# CYB631 Automating Information Security with Python and Shell Scripting

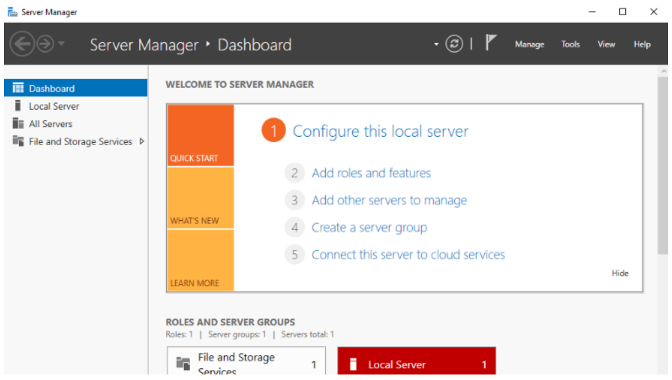
## Lab 3: Managing and Hardening Hosts

Your Name: **Vaibhav A Mayekar**

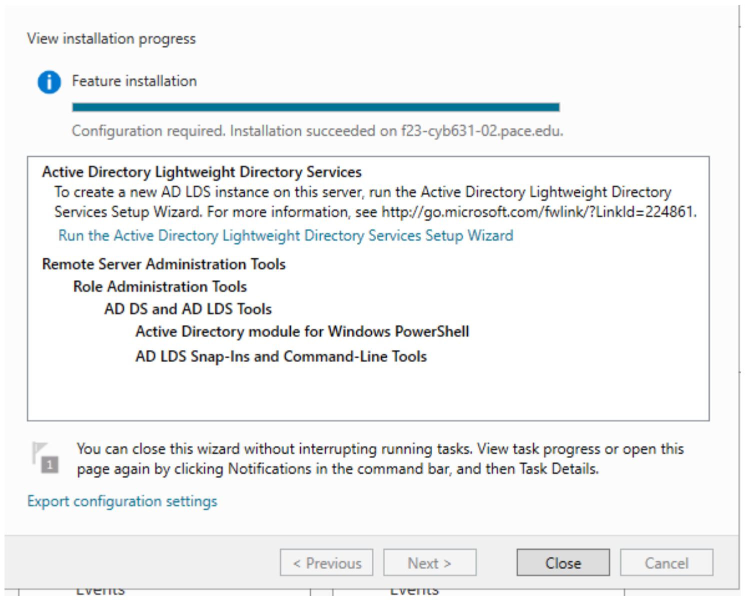
## Exercises:

## [Exercise I: Install Active Directory Lightweight Service]

1. Install Active Directory Lightweight Services (AD LDS), a lightweight version of Active Directory.
2. In the search box next to Windows Start, type "server manager" and run as "administrator" to open server manager. The you should see the Dashboard like the one below.


Server Manager
Server Manager ‣ Dashboard
Manage
Tools
View
Help
Dashboard
Local Server
All Servers
File and Storage Services
WELCOME TO SERVER MANAGER
QUICK START
WHAT'S NEW
LEARN MORE
1 Configure this local server
2 Add roles and features
3 Add other servers to manage
4 Create a server group
5 Connect this server to cloud services
Hide
ROLES AND SERVER GROUPS
Roles: 1 | Server groups: 1 | Servers total: 1
File and Storage Services 1
Local Server 1

3. Click on **Server Manager**, then **Add Roles and Features**, and check **Active Directory Lightweight Directory Service (AD LDS)**. Follow the instructions to install it. You may need to click on **Add Features, Next, and Install**.
4. Once AD LDS is installed, we will need to run the Active Directory Lightweight Service Setup Wizard. Click on the link in the message box to do so.


View installation progress

Feature installation

Configuration required. Installation succeeded on f23-cyb631-02.pace.edu.

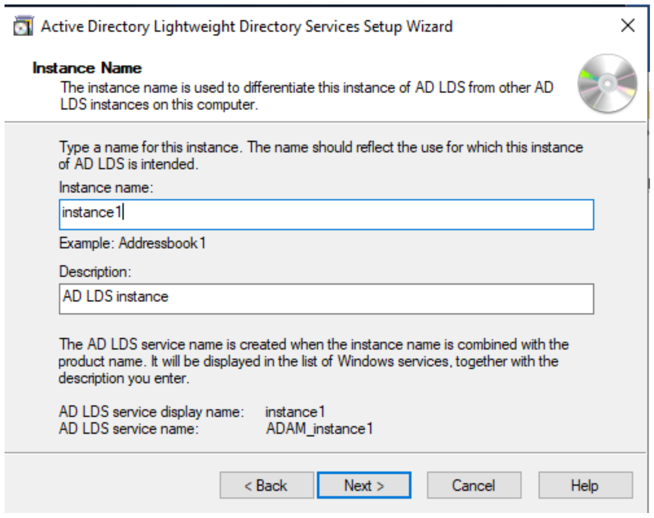
Active Directory Lightweight Directory Services
To create a new AD LDS instance on this server, run the Active Directory Lightweight Directory Services Setup Wizard. For more information, see http://go.microsoft.com/fwlink/?LinkId=224861.
Run the Active Directory Lightweight Directory Services Setup Wizard

Remote Server Administration Tools
Role Administration Tools
AD DS and AD LDS Tools
Active Directory module for Windows PowerShell
AD LDS Snap-Ins and Command-Line Tools

You can close this wizard without interrupting running tasks. View task progress or open this page again by clicking Notifications in the command bar, and then Task Details.

