**CySA+ CS0-002 Exam Topics Notes**

**1.0 Threat and Vulnerability Management**

**1.1 Explain the importance of threat data and intelligence**

- Intelligence sources:

- **Open-source intelligence**: publicly available information

- **Proprietary/closed-source intelligence**: info with restricted access (e.g. police record)

- **Timeliness**: timely receipt/operationalisation (impact > intelligence cost)

- **Relevancy**: must address a threat and allow for effective action; usable delivery format

- **Accuracy**: must save organisations more in success than errors/mistakes

- **Confidence levels**

- Indicator management:

- **STIX**: describes cyber threat information (motivation, abilities, capabilities, response)

- **TAXII**: describes how threat info (STIX) can be shared (hub-and-spoke;   
 source/subscriber; peer-to-peer); discovery, collection management, inbox, poll

- **OpenIOC**: standard format for defining/recording/sharing artifacts

- Threat classification:

- **Known threat vs unknown threat**: external/removable media, attrition, web, email,   
 impersonation, improper usage, equipment loss/theft etc.

- **Zero-day**: unknown vulnerabilities that have no patches

- **APT**: skilled attackers supported by extremely large resources

- Threat actors:

- **Nation-state**: geopolitically motivated groups with dedicated resources/personnel,   
 extensive planning & coordination

- **Hacktivist**: ideologically motivated groups that rely on widely available tools

- **Organised crime**: profit-driven groups that target PII, credit cards etc.

- Insider threat:

- **Intentional**: disgruntled or profit-driven employee stealing/damaging/exposing systems

- **Unintentional**: personal negligence/poor security practices

- Intelligence cycle:

- **Requirements**: determine exact customer requirements (IRs), how it should be collected

- **Collection**: gather data from wide array of desired/reliable/timely sources

- **Analysis**: raw info + other sources => intelligence; assess importance/accuracy/reliability

- **Dissemination**: timely conveyance of intelligence in appropriate format to customers

- **Feedback**: solicit feedback from customer, refine existing IRs

- **Commodity malware**: widely available paid/free malware used by many threat actors

- Information sharing and analysis communities:

- **Healthcare**: H-ISAC, Healthcare Ready

- **Financial**: FS-ISAC

- **Aviation**: A-ISAC

- **Government**: EI-ISAC (elections), DIB-ISAC (defense), NEI (nuclear)

- **Critical infrastructure**: E-ISAC (electricity), ONG-ISAC (oil & gas), PT-ISAC (public transit)

**1.2 Given a scenario, utilise threat intelligence to support organisational security**

- Attack frameworks:

- **MITRE ATT&CK**: tactics & techniques in developing threat models and methodologies

- **The Diamond Model of Intrusion Analysis**: intelligence on network intrusion

events using 4 elements (adversary, capability, infrastructure, victim)

- **Kill chain**: visibility into attack; reconnaissance -> weaponisation -> delivery ->   
 exploitation -> installation -> C2 -> actions on objectives

- Threat research:

- **Reputational**: detects threats with IP/domain/file reputations

- **Behavioural**: detects unknown threats based on their behaviour

- **IOC**: forensic data that identify potentially malicious activity on systems/networks

- **CVSS**: measure severity of security flaws (AV, AC, Au, C, I, A)

- Threat modelling methodologies:

- **Adversary capability**: adversarial toolsets/skillsets/evasion techniques

- **Total attack surface**: total of all different attack vectors an attacker can exploit

- **Attack vector**: describes how an attack can exploit the vulnerability

- **Impact**: magnitude of adverse impact on organisation

- **Likelihood**: likelihood that threat source will initiate risk & likelihood that the risk has   
 adverse effects on the organisation

- Threat intelligence sharing with supported functions

- **Incident response**: detect threats quicker, less disruptively prevent attacks, respond   
 quicker to adversaries

- **Vulnerability management**: provides context by identifying exploits and adding to   
 vulnerabilities list

- **Risk management**: rapidly receive and use actionable data about latest threats

- **Security engineering**: adapt to emerging threats

- **Detection and monitoring**: update signature database, monitor/detect new threats

**1.3 Given a scenario, perform vulnerability management activities**

- Vulnerability identification:

- **Asset criticality**: impact if CIA was breached; sensitivity of data & business criticality

- **Active vs passive scanning**: interact with targets VS use stored data to find info/identify   
 targets

- **Mapping/enumeration**: host/asset/network/infrastructure/systems discovery/mapping

- Validation:

- **True positive**: scanner correctly identifies existing vulnerability

- **False positive**: reported vulnerability that doesn't exist (verify patch/versions, or   
 attempt actual attack)

- **True negative**: scanner correctly doesn't alert on non-existent vulnerability

- **False negative**: scanner alerts on non-existent vulnerability

- Remediation/mitigation:

- **Configuration baseline**: perform anomaly analysis; provides info on OS/apps

- **Patching**: maintain current security patch levels on OS/apps (with e.g. SCCM)

- **Hardening**: disable unnecessary ports/services, centralised control, secure config etc.

