DEFENSIBLE SECURITY ARCHITECTURE

How to implement security principles?

Barbara Krašovec

TABLE
of contents

01. Introduction to defensible security architecture

02. Data security

03. Endpoint security

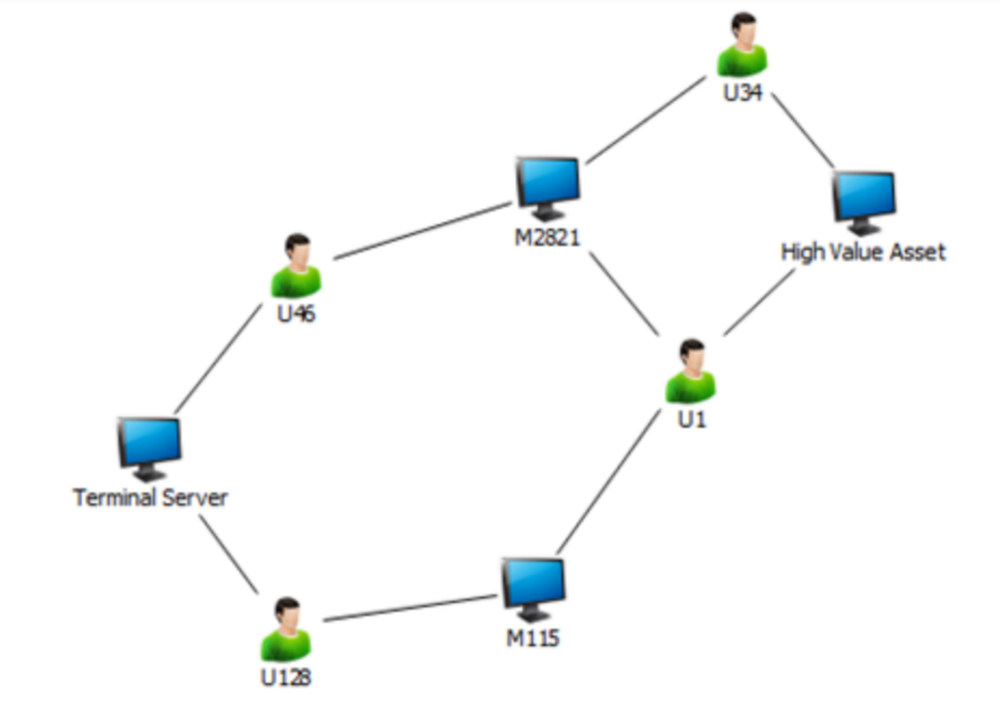
04. Physical security

05. Application security

Think like an attacker
”Think like a chef and see how well you do in the kitchen…”
- Adam Shostack

Defensible security architecture is a security model that designs, builds, operates and defends an infrastructure while continuously applying threat modelling and analysis during each of the process steps in a continuous lifecycle. It is based on the Zero trust principle.

Defenders think in lists, attackers think in graphs.



Defender: “What do I have, and is each thing protected?”

Attacker: “What relationships exist, and what sequence of actions connects me to something valuable?”

- https://medium.com/@johnlatwc/defenders-mindset-319854d10aaa
- https://github.com/JohnLaTwC/Shared

Defensible architecture pillars

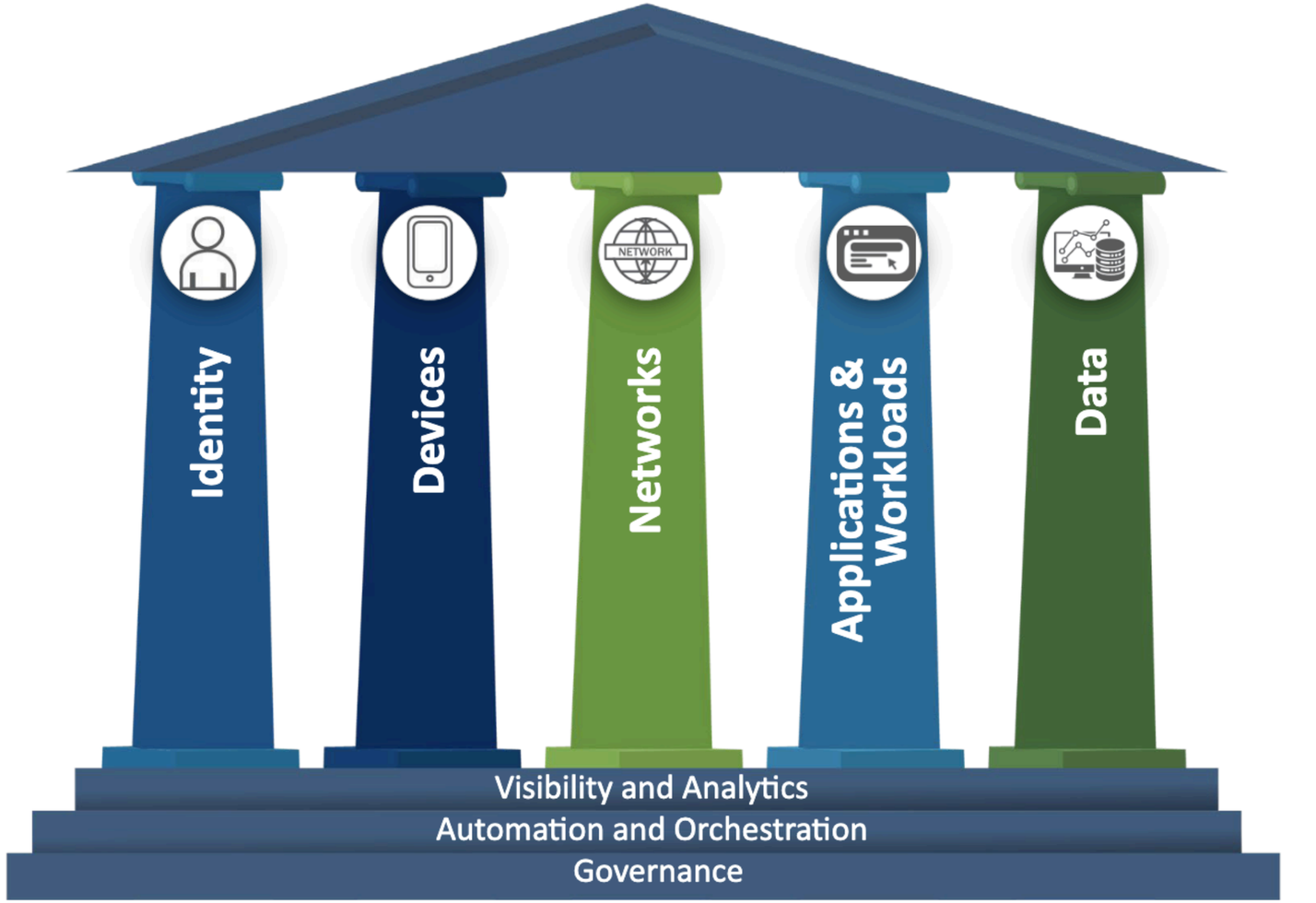
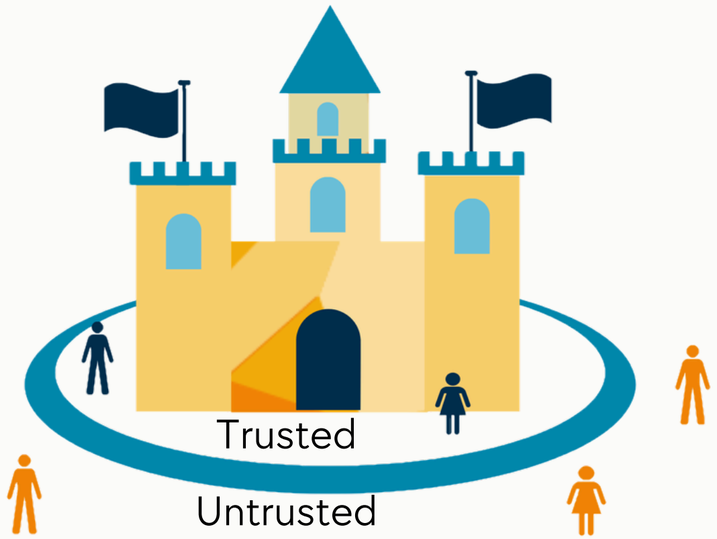


