# Security Architecture

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**LAYERED SECURITY DESIGN**

- **Identity** — Verify access
- **Network** — Limit paths
- **Application** — Validate requests
- **Cloud** — Define responsibilities

## 01 Security Architecture 29%

### Trust boundaries and segmentation

- **Zero trust** evaluates each access request using identity, device posture, resource sensitivity and context. Apply least privilege, continuous reassessment and automated policy enforcement.
- **Segmentation** limits lateral movement. A VLAN separates broadcast domains; enforce intersegment access with firewalls or ACLs. **Microsegmentation** also controls east-west and same-host VM traffic.
- **Screened subnets** separate public services from internal systems. Restrict web-to-database flows to required destinations and ports; isolate management interfaces from user and public networks.
- **ACLs** match source, destination, protocol, port and direction. Check rule order, implicit denial, subnet membership and return traffic. Stateful controls track sessions; stateless rules need explicit return allowances.
- **SDN** separates control from forwarding and programs network policy. Protect controller APIs, administrative identities and configuration channels; central control concentrates both enforcement and compromise impact.

### Inspection, routing and remote access

- **NIDS** monitors and alerts out of band; **NIPS** can block inline. **HIDS/HIPS** observe or enforce on hosts. Select alert-only operation when false positives must not interrupt service.
- **NGFW** adds application-aware controls; **UTM** combines security functions. **WAF** filters web requests. A **forward proxy/SWG** governs client web access; a reverse proxy fronts servers.
- **VPN full tunneling** routes all client traffic through corporate inspection; split tunneling sends selected traffic directly. **ZTNA** grants scoped application access using identity and posture.
- **Jump hosts**, bastions and RDP gateways concentrate administrative entry. Require MFA, limited targets, session auditing and restricted management paths; separate vendor access from corporate user access.
- **BGP prefix filters** restrict accepted or advertised routes while preserving dynamic routing. **DNSSEC** authenticates DNS data; **TSIG** authenticates supported DNS exchanges such as zone transfers. DNS filtering blocks known risky destinations.

# Security Architecture

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**LAYERED SECURITY DESIGN**

- **Identity** — Verify access
- **Network** — Limit paths
- **Application** — Validate requests
- **Cloud** — Define responsibilities

## Cloud services and security boundaries

- **IaaS** leaves customers responsible for guest OSs, applications, identities and data. **PaaS** shifts OS/runtime operation to the provider. **SaaS** shifts application operation; customers still govern access, data and configuration.
- **FaaS/serverless** removes server administration from the customer but retains code, dependency, permission and data responsibilities. Managed container services vary in who patches nodes and runtimes.
- **Public**, private, community and hybrid clouds describe deployment boundaries. Multitenancy depends on isolation controls; single tenancy reduces shared-resource exposure but still needs patching, access controls and resilience.
- **CASB** discovers cloud use and applies SaaS access, DLP and OAuth-app policies. Proxy modes can enforce inline; API modes inspect supported service activity and stored data; log analysis discovers shadow IT.
- **CSPM** identifies cloud configuration and compliance gaps. **SASE** combines cloud-delivered network security with WAN connectivity. Protect management-plane credentials, storage permissions and API access across accounts.

## Availability and scalable services

- **Load balancing** distributes requests; horizontal scaling adds instances, vertical scaling enlarges one instance. Autoscaling needs useful metrics, limits and startup capacity. One backend remains a single point of failure.
- **CDNs** cache content near users, reduce origin load and can absorb attacks. Protect the origin against bypass; restrict caching of private content and configure authentication-aware cache rules.
- **Read replicas** offload reports from transactional writes. Account for replica lag and consistency. Local caches reduce bandwidth use but need expiration, invalidation and protection of sensitive entries.
- **Availability zones** isolate local failures; regions isolate wider outages. Provider and technology diversity can reduce correlated failures, with added cost, integration and operational complexity.
- **Self-healing** requires end-to-end health checks, spare capacity and verified restart success. Upstream **DDoS scrubbing** addresses link saturation; WAFs, bot controls and rate limits address application-layer abuse.

