HACKTHEBOX

INTRODUCTION TO WINDOWS COMMAND LINE

CHEAT SHEET

Introduction to Windows Command Line

Admin Commands

Command	Description
`xfreerdp /v:<target IP> /u:<user> /p:<password>`	Initiate a RDP connection with the target host.
`ssh <user>@<target IP>`	Connect to target host via SSH.
`<PIPE>`	When you see `<PIPE>` specified in the commands below, it is saying to use the Pipe key (shift+backslash on US Keyboard layouts).

General Commands

Command	Description
`help <command>`	Provides help information for Windows commands.

Command	Description
`Get-Help <cmdlet>`	Displays help about Windows PowerShell cmdlets and concepts.
`Update-Help`	Downloads and installs the most up-to-date help files for Windows PowerShell.
`CTRL-C`	Interrupts a currently running process.
`Get-Module`	View the modules loaded into your PowerShell session.
`Import-Module`	Import a module into your PowerShell session.
`Get-Command`	View all commands, cmdlets, functions, and aliases loaded into your PowerShell session.
`Set-Location <path>`	Changes our location in the filesystem. Same as using CD.
`Get-Content <file>`	View the contents of an object. Similar to type or cat.
`systeminfo`	Displays operating system configuration information for a local or remote machine.
`hostname`	Displays the name of the current host.
`ver`	Displays the current Windows version.

Terminal History

Command/Key	Description
`doskey /history`	Prints out the session's command history to the terminal or output it to a file when specified.
`page up`	Places the first command in our session history to the prompt.
`page down`	Places the last command in history to the prompt.

HTB ACADEMY CHEATSHEET

Command/Key	Description
`⇧`	Scrolls up through our command history to view previously run commands.
`⇩`	Scrolls down to our most recent commands run.
`⇨`	Types the previous command to prompt one character at a time.
`F3`	Retypes the entire previous entry to our prompt.
`F5`	Pressing F5 multiple times allows us to cycle through previous commands.
`F7`	Opens an interactive list of previous commands.
`F9`	Enters a command to our prompt based on the number specified. The number corresponds to the command's place in our history.

File & Directory Commands

CMD.exe

Command	Description
`dir`	Lists directory contents.
`dir /A <attributes>`	List directory contents with the specified attributes.
`dir /A:H`	List hidden files in the current directory.
`dir /A:R`	List read-only files in the current directory.
`cd`	Prints current working directory.
`chdir`	Prints current working directory. Alternate command.
`cd <path>`	Changes the directory.

Command	Description
`chdir <path>`	Changes the directory. Alternate command.
`tree <path>`	Graphically displays the directory structure from the specified path.
`tree /F <path>`	Graphically displays the directory structure from the specified path, including files within the directory
`cls`	Clears the terminal.
`mkdir <directory name>`	Creates a directory in the current working directory(or specified directory) with the specified name.
`md <directory name>`	Creates a directory in the current working directory(or specified directory) with the specified name. Alias of mkdir.
`rmdir <directory name>`	Removes a directory in the current working directory(or specified directory) with the specified name.
`rd <directory name>`	Removes a directory in the current working directory(or specified directory) with the specified name. Alias of rmdir
`rmdir /S <directory name>`	Recursively removes all directories and files in the specified directory.
`move [source] [destination]`	Move file(s) from the source folder to the destination folder.
`copy [source] [destination]`	Copy file(s) from the source folder to the destination folder. Only works with files and not folders.
`copy [source] [destination] /V`	Copy file(s) from the source folder to the destination folder. Validates that the file or files are copied correctly.
`xcopy [source] [destination]`	Copy file(s) and folder(s) from the source folder to the destination folder. Replaced by Robocopy and currently deprecated.
`xcopy /E [source] [destination]`	Copy file(s) and folder(s) from the source folder to the destination folder, including empty directories.

HTB ACADEMY CHEATSHEET

Command	Description
`xcopy /K [source] [destination]`	Copy file(s) and folder(s) from the source folder to the destination folder. Retains the current attributes of the copied files.
`robocopy [source] [destination]`	Copy files(s) and folder(s) from the source folder to the destination folder. It has a more robust feature set compared to xcopy.
`robocopy /E /MIR /A-:SH [source] [destination]`	Copy files(s) and folder(s) from the source folder to the destination folder. Mirrors the destination directory to the source and clears any additional attributes using the `/A-:SH` parameter.
`more <file>`	Displays the output of a file or command one screen at a time.
`more /S <file>`	Displays the output of a file or command one screen at a time. Compresses multiple blank lines into a single line.
`<command> <PIPE> more`	Displays the output of a command through a <PIPE> to `more`.
`type <file>`	Displays the contents of a file.
`fsutil file createNew <filename> <length>`	Creates a new file with a specified file name and length.
`echo "example string" > <filename>`	Writes the contents provided into a new or existing file with the specified filename. If the file does not exist, a new one will be created; otherwise, the previous file's contents will be overwritten.
`echo "example string" >> <filename>`	Appends the provided contents to an existing file.
`ren <filename1> <filename2>`	Renames a file.
`del <file>`	Deletes a file or files.
`del /A:R <file>`	Deletes a file or files with the read-only attribute set.

