Ethics & Security

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

AUTHORIZED SECURITY TESTING

Authorize Scope and rules → **Assess** Gather evidence → **Remediate** Reduce risk → **Verify** Retest controls

01 Information security and ethical hacking

Authorization, scope and risk

- **Written authorization** establishes permission to test. **Scope** defines systems and boundaries; **rules of engagement** define permitted techniques, timing, escalation contacts and stop conditions. An **NDA** governs disclosure of confidential information.
- **Black box** testing starts with little internal knowledge; **white box** testing supplies extensive design or source information; **gray box** testing supplies partial knowledge or credentials. Internal versus external describes the testing vantage point.
- **Confidentiality** prevents unauthorized disclosure; **integrity** protects against unauthorized alteration; **availability** preserves timely access. **Authentication** verifies identity, **authorization** grants actions, and **accountability** links actions to identities.
- **Threats** can exploit **vulnerabilities** to harm assets. Assess likelihood and impact in business context. **Inherent risk** precedes controls; **residual risk** remains afterward. Owners may mitigate, avoid, transfer or explicitly accept risk within tolerance.
- **Least privilege**, separation of duties, secure defaults and **defense in depth** reduce exposure. Combine administrative, technical and physical controls; a perimeter firewall cannot replace endpoint hardening or access governance.

Attack models and adversary goals

- The familiar hacking sequence is **reconnaissance → scanning/enumeration → gaining access → maintaining access → covering tracks**. An authorized engagement finishes with evidence, cleanup, reporting and retesting under its agreed rules.
- The **Cyber Kill Chain** orders reconnaissance, weaponization, delivery, exploitation, installation, command and control, and actions on objectives. Building a payload is weaponization; sending it is delivery; executing an exploit is exploitation.
- **MITRE ATT&CK** maps adversary tactics, techniques and procedures. A **tactic** explains the objective; a **technique** explains how it is pursued. Persistence, privilege escalation, defense evasion and lateral movement describe different objectives.
- The **Diamond Model** connects adversary, infrastructure, capability and victim. An **APT** pursues sustained objectives through repeated access, persistence, internal discovery and expansion; long dwell time alone does not identify its sponsor.
- **White hats** act with authorization; **black hats** pursue malicious goals; **gray hats** may test without permission. **Hacktivists** pursue political or social aims; state-sponsored groups serve national objectives; script kiddies rely on others' tools.

Ethics & Security / Reconnaissance & Assessment

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

AUTHORIZED SECURITY TESTING

Authorize Scope and rules → **Assess** Gather evidence → **Remediate** Reduce risk → **Verify** Retest controls

Governance, intelligence and response

- **ISO/IEC 27001** specifies an information security management system; **27002** supplies control guidance; **27005** addresses information security risk management. **PCI DSS** concerns payment-card data, **HIPAA** health information, and **SOX** financial reporting controls.
- **Strategic intelligence** supports leadership decisions; **tactical intelligence** describes adversary techniques; **operational intelligence** adds campaign context; **technical intelligence** supplies indicators such as hashes and domains. Correlate indicators with behavior and confidence.
- **Incident response** prepares people and procedures, detects and analyzes incidents, contains damage, eradicates causes, restores service and captures lessons. Triage considers severity, scope, affected assets and business impact.
- Correlate **identity, endpoint, DNS, proxy, network and cloud logs** on synchronized clocks. Protect centralized records from alteration. **MTTD** measures detection time; define whether **MTTR** means response, repair or recovery before comparing it.
- **AI-assisted security** can scale discovery, prioritize anomalies and forecast patterns from data. Validate its findings with evidence and human review; automation can reproduce bad assumptions, miss unfamiliar behavior and exceed scope if uncontrolled.

02 Reconnaissance, scanning and vulnerability analysis

Public information and footprinting

- **Passive reconnaissance** uses third-party or already available information; **active reconnaissance** sends probes to the target. Scanning, service interrogation and zone-transfer requests are active even when they are quiet.
- Search operators such as **site:**, **inurl:**, **intitle:** and **filetype:** narrow indexed material. Archives, certificate transparency, job postings, repositories and public documents can reveal historic endpoints, technologies and accidental secrets.
- **WHOIS/RDAP** exposes available registration and allocation records; privacy services may conceal contacts. RIR regions include **ARIN** for North America, **RIPE NCC** for Europe and much of the Middle East, **APNIC**, **LACNIC** and **AFRINIC**.
- **Email headers** reveal relay paths through Received fields; trust headers added by known infrastructure. Display names, sender-supplied dates and IP geolocation are weak identity evidence. SPF, DKIM and DMARC provide separate sender-domain signals.

Reconnaissance & Assessment

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

FROM EXPOSURE TO VERIFIED FIXES

Discover Assets and services → **Enumerate** Accessible detail → **Assess** Validate and prioritize → **Verify** Confirm remediation

- **Maltego** maps relationships; **theHarvester** collects public identity and infrastructure signals; **Recon-ng** organizes reconnaissance; **Sherlock** correlates usernames; **CeWL** builds site-derived wordlists. Public social profiles and geotags can expose roles and routines.
- The **deep web** is content absent from ordinary search indexes; the **dark web** uses specialized access networks. Neither guarantees anonymity or reliable evidence. Limit collection to authorized sources and handle exposed personal data minimally.

DNS records and disclosure

- **A/AAAA** map names to IPv4/IPv6 addresses; **CNAME** aliases names; **MX** identifies mail exchangers; **NS** identifies authoritative servers; **PTR** supports reverse lookups; **TXT** carries text-based metadata and policies.
- **SOA** describes zone authority and administrative parameters, including serial and refresh values. A **zone** is an administered portion of the namespace represented by resource records; it need not equal an entire domain tree.
- **AXFR** transfers a complete zone over TCP. Restrict transfers to approved secondary servers. **dig** and **nslookup** query DNS; **DNSRecon** and **DNSdumpster** support discovery through different query and data sources.
- **Cache snooping** asks whether a resolver holds a record, often with recursion disabled using **+norecurse**. A cached answer is a limited usage clue; an empty answer does not prove nobody visited the domain because TTL, cache policy and resolver choice matter.
- **Cache poisoning** inserts false cached records; **DNS hijacking** changes resolver settings or authoritative data. **Split DNS** exposes different internal and external views. Protect registrar access, authoritative updates and recursive resolver access.
- **DNSSEC** authenticates signed DNS data and detects alteration through a trust chain. It does not encrypt queries or prevent DNS tunneling. **Fast flux** rotates endpoint addresses; DNS tunneling carries encoded command or exfiltration data in DNS traffic.