# Security Architecture

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**LAYERED SECURITY DESIGN**

- **Identity** — Verify access
- **Network** — Limit paths
- **Application** — Validate requests
- **Cloud** — Define responsibilities

## Identity, federation and authorization

- **Identity proofing** establishes who an applicant is; authentication verifies a claimed identity; authorization permits actions. SSO reduces repeat sign-ins; federation trusts identity assertions across organizational boundaries.
- **SAML** carries signed federation assertions. **OpenID Connect** adds authentication to OAuth 2.0. **OAuth** delegates API authorization; scopes and consent limits prevent excessive third-party application access.
- Validate token **signature, issuer, audience, expiry and intended use**. Use short-lived access tokens and protected, revocable refresh credentials; a signed JWT is not automatically encrypted or instantly revocable.
- **Kerberos** uses tickets and depends on trusted services and time synchronization. **RADIUS** supports network-access AAA; **TACACS+** separates AAA for device administration. **EAP-TLS** uses client and server certificates.
- **RBAC** maps roles to permissions; **ABAC** also evaluates attributes such as location, device and classification. **MAC** enforces centrally controlled labels and clearances; **DAC** lets owners delegate access.

## Web, API and session architecture

- **API gateways** centralize routing, token validation, allowed methods, quotas and monitoring. Enforce object-level authorization in the application for every requested record; possession of an identifier grants no access.
- **Rate limits**, quotas and bot detection constrain scraping and expensive unauthenticated requests. CAPTCHA adds friction but is bypassable; authentication alone does not prevent resource exhaustion.
- **TLS termination** makes plaintext visible at the proxy or WAF. Route all intended requests through inspection, protect the next hop and plan certificate trust; encrypted bypass paths create blind spots.
- **X-Forwarded-For** or standardized Forwarded headers preserve client addresses behind proxies. Trust headers only from approved proxies that overwrite untrusted client values; correlate with proxy and application logs.
- **Secure cookies** require HTTPS; **HttpOnly** restricts script access; **SameSite** helps limit cross-site requests. Rotate sessions after authentication, expire idle sessions and invalidate sessions after compromise.

# Security Architecture

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**LAYERED SECURITY DESIGN**

- **Identity** — Verify access
- **Network** — Limit paths
- **Application** — Validate requests
- **Cloud** — Define responsibilities

## Secure development and software supply chains

- Establish **security requirements** and threat models before coding. Trace data flows and trust boundaries; use secure frameworks and shared validation, encoding and filesystem libraries to prevent recurring flaws.
- **SAST** examines code without running it; IDE integration gives early feedback. **DAST** tests running interfaces; **IAST** instruments execution. **Fuzzing** supplies malformed inputs to expose crashes and edge cases.
- **SCA** inventories third-party packages, versions, licenses and known flaws. Maintain an **SBOM**, monitor vulnerability disclosures and retest upgrades, including transitive dependencies and device firmware components.
- **CI/CD gates** combine code review, SAST, dependency/image scanning and policy checks. Linters standardize code; version-controlled infrastructure definitions and drift detection keep deployed configuration aligned.
- **Secrets managers** provide controlled runtime access to credentials. Keep secrets out of source, images, Dockerfiles and logs; restrict pipeline identities, rotate exposed secrets and separate build approval from production deployment.
- **Code signing** establishes publisher origin and artifact integrity when trust is validated. It does not prove code is safe. Protect signing keys and publish trusted hashes separately from a potentially compromised download site.

## OT, IoT and machine learning

- **ICS/SCADA** prioritizes safety and reliable control. **PLCs** execute control logic, often ladder logic; **historians** collect process data. Isolate IT, supervisory control and field devices with tightly allowed conduits.
- **Modbus RTU/TCP** is a legacy industrial protocol; Modbus TCP commonly uses port **502**. Legacy Modbus and DNP3 deployments may lack authentication or encryption; assess supported security extensions and safe compensating controls.
- **Safety instrumented systems** independently enforce safe operating limits. Protect engineering stations, firmware and command integrity; maintain tested manual/offline control. Cyber compromise can cause physical harm despite perimeter firewalls.
- **Legacy and embedded systems** may lack patch support or resources for modern agents. Use approved hardening, allowlisted flows, isolated management and passive monitoring; validate active scans and updates with operations staff.
- **IoT** needs unique credentials, minimal services, secure updates and isolated segments with NDR. **Zigbee** supports low-power personal-area mesh networks; deterministic control timing must survive security changes.
- **Machine learning** risks include biased training data, poisoning, evasion and privacy leakage. Validate provenance and representative datasets, protect training pipelines and monitor drift and decision quality.