Figure 1: Zero Trust Maturity Model Pillars[8]

Source: https://www.cisa.gov/sites/default/files/2023-04/CISA_Zero_Trust_Maturity_Model_Version_2_508c.pdf

Deficiencies of traditional approach



Everything inside the firewall is trusted.

- How about insider threats?
- How about data breaches?
- What if services are running in the cloud?
- What if data is stored in the cloud?
- What if the equipment is used off-premises?
- What if employees work remotely?

Access control in this model is managed bv VPN. This is a problem:

- VPN becomes a single point of failure
- Traffic is encrypted, difficult to catch a malicious user.
- Encrypted traffic means slower performance
- Based on the server-client, when the server is overloaded, it has impact on usage (scalability problem).
- VPNs require maintenance, on server and client side.

Image source: https://www.clouddirect.net/a-beginners-guide-to-zero-trust/t

Threats

Non-actor driven

Usually unintentional, a result of negative outcomes from operations. Caused by:

- natural disaster
- errors in systems (bugs)
- human error (accidents, negligence)

Actor driven

Usually deliberate/intentional, caused by deliberate actions from actors/groups.

Threat modelling

Strategically thinking about what can go wrong.

Design principles for a defensible architecture

BASIC RULES:

- assign fundamental security principles
- keep it simple
- define critical resources and audit sensitive events
- protect your data
- choose verified security tools (no time for innovation here)
- provide training/education

WHAT TO MONITOR?

- privileged accounts activity
- brute force authentication failures
- authentication anomalies
- session anomalies (spikes on firewall)
- account anomalies
- excessive service failures
- unusual traffic
- signature matching to known vulnerabilities
- insider threat indications
- data exfiltration indicators
- IoC detection

DATA SECURITY

Data security

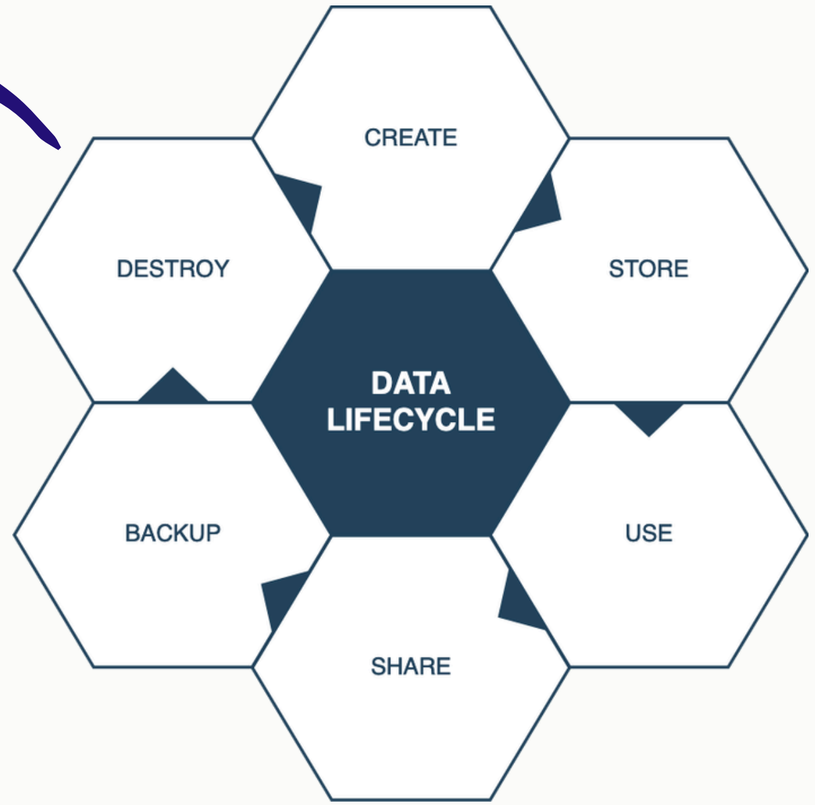
Ponemon Institute evaluates that the average cost of a data breach worldwide is 4.35M dollars.

Data security protects data from malicious threats: activity monitoring, network security, access control, encryption, and authentication.

WHICH DATA?

- **data in motion** (secure transfers + endpoint security of devices used in communication)
- **data at rest** (storage location)
- **data in use, data in memory** (host security)

Data lifecycle



Data security is the process of safeguarding digital information **throughout its entire life cycle** to protect it from corruption, theft, or unauthorized access

GOVERNANCE -Who can create it, what is critical data, how to protect it?
Data security policy, classification, catalogue, resilience

DISCOVERY - where is the data, in DB, files, on the network?

PROTECTION - How to protect it?
Encryption, key management, access controls, backups, replicas etc.

COMPLIANCE
Compliance to the organisation's data and retention policy, regulations, GDPR etc.

DETECTION
Monitor who access data, UBA (User Behaviour analytics) and report if any unusual activity.

RESPONSE
Ensure fast and automatic recovery.

Data integrity

- ***FIM software***

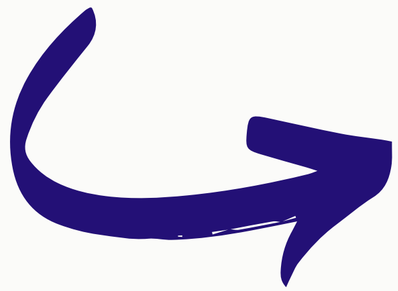
  is a software that monitors and detects file changes that could be indicative of a cyberattack and reports them.