Export configuration settings

5. In the next step, select to install "a unique instance" and then use the default value for Instance name and Ports, like the ones below.


Active Directory Lightweight Directory Services Setup Wizard

Instance Name
The instance name is used to differentiate this instance of AD LDS from other AD LDS instances on this computer.

Type a name for this instance. The name should reflect the use for which this instance of AD LDS is intended.

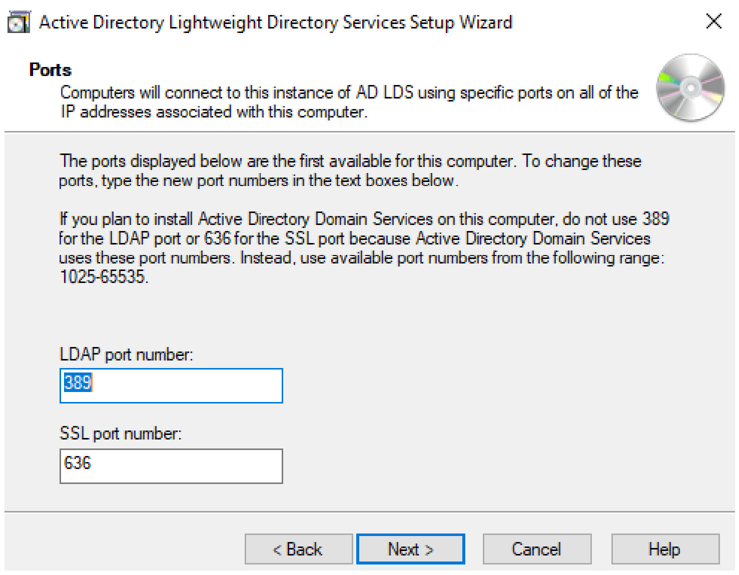
Instance name: instance1

Example: Addressbook1

Description: AD LDS instance

The AD LDS service name is created when the instance name is combined with the product name. It will be displayed in the list of Windows services, together with the description you enter.

AD LDS service display name: instance1
AD LDS service name: ADAM_instance1


Active Directory Lightweight Directory Services Setup Wizard

**Ports**

Computers will connect to this instance of AD LDS using specific ports on all of the IP addresses associated with this computer.

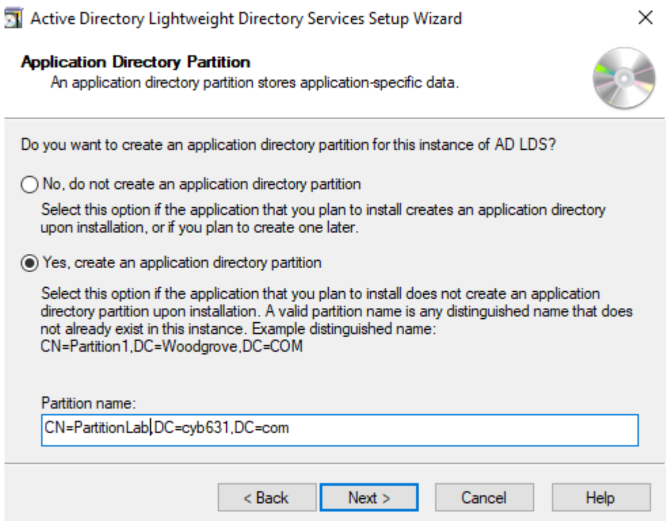
The ports displayed below are the first available for this computer. To change these ports, type the new port numbers in the text boxes below.

If you plan to install Active Directory Domain Services on this computer, do not use 389 for the LDAP port or 636 for the SSL port because Active Directory Domain Services uses these port numbers. Instead, use available port numbers from the following range: 1025-65535.

LDAP port number: 389

SSL port number: 636

6. Next, select "Yes, create an application directory partition. The, give unique names to the partition, like the one below.


Active Directory Lightweight Directory Services Setup Wizard

**Application Directory Partition**

An application directory partition stores application-specific data.

Do you want to create an application directory partition for this instance of AD LDS?

