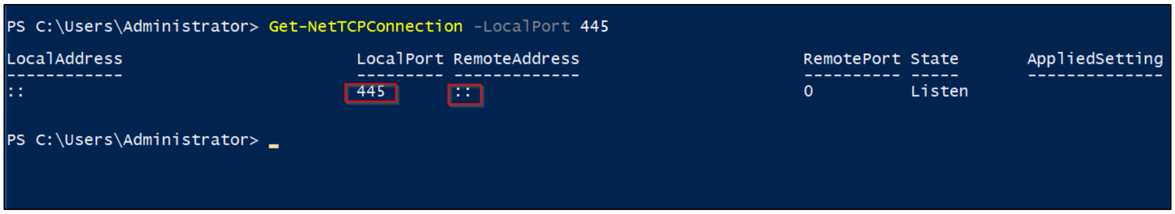


```
PS C:\Users\Administrator> Get-NetTCPConnection | Where-Object -Property State | measure


Count    : 21
Average  :
Sum      :
Maximum  :
Minimum  :
Property :
```

here all the ports are listening except one. so the total no of listening is 20

4.7 What is the remote address of local port listening on port 445

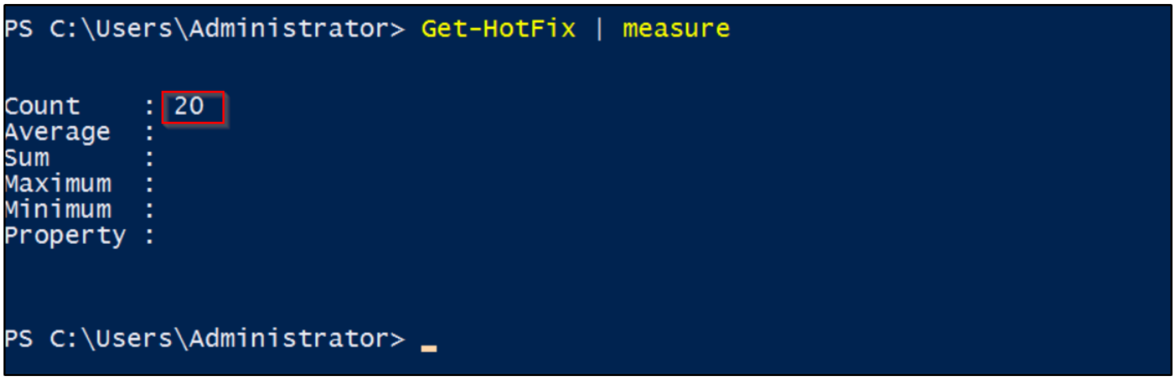
```
PS C:\> Get-NetTCPConnection -localPort 445
```



```
PS C:\Users\Administrator> Get-NetTCPConnection -LocalPort 445

LocalAddress                        LocalPort RemoteAddress                          RemotePort State      AppliedSetting
------------                        --------- -------------                          ---------- -----      --------------
::                                  445       ::                                     0          Listen

PS C:\Users\Administrator> _
```

4.8 How Many patches have been applied

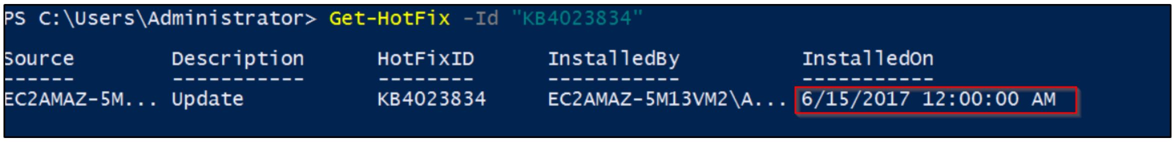
```
PS C:\> Get-HotFix | measure
```



```
PS C:\Users\Administrator> Get-HotFix | measure


Count    : 20
Average  :
Sum      :
Maximum  :
Minimum  :
Property :



PS C:\Users\Administrator> _
```