  - FIM software: Tripwire, Samhain, OSSEC
  - you can set audit.rules on Linux, but only check sensible/critical folders

- ***HIDS software***

  stands for a host-based intrusion detection system and represents an application that monitors a computer or network for suspicious activities.

  - HIDS tools: OSSEC, Wazuh, AIDE



PROBLEMS:

- Poorly configured FIM software can lead to excessive alerts causing alert fatigue
- Not suitable for big data

Data protection risk assessment

Risk assessment of the data should be conducted, to identify and evaluate the potential threats and vulnerabilities that may affect the confidentiality, integrity, and availability of data

Regulations: https://commission.europa.eu/law/law-topic/data-protection_en

SOME COMMON THREATS:

- unauthorised access
- data disclosure
- ransomware
- phishing
- insider threat
- data loss
- data corruption
- malware
- software vulnerability
- DoS
- MITM attack
- SQL injection
- data tampering
- data unavailability

- **CLASSIFICATION OF DATA:** based on the severity of the risk and the likelihood of its compromise or exposure.

Data security controls

- provide lifecycle management,
- **IAM** (strong password policy, centralised user and access management, least privilege)
- **access controls** (ACLs, firewall, authN, authZ)
- **data masking**, obscure raw data and only display selected portions during operations
- **monitoring** activity of authorised users
- monitoring any unusual activity (large transfers, access from unusual location etc.)
- carefully chosen data storage **rotation**
- **data policy** for storage, deletion, access
- **endpoint security**
- **auditing** and penetration tests
- secure data transfers (TLS/SSL),
- provide **backup** and **replication**,
- **encryption and key managemen**t (on AWS, newly added resources will be terminated if encryption is not enabled
- security tools: SIEM, FIM.

Backup

- maintain an offline, encrypted backup
- regularly test disaster recovery
- maintain “golden images” of critical systems
- for cloud services, consider a multi-cloud solution to avoid vendor lock-in

COMMON MISTAKES

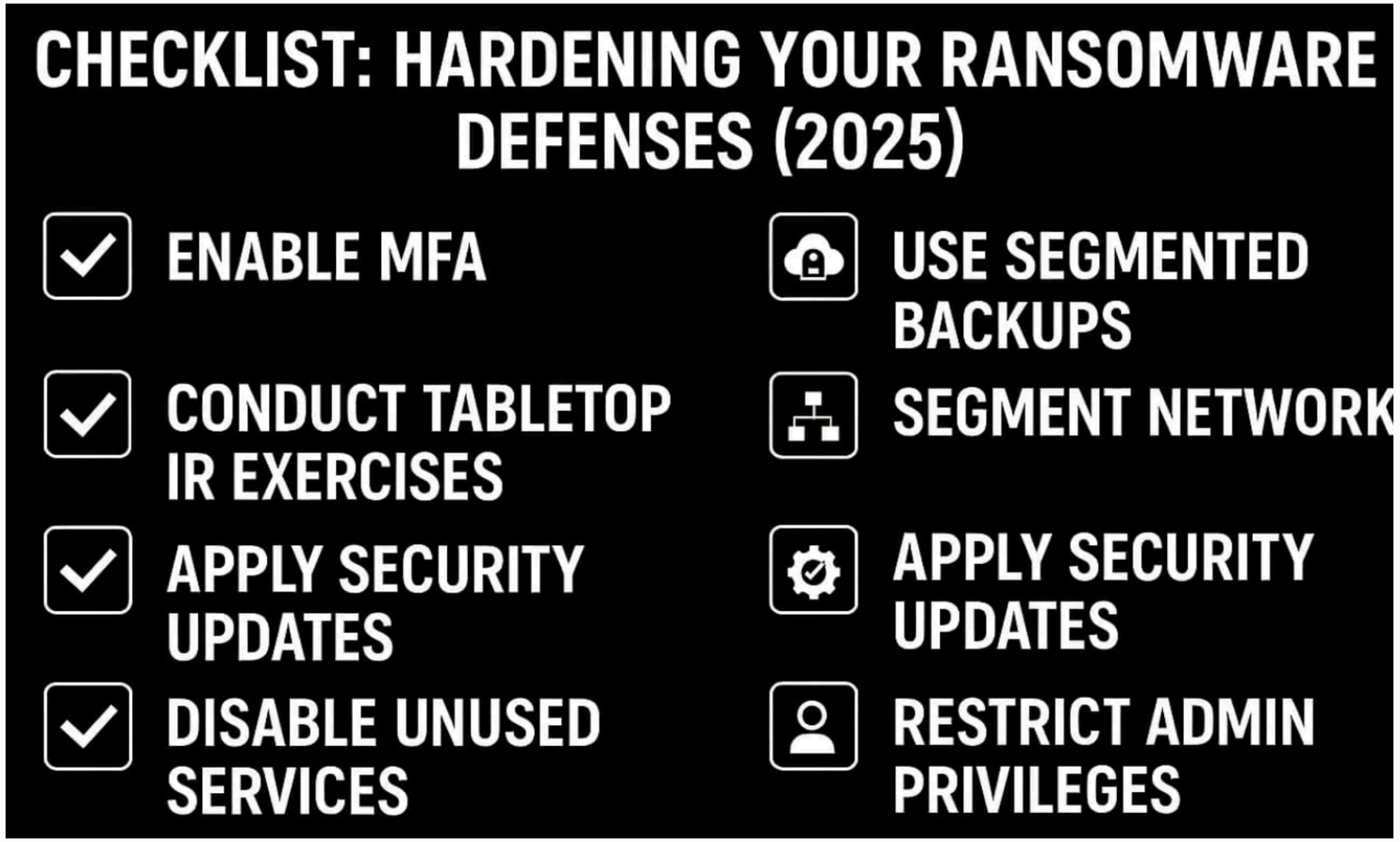
- no backup
- restoring from backup doesn’t work
- backup is very old
- backup is on the same storage as a compromised server

Backup

3-2-1 Rule

- **3 copies of data** – the original and two backups
- **2 different storage types** – a combination of hard drives, the cloud, or tape
- **1 copy outside the organisation** – store one backup completely separate from your system

Ransomware protection



- **Ransomware is malware that encrypts files, demanding payment—often in cryptocurrency—to restore access.**
- **Lateral movement in the network**

Types of multi-tiered ransomware attacks

Traditional ransomware	Double extortion ransomware	Triple extortion ransomware	Quadruple extortion ransomware
■ Encrypts files on the victim's system.	■ Encrypts files on the victim's system.	■ Encrypts files on the victim's system.	■ Encrypts files on the victim's system.
	■ Exfiltrates data and threatens to publish or leak it if the ransom isn't paid.	■ Exfiltrates data and threatens to publish or leak it if the ransom isn't paid.	■ Exfiltrates data and threatens to publish or leak it if the ransom isn't paid.
		■ Threatens to disrupt the victim organization's operations through attacks, such as a DDoS, if the ransom isn't paid.	■ Threatens to disrupt the victim organization's operations through attacks, such as a DDoS, if the ransom isn't paid.
			■ The attackers threaten to notify customers, suppliers, stakeholders and other partners about the breach and might threaten to release the victim's data.