○ No, do not create an application directory partition

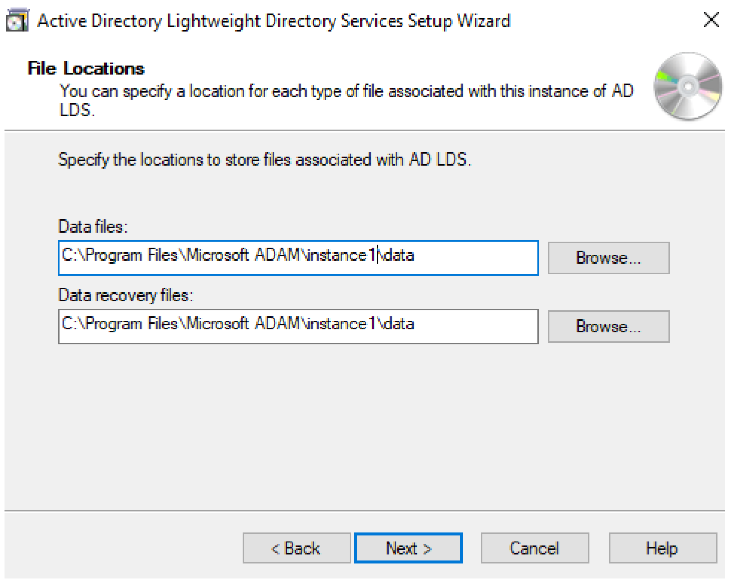
Select this option if the application that you plan to install creates an application directory upon installation, or if you plan to create one later.

● Yes, create an application directory partition

Select this option if the application that you plan to install does not create an application directory partition upon installation. A valid partition name is any distinguished name that does not already exist in this instance. Example distinguished name: CN=Partition1,DC=Woodgrove,DC=COM

Partition name: CN=PartitionLab,DC=cyb631,DC=com

7. Use default names for file location.


Active Directory Lightweight Directory Services Setup Wizard

**File Locations**

You can specify a location for each type of file associated with this instance of AD LDS.

Specify the locations to store files associated with AD LDS.

Data files:
C:\Program Files\Microsoft ADAM\instance1\data

Data recovery files:
C:\Program Files\Microsoft ADAM\instance1\data

8. Next, for Service Account Selection, choose Network service account.
9. Next, for AD LDS Administrators, choose "Currently logged on user."
10. Next, for Importing LDIF Files, click on the ones that we will use for applications. These are text files which represent data and commands used by LDAP instance. For our testing, click on **MS-User.LDF**.
11. Then follow the instructions to complete the Setup Wizard. Once it is completed, you can click on AD LDS on the Server Manager to see the details of the instance.

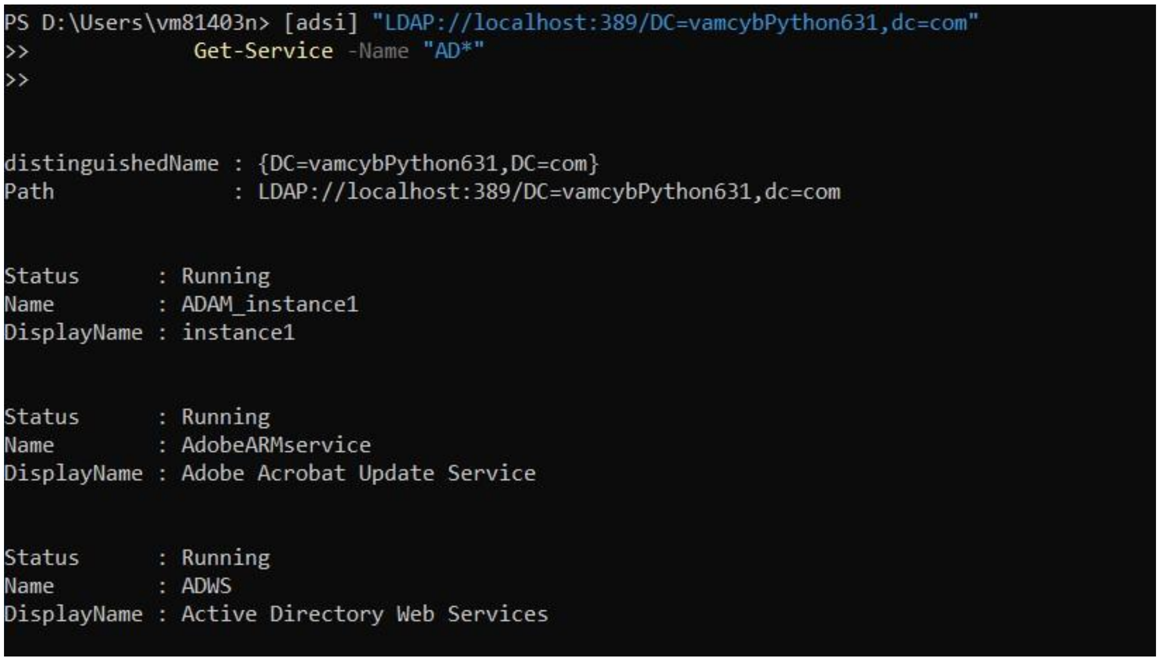
## [Exercise II: Test PowerShell on AD LDS]

12. Show the instance that we built earlier, and show the services that were run by the AD service. You should see ADAM instance is running and ADWS service is provided.