- **Compensating controls**: when system can't be upgraded/patched; isolate and place   
 compensating controls in front

- **Risk acceptance**: don't take any action against risk (low risk; ALE < mitigation cost)

- **Verification of mitigation**: audits (formal), assessments (informal), patch levels,   
 repeated vulnerability scanning

- Scanning parameters and criteria:

- **Risks associated with scanning activities**: scans consume bandwidth and resources, and   
 risk business process interruptions (tune intensity & scan times)

- **Vulnerability feed**: SCAP (e.g. CCE [config], CPE [product names], CVE [vulnerabilities],   
 CVSS [severity], XCCDF [checklist results], OVAL [testing procedures used by checklists])

- **Scope**: extent of scan (included systems/networks; host discovery methods; what tests   
 will be conducted against active hosts)

- **Credentialed vs non-credentialed**: can confirm an issue by accessing OS/database/app   
 info VS chance of false positives/negatives

- **Server-based vs agent-based**: central server remotely scans hosts VS agent installed on   
 targets perform internal scans and report back to the server

- **Internal vs external**: gives different perspectives; insider threat vs external attacker

- Special considerations:

- **Types of data**: health, financial, PII etc.; data classification

- **Technical constraints**: capabilities of the scanning system => frequency limitations

- **Workflow**: remediation workflow (detection -> remediation -> testing);

- **Sensitivity levels**: minimum severity rating (low, medium, high, critical)

- **Regulatory requirements**: PCI DSS (internal & external; at least quarterly by qualified   
 professional or ASV); FISMA (updated scanning tools, update vulnerability list before

/after scan, some authenticated scans, determine discoverable info and correct them)

- **Segmentation**: compliance networks can be segmented to reduce scan scope

- **IPS, IDS, and firewall settings**: internal = insider threat; external = external attack

- Inhibitors to remediation:

- **MOU**: non-legally binding; customer must participate in including scanning in MOU

- **SLA**: customer expectations of security, performance & uptime

- **Organisational governance**: may block config changes needed for scanning; limited   
 resources and support

- **Business process interruption**: taking down systems can cause significant interruption

- **Degrading functionality**: service degradation can lead to business process interruption

- **Legacy systems**: EoL unsupported systems don't get security updates

- **Proprietary systems**: different vendors; some vendors will not have patches/updates

**1.4 Given a scenario, analyse the output from common vulnerability assessment tools**

- Web application scanner:

- **OWASP ZAP**

- **Burp suite**

- **Nikto**

- **Arachni**: evaluate web application security; scanning, scripted audits, vulnerability scans

- Infrastructure vulnerability scanner:

- **Nessus**

- **OpenVAS**

- **Qualys**

- Software assessment tools and techniques:

- **Static analysis**

- **Dynamic analysis**

- **Reverse engineering**

- **Fuzzing**

- Enumeration:

- **Nmap**: returns port listing, MAC address, OS/kernel version, network distance, runtime

- **hping**: sends TCP/UDP/ICMP/RAW-IP; firewall testing, TCP/IP auditing, network testing

- **Active vs passive**

- **Responder**: LLMNR/NBT-NS poisoner/rogue authentication server => steal NTLM hashes

- Wireless assessment tools:

- **Aircrack-ng**: suite of WiFi monitoring, attacking, testing & cracking (WEP/WPA) tools

- **Reaver**: brute force against WPS PINs to recover WPA/WPA2 passphrases

- **oclHashcat**: GPU-based hash cracker with dictionaries, masks, rules etc.

- Cloud infrastructure assessment tools:

- **ScoutSuite**: security posture assessment of cloud environments, highlights risks

- **Prowler**: AWS security best practices assessment, auditing, hardening, forensics

- **Pacu**: AWS exploitation framework; modules to exploit AWS configuration flaws

**1.5 Explain the threats and vulnerabilities associated with specialised technology**

- **Mobile**: malware; unpatched devices; jailbreaking; data leaks; OS vulnerabilities

- **IoT**: weak passwords; insecure services; lack of security update; outdated component use;   
 insecure data transfer/storage; lack of secure/physical device management

- **Embedded**: programming errors; web vulnerability; weak access control/authentication

- **RTOS**: RCE; DoS; information leak; improper access control

- **SoC**: low-level hardware bugs (boot header modification; partition header table parsing)

- **FPGA**: fault injection; hardware trojans; design leaks; foundry fabrication

- **Physical access control**: insufficient access control; lack of training; unattended assets

- **Building automation systems**: hardcoded secret; BOF; XSS; path traversal; auth bypass

- Vehicles and drones:

- **CAN bus**: DoS; unauthorised remote access

- **Workflow and process automation systems**: 3rd party platform vulnerabilities; IAM issue

- **ICS**: improper credentials management; weak firewall rules; network design weaknesses

- SCADA:

- **Modbus**: plaintext transmission; no authentication; command injection; weak sessions

**1.6 Explain the threats and vulnerabilities associated with operating in the cloud**

- Cloud service models:

- **SaaS**: customer only chooses application; hardware managed by provider; access control

- **PaaS**: configurable hardware + software/development tools; data protection

- **IaaS**: configurable hardware; VM management (VM escape; virtual host patching; virtual   
 guest issues [patching]; virtual network issues)

- Cloud deployment models:

- **Public**: public cloud provider sells services to consumers

- **Private**: internal enterprise service to internal customers

- **Community**: several companies work on same platform

- **Hybrid**: mix of on-premises, private cloud & public cloud

- **FaaS/serverless architecture**: apps are hosted by 3rd party; all server software/hardware   
 management is done by the provider

- **IaC**: managing/provisioning DCs using machine-readable files

- **Insecure API**: Internet-exposed management APIs can have software vulnerabilities (e.g.   
 anonymous access; plaintext authentication; improper authorisations)

- **Improper key management**: unencrypted; Internet-exposed key server; weak/reused key