Ransomware protection

Pay the ransom?

- Short answer: NO
- Decryption only works in 60% of the cases (statistics for 2024), 30% of the companies pay.
- Pay or not to pay: https://www.ncsc.gov.uk/guidance/organisations-considering-payment-in-ransomware-incidents

USEFUL LINKS:

- CISA STOP RANSOMWARE: https://www.cisa.gov/stopransomware/ransomware-guide
- CIS CONTROLS: https://www.cisecurity.org/insights/blog/7-steps-to-help-prevent-limit-the-impact-of-ransomware
- National Cyber Security Centre: https://www.ncsc.gov.uk/guidance/mitigating-malware-and-ransomware-attacks
- NIST: Your paragraph text

ENDPOINT SECURITY

What are endpoints?

From a hardware perspective:

- server
- desktop
- IoT device
- network device
- laptop

From a software perspective:

- Linux
- Unix
- Windows
- MacOS

Endpoint security

GOAL OF ENDPOINT SECURITY:

- reduce attack surface
- reduce consequences of potential attack
- detect suspicious activity

Endpoint security refers to securing endpoints from cybersecurity threats. Endpoints can create entry points to organizational networks which cybercriminals can exploit. Endpoint security protects these entry points from malicious attacks. (Kaspersky)

HOW?

- patching
- encryption
- regular updates
- access control
- turning off unnecessary services

Prerequisites

- Asset inventory
- Configuration management
- Change control
- Security policies

Endpoint security controls

SECURITY CONTROLS

- regularly updated asset inventory
- policies
- patching
- encryption
- remote wipe
- location tracing
- EDR - endpoint detection and response
- access controls
- turning off unnecessary services

POLICIES define what is allowed:

- which devices
- which protocols
- which OS
- ..

- Password policies
- Portable device policy
- Acceptable devices
- BYOD
- Encryption
- Password policy

PHYSICAL SECURITY

Physical security

Physical security is the protection of personnel, hardware, software, networks and data from physical actions and events that could cause serious loss or damage to an enterprise, agency or institution. This includes **protection from fire, flood, natural disasters, burglary, theft, vandalism and terrorism**. (definition by Techtarget)

- Is important equipment vulnerable?
- Where can the equipment be used?
- Who is responsible for maintenance?
- Are policies in place for using equipment that leaves the premises?

ISO27001:2022 includes physical and environmental security controls to safeguard information systems from physical threats. It expands on the control related to safe areas to cloud environments.

Physical security - threats

What are the threats that physical security controls should tackle:

- Fire,
- water damage,
- destruction of equipment,
- earthquakes,
- failure of air conditioning,
- loss of power supply,
- remote spying,

- eavesdropping
- tempering,
- disclosure,
- unauthorised use,
- corruption of stored data
- theft
- etc.

Physical security by ISO27k

- **Secure areas (including virtual/cloud):** walls, card-controlled entry gates, physical security for offices, data centres, protection against flood, fire, and earthquake, access control for secure areas, IAM etc.
- **Physical entry controls**: CCTV surveillance, security guards, protective barriers, locks, perimeter intrusion detection, policy on visitor management process etc.
- **Equipment security:** protection against power failures, unauthorised usage, fire protection, clear policies for removable storage media, and policies on data removal that are saved on the equipment.
- **Reuse of equipment:** policy on data erasure and destruction
- **Protection against environmental threats:** controls for monitoring environmental conditions, such as temperature, humidity, air quality

Securing IoT devices

Internet of Things (IoT) devices are Internet-connected objects, such as networked security cameras, smart refrigerators, fire alarms etc.

How to protect them?

IoT devices have firmware and software which is essentially an operating system that needs to be protected:

- Upgrade firmware and software regularly, especially when security patches are available.
- Use strong passwords to access the devices.
- Disable physical access to the devices.
- Use encryption to protect the communication of the device.
- Turn off features that you don't need.

Endpoint security measures

Configuration Management

Process of monitoring/deploying the hardware and software configuration in line with IT policies.

- Enables consistency and automation,
- enables traceability of configuration changes,
- reduces security breaches,
- reduced time to restore service,
- efficient change management,
- easier upgrade automation,
- higher quality of service,
- control over running processes and permissions over the files,
- configuration backup and documentation.

CFEngine
puppet labs
PalletOps
ANSIBLE
CHEF
SALTSTACK

Security policies

Password policies
Software policies
Hardware policies
Data protection policies
Access Policies
AUP

- How many characters are in the password? How often does it need to be changed?
- Which software is obligatory? SELinux installed or disabled? Which is forbidden? Server checklist required?
- Network devices from which vendor? A template for hardware configuration provided?
- When can data be deleted? Who can delete it? Where it will be stored, who will have access to it? From where?
- Which services can run on a server? Which services should be disabled?
- Who can access an endpoint, from where? Who has administrative privileges?
- Get consent from the user for monitoring, using and wiping his data. Also important for the user to know what is allowed and what is the normal operation.

Security policies templates

Document	Who should complete the template?	Audience	Description	Link
Top Level Infrastructure Policy	Infrastructure Management	All Infrastructure Participants (abides by)	This policy template defines the roles of actors in the Research Infrastructure and binds the policy set together	Google Doc
Incident Response Procedure	Infrastructure Management & Security Contact	Infrastructure Security Contact, Services (abides by)	This template procedure provides a step-by-step breakdown of actions to take following a security incident.	Google Doc
Membership Management Policy	Infrastructure Management	Research Community (abides by)	This policy template defines how Research Communities should manage their members, including registration and expiration.	Google Doc
Acceptable Authentication Assurance	Infrastructure Management	Research Community, Services (abide by)	This is a placeholder for the Infrastructure to determine rules for the acceptable assurance profiles of user credentials.	Google Doc
Risk Assessment	Infrastructure Management, Services & Security Contact	Infrastructure Management (completes)	This table can be used as a starting point for identifying whether a full Data Protection Impact Assessment is required.	Google Doc
Policy on the Processing of Personal Data	Infrastructure Management & Data Protection Contact	Research Community, Services (abide by)	This document defines the obligations on Infrastructure Participants when processing personal data.	Google Doc

AuthN and AuthZ

Authentication is the process of verifying who a user is, while authorization is the process of verifying what they have access to

- **single source of truth = IAM system**, used for provisioning and de-provisioning of accounts
  - Approval process (is your account still valid? For how long?)
- secure systems rely on **MFA** -> trend is passwordless authentication (SSO)
  - **something that you have** (access card, OTP generator on mobile device)
  - **something that you are** (fingerprint, face recognition)
  - **something that you know** (password)
- Audit your procedures and monitor user activity

Federated identity - when you connect to multiple service providers using the same identity provider. This is a big problem in case of stolen credentials or compromise.

