Welcome back to the CCSP Bootcamp

Your instructors:

Michael

and

Dave

Class will begin at 11:00 A.M. Eastern Standard Time (EST)

Domain 4

Cloud Application Security

4.1 Advocate Training and Awareness for Application Security

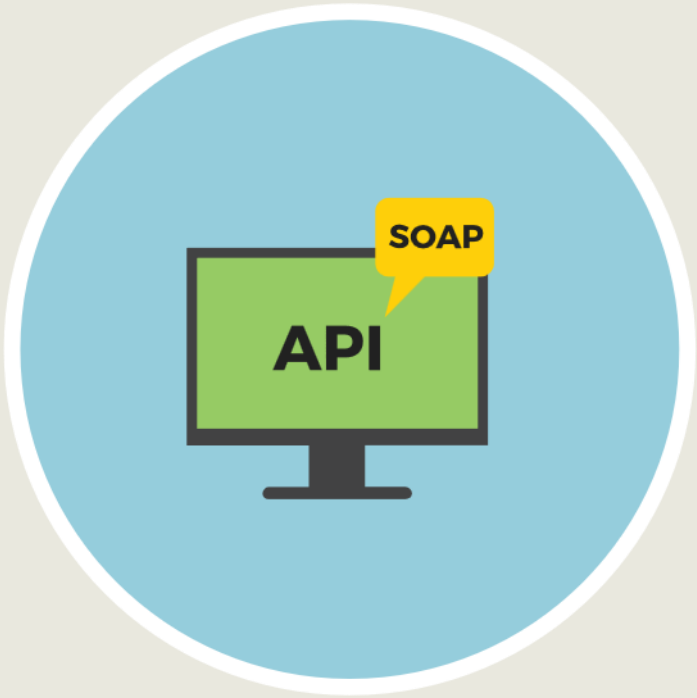
- Cloud Development Basics
- Common Pitfalls
- Common Cloud Vulnerabilities

Cloud Development Basics

- Cloud development usually entails Integrated Development Environments (IDEs), application lifecycle management initiatives, and application security testing
- The developer should ask several questions to determine if the proposed application is “Cloud-friendly”
  - What would the impact be if the data or information crossed geographic boundaries?
  - What if an employee of the cloud provider accessed the data or application?
  - What if the program failed to meet the planned results?
  - What if the app is manipulated by a corporate outsider?
  - What if the data or application were modified unexpectedly?
  - What if the application were subject to downtime?

Understanding APIs

SOAP vs. REST-based APIs



- It is important for developers to understand that in most cloud deployments, access is acquired through the means of an Application Programming Interface (API)
- These APIs will utilize tokens instead of traditional usernames and password credentials
- **Simple Object Access Protocol (SOAP)** uses an envelope then HTTP (or FTP/SMTP) to transfer data; only supports XML format; slower; no caching, scalability can be complex; Used when REST is not feasible
- **Representational State Transfer (REST)** uses simple HTTP protocol and supports many different data formats like JSON, YAML, XML; Restful APIs are widely used; Performance and scalability are good, and it uses caching as well

Common Pitfalls and Vulnerabilities

- Not all applications are ready for the cloud
- The Initial phases of the decision to use PaaS and introduce a SDLC are often rushed
- On-premise does not always migrate or transfer (and vice-versa)
- Lack of training and awareness of new development techniques
- Lack of guidelines, strategy, and documentation
- Difficulty and complexity of integrating with cloud

Common Web Vulnerabilities – OWASP Top 10

1. A1—Injection: Injection flaws like SQL, OS, and LDAP happen when untrusted data is sent to an interpreter as part of a command or query then the attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization
2. A2—Broken Authentication and Session Management: Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities
3. A3—Cross-Site Scripting (XSS): XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation or escaping allowing attackers to execute scripts in the victim's browser, which can hijack user sessions, deface websites, or redirect the user to malicious sites
4. A4—Insecure Direct Object References: A direct object reference happens when a developer exposes a reference to an internal implementation object like a file, directory, or database key - without an access control check or other protection, attackers can manipulate these references to access unauthorized data
5. **Security Misconfiguration: inactive virtual images and disks that have not been updated or patched**

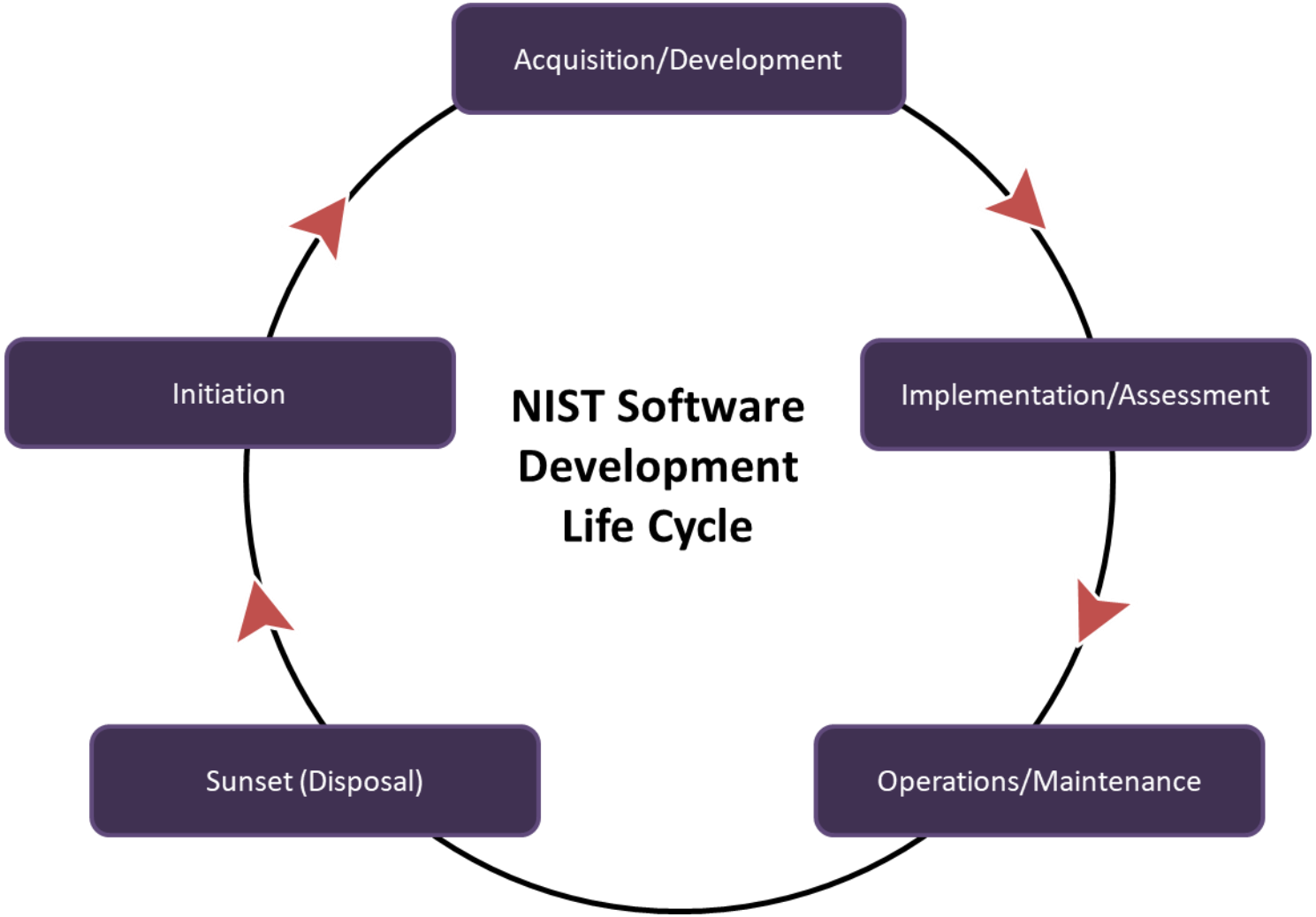
Common Web Vulnerabilities – OWASP Top 10

6. A6—Sensitive Data Exposure: Many web applications do not properly protect sensitive data (credit cards, tax IDs) so attackers may steal or modify such weakly protected data to conduct fraud or identity theft
7. A7—Missing Function Level Access Control: Most web apps verify function-level access rights before making the functionality visible in the UI; However, applications must conduct the same access control checks on the server when each function is accessed
8. A8—Cross-Site Request Forgery (CSRF): This attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application
9. A9—Using Components with Known Vulnerabilities: Components, such as libraries, frameworks, and other software modules, almost always run with full privileges
10. Unvalidated Redirects and Forwards: Web applications frequently redirect and forward users to other pages and websites using untrusted data to determine the destination pages - without proper validation, attackers can redirect victims to phishing or malware sites

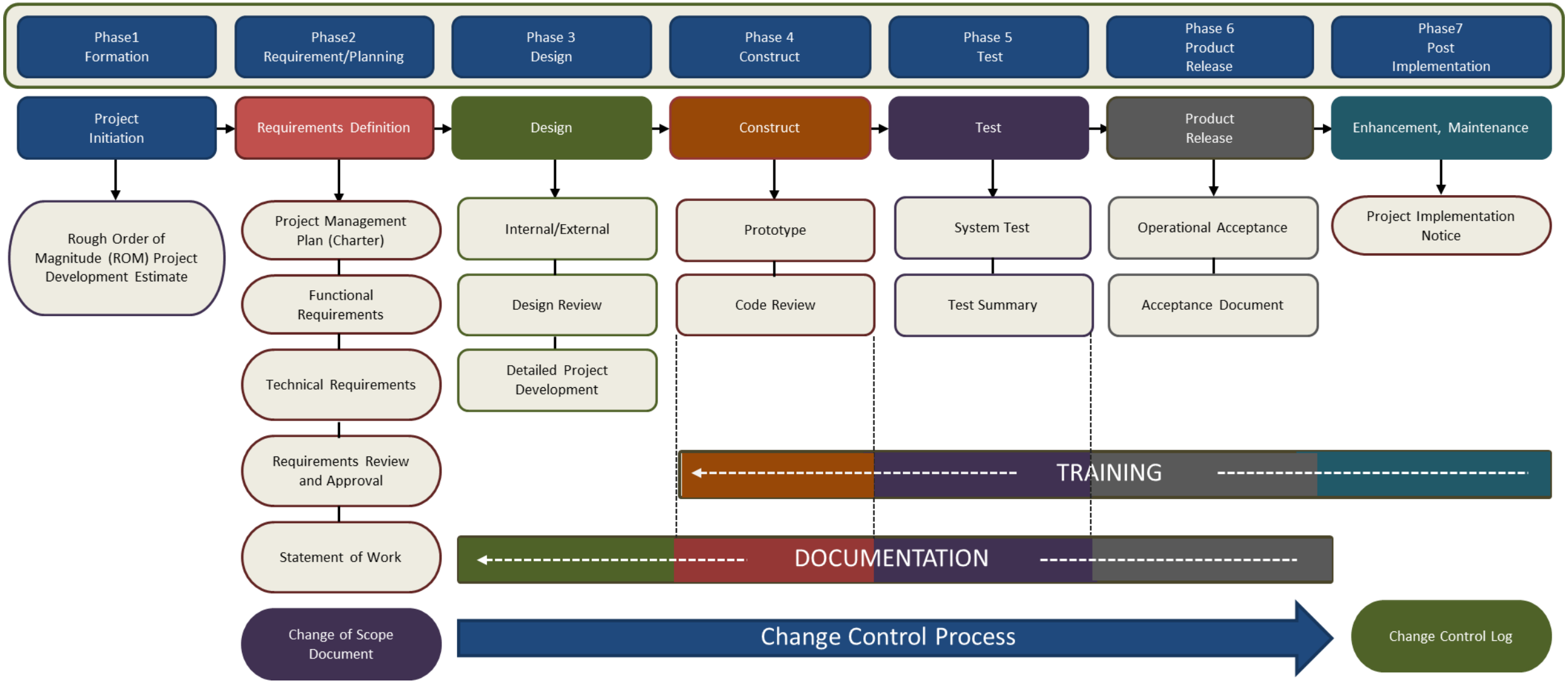
4.2 Describe the Secure Software Development Life Cycle (SDLC) Process

- Business Requirements
- Phases and Methodologies

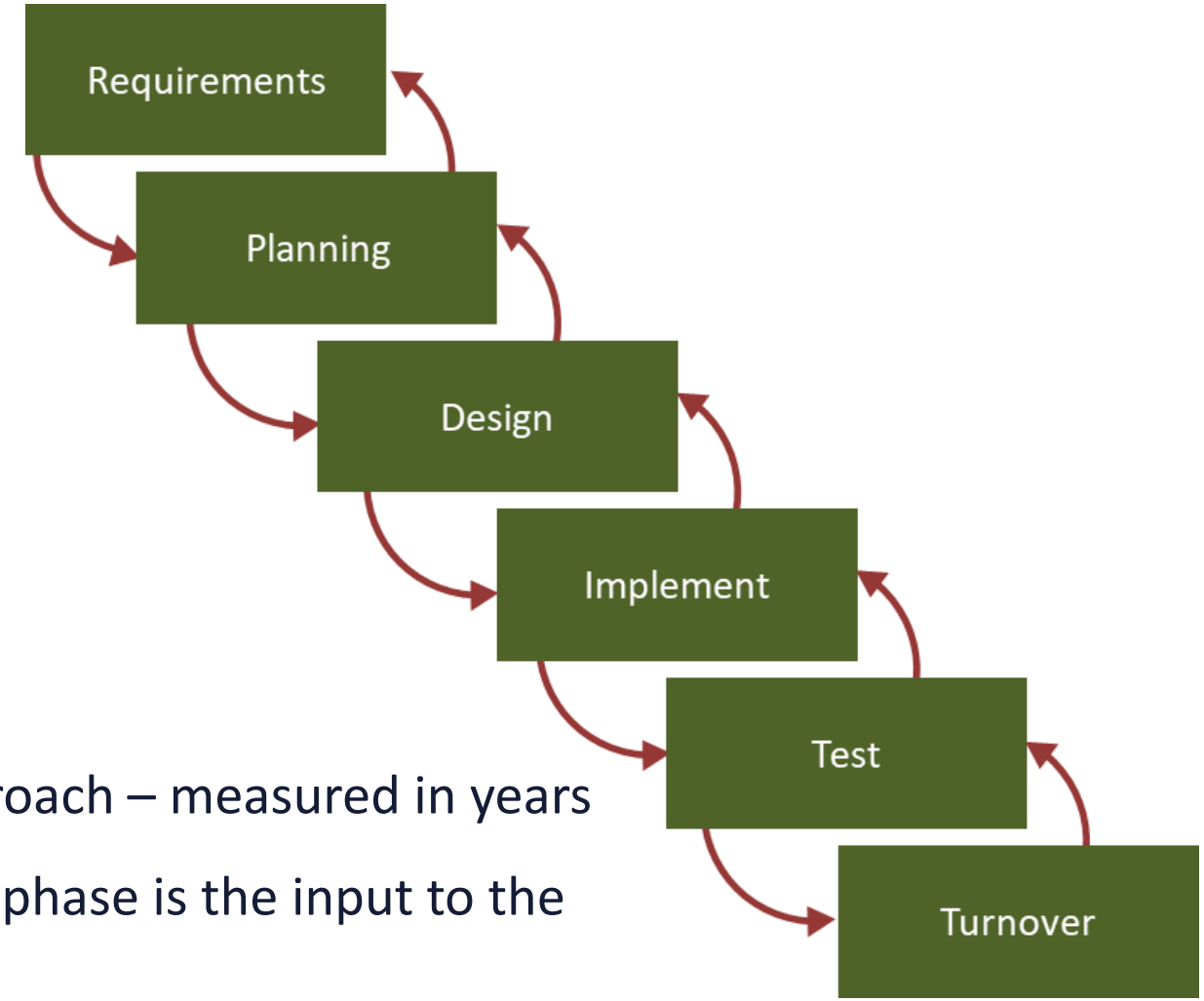
NIST Software/System Development Lifecycle