4.9 When was the patch with ID KB4023834 installed

```
PS C:\> Get-HotFix -Id "KB4023834"
```

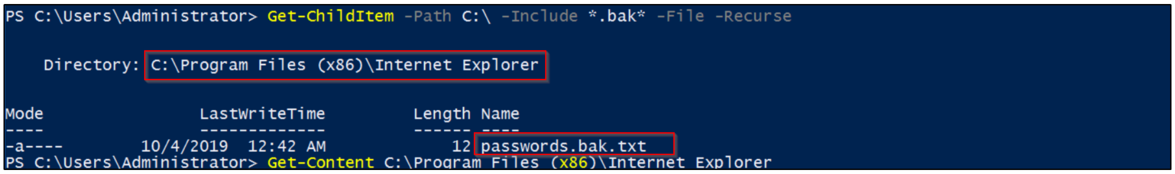


```
PS C:\Users\Administrator> Get-HotFix -Id "KB4023834"

Source        Description      HotFixID      InstalledBy          InstalledOn
------        -----------      --------      -----------          -----------
EC2AMAZ-5M... Update           KB4023834     EC2AMAZ-5M13VM2\A... 6/15/2017 12:00:00 AM
```

4.10 Find the contents of a backup file.

Find what is and where is backup file

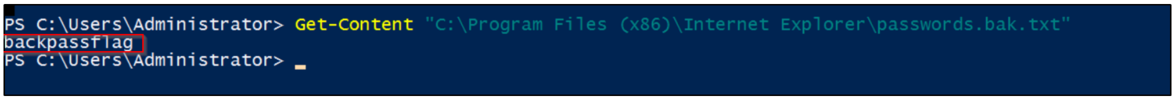
```
PS C:\> Get-ChildItem -Path C:\ -Include *.bak* -File -Recurse
```



```
PS C:\Users\Administrator> Get-ChildItem -Path C:\ -Include *.bak* -File -Recurse


    Directory: C:\Program Files (x86)\Internet Explorer


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        10/4/2019  12:42 AM             12 passwords.bak.txt
PS C:\Users\Administrator> Get-Content C:\Program Files (x86)\Internet Explorer
```

Now, Read the passwords.bak.txt

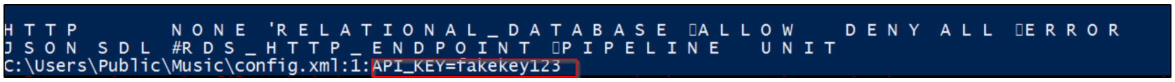
```
PS C:\> Get-Content "C:\Program Files (x86)\Internet Explorer\passwords.bak.txt"
```



```
PS C:\Users\Administrator> Get-Content "C:\Program Files (x86)\Internet Explorer\passwords.bak.txt"
backpassflag
PS C:\Users\Administrator> _
```

4.11 Search for all files containing API_KEY

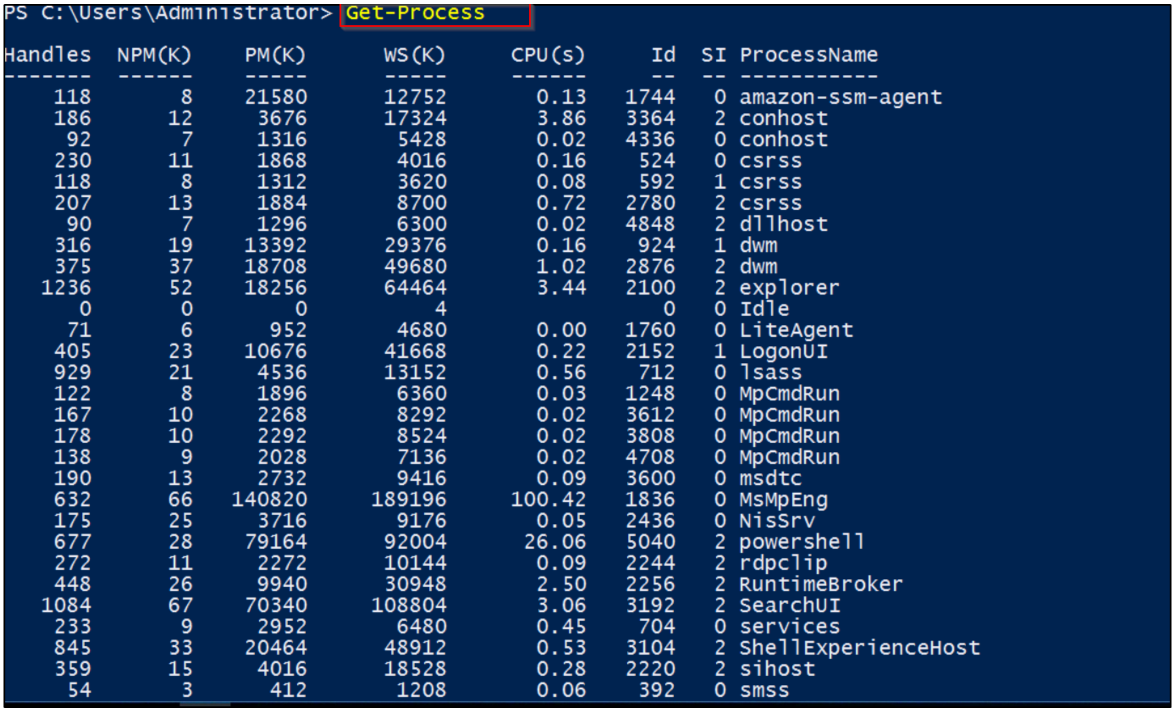
```
PS C:\> Get-ChildItem -Path C:\ -Recurse | Select-String -Pattern API_KEY
```



```
H T T P        N O N E 'R E L A T I O N A L _ D A T A B A S E A L L O W     D E N Y  A L L  E R R O R
J S O N  S D L  #R D S _ H T T P _ E N D P O I N T P I P E L I N E    U N I T
C:\Users\Public\Music\config.xml:1:API_KEY=fakekey123
```