Change control

FILE INTEGRITY CHECKS

- define which files should be monitored for change
- an attacker will always change some files on the system

File integrity checks use a mathematical function called a one-way hash. Hashes are stored and cannot be modified. At defined intervals, the hash against the original is checked. If the hashes do not match, an alert is issued. An alert is sent also when a new file appears in a folder. Poorly configured FIM and HIDS systems can lead to excessive alerts causing Alert Fatigue.

FIM = file integrity monitoring

- monitors and detects file changes that could be indicative of a cyberattack and reports them

HIDS = host intrusion detection system

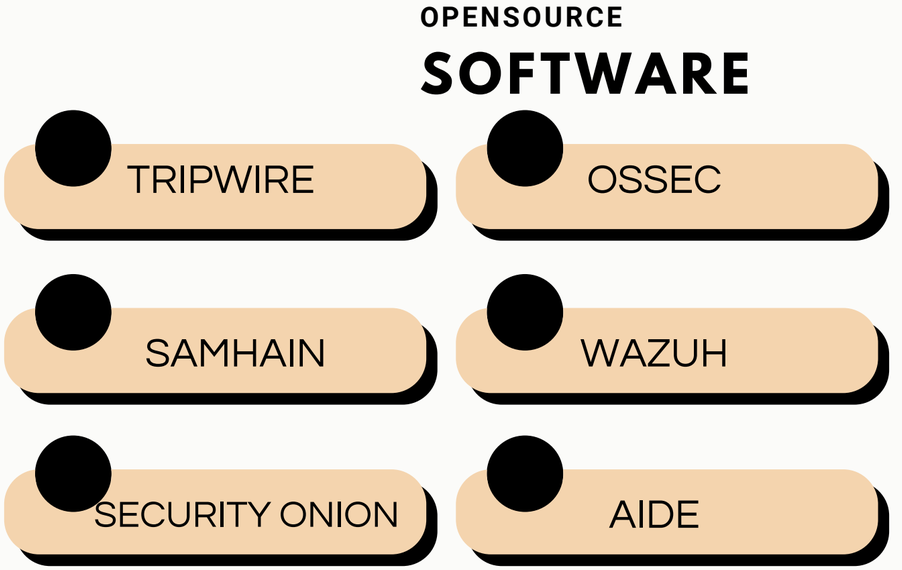
- monitors traffic on the host
- checks file changes, abnormal activity in logs or network traffic

```
Check trusted computing base:
#kernel modules
/lib/modules

#binaries
/bin, /sbin, /usr/bin, /usr/sbin,
/usr/local/bin, /usr/local/sbin

#system configuration
/etc

#critical files in:
/var/spool, /boot, /home
```



Software-based endpoints

Linux host hardening: OS hardening

- modify kernel settings at runtime (sysctl), blacklist uneeded kernel modules,
- network: close unneeded ports, limit access and services (ACLs) also verify which ports are opened
- protect files - minimise access rights, FIM,
- automate what you can, use configuration management tools,
- access: use MFA or at least SSH keys to log in, use auditing, for passwords use strong change policy,
- logging and monitoring: use central logging.
- protect configuration files
- enable SElinux, auditing, and AppArmor to limit the capabilities of programs
- use VPN/Wireguard or similar to connect to other segments
- disable X11
- use also software firewall (nftables, firewalld, iptables etc)
- disable root cron jobs
- check for backdoors and rootkits

Host hardening - packages and services

- **automate OS deployment and configuration** (using Foreman, Puppet, Ansible or others),
- **minimise the number of installed packages** (dnf remove package),
- **disable/mask unused services** (systemctl disable service),
- **update** regularly,
- **apply patches** and verify them with services like Nessus, Pakiti, OpenVas,
- subscribe to and follow security notifications for the software you rely on,
- ensure that all remote server management interfaces (e.g. BMC subsystems) are not exposed externally, access should be restricted to administrative users with ACLs in place. Disable default built-in administrative accounts that might be provided by default by the vendor. A **unique administrative account configuration** for your management systems is usually a good practice.
- regularly **scan for vulnerabilities**.

Patch and vulnerability management

- **EGI SVG vulnerability handling**
  - Reporting a vulnerability: report-vulnerability@egi.eu
  - Public advisories: https://advisories.egi.eu/
  - SVG Issue handling procedure: https://documents.egi.eu/public/ShowDocument?docid=3867
  - Vulnerability management guidelines: https://confluence.egi.eu/display/EGIBG/Vulnerability+Handling+Guide+for+System+Administrators

- Patch cadence policy covering OS, kernel, firmware (BMC / iLO / iDRAC), and driver stacks (GPU, InfiniBand / OPA)
- Plan regular upgrade cycles even if no critical vulnerabilities are reported

Some useful tools:

- CVE tracking and vulnerability scanning: **OpenVAS**, **Nessus**, **Trivy** for container images
- **Pakiti** — centralised patch-status tracking, purpose-built for grid / HPC (https://github.com/CESNET/pakiti-server)
- **Dependency track** - SBOM scanning and dashboard: https://dependencytrack.org/
- **Shodan** to monitor network exposure: https://www.shodan.io/
- **Shadowserver** to check network exposure and vulnerabilities: https://www.shadowserver.org/

CVEs and the kernel

How to verify which CVEs fixes are included in the kernel package:

```
rpm -q --changelog kernel-<version> | grep -i CVE #for RHEL and derivatives

apt changelog linux-image-$(uname -r) #Debian/Ubuntu
```

Check if there are any security updates available:

```
sudo dnf updateinfo list security
sudo dnf updateinfo info CVE-2024-XXXXX # details on a specific CVE
sudo yum --security check-update
```

Check CVEs in CVE/NVD databases using NVD REST API or compare with version in OVAL

```
curl -s "https://services.nvd.nist.gov/rest/json/cves/2.0?keywordSearch=linux+kernel+6.6" |
jq '.vulnerabilities[].cve.id'
```

If you are not sure if the update requires restarting (install yum-utils)

```
needs-restarting -r
```

Host hardening - users and access

- It is poor practice if multiple admins share the same root account and/or use the same password on multiple servers - provide **unique accounts to admins**, and use session recording.
- use **central IAM** for user management,
- user credentials should never be transferred over an insecure channel,
- use **MFA for SSH login**. **Disable password login** to the servers, **disable root login**,
- enable **auditing**,
- ensure that **private keys** (SSH, X.509,...) are not stored passwordless on publicly accessible servers (such as user interfaces or login nodes). Ideally, no private keys should be stored on publicly accessible servers,
- configure **termination for inactive sessions**,
- users should sign **AUP**,
- apply secure **access rules**,
- **lock account after multiple failed attempts** (faillog or fail2ban)
- use **encryption** (sftp, scp, ssh) for file transfer and accessing the machines
- make sure no other user than root has UID 0 (awk -F: '($3 == "0") {print}' /etc/passwd)
- **lock all accounts with empty passwords**

```
for user in `awk –F: '($2 == "") {print}'` /etc/shadow; do passed –l $user;done
```

Provide account provisioning and de-provisioning policies, and use an identity governance system (IAM = central service for account creation/deletion): the user accounts are created, modified and terminated on a single service which ensures consistency. By configuring user accounts for each service locally, it will require a lot of work to keep the validity of users (are they still employed, entitled to use the services?) in sync with their access to the servers.