# Security Operations

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**EVIDENCE-LED RESPONSE**

**Detect** Correlate evidence → **Contain** Limit harm → **Recover** Restore safely → **Improve** Test corrections

## 02 Security Operations 30%

### Threat intelligence and adversary modeling

- **Threat hunting** proactively tests hypotheses against endpoint, identity and network evidence. An absence of alerts does not establish safety; select telemetry that can confirm or refute the hypothesis.
- **MITRE ATT&CK** maps observed tactics and techniques to detection and defensive gaps. Tactics describe objectives; techniques describe methods. Use the ICS knowledge base for industrial environments.
- **Cyber Kill Chain** models intrusion progression. The **Diamond Model** relates adversary, capability, infrastructure and victim. These models organize evidence without proving actor identity.
- **OSINT** uses public information; **ISACs/ISAOs** support trusted industry sharing. Evaluate intelligence for relevance, timeliness and confidence; use behavioral TTPs when file hashes change between victims.
- **Honeypots**, honeytokens and decoy documents reveal interaction with resources legitimate users rarely need. Isolate deception infrastructure and monitor it; avoid giving an attacker a useful pivot.

### Telemetry and detection quality

- **SIEM** centralizes and correlates logs. Synchronize clocks, retain original events and normalize host/user identifiers so proxy, identity, endpoint, application and cloud activity can be joined.
- **EDR** records process, file and network behavior and supports host isolation. **NDR** analyzes network behavior; **UEBA** finds deviations such as impossible travel and unusual privileged activity. Validate anomalies against context.
- **FIM** detects unauthorized file changes; **DAM** monitors database actions, including privileged queries. Baselines, hashes and audit trails distinguish content changes from ordinary access.
- **Flow logs** show connection metadata; packet captures show available payload details. HTTPS can hide content from network sensors; authorized TLS inspection or endpoint/application telemetry supplies visibility.
- **False positive** means benign activity alerted; **false negative** means malicious activity missed. True positives and true negatives are correct classifications. Tune narrow rules and test normal business workflows before blocking.

# Security Operations

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**EVIDENCE-LED RESPONSE**

**Detect** Correlate evidence → **Contain** Limit harm → **Recover** Restore safely → **Improve** Test corrections

## Vulnerability management and prioritization

- Begin with an accurate **asset inventory**, owners, exposure and criticality. Track findings through assignment, due dates, remediation and verification; **scan-patch-scan** confirms changes rather than assuming success.
- **Credentialed scans** inspect local configuration and installed fixes. Unauthenticated scans show external exposure. **Agents** cover off-network hosts and help deduplicate multihomed assets; passive discovery reduces OT disruption.
- **CVE** identifies a disclosed vulnerability; **CWE** classifies a weakness. **CVSS** scores severity. Prioritize using exploit activity, exposure, privileges, business impact and compensating controls alongside severity.
- **CVSS v3** groups metrics into base, temporal and environmental. Read **AV** attack vector, **AC** complexity, **PR** required privileges and **UI** interaction; environmental impact requirements reflect the actual asset.
- **SCAP** automates standardized assessment: **XCCDF** expresses checklists, **OVAL** describes checks, **CPE** identifies platforms and **ARF** packages results. Validate version-only findings against vendor backports; custom checks cover internally discovered flaws.

## Incident response and containment

- Follow **preparation; detection and analysis; containment, eradication and recovery; post-incident improvement**. Define reporting channels, authority, escalation and evidence handling before an event.
- **Containment** stops spread or loss: isolate affected hosts, restrict malicious egress or suspend compromised identities. Coordinate with safety and availability needs; preserve volatile evidence where feasible without allowing avoidable harm.
- **Eradication** removes persistence and root causes, patches exploited weaknesses and rotates exposed credentials. Recovery restores a known-good state, verifies data integrity and watches for recurrence.
- **Ransomware** response combines isolation, scope analysis and tested offline or immutable backups. Restoring data without removing the entry path and persistence invites reinfection.
- **SOAR** orchestrates tool actions through APIs. Playbooks define response decisions; runbooks detail repeatable steps. Constrain automation privileges and use approval gates for disruptive or uncertain actions.

# Security Operations

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**EVIDENCE-LED RESPONSE**

**Detect** Correlate evidence → **Contain** Limit harm → **Recover** Restore safely → **Improve** Test corrections

## Malware, persistence and exfiltration

- **Fileless malware** can execute through PowerShell or other trusted interpreters without a conventional disk payload. Correlate script logs, parent-child processes, memory artifacts and network connections.
- **Persistence** may use systemd units, scheduled tasks, startup entries, services or modified firmware. Ending one process does not remove its startup mechanism; patching an entry vulnerability does not remove existing persistence.
- **Reverse shells** connect outward to attacker infrastructure. A **pivot** uses a compromised system to reach other networks. Periodic beaconing or IRC-like commands can indicate C2 but need process and destination context.
- **Exfiltration** may use HTTPS, DNS, ICMP, tunnels, removable media or covert channels. Compare destinations, volume, timing and account behavior with baselines; protect IPv6 paths as well as IPv4.
- **Sandbox detonation** observes suspicious attachments or binaries in isolation and yields behavioral indicators. Malware may detect analysis environments; propagate validated detections to EDR and other controls rather than relying only on hashes.