Host discovery and fingerprinting

- **TCP** establishes a connection with SYN → SYN/ACK → ACK. FIN normally closes a direction; RST aborts a connection; ACK acknowledges sequence space. **UDP** has no equivalent connection handshake.
- **ARP discovery** works on the local IPv4 link; **ICMP Echo**, TCP SYN/ACK probes and UDP probes can discover routed hosts. A missing reply may reflect filtering, loss or rate limiting rather than an unused address.
- **Nmap -sn** performs discovery without a port scan; **-Pn** skips the normal discovery decision and treats targets as up for scanning. **-PS**, **-PA**, **-PE** and **-PP** select SYN, ACK, ICMP Echo and timestamp discovery probes.
- **Active OS fingerprinting** compares crafted-probe responses. **Passive fingerprinting** observes normal traffic. TTL, window size, TCP options and sequence behavior are clues; hops, middleboxes and configuration prevent a single TTL from proving an OS.

Reconnaissance & Assessment

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

FROM EXPOSURE TO VERIFIED FIXES

Discover Assets and services → **Enumerate** Accessible detail → **Assess** Validate and prioritize → **Verify** Confirm remediation

- **Banner grabbing** and **-sV** identify service software and versions; **-O** requests OS detection. A port number suggests a service but does not prove it. Correlate application responses, configuration and authenticated inventory.
- **Traceroute** varies TTL or hop limit and interprets expiry responses; implementations may use UDP, ICMP or TCP. Repeated hops suggest shared routing infrastructure, while filtered responses leave gaps rather than proving no router exists.

Port scans and interpretation

- **SYN scan (-sS)** is half-open: SYN/ACK suggests open, RST suggests closed, and silence or selected ICMP errors suggest filtering. **Connect scan (-sT)** uses the operating system to complete connections. Both can be detected and logged.
- **ACK scan (-sA)** maps filtering: RST indicates unfiltered; silence or filtering errors indicate filtered. It does **not** establish whether the probed port is open.
- **NULL (-sN)** sends no TCP flags; **FIN (-sF)** sends FIN; **Xmas (-sX)** sends FIN, PSH and URG. On compatible stacks, RST indicates closed; silence is **open|filtered**. Other stacks may respond differently.
- **UDP scan (-sU)** treats an application response as open and ICMP type 3/code 3 as closed. Silence is generally **open|filtered**; service-specific probes and retries can clarify it. **SCTP INIT (-sY)** tests SCTP services.
- **Idle scan (-sI)** infers target behavior through a suitable quiet host's predictable IPID changes. **Decoys (-D)** add apparent sources; **fragmentation (-f)** changes packet structure; timing options change rate. None guarantees stealth or bypass.
- **MAC spoofing** affects the local link; remote routed systems normally see their adjacent router's MAC. **--spoof-mac 0** requests a randomized MAC for supported raw-packet activity. IP spoofing also disrupts return paths and does not itself create anonymity.

Service enumeration and ports

- **SMB** commonly uses TCP **445**, or TCP **139** through NetBIOS; NetBIOS naming and datagrams use UDP **137/138**. **nbtstat** queries name tables; **RID cycling** maps identifiers to accounts. Anonymous **IPC$ null sessions** depend on permissions and configuration.
- **LDAP 389** exposes directory queries; use TLS through **StartTLS** or **LDAPS 636**, plus authenticated binds and ACLs. **Global Catalog 3268/3269** supports forest-wide searches. **ldapsearch** discovers RootDSE naming contexts and queries a chosen base DN.
- **SNMP 161/UDP** exposes managed objects; traps commonly use **162/UDP**. **MIBs** define objects addressed by **OIDs**. **snmpwalk** traverses accessible objects. SNMPv1/v2c community strings lack encryption; **SNMPv3 authPriv** adds authentication and privacy.

Reconnaissance & Assessment / Systems & Malware

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

FROM EXPOSURE TO VERIFIED FIXES

Discover Assets and services → **Enumerate** Accessible detail → **Assess** Validate and prioritize → **Verify** Confirm remediation

- **SMTP 25** commands **VRFY**, **EXPN** and recipient checks can disclose users or lists. Restrict unnecessary verification and avoid revealing account existence. **FTP 21**, **Telnet 23** and **TFTP 69/UDP** require special attention to plaintext or weak access controls.
- **NTP 123/UDP** may disclose peers and host details. **ntpq -p** lists peers and synchronization metrics; **ntptrace** follows time sources; legacy **ntpdc** queries daemon control data. Restrict management queries separately from ordinary time synchronization.
- **RPC 135**, **NFS 2049**, **SSH 22**, **DNS 53**, **HTTP 80**, **HTTPS 443** and **SQL Server 1433** are useful service clues. SQL Server Browser commonly uses **1434/UDP**. Confirm protocol and transport before drawing conclusions.

Assessment, prioritization and reporting

- **Vulnerability management** cycles through asset inventory and baselines, scanning, risk assessment, remediation, verification and monitoring. A **penetration test** demonstrates selected attack paths and impact within scope; a scan alone does not prove exploitability.
- **Credentialed scans** inspect local patches, policies and permissions; **unauthenticated scans** show externally observable exposure. Host, network, application and wireless assessments have different visibility. Passive assessment observes traffic; active assessment sends probes.
- **Inference-based scanning** discovers protocols and services, then selects relevant checks. **Nessus**, **OpenVAS/Greenbone** and **Qualys** assess vulnerabilities; **Nikto** targets web-server exposure; **Burp Suite** supports interactive web testing. Tool output requires validation.
- **CVE** identifies a published vulnerability; **CWE** classifies a weakness; **CVSS** estimates technical severity, not complete business risk. In **CVSS 3.1**, base metrics describe intrinsic qualities, temporal metrics changing conditions, and environmental metrics local context.
- CVSS 3.1 bands are **0 none**, **0.1–3.9 low**, **4.0–6.9 medium**, **7.0–8.9 high**, **9.0–10 critical**. Prioritize using exposure, active exploitation, asset criticality, impact and available controls alongside the score.
- **Reports** separate scope and methodology, findings and affected assets, evidence and impact, severity and remediation. Apply trusted patches through tested change control and rescan. **Virtual patching** filters exploit paths temporarily; it does not repair vulnerable code.