- **Unprotected storage**: insider threats; malicious file entry; impersonation; worm that is   
 auto-synced to the cloud, and spread from the cloud to other users

- Logging and monitoring:

- **Insufficient logging and monitoring**: late detection; undetected password spraying;   
 ignored alerts; unidentified suspicious activity

- **Inability to access**: access logs provide info about failed requests made to cloud

**1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities**

- Attack types:

- **XML attack**: WAF; disable external entities; input validation; sensitive data not serialised

- **SQL injection**: WAF; input sanitisation; least privilege restrictions for databases

- Overflow attack:

- **Buffer**: ASLR/DEP; NX bit; use secure functions; higher-level languages; input validation

- **Integer**: range checking; prefer unsigned integers; use safer code implementations

- **Heap**: higher-level languages; input validation; safe compilers; patching

- **Remote code execution**: avoid using user input inside evaluated code; strict file upload   
 extensions etc.

- **Directory traversal**: ensure user cannot supply entire file path; accept known-good input

- **Privilege escalation**: avoid using administrative privileges; separate privilege areas

- **Password spraying**: MFA; strong passwords; user training; logging/monitoring

- **Credential stuffing**: MFA; CAPTCHA; unpredictable usernames; check against leaks

- **Impersonation**: use of session identifiers; packet filtering; DAI; encrypted protocols

- **Man-in-the-middle attack**: session encryption; ensure only valid certificates are used

- **Session hijacking**: key/cookie/link encryption; Secure & HttpOnly flags for cookies

- **Rootkit**: patching; layered security; heuristic analysis; antivirus

- Cross-site scripting:

- **Reflected**: WAF; use appropriate response headers; avoid suspicious links

- **Persistent**: WAF; filter input & encode data on output; escape HTML data on arrival

- **DOM**: don't treat untrusted data as code; delimit untrusted data as quoted strings

- Vulnerabilities:

- **Improper error handling**: info leak through over-detailed error messages

=> error handling policy; error logging; graceful handling of all possible errors

- **Dereferencing**: get value (NULL) in memory pointed by pointer; process failure

=> higher-level programming languages; sanity-check pointers prior to use

- **Insecure object reference**: (IDOR) exposure of reference to internal object

=> user authorisation; make objects harder to enumerate (e.g. random over increments)

- **Race condition**: produces unexpected result when timing of actions impact other actions

=> careful programming; locking (at most one thread can modify database)

- **Broken authentication**: brute-forcing credentials; unexpired session tokens

=> MFA; no default creds; password policy; delay failed attempts; session management

- **Sensitive data exposure**: steal keys; MITM; steal plaintext data (server/transit/client)

=> data classification; secure encryption; key management; salted hashes

- **Insecure components**: public exploits for known vulnerabilities

=> check product versions; monitor for unmaintained products (virtual patch/WAF)

- **Insufficient logging and monitoring**: lack of timely response; late detection/monitoring

=> failure logging; centralised logs; tamper prevention; timely incident response

- **Weak or default configurations**: unpatched flaws; default accounts; unprotected files

=> hardening; minimalistic platforms; segmentation; review & update configurations

- Use of insecure functions:

- **strcpy**: allows BOF => input validation; use secure functions

**2.0 Software and Systems Security**

**2.1 Given a scenario, apply security solutions for infrastructure management**

- **Cloud vs on-premises**: all managed by SP vs local physical/logical management

- Asset management:

- **Asset tagging**: assign labels including classification; unique ID; asset tracking system

- Segmentation:

- **Physical**: placing network devices to control access => new hardware + additional costs

- **Virtual**: VLANs/subnets on top of existing infrastructure => no new hardware/costs

- **Jumpbox**: intermediary connection point from untrusted to trusted network

- System isolation:

- **Air gap**: isolate system other networks/Internet; physical isolation (transfer with USBs)

- Network architecture:

- **Physical**: defense-in-depth security appliance; segmentation; physical security

- **Software-defined**: TLS; secure tunnelling; SDN controller hardening; access control

- **VPC**: traffic/anomaly monitoring; ingress/egress traffic control; secure VPC connections

- **VPN**: strong authentication; avoid DNS leaks; use a kill switch (drop Internet if VPN fails)

- **Serverless**: log monitoring; IAM; secured secrets; input validation; secure libraries

- **Change management**: change identification -> request -> request review -> prioritisation   
 -> evaluation/impact analysis -> approval/rejection -> testing -> implementation -> review

- Virtualisation:

- **VDI**: desktop OS on central server; centralised management, easy patching, antivirus

- **Containerisation**: isolate from host OS; monitoring; VA process; patch base & app image

- Identity and access management:

- **Privilege management**: least privilege; privileged account usage monitoring; prevent   
 privilege creep; role-based authorisation

- **MFA**: multiple authentication methods (knowledge; possession; biometric; location)

- **SSO**: authenticate once to use multiple systems; reduces password reuse/resets/support

- **Federation**: sharing of customer info to SPs; trust relationship between IdP, SP and user

- **Role-based**: access decision is based on roles; permissions assigned to roles not users

- **Attribute-based**: based on context (e.g. time, location, access frequency, behaviour)

- **Mandatory**: end users cannot modify security permissions set by administrators

- **Manual review**: review of access change logs, alerts, employee profiles, procedures

- **CASB**: policy enforcement/data protection point between consumers and SP (place   
 organisational policies on users accessing 3rd party, uncontrolled cloud services)