Command	Description
`del /A:H <file>`	Deletes a file or files with the hidden attribute set.
`erase <file>`	Deletes a file or files. Interchangeable with `del` command.

PowerShell

Command	Alias	Description
`Get-Item`	gi	Retrieve an object (could be a file, folder, registry object, etc.)
`Get-ChildItem`	ls / dir / gci	Lists out the content of a folder or registry hive.
`New-Item`	md / mkdir / ni	Create new objects. (can be files, folders, symlinks, registry entries and more)
`new-item -name "Name" -ItemType <directory/file>`	Specify the new items name and object type.	
`Set-Item`	si	Modify the property values of an object.
`Copy-Item`	copy / cp / ci	Make a duplicate of the item.
`Rename-Item`	ren / rni	Changes the object name.
`Rename-Item .\Object-1.md -NewName Object-2.md`	Rename object-1 to object-2.	
`Remove-Item`	rm / del / rmdir	Deletes the object.
`Get-Content`	cat / type	Displays the content within a file or object.
`Add-Content <file> "Content to add"`	ac	Append content to a file.

Command	Alias	Description
`Set-Content`	sc	overwrite any content in a file with new data.
`Clear-Content`	clc	Clear the content of the files without deleting the file itself.
`Compare-Object`	diff / compare	Compare two or more objects against each other. This includes the object itself and the content within.

Input/Output Operators

Operator	Description
`[command] > [file]`	Redirects the output from a command into a file. Overwrites the specified files' contents.
`[command] >> [file]`	Redirects the output from a command into a file. Appends additional output without overwriting the file's original contents.
`[command] < [file]`	Redirects the output of the file and passes it into the command.
`[command] | [command2]`	Redirects the output of the first command into a `<PIPE>` and provides it to the second command.
`[command] & [command2]`	Executes both commands in succession. It does not perform checks to see if either command passes or fails.
`[command] && [command2]`	Checks to see if the first command executes successfully and then executes the second command. If the first command fails, the current command execution halts and the second command is not executed.
`[command] || [command2]`	Checks to see if the first command fails to execute successfully and, if so, proceeds to execute the second command.

Find & Filter Content

CMD.exe

Command	Description
`where <file>`	Displays the location of file(s) provided.
`where /R <working directory> <file>`	Recursively searches for the file(s) provided starting from the specified directory.
`find "example string" <file>`	Searches for a string of text in a file or files, and displays lines of text that contain the specified string.
`findstr`	Searches for patterns of text in files. Similar to `grep` on Unix/Linux.
`comp <file1> <file2>`	Compares the contents of two files or sets of files byte-by-byte.
`fc <file1> <file2>`	Compares two files or sets of files and displays the differences between them.
`sort`	Reads input, sorts data, and writes the results to the screen, a file, or another device.

PowerShell

Command	Description
`Get-Item <item> <PIPE> get-member`	Use Get-Item to select an object and then Get-Member to view the object's properties.
`Get-Item <item> <PIPE> Select-Object -Property *`	Select an object and then view its Property values.
`Get-Item * <PIPE> Select-Object -Property Name,PasswordLastSet`	Select objects and then filter to view specific properties.

Command	Description
`Get-Item * <PIPE> Sort-Object -Property Name <PIPE> Group-Object -property Enabled`	Sort and view Objects by a specific property setting.
`Get-ChildItem -Path C:\Users\MTanaka\ -File -Recurse`	List all File objects in the directory specified.
`Get-Childitem -Path C:\Users\MTanaka\ -File -Recurse -ErrorAction SilentlyContinue <PIPE> where {($_.Name -like "*.txt")}`	Search for all objects with the '.txt' file extension.
`Get-Childitem –Path C:\Users\MTanaka\ -File -Recurse -ErrorAction SilentlyContinue <PIPE> where {($_.Name -like "*.txt" -or $_.Name -like "*.py" -or $_.Name -like "*.ps1" -or $_.Name -like "*.md" -or $_.Name -like "*.csv")}`	Search for objects matching a list of different file extensions.
`Get-ChildItem -Path C:\Users\MTanaka\ -Filter "*.txt" -Recurse -File <PIPE> sls "Password","credential","key"`	Searching for keywords within an object's content.