03 System hacking and malware

Password attacks and credential protection

- **Brute force** explores candidate combinations; **dictionary attacks** try wordlists; **hybrid/rule-based attacks** mutate likely words; **combinator/PRINCE** approaches combine candidate elements. **Rainbow tables** precompute hash relationships and lose their broad reuse advantage with unique salts.
- **Password spraying** tries a few likely passwords across many users; **credential stuffing** reuses breached username/password pairs. Distributed low-rate attempts can evade simple per-IP thresholds; correlate attempts across accounts, devices and time.

Systems & Malware

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

HOST COMPROMISE AND DEFENSE

- **Credentials** — Protect identities
- **Privileges** — Limit authority
- **Persistence** — Monitor changes
- **Behavior** — Correlate evidence

- **Online guessing** interacts with authentication controls; **offline cracking** tests stolen hashes without server lockouts. **Hashcat** and **John the Ripper** support offline recovery; **Hydra** primarily tests online login services. Hardware acceleration improves throughput, not password entropy.
- Store passwords using a unique random **salt** and a deliberately costly password KDF such as **Argon2id**, **scrypt**, **bcrypt** or appropriately configured **PBKDF2**. A separate protected **pepper** adds another secret; fast unsalted hashes enable efficient guessing.
- Use long unique passwords, password managers, **phishing-resistant MFA**, rate limits, breached-password screening and generic login errors. Smart card plus PIN combines possession and knowledge; two knowledge checks are still one factor type.

Windows authentication and lateral movement

- Local Windows account hashes reside in the **SAM**; **LSASS** handles authentication material in memory. **NTLM** hashes and Kerberos tickets differ from plaintext passwords. Legacy **LM** storage is weak and generally disabled on modern Windows.
- **Pass-the-hash** uses a reusable NTLM hash to authenticate without recovering the password; **pass-the-ticket** reuses Kerberos tickets. Restrict privileged logons, unique local administrator credentials, credential exposure and unnecessary remote authentication paths.
- **Kerberoasting** requests a service ticket for an SPN and attacks the service-account-encrypted portion offline. Prefer long managed service-account passwords, strong Kerberos encryption and least privilege; monitor unusual service-ticket requests.
- **LLMNR/NBT-NS poisoning** supplies false local name-resolution replies to attract authentication. **NTLM relay** forwards an authentication exchange to another service; it differs from cracking a captured response. SMB signing, channel binding and reduced NTLM use disrupt relevant relay paths.
- **PetitPotam** is associated with coerced authentication through EFSRPC and relay chains such as exposed AD CS enrollment. **Internal Monologue** abuses local authentication interfaces to obtain responses; it is distinct from directly dumping LSASS memory.
- **Lateral movement** reaches other hosts or identities; **pivoting** routes access through a foothold. Correlate remote logons, service creation, scheduled tasks and east-west traffic; network segmentation limits how far stolen credentials can travel.

Systems & Malware

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

HOST COMPROMISE AND DEFENSE

- **Credentials** — Protect identities
- **Privileges** — Limit authority
- **Persistence** — Monitor changes
- **Behavior** — Correlate evidence

Privilege escalation and persistence

- **Vertical escalation** gains higher privileges; **horizontal escalation** accesses another peer's resources. Review service accounts, privileged groups, weak ACLs, exposed secrets, Linux **sudo/SUID** settings and writable scheduled-task or cron locations.
- **DLL search-order hijacking** places an attacker-controlled library in a searched location; code runs with the loading process's rights. Use explicit trusted paths and protected directories. **UAC bypass** abuses elevation behavior and is distinct from a kernel exploit.
- **Buffer overflow** writes beyond allocated memory; **integer overflow** can undermine size calculations. **DEP/NX** restricts executable data pages; **ASLR** randomizes locations; stack canaries detect some corruption. These mitigations complement bounds-safe programming.
- A **TOCTOU race** changes a resource between checking it and using it. Use atomic operations and stable file handles rather than separate pathname checks. Client-side validation and hidden fields cannot enforce server-side security decisions.
- **Persistence** may use services, scheduled tasks, startup folders, registry Run keys, cron, webshells or firmware. **Meterpreter** is a post-exploitation agent; commands such as getuid, getsystem, hashdump and keyscan_start have different purposes and privilege requirements.
- **Anti-forensics** includes log tampering, timestamp changes, hiding and artifact wiping. Deleting a filename does not guarantee erasure; overwriting does not guarantee removal from snapshots or remapped storage. Preserve evidence and remove test artifacts through agreed cleanup.

Malware families and concealment

- A **virus** infects a host file or boot structure; a **worm** self-propagates; a **Trojan** disguises malicious functionality as useful software. A **RAT** provides remote control; a **botnet agent** accepts coordinated commands from a controller or peer network.
- **Ransomware** denies access or extorts victims, often with data theft. **Spyware** monitors activity; **keyloggers** capture input through software, browser scripts or hardware; **PUAs/adware** introduce unwanted behavior. A **logic bomb** activates on a condition or time.
- **Multipartite viruses** infect multiple target types such as boot areas and files. **Polymorphic malware** varies its encoded appearance or decryptor; **metamorphic malware** rewrites equivalent code. A **stealth virus** intercepts observations to conceal changes.

Systems & Malware / Network & Perimeter

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

HOST COMPROMISE AND DEFENSE

- **Credentials** — Protect identities
- **Privileges** — Limit authority
- **Persistence** — Monitor changes
- **Behavior** — Correlate evidence

- **Rootkits** conceal access at user, kernel, boot, hypervisor or firmware layers. A **bootkit** alters startup execution; a **hypervisor rootkit** mediates the OS through virtualization; firmware implants may survive OS reinstalls. Recovery may require trusted rebuilding or firmware replacement.
- A **wrapper/binder** combines a legitimate program and payload; a **dropper** installs embedded malware; a **downloader** retrieves it; a **packer** transforms its representation. These delivery and concealment roles can coexist in one sample.

Fileless behavior and malware analysis

- **Fileless attacks** abuse memory, scripts and trusted tools such as PowerShell, WMI or BITS. **Living off the land** uses legitimate capabilities for malicious purposes. The absence of a new executable does not mean the absence of artifacts or persistence.
- Collect **PowerShell script-block logs**, process ancestry, command lines, scheduled tasks, registry changes and network activity. Retain AMSI and endpoint telemetry; detect unusual script/tool combinations and restrict unnecessary execution capabilities.
- **Static analysis** examines hashes, strings, headers, imports and code without running it. **Dynamic analysis** observes execution, files, registry, processes and traffic in an isolated disposable environment. Simulated services can reveal behavior without production connectivity.
- **YARA** matches file or memory patterns; **Snort** rules primarily match network activity. **yarGen** helps select sample strings while filtering common benign strings. Test rules against both malicious and benign samples to reduce false positives.
- Sandboxes may be evaded through environment checks, delayed execution or user-activity requirements. Combine behavioral analysis, memory inspection, integrity baselines and network evidence. Isolate affected hosts and protect tested offline or immutable backups during recovery.

