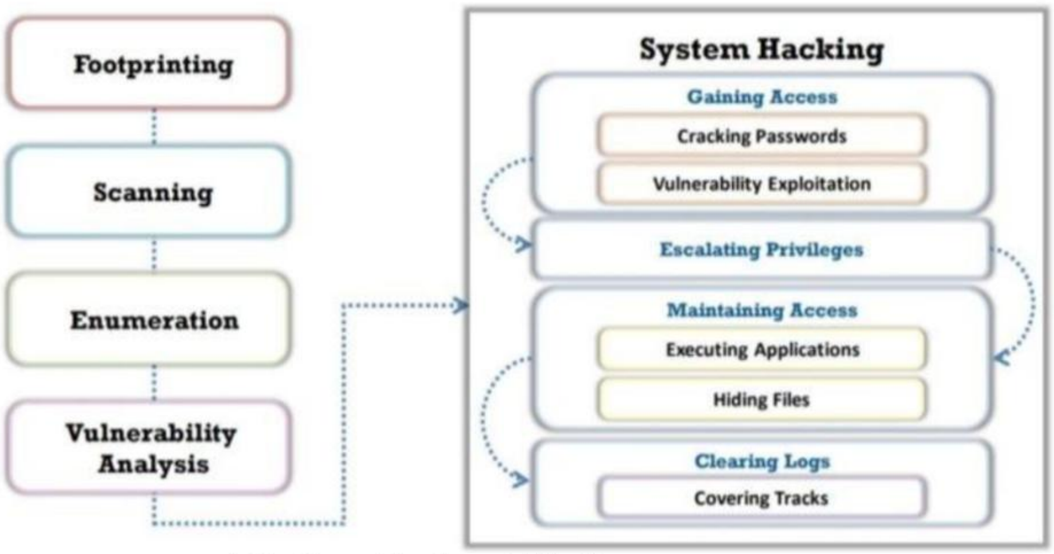
Hacking Methodologies

What are *hacking methodologies*?

- Hacking methodologies are repeatable, structured approaches attackers (or ethical testers) use to find, exploit, and analyze security weaknesses.
- Think of them as step-by-step playbooks that guide how someone moves from “I know nothing about the target” to “I have control or useful data” — and how defenders should think about stopping each step.

Hacking Methodology



This Shows The Steps Of Hacking to any system.

1. Footprinting (Reconnassciance / Information Gathering)

- **Definition**
  Passive or active information gathering to build a profile of a target (organization, domain, IP range, application, person).

- **Types**
  1. Passive (OSINT): search engines, social media, public repos, CT logs, company websites.
  2. Active: DNS queries, ping/WHOIS queries, web site probing.

3. Specialized: DNS footprinting, vendor/third-party mapping, wireless/physical footprinting.

❖ **Purpose**

- Map attack surface before active testing.
- Find exposed services, people to target, and publicly leaked secrets.
- Prepare social-engineering (phishing) or technical tests.

❖ **Common tools / example commands (awareness-only)**

- OSINT/web: search engines, Google Dorking, Hunter.io, LinkedIn, GitHub search.
- DNS/WHOIS: whois domain.com, dig domain.com ANY, nslookup or dig for records.
- Certificate logs: Certificate Transparency explorers / openssl s_client -connect host:443 (to view cert).
- Shodan / Censys for indexed internet devices.

❖ **Information collected**

Domain and subdomains, public IP ranges, hostnames, email formats, employee names, tech stack, public repos, certificates, leaked secrets, vendor relationships.

2. Scanning

❖ **Definition**

Active probing of hosts and networks to discover live hosts, open ports, running services and (optionally) versions.

❖ **Types**

1. Host discovery (ping, ARP scans).
2. Port scanning (TCP SYN, TCP Connect, UDP).
3. Service & version detection (banner grabbing).
4. OS fingerprinting.
5. Web / API scanning and mass scans.

❖ **Purpose**

Determine what services are exposed and where to focus deeper testing.
Produce an inventory of reachable services and entry points.

❖ **Common tools / example commands (awareness-only)**

- nmap (host discovery, port scan, service/version detection).
- masscan for large-scale fast scans.
- arp-scan for local LAN host discovery.
- curl or http for simple HTTP checks.
- Web scanners: basic crawlers or site-mapping tools (crawler/gobuster/dirb for directory discovery).

- Cloud provider consoles / security groups for cloud exposure checks.

❖ **Information collected**
Which IPs/hosts are alive, which TCP/UDP ports are open, identified services, approximate versions, reachable web endpoints.

3. Enumeration

❖ **Definition**
Targeted querying of discovered services to extract detailed data (user lists, shares, directories, APIs, database schema, service-specific info).

❖ **Types**
1. Network service enumeration (SMB, LDAP, SNMP).
2. Web app/API enumeration (endpoints, parameters, directories).
3. User/account enumeration (username discovery via services).
4. Resource enumeration (shares, printers, databases).

❖ **Purpose**
Turn a list of open services into actionable information an attacker might use (usernames, share names, API endpoints).
Find misconfigurations and paths that lead to access.

❖ **Common tools / example commands (awareness-only)**
- SMB/Windows: smbclient, enum4linux (to list shares, users).
- LDAP: ldapsearch for directory info.
- Web: gobuster / dirb / crawlers to find directories and hidden endpoints.
- SNMP: snmpwalk (if community strings exposed).
- Specialized scripts and service-specific enumeration tools.

❖ **Information collected**
Usernames and groups, share names and permissions, accessible directories/files, API endpoints/parameters, service configuration details.