4.12 Command to use to lists all the running processes

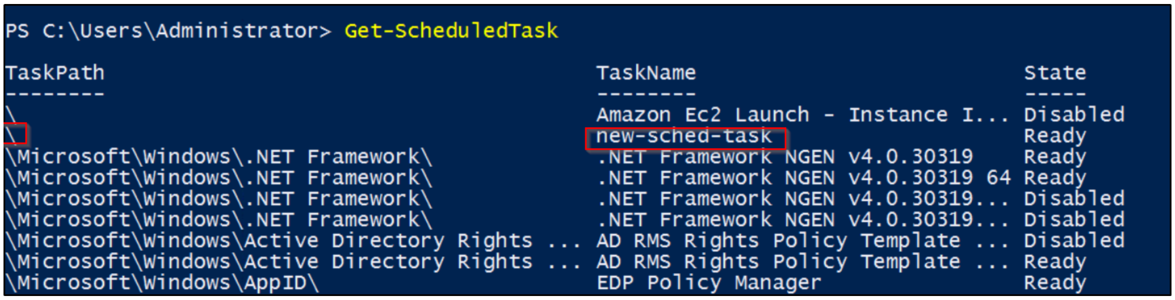
```
PS C:\> Get-Process
```



```
PS C:\Users\Administrator> Get-Process

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
    118       8    21580      12752       0.13   1744   0 amazon-ssm-agent
    186      12     3676      17324       3.86   3364   2 conhost
     92       7     1316       5428       0.02   4336   0 conhost
    230      11     1868       4016       0.16    524   0 csrss
    118       8     1312       3620       0.08    592   1 csrss
    207      13     1884       8700       0.72   2780   2 csrss
     90       7     1296       6300       0.02   4848   2 dllhost
    316      19    13392      29376       0.16    924   1 dwm
    375      37    18708      49680       1.02   2876   2 dwm
   1236      52    18256      64464       3.44   2100   2 explorer
      0       0        0          4                 0   0 Idle
     71       6      952       4680       0.00   1760   0 LiteAgent
    405      23    10676      41668       0.22   2152   1 LogonUI
    929      21     4536      13152       0.56    712   0 lsass
    122       8     1896       6360       0.03   1248   0 MpCmdRun
    167      10     2268       8292       0.02   3612   0 MpCmdRun
    178      10     2292       8524       0.02   3808   0 MpCmdRun
    138       9     2028       7136       0.02   4708   0 MpCmdRun
    190      13     2732       9416       0.09   3600   0 msdtc
    632      66   140820     189196     100.42   1836   0 MsMpEng
    175      25     3716       9176       0.05   2436   0 NisSrv
    677      28    79164      92004      26.06   5040   2 powershell
    272      11     2272      10144       0.09   2244   2 rdpclip
    448      26     9940      30948       2.50   2256   2 RuntimeBroker
   1084      67    70340     108804       3.06   3192   2 SearchUI
    233       9     2952       6480       0.45    704   0 services
    845      33    20464      48912       0.53   3104   2 ShellExperienceHost
    359      15     4016      18528       0.28   2220   2 sihost
     54       3      412       1208       0.06    392   0 smss
```