Enhanced SDLC

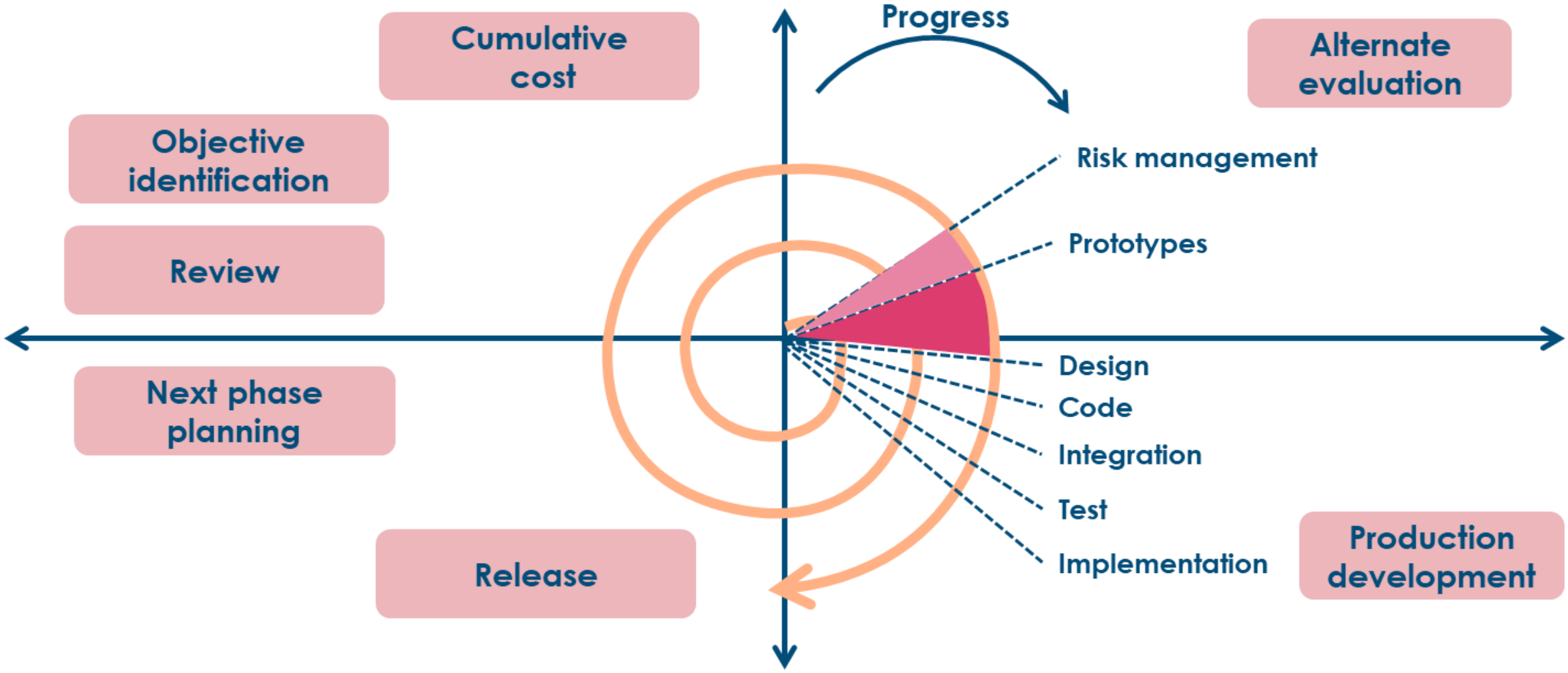


Waterfall Software Development



- Sequential approach – measured in years
- Output of each phase is the input to the next phase
- Little flexibility, not adaptable, very predictable, testing done in the end

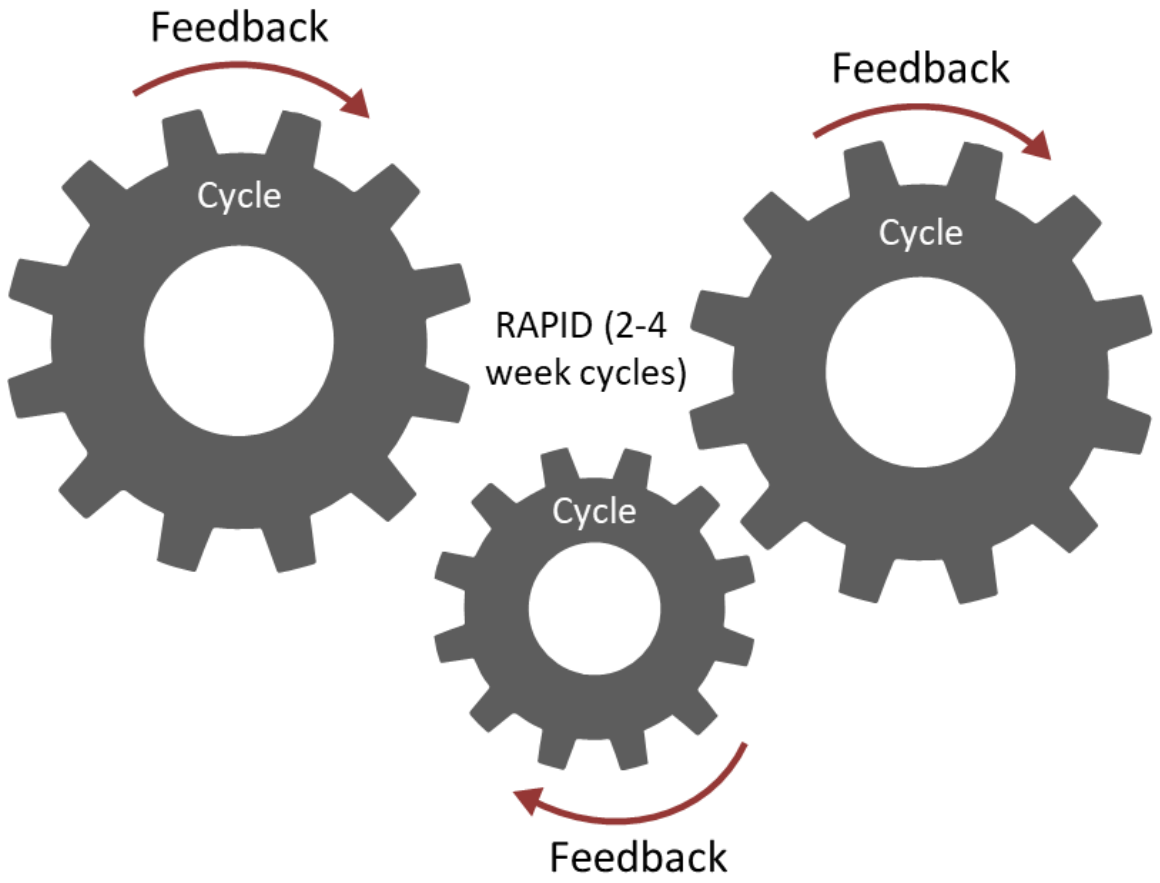
Spiral Development Model



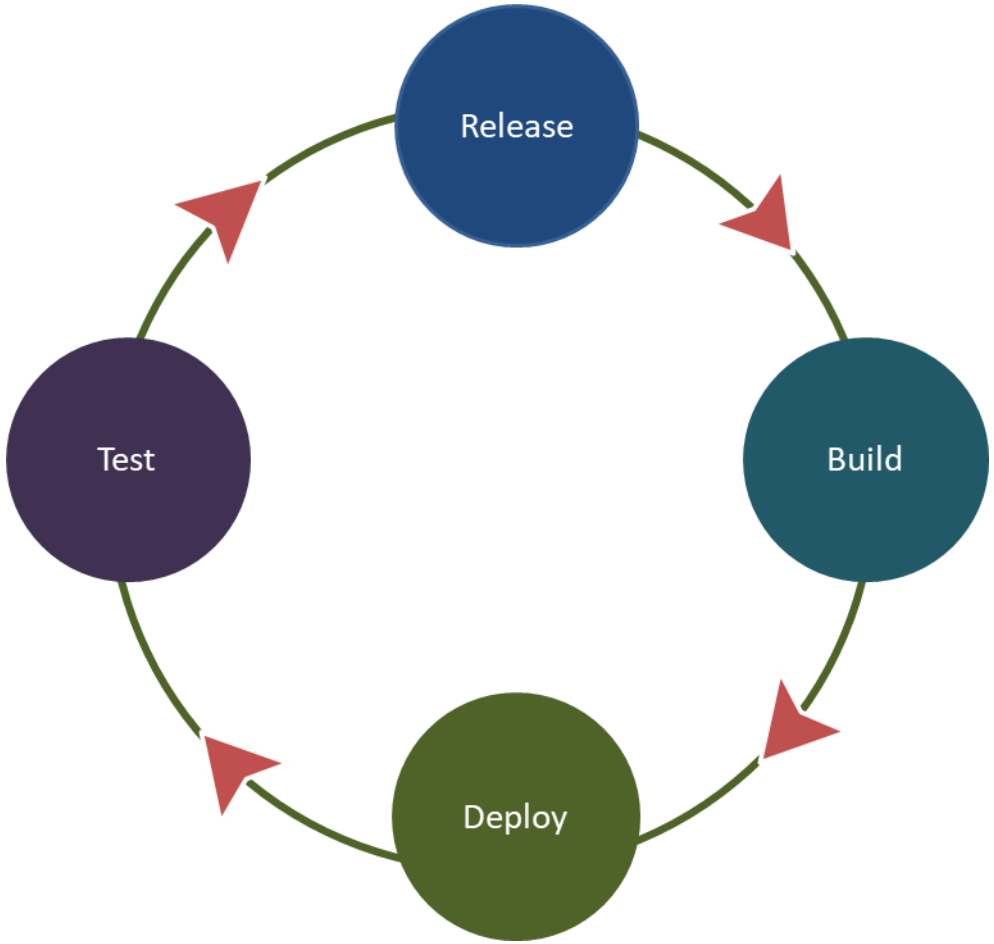
Agile Software Development

Excellent for smaller projects

- Evolutionary approach – measured in weeks
- Collaboration of cross-functional teams
- Very flexible, adaptable, not predictable, testing done during development
- Very high level of customer involvement throughout the project
- Works tightly with Agile Project Management



Continuous Integration (CI) – Continuous Deployment (CD) or CI/CD



- Continuous Integration (CI) is a development technique that forces developers to integrate code into a shared repository several times a day
- Each check-in is then verified by an automated build, allowing teams to detect problems early
- The goal is to detect and locate bugs and security flaws quickly
- Very popular method at AWS and GCP for developing traditional apps as well as containers and microservices

DevSecOps

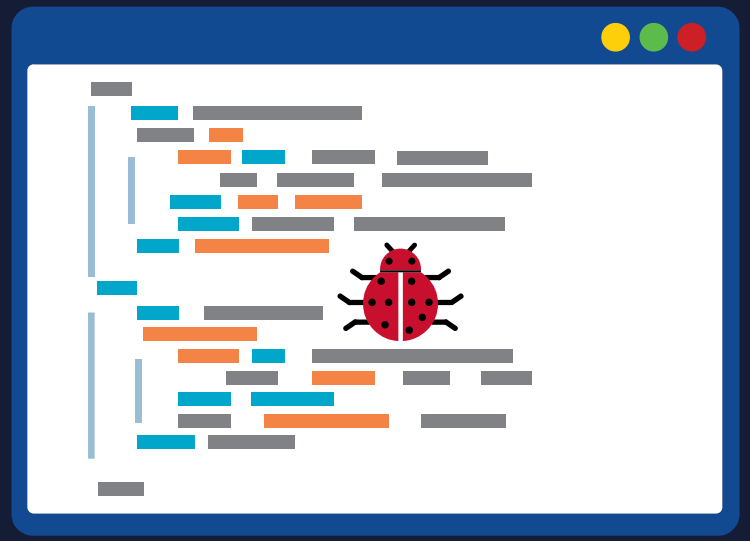
Development + Security + Operations

- It is a clipped compound referring to a set of practices that accentuate the collaboration and communication of both software developers and IT professionals that automate the software delivery process
- DevOps is a methodology for building software quickly by linking development and operations
- DevSecOps involves considering application and infrastructure security from the start and automating some security gates to keep the DevOps workflow from slowing down
- Choosing the right tools to integrate security continuously is critical

4.3 Apply the Secure Software Development Life Cycle (SDLC)

- Avoid Common Vulnerabilities During Development
- Cloud-specific Risks
- Quality Assurance
- Threat Modeling
- Software Configuration Management and Versioning

Source-code Weaknesses

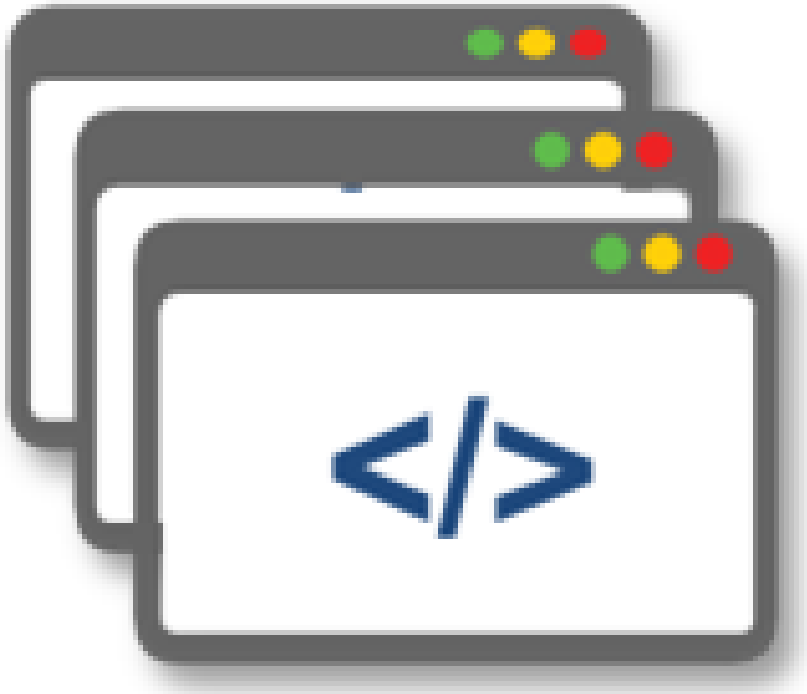


- Code vulnerabilities exist because solid secure development is quite difficult
- Experts often agree that open-source code libraries are far more secure than commercial software
- **Open-source code isn't inherently more secure; it is more securable**
- Most organizations do not have a clearly defined policy to confirm that developers wanting to use a section of software code go through an authorization process
- Since open-source components exist in almost all codebases, keeping up with open-source components in your software is an overwhelming task – including tracking the forks, versions, and state of updates to the code

.