04 Network and perimeter security

Sniffing and switched-network attacks

- **Promiscuous mode** accepts frames delivered to a wired interface regardless of destination; it does not make a switch forward every frame to that port. **Monitor mode** captures supported wireless frames. Authorized **SPAN/mirroring** provides selected switch traffic visibility.
- **Wireshark** decodes packets; display filters narrow existing captures; **Follow TCP Stream** reconstructs conversations. **tcpdump/WinDump** capture traffic; **tcptrace** analyzes TCP traces. TLS payloads remain encrypted unless suitable session secrets or endpoint access are available.
- **ARP poisoning** sends false IPv4-to-MAC mappings to redirect local traffic. Unsolicited replies and mapping changes are clues, but redundant gateways and proxy ARP can be legitimate. **Dynamic ARP Inspection** validates ARP using trusted bindings and policy.

Network & Perimeter

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

LAYERED NETWORK PROTECTION

- **Link** — Switch safeguards
- **Transport** — Secure sessions
- **Perimeter** — Enforce policy
- **Detection** — Observe and respond

- **MAC flooding** overloads a switch's forwarding table and may cause unknown-unicast flooding. **Port stealing** manipulates MAC learning toward an attacker port. **Port security** limits learned addresses; flooding does not inherently defeat VLAN separation.
- **DHCP starvation** exhausts leases with fabricated clients; a **rogue DHCP server** supplies malicious gateway or DNS settings. **DHCP snooping** permits server replies only on trusted ports and builds bindings; rate limits and port controls address starvation.
- **STP attacks** inject superior bridge information. **Root Guard** preserves intended root placement; **BPDU Guard** protects edge ports from BPDUs. Prevent **VLAN hopping** with explicit access/trunk settings and careful native-VLAN design. Use encryption and NAC alongside switch controls.

Social engineering and physical access

- **Phishing** uses deceptive messages; **spear phishing** targets selected people; **whaling** targets senior personnel; **BEC** exploits trusted business communications, sometimes through a genuinely compromised mailbox. **Vishing** uses voice and **smishing** uses SMS.
- **Clone phishing** copies a genuine message while replacing its link or attachment. **Search-engine phishing** attracts victims through deceptive search results. **Pharming** redirects name resolution or traffic; a **watering hole** compromises a site the intended victims frequently visit.
- **Pretexting/impersonation** invents a credible role or story; **elicitation** draws information from conversation; a **honey trap** builds emotional trust; **quid pro quo** offers a benefit for cooperation. Authority, urgency and familiarity increase pressure to bypass checks.
- **Baiting** uses attractive media or rewards; **scareware** promotes a fake remedy through fear; **hoax messages** spread false warnings. **Shoulder surfing**, eavesdropping and **dumpster diving** collect information from people, surroundings and discarded material.
- **Tailgating** follows authorized people through controlled entry. Verify visitors and escort them; use locks, badges, vestibules and monitoring appropriate to the site. **Bollards** resist vehicle intrusion; door controls address a different physical threat.
- Verify requests through a **known independent channel**, inspect actual link destinations and report suspicious activity. Caller ID, display names, corporate logos and public project details do not establish identity. Train staff and require dual approval for sensitive transfers or access changes.

Network & Perimeter

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

LAYERED NETWORK PROTECTION

- **Link** — Switch safeguards
- **Transport** — Secure sessions
- **Perimeter** — Enforce policy
- **Detection** — Observe and respond

Denial of service and resilience

- **DoS** depletes availability; **DDoS** distributes the sources. **Volumetric** attacks exhaust bandwidth; **protocol/state attacks** exhaust network or connection handling; **application-layer attacks** exhaust expensive operations. Botnets may turn internal IoT devices into attack sources.
- **SYN floods** leave half-open TCP connections waiting for final ACKs. SYN cookies/proxies, connection controls and upstream protection reduce impact. **UDP/ICMP floods** can consume bandwidth or processing without incomplete TCP handshakes.
- **Reflection** sends replies to a spoofed victim address; **amplification** makes replies larger than requests. DNS and NTP can be abused when exposed. Restrict open reflectors and apply ingress/egress source validation; distributed reflection is **DRDoS**.
- **HTTP GET/POST floods** consume application work; **Slowloris** dribbles incomplete headers; **slow POST** dribbles bodies. Request deadlines, minimum transfer rates, connection limits, caching and reverse proxies reduce low-bandwidth connection exhaustion.
- **Ping of Death** uses oversized or malformed IP reassembly; **Teardrop** abuses overlapping fragments. **Pulse-wave** attacks alternate bursts and quiet periods. HTTP/2 stream churn or implementation leaks can exhaust resources; patch protocol stacks and bound per-client work.
- **Traffic scrubbing** removes attacks upstream before a saturated link. WAFs address HTTP abuse; load balancing distributes work; rate limiting bounds demand. Blackholing drops legitimate traffic too. Use multi-layer mitigation, provider coordination and tested graceful degradation.

Session hijacking and lifecycle defenses

- **Session sidejacking** captures exposed session tokens; **replay** presents a valid stolen token again; **prediction** exploits low entropy or patterns. A bearer token can confer access without repeating MFA. Capturing TLS ciphertext alone does not reveal a reusable application token.
- **Session fixation** causes a victim to authenticate using a token already known to the attacker. Regenerate identifiers after login and privilege changes. **Session donation** instead lures a victim into using an attacker-owned account, potentially storing the victim's data there.
- **Active TCP hijacking** injects traffic within accepted sequence/acknowledgment windows and may desynchronize the client. **Blind hijacking** lacks direct traffic visibility. Forged **RST** packets can terminate sessions without taking them over; ACK anomalies are clues, not proof.