4.13 What is the path of the scheduled task called new-sched-task?

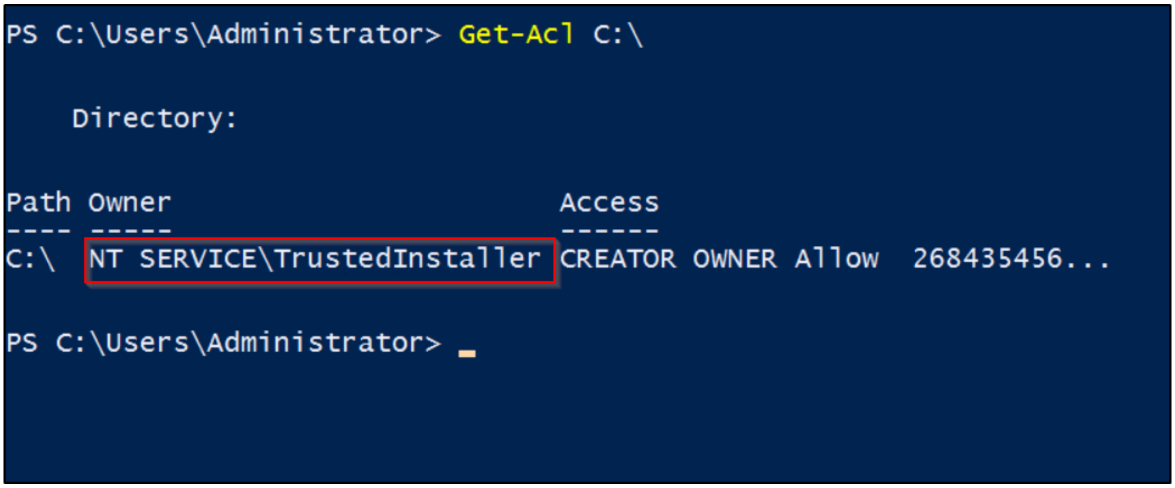
```
PS C:\> Get-ScheduledTask -TaskName new-sched-task
```



```
PS C:\Users\Administrator> Get-ScheduledTask

TaskPath                                       TaskName                          State
--------                                       --------                          -----
\                                              Amazon Ec2 Launch - Instance I... Disabled
\                                              new-sched-task                    Ready
\Microsoft\Windows\.NET Framework\             .NET Framework NGEN v4.0.30319    Ready
\Microsoft\Windows\.NET Framework\             .NET Framework NGEN v4.0.30319 64 Ready
\Microsoft\Windows\.NET Framework\             .NET Framework NGEN v4.0.30319... Disabled
\Microsoft\Windows\.NET Framework\             .NET Framework NGEN v4.0.30319... Disabled
\Microsoft\Windows\Active Directory Rights ... AD RMS Rights Policy Template ... Disabled
\Microsoft\Windows\Active Directory Rights ... AD RMS Rights Policy Template ... Ready
\Microsoft\Windows\AppID\                      EDP Policy Manager                Ready
```

4.14 Who is the Owner of C:\

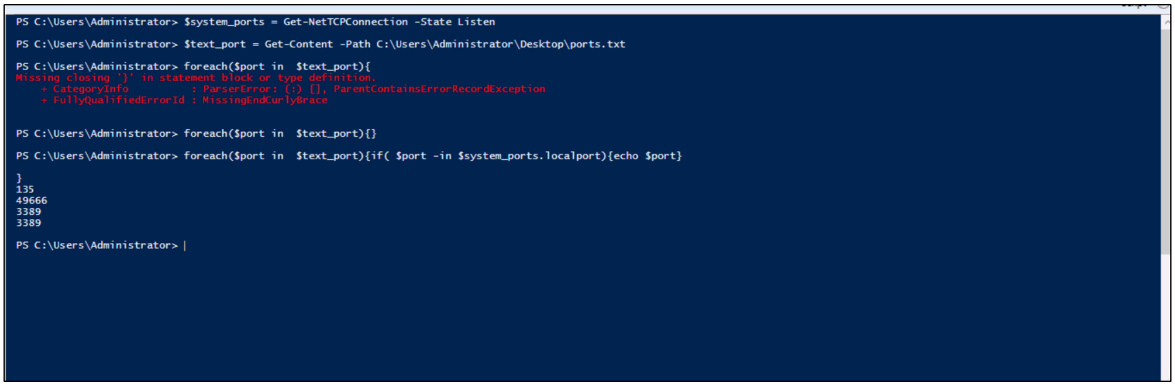
```
PS C:\> Get-Acl C:\
```



```
PS C:\Users\Administrator> Get-Acl C:\


    Directory:

Path Owner                       Access
---- -----                       ------
C:\  NT SERVICE\TrustedInstaller CREATOR OWNER Allow  268435456...

PS C:\Users\Administrator> _
```

5. Basic Scripting Challenges

basic script to use this list to see if the local port is listening.