    **[adsi] "LDAP://localhost:389/cn=PartitionLab,dc=cyb631,dc=com"**

    **Get-Service -Name "AD*"**

13. Paste a screenshot of your results above.



```
PS D:\Users\vm81403n> [adsi] "LDAP://localhost:389/DC=vamcybPython631,dc=com"
>>             Get-Service -Name "AD*"
>>


distinguishedName : {DC=vamcybPython631,DC=com}
Path              : LDAP://localhost:389/DC=vamcybPython631,dc=com


Status      : Running
Name        : ADAM_instance1
DisplayName : instance1


Status      : Running
Name        : AdobeARMservice
DisplayName : Adobe Acrobat Update Service


Status      : Running
Name        : ADWS
DisplayName : Active Directory Web Services
```

14. Review the container, **domain**.

**$domain=[adsi]  "LDAP://localhost:389/cn=PartitionLab,dc=cyb631,dc=com"**

**$domain | format-list ***

15. Add user information to the directory.

**Add Use$user = $domain.Create("User", "cn=PartitionLab")**

**$user.Put("userPrincipalName", "KenMyer@cyb631.com")**

**$user.Put("displayName", "Ken Myer")**

**$user.SetInfo()**

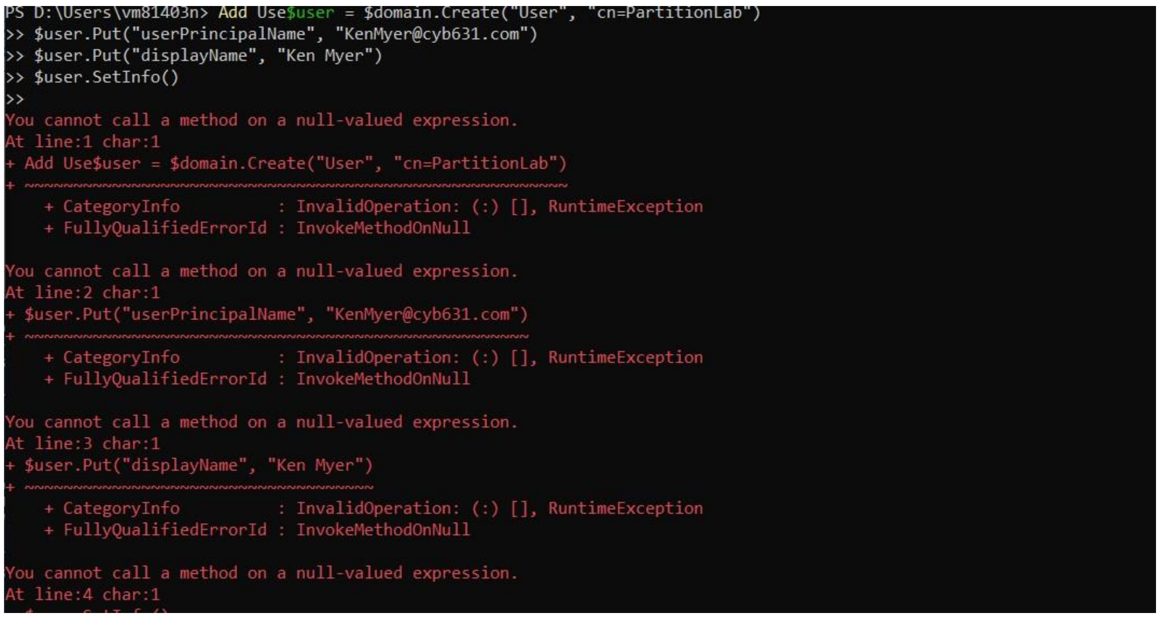
16. Display user information.

**$user.userPrincipalName**

**$user.displayName**

17. Paste a screenshot of your results above.

**No output getting an error that user cannot be generated.**



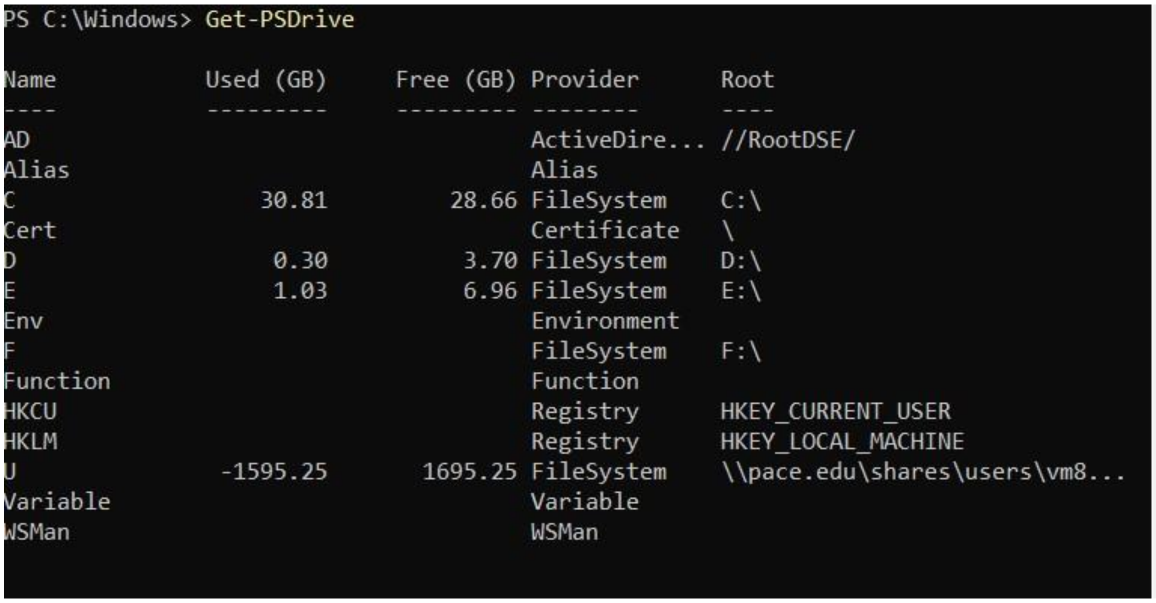
```
PS D:\Users\vm81403n> Add Use$user = $domain.Create("User", "cn=PartitionLab")
>> $user.Put("userPrincipalName", "KenMyer@cyb631.com")
>> $user.Put("displayName", "Ken Myer")
>> $user.SetInfo()
>>
You cannot call a method on a null-valued expression.
At line:1 char:1
+ Add Use$user = $domain.Create("User", "cn=PartitionLab")
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [], RuntimeException
    + FullyQualifiedErrorId : InvokeMethodOnNull

You cannot call a method on a null-valued expression.
At line:2 char:1
+ $user.Put("userPrincipalName", "KenMyer@cyb631.com")
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [], RuntimeException
    + FullyQualifiedErrorId : InvokeMethodOnNull

You cannot call a method on a null-valued expression.
At line:3 char:1
+ $user.Put("displayName", "Ken Myer")
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [], RuntimeException
    + FullyQualifiedErrorId : InvokeMethodOnNull

You cannot call a method on a null-valued expression.
At line:4 char:1
```