Admin accounts

- **Holding the admin keys entails responsibility.**

Negligence of administrators:

- logging on insecure endpoints (keyloggers, hacked db, etc.)
- sharing admin accounts
- using admin account for daily tasks
- poor password management
- too many admin accounts on the system
- SSH keys without a password
- using the same SSH keys to access different infrastructures, or never rotated
- Not revoking access when it's no longer needed
- Lack of audit logging / not reviewing sudo logs
- Storing credentials/keys insecurely
- Visibility of plain text passwords in the bash history (eg. when connecting to databases)
- No SSH session timeout / leaving sessions unattended

What is the onboarding/offboarding procedure in your organisation?
What is your policy on key rotation?

What to log?

Logging

- problem are different formats, timestamps, timezones
- use centralised log management, then analyse
- normalise logs (same format for all)
- provide log rotation
- specify log rotation policy (diskspace, regulatory requirements)
- visualise vital logs
- software: NXlogs, ELK, Graylog, Loki, rsyslog, syslog-ng

What happens if you leave your log server open to the Internet?

```
2025-02-19T05:23:44.673996 - - - - vxor.vv loves dragons a bit too much.
Really good dragon porn image (you have been warned) btw [REDACTED URL].
Also this is a friendly reminder that you should fix your security...
```

When you leave 514/tcp and 514/udp open for the world - something may happen eventually.

→ redirection to a phishing site. What else?

Logging checklist

https://www.sans.org/brochure/course/log-management-in-depth/6

Good logging isn't simply "collect as many logs as possible." It should produce useful, trustworthy evidence that can be investigated

Collect the right events → make them trustworthy → correlate them → retain them → actually use them.

SANS CRITICAL LOG REVIEW CHECKLIST FOR SECURITY INCIDENTS

This cheat sheet presents a checklist for reviewing critical logs when responding to a security incident. It can also be used for routine log review.

GENERAL APPROACH

1. Identify which log sources and automated tools you can use during the analysis.
2. Copy log records to a single location where you will be able to review them.
3. Minimize "noise" by removing routine, repetitive log entries from view after confirming that they are benign.
4. Determine whether you can rely on logs' time stamps; consider time zone differences.
5. Focus on recent changes, failures, errors, status changes, access and administration events, and other events unusual for your environment.
6. Go backwards in time from now to reconstruct actions after and before the incident.
7. Correlate activities across different logs to get a comprehensive picture.
8. Develop theories about what occurred; explore logs to confirm or disprove them.

POTENTIAL SECURITY LOG SOURCES

- Server and workstation operating system logs
- Application logs (e.g., web server, database server)
- Security tool logs (e.g., anti-virus, change detection, intrusion detection/prevention system)
- Outbound proxy logs and end-user application logs
- Remember to consider other, non-log sources for security events.

TYPICAL LOG LOCATIONS

- Linux OS and core applications: /var/log
- Windows OS and core applications: Windows Event Log (Security, System, Application)
- Network devices: usually logged via Syslog; some use proprietary locations and formats

WHAT TO LOOK FOR ON LINUX

Successful user login	**"Accepted password", "Accepted publickey", "session opened"**
Failed user login	**"authentication failure", "failed password"**
User log-off	**"session closed"**
User account change or deletion	**"password changed", "new user", "delete user"**
Sudo actions	**"sudo: … COMMAND=…" "FAILED su"**
Service failure	**"failed" or "failure"**

WHAT TO LOOK FOR ON WINDOWS

- Event IDs are listed below for Windows 2000/XP. For Vista/7 security event ID, add 4096 to the event ID.
- Most of the events below are in the Security log; many are only logged on the domain controller.

User logon/logoff events	**Successful logon 528, 540; failed logon 529-537, 539; logoff 538, 551, etc**
User account changes	**Created 624; enabled 626; changed 642; disabled 629; deleted 630**
Password changes	**To self: 628; to others: 627**
Service started or stopped	**7035, 7036, etc.**
Object access denied (if auditing enabled)	**560, 567, etc**

WHAT TO LOOK FOR ON NETWORK DEVICES

- Look at both inbound and outbound activities.
- Examples below show log excerpts from Cisco ASA logs; other devices have similar functionality.

Traffic allowed on firewall	**"Built … connection", "access-list … permitted"**
Traffic blocked on firewall	**"access-list … denied", "deny inbound"; "Deny … by"**
Bytes transferred (large files?)	**"Teardown TCP connection … duration … bytes …"**
Bandwidth and protocol usage	**"limit … exceeded", "CPU utilization"**
Detected attack activity	**"attack from"**
User account changes	**"user added", "user deleted", "User priv level changed"**
Administrator access	**"AAA user …", "User … locked out", "login failed"**

WHAT TO LOOK FOR ON WEB SERVERS

- Excessive access attempts to non-existent files
- Code (SQL, HTML) seen as part of the URL
- Access to extensions you have not implemented
- Web service stopped/started/failed messages
- Access to "risky" pages that accept user input
- Look at logs on all servers in the load balancer pool
- Error code 200 on files that are not yours

Failed user authentication	**Error code 401, 403**
Invalid request	**Error code 400**
Internal server error	**Error code 500**

OTHER RESOURCES

- **Windows event ID lookup:** www.eventid.net
- **A listing of many Windows Security Log events:** ultimatewindowssecurity.com/.../Default.aspx
- **Log analysis references:** www.loganalysis.org
- **A list of open-source log analysis tools:** securitywarriorconsulting.com/logtools
- **Anton Chuvakin's log management blog:** securitywarriorconsulting.com/logmanagementblog
- **Other security incident response-related cheat sheets:** zeltser.com/cheat-sheets

Authored by Anton Chuvakin (chuvakin.org) and Lenny Zeltser (zeltser.com).

Reviewed by Anand Sastry.

Distributed according to the Creative Commons v3 "Attribution" License.

Cheat sheet version 1.0.