## Forensic acquisition and evidence

- Follow **order of volatility**: capture relevant live memory, processes and network state before shutdown can destroy them, when safe and authorized. Document tool actions and unavoidable changes to the system.
- **Bit-level imaging** preserves allocated and unallocated storage. Use write protection where appropriate, retain originals securely and analyze verified working copies; snapshots complement a collection plan but may omit memory or external volumes.
- **Cryptographic hashes** validate evidence integrity before and after transfer. A hash does not identify the custodian, prove lawful acquisition or independently establish who created a file.
- **Chain of custody** records collection, identities, timestamps, transfers, storage and purpose. Gaps weaken evidentiary reliability; secure evidence access and maintain a complete audit trail.
- **Legal hold** suspends deletion of relevant records; **e-discovery** identifies, preserves, searches and produces responsive electronic information. Coordinate scope, authority and handling with counsel.

# Security Operations / Engineering & Cryptography

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**EVIDENCE-LED RESPONSE**

**Detect** Correlate evidence → **Contain** Limit harm → **Recover** Restore safely → **Improve** Test corrections

## Investigation tools and observable artifacts

- **ss/netstat** associate sockets with endpoints and connection states; **lsof** links files and sockets to processes. Correlate PIDs with **ps**, executable paths, service ownership and startup configuration.
- **systemctl** inspects, stops or disables systemd services: stopping affects current execution; disabling affects configured startup. **journalctl**, authentication logs and **dmesg** help trace service, login and device events.
- **tcpdump/Wireshark** capture and analyze traffic; **Nmap** discovers services. Common mappings include SSH/SFTP **22**, DNS **53**, HTTP **80**, HTTPS **443**, SMB **445**, IMAPS **993**, SQL Server **1433** and PostgreSQL **5432**.
- **dd/dcfldd** acquire storage images; **Foremost** carves files by signatures; **Volatility** analyzes memory. **ExifTool** extracts metadata, potentially including GPS data; disassemblers and debuggers support binary analysis.
- **ssdeep** compares fuzzy similarity, not forensic identity. **strings** exposes readable content; static inspection and isolated dynamic analysis serve different purposes. Avoid executing untrusted binaries on an analyst workstation.
- **Regular expressions** need the correct engine, escaped metacharacters and narrow context. `^\d{5}(-\d{4})?$` matches five digits with an optional four-digit suffix; test valid, invalid and near-match inputs before enforcement.

## 03 Security Engineering and Cryptography 26%

## Host hardening and privilege control

- **Least functionality** removes unnecessary services, listeners, software and interfaces. Bind internal services to intended interfaces, restrict administration and disable unneeded components on high-value servers.
- **PAM** limits privileged access through approval, short-lived elevation, credential vaulting and session records. Remove routine local administrator rights and separate administrative from daily-use identities.
- **Application allowlisting** uses deny-by-default execution policy. Control interpreters, scripts and child processes as well as binaries; assess business compatibility and monitor attempted bypasses.
- **SELinux** applies mandatory access control to labeled subjects and objects. **Enforcing** blocks policy violations; permissive logs without blocking. Review audit denials and labels before changing policy.
- **Immutable systems** are replaced from a trusted image rather than modified in place. Containers share the host kernel; **namespaces** isolate views and **cgroups** limit resources. Scan images, limit capabilities and keep persistent data outside disposable instances.