Common Programming Weaknesses

- Poor error handling
- Poor exception handling
- Improper input validation
- Not relying on stored procedures
  - Precompiled groups of code, statements, and commands that can be called later
- Unsecure usage of code repositories
- Leaving inoperative dead code
- Redundancy in the code (no normalization)



Threat Modeling

S.T.R.I.D.E.

- Spoofing Tampering Repudiation Information Message Disclosure Denial of Service and Elevation of Privilege is a developer-focused threat modeling tool
- Microsoft threat modeling methodology that aligns with their Trustworthy Computing directive of January 2002
- Focus is to help ensure that Microsoft's Windows software developers think about security during the design phase
- Goal is to get an application to meet the security properties of CIA along with authentication, authorization, and non-repudiation
- Once the security SME builds the DFD-based threat model, system engineers or other experts check the application against the STRIDE threat model classification scheme

STRIDE

Threat	Definition	Property	Example
Spoofing	Pretending to be someone else	Authentication	Hack victim's email and to send messages as the victim
Tampering	Changing data or code	Integrity	Software executive file is tampered with by hackers
Repudiation	Claiming not to do a particular action	Non-repudiation	"I have not sent an email to users"
Information Disclosure	Leaking sensitive information	Confidentially	Making credit card information available on the internet
Denial of Service	Non-availability of service	Availability	Web application not responding to user requests
Elevation of privilege	Ability to perform unauthorized action	Authorization	Normal user can delete admin account

Admin. "STRIDE: Acronym of Threat Modeling System." All About Testing, February 21, 2019. https://allabouttesting.org/stride-acronym-of-threat-modeling-system/.

Comparing Threat Modeling Methods

	OCTAVE	Trike	P.A.S.T.A	Microsoft	VAST
Implement application security at design time	✓	✓	✓	✓	✓
Identify relevant mitigating controls	✓	✓	✓	✓	✓
Directly contributes to risk management	✓	✓	✓		✓
Prioritize threat mitigation efforts	✓	✓	✓		✓
Encourage collaboration among all stakeholders	✓	✓			✓
Outputs for stakeholders across the organization	✓				✓
Consistent repeatability		✓			✓
Automation of threat modeling process		✓			✓
Integrates into an Agile DevOps Environment					✓
Ability to scale across thousands of threat models					✓

"Threat Modeling Methodologies." ThreatModeler Software, Inc. Accessed June 7, 2021. https://threatmodeler.com/threat-modeling-methodologies-/.

Vulnerability and Penetration Testing

- Vulnerability assessment and penetration testing both add value to supporting security of applications and systems prior to going into production
- **Vulnerability scans and tests** attempt to identify and report on known vulnerabilities in a system
- Automated and manual scheduled scanning results should map to an established risk ledger (log) and risk ratings combined with potential exposures and countermeasures
- Most often, vulnerability assessments are performed as white box tests, where the tester knows the application and has complete knowledge of the running environment
- **Penetration testing** is a process used to collect information and actively expose vulnerabilities in an application or system by conducting actual exploits and red team attacks
- Penetration testing is often a black or gray box test, where the tester assumes the attacker role with little or no knowledge of the application and must discover any security issues

Penetration Testing Frameworks

- **SSAF** - framework provided by Open Information Systems Security Group (OISSG), a not-for-profit organization based in London
- **OSSTMM** - open-source security testing created by ISECOM (Institute for Security and Open Methodologies
- **OWASP** - popular methodology used widely by security professionals, create by a non-profit organization focused on advancing software security
- **PTES** - Penetration Testing Execution Standard (PTES) methodology was developed to cover the key parts of a penetration test
- **NIST** - National Institute of Standards and Technology (NIST) provides a manual that is best suited to improve the overall cybersecurity of an organization

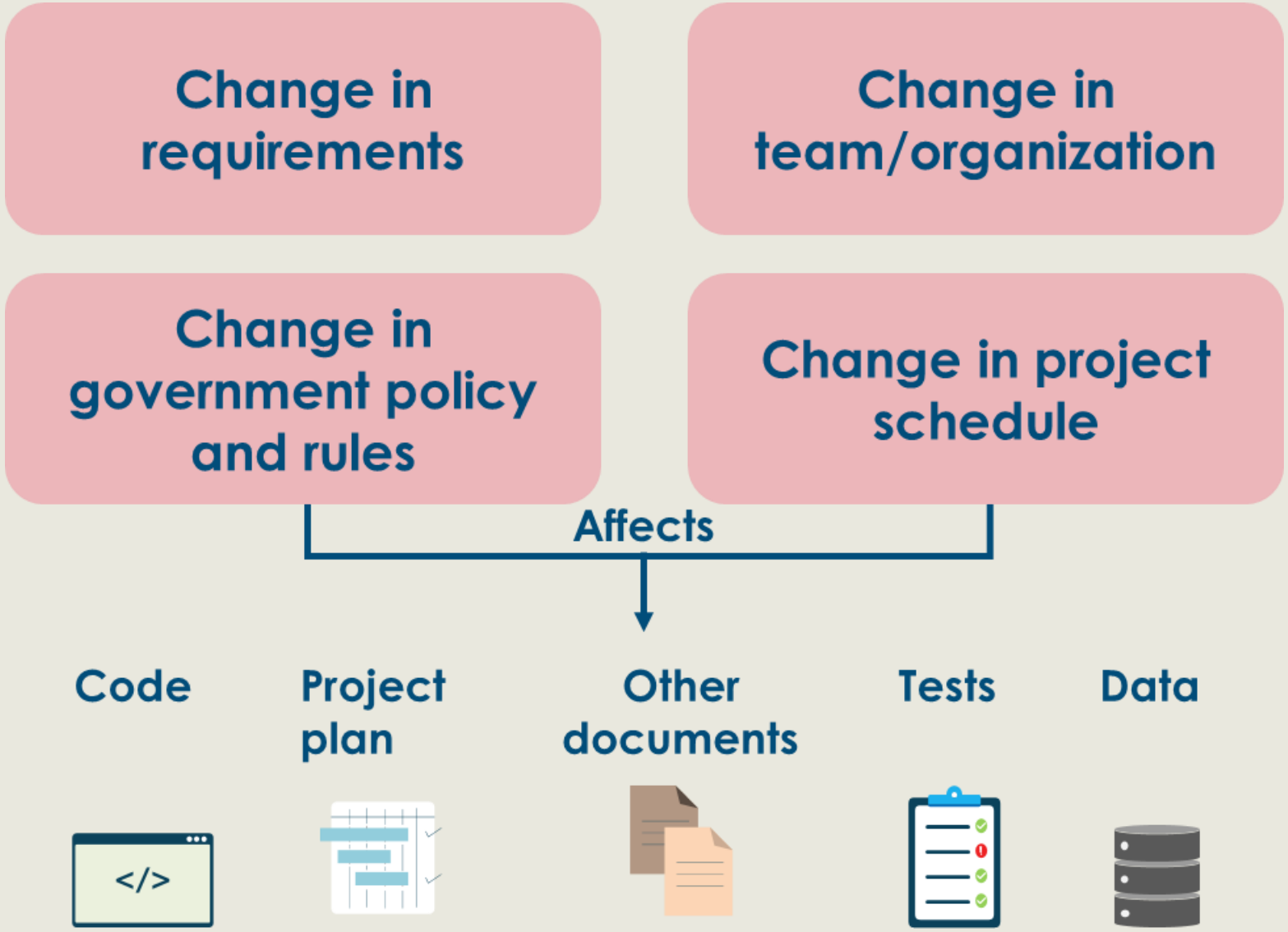
Software Configuration Management (SCM)

- Software configuration management (SCM) is a software engineering process to systematically manage, organize, and control the changes in the documents, codes, and other artifacts during the software development life cycle
- The primary goal is to enhance productivity and minimize errors
- SCM is part of the cross-disciplinary field of configuration management (integrated product teams – IPT) and can correctly determine the revision history

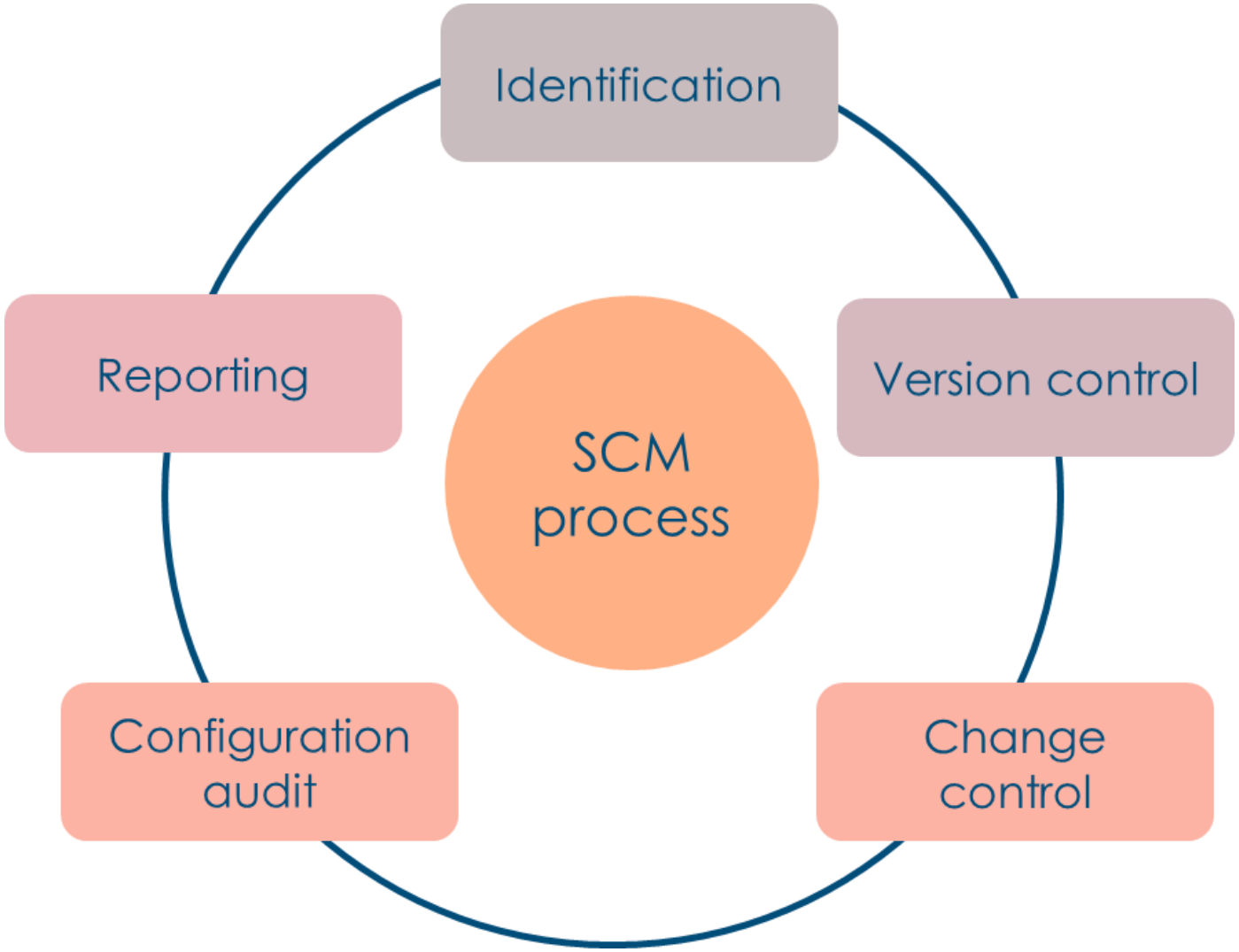
Reasons to Use SCM

- There are several people working on applications that are continually updating (CI/CD or Spiral development)
- Multiple versions, branches, micro-services, and programmers are involved in a software project, and the team is geographically dispersed yet is working concurrently
- Changes in customer requirements, policy, budget, and schedules need to be accommodated
- Software must be able to run on different platforms and operating systems
- There is a critical need to develop coordination among cross-functional stakeholders
- Need to control the costs involved in making changes to an app

Software Configuration Management (SCM)



Software Configuration Management (SCM)



Code Repository Security

- Your code is only as secure as the methods and systems used to generate it
- Some of the advantages that come with using a secure code repository are version control, peer review, and built-in auditing
- It is critical that the repository (such as GitHub or AWS) is an adequately secure central point of code storage and management
- Attackers can change a code base without your knowledge or permission due to loss/compromise of access credentials or breach of the core service
- If appropriate due diligence is applied to security measures, the benefits of using a code repository far outweigh the risks

Code Repository Security

- Select a repository you trust completely
- Consider the exposure and customer base of your repository
- Protect access credentials
- Separate secret credentials from source code
- Repository access should be revoked quickly when not needed or if compromised

Code Repository Security

- Include open code in your risk model
- Review all code changes
- Realize that external code changes may be malicious
- Protect your identity if using a publicly accessible repository
- Ensure that your code is backed up