- **Honeypot**: intentionally vulnerable system that monitors attackers for intentions &   
 blacklists the IP address

- **Monitoring and logging**: SIEM; privileged use/change/grant, account creation/   
 modification, terminated account usage, account lifecycle events, separation of duty

- **Encryption**: salted hashes; encrypted traffic; encrypted keys/data/session identifiers

- **Certificate management**: creation -> storage -> dissemination -> suspension -> revocation

- **Active defense**: IdP notifies account owners/SPs; SPs respond to IdP/authorisation   
 system/account compromise

**2.2 Explain software assurance best practices**

- Platforms:

- **Mobile**:

- **Web application**

- **Client/server**

- **Embedded**

- **SoC**

- **Firmware**

- **SDLC integration**: requirements/criteria definition; secure design; static analysis and peer   
 code review; testing & analysis + user acceptance testing

- **DevSecOps**: identify vulnerabilities; find & prioritise risk remediation; secure workflow

- Software assessment methods:

- **User acceptance testing**: ensures software users are satisfied with the functionalities

- **Stress test application**: ensure application availability and scalability; maximum load

- **Security regression testing**: ensure no new vulnerabilities/misconfigurations are   
 introduced by patches/updates (e.g. change control, VCS, SCM)

- **Code review**: pair programming; over-the-shoulder; pass-around; tool-assisted

- Secure coding best practices:

- **Input validation**: validate all untrusted data; specify character sets + data types/length;   
 whitelist allowed characters; additional controls for hazardous characters

- **Output encoding**: encode all unsafe characters; sanitise SQL, XML queries & OS cmds

- **Session management**: short session inactivity timeout; new session identifier   
 generation; logout available from any authorised page; secure session ID algorithms

- **Authentication**: central, segregated authentication; POST requests; unspecific error   
 codes; encrypted & securely stored (salted hash) credentials

- **Data protection**: least privilege; protect/purge sensitive caches; secure encryption; no   
 plaintext password storage; disable client-side caching; access controls for sensitive data

- **Parameterised queries**: use placeholders to separate query and data => prevents SQL   
 query altering (SQLi)

- **Static analysis tools**: thorough white-box code review to identify programming errors

- **Dynamic analysis tools**: test inputs during code execution for complex vulnerabilities

- **Formal methods for verification of critical software**: Fagan inspection (planning ->   
 overview -> preparation -> meeting -> rework -> follow-up)

- Service-oriented architecture:

- **SAML**: message confidentiality & integrity (TLS); validate protocol, signatures etc.

- **SOAP**: exchange structured info for web services (extensibility [extensions] + neutrality   
 [over any app/transport layer protocol] + independence [any programming model])

- Token-based/digest authentication; validate digital signatures; encrypt data with keys

- **REST**: access & manipulate textual representations of web resources with HTTP

- HTTPS; access control; API keys; whitelist HTTP methods; input validation

- **Microservices**: app is a collection of loosely coupled services; lightweight protocols

- IAM with OAuth; defense in depth; use open source crypto libraries; automatic security   
 updates; distributed monitoring/scanning; single point of entry (API gateway)

**2.3 Explain hardware assurance best practices**

- Hardware root of trust:

- **TPM**: generates/stores cryptographic keys; full disk encryption; keeps hardware locked   
 until authentication is complete; motherboard-embedded chip

- **HSM**: manage/generate/store cryptographic keys; removeable/external device

- **eFuse**: manufacturer can change circuits on a chip while it is in operation

- **UEFI**: secure boot (only signed apps used at boot; OS needs recognised key to boot)

- **Trusted foundry**: DoD program to secure supply chain of hardware for military

- Secure processing:

- **Trusted execution**: assures OS trust using TPM; prevents system/BIOS code corruption   
 or platform configuration modification from stealing sensitive data (Intel)

- **Secure enclave**: separately booted microkernel to store private decryption keys; apps   
 never have direct access to the keys (Apple)

- **Processor security extensions**: core can switch to secure state (only trusted code can   
 run; can access secure memory; strict security state entry control) (ARM)

- **Atomic execution**: cannot be interrupted by other threads; thread locking; shared data   
 is always valid => thread safety

- **Anti-tamper**: unusual screws/bolts; secure cryptoprocessors; zeroise when tampered;   
 chips can't be accessed externally; fracture when interfered

- **Self-encrypting drive**: user password to decrypt media; encrypt as data is written and   
 decrypt as data is retrieved; encryption is invisible to user (can't be turned off)

- **Trusted firmware updates**: copy images from non-secure to secure memory; image   
 identification/authentication (Intel)

- **Measured boot and attestation**: object signature hashes are recorded in TPM (measured   
 boot); host reliably authenticates hardware/software config to remote server to   
 determine level of trust in platform (remote attestation)

- **Bus encryption**: encrypted instructions in data bus; executed by cryptoprocessor

**3.0 Security Operations and Monitoring**

**3.1 Given a scenario, analyse data as part of security monitoring activities**

- **Heuristics**: detects unknown (no signature) threats based on their behaviour

- **Trend analysis**: identifies unexpected changes that don't match expected growth rates;   
 predicts behaviours based on existing data (e.g. network congestion based on bandwidth)

- Endpoint:

- Malware:

- **Reverse engineering**: sandboxing; code detonation; software fingerprinting to compare   
 malware against existing hashes; decompilers/disassemblers

- **Memory**: monitor process memory consumption & set thresholds; prevent   
 BOF/insufficient memory allocation & memory leaks (causes app/system crash)

- System and application behaviour:

- **Known-good behaviour**: establish baselines to compare against for anomalies

- **Anomalous behaviour**: suspicious activity that deviates from the baseline model

- **Exploit techniques**: memory overflows; DoS; beaconing (botnet); data exfiltration;   
 privilege escalation; new accounts etc.