# Engineering & Cryptography

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**PROTECT THE FULL LIFECYCLE**

- **Platform** — Harden and attest
- **Code** — Prevent flaws
- **Keys** — Control use
- **Data** — Protect all states

## Hardware roots of trust and memory protection

- **Secure Boot** verifies authorized boot components. **Measured Boot** records measurements in TPM **PCRs**; attestation reports platform state. Measurements support detection and key-release policy rather than automatically blocking execution.
- **TPMs** seal keys to platform conditions; hardware changes can require recovery material. **Secure enclaves** isolate sensitive processing. **HSMs** protect cryptographic keys and perform controlled operations.
- **eFuses** can store irreversible device state or tamper indicators. Provision unique device keys with a renewal strategy; a compromised immutable key may require hardware replacement.
- **NX/XN/DEP** prevents execution from designated data pages. **ASLR** randomizes memory locations. Return-oriented programming reuses executable code; combine these defenses with control-flow protection and fixes for memory bugs.
- **Side channels** infer secrets from timing, power or other physical leakage; fault injection manipulates execution. Use constant-time implementations, tamper resistance and appropriate physical controls; memory encryption does not eliminate all side channels.

## Mobile devices and wireless access

- **BYOD** is personally owned; **COPE** is corporate owned with personal use; **CYOD** offers an approved device choice. Ownership policy determines monitoring, support, privacy and wipe authority.
- **MDM/UEM** enforces device configuration, encryption, updates and compliance. **MAM** protects corporate apps and data through app policies, managed sharing and selective wipe; separate personal and corporate storage.
- **Conditional access** combines identity and device signals before resource access. **NAC** checks posture and can quarantine noncompliant endpoints. Device certificates identify devices but do not alone prove patch or agent health.
- **Remote wipe** depends on device reachability and enforcement. Full-disk encryption protects powered-off lost devices; revoke sessions and credentials too. Restrict sideloading, unsafe auto-join and unauthorized removable media.
- **WPA3-Personal SAE** resists passive offline password guessing. Enterprise **802.1X/EAP-TLS** needs trusted server validation and client certificates. **WIDS** alerts on wireless anomalies; geofencing and MAC randomization serve location policy and tracking reduction.

# Engineering & Cryptography

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**PROTECT THE FULL LIFECYCLE**

- **Platform** — Harden and attest
- **Code** — Prevent flaws
- **Keys** — Control use
- **Data** — Protect all states

## Secure code and input handling

- **SQL injection** mixes untrusted input with query syntax. Use parameterized queries and limited database accounts. LDAP injection requires LDAP-safe parameter handling or escaping appropriate to the filter or distinguished-name context.
- **XSS** executes attacker-controlled content in a browser. Use context-aware output encoding, safe templating and HTML sanitization when markup is allowed; a CSP adds defense in depth.
- **CSRF** abuses a browser's authenticated session. Use anti-CSRF tokens, origin checks and suitable SameSite cookies. XSS and CSRF require different primary controls.
- **Directory traversal/file inclusion** abuses attacker-controlled paths. Canonicalize and confine paths to approved roots, allowlist resources and enforce authorization. Rejecting one `../` pattern is insufficient.
- **Command injection** passes input to a shell; prefer structured library APIs and validated arguments. **XXE** abuses XML external entities; disable unnecessary entity and external-resource resolution.
- **Buffer overflows** follow missing bounds checks, including unsafe string calls such as gets/strcpy. Use safe APIs and memory-safe designs. **Integer overflow** affects sizes/calculations; **race conditions** need synchronization and atomic operations.
- **Error handling** returns generic client messages while retaining protected diagnostic logs. Prevent stack traces, paths, account enumeration and secrets from leaking; validate authorization on the server, not only in browser code.

## Cryptographic purposes and algorithms

- **Encryption** provides confidentiality. **Hashing** detects changes against a trusted digest. **HMAC** authenticates messages with a shared secret but does not encrypt them or provide public-verifiable non-repudiation.
- **Digital signatures** use a private signing key and a public verification key for origin and integrity assurance. Non-repudiation also depends on key protection, identity validation and reliable records.
- **Symmetric encryption** efficiently protects bulk data. **Asymmetric cryptography** supports signatures and key establishment. **ECDH** establishes shared secrets; **ECDSA** signs; **AES** encrypts. These functions are not interchangeable.

# Engineering & Cryptography

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**PROTECT THE FULL LIFECYCLE**

- **Platform** — Harden and attest
- **Code** — Prevent flaws
- **Keys** — Control use
- **Data** — Protect all states

- **Password hashing** uses a unique salt and a deliberately expensive password KDF such as PBKDF2 or an appropriate memory-hard scheme. Salts defeat precomputed tables; they are not secrets or a substitute for work factors.
- **Homomorphic encryption** supports specified computation on ciphertext without revealing plaintext to the processor. Supported operations and overhead constrain its use; it is different from decrypting data inside a trusted execution environment.