Network & Perimeter

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

LAYERED NETWORK PROTECTION

- **Link** — Switch safeguards
- **Transport** — Secure sessions
- **Perimeter** — Enforce policy
- **Detection** — Observe and respond

- **Man-in-the-browser** malware operates at a compromised endpoint; **MitM** operates along the communication path. Authenticated TLS protects transit, while endpoint compromise can alter data before encryption. Sender-constrained or channel-bound tokens can reduce replay where supported.
- Use **Secure** cookies for HTTPS transport, **HttpOnly** to block script reads, and **SameSite** to constrain cross-site cookie sending. HttpOnly does not prevent XSS from making authenticated requests. Avoid session IDs in URLs, logs and referrers.
- Generate tokens with a **cryptographically secure RNG**; enforce idle and absolute expiry, revoke sessions on logout and compromise, and protect refresh tokens. Use end-to-end HTTPS, certificate validation and HSTS; add CSRF controls and reauthentication for sensitive actions.

Firewalls, detection and deception

- **Packet filters** evaluate headers; **stateful firewalls** track connections; **circuit gateways** mediate sessions; **application proxies/WAFs** inspect supported application traffic. NAT changes addressing and is not a substitute for an explicit access policy.
- **NIDS** monitors network traffic; **HIDS/EDR** observes host activity. **IDS** alerts; inline **IPS** can block. **Signature detection** recognizes known patterns; **anomaly detection** finds deviations; stateful protocol analysis checks expected protocol behavior.
- **True positive** means a correct malicious alert; **false positive** means a benign event alerted; **false negative** means an attack was missed; **true negative** means benign activity was correctly unflagged. Tune using representative traffic and incident evidence.
- **Evasion** exploits differences between sensor and endpoint interpretation. Fragmentation, segmentation, alternate encoding and parser disagreements require consistent **reassembly and normalization**. An **insertion attack** adds data a sensor accepts but the endpoint rejects.
- **DNS, HTTP(S), ICMP or ACK tunneling** can conceal traffic inside allowed protocols. Apply application-aware egress controls, DNS/proxy telemetry and behavioral correlation. **Firewalking** uses TTL expiry to infer filtering; encrypted traffic still exposes useful flow metadata.
- **Low-interaction honeypots** emulate services; **high-interaction honeypots** provide richer real systems and higher containment risk. A **honeynet** groups decoys. Service inconsistencies or timing can suggest deception but do not prove it; isolate and instrument decoys carefully.

Web Applications & APIs

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

ENFORCE EACH TRUST BOUNDARY

Request Validate structure → **Identity** Authenticate caller → **Resource** Authorize access → **Execution** Separate data and code

05 Web servers, applications and APIs

Server exposure and hardening

- The **document root** holds published content; the **server root** holds server installation or configuration material; virtual mappings expose other paths. Virtual hosts share infrastructure through host-based routing. Keep secrets and backups outside served paths with restrictive permissions.
- **Headers, errors, default pages, favicon signatures and HTTP behavior** support fingerprinting. **robots.txt** gives crawler instructions, not authorization. **Website mirroring** copies accessible content for analysis; authenticated or dynamic paths may require user-directed crawling.
- Patch the OS, web server, modules and dependencies; remove unused accounts, services, extensions and sample content. Use a restricted service identity, protected configuration, TLS and centralized logs. Hiding a banner alone does not repair a vulnerability.
- **Unrestricted uploads** can introduce webshells or stored active content. Allow approved formats, verify file contents, bound size, rename files, scan them, and store outside executable web paths. An extension or client-provided MIME type is insufficient validation.
- **Webshells** expose server-side execution through HTTP; **defacement** changes visible content. Detect unexpected file changes, process launches and outbound connections. Separate database and administrative access from public web tiers and validate authorization on every route.

Access control, APIs and service architecture

- **Broken access control** lets users exceed their permissions. **IDOR/BOLA** omits authorization for a referenced object; changing an object ID may expose another tenant's data. Unpredictable IDs help reduce guessing but never replace ownership checks.
- **Function-level authorization** protects administrative operations; **property-level authorization** protects fields. **Mass assignment** binds unexpected client fields such as role or owner. Use explicit input/output models and enforce permissions server-side at each boundary.
- **REST** uses resources and HTTP methods; **SOAP** uses structured XML messages. **WSDL** describes service operations and types; examining it maps the interface. **WS-Security** supports message-level protection; signed elements must match the elements the application actually processes.
- **XML signature wrapping** exploits mismatches between signed content and consumed content. **WS-Addressing/SOAPAction** manipulation targets routing or action interpretation. Validate schemas, element identity, destinations and authorization instead of trusting an envelope's presence of a signature.

Web Applications & APIs

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

ENFORCE EACH TRUST BOUNDARY

Request Validate structure → **Identity** Authenticate caller → **Resource** Authorize access → **Execution** Separate data and code

- **Webhooks** push event callbacks; authenticate senders, verify signatures on the expected body, reject stale/replayed events and restrict destinations. A webhook is different from a webshell. Apply authentication, authorization, input validation, rate limits and quotas to APIs.
- **Business-logic flaws** misuse valid operations or skip workflow stages. Enforce state transitions, transaction rules, pricing and verification steps on the server. **CORS** controls browser reading across origins; allow trusted origins narrowly and never treat CORS as API authorization.

Browser injection and request forgery

- **Reflected XSS** returns attacker input in a response; **stored XSS** persists it for later users; **DOM XSS** uses unsafe client-side source-to-sink flows. Execution occurs in the affected origin and can alter transactions even when cookies are HttpOnly.
- Prevent XSS with **context-sensitive output encoding**, safe DOM APIs, reviewed HTML sanitization and a restrictive **CSP**. HTML, JavaScript, URL and CSS contexts require different handling. Client-side input checks alone are bypassable.
- **CSRF** causes a victim's browser to submit an unwanted authenticated request. Use server-validated anti-CSRF tokens, appropriate SameSite cookies, origin checks and reauthentication for high-impact actions. CSRF does not inherently disclose the victim's session token.
- **Clickjacking** overlays or frames a legitimate interface to redirect user clicks. Use CSP **frame-ancestors** and suitable **X-Frame-Options** protection. Clickjacking controls do not automatically prevent XSS or CSRF.
- **CRLF/header injection** can alter response headers or split responses. **Request smuggling** exploits front-end/back-end parsing disagreement; **cache poisoning** stores a harmful response for other users. Reject ambiguous messages and keep cache keys consistent with response-varying inputs.