- **File system**: FIM; file creation/modification/deletion; prevent drive capacity outage

- **UEBA**: pattern-based user activity anomaly detection (for insider threats; detecting if   
 attacker has compromised system/breaches/brute-forces/super-user creations)

- Network:

- URL and DNS analysis:

- **Dynamically generated algorithms**: malware creates a large number of domain names   
 to connect to C2 servers => harder botnet control; uses datetime, words etc.

- **Flow analysis**: monitor bandwidth, flow sources, utilisation, endpoints, applications

- Packet and protocol analysis:

- **Malware**: check destination IP address/port, protocols, flag fields, sequence no. etc.

- Log review:

- **Event logs**: logins, service start/stop, file activity, rights usage; Windows (application   
 logs, security logs, setup logs, system logs, ForwardedEvents logs)

- **Syslog**: 8 log levels (EACEWNID); event notification (facility [log generator] + severity)

- **Firewall logs**: successful/blocked traffic characteristics; threat attempts; bandwidth use

- **WAF**: web traffic; scalability thresholds; detailed requests log (e.g. status, header info)

- **Proxy**: user/app requests; user agents; HTTP methods; response length; resource access

- **IDS/IPS**: attack attempts alert; attack types/sources, target devices; trends

- Impact analysis:  
 - **Organisation impact vs localised impact**: threat has organisational scope vs local scope

- **Immediate vs total**: impact of threat when activated vs until resolved

- SIEM review:

- **Rule writing**: take action (e.g. trigger alert) if event occurs => quick incident response

- **Known-bad IP**: global blacklists of suspected malicious IPs/URLs; reputation analysis

- **Dashboard**: overview of aggregated info; customise to include important events, graphs

- Query writing:

- **String search**: searches in (specified) columns & tables for string (wildcards/conditions)

- **Script**: use languages to query for items from event logs (according to e.g. time, severity)

- **Piping**: redirects output as input to following command for filtering/sorting/aggregating

- E-mail analysis:

- **Malicious payload**: antivirus + email gateway (ML + real-time IP reputation) +   
 attachment scanning (sandboxing; behaviour-based analysis)

- **DKIM**: receiver checks that domain owner indeed sent/authorised the email + assures   
 message/attachments weren't modified (encrypted signature)

- **DMARC**: prevents spam/spoofing/phishing through DMARC policies; defines email   
 authentication, actions on failed emails, reporting (XML statistics; message copies)

- **SPF**: prevents spammers sending emails on behalf of domain; publishes authorised mail   
 servers (allowed to send on behalf of domain); gives receivers trust info on email origin

- **Phishing**: source IP; URLs; attachments; typosquatting; sending domains (SPF)

- **Forwarding**: compromised inbox automatically forwards received email to attacker

- **Digital signature**: ensures sender authenticity + prevents message tampering (unique)

- **E-mail signature block**: customisable text at bottom of email (not unique)

- **Embedded links**: URL analysis to identify known spam/threat against blacklist

- **Impersonation**: prevent spoofing (SPF/DKIM/DMARC) + user education (check address)

- **Header**: fields (e.g. Received, Reply-To, Return-Path, SPF, X-Mailer, X-Distribution)

**3.2 Given a scenario, implement configuration changes to existing controls to improve   
 security**

- **Permissions**: DAC (end users can delegate/control permissions); MAC (end users cannot   
 modify permissions); RBAC (rights granted to roles) => limits access/functions

- **Whitelisting**: only allows specific IP/MAC addresses, apps, files, emails (more strict)

- **Blacklisting**: prevents specific IP/MAC addresses, apps, files, emails (simple, less secure)

- **Firewall**: add stateful filtering rules/ACLs; prevent traffic based on 5-tuple or L7 content

- **IPS rules**: connection-based block; rules to identify known attack signatures => action

- **DLP**: detects/prevents sensitive data exfiltration; compliance; data tracking/visibility

- **EDR**: detects endpoint activities/events for visibility (signature-based/behavioural   
 analysis) + context with threat intelligence => quick incident response

- **NAC**: 802.1x; agent-based (requesting devices needs special software) or agentless (web   
 browser authentication); in-band (dedicated appliances) or out-of-band (existing network   
 infrastructure)

- **Sinkholing**: DNS server responds with IP address of sinkhole system which remediates   
 botnet-infected system looking for C2 server

- Malware signatures:

- **Development/rule writing**: record malware identifiers (e.g. unique strings, malware   
 families, resources within malware, called function bytes)

- **Sandboxing**: detects unknown malware based on behaviours, not signatures

- **Port security**: restricts source MAC addresses that can connect to port; static or dynamic   
 filtering (e.g. maximum no. of MAC addresses, MAC address moved to different port)

**3.3 Explain the importance of proactive threat hunting**

- **Establishing a hypothesis**: intelligence-driven (TTPs through IOCs); awareness-driven   
 (network changes, most important assets); analytics-driven (models to avoid bias)

- **Profiling threat actors and activities**: motivations, objectives, targets, geolocations,   
 languages, budget, technical skills => relevancy to organisation & threat severity