User Commands

CMD.exe

Commands	Description
`whoami`	Displays the username of the currently logged-on user.
`whoami /priv`	Displays the security privileges of the current user.
`whoami /groups`	Displays the user groups that the current user belongs to.
`whoami /all`	Displays all information about the current user, including username, security identifiers (SID), privileges, and groups.
`net user`	Displays a list of the user accounts on the computer

HTB ACADEMY CHEATSHEET

Commands	Description
`net localgroup`	Displays the name of the server and the names of local groups on the computer.
`net group`	Displays the name of a server and the names of groups on the server. Only able to be used if the machine is joined to the domain.

PowerShell

Commands	Description
`Get-LocalGroup`	View all groups specific to the host only.
`Get-LocalUser`	View all local users. Similar to net user.
`New-LocalUser -Name "username" -NoPassword`	Create a new Local user.
`Set-LocalUser -Name "username" -Password $Password -Description "users description"`	Modify a local user's settings.
`Get-LocalGroupMember -Name "Group Name"`	Check Group membership.
`Add-LocalGroupMember -Group "Group Name" -Member "User-To-Add"`	Add a user to a local group.
`Get-WindowsCapability -Name RSAT* -Online | Add-WindowsCapability -Online`	Install Remote System Administration Tools.
`Get-Module -Name ActiveDirectory -ListAvailable`	Locate the Active Directory module.
`Get-ADUser -FIlter *`	List all domain users.
`Get-ADUser -Identity <name>`	Show a specific domain user and its properties.

Commands	Description
`Get-ADUser -Filter {EmailAddress -like '*greenhorn.corp'}`	Filter domain users based on the EmailAddress property.
`New-ADUser -Name "UserName" -Surname "Last Name" -GivenName "First Name" -Office "Security" -OtherAttributes @{'title'="Sensei";'mail'="UserName@greenhorn.corp"} -Accountpassword (Read-Host -AsSecureString "AccountPassword") -Enabled $true`	Create a New Domain user and set its properties such as name, password, and other attributes.
`Set-ADUser -Identity <UserName> -Description " Information we want in the description field"`	Modify the property settings of a domain user.

Networking Commands

CMD.exe

Command	Description
`ipconfig`	View basic networking configurations.
`ipconfig /?`	Displays help and usage information for `ipconfig`.
`ipconfig /all`	View detailed networking configuration information.
`net`	CLI utility containing multiple commands to manage and configure network resources.
`net share`	Displays info about all of the resources that are shared on the local computer.
`net view`	Displays a list of domains, computers, or resources being shared by the specified computer.
`arp`	Displays and manages the contents and entries within the `Address Resolution Protocol` (ARP) cache.

Command	Description
`arp /a`	Displays the contents and entries contained within the `Address Resolution Protocol` (ARP) cache.
`netstat -an`	Display current network connections.
`nslookup <query>`	Query DNS for a name or address.

PowerShell

Command	Description
`Get-NetIPInterface -ifIndex <#>`	Retrieve network adapter `properties` of the interface listed as ifIndex #.
`Get-NetIPAddress`	Retrieves the `IP configurations` of each adapter. Similar to `IPConfig`.
`Get-NetNeighbor`	Retrieves the `neighbor entries` from the cache. Similar to `arp -a`.
`Get-Netroute`	Will print the current `route table`. Similar to `IPRoute`.
`Set-NetAdapter`	Set basic adapter properties at the `Layer-2` level, such as VLAN id, description, and MAC-Address.
`Set-NetIPInterface`	Modifies the `settings` of an `interface` to include DHCP status, MTU, and other metrics.
`Set-NetIPAddress`	Modifies the `configuration` of a network adapter.
`Disable-NetAdapter`	Used to `disable` network adapter interfaces.
`Enable-NetAdapter`	Used to turn network adapters back on and `allow` network connections.