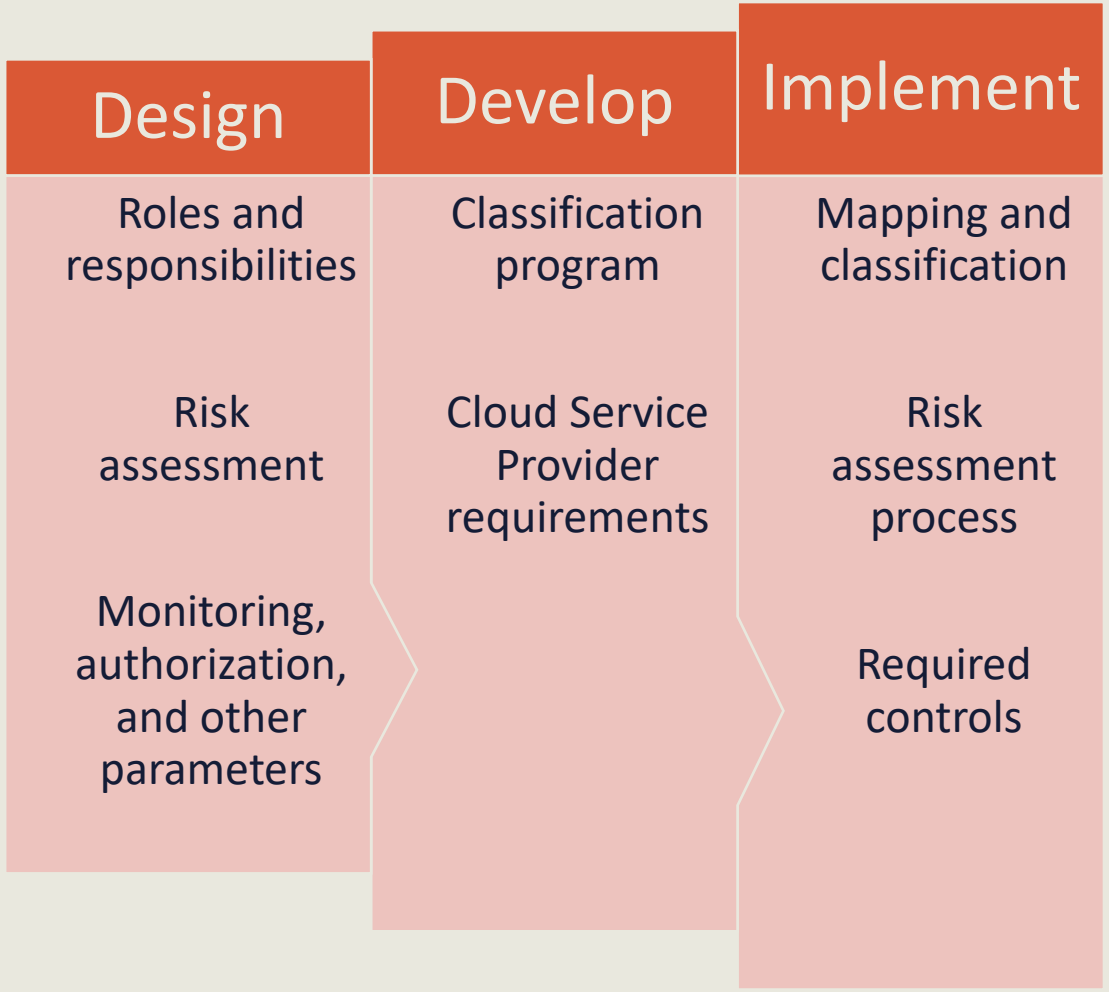
4.4 Apply Cloud Software Assurance and Validation

- Functional Testing
- Security Testing Methodologies

Software Assurance

- The key objective of the Software Assurance Program is to shift the security paradigm from patch management to software assurance
- Encourage developers to raise overall software quality and security from the start
- Emphasize the usage of tested standard libraries and modules
- Employ industry-accepted approaches that recognize that software security is fundamentally a software engineering issue that must be addressed systematically throughout the software development life cycle

Cloud Assurance Program Phases



Quality Assurance with SCA

Security Control Assessment (SCA)

- Security Control Assessment (SCA) is a formal evaluation of a system against a pre-defined set of controls
- Performed with, or independently of, a full **Security Test and Evaluation (ST&E)**, which is carried out as part of an official security authorization
- Often conducted as part of an official accreditation or certification process

Quality Assurance with SCA

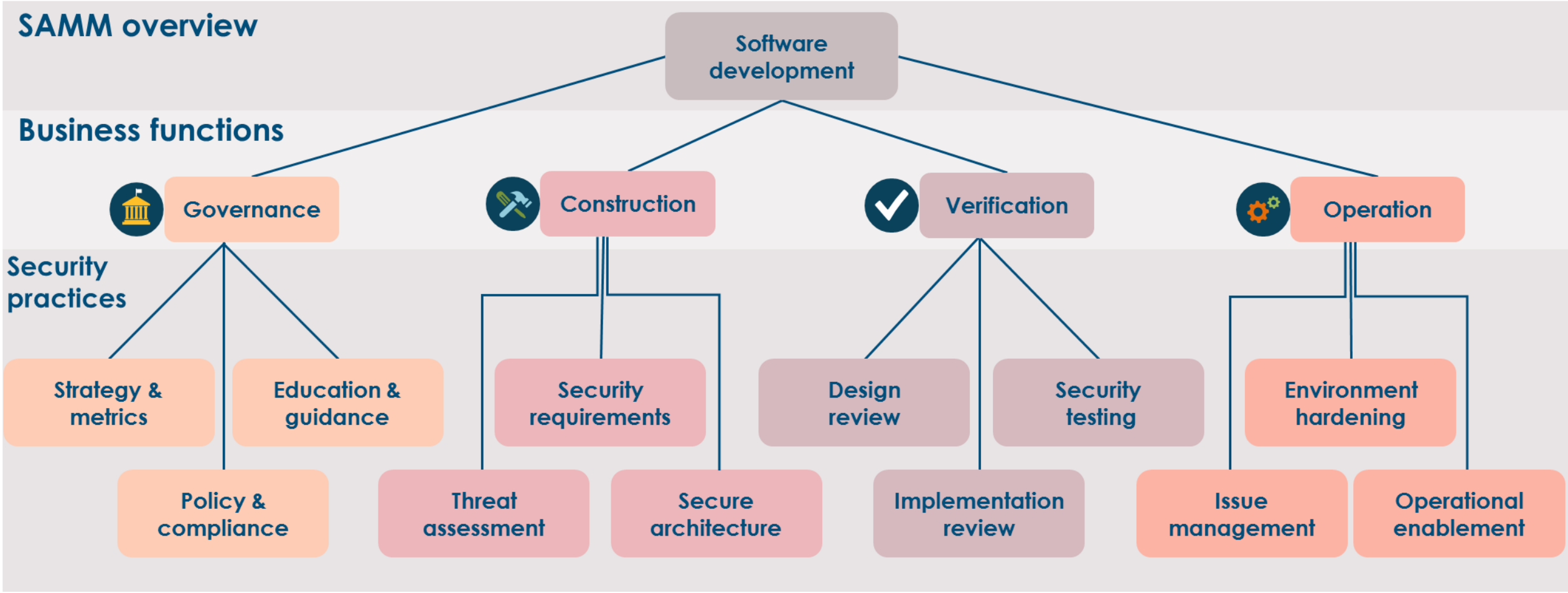
Security Test and Evaluation (ST&E)

- SCA and ST&E will appraise the operational plan (or planned implementation) of controls
- Results are a risk assessment report that represents a gap analysis, documenting the system, application, or data risk
- Tests conducted should include audits, security reviews, vulnerability scanning, and penetration testing

Software Assurance Maturity Model (SAMM)

- The Software Assurance Maturity Model (SAMM) is an open framework from OWASP to assist organizations in developing and deploying a secure software delivery strategy that is focused on the detailed risks facing the enterprise. The resources offered by SAMM will assist in:
  - Appraising the organization's current software security initiatives
  - Constructing a well-adjusted software security assurance program using established iterative processes
  - Establishing tangible continual improvement methodologies to a software security assurance program
  - Defining and gauging security-related tasks throughout the enterprise

Software Assurance Maturity Model (SAMM)



"Software Assurance Maturity Model." OWASP.org. Accessed June 8, 2021. https://owasp.org/www-pdf-archive/SAMM_Core_V1-5_FINAL.pdf.

Software Diversity

- Software diversity is an application development methodology where two or more functionally duplicate versions of the app are developed from the same specification
  - The process uses different developers or programming teams
- The goal is better error detection, improved consistency, and fewer programming errors
- Automated software diversity uses randomization to significantly increase the difficulty of exploiting the huge amounts of low-level code in existence
- Diversity-based defenses are motivated by the assumption that a single attack will fail against multiple targets with unique attack surfaces

SAST

Static Application Security Testing (SAST)

- SAST is commonly defined as a **white-box test**, where an analysis of the application source code, byte code, and binaries is carried out by the application test **without executing the code**
- It is used to find coding errors and omissions that are symptomatic of security vulnerabilities
- SAST is often used as a test method when the tool is under development - **earlier in the development lifecycle**
- It can be used to find SQL injection attacks, cross-site scripting errors, buffer overflows, unhandled error conditions, and probable back doors into the application

DAST

Dynamic Application Security Testing (DAST)

- **Due to the nature of SAST being a white-box test tool, SAST typically delivers more comprehensive results than those found using DAST**
- DAST is considered a black-box test, where the tool must find distinct execution paths in the application being analyzed
- Unlike SAST, which analyzes code that is not running, DAST is used against **applications in their running state**
- It is primarily considered effective when testing exposed HTTP and HTML interfaces of web applications
- Static and dynamic application tests work in concert to improve the reliability of applications being built and bought by organizations

RASP

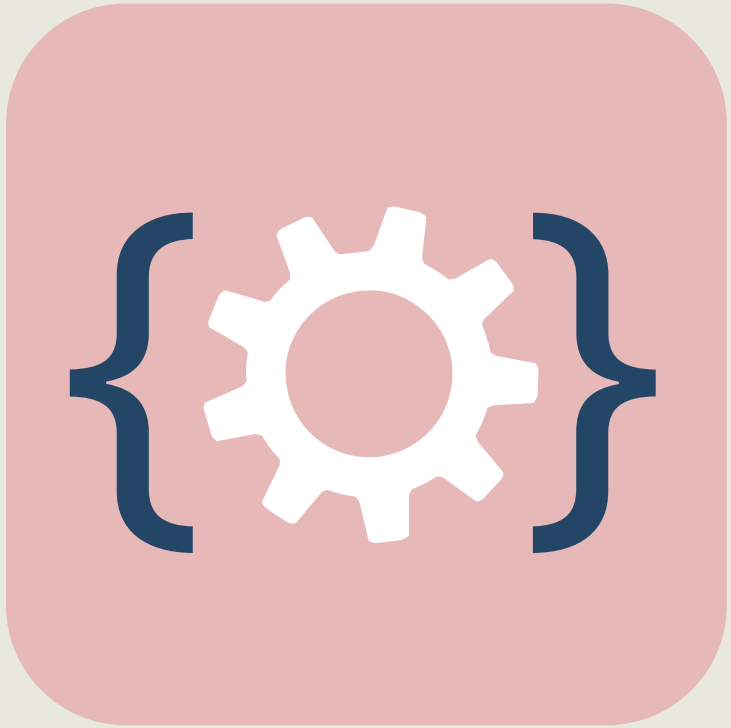
Runtime Application Self Protection (RASP)

- RASP is usually employed to focus on applications that have self-protection capabilities built into their runtime environments, which have full insight into application logic, configuration, and data and event flows
- RASP prevents attacks by “self-protecting” or reconfiguring automatically without human intervention in response to certain conditions (threats, faults, etc.)

4.5 Use Verified Secure Software

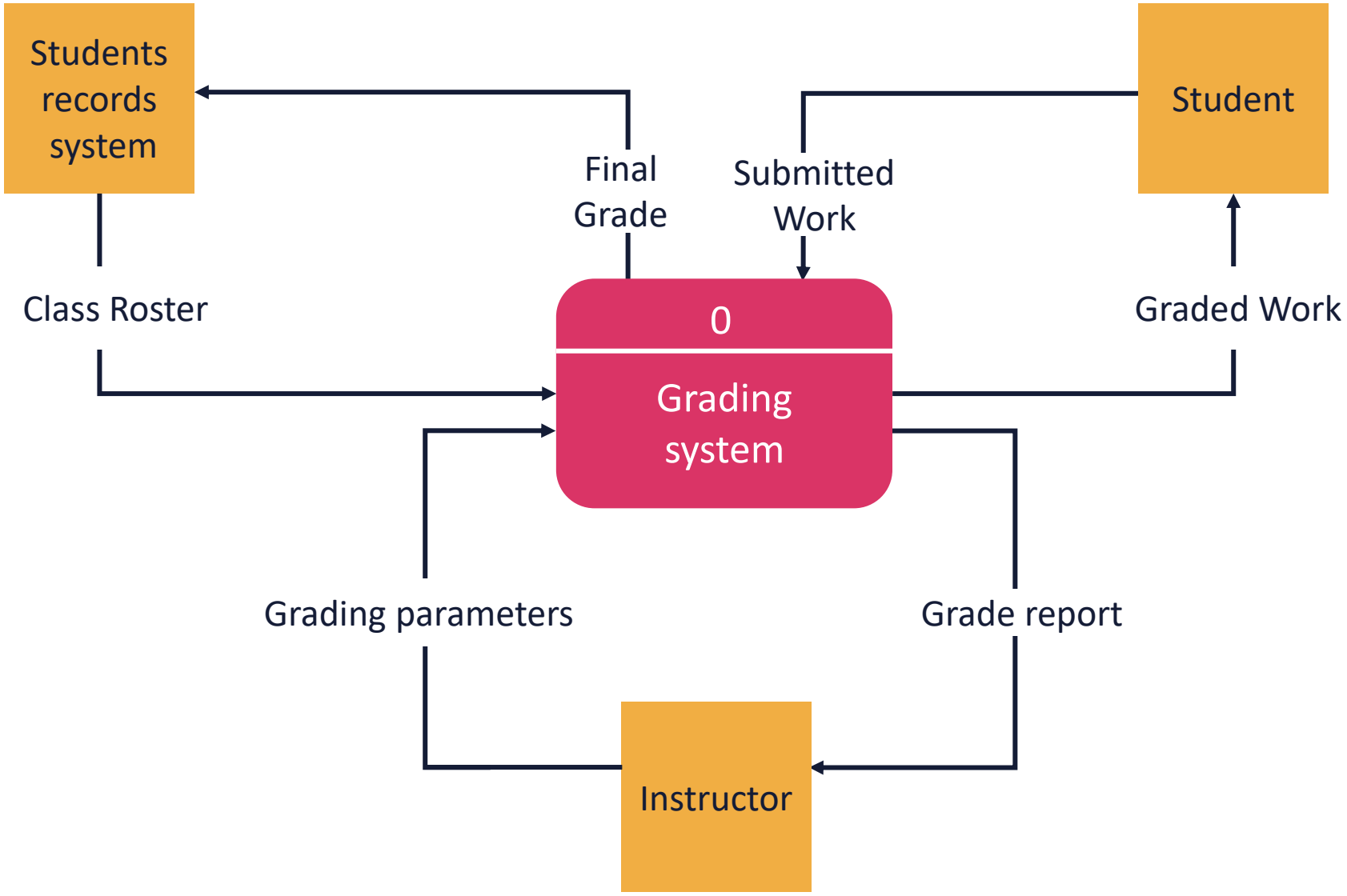
- Approved Application Programming Interfaces (API)
- Supply-chain Management
- Third Party Software Management
- Validated Open-Source Software