- Threat hunting tactics:

- **Executable process analysis**: behaviour anomaly analysis (execution path, parent name)

- **Reducing the attack surface area**: eliminate complexity; attack simulation; endpoint   
 visibility + network policies; network segmentation; assessments & traffic analysis

- **Bundling critical assets**: assets grouped together for ease of management & control

- **Attack vectors**: how attacker compromises systems through exploiting vulnerabilities

- **Integrated intelligence**: knowledge + info + collaboration => rapid actionable intelligence

- **Improved detection capabilities**: detect unidentified threat activity based on TTP analysis

**3.4 Compare and contrast automation concepts and technologies**

- **Workflow orchestration**: scalable cloud resource provisioning to achieve business targets

- **Scripting**: programming languages to automatically manage tasks, e.g. configure devices

- **API integration**: controller interaction with systems; seamless connectivity between apps

- **Automated malware signature creation**: inbound unknown file monitoring for file   
 behaviour & content classifiers; signature generated based on malware classification

- **Data enrichment**: add context to data (e.g. asset inventory tools, 3rd party databases) =>   
 meaningful insights + threat prioritisation + quick investigation/action

- **Threat feed combination**: combine machine data from many sources to SIEM, UEBA

- **Machine learning**: finds patterns in data; threat anomaly monitoring; detects unidentified   
 malware; analyses encrypted traffic; make predictions based on activity

- Use of automation protocols and standards:

- **SCAP**: security automation with languages (OVAL), enumeration (CVE, CPE, CCE), metrics   
 (CVSS), integrity (TMSAD for authentication & traceability of security data)

- **Continuous integration**: frequent code commits; automatic code testing; master code   
 branch remains production-ready

- **Continuous deployment/delivery**: deliver & deploy ASAP; identical development + test +   
 production environment configuration

**4.0 Incident Response**

**4.1 Explain the importance of the incident response process**

- Communication plan:

- **Limiting communication to trusted parties**: law enforcement, information sharing   
 partners (ISAC), vendors/manufacturers, actual/potential victims, media <= policies

- **Disclosing based on regulatory/legislative requirements**: data breach notification laws

- **Preventing inadvertent release of information**: always consult legal counsel/public   
 relations before communicating with law enforcement, media, public etc. <= controls

- **Using a secure method of communication**: security-tested messaging platforms;   
 message retention/monitoring/response

- **Reporting requirements**: regulations; classification/storage/retention/expiration policies

- Response coordination with relevant entities:

- **Legal**: ensures team complies with laws/policies/regulations + leader compliance advice

- **Human resources**: investigates potential employee malfeasance

- **Public relations**: coordinate communications with the media & the public

- **Internal and external**: within team for rapid response + externally for advice/regulatory

- **Law enforcement**: when incident has criminal nature => investigation cooperation

- **Senior leadership**: makes critical business decisions; allocates budget & staff; comms

- **Regulatory bodies**: provides advice/guidance on regulatory/legal compliance

- Factors contributing to data criticality:

- **PII**: info which can distinguish an individual's identity, e.g. name, SSN, DoB, addresses

- **PHI**: HIPAA-regulated individuals' health info, e.g. medical records, health conditions

- **SPI**: doesn't identify individual, but is private/can harm person if made public

- **High value asset**: critical info; serious impact to organisation's business/mission ability

- **Financial information**: private info about assets, payments, cards, accounts etc.

- **Intellectual property**: proprietary product development plans, formulae, trade secrets

- **Corporate information**: sensitive info, e.g. corporate accounting, merger/acquisition

**4.2 Given a scenario, apply the appropriate incident response procedure**

- Preparation:

- **Training**: appropriate training on roles & responsibilities; incident preparation

- **Testing**: incident response drill scenarios, mock data breaches => IR plan evaluation

- **Documentation of procedures**: tactical details prepared & used during incidents

- Detection and analysis:

- **Characteristics contributing to severity level classification**: functional impact, economic   
 impact, recoverability effort, data (information) impact rating

- **Downtime**: amount of time that service is unavailable; time until recovery

- **Recovery time**: possibility/predictability of recovery time; resource requirements

- **Data integrity**: modification or deletion of sensitive/proprietary/regulatory/legal info

- **Economic**: financial losses classified according to thresholds

- **System process criticality**: prioritise systems based on how vital it is to operation

- **Reverse engineering**: analyse malware, identify how it works => establish IOCs for rules

- **Data correlation**: info from multiple sources => centrally analyse to identify attacks

- Containment:

- **Segmentation**: network segmentation with firewalls; isolate attacker to quarantine   
 network (strictly controlled VLAN for compromised host analysis)

- **Isolation**: allow attacker access to systems (quarantine network via Internet) but restrict   
 access to other systems, e.g. sandbox, honeypot

- Eradication and recovery:

- **Vulnerability mitigation**: perform vuln scans; protect systems against future attacks

- **Sanitisation**: clear (sanitise against simple recovery, factory reset); purge (prevent even

laboratory recovery, e.g. degaussing); destroy (unable for re-use, e.g. melting)

- **Reconstruction/reimaging**: all compromised hosts should be rebuilt from scratch/known   
 trusted backup; ensure backups don't re-introduce the vulnerability

- **Secure disposal**: encrypt/delete => physically destroy media => 3rd party collector

- **Patching**: patch directly involved systems -> indirectly involved systems -> other systems