Server-side injection and file handling

- **Path traversal** escapes an allowed directory; **LFI** includes local files in application processing; **RFI** includes remote resources where supported. Use fixed resource mappings, canonical path checks and least-privilege file access; exposure of a file does not always imply code execution.
- **SSRF** makes a server fetch attacker-selected destinations, potentially reaching internal services or cloud metadata. Allow approved destinations and schemes, validate resolved addresses and redirects, restrict egress and protect metadata endpoints.
- **XXE** abuses XML external entity resolution for file disclosure or outbound requests. Disable unnecessary DTDs and external entities and use secure parsers. **Insecure deserialization** reconstructs untrusted objects and can trigger dangerous behavior; prefer simple validated data formats.

Web Applications & APIs / Wireless Security

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

ENFORCE EACH TRUST BOUNDARY

Request Validate structure → **Identity** Authenticate caller → **Resource** Authorize access → **Execution** Separate data and code

- **Command injection** sends untrusted data to OS command execution; **SSTI** interprets input as template instructions; **SSI injection** inserts server-side include directives. Prefer safe APIs and fixed templates instead of building executable instructions from input.
- **LDAP injection** changes directory search logic; **XPath injection** changes XML queries. Use supported parameterization or context-correct escaping and narrow search permissions. Structured input remains untrusted when retrieved later from storage.
- **HTTP parameter pollution** supplies duplicate parameters with inconsistent interpretations; **parameter fragmentation** spreads components across fields. Canonicalize once and reject ambiguous inputs. A WAF can reduce exposure but cannot substitute for secure query construction and authorization.

SQL injection mechanisms and defenses

- **SQL injection** occurs when untrusted input becomes SQL structure. Data-entry paths include query strings, forms, headers, cookies and stored values. **Tautology** changes conditions; **UNION-based injection** combines compatible result sets; **stacked/piggybacked queries** add another statement where supported.
- **Error-based SQLi** exposes database details through errors. **Boolean-based blind SQLi** infers facts through content differences; **time-based blind SQLi** infers them through controlled delays. Heavy-query timing uses expensive work; ordinary latency is not sufficient confirmation.
- **Out-of-band SQLi** returns data through another channel such as DNS or HTTP, when database capabilities and egress allow it. **Second-order SQLi** stores data that later becomes unsafe query syntax; safe initial storage does not make later concatenation safe.
- Evasion may vary case, whitespace, comments, encoding, string construction or parameter handling. Database dialect and parser behavior matter; no payload or bypass works universally. Compare repeatable, non-destructive behavior in an authorized test.
- **Prepared statements with bound parameters** separate values from SQL syntax. Allowlist dynamic identifiers, minimize database privileges and suppress public diagnostic details. Stored procedures are safe only when their internals avoid unsafe dynamic SQL; ad hoc escaping is fragile.

06 Wireless network security

Wi-Fi standards and authentication

- **WEP** uses RC4 with a **24-bit IV** and has serious key-recovery and integrity weaknesses. Legacy **WPA/TKIP** improves on WEP but remains obsolete. **WPA2 with AES-CCMP** replaces TKIP/RC4 with stronger encryption and message integrity.
- **WPA2-Personal** derives keys from a shared passphrase and SSID; the four-way handshake confirms keys and establishes session material. Capturing a handshake permits offline guesses against weak passphrases; it does not reveal the passphrase directly.

Wireless Security

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

WIRELESS PROTECTION

- **Discover** — Passive or active
- **Authenticate** — Verify identities
- **Encrypt** — Protect exchanges
- **Monitor** — Find rogue devices

- **WPA2/WPA3-Enterprise** uses **802.1X/EAP**, commonly through **RADIUS**, for individual authentication and derived session keys. **EAP-TLS** uses certificates. Clients must validate the authentication server certificate and expected identity to resist rogue authentication servers.
- **WPA3-Personal SAE** replaces PSK-style authentication and resists passive offline dictionary attacks. **Protected Management Frames (PMF)** protect specified management traffic. WPA3-Enterprise includes a stronger optional 192-bit security mode; all endpoints must support the selected profile.
- **Transition mode** preserves WPA2 compatibility and can expose weaker paths. Disable unnecessary legacy modes and patch implementations against **Dragonblood**-related weaknesses. Hidden SSIDs and MAC allowlists provide limited protection because identifiers remain observable or spoofable.

Wireless attacks, tools and defenses

- A **rogue AP** is unauthorized infrastructure; an **evil twin** impersonates a trusted network. **Wardriving** discovers wireless networks while moving. Passive listening collects beacons and exchanges; active discovery sends probes and can be logged.
- **Deauthentication/disassociation attacks** forge management traffic to disconnect clients where protection is absent. **Jamming** disrupts radio availability through interference. These differ from capturing or decrypting data; neither guarantees password recovery.
- **KRACK** exploits key reinstallation and associated nonce/replay-state reuse during handshake processing; patch clients and access points. **WPS PIN attacks** target provisioning rather than the strength of AES; disable unnecessary WPS.
- **airmon-ng** configures monitor interfaces; **airodump-ng** captures wireless traffic; **aireplay-ng** injects supported frames; **aircrack-ng** tests key recovery. **Kismet** supports wireless discovery; **wash** identifies WPS exposure; **Reaver** targets WPS PIN weaknesses.
- Segment guest and IoT networks, use strong unique passphrases or managed enterprise identities, validate certificates, update firmware and monitor rogue devices with **WIDS/WIPS**. RF range management helps containment but does not replace authentication and encryption.
- **Bluetooth bluejacking** sends unsolicited messages; **bluesnarfing** steals information; **bluebugging** gains device control. BLE attack feasibility depends on pairing mode, firmware and captured material. Patch devices, disable unused radios and restrict pairing; non-discoverable mode alone is insufficient.

Mobile, IoT & OT

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CONNECTED-DEVICE TRUST

Inventory Devices and roles → **Protect** Code and credentials → **Segment** Restrict communication → **Recover** Restore safely

07 Mobile platforms, IoT and operational technology

Mobile attack surfaces and device controls

- An Android **APK** packages code and resources; **AndroidManifest.xml** declares components and permissions; **DEX** contains bytecode. Reverse engineering reveals behavior and embedded secrets. **ADB** is a powerful debugging interface; exposed network ADB, often TCP **5555**, needs strict control.
- **Rooting** and **jailbreaking** remove platform restrictions and weaken normal trust assumptions. A **tethered** jailbreak needs a computer to start the modified environment; **untethered** modifications survive normal reboots; **semi-untethered** devices reboot normally but need an on-device reactivation app.
- **Repackaged apps** preserve legitimate appearance while adding code; **Agent Smith** is associated with replacement of installed Android apps. **SIMjacker** abuses vulnerable SIM toolkit behavior; **Spearphone** infers loudspeaker audio through motion-sensor side channels. Capabilities depend on the affected platform.
- **MDM** manages devices; **MAM** controls enterprise apps and data. Under **BYOD**, separate work data, restrict risky apps, review permissions, enforce encryption and screen locks, and require app-level protection for sensitive workflows.
- Protect local secrets using platform keystores and appropriate hardware-backed storage. Use authenticated TLS and secure token lifecycle management; avoid plaintext credentials in files, logs or backups. Root detection and certificate pinning are supporting controls, not complete defenses.
- Register devices for **remote lock/locate/wipe**, revoke sessions and credentials after loss, and maintain an offboarding process. Remote wipe depends on device reachability; storage encryption and short-lived access reduce exposure before a wipe arrives.