OWASP API Top Ten

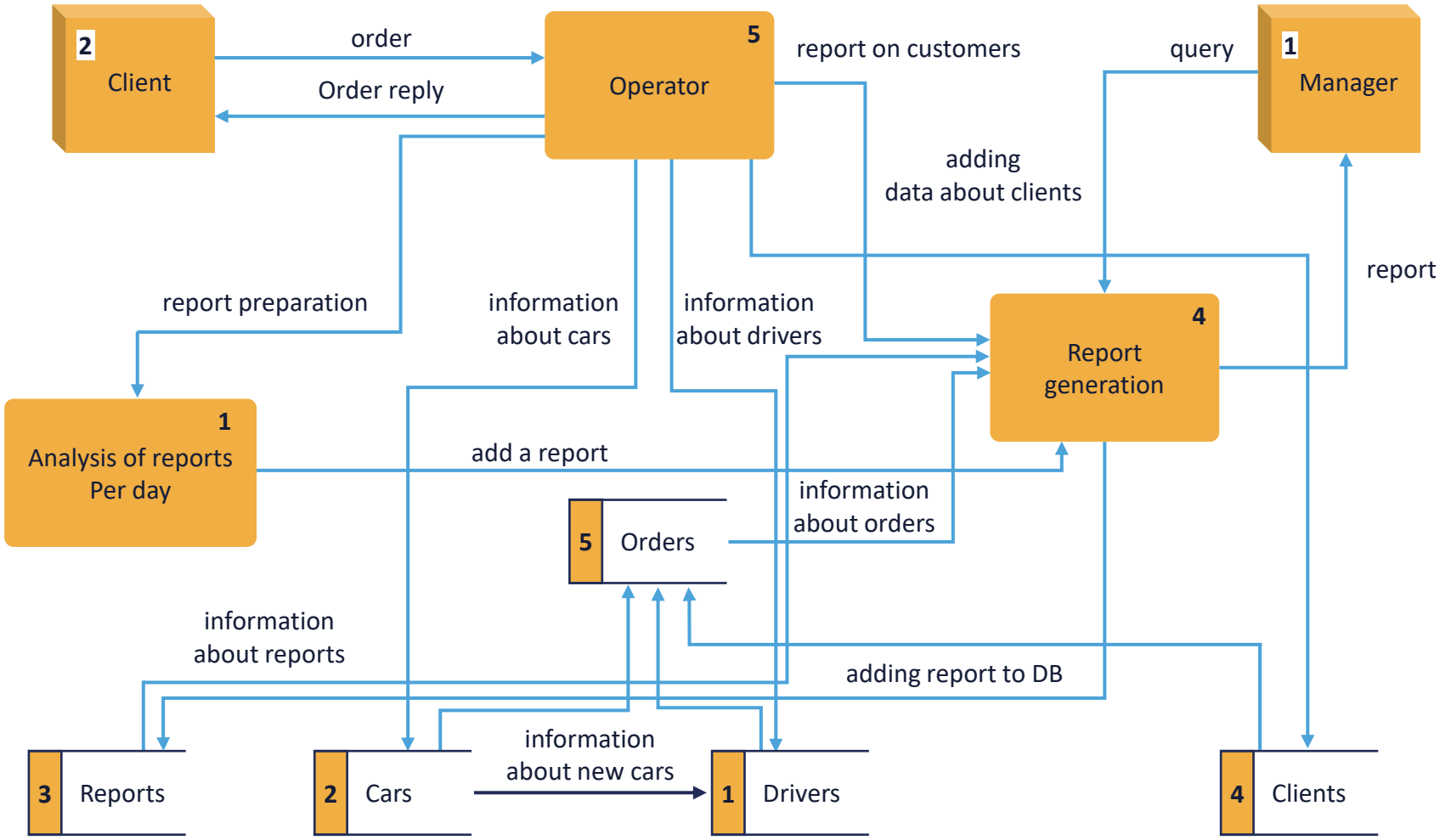


- API1:2019 Broken Object Level Authorization
- API2:2019 Broken User Authentication
- API3:2019 Excessive Data Exposure
- API4:2019 Lack of Resources & Rate Limiting
- API5:2019 Broken Function Level Authorization
- API6:2019 Mass Assignment
- API7:2019 Security Misconfiguration
- API8:2019 Injection
- API9:2019 Improper Assets Management
- API10:2019 Insufficient Logging & Monitoring

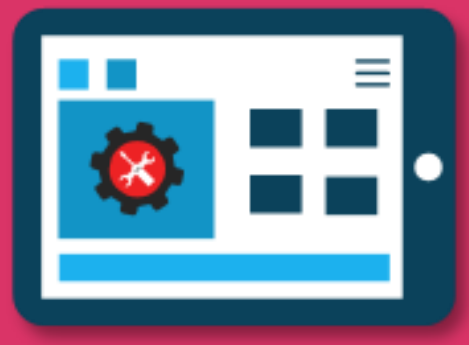
Sample Data Flow Diagram (DFD): basic



Sample Data Flow Diagram (DFD): complex



Commercial Off-the-shelf (COTS)



- A common commercial off-the-shelf as-is solution
- COTS products are intended to be easily installed and to interoperate tightly with existing system components
- Almost all software bought by the public computer user fits into the COTS category (operating systems, office product suites, word processing, and e-mail programs)
- One of the major advantages of mass-produced COTS software is that it is relatively low cost

Validate Open-Source Software

Secure Code Reviews

- Directing an informal or formal secure code review is another approach to assessing code for appropriate security controls
- An informal code review may involve a software engineer evaluating sections of code, looking for vulnerabilities
- A formal code review may involve the use of trained teams of reviewers that are assigned specific roles as part of the review process, as well as the use of a tracking system to report on vulnerabilities found
- The integration of a code review process into the SDLC can improve the quality and security of the code being used or developed

Open-Source Vulnerabilities

- Many enterprises and products (90% by some estimates) use at least one open-source component, often without being aware of it
- Normally, this software is built using public community collaboration and is preserved and updated on a voluntary basis
- Open-source software can be used according to a diversity of licenses, depending on what the developers have implemented
  - There are over 200 types of licenses that can be used with open-source software
- Lack of warranty for its security, support, or content
- No claims or obligations to be secure

Open-Source Vulnerabilities

- Open-source software often includes or demands the use of vulnerable third-party libraries which can involve intellectual property challenges
- Lax integrations oversight and control
  - Dev teams often have non-existent review processes for open-source components
- Operational inadequacies requiring additional work for proper DevSecOps
- Poor development practices and procedures
  - Risks increase as developers commonly copy and paste chunks of code from open-source software
  - Developers often transfer components through email or use poor repository security practices

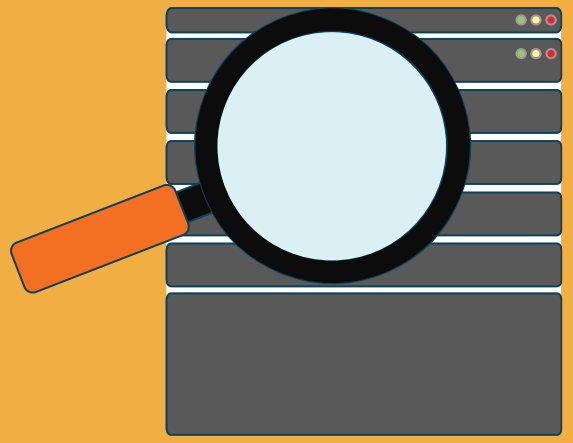
4.6 Comprehend the Specifics of Cloud Application Architecture

- Supplemental Security components (e.g., Web Application Firewall (WAF), Database Activity Monitoring (DAM), Extensible Markup Language (XML) firewalls, Application Programming Interface (API) gateway)
- Cryptography
- Sandboxing
- Application Virtualization and Orchestration

Web Application Firewalls (WAF)

- An appliance, server plugin, or CSP service that applies a set of rules to an HTTP/S connection
- The AWS WAF runs on an Application Elastic Load Balancer (with TLS Listener enabled), a CloudFront CDN distribution, or the API Gateway
- WebACLs filter for common attacks, such as:
  - Cross-site scripting (XSS)
  - SQL injection
  - Cross-site request forgery (CSRF)
  - Buffer overflows
  - DDoS and botnets
  - Custom WebACL rules

Database Activity Monitoring (DAM)



- A suite of cross-platform tools or enterprise services used to identify and report on fraudulent, illegal, or other undesirable behavior concerning data
- A DAM system should have minimal or no impact on user operations and productivity
- Modern solutions deploy a comprehensive toolkit for:
  - Visibility, discovery and classification
  - Vulnerability protection
  - Application-level analysis and intrusion prevention
  - Support for unstructured data security
  - Identity and access management integration
  - Risk management support

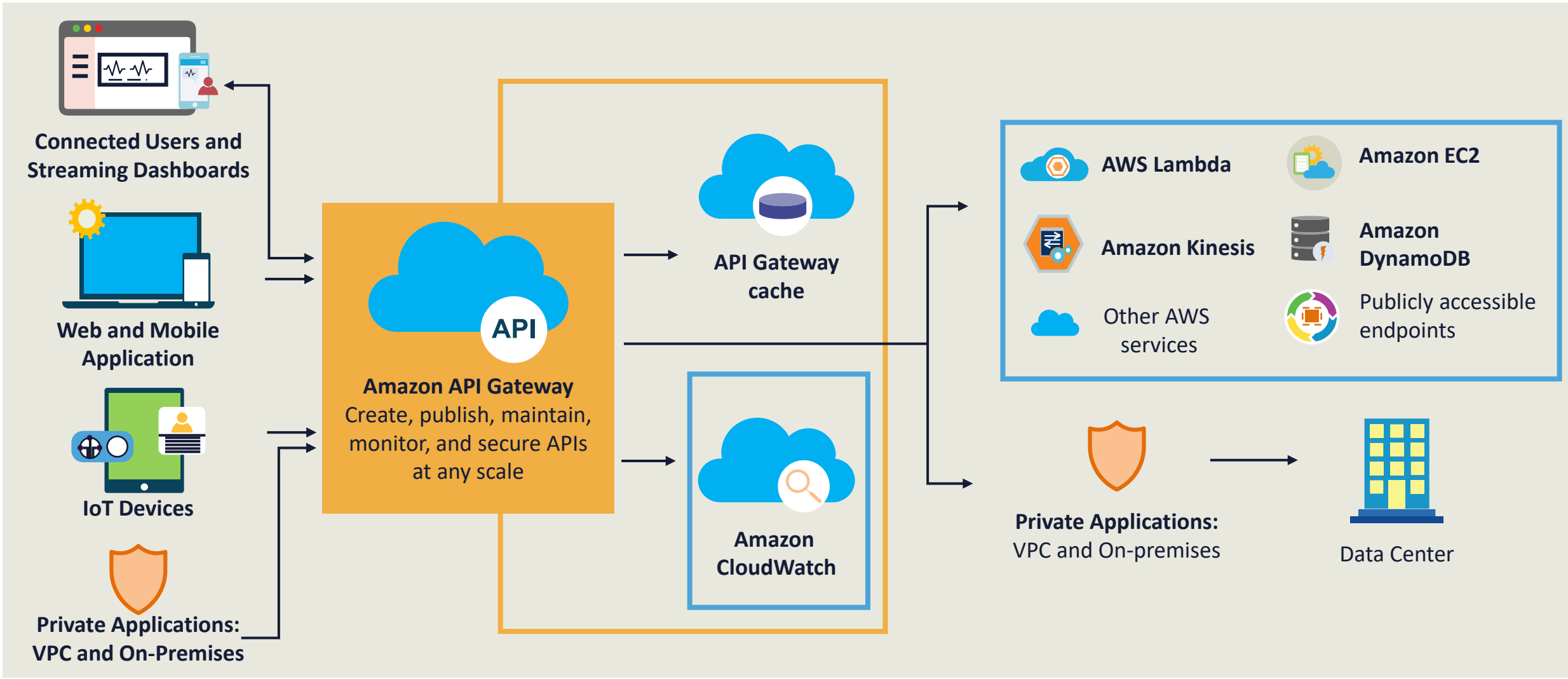
XML Firewalls

- An XML Firewall processes XML requests and responses over HTTP/S and contains a processing policy with a set of request, response, two-way, and error rules
  - Processing can include AAA, transformations, schema validation, logging, and cryptographic operations
- Through the processing policy, the XML Firewall can apply all processing actions to the request and response message, regardless of format
- Although an XML Firewall processes XML documents of all types, including SOAP-formatted messages, it can accept unprocessed (text or binary) documents

API Gateways

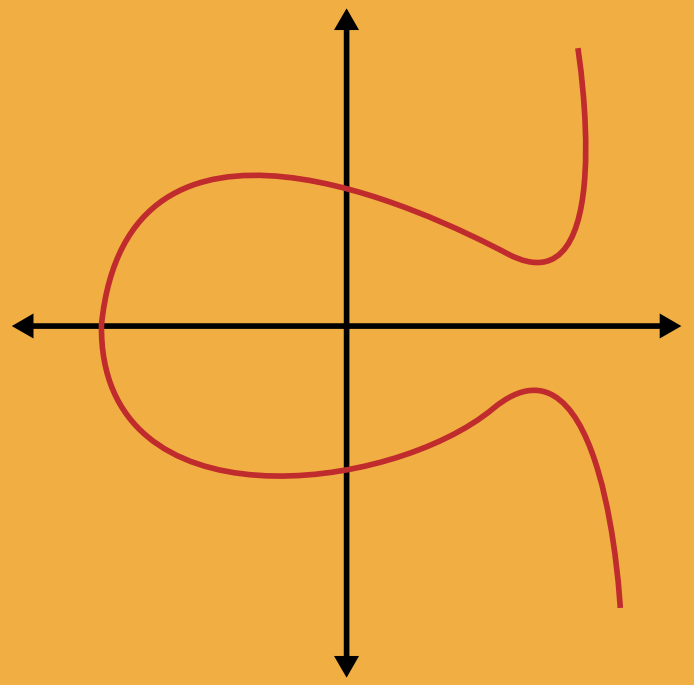
- An API Gateway is usually a fully managed cloud service that enables developers to create, publish, maintain, monitor, and secure APIs at any scale
- APIs act as the "front door" for applications to access data, business logic, or functionality from backend services
- AWS supports RESTful APIs and WebSocket APIs to enable real-time two-way communication applications
- API Gateways will now support containerized and serverless workloads, as well as web applications.