## Cipher modes and authenticated encryption

- **ECB** exposes repeated plaintext patterns. **CBC** needs an unpredictable IV, correct padding and separate integrity protection. Error or timing differences can create padding-oracle vulnerabilities.
- **CFB** turns a block cipher into a stream-like mode. **CTR** generates a keystream from counters; nonce/counter reuse under one key is dangerous. Mode choice does not remove authentication requirements.
- **AEAD** combines confidentiality with integrity and authenticates additional unencrypted metadata. **AES-GCM** and **ChaCha20-Poly1305** are common examples; GCM requires nonce uniqueness for each key.
- **ChaCha20-Poly1305** performs well in software on platforms without AES acceleration. Disable obsolete **RC4**, weak/export suites and **NULL** encryption; do not select a cipher solely by its name or key length.

## Key ownership and lifecycle

- **Key management** covers secure generation, distribution, use, rotation, backup, recovery, revocation and destruction. Separate keys from encrypted data and restrict decryption permissions and audit access.
- **Key escrow** entrusts recovery material to an authorized custodian. **Distribution** delivers keys to intended users; rotation replaces keys. **Revocation** rejects trust in compromised credentials, while zeroization removes key material.
- **Envelope encryption** encrypts data with a data key and wraps that key with a key-encryption key. HSMs and managed vaults protect key operations; protect backup keys and MFA seed recovery with equivalent care.
- **Customer-managed keys** improve control but may still be usable by an authorized cloud service. Keeping plaintext from the provider requires an encryption and processing design in which the provider cannot obtain decryption capability.

## PKI, certificate identity and trust

- **CAs** issue signed certificates; **RAs** validate enrollment identities. A **CSR** contains a public key and requested attributes, not a private key. Validate the leaf-to-intermediate-to-trusted-root chain.
- **SAN** identifies supported DNS names, IP addresses or email identities. A wildcard usually covers one subdomain level, not the apex or unrelated domains. Multiple SAN entries can cover distinct authorized names.

# Engineering & Cryptography

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**PROTECT THE FULL LIFECYCLE**

- **Platform** — Harden and attest
- **Code** — Prevent flaws
- **Keys** — Control use
- **Data** — Protect all states

- **Key Usage/EKU** constrains certificate purposes, such as server authentication, client authentication, code signing or email. A valid web certificate is not automatically a valid code-signing certificate.
- **DV**, OV and EV describe different identity-validation levels, not stronger encryption. Public clients need a trusted issuer or deliberate trust enrollment; self-signing alone does not establish a trusted identity.
- **PKCS#10** represents a CSR; **PKCS#12/PFX** can package certificates and private keys; **DER/PEM** are encoding representations. Protect private-key exports and include the needed intermediate chain without distributing CA private keys.

## Revocation, TLS and secure messaging

- **CRLs** list revoked certificates; delta CRLs publish changes to a base list. **OCSP** queries status; **OCSP stapling** lets the server supply a signed status response, reducing client lookups.
- Suspected **private-key exposure** requires replacement key material and revocation of affected certificates. CRL/OCSP status does not prove a key was never stolen. Re-encrypt or rotate dependent secrets when necessary.
- **TLS 1.2/1.3** needs compatible secure algorithms and correct certificate validation. **Ephemeral DHE/ECDHE** provides forward secrecy, limiting exposure of earlier sessions if a long-term authentication key is later stolen.
- **HSTS** directs browsers to use HTTPS; preloading can protect the first visit. **Certificate pinning** constrains expected identities in managed clients but needs trusted initial provisioning and a safe rotation/recovery plan.
- **mTLS** authenticates both endpoints with certificates. **S/MIME** signs mail with the sender's key and encrypts it for the recipient's public key; confirm address bindings, certificate purpose, chain and validity.

## TLS and VPN troubleshooting

- For certificate errors, check **hostname/SAN**, validity, local time, trust anchors, intermediate chain, revocation status and EKU. Automate certificate inventory, renewal, deployment verification and expiry alerts.
- A **version/cipher mismatch** means the peers cannot agree on supported settings. Inspect both sides; remove insecure fallback rather than enabling obsolete algorithms to silence the error.
- **TLS inspection** re-signs traffic with an enterprise CA. Managed clients must trust the inspection chain; pinning or mutual authentication can prevent interception. Investigate failures before adding broad bypasses.
- **IKE/IPsec** peers must agree on authentication, encryption, integrity, DH group and traffic selectors. Use modern authenticated encryption, correct identities and compatible rekey settings; TLS and IPsec negotiation are distinct.
- After an **OpenSSL memory-disclosure flaw**, patch affected software and assess exposed keys, credentials and sessions. A patch closes the weakness but cannot recover already disclosed secrets.