IoT attack surfaces and firmware trust

- Map **device, firmware, network services, local interfaces, mobile app, cloud and management APIs**. Default credentials, insecure ecosystem interfaces, weak data protection, missing updates and poor lifecycle management can compromise the whole system.
- **Shodan/Censys** identify exposed services; manufacturer records and **FCC IDs** may identify device models and radios. Inventory firmware versions and protocols before testing. Internet visibility is an exposure signal, not proof of a particular vulnerability.
- **Firmware analysis** obtains an authorized image, identifies format and architecture, examines strings and embedded files, and may extract filesystems or emulate code. **file**, **strings** and **binwalk** serve different inspection steps; never assume an extracted secret is still valid.

Mobile, IoT & OT / Cloud Security

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CONNECTED-DEVICE TRUST

Inventory Devices and roles → **Protect** Code and credentials → **Segment** Restrict communication → **Recover** Restore safely

- **Secure boot** verifies trusted code before execution; **signed updates** verify update origin and integrity; **anti-rollback** prevents reintroducing vulnerable releases. Protect signing keys, require authenticated updates and maintain a supported recovery path.
- **Replay** reuses a captured command; counter it with authenticated freshness such as nonces or counters. **DNS rebinding** can use browser-origin behavior to reach local devices; validate host/origin expectations and require device authentication.
- **Side channels** infer secrets from timing, power, cache or electromagnetic effects; **fault injection** deliberately perturbs execution using voltage, clock or other physical disturbances. Segment devices, restrict debug interfaces and protect both hardware and protocol boundaries.

OT architecture and safe assessment

- **PLCs** execute control logic; **HMIs** expose operator interaction; **SCADA** supervises distributed processes; **DCS** coordinates plant control. **Closed-loop** control uses feedback; **open-loop** control does not adjust from output feedback.
- Traditional **Modbus/TCP 502** reads coils and registers without built-in authentication or encryption. **Siemens S7 102/TCP** and **EtherNet/IP 44818** are useful industrial service clues. Protocol identity and security depend on the actual device and deployment.
- **Safety and availability** constrain OT testing. Begin with inventory and passive observation; coordinate active probes, scans and changes with plant owners. A fragile controller may fail under ordinary IT scanning or authentication testing.
- Use **zones and conduits**, industrial DMZs, tightly controlled remote access and allowlisted communications to separate IT, IoT and control networks. Avoid direct public exposure and apply MFA at supported access gateways.
- Validate firmware and logic changes, maintain known-good configurations, monitor industrial command patterns and rehearse safe restoration. Isolate a compromised device through an operationally safe containment plan; abrupt shutdown can create physical hazards.

08 Cloud computing and container security

Cloud models, responsibility and data protection

- Cloud characteristics include **on-demand self-service**, **broad network access**, **resource pooling**, **rapid elasticity** and **measured service**. Elasticity adjusts capacity; measured service records consumption for management and billing.
- **IaaS** leaves customers responsible for guest OS and applications; **PaaS** manages more of the runtime; **SaaS** supplies the application. Customers still govern identities, permissions, data handling and supported configuration under the shared-responsibility model.

Cloud Security

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CLOUD SECURITY BOUNDARIES

- **Identity** — Effective permissions
- **Workload** — Runtime isolation
- **Data** — Access and key control
- **Telemetry** — Logs and response

- **Public**, **private**, **community** and **hybrid** describe deployment models. NIST roles include consumer, provider, broker, auditor and **carrier**, which supplies transport/connectivity. An HA cluster handles failures; load balancing distributes work; neither automatically secures workloads.
- Review storage **resource policies and IAM permissions**, including grants to all authenticated users, cross-account principals and public access. Private-looking URLs are not access controls. Encrypt transit and storage, restrict access and record data-plane activity where needed.
- **Client-side encryption with independently controlled keys** can keep provider-held ciphertext unreadable to the provider, with application functionality tradeoffs. Protect keys separately and use immutable or offline backups, separated administrator roles and tested restores.

Cloud identities and attack paths

- Enumerate **users, roles, attached policies, trust relationships and effective permissions**. Apply least privilege, MFA, short-lived credentials and timely deprovisioning. Rotate exposed secrets and investigate their use; simply deleting a public repository copy is insufficient.
- **Instance metadata services (IMDS)** can expose workload role credentials to authorized local processes. SSRF or local compromise may reach them. Use hardened metadata access, narrow instance roles, egress controls and workload isolation; managed identities can be abused when overprivileged.
- **Serverless functions** require per-function permissions, validated event inputs, secure dependencies and activity logs. The provider managing servers does not eliminate application injection, secrets exposure, resource exhaustion or inadequate monitoring.
- **Cloud hopper** describes compromise through a managed service provider into customers. **Man-in-the-cloud** abuses synchronization tokens; **Golden SAML** forges federation assertions using stolen signing material. Protect MSP trust, sync tokens and identity-provider keys as high-value assets.
- **Cryptojacking** steals compute for mining; VM **side channels** infer information through shared hardware; **container escape** crosses workload isolation; firmware implants may persist across reassignment. Tenant isolation and secure provisioning address risks beyond ordinary account passwords.
- **CASB** adds visibility and policy for cloud-service use; **CSPM** finds cloud misconfiguration; **CWPP** protects workloads. Centralize control-plane, identity, workload and data-access telemetry; alert on new keys, logging changes, unusual API calls and abnormal egress.