API GATEWAY AT AMAZON WEB SERVICES



Source: https://aws.amazon.com/api-gateway/

Cryptographic Best Practices



- Gravitate towards elliptic curve (ECDSA) and ephemeral protocol suites (IKEv2) when protecting data in transit
- Digitally sign all external API calls to cloud resources
- Never embed cryptographic keys or credentials in the API or in a code repository
- Always use the most recent TLS instead of SSL
- Consider DNSSEC and HSTS on web servers
- Key management must be compliant and CSPs offer secure options as well CloudHSM
- AES-GCM-256 is an Authenticated Encryptor Authenticated Decryptor (AEAD) that does not need a separate HMAC

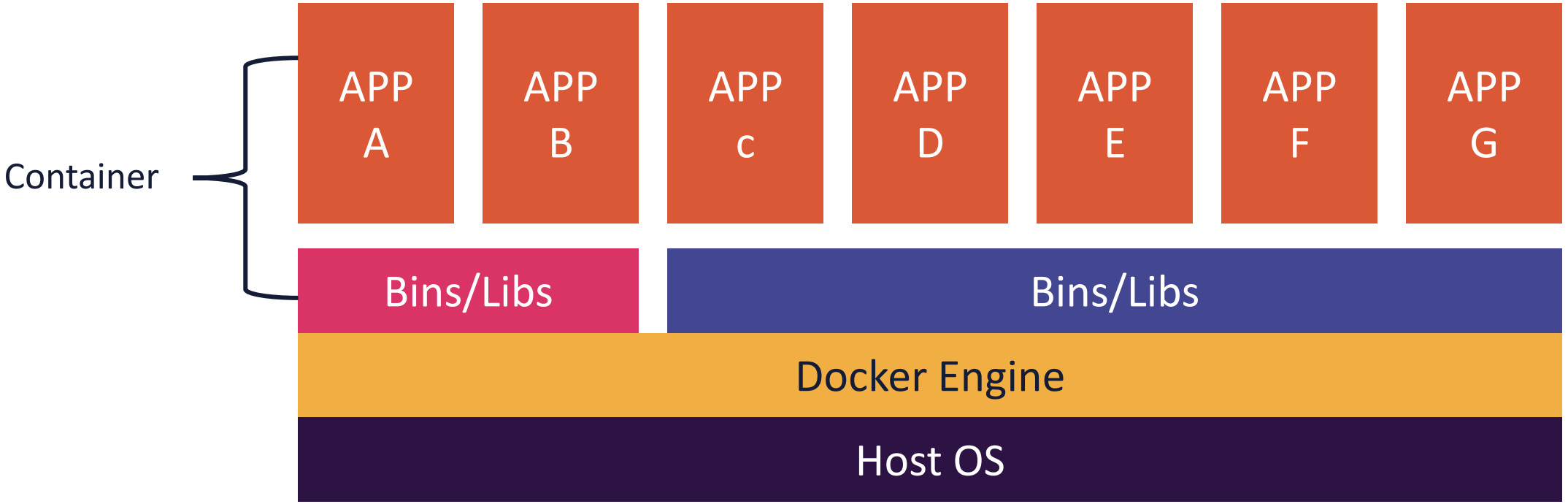
HTTP Strict-Transport-Security (HSTS)

- If a web site accepts an HTTP connection and redirects it to HTTPS, users may initially access the non-encrypted version of the site before being redirected
- This creates a vulnerability to man-in-the-middle attacks, as the redirect can be exploited to direct users to a malicious site instead of the secure version of the original site
- HTTP Strict-Transport-Security (HSTS) allows a TLS web site to instruct browsers that it should only be accessed using HTTPS, instead of using HTTP
- The web server employs the HTTP Strict-Transport-Security header

Containers

- Application container technologies, also known as containers, are a form of operating system virtualization combined with application software packaging
- Containers provide a portable, reusable, and automatable way to package and run applications
- DevSecOps should adapt the organization’s operational culture and technical processes to support the new way of developing, running, and supporting applications made possible by containers
- Use container-specific host OSs instead of general-purpose ones to reduce attack surfaces
- Adopt container-specific vulnerability management tools and processes for images to prevent compromises

APPLICATION CONTAINERS



4.7 Design Appropriate Identity and Access Management (IAM) Solutions

- Federated Identity
- Identity Providers
- Single Sign-On (SSO)
- Multi-factor Authentication
- Cloud Access Security Broker (CASB)

More about CASB

- One of the first companies to introduce a product labeled as a “CASB” was Sky High Networks acquired by McAfee in January 2018
- The Cloud Access Broker is also called a Cloud Access Gateway
- They are API-based (AWS PrivateLink partners) or Proxy-based (Palo Alto Aperture)
- The 4 Pillars of CASB are:
  - Visibility
  - Compliance
  - Data Security
  - Threat Protection

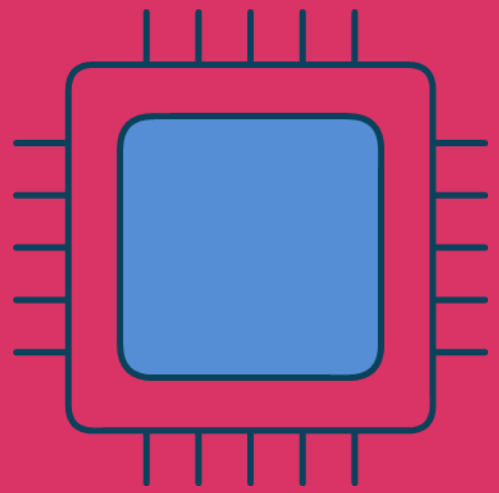
Domain 5

Cloud Security Operations

5.1 Implement and Build Physical and Logical Infrastructure for Cloud Environment

- Hardware Specific Security Configuration Requirements (e.g., Basic Input Output System (BIOS), settings for virtualization and Trusted Platform Module (TPM), storage controllers, network controllers)
- Installation and Configuration of Virtualization Management Tools
- Virtual Hardware Specific Security Configuration Requirements (e.g., network, storage, memory, CPU)
- Installation of Guest Operating System (OS) Virtualization Toolsets

Boot Integrity

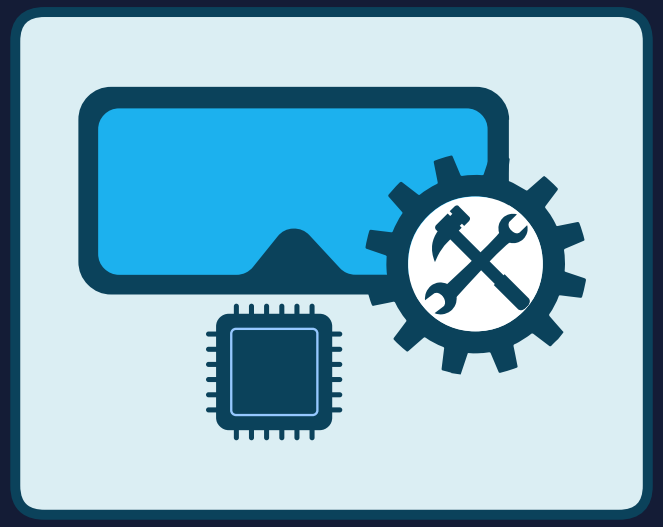


- UEFI – Unified Extensible Firmware Interface replaces legacy BIOS (basic input/output system)
  - Low-level software for booting the device
  - Tests the hardware components (POST)
  - Gets the OS up and running
- Offers the ability to protect the device at a lower level with passwords
- Restricts people from booting from removable devices
- Prevents users from changing UEFI settings without permission
- Prevents users from booting other OSs or installing over current OS

Boot Integrity

- Computer chip (microcontroller)
  - Installed on the device or built into PCs, tablets, and phones
  - Tamper-resistant security chip
  - Stores info needed to authenticate the platform
  - Passwords, certificates, and encryption keys
- Provides the following for the platform:
  - Integrity (ensures system has not been altered at a low level)
  - Authentication (ensures system is in fact the correct system)
  - Privacy (ensures system is protected from prying eyes)

Hardware Root of Trust



- Hardware root of trust
  - Anchoring the trustworthiness of a system to hardware not software
  - Hardware solutions are more secure than software solutions
  - Less susceptible to attacks since security solutions are on-chip
- Foundations of a Trusted Execution Environments (TEE) or Trusted Computing (TC):
  - TPM – module embedded in a system
  - SED – self-encrypting drives
  - HSM – dedicated crypto processor

Cloud Host Servers: Best Practices

- Secure build: To implement fully, follow the specific recommendations of the vendor to securely deploy their operating system
- Secure initial configuration: Depends on different variables, such as OS vendor, operating environment, business requirements, regulatory requirements, risk assessment, risk appetite, and hosted workload(s)
- Host hardening and patching: The more automated the better
- Host lock-down: Implement host-specific security measures including:
  - Blocking of non-root access to the host under most circumstances
  - Only allowing the use of secure communication protocols/tools
  - Configuration and use of host-based firewall
  - Use of role-based access controls
- Secure ongoing configuration maintenance: Achieved through the following:
  - Patch management of hosts, guest OSs, and application workloads
  - Periodic vulnerability assessment scanning
  - Periodic penetration testing of hosts and guest operating systems

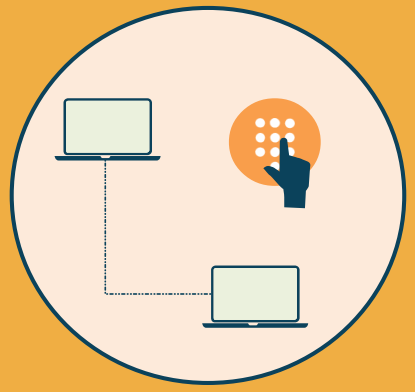
Storage and Network Controllers: Best Practices

- **Storage controllers may be in use for iSCSI, Fiber Channel (FC), or Fiber Channel over Ethernet (FCoE) – uses initiators and targets**
- Prevent oversubscription with a dedicated LAN for iSCSI traffic
- Do not share the storage network with other network traffic such as management, fault tolerance or vMotion/Live Migration
- Use encryption mechanisms such as IPsec AH or 802.11AE MACsec
- iSCSI can use several authentication mechanisms: Kerberos, Secure Remote Password (SRP), or SPKM1/2 (Simple Public-Key Mechanism)
- The key to virtual network security is isolation and compartmentalization using VXLAN, SDS, and PVLAN implementations
- When using internal and external networks, always create a separate isolated virtual switch with its own physical network interface cards and do not mix internal and external traffic on the virtual switches

5.2 Operate Physical and Logical Infrastructure for Cloud Environment

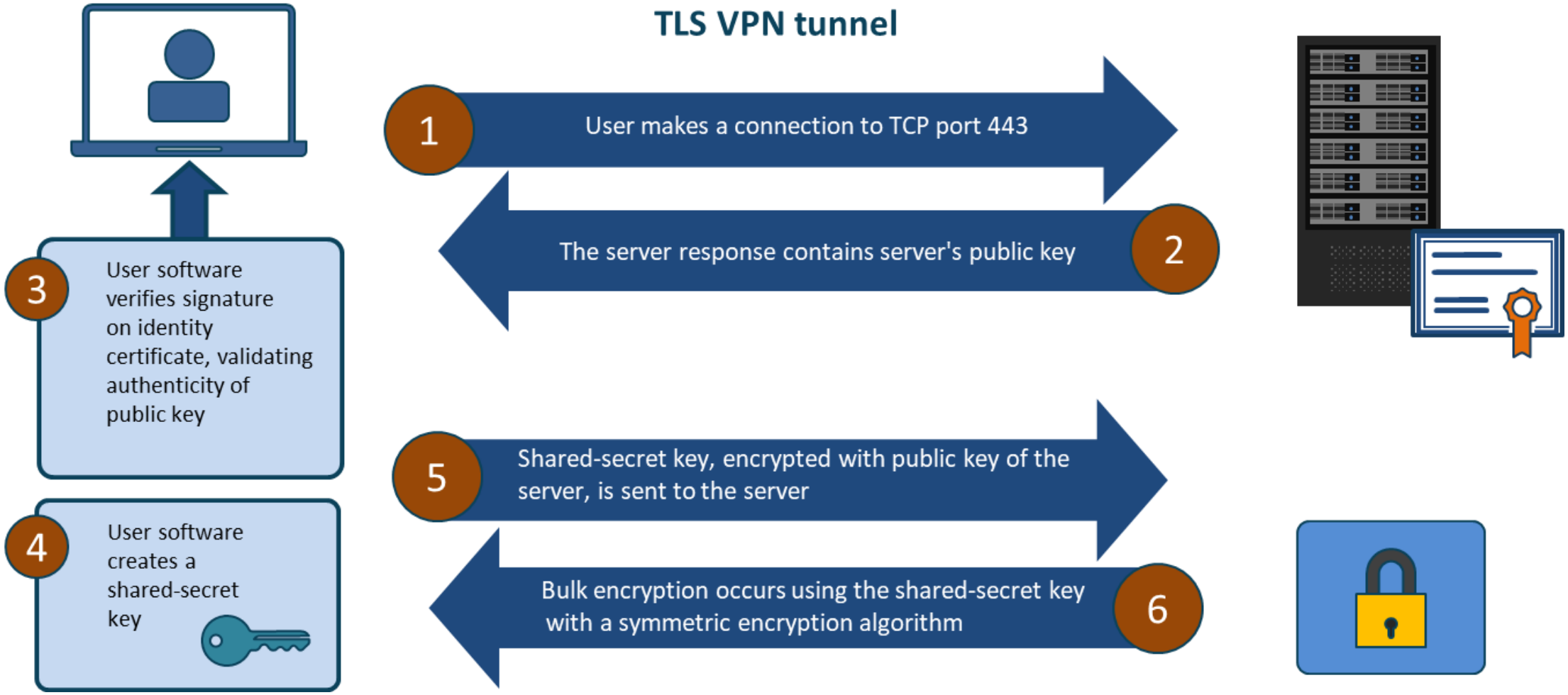
- Configure Access Control for Local and Remote Access (e.g., Secure KVM, console-based access mechanisms, RDP)
- Secure Network Configuration (e.g., VLAN, TLS, DHCP, DNS, VPN)
- OS Hardening Through the Application of Baselines (e.g., Windows, Linux, VMware)
- Availability of Stand-Alone Hosts
- Availability of Clustered Hosts (e.g., Distributed Resource Scheduling (DRS), Dynamic Optimization (DO), storage clusters, maintenance mode, High Availability)
- Availability of Guest Operating System (OS)

Secure KVM



- Access to hosts should be done by secure KVM and access to KVM devices should require a checkout (service desk) process
- Secure KVM will prevent data leakage from the server to the connected computer, as well as preventing unsecure emanations
- According to the Common Criteria, a secure KVM will do the following:
  - Isolated data channels
  - Tamper-warning labels on each side of the KVM
  - Housing intrusion detection
  - Fixed firmware
  - Tamper-proof circuit board
  - Safe buffer design
  - Selective USB access
  - Push-button control

Transport Layer Security (TLS)



TLS Best Practices

- Prevent downgrade attacks from web clients
- Use HTTP Strict Transport Security (HSTS)
- Use the most recent security suites (no RC4 or DES/3DES)
- Do not let vendor-installed code intercept traffic
- Verify encryption
- Perform OCSP stapling from browsers to enforce certificate expirations
- Implement Certificate Pinning to trusted CAs

