WINDOWS COMMANDS:

net user [username] [password] /add

this command is used to add, delete, and otherwise manage the users on the system. For command manual, type: '**net user /?**'

* adding dollar sign '**$**' to the username [**username$**] will hide the user from being viewed in the ' **net users** ' command

net localgroup [groupname] [username] /add (windows)

net localgroup: this command is used to add, delete, and manage local groups on the system. For command manual, type:'**net localgroup /?**'

wmic logicaldisk get name

Display each of the logical disk drives on the computer. If you wanted to see all the options for this command, including name, type: '**wmic logicaldisk get /?**'

wmic: short of "WMI Console" (Windows Management Instrumentation), located at **C:\Windows\System32\wbem\WMIC.exe** .

wmic qfe get caption, Description, HotFixID, InstalledOn

qfe : **quick-fix engineering** is term for the delivery of individual service updates to its OS and aplication programs. It is formerly called a **hotfix**, **qfe** can be used to describe both the method of delivering and applying a patch or fix, and also to refer to any individual fix. *These updates are not listed in the registry

HotfixID / knowledge base ID (KBID) : is a uniqe number for each patch/update release that help tracking it. The ID can be used to check known vulnerabilities which still unpached.

Caption : get url with more information about the **KBID**

Description : descriptiong about the **KBID**

InstalledOn : the date the **qfe** was installed.

wmic service get name,startname

this command display the name of the service (**name**), and its privileges (**startname**)

wmic service get name,displayname,pathname,startmode |findstr /i "[string]" |findstr /i /v "c:\windows\\" |findstr /i /v """

This command looks for unquoted services.

wmic service get [parameters] : display information about all services.

findstr : searchs for patterns of text in files, same as **grep** command in unix.

|findstr /i : ignore the case of the characters when searching for the string.

|findstr /v "[string]" : print only lines that don't contain a match.

accesschk.exe everyone -uqws c:*.dll

this command checks all the **dlls** that have **read** and **write** permissions.

accesschk.exe : a command-line tool which is part of **sysinternals**. **accesschk.exe** shows the effective permissions on securable objects, account rights for a user or group, or token details for a process

everyone : this is a group that includes all memebers of the authenticated users group as well as the built-in guest account, and several other built-in accounts such as **SERVICE**, **LOCAL_SERVICE**, **NETWORK_SERVICE**, etc…

-u : suppress errors.

-q : omit banner.

-w : show objects with write access

-s : recurse.

c:*.dll : search all **dlls** in C drive.

tasklist /v

this command displays a list of currently running processes on the system.

tasklist /v : displays verbose task information.

schtasks /query /fo LIST /v

This command lists all tasks scheduled to run on the system.

schtasks : in our case this command is used to list scheduled tasks, this command can be used with these parameters: add(**/create**), remove(**/delete**), start(**/run**), stop(**/end**), display(**/query**), change(**/change**) scheduled tasks.

/query : lists all the tasks scheduled to run.

/fo : specifies the output format. *there is 3 types of formats (TABLE,LIST,CSV)

LIST : formats the output as a list.

/v : (verbose)request a detailed display.

Invoke-dllInjection -ProcessID [ID] -Dll [path-to-dll]

This command can be executed on **powershell** in order to perform dll injection into the process id.

* **Invoke-dllInjection** : Invoke-DllInjection injects a Dll into an arbitrary process. It does this by using VirtualAllocEx to allocate memory the size of the DLL in the remote process, writing the names of the DLL to load into the remote process spacing using WriteProcessMemory, and then invoking LoadLibraryA in the context of the remote process.

-ProcessID : provied the id of the process you want to inject a dll into.

-Dll : provides the path of the dll to inject.

Mimikatz commands:

-**privilege::debug** : this command enters debugging mode wil local administrator privileges.

-**token::elevate** : this command performs privilege escalation on the system by impersonating a token (tokens of the system can be checked by ' **Token::list** ' command.

-**sekurlsa::logonpasswords** : this command lists all available provider credentials, it usually shows recently logged on user and computer credentials.

-**sekurlsa::logonpasswords [filename]** : dumps the lsass process into a file. (another way to dump lssas process is via sysinternals-procdump , or via task manager).

-**sekurlsa::minidump [filename]** : making mimikatz interact with the dumped file and checking the credentials instead of interacting with the current running process.

* the ' **sekurlsa** ' module interacts with protected memory. This module extracts passwords, keys, pin codes, tickets from the memory of lsass (Local Security Authority Subsystem Service). Using this module requires administrator access with debug rights or system rights.