## [Exercise III:  Windows Registry]

18. To see where the registry hive keys are on the drive.

    **Get-PSDrive**

19. Paste a screenshot of your results.



```
PS C:\Windows> Get-PSDrive

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
AD                                     ActiveDire... //RootDSE/
Alias                                  Alias
C                  30.81         28.66 FileSystem    C:\
Cert                                   Certificate   \
D                   0.30          3.70 FileSystem    D:\
E                   1.03          6.96 FileSystem    E:\
Env                                    Environment
F                                      FileSystem    F:\
Function                               Function
HKCU                                   Registry      HKEY_CURRENT_USER
HKLM                                   Registry      HKEY_LOCAL_MACHINE
U               -1595.25       1695.25 FileSystem    \\pace.edu\shares\users\vm8...
Variable                               Variable
WSMan                                  WSMan
```

20. On the lower left corner, run “regedit” as a Windows command. This will open up Windows registry editor. This will show all of registry hive keys. Explore the registry and review what they are. Please list two of them here

**HKEY_USERS**

**HKEY_LOCAL_MACHINE**.

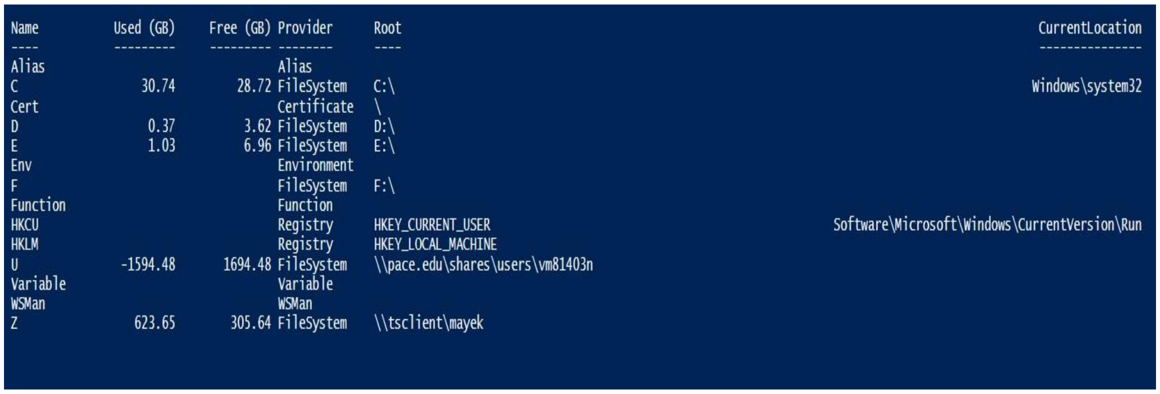
21. Close regedit. We will now use PowerShell to retrieve registry. Under PowerShell ISE,

    **Set-Location HKCU:\Software\Microsoft\Windows\CurrentVersion\Run**

    **$item=Get-ItemProperty .**

    **$item**

22. This will show hive keys under current users (HKCU). Paste your results here.



```
Name         Used (GB)     Free (GB) Provider     Root                                                   CurrentLocation
----         ---------     --------- --------     ----                                                   ---------------
Alias                                Alias
C                30.74         28.72 FileSystem   C:\                                                    Windows\system32
Cert                                 Certificate  \
D                 0.37          3.62 FileSystem   D:\
E                 1.03          6.96 FileSystem   E:\
Env                                  Environment
F                                    FileSystem   F:\
Function                             Function
HKCU                                 Registry     HKEY_CURRENT_USER             Software\Microsoft\Windows\CurrentVersion\Run
HKLM                                 Registry     HKEY_LOCAL_MACHINE
U             -1594.48       1694.48 FileSystem   \\pace.edu\shares\users\vm81403n
Variable                             Variable
WSMan                                WSMan
Z               623.65        305.64 FileSystem   \\tsclient\mayek
```