Distributed Resource Scheduling (DRS)

- DRS continually monitors your cluster utilization to assure that VMs get their resources in the most optimal fashion
- DRS is responsible for keeping the cluster balanced, so the utilization of the ESXi hosts is equal
- DRS also works closely with resource management that guarantee specific resources for your VMs
- DRS constantly monitors a lot of parameters for the peak performance and placement:
  - Host resource capacity
  - Resource reservations
  - Datastore connectivity
  - Actual resource demand from virtual machines
  - Reservations, Shares and Limits (R, L, S)

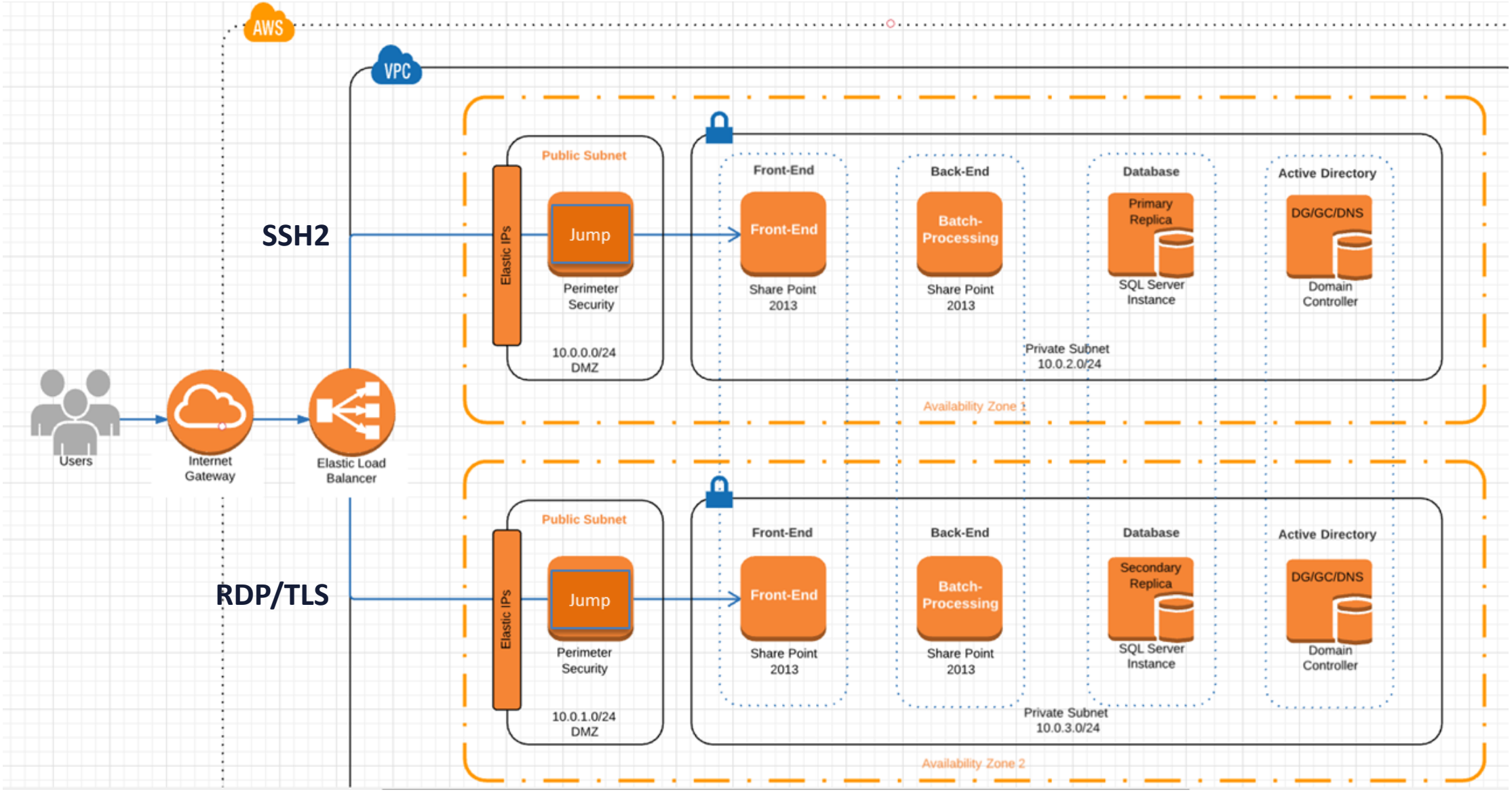
Dynamic Optimization (DO)

- Dynamic optimization facilitates live migration of VMs and VHDs within a host cluster
- The migration is based on the settings you designate, to improve load balancing among hosts and cluster shared storage, and to correct the placement issues for VMs
  - **Compute Dynamic optimization** is the optimization of hosts in a cluster to optimize performance by migrating VMs across host
  - Host performance thresholds you can set are CPU and Memory
  - **Storage Dynamic Optimization** is optimization of disk space and is performed on cluster shared storage (CSV, file shares) to optimize storage space availability by migrating Virtual Hard Disks (VHD) across shared storage
  - You can set free storage space threshold on cluster shared storage

5.3 Manage Physical and Logical Infrastructure for Cloud Environment

- Access Controls for Remote Access (e.g., RDP and SSH)
- OS Baseline Compliance Monitoring and Remediation
- Patch Management
- Performance and Capacity Monitoring (e.g., network, compute, storage, response time)
- Hardware Monitoring (e.g., Disk, CPU, fan speed, temperature)

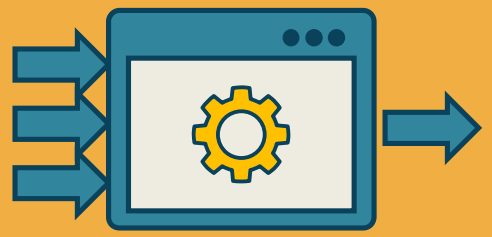
Using Bastion (Jump) Hosts



AWS Systems Manager

- Systems Manager enables you to manage servers running on AWS and in your on-premises data center through a single interface
- It securely communicates with a lightweight agent installed on your servers to execute management tasks
- This helps you manage resources for Windows and Linux operating systems running on Amazon EC2 or on-premises
- You can easily and securely access your Amazon EC2 instances through an interactive one-click browser-based shell or through the AWS CLI without having to open inbound ports, maintain bastion hosts, or manage SSH keys

AWS AppStream

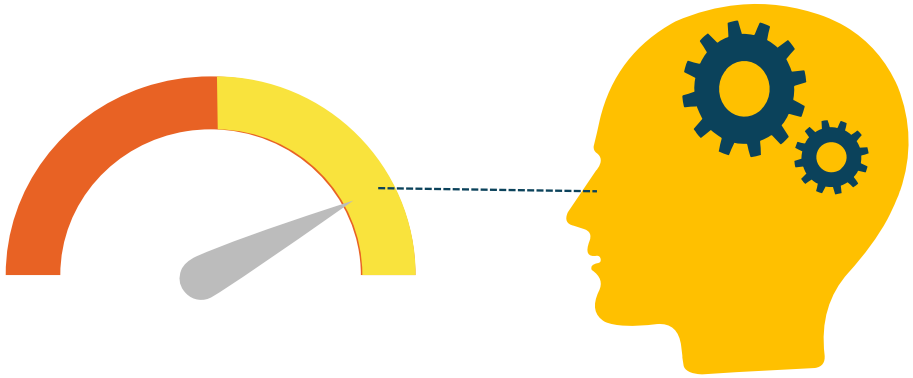


- An SSO dynamic bastion solution to allow your administrators to manage their environment without giving them a full (possibly unsecure) bastion/jump host
- AppStream spins-up fresh instances each time a user requests access, a compromised instance will only last for the duration of a user session
- As soon as the session closes and the Disconnect Timeout period is reached, AppStream terminates the instance
- You will also potentially reduce your costs because AppStream 2.0 has built-in auto-scaling to increase and decrease capacity based on user demand

Benchmarks

CIS, CSA, ISO/IEC, DoD

- A technique to improve an organization's information security management by establishing a standard
- CIS Benchmarks™ are best practices to securely configure various systems and are available for more than 140 technologies
- Established using a special method constructed from an accord of global cybersecurity experts from across the globe
- CIS Benchmarks™ are security configuration guides created by government, business, industry, and academia



Google Cloud Operations Suite for Performance and Capacity Monitoring

- Collect metrics, logs, and traces across Google Cloud and your applications
- Use built-in out-of-the-box dashboards and views to monitor the platform and applications
- Perform advanced queries and analyses on your data
- Set up appropriate performance and availability indicators
- Set up alerts and notification rules with your existing systems
- **Can add AWS resources to Operations Workspace!**

5.3 Manage Physical and Logical Infrastructure for Cloud Environment (cont.)

- Configuration of Host and Guest O/S Backup and Restore Functions
- Network Security Controls (e.g., firewalls, Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), honeypots, vulnerability assessments, network security groups)
- Management Plane (e.g., scheduling, orchestration, maintenance)

Mobile Devices and IoT with the Cloud

- Cloud-based services will result in many mobile IT users accessing business data and services without traversing the corporate network
- This will increase the need for enterprises to place security controls between mobile users and cloud-based services
- Placing large amounts of sensitive data in a globally accessible cloud leaves organizations open to large, distributed threats
- Attackers no longer have to come onto the premises to steal data, and they can find it all in the one “virtual” location

Enterprise Mobility Management (EMM)

MDM + MAM

- Organizations must securely configure and implement each layer of the technology stack, including mobile hardware, firmware, O/S, management agent, and the apps used for business
- Solution should reduce risk, so employees are able to access the necessary data from nearly any location, over any network, using a wide variety of mobile devices
- Enterprise mobility management is the combination of mobile device management (MDM) and mobile application management (MAM)

Enterprise Mobility Management (EMM)

MDM + MAM

- There are three basic core competencies that all organizations need from an EMM solution:
  - Visibility: understanding what's running on mobile devices is the key to discovering potential risks and adhering to compliance policies
  - Secure access: providing the ability for mobile users to securely authenticate and authorize access to apps and data
  - Data protection: offering dynamic anti-malware and data loss prevention capabilities to help limit the risk of attacks and data breaches

5.4 Implement Operational Controls and Standards (e.g., ITIL 4, ISO/IEC 20000-1)

- Change Management
- Continuity Management
- Information Security Management
- Continual Service Improvement Management
- Incident Management
- Problem Management

5.4 Implement Operational Controls and Standards (cont.)

- Release Management
- Deployment Management
- Configuration Management
- Service level Management
- Availability Management
- Capacity Management

5.5 Support Digital Forensics

- Forensic Data Collection Methodologies
- Evidence Management
- Collect, Acquire and Preserve Digital Evidence

Why Perform Forensic Investigations?

- Laws have been violated
- Organizational policies have been violated
- Systems have been attacked
- Data and identity have been breached
- Intellectual property has been exfiltrated
- Privileged insiders are suspected of crimes
- Next phase of incident response

Forensic Lifecycle

1. Identification

Detecting the incident

- Once you have determined this is not an event but rather an incident that needs forensics, you will need to identify and classify
- Forensics may not occur until later in the incident response lifecycle (remediation or reporting)
- The notification can come from
  - personal complaint by phone or text
  - monitoring system alarm or alert
  - audit result
  - IDS/IPS/EDR sensor alarm, or
  - notification from trusted or anonymous source

2. Collection

Order of volatility sets the priority

1. CPU, cache, and register content
2. Routing table, ARP cache, process table, and kernel statistics
3. Memory
4. Temporary file system/swap space
5. Data on hard disk
6. Remotely logged data
7. Data on archival media

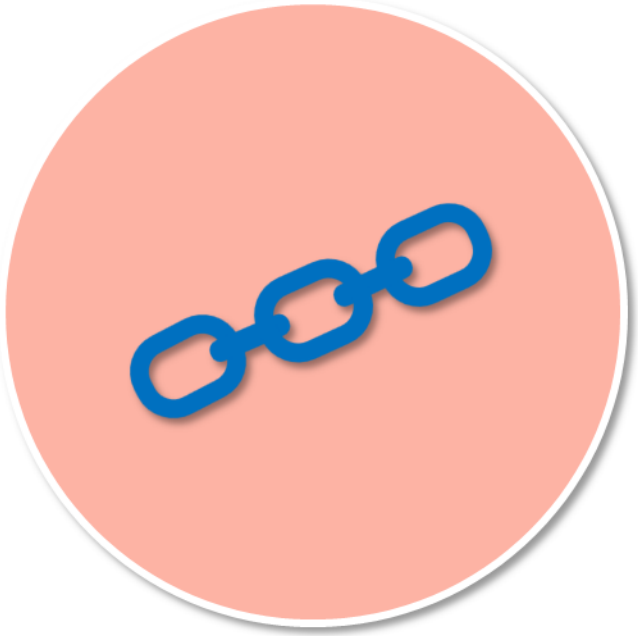
- Forensic toolkits (EnCase from Guidance) and write-blockers

Common utilities include:

- Tcpdump and dd
- nbtstat and netstat
- nc (Netcat)
- memcopy
- tshark
- foremost

2. Collection

Maintain the chain of custody



- Imaging technologies (create copies)
  - Memory dumps from write blockers
  - HDD bit-level copy, sector-by-sector
  - Include deleted files, slack spaces, and unallocated clusters
  - Look for encrypted volumes and files
- Digital pictures and interviews
- Provide a history of the handling of the evidence to maintain integrity, provide accountability, prohibit tampering, and provide assurance through the entire life cycle

Chain of Custody Documentation

Chain of custody				
Registered mail	Date/Time	Released by	Received by	Reason
	Date	Name/Agency/Organization	Name/Agency/Organization	
	Time	Signature	Signature	
	Date	Name/Agency/Organization	Name/Agency/Organization	
	Time	Signature	Signature	
	Date	Name/Agency/Organization	Name/Agency/Organization	
	Time	Signature	Signature	
	Date	Name/Agency/Organization	Name/Agency/Organization	
	Time	Signature	Signature	

2. Collection

Media management

- Should have a software inventory system for configuration items and a category for components removed for investigative and forensic purposes
- Collected media (all types) must be classified and labelled
- Secure storage facilities with dual operators include
  - locked rooms
  - locked cabinets
  - safes, and
  - offsite storage facilities