Cloud Security / Cryptography & Trust

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CRYPTOGRAPHIC PROPERTIES

- **Encryption** — Confidentiality
- **MAC** — Shared-secret integrity
- **Signature** — Public verification
- **PKI** — Identity and trust

Docker and Kubernetes boundaries

- **Docker client** sends requests; the **daemon** manages containers, images, volumes and networks; a **registry** stores images. Access to a powerful daemon socket can effectively confer host control. Containers normally share a host kernel rather than providing a full VM boundary.
- Build from trusted minimal images, scan dependencies, verify **image signatures** and pin approved artifacts. Protect build and registry credentials, remove embedded secrets and run with a non-root identity where possible.
- Kubernetes **API server** exposes the control interface; **etcd** stores cluster state; the **scheduler** selects nodes for pending pods; **controllers** reconcile desired state; **kubelet** manages workloads on each node. Self-healing restarts or replaces workloads, not compromised credentials.
- Apply **RBAC** and scoped service accounts; protect bearer tokens from logs, images and unnecessary mounts. Restrict API access, encrypt sensitive cluster data and keep audit trails. A leaked service token carries the permissions of its identity.
- Avoid privileged containers and unnecessary host mounts or capabilities. Use **seccomp**, supported OS confinement, resource limits and network policies. Separate sensitive tenants where stronger isolation is required; orchestration does not automatically enforce every security boundary.

09 Cryptography and trust

Encryption algorithms and key direction

- **Symmetric encryption** uses a shared secret and efficiently protects bulk data. **Asymmetric cryptography** uses a public/private pair for supported encryption, signatures or key establishment. **Hybrid encryption** protects data with a symmetric key and securely establishes or wraps that key.
- **AES** uses **128-bit blocks** and **128/192/256-bit keys**. **DES** uses 64-bit blocks and a 56-bit effective key; **3DES** repeats DES with multiple keys but remains legacy. A large key does not remove small-block or protocol-design weaknesses.
- **Blowfish** uses 64-bit blocks and variable keys up to 448 bits; **Twofish** uses 128-bit blocks and keys up to 256 bits; **Serpent** uses 128-bit blocks and 32 rounds. **CAST-128** is a 64-bit-block Feistel cipher. Algorithm recognition does not imply suitability for a new design.
- **RSA** supports appropriately padded encryption and signatures; **ECC** is a family used for mechanisms such as ECDH and ECDSA. **DSA** is a signature algorithm. **Diffie–Hellman** establishes a shared secret; authenticate the exchange to prevent MitM substitution.
- For confidential public-key messaging, protect the content key for the **recipient's public key**; the recipient uses their **private key**. The sender signs with the **sender's private key**; recipients verify with the **sender's public key**. Encryption and signing provide different properties.

Cryptography & Trust

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CRYPTOGRAPHIC PROPERTIES

- **Encryption** — Confidentiality
- **MAC** — Shared-secret integrity
- **Signature** — Public verification
- **PKI** — Identity and trust

Integrity, certificates and key management

- A **cryptographic hash** maps data to a fixed-length digest; it is not reversible encryption. A trusted digest detects changes, but an attacker able to replace both file and digest can bypass that check. **HMAC** adds shared-secret authentication; a **digital signature** enables public verification.
- A **collision** finds any two inputs with the same digest; a **preimage** finds input for a chosen digest; a **second preimage** matches a given input's digest. The **birthday effect** reduces generic collision work for an n-bit hash to roughly 2^(n/2).
- **Certificates** bind identities to public keys through an issuer's signature. Validate chain, identity, validity and applicable revocation status. Protect private keys and trust anchors; a valid certificate does not prove the application behind it is secure.
- **PKI** uses certificate authorities and structured trust chains; **OpenPGP's web of trust** uses endorsements among keys. **GPG** implements OpenPGP; **S/MIME** uses certificate-based email encryption and signatures. Signatures support integrity, origin assurance and nonrepudiation when identity and key custody are reliable.
- **TPMs** protect keys and can bind release to platform state; **BitLocker** integrates Windows volume protection and recovery management; **VeraCrypt** supports encrypted containers and volumes. Encryption at rest does not stop a compromised authorized process from reading an unlocked volume.
- **Key escrow/archival** enables controlled recovery but creates sensitive recovery material. Separate access, back up required recovery keys, rotate and revoke compromised keys, and define retention and destruction. Losing the only key can make encrypted data unrecoverable.

Cryptography & Trust

Certified Ethical Hacker

KNOWLEDGE CHEATSHEET

CRYPTOGRAPHIC PROPERTIES

- **Encryption** — Confidentiality
- **MAC** — Shared-secret integrity
- **Signature** — Public verification
- **PKI** — Identity and trust

Protocols, cryptanalysis and emerging risks

- **TLS** provides authenticated encrypted transport when configured and validated correctly. **IPsec** protects IP traffic; **SSH/SFTP** secures remote access and file transfer; **FTPS** adds TLS to FTP. TLS does not make an insecure endpoint or stolen bearer token trustworthy.
- **Ciphertext-only** analysis has encrypted data; **known-plaintext** analysis has matching original/encrypted examples; **chosen-plaintext** analysis selects inputs for encryption; **chosen-ciphertext** analysis obtains decryption-related feedback on selected ciphertexts.
- **Linear cryptanalysis** uses statistical approximations; **differential cryptanalysis** studies input/output differences; **meet-in-the-middle** matches intermediate results in layered encryption. It differs from a network man-in-the-middle. **Padding oracles** exploit distinguishable decryption feedback.
- **Side channels** expose timing, power or cache behavior rather than directly breaking the cipher. Use constant-time cryptographic implementations, authenticated encryption and uniform error handling. Nonces/IVs must follow the selected mode's rules; **ECB** reveals repeated-block patterns.
- **DROWN** abuses legacy SSLv2 exposure and key reuse; **CRIME** concerns TLS compression leakage; **BREACH** concerns HTTP response compression. Disable obsolete protocols and risky compression contexts, patch implementations and avoid custom cryptographic schemes.
- **Steganography** conceals the existence of data inside other content; encryption conceals its meaning. **Steganalysis** looks for suspicious structure or statistical artifacts. Traffic volume, destinations and file changes provide supporting evidence but do not alone prove hidden content.
- **Harvest now, decrypt later** threatens long-lived data protected by quantum-vulnerable public-key systems. Build cryptographic inventories and algorithm agility, then migrate to validated **post-quantum** mechanisms. Increasing AES key size alone does not repair a vulnerable RSA/ECC key-establishment path.
- In blockchain, **51% attacks** can manipulate consensus under the affected protocol's assumptions; **eclipse attacks** isolate a node's peers; **Finney attacks** involve pre-mined double-spend sequences; **sandwich attacks** manipulate transaction ordering. Consensus, network and application risks are distinct.