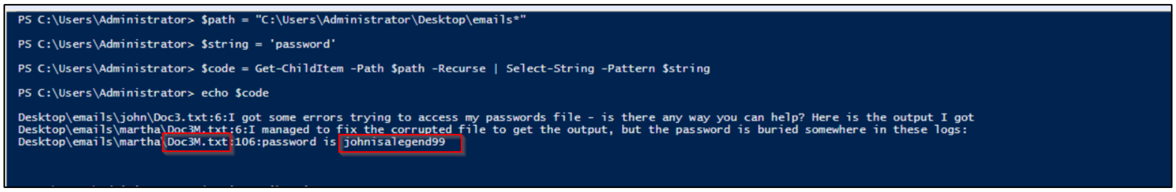
```
PS C:\Users\Administrator> $system_ports = Get-NetTCPConnection -State Listen

PS C:\Users\Administrator> $text_port = Get-Content -Path C:\Users\Administrator\Desktop\ports.txt

PS C:\Users\Administrator> foreach($port in  $text_port){
Missing closing '}' in statement block or type definition.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : MissingEndCurlyBrace


PS C:\Users\Administrator> foreach($port in  $text_port){}

PS C:\Users\Administrator> foreach($port in  $text_port){if( $port -in $system_ports.localport){echo $port}

}
135
49666
3389
3389

PS C:\Users\Administrator> |
```

5.1 What file contain Password

- DOC3M.txt

5.2 What is the Password

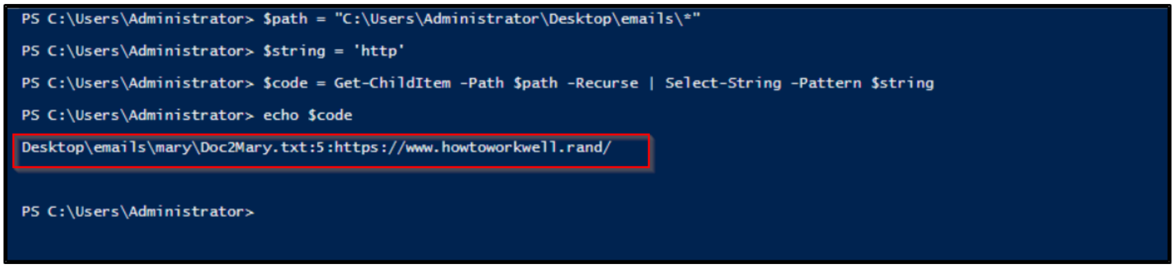
- johnislegend99



```
PS C:\Users\Administrator> $path = "C:\Users\Administrator\Desktop\emails*"

PS C:\Users\Administrator> $string = 'password'

PS C:\Users\Administrator> $code = Get-ChildItem -Path $path -Recurse | Select-String -Pattern $string

PS C:\Users\Administrator> echo $code

Desktop\emails\john\Doc3.txt:6:I got some errors trying to access my passwords file - is there any way you can help? Here is the output I got
Desktop\emails\martha\Doc3M.txt:6:I managed to fix the corrupted file to get the output, but the password is buried somewhere in these logs:
Desktop\emails\martha\Doc3M.txt:106:password is johnisalegend99
```

5.3 What file contain HTTP link

- DOC2Mary.txt



```
PS C:\Users\Administrator> $path = "C:\Users\Administrator\Desktop\emails\*"

PS C:\Users\Administrator> $string = 'http'

PS C:\Users\Administrator> $code = Get-ChildItem -Path $path -Recurse | Select-String -Pattern $string

PS C:\Users\Administrator> echo $code

Desktop\emails\mary\Doc2Mary.txt:5:https://www.howtoworkwell.rand/


PS C:\Users\Administrator>
```

6. Intermediate Scripting

6.1 How many open port do you find between 130 and 140 (inclusive these two)

- 11