## [Exercise IV:  Windows Management Instrumentation]

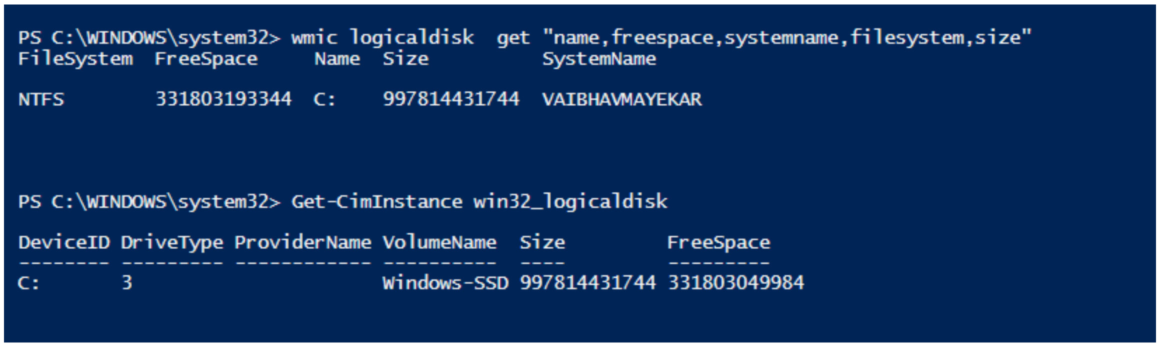
23. Let access logical disk information using WMIC.

    **wmic logicaldisk  get "name,freespace,systemname,filesystem,size"**

24. Now, let access the same information using CIM cmdlet. Get-CimIntance obtains an CIM instance of a class, in this case, win32_logicaldisk

    **Get-CimInstance win32_logicaldisk**

25. Paste a screenshot of the results from above.



```
PS C:\WINDOWS\system32> wmic logicaldisk  get "name,freespace,systemname,filesystem,size"
FileSystem  FreeSpace     Name  Size          SystemName

NTFS        331803193344  C:    997814431744  VAIBHAVMAYEKAR



PS C:\WINDOWS\system32> Get-CimInstance win32_logicaldisk

DeviceID DriveType ProviderName VolumeName  Size         FreeSpace
-------- --------- ------------ ----------  ----         ---------
C:       3                      Windows-SSD 997814431744 331803049984
```

26. To see all of the classes available,

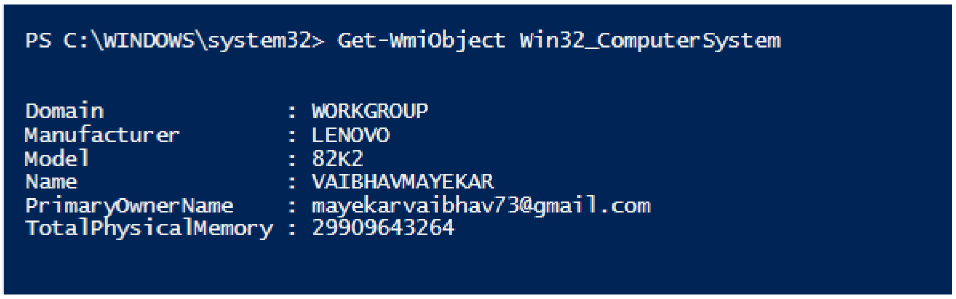
**Get-CimClass -ClassName ***

**Win32_ACE**

27. Let us try a WMI cmdlet to obtain computer information.

    **Get-WmiObject Win32_ComputerSystem**

28. Paste the results of your results above.



```
PS C:\WINDOWS\system32> Get-WmiObject Win32_ComputerSystem


Domain              : WORKGROUP
Manufacturer        : LENOVO
Model               : 82K2
Name                : VAIBHAVMAYEKAR
PrimaryOwnerName    : mayekarvaibhav73@gmail.com
TotalPhysicalMemory : 29909643264
```

29. You can use the CIM cmdlet to obtain similar information. CIM cmdlet is more portable since it is across platform.

    **Get-CimInstance CIM_ComputerSystem**

30. Try another CIM cmdlet to receive process information.

    **Get-CimInstance Win32_Process | Select Name,ProcessId,ThreadCount**

31. You can use the WQL language, similar to SQL, to access the information.

**Get-CimInstance -query "select * from win32_service where StartMode='auto'"**

32. Explain what the results of the above command mean.

    **The command retrieves a list of Windows services that are set to start automatically when the computer boots up and also the results include information about these services, such as their names, display names, current states**