Command	Description
`Restart-NetAdapter`	Used to restart an adapter. It can be useful to help push `changes` made to adapter `settings`.
`test-NetConnection`	Allows for `diagnostic` checks to be run on a connection. It supports ping, tcp, route tracing, and more.
`Get-WindowsCapability -Online <PIPE> Where-Object Name -like 'OpenSSH*'`	List Windows packages for OpenSSH.
`Add-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0`	Install the SSH package to the host.
`ssh <user>@<ip address>`	Basic SSH connect string.
`ssh-keygen`	Generate SSH keys for the user you run the command as. This enables the use of the user for remote login.
`winrm quickconfig`	Enable WinRM.
`Test-WSMan -ComputerName "10.129.224.248"`	Test if the host specified has WinRM running.
`Enter-PSSession -ComputerName 10.129.224.248 -Credential htb-student -Authentication Negotiate`	Start a remote PowerShell session with the host specified.

Environment Variables

Command	Description
`%EXAMPLE_VARIABLE%`	Example format for an environment variable.
`set`	Prints all available environment variables on the system.
`set <%VARIABLE_NAME%>`	Prints out the value of the environment variable specified. It can also be used to set the variable's value.

Command	Description
`echo <%VARIABLE_NAME%>`	Prints out the value of the environment variable specified. It cannot make any edits to variables and will only print out the values to the console.
`set <%VARIABLE_NAME%>=<Value>`	Creates a new environment variable or modifies an existing one and sets the value for the current command line session.
`setx <%VARIABLE_NAME%> <Value>`	Creates a new environment variable or modifies an existing one and sets the value globally by making changes to the registry.
`set <%VARIABLE_NAME%>=`	Removes the environment variable with the specified name for the current command line session.
`setx <%VARIABLE_NAME%> ""`	Removes the environment variable with the specified name globally.

Services

CMD.exe

Command	Description
`sc query`	Lists all `running` services and provides additional information for each service.
`sc query <Name>`	Lists details about a specific service by name.
`sc start <Name>`	Start a service by name.
`sc stop <Name>`	Stop a service by name.
`sc config <Name> start= disabled`	Change settings of the service specified.
`tasklist /svc`	Provide a list of services running under each process on the system.

Command	Description
`net start`	List all `running` services.
`wmic service list brief`	List all services on the system using `WMIC`. Includes information such as: `ExitCode`, `Name`, `ProcessID`, `StartMode`, `State`, and `Status`.

PowerShell

Command	Description
`Get-service`	List all services
`Get-Service <PIPE> ft DisplayName,Status`	List all services and format their information by DisplayName and Status.
`Get-Service <PIPE> where DisplayName -like '*Name*' <PIPE> ft DisplayName,ServiceName,Status`	Query for a specific service whose name matches '*name*'.
`Start-Service <Name>`	Start a service by name.
`Stop-Service <Name>`	Stop a service by name.
`Set-Service -Name <Name> -StartType Disabled`	Change settings of the service specified.
`Get-service -ComputerName ACADEMY-ICL-DC`	Remote query of a hosts services.
`Get-Service -ComputerName ACADEMY-ICL-DC | Where-Object {$_.Status -eq "Running"}`	Remote query of services filtered to only show those that are Running.
`Invoke-command -ComputerName ACADEMY-ICL-DC,LOCALHOST -ScriptBlock {Get-Service -Name 'windefend'}`	Issue the Get-Service command on a list of hosts.

Scheduled Tasks

Command	Description
`schtasks`	Displays all tasks scheduled on the local machine.
`schtasks /query`	Displays all tasks scheduled on the local machine. Interchangeable with `schtasks` command.
`schtasks /query /V /FO list`	Displays all scheduled tasks with `verbose` information in a `list` format.
`schtasks /create`	Allows for the creation of scheduled tasks.
`schtasks /create /sc <Schedule Frequency> /tn <Task Name> /tr <Program Path>`	Creates a new scheduled task based on a select `schedule`, with a provided `name`, and a `program` specified to run when the task starts.
`schtasks /change`	Allows for modification of an existing scheduled task.
`schtasks /change /tn <Task Name> /ru <Username> /rp <Password>`	Modifies a scheduled task with a specified `name` to run under the `permissions` of the `user account` using the provided `password` for authentication.
`schtasks /delete`	Allows for the deletion of scheduled tasks.
`schtasks /delete /tn <Task Name>`	Deletes a scheduled task with the matching name.

Interacting With The Web

Command	Description
`Invoke-WebRequest -Uri "https://website-to-visit" -Method GET`	Utilizes Invoke-WebRequest to browse to a website and issue a GET request.
`Invoke-WebRequest -Uri "https://website-to-visit.html" -Method GET <PIPE> fl Images`	Issues a GET request to the site specified and then pipes the output to format a list of all image files listed in the site.