4. Vulnerability Analysis (Vulnerabilities)

❖ **Definition**
Identify and prioritize weaknesses in discovered services and systems (unpatched software, weak configs, default creds, vulnerable components).

❖ **Types**
1. Unauthenticated scanning (external view).
2. Authenticated scanning (with credentials; deeper checks).

3. Configuration checks (TLS/SSH config, weak ciphers).
4. Dependency/component scanning (libraries, frameworks).

❖ **Purpose**
Find likely exploitable issues and rank them for remediation.
Provide evidence of risk to prioritize patching and configuration fixes.

❖ **Common tools / example commands (awareness-only)**

- Vulnerability scanners: Nessus, OpenVAS, Qualys (awareness only).
- CVE databases / NVD for research.
- Dependency scanners for code (software composition analysis).
- Manual checks for weak TLS, default credentials, exposed admin consoles (e.g., check web admin endpoints).

❖ **Information collected**
Identified CVEs, misconfigurations, outdated versions, weak/absent controls (no MFA), default or weak credentials, severity/prioritization data.

5. System Hacking (Gaining Access / Exploitation)

❖ **Definition**
Using identified weaknesses (technical or human) to obtain an initial foothold — a session, shell, valid credentials or a compromised account.

❖ **Types / sub-steps**

1. Credential-based compromise (password reuse, phishing).
2. Exploitation of service or application vulnerabilities.
3. Web-app exploitation leading to auth bypass or RCE (in authorized/controlled labs).
4. Social engineering / phishing to obtain credentials or execution.

❖ **Purpose**
Obtain an initial foothold so an attacker can escalate privileges, move laterally, and access valuable assets.

❖ **Common tools / example categories (awareness-only)**

- Post-scan tooling and frameworks used in authorized pentests (e.g., exploitation frameworks, password cracking toolkits, credential auditing tools).
- Phishing platforms and social-engineering toolkits (for authorized training).
- Endpoint analysis and EDR tools for defenders to detect and remediate.

❖ **Information collected / outputs**

Valid credentials, active sessions, shells or remote access tokens, evidence of successful compromise (for authorized testing): which account was compromised, what access it provides.

Step involved in System hacking

1. Initial Recon / Context review (target-specific)

- What happens: Attacker learns about the specific system: OS, services, applications, users, and exposure surface.
- Common technique categories: passive OSINT, banner observation, service/version info from prior scans.
- Defenses: limit public exposure, harden service banners, maintain up-to-date asset inventory.

2. Vulnerability selection / Attack planning

- What happens: From known services/versions the attacker chooses likely weaknesses to try (patch gaps, weak auth, misconfigs).
- Common technique categories: mapping service → known CVEs, configuration weaknesses, leaked creds.
- Defenses: patch management, configuration baselines, credential hygiene, vulnerability scanning and triage.

3. Initial Access / Exploitation attempt

- What happens: Attacker attempts to obtain a foothold — e.g., a valid session, API token, file upload point, or remote command execution.
- Common technique categories: credential use (phishing/reuse), exploiting vulnerable services, web-app flaws, or social engineering.
- Defenses: MFA, input validation, WAF, phishing training, remove default/weak creds, strong authentication.

Safety note: I won't provide step-by-step exploit instructions or payloads. Focus on detection & prevention.

4. Establishing a Foothold / Post-exploit setup

- What happens: Once access exists, attacker stabilizes it: keep a session alive, create persistence, or obtain tokens.
- Common technique categories: backdoors, scheduled tasks, saved credentials, creating stealth accounts.
- Defenses: endpoint protection (EDR), application allowlisting, restrict scheduled task rights, detect new accounts and unusual persistence mechanisms.

5. Privilege Escalation

- What happens: Attacker tries to move from limited user to admin/root to control more of the host.
- Common technique categories: exploiting local privilege escalation bugs, abusing misconfigured services, credential harvesting.
- Defenses: patch hosts, apply least privilege, use secure configuration (no local admin by default), EDR alerts on suspicious token use.

6. Lateral Movement & Internal Recon

- What happens: From the compromised host the attacker explores the internal network to find more valuable targets.
- Common technique categories: credential reuse, pass-the-hash/token abuse, exploitation of internal services, SMB/SMBv1 abuse (historically).
- Defenses: network segmentation, restrict lateral protocol access, use privileged access workstations, monitor for unusual internal authentication patterns.

7. Data Access & Exfiltration (objective-focused)

- What happens: Attacker locates sensitive data and moves it out (or encrypts it for ransom).
- Common technique categories: bulk reads, database dumps, compressing and transferring data to external hosts.
- Defenses: data classification, DLP, restrict egress, inspect TLS termination points, anomaly detection on large reads or transfers.

8. Persistence hardening / Maintain access

- What happens: Attacker strengthens ways to return despite remediation attempts.

- Common technique categories: multiple backdoors, scheduled jobs, abusing legitimate services for command-and-control.
- Defenses: rotate keys/credentials, rebuild compromised hosts from clean images, proactive hunting.

9. Covering Tracks & Anti-forensics

- What happens: Attacker attempts to erase indicators (logs, artifacts) to delay detection and investigation.
- Common technique categories: log deletion/modification, timestomping, removing shell history.
- Defenses: central immutable logging, remote log forwarding, tamper-evident storage, SIEM alerts on missing/altered logs.

10. Cleanup / Exit (attacker choice)

- What happens: Either attacker leaves remnants (for future return) or removes traces; defenders may recover and remediate.
- Defenses: incident response playbook, forensics, rotate credentials, rebuild systems, post-incident lessons & stronger controls.