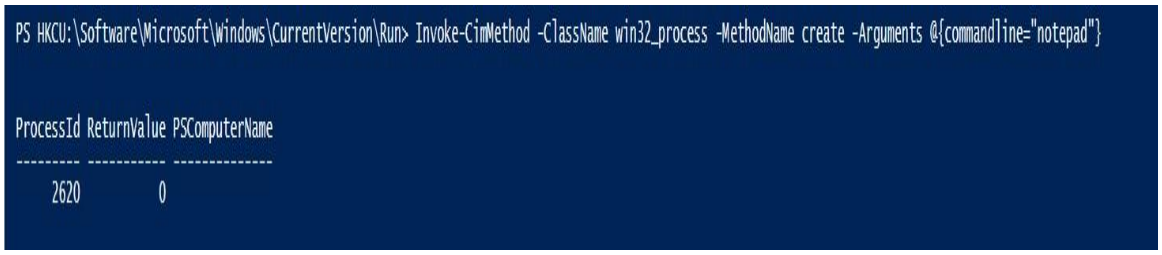
33. Invoke a method supported by a WMI or CIM class. The following invoke a notepad process.

**Invoke-CimMethod -ClassName win32_process -MethodName create -Arguments @{commandline="notepad"}**

34. What happened after you run the script above?

    **A new instance of the Notepad process should be created, and the Notepad application should open on your computer.**

    Paste a screenshot of your results from PowerShell ISE.



```
PS HKCU:\Software\Microsoft\Windows\CurrentVersion\Run> Invoke-CimMethod -ClassName win32_process -MethodName create -Arguments @{commandline="notepad"}


ProcessId ReturnValue PSComputerName
--------- ----------- --------------
     2620           0
```

## [Exercise V:  Configure Windows Firewall]

35. PowerShell has a NetSecurity Module for configuring Firewall and IPSec.

    **Get-Command -module NetSecurity**

36. To enable Firewall to all profiles.

    **Set-NetFirewallProfile -All -Enabled True**

37. Let us try several firewall rules. Block access to web servers inside this host.

    **New-NetFirewallRule -DisplayName "HTTP-Inbound" -Profile Any -Direction Inbound -Action Block -Protocol tcp -LocalPort @('80','443')**

38. Use a browser to see if you can access servers on the Internet. You should be able to do so at this point. Now, block access from external web servers to this host.

    **New-NetFirewallRule -DisplayName "HTTP-outbound" -Profile Any -Direction outbound -Action Allow -Protocol tcp -RemotePort @('80','443')**

39. Now, try use a web browser to see if you are able to access external web servers. Are you still be able to do that now? **No** Why or Why not?

    **When you block outbound HTTP traffic, you are preventing your computer from sending HTTP requests to external web servers. This means that you will not be able to load any web pages, or use any other applications that rely on HTTP traffic, such as email clients or social media apps.**

40. Now, let us open our Internet web access. We will have to modify the rule that we set earlier.

    **Set-NetFirewallRule -DisplayName "HTTP-outbound" -Profile Any -Direction outbound -Action Allow -Protocol tcp -RemotePort @('80','443')**

41. You can also remove firewall rules.

    **Remove-NetFirewallRule -Action Block**

## [Exercise VI: Develop a PowerShell to Automatically Configure Windows Firewall]

42. Please write a PowerShell script to enable Windows Firewall and include at least 2 new rules. These rules are:

    Rule 1: block any attempts to access SSH servers from outside to inside of your Active Directory domain.

    Rules 2: block any attempts to access DNS servers from outside to inside of your Active Directory domain.

43. Paste a screenshot of the script.

**# Enable Windows Firewall**

**Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled True**

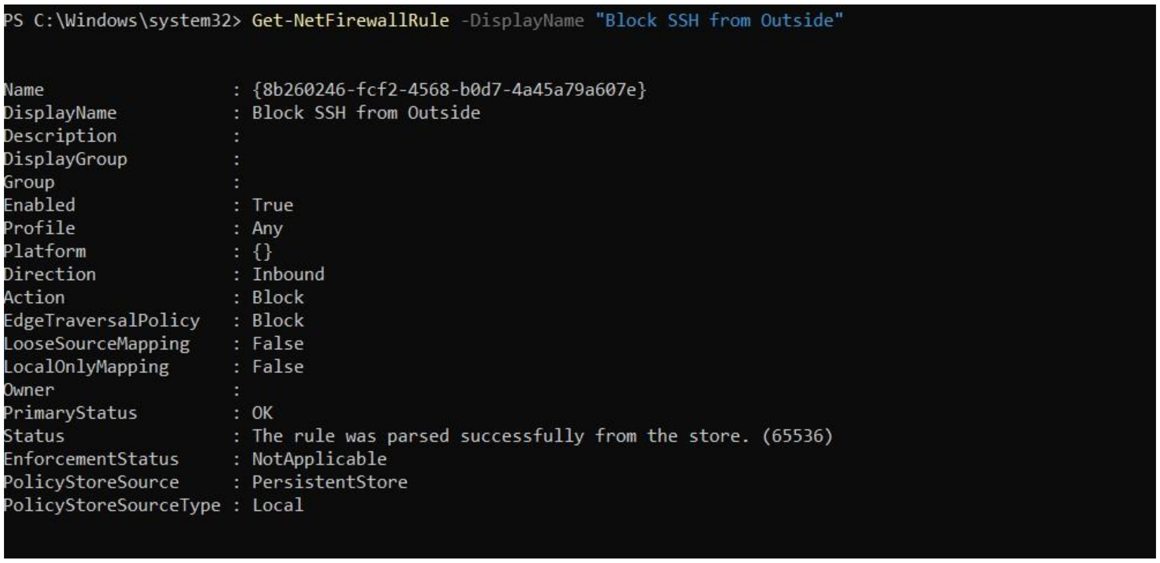
**# Create SSH blocking rule**

**New-NetFirewallRule -DisplayName "Block SSH from Outside" -Direction Inbound -Protocol TCP -Action Block -Enabled True -RemoteAddress Any -LocalPort 22**