APPLICATION SECURITY

Application security

FACTS

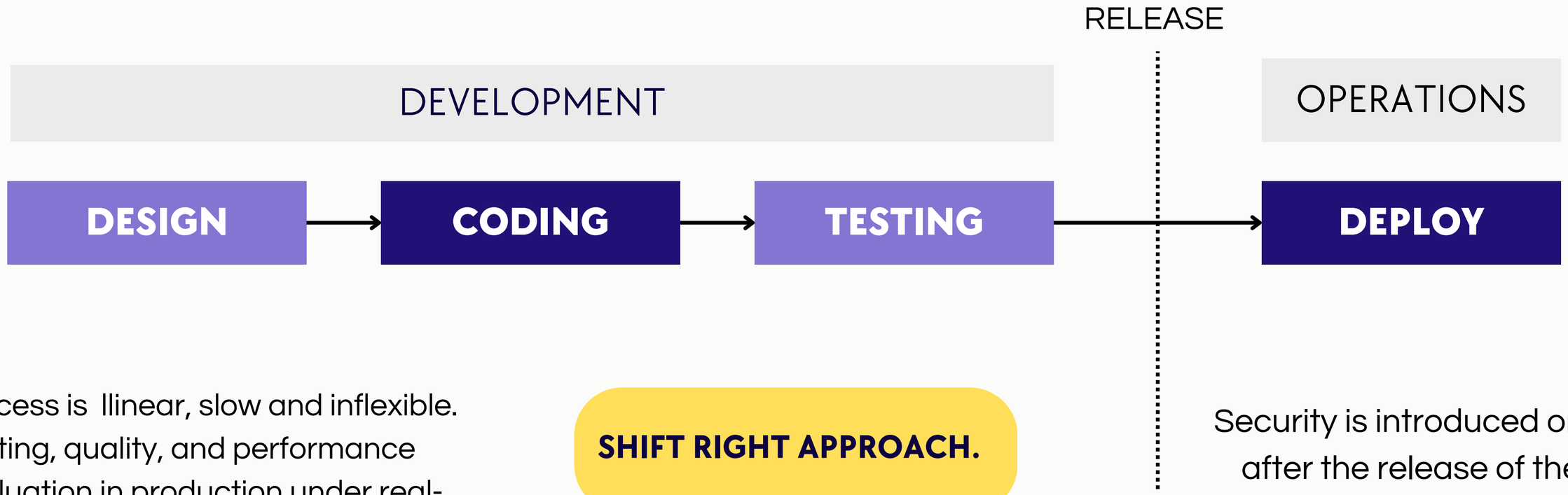
- All software has bugs and vulnerabilities.
- Little software is written from scratch (taken from other sources and modified).
- We are often not aware of the dependencies/libraries used.
- Security must be implemented by design.
- If you find a vulnerability in the development phase, it is less costly as when the software is already released

DEFINITION

Application security (short AppSec) includes all tasks that introduce a secure software development life cycle to development teams.

Objective: to improve security practices and, through that, to find, fix and preferably prevent security issues within applications. It encompasses the whole application life cycle. (source Wikipedia)

Traditional approach to software development

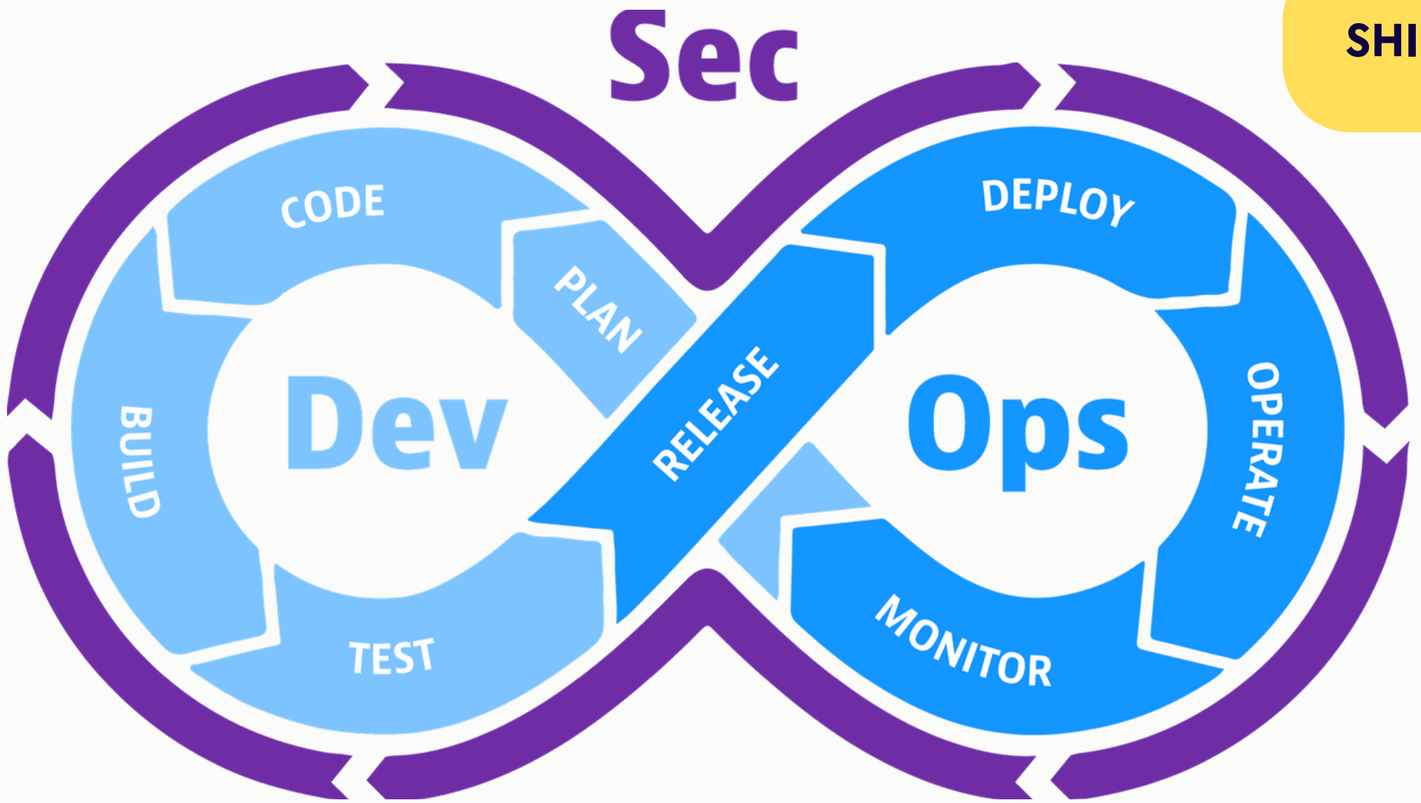


- Process is llinear, slow and inflexible.
- Testing, quality, and performance evaluation in production under real-world conditions

SHIFT RIGHT APPROACH.

Security is introduced only after the release of the software.

Dev-ops approach to software development



SHIFT LEFT APPROACH.

DevOps team guarantees application security at the earliest stages in the development lifecycle, as part of an organizational pattern known as DevSecOps.