- **Restoration of permissions**: perform account review; check for principle of least   
 privilege violations; ensure only authorised user accounts exist on every system

- **Reconstitution of resources**: rebuild systems and apply updates and patches

- **Restoration of capabilities and services**: bring affected systems back into production

- **Verification of logging/communication to security monitoring**: configured to meet   
 logging policy requirements; check centralised log receipt; log automation

- Post-incident activities:

- **Evidence retention**: follow retention policies (no court use); consult legal counsel before   
 discarding (prosecution); US government agencies must retain records for 3 years

- **Lessons learned report**: evaluates how incident response was performed; suggest   
 improvements in the future; evaluate plan/procedure effectiveness

- **Change control process**: document emergency changes that bypassed normal   
 configuration management/change control process (return to them post-incident)

- **Incident response plan update**: find plan deficiencies; make changes to IR plan

- **Incident summary report**: useful in new security control development/training; legal   
 record; previously undetected deficiencies; event timeline + root cause + evidences +   
 actions & their reasons + validation results + lessons learned

- **IOC generation**: IOCs based on network/host artifacts, addresses, hashes, tools, TTPs

- **Monitoring**: full network visibility; continuous monitoring for future persistent attack

**4.3 Given an incident, analyse potential indicators of compromise**

- Network-related:

- **Bandwidth consumption**: causes service outages/disruptions => flow data tools,   
 threshold-based alarms, real-time graphs, SNMP device-level load monitoring

- **Beaconing**: HTTP/S traffic sent to C2 server from a botnet system => IDPS with known C2   
 server rules, behaviour-based analysis, outbound traffic analysis

- **Irregular peer-to-peer communication**: P2P botnets => DNS lookup anomaly detection

- **Rogue device on the network**: wired/wireless rogues => validate MAC addresses to   
 whitelist, OUI checking, network scans, site surveys, traffic analysis, port security/NAC

- **Scan/sweep**: port scanning, repeated requests etc. => IDPS + SIEM (attack correlation)

- **Unusual traffic spike**: scan/attack traffic => anomaly/heuristics detect; protocol analysis

- **Common protocol over non-standard port**: exploit/exfil route or vulnerable service

- Host-related:

- **Processor consumption**: new software/process or DoS => CPU utilisation/processes   
 using CPU/process runtime/spike monitoring

- **Memory consumption**: insufficient memory allocation/memory leaks (-> crash) =>   
 memory consumption/processes monitoring, thresholds & alarms, periodic restarts

- **Drive capacity consumption**: outage => real-time disk utilisation monitoring (e.g. SCOM,   
 Nagios), daily reports (SCCM)

- **Unauthorised software**: SCCM (central installation management/reporting),   
 antimalware, file blacklisting/app whitelisting (limit installations)

- **Malicious process**: compromised host => antimalware, process monitoring

- **Unauthorised change**: file creation, setting changes => logs, SIM/SIEM, FIM, monitoring

- **Unauthorised privilege**: privilege use attempts, escalation => SIM/SIEM, log + analysis

- **Data exfiltration**: big outbound comms => anomaly detection, outbound IDPS rules, DLP

- **Abnormal OS process behaviour**: unusual process/command execution => attacker use   
 of system (for e.g. data exfiltration/privilege escalation/remote execution/enumeration)

- **File system change or anomaly**: new/removed/modified files (e.g. malware) => FIM

- **Registry change or anomaly**: persistence (auto-start) => RegMon, registry monitoring

- **Unauthorised scheduled task**: adware, persistence => Task Scheduler/event monitoring

- Application-related:

- **Anomalous activity**: log analysis, baseline anomaly detection, FIM, user training

- **Introduction of new accounts**: admin account creation approvals & change   
 management workflows, user creation logs, granted privileges tracking

- **Unexpected output**: improper output/errors/issues => output validation by admin

- **Unexpected outbound communication**: beaconing, data exfiltration => network   
 monitoring, outbound IDPS rules, pattern-based behaviour analysis

- **Service interruption**: app/server restart, DoS => app/service status monitoring

- **Application log**: Windows app log (SCOM), /var/log, transactional logs, error messages

**4.4 Given a scenario, utilise basic digital forensics techniques**

- Network:

- **Wireshark**: GUI tool to apply filters, reassemble streams, search captured packets

- **tcpdump**: CLI tool for capturing & analysing PCAP traffic + advanced header filtering

- Endpoint:

- **Disk**: Registry, autorun keys, MFT, event logs, INDX files, change logs, volume shadow   
 copies, user artifacts, Recycle Bin, hibernation files/memory dumps, temporary   
 directories, app logs, removable devices

- **Memory**: Linux kernel extensions fmem & LiME(access to physical memory and copy   
 data); Windows DumpIt (copy physical memory to USB) & crash dump (%SystemRoot%

\MEMORY.DMP, live memory); Volatility Framework (extract encryption keys, user   
 activity/rootkit analysis)

- **Mobile**: physical (acquire SIM card, memory cards, backups); logical (image of logical   
 storage volumes); manual access (review/record unlocked phone); filesystem (deleted   
 files & existing files details [e.g. search histories, messages, call records])

- **Cloud**: determine contract info regarding investigations -> legal recourse with vendor ->   
 identify data & their availability -> work with vendor

- **Virtualisation**: easy disk/memory images with snapshots; dead vs live analysis

- **Legal hold**: obligation to preserve electronic data for legal investigation

- **Procedures**: form problem statement -> determine required data & their locations ->   
 document & review plan -> acquire & preserve evidence -> initial analysis & track actions   
 -> deeper investigation & review missing/additional data -> report on findings