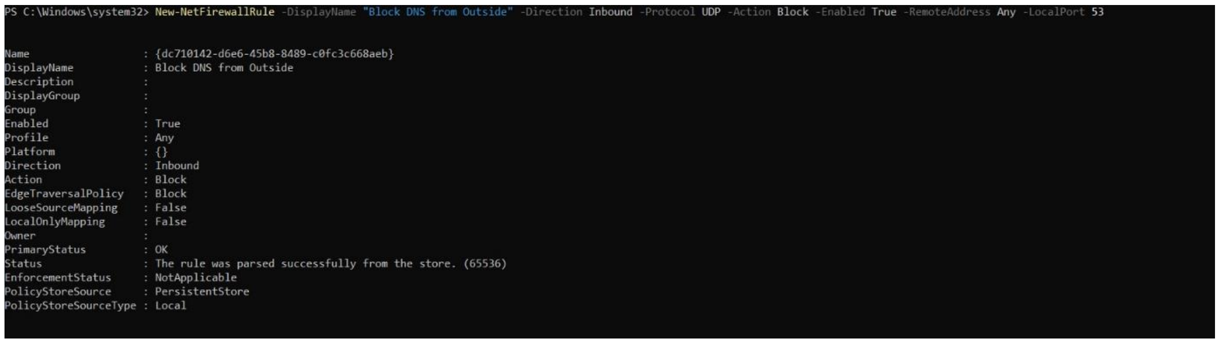
**# Create DNS blocking rule**

**New-NetFirewallRule -DisplayName "Block DNS from Outside" -Direction Inbound -Protocol UDP -Action Block -Enabled True -RemoteAddress Any -LocalPort 53**

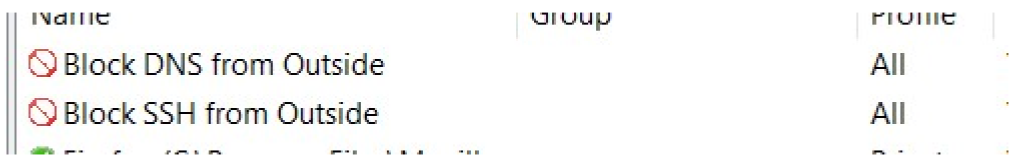
44. Paste a screenshot of results from running the script (the results can be long, and you only must show the beginning part of the results).



```
PS C:\Windows\system32> Get-NetFirewallRule -DisplayName "Block SSH from Outside"


Name                  : {8b260246-fcf2-4568-b0d7-4a45a79a607e}
DisplayName           : Block SSH from Outside
Description           :
DisplayGroup          :
Group                 :
Enabled               : True
Profile               : Any
Platform              : {}
Direction             : Inbound
Action                : Block
EdgeTraversalPolicy   : Block
LooseSourceMapping    : False
LocalOnlyMapping      : False
Owner                 :
PrimaryStatus         : OK
Status                : The rule was parsed successfully from the store. (65536)
EnforcementStatus     : NotApplicable
PolicyStoreSource     : PersistentStore
PolicyStoreSourceType : Local
```



```
PS C:\Windows\system32> New-NetFirewallRule -DisplayName "Block DNS from Outside" -Direction Inbound -Protocol UDP -Action Block -Enabled True -RemoteAddress Any -LocalPort 53


Name                  : {dc710142-d6e6-45b8-8489-c0fc3c668aeb}
DisplayName           : Block DNS from Outside
Description           :
DisplayGroup          :
Group                 :
Enabled               : True
Profile               : Any
Platform              : {}
Direction             : Inbound
Action                : Block
EdgeTraversalPolicy   : Block
LooseSourceMapping    : False
LocalOnlyMapping      : False
Owner                 :
PrimaryStatus         : OK
Status                : The rule was parsed successfully from the store. (65536)
EnforcementStatus     : NotApplicable
PolicyStoreSource     : PersistentStore
PolicyStoreSourceType : Local
```



| Name | Group | Profile |
|---|---|---|
| Block DNS from Outside | | All |
| Block SSH from Outside | | All |

45. Briefly discuss the advantages of using PowerShell scripts to configure Windows Firewall.

**Using PowerShell scripts to configure Windows Firewall offers the benefits of automation, ensuring consistent rule enforcement across multiple systems, scalability for efficient management, customization to tailor rules to specific needs, version control for tracking changes, logging and reporting for monitoring and auditing, and seamless integration with other tools and systems, all of which collectively enhance network security and simplify rule management.**