Five pillars of DevOps:

- culture,
- automation,
- measurement,
- sharing, and
- learning

Secure Coding Practice Quick-reference Guide

1. Introduction
2. Checklist
2.1 Input validation
2.2 Output encoding
2.3 Authentication and password management
2.4 Session management
2.5 Access control
2.6 Cryptographic practices
2.7 Error handling and logging
2.8 Data protection
2.9 Communication security
2.10 System configuration
2.11 Database security
2.12 File management
2.13 Memory management
2.14 General coding practices

Secure coding

- **OWASP standard**

OWASP Developer Guide: https://owasp.org/www-project-developer-guide/

OWASP Secure Coding Practices Checklist: https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/stable-en/02-checklist/05-checklist.html

- **Recommendations**
- **Take code from trusted sources and use trusted libraries (however log4j)**
- **keep software bill of materials (you want to document all dependencies and where they came from)**
- **Apply security measures before the release of softare**

CI/CD Pipeline Security

- **Secrets management** — never hardcode credentials/API keys/tokens in code or CI config, use a dedicated secrets manager (Vault, AWS Secrets Manager, etc.) with short-lived, scoped credentials
- **Dependency scanning (SCA)** — automatically scan third-party libraries and dependencies for known vulnerabilities on every build (Dependabot, Snyk, OWASP Dependency-Check)
- **Static & dynamic testing in-pipeline** — run SAST on every commit/PR; run DAST against staging before promotion to production
- **Signed builds and artefact integrity** — cryptographically sign build artefacts and verify signatures before deployment, so a tampered artefact can't silently reach production
- **Software Bill of Materials (SBOM)** — generate and store an SBOM per release, so you can quickly answer "are we affected?" when the next log4j-style incident hits
- **Least privilege for pipeline service accounts** — CI/CD runners should only have the permissions needed for their specific job, not broad admin/deploy rights
- **Immutable build environments** — use ephemeral, reproducible build containers rather than long-lived build servers that can drift or be persistently compromised
- **Branch protection & mandatory review** — no direct pushes to main/production branches, require peer review and passing checks before merge

Top 10 vulnerabilities by OWASP

1. Injection
2. Cryptographic failures
3. Insecure Design
4. Broken Access Controls
5. Security Misconfiguration
6. Vulnerable and Outdated Components
7. Identification and Authentication Failures
8. Software and Data Integrity Failures
9. Security Logging and Monitoring Failures
10. Server-Side Request Forgery (SSRF)

- **How to prevent these?**
  - use security scanning tools
  - use encryption
  - use central logging
  - check if authorisation rules apply: verify that a requested action or service is approved for a specific entity
  - ensure proper authentication, login pages must use TLS, users validity should be properly managed - the system needs both defined provisioning and de-provisioning, recommended to use a single source of truth (use SSO when possible)
  - with vulnerability testing (SAST, DAST)

To summarize

- defensible security architecture tackles security on multiple layers and implements security controls based on the fundamental security principles
- combination of organisational, technical controls
- XSS is the main reason for website compromises (implement validation) and most of the AI generated code consists od XSS injections
- phishing is still the main attack vector (fake websites, malicious attachments, QR codes)
- stolen credentials are a problem - in phishing and business email compromises (implement MFA, private keys and increase security awareness)
- monitor your systems for anomalies.

- **THREAT LANDSCAPE**
- **LAYERED DEFENCE**
- **SECURITY AWARENESS**
- **CONSTANT MONITORING**
- **INCIDENT RESPONSE**

Questions?

Some other aspects

HPC Security

- ***Challenges***

- Hard to detect anomalies and inappropriate activities.
- What to monitor?
- No shared threat intelligence and best practices.
- Ensuring a balance between security and performance.
- It is easy "to hide" in the system.
- File integrity - too many files to scan them.
- Cannot really separate development and production environment (too large scale).
- Rapidly changing environment.
- Scalability of consequences of an incident.

- ***Main risks***

- misuse of cycles (e.g. bitcoin mining)
- loss of file integrity,
- intelectual property confidentiality,
- insider threats,
- supply chain attacks.

HPC is traditionally a closed system with little focus on security. With international collaboration, large datasets and access to external storage systems, HPC needs to open some components to the Internet, and security gains importance.

Vendor security assessment

Vendor security assessment

- Make security part of the procurement process and put security requirements in the contract with the vendor.

Assess vendor's security maturity:

- vendor's transparency
- vendor's support in security incidents, vulnerability handling
- vendor's approach to product support (regular maintenance, defined product lifecycle, development process)

What to require from the vendor:

- defined security policies and procedures,
- redundancy for critical services,
- automated processes for building and deployment,
- detailed software documentation,
- controls for software and updates.

Fundamental security measures

- implement zero trust
- conduct regular vulnerability scanning
- update OS/packages regularly
- harden devices and host
- limit the use of RDP and other desktop services (most often entry point)
- for Samba, disable SMBv1 and SMBv2, only use SMBv3 (often used to propagate malware in lateral movement)
- limit SMB traffic, disable external access to SMB/NFS

COMMON MISTAKES

- only a single control is used to protect a service (firewall)
- access to SMB not restricted from internal networks
- no OS hardening in place
- storage systems not in a separate network segment
- also same storage used for multiple services (in DMZ, internal network etc.)

User-related security measures

- educate people on phishing
- apply MFA
- use central IAM (easy to block user's access to other machines in case of compromise
- disable root login
- change default admin credentials
- enforce account lock-out after multiple failed attempts
- store passwords in secure DB with strong hashing algorithms
- don't save passwords in browsers or in plain text files
- implement a strong password policy (password at least 15 characters)

COMMON MISTAKES

- all mentioned not applied

Email security

- implement filters on the email gateway
- use SPAM filters and apply additional rules for recognised phishing emails
- ensure macro scripts are disabled on MS Office files transmitted via email (recent versions of MS Office have this set by default -und Application/Settings/Security)
- provide trainings
- run phishing campaigns

COMMON MISTAKES

- users' ignorance (click on malicious link or they open a malicious attachment)

Longer is stronger.

Password security

- Use a random string of mixed-case letters, numbers and symbols. For example: cerwr.wfXmnZK65rf*&DaaD
- Use passphrases - create a memorable phrase of 4 – 7 unrelated words. For example: Horse in the bay ship blue horizon.
- Make them unique
- Use MFA where possible.

Salting password

Password salting is a technique to protect passwords stored in databases by adding a random salt (some characters)

```
User Password 1 – orange
Salt 1 – y3Unz
Salt added to password 1 – orangey3Unz
Hash ("orange" + salt) 1 - 024ca8e38b33f9116c151123eb432d20
User Password 2 – orange
Salt 2 – 1Hx$
Salt added to password 2 – orange1Hx$
Hash ("orange" + salt) 2 - 5d41402abc4b2a76b9719d911017c592
```

Source: https://www.techtarget.com/searchsecurity/definition/salt