# Engineering & Cryptography / Governance, Risk & Compliance

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**PROTECT THE FULL LIFECYCLE**

- **Platform** — Harden and attest
- **Code** — Prevent flaws
- **Keys** — Control use
- **Data** — Protect all states

## Data confidentiality and usage controls

- **Full-disk encryption** protects stored media, especially lost devices. File, field or record encryption narrows protection boundaries; encrypt container persistent volumes and backups as well as disposable compute.
- **Masking** hides values in a view. **Tokenization** substitutes a surrogate and protects the mapping; recurring payments can use reusable authorized tokens without exposing PANs to every application. Neither is simply encoding.
- **DLP** discovers classified data at rest, in motion and at endpoints. Combine labels, patterns, context and document fingerprints; use staged alerting/quarantine and tune false positives to preserve legitimate work.
- **DRM/IRM** constrains authorized document use such as printing or forwarding. Watermarks support attribution; **steganography** hides data within another medium, while steganalysis seeks evidence of concealment.
- **SPF** authorizes sending infrastructure for an envelope domain; **DKIM** signs selected mail content; **DMARC** evaluates alignment with the visible From domain and supplies policy/reporting. These controls do not guarantee a message is harmless.
- **Sanitization** must match the medium: clear, purge or destroy as appropriate. Degaussing targets magnetic media, not SSDs. Verified cryptographic erase depends on prior encryption and complete key destruction; formatting alone leaves recoverable data.

## 04 Governance, Risk, and Compliance 15%

## Risk assessment and treatment

- Define the **assessment boundary**, assets, data flows and owners before evaluating threats. A vulnerability is a weakness; a threat can exploit it; risk combines likelihood and adverse impact in business context.
- **Avoid** stops the risky activity; **mitigate** reduces likelihood or impact; **transfer** allocates financial consequences, for example through insurance; **accept** records an authorized decision to retain risk.
- **Inherent risk** precedes controls; **residual risk** remains afterward. Reassess mitigation effectiveness and compare residual risk with approved appetite and tolerance. Outsourcing does not remove organizational accountability.
- A **risk register** tracks scenarios, owners, likelihood, impact, controls, response plans and review dates. Give exceptions an approver, rationale, compensating controls and expiry or remediation target.
- **Qualitative analysis** ranks relative risk; **quantitative analysis** estimates monetary loss. Use scenarios tied to business objectives and explain uncertainty rather than presenting estimates as guarantees.

# Governance, Risk & Compliance

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**BUSINESS-DRIVEN RISK DECISIONS**

**Scope** — Assets and owners → **Assess** — Likelihood and impact → **Treat** — Controls and acceptance → **Review** — Residual risk

## Quantitative risk and measurable objectives

- **SLE = asset value × exposure factor**. Exposure factor is the estimated fraction lost in one event. **ALE = SLE × ARO**, where ARO is expected events per year.
- Estimate **ARO** from the relevant event: two successful incidents in four years implies 0.5 per year, regardless of how many unsuccessful attempts occurred. Match units and observation periods.
- **Control cost-benefit** compares total control cost with expected loss reduction and mandatory requirements. A financially acceptable risk may still violate a binding obligation; record the decision and its assumptions.
- **KPIs** measure performance; **KRIs** signal risk exposure. Track patch compliance, remediation time, logging coverage and phishing outcomes. Maturity models assess process capability, not proof that no breach can occur.

## Business impact and recovery design

- **BIA** prioritizes mission-essential functions, dependencies, revenue, safety and outage impacts. Use it to set recovery priorities, minimum acceptable service levels and required recovery capacity.
- **RTO** is the target time to restore service; **RPO** is the tolerable data-loss window. **MTD** is the maximum tolerable disruption. Plan enough time within MTD for recovery and business resumption.
- **MTBF** describes average time between repairable failures; **MTTR** describes average restoration/repair duration. Higher availability needs redundancy, maintainability and tested recovery, not just favorable hardware statistics.
- **BCP** sustains business operations; **DRP** restores technology. Document people, communications, alternate sites, dependencies, recovery sequence, access and validation. Cold sites need more setup; hot sites provide more readiness at higher cost.
- **Backups** provide recoverable history; replication reduces data loss but can copy corruption or ransomware. Verify replication lag, restore integrity and protected/offline copies; soft delete and immutability address different deletion risks.