-**lsadump::sam** : this command enable connection to the SAM database and dumping all the credentials of the local users, it will enable extracting the SAM file.

* the ' **lsadump** ' module interacts with the Windows Local Security Authority (LSA) to extract credentials. This moudle requires NT-Authority privilages.

-**log** : extracts mimikatz output to .txt file

-**event::drop** : patches the event servoce to avoid log creation.

-**event::clear** : clear all logs from event log without log cleared event (1102) being logged.

certutil -encode [filename] [certificated-filename]

This command certificate the file (e.g LSASS dump file) in order to export it to another system.

Usually used when the system prevent exporting important files out of it like LSASS.

Certutil: is part of certificate services and used to dump and display certification authority (CA) configuration information, configure Certificate Services, backup and restore CA components, and verify certificates, key pairs, and certificate chains.

-encode: encodes a file to base64.

LINUX COMMANDS:

adduser [username] , useradd [username]

Adding user to the system. The main difference between adduser and useradd is the adduser command asks for user details, sets up home directory and other settings, creats user directory in **/home** automatically.

adduser [username] [groupname]

adding the specified username to specified group.

groups

This command prints the groups the is user in.

fdisk -l

this command list the partition tables for the specified devices and then exit. It will mention the partitions exists in **/proc/partitions**.

mount [partition] [location]

This command is used to mount a filesystem.

All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at **/**. These files can be spread out over several devices. The mount command serves to attach the filesystem found on some device to the big file tree. the umount command will detach it again.

[**pratition**] : usually the largest **/dev/sda** is the wanted file system.

[**location**]: **/mnt** directory is usually in use for this case.

mount --rbind [source] [directory]

--rbind: recursively binds the source to a destination.

chroot [mounted-partition-path]

Change root direcroty to the mounted partition

sync

Shotcut of (sysnopsis) ,This command writes the data from memory out to disk.

*the kernel keeps data in memory to avoid doing (relatively slow) disk reads and writes. This improves performance, but if the computer crashes, data may be lost or the file system corrupted as a result. sync ensures that everything in memory is written to disk.

bcdedit /set {default} safeboot minimal

This command enables safe mode, (in safe mode Windows Defender will not check the hash of the files, this means that it won't detect any changes)

Bcedit (boot configuration data): used for managing boot configuration data store, it contains configuration parameters and controls how the operating system is booted.

Administative privilages are requierd in order to use this command. (must run CMD as admin)

/set: sets an entry option value in the boot configuration data store.

{default}: specifies a virtual identifier that corresponds to the boot manager for RAM disk devices. Check '**bcdedit /? ID**' for more info about identifiers.

Using ' **bcdedit /deletevalue {default} safeboot** ' will disable safe mode and the system will return to normal statup mode.

hashcat -m 1000 -o [cracked-hashes-file] --force [hashes-file] [wordlist]

This command is used to crack hashes, since hashes cannot be reversed the it is possible to guess the password by matching the hash with database that have passwords and their hashes, if the hashes are matched then it can guess the password.

-m : hash type (e.g ntlm, md5 sha-*..etc), check hashes types here.

-o : defines outfile for recovered hash (saves the crackes hashes in file).

--force: ignore warnings.

unshadow /etc/passwd /etc/shadow > [filename]

This command will basically combine the data of **/etc/passwd** and **/etc/shadow** to create new file with username and password details.

cat [path/to/logfile.log] | grep – Eav "[username|uid]" > [variable] && sudo mv [variable] [path/to/logfile.log]

This command is used for covering tracks, it grep all logs that are not related to [**username|uid**], saves them into [**variable**] and then overwriting the original log.file via mv command ' **mv [variable] [path/to/log.file]** '

Grep -Eav : **E**=interpret pattern as an extended regular expression, **a**=process binary file as text file, **v**=select all non-matching lines, in our case, everything that don't have the specified [**username|uid**].

grub-mkpasswd-pbkdf2

This command will generates a PBKDF2 password string suitable for use in a GRUB configuration file.

Pdkdf2 : this is a salted password hash, meaning that in addition to the password, the hash function takes as input another string, the salt, which is generated randomly when the password is set or modified.

*more info about setting grub password can be found here

grep -rnw "[path/to/file]" -e '[string]' --color

this command looks for text within files.

-r : search for text within files recursively.

-n : display line numbers.

-w : matches the whple word.

-e : using the specified pattern for matching, in our case is ' string '

--color : colorizing grep results.

ln -s [path/to/file] [filename] (symlink)

Make links between files. Providing the ability to link an actual file or directory from another location and work with it normally. *editing the linked file will not affect the file in the original location too.