- Hashing:

- **Changes to binaries**: compare hashes (MD5/SHA1) to ensure integrity (chain of custody)

- **Carving**: extract files from unallocated space with magic numbers; cluster-based (file start   
 near FAT/NTFS cluster boundary), sector-based (de-clustered files), byte-based (file in file)

- **Data acquisition**: copies all (used, slack, unallocated) spaces; dd/FTK Imager + write   
 blocker; forensic copy devices (duplicate) => compare both hashes, chain of custody

**5.0 Compliance and Assessment**

**5.1 Understand the importance of data privacy and protection**

- **Privacy vs security**: personal data collection/sharing vs protect data against illegal access

- Non-technical controls:

- **Classification**: classification schema based on risk after breach (e.g. secret, sensitive)

- **Ownership**: ownership of info created/used by organisation; owner must protect data

- **Retention**: what info is maintained; length of time data categories are retained for

- **Data types**: regulatory (PII, PHI, cards), intellectual property, corporate confidential info

- **Retention standards**: according to law/regulation/industry category, global compliance

- **Confidentiality**: prevent unauthorised access/disclosure/theft of privacy information

- **Legal requirements**: Privacy Act, FERPA, HIPAA, PCI DSS, GLBA, SOX, notification laws

- **Data sovereignty**: privacy data in another country that is subject to local laws

- **Data minimisation**: collected data shouldn't be held/used unless clearly stated (GDPR)

- **Purpose limitation**: data collected for specified, legitimate, explicit purposes & not   
 further processed in a way not compatible with the purposes (GDPR)

- **NDA**: legal contract that prevents sharing confidential data (e.g. IP) with 3rd parties

- Technical controls:

- **Encryption**: symmetric/public-key encryption, secure key management, key size

- **DLP**: detects/prevents sensitive data exfiltration; compliance; data tracking/visibility

- **Data masking**: structurally similar but inauthentic version of data; for testing/training

- **Deidentification**: separate PII from PHI; deidentified data can be HIPAA non-compliant

- **Tokenisation**: swap sensitive data (cloud vault) with random numbers (and swap back)

- DRM:

- **Watermarking**: steganographically in video/audio; integrity, ownership, licensed user

- **Geographic access requirements**: checks geolocation with system/IP address or GPS

- **Access controls**: A&A, logging, least privilege, MFA, MAC/DAC/RBAC etc.

**5.2 Given a scenario, apply security concepts in support of organisational risk mitigation**

- **Business impact analysis**: identify critical technologies/processes, prioritisation, recovery   
 time objectives, financial/operational/legal impact, requirements for recovery

- **Risk identification process**: determine/document/communicate potential risks

- Risk calculation:

- **Probability**: likelihood that threat will execute attack + risk having adverse effects

- **Magnitude**: the adversity of the impact the risk has on the organisation

- **Communication of risk factors**: consult stakeholders; decision makers avoid risky practice  
 - Risk prioritisation:

- **Security controls**: prioritise upon manageability (risk control vs risk occurrence time)

- **Engineering tradeoffs**: risk mitigation costs vs ALE; based on risk appetite

- **Systems assessment**: prioritise assets, identify vulnerabilities, assess control & impact

- **Documented compensating controls**: mitigates risk for noncompliant exceptions

- Training and exercises:

- **Red team**: attacker that attempts to gain access to protected network

- **Blue team**: secure target environment and keep red team out

- **White team**: coordinate/maintain/referee the wargame, and monitor results

- **Tabletop exercise**: role/responsibility/response discussions in emergency simulations

- Supply chain assessment:

- **Vendor due diligence**: evaluate risks involved in partnership with potential vendor

- **Hardware source authenticity**: NSA certified Trusted Foundry secure production OEMs

**5.3 Explain the importance of frameworks, policies, procedures, and controls**

- Frameworks:

- **Risk-based**: controls designed around specific risks => flexibility, unaddressed risks

- **Prescriptive**: single requirement list that must be addressed => standardisation, costly

- Policies and procedures:

- **Code of conduct/ethics**: employee accountable for own behaviour; support values,   
 principles, standards; ethical/legal decision making; restricted info disclosure

- **AUP**: clear directions on permissible uses of resources

- **Password policy**: password length/complexity requirements, reuse limitation

- **Data ownership**: states the ownership of the info created/used by the organisation

- **Data retention**: what info is maintained & length of time categories are retained for

- **Account management**: account lifecycle (provision => active use => decommission)

- **Continuous monitoring**: how monitoring is performed; monitoring technology usage

- **Work product retention**: review/retention period/destruction for documents

- Control types:

- **Managerial**: security assessment, planning, risk identification, evaluation of controls

- **Operational**: practices and procedures that follow security requirements

- **Technical**: systems/devices/software/settings etc. that enforce CIA requirements

- **Preventative**: proactive measures to prevent incidents, e.g. firewalls, training

- **Detective**: detects and captures information on incidents, e.g. alarms, notifications

- **Responsive**: responds to breach and restores initial behaviours of systems, e.g. backups

- **Corrective**: remediates incident or limits damage, e.g. patching, antimalware

- Audits and assessments:

- **Regulatory**: PCI DSS (internal & external vulnerability scanning by professional or ASV)

- **Compliance**: HIPAA, GLBA, SOX, FERPA, FISMA, data breach notification laws