## Exercises and operational assurance

- **Tabletop exercises** discuss scenarios without disrupting production. Walk-throughs review procedures; **parallel tests** exercise recovery systems alongside production; full-interruption tests deliberately transfer operations and carry greater risk.
- Test recovery using **current data**, realistic capacity, application dependencies and business transactions. Measure achieved RTO/RPO, verify restoration and failback, and investigate silent replication or health-check failures.
- **Penetration tests** validate exploitable paths; red teams emulate adversaries, while purple teams collaborate with defenders. Map emulation to ATT&CK techniques to test specific detections and responses.

# Governance, Risk & Compliance

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**BUSINESS-DRIVEN RISK DECISIONS**

**Scope** Assets and owners → **Assess** Likelihood and impact → **Treat** Controls and acceptance → **Review** Residual risk

- **Rules of engagement** define authorized targets, timing, methods, stop conditions, contacts and data handling. Confirm cloud-provider testing constraints and include system owners before any disruptive work.
- **After-action reviews** identify root causes and improvements. Assign owners and deadlines to corrective actions; measure training outcomes with sanctioned phishing simulations and strengthen technical controls when training fails.

## Governance, change and account lifecycle

- **Policies** state requirements; standards define mandatory specifics; procedures give steps; guidelines advise. Control types include administrative, technical and physical, serving preventive, detective, corrective or compensating functions.
- **Change management** records scope, risk, testing, approvals, implementation and rollback. Submit added project scope through change control; preserve emergency-change evidence and review afterward.
- **Separation of duties** prevents one person from controlling an entire sensitive workflow. Dual control requires two participants; job rotation and mandatory leave can expose concealed misuse.
- **IAM lifecycle** integrates authoritative HR events with provisioning and termination. Review stale accounts, excessive privileges, service identities and third-party grants; make access removal prompt and auditable.
- **Physical controls** combine badges, visitor logs, escorts, cameras, locks and access-control vestibules. Correlate entry records and video to investigate tailgating; an air gap does not stop insiders or removable media.

## Privacy, data governance and compliance

- **Data owners** decide classification and access; custodians operate protection controls; stewards maintain quality and governance. Labels and tags connect sensitivity to retention, DLP and attribute-based access rules.
- **Privacy impact assessments** document personal-data use, flows, necessity and risks. Minimize collection and retention; determine lawful purposes, transparency and applicable individual rights rather than assuming encryption alone establishes compliance.
- **GDPR** distinguishes a controller that determines purposes and means from a processor acting on its behalf. Data location, international transfers, deletion requests and retention obligations need explicit governance.

# Governance, Risk & Compliance

CompTIA Advanced Security Practitioner (CASP+)

KNOWLEDGE CHEATSHEET

**BUSINESS-DRIVEN RISK DECISIONS**

**Scope** — Assets and owners → **Assess** — Likelihood and impact → **Treat** — Controls and acceptance → **Review** — Residual risk

- **PCI DSS** addresses payment account data and the cardholder-data environment. Provider certification does not make every customer integration compliant; define scope and shared responsibilities and minimize stored account data.
- **ISO 27001** addresses an information security management system; **ISO 27018** addresses PII protection in public-cloud processing. NIST frameworks and control catalogs, OWASP guidance and audit reports serve different governance and assurance purposes.

## Vendor risk, contracts and accountability

- **Due diligence** investigates vendor capability and risk before decisions; due care maintains reasonable safeguards. Review financial viability, ethics, support hours, supply chains, geographic resilience and evidence of control operation.
- **SLAs** specify measurable service commitments and remedies; **MSAs** set broader terms; **SOWs** define work and deliverables. **NDAs** protect confidentiality; **ISAs** define interconnection security; **BAAs** address applicable health-data relationships.
- **MOUs** record shared intentions and responsibilities; enforceability depends on wording and context. Establish breach reporting, audit access, data ownership, deletion and subcontractor responsibilities in appropriate binding agreements.
- **Vendor lock-in** makes migration difficult through proprietary interfaces, formats or contract terms. Test exports, portability and exit procedures; consider termination costs and support loss. Assess each critical product, not only the vendor brand.
- **Source code escrow** releases maintained code under defined conditions such as supplier failure. Confirm build dependencies, documentation, licensing and release triggers; escrow is useful only if the organization can use the material.
- **Data residency** specifies storage location; sovereignty concerns applicable legal authority. Review backups, replicas, support access and subprocessors as well as the primary region; an architectural label alone does not guarantee locality.