Command	Description
`Invoke-WebRequest -Uri "https://website-to-visit\file.ps1" -OutFile "C:\<filename>"`	Downloads a file from the website and writes it to disk with -Outfile.
`(New-Object Net.WebClient).DownloadFile("https://website-to-visit\tools.zip", "Tools.zip")`	Uses the .NET string Net.WebClient to download a file from the URL specified.

Event Log

Command	Description
`wevtutil el`	Uses the Windows Events Commandline utility to enumerate all log sources.
`wevtutil gl "name"`	Will gather config information about the log specified.
`wevtutil qe <Name> /c:5 /rd:true /f:text`	Query a log for events.
`wevtutil epl <Name> C:\system_export.evtx`	Export a Log.
`Get-WinEvent -ListLog *`	List all logging facilities using PowerShell cmdlets.
`Get-WinEvent -LogName 'Name' -MaxEvents 5 <PIPE> Select-Object -ExpandProperty Message`	View the messages of a specific log.
`Get-WinEvent -FilterHashTable @{LogName='Security';ID='4625 '}`	Query for a specific log by eventID.

Windows Registry

Registry Hives

Hives	Description

Hives	Description
`HKEY_LOCAL_MACHINE` (`HKLM`)	This subtree contains information about the computer's physical state, such as hardware and operating system data, bus types, memory, device drivers, and more.
`HKEY_CURRENT_CONFIG` (`HKCC`)	This section contains records for the host's current hardware profile. (shows the variance between current and default setups) Think of this as a redirection of the HKLM CurrentControlSet profile key.
`HKEY_CLASSES_ROOT` (`HKCR`)	Filetype information, UI extensions, and backward compatibility settings are defined here.
`HKEY_CURRENT_USER` (`HKCU`)	Value entries here define each user's specific OS and software settings. Roaming profile settings, including user preferences, are stored under HKCU.
`HKEY_USERS` (`HKU`)	The local computer's default User profile and current user configuration settings are defined under HKU.

Registry Commands

Command	Description
`Get-Item -Path Registry::<HIVE>\Path-to-key\ <PIPE> Select-Object -ExpandProperty Property`	See the sub-keys and properties of a registry key.
`Get-ChildItem -Path <HIVE>:\Path-to-key -Recurse`	Recursively search through a Key and all subkeys.
`Get-ItemProperty -Path Registry::<HIVE>\Path-to-key\key`	View the properties and values of a specific key.
`REG QUERY <HIVE>\PATH\KEY`	Use reg.exe to query the registry.
`REG QUERY <HIVE> /F "Password" /t REG_SZ /S /K`	Search for specific strings within the Registry hive.

Command	Description
`New-Item -Path <HIVE>:\PATH\ -Name KeyName`	Create a new Registry Key.
`New-ItemProperty -Path <HIVE>:\PATH\KEY -Name "ValueName" -PropertyType String -Value "C:\Users\htb-student\Downloads\payload.exe"`	Set a new Value pair within a registry Key.
`REG add "<HIVE>\PATH\KEY" /v access /t REG_SZ /d "C:\Users\htb-student\Downloads\payload.exe"`	Use Reg.exe to create a new key/value pair.
`Remove-ItemProperty -Path <HIVE>:\PATH\KEY -Name "name"`	Delete a key/value from the registry.

PowerShell Scripting

PowerShell Extensions

Extension	Description
`PS1`	The *.ps1 file extension represents executable PowerShell scripts.
`PSM1`	The *.psm1 file extension represents a PowerShell module file. It defines what the module is and what is contained within it.
`PSD1`	The *.psd1 is a PowerShell data file detailing the contents of a PowerShell module in a table of key/value pairs.

Commands For Building A Module

| **Command** | **Description** |
| `New-ModuleManifest \Path\<filename>` | This will create the initial manifest for a PowerShell module in the directory you specify. | | `ni <filename>.psm1 -ItemType File` | Creates a PowerShell module file. | | `Import-Module <modulename>` | Can be used to import a module into your PowerShell session or to specify modules to import when you run a PowerShell module. | | `$Variable = <input>` | Creates a callable variable and sets its value to the input specified. | | `function <name> { Tasks to run }` | Create a new function within a Module for use. | | `# Comment block` | Creates a one-line comment in a script or Module. | | `<# Comments #>` | Creates a multi-line comment block. Everything that falls within the <# #> regardless of line count will be considered a part of the comment block. | | `Export-`

HTB ACADEMY CHEATSHEET

`ModuleMember -Function <name> -Variable <variablename>` | Specifies that the functions and variables listed can be exported by other scripts, sessions, or modules. |