-s : makes symbolic link instead of hard link.

find "[path/to/file]" -type f -exec grep -rnw "[string]" --color {} \;

Another way to find text within files.

*Find command search for files in a directory hierarchy.

-type : what to search for (**f**=file, **d**=directory, **l**=symbolic link, **s**=socket..etc)

-exec : the execute command run the specified command on the selected file, in our case the specified file [**path/to/file**] is found via **find** command and **grep** command will be executed on the file.

--color : colorizing grep results.

{} : if running find with **-exec** the '**{}**' expands to the filename of each file found, which means the executed command via **-exec**, in our case it is grep gets every found filename as an argument, it call the **grep** command once for each file found. In short, the **{}** is the output of find.

\; : ';' ends the command executed by **-exec**, in our case it ends the grep command. '****' escapes the '**;**' so the shell running find inside does not take '**;**' as its own special character.

find / -perm /[perm-number] -user [username] -exec ls -ldb {} \; 2>/dev/null

This command will find the files with special permissions across the system, it will list them and dump the errors to **/dev/null**.

-perm : permission. Must specifie permission number after.

-[perm-number] : **4000**=setuid, **2000**=setgid, **1000**=sticky.

-user : file owned by specified [**username**].

-exec [command] {} \; : already explained in previous commands.

Ls -ldb: **l**=long listing format, **d**=list directory entries, **b**=print octal escapes for nongraphic characters.

2>[path/to/file] : the number **2** represents the standard errors. This command send all errors to specified file, in our case **its /dev/null**.
*number **1** represents standard input, number **2** represents standard output.

find / -type f -writable 2 > /dev/null

this command can be used to search the system for files which users have write permissions.

-type f : aleady explained in previous commands.

-writeable : matches files which are writable.

*the **-writable** takes into account access control lists and other permissions artefacts which the **-prem** test ignores.

2 > [file-to-dump-errors] : alreay explained in previous commands.

top -u [user]

this command prints all running services and processes in the system.

-u : print running services and processes on the specified user.

chmod +t [dirname] , **Chmod u+s [filename]** , **chmod g+s [filename]** (linux)

Chmod changes file mode bits.

+t : gives the folder and it files special permission that prevent other users from making changes in.

u+s : this is the SetUID bit, it enforces user ownership on an executable file. When it is set, the file will execute with the file owner's user ID, not the person running it. In short, this will give the regular user priviliged permission to execute the file.

g+s : this is the SetGID bit, enforce the group ownership.

*when setting **UID/GID** on file, only the specified file will have special permissions, when it is set on directory every file in the directory will have special permissions.

crontab -u [username] -e , **crontab -u [username] -l**

This command creates cronjob (scheduled task).

-u : specifies a [**username**]

-e : updating the crontab

-l : list the current crontabs.

* the crontabs files are stored in **/var/spool/cron/crontabs**, a non privilaged user can't access this path, but can navigate and read files from **/etc/cron.***, which means that scheduled tasks can be checked via the ' **ls /etc/cron.*** ' command.

"--reference=[filename]"

This functionality allows copying file permissions and ownership to another files.

echo "[injection]" >> [/path/to/file]

this command used for writable file injection.

echo $PATH

this command shows us the path where the system looks for commands we run,

*the **PATH** is an enviromental variable that allows running programs from any directory, it also directs the shell to ordered list of directories where executable may be searched for. The PATH variable prevents us from having to write out the entire path to program on the CLI every time we run it.

strace [command]

this command runs the specified *command* until it exits. It intercepts and records the system calls which are called by a process and the signals which are received by a process. The name of each system call, its arguments and its return value are printed on standard error or to the file specified with the -o option. This is good command for debugging.

env (linux)

this command prints the variables of the current environment and their values.

*below is brief description of some commonly-used environment variables:

EDITOR= the default file editor to be used

HOME= the current user's home directory.

SHELL= the location of the current user's shell program

TERM= the current terminal emulation

PATH= pathnames to be searched when executing commands.

visudo (linux)

visudo edits the ***sudoers*** file in a safe mode (located at **/usr/sudoers**). visudo locks the ***sudoers*** file against multiple simultaneous edits, provides basic sanity checks, and checks for parse errors. If the ***sudoers*** file is currently being edited you will receive a message to try again later.

sudo -l , **sudo -V** , **sudo -u [username/id]** (linux)

this command shows options about sudo for the current user.

-l : list the allowed commands for the current user.

-V : checking the sudo version.

-u : run a specified command with the specified user.