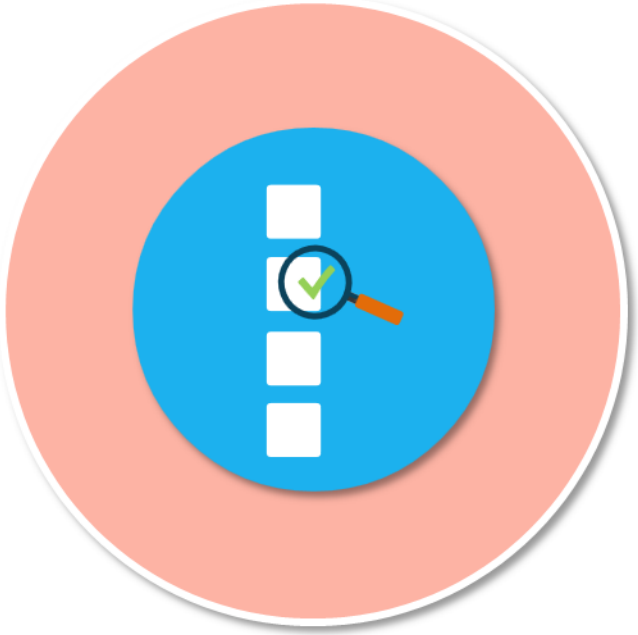
3. Examination

Where data becomes information

- Examination involves finding the relevant data in order to have the proper information to analyze in the next step
- Use tested techniques for
  - validation
  - filtering (i.e., user SIDs)
  - pattern matching
  - tracing
  - hidden data discovery, and
  - data extraction

4. Analysis

Building a solid forensic case



- Operating on the relevant data, facts, and artifacts from collection and examination phases
- May determine that more work should be done in earlier steps
    - If more information is needed, then iterate back to collection and examination
- Answer the 'who, what, where, when, why, and how?'
- Infer motive, opportunity, means
- Is a combination of an art and science that can take years to master
- Use expert judgment of others

5. Reporting

Communicate results effectively

- Track people hours and expenses in case software
- Provide electronic and physical documents of all findings
- Meet with proper authorities and possibly prepare to offer expert testimony
- Provide any needed clarification
- Identify overall impact on business and recommend any countermeasures
- Who, what, when, and how – important for court and other proceedings

5.6 Manage Communication with Relevant Parties

- Vendors
- Customers
- Partners
- Regulators
- Other Stakeholders

RACI Charts for Mapping Roles

R – Responsible **A** – Accountable **C** – Consulted **I** - Informed

	GRC* Department	Legal Department	Security Team	IT Operations
Establish the provider requirements	R/A	C	C	I
Build the governance scheme	R/A	C	C	I
Assess cloud vendor	A	I	R	R
Build the architecture	I	I	A/R	R
Conduct cloud migration	I	I	C	A/R

**GRC – Governance, Risk, and Compliance*

5.7 Manage Security Operations

- Security Operations Center (SOC)
- Monitoring of Security Controls (e.g., firewalls, Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), honeypots, vulnerability assessments, network security groups)
- Log Capture and Analysis (e.g., Security Information and Event Management (SIEM), log management)
- Incident Management

Security Operations Centers (SOC)

- For some organizations, the security operations is a function of the Network Operations Center (NOC)
- Many data centers have a centralized control center (or control tower) for continuous monitoring and visibility
  - Syslog, SNMPv3, SIEM, SOAR, and more
- The SOC does not have to be physically located in the data center or even on the same campus
- Most Cloud Service Providers have SOC's that are regional in scope and manage multiple zones (availability zones) and datacenters across a metropolitan area network (MAN) or Wide Area Network (SD-WAN) over high-speed fiber connections

Incident Management and Response

- Steps taken when a negative event disrupts normal operations
- Primary goal is to reduce the immediate impact
- Should have documented incident types/category definitions based on risk assessments, risk registers, and business impact analysis (BIA)
- Know roles and responsibilities of the first responders, including reporting requirements and escalation processes
- Collect contact lists, public relations people, and legal teams
- Best practice is to have pre-performed exercises, drills, and simulations

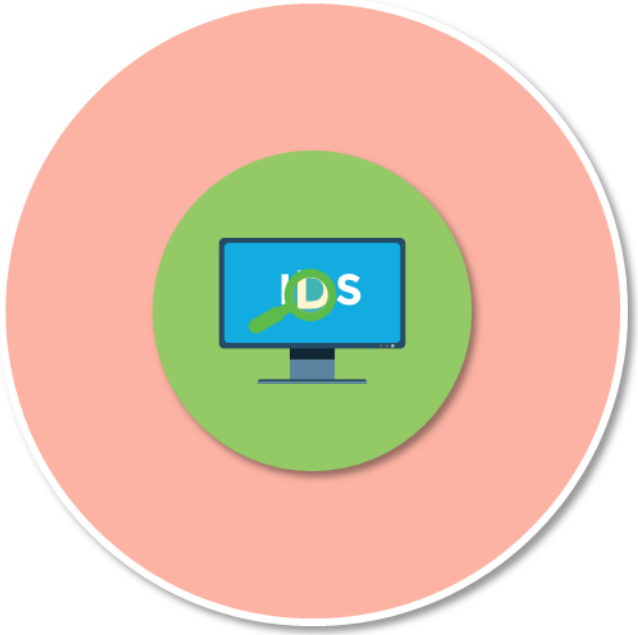
Incident Response Lifecycle

Preparation

- Involves all information gathering, missions, charters, and project initiation tasks
- Get buy-in and funding from executive management in order to know the scope of the response plan
- Establish incident response teams
  - Determine the roles and responsibilities of internal employees on incident response teams
- Establish first responders and processes for communication to relevant stakeholders
- Conduct IR exercises and drills based on budget

Incident Response Lifecycle

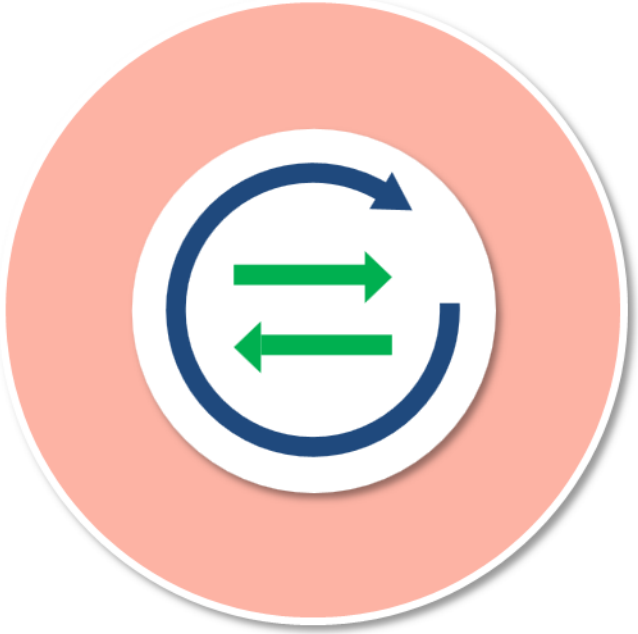
Detection



- Also referred to as "Identification"
- Separate an event from an incident or breach immediately, using pre-defined metrics and experience
- Categorize and prioritize the incident based on an established risk register or risk ledger
    - When did it occur?
    - How were you alerted?
    - Who made the discovery?
    - What is the scope of impact?
    - Does it qualify for escalation or disaster recovery?
    - **Can you quickly identify the root cause?**

Incident Response Lifecycle

Response



- Main goal is containment of the outbreak or malware exploit
- Implement short-term processes, such as disconnecting devices from the network
- Use firewalls, NG-IPS, ML algorithms, and other forensic tools to maintain separation, containment, and segregation
- Evaluate backups and snapshots for future recovery

Incident Response Lifecycle

Mitigation

- This step is also called "eradication" and is often integrated with the previous phase, Containment/Response, as opposed to being a separate action
- Involves determining the root cause of the incident and applying immediate remedies if available
- Involves removing all indicators of compromise and any action, artifacts, remnants, or fingerprints associated with the attack

Incident Response Lifecycle

Recovery

- The process of restoring negatively affected data, applications, systems, and devices to an established baseline performance level or, if possible, the original state
- **This often involves only remediation to a certain operational point and not total recovery**
- During this process, it is vital to establish that you are not in danger of another incident or breach
- Will often involve business impact analysis (BIA) metrics and indicators like RTO, RPO, and MTTR

Incident Response Lifecycle

Remediation

- This is more elaborate than recovery, as it involves a remedy that puts the application or system into a state before the incident occurred
- Remediation may take hours or weeks depending on the fact that the incident may rise to the state of a disaster or catastrophe and business continuity is occurring
- Recovery and remediation are often combined into the same phase or stage of incident response

Incident Response Lifecycle

Reporting

- Reports should be generated from physical, digital, and/or audio notes taken throughout the entire process
- Final reports should meet these requirements:
  - Be concise and comprehensive
  - Generate with different audiences in mind
  - Use newer graphical representation with Python and R programming tools
  - Include recommendations to prevent future incidents
  - Take problem management into consideration

Incident Response Lifecycle

Lessons learned

- Knowledge gained from the process of conducting the program
- Sessions usually held at the response close-out
- To share and use knowledge derived from an experience
- Endorse the recurrence of positive outcomes
- Prevent the recurrence of negative outcomes
- Try to avoid "blamestorming," although someone may be ultimately held accountable if expected due care and diligence were not performed

Domain 6

Legal, Risk and Compliance

6.1 Articulate Legal Requirements and Unique Risks within the Cloud Environment

- **Conflicting International Legislation**
- Evaluation of Legal Risks Specific to Cloud Computing
- Legal Framework and Guidelines
- eDiscovery (e.g., ISO/IEC 27050, Cloud Security Alliance (CSA) Guidance)
- Forensics Requirements

PROCESS OF SETTING LEGAL AND REGULATORY REQUIREMENTS

1. Understand the law and relevant regulations that apply to the organization
2. Classify data or operations that might require special attention
3. Establish provider contractual negotiation guidelines
4. Set provider evaluation criteria
5. Understand requirements coming from contractual obligations

6.2 Understand Privacy Issues

- **Difference Between Contractual and Regulated Private Data** (e.g., Protected Health Information (PHI), Personally Identifiable Information (PII))
- Country-Specific Legislation Related to Private Data (e.g., Protected Health Information (PHI), Personally Identifiable Information (PII))
- **Jurisdictional Differences in Data Privacy**
- Standard Privacy Requirements (e.g., ISO/IEC 27018, Generally Accepted Privacy Principles (GAPP), **General Data Protection Regulation (GDPR))**

Contractual vs. Regulated Private Data

In cloud computing, the legal responsibility for data processing falls to the consumer or user who solicits the services of a CSP

As in all other cases in which a third party is given the task of processing personal data, the user, or data controller, is responsible for ensuring that the relevant requirements for the protection and compliance with requirements for PII and PHI are satisfied or met

- **Contractual PII**
- Where an organization or entity processes, transmits, or stores PII **as part of its business or services**, this information is required to be adequately protected in line with relevant local state, national, regional, federal, or other laws
- The relevant contract should list the applicable rules and requirements from the organization who “owns” the data and the applicable laws to which the provider should adhere
- **Regulated PII**
- The key focus and distinct criteria to which the regulated PII must adhere is required **under law and statutory requirements**, as opposed to the contractual criteria that may be based on best practice or organizational security policies

GDPR

- GDPR is concerned with data protection and privacy in the EU and European Economic Area
- Covers all countries, citizens, and areas under its jurisdiction regardless of where the data is created, processed, or stored
- It does apply to other countries doing business with EU entities and is very strict
- On Oct 6, 2015, the European Court of Justice (ECJ) – Europe's highest court – concluded that the US-EU Safe Harbor agreement between the European Commission and the U.S. Department of Commerce was invalid
- **The Privacy Shield Framework has replaced Safe Harbor**
- Organizations can self-certify but certification must be done annually

GDPR

- U.S. companies with operations in Europe should consider adopting the following action:
  - Assess personal data flows from EU-to-US to define the scope of the cross-border privacy compliance challenge
  - Assess model contracts as at least a temporary replacement to Safe Harbor
  - Assess readiness to meet model clauses, remediate gaps, and organize audit artifacts of compliance with the clauses
  - Conduct a GDPR readiness assessment
  - Budget for GDPR remediation
  - Elevate risk and mitigation plans to the Board level
  - Enhance EU privacy programs to ensure they can pass an EU regulator audit, or litigation challenge, remediating GDPR gaps along the way
  - Conduct EU data-breach notification stress tests
  - Monitor changes in EU support for model contracts and binding corporate rules and prepare to shift to the operational adequacy mechanism

Trans-border Data and Information Flow

- Considerations should always include the flow of data, information, and goods across international borders and all legal and regulatory implications
  - These issues can change rapidly based on various geo-political factors
- Security initiatives must also consider variances in cultural norms
  - Customs, sensitivities, and behaviors (for example, European vs. Asian customs)
- Policies, controls, and procedures can differ based on region
- Countries are typically under different regulations and mandates
  - AGATE, IDABC, OBASHI, ITIL, ISO, or TOGAF
  - The Department of Commerce's Bureau of Industry and Security (BIS) controls nonmilitary cryptographic exports
- Cloud computing is transcending traditional boundaries and jurisdictional barriers and introducing new challenges

6.3 Understand Audit Process, Methodologies, and Required Adaptations for a Cloud Environment

- Internal and External Audit Controls
- Impact of Audit Requirements
- Identify Assurance Challenges of Virtualization and Cloud
- Types of Audit Reports (e.g., SSAE, SOC, and ISAE)
- Restrictions of Audit Scope Statements (SSAE and ISAE)
- Gap Analysis
- Audit Planning
- Internal Information Security Management System (ISMS)

COMPARING SECURITY AUDITING STANDARDS

Standards applicable to cloud security auditing.			
Standard	**Type**	**Strength**	**Sponsoring organization**
Service Organization Control (SOC) 2	Audit for outsourced services	Technology neutral	American Institute of CPAs
ISO 27001 and 27002	Traditional security audit	Technology neutral	ISO
NIST 800-53 rev, 4	Federal government audit	Technology neutral	National Institute of Standards and Technology
Cloud Security Alliance (CSA)	Cloud-specified audit	Dedicated to cloud security auditing	CSA
Payment Card Industry (PCI) Data Security Standard (DSS)	PCI Qualified Security Assessor cloud supplement	Cloud specific and Provides guidance	PCI DSS

6.3 Understand Audit Process, Methodologies, and Required Adaptations for a Cloud Environment (cont.)

- Internal Information Security Controls System
- Policies (e.g., organizational, functional, cloud computing)
- Identification and Involvement of Relevant Stakeholders
- Specialized Compliance Requirements for Highly-Regulated Industries (e.g., North American Electric Reliability Corporation/Critical Infrastructure Protection (NERC/CIP), HIPAA, PCI-DSS)
- Impact of Distributed Information Technology (IT) Model (e.g., diverse geographical locations and crossing over legal jurisdictions)

6.4 Understand Implications of Cloud to Enterprise Risk Management

- Assess Providers Risk Management Programs (e.g., controls, methodologies, policies)
- **Difference Between Data Owner/Controller vs. Data Custodian/Processor (e.g., risk profile, risk appetite, responsibility)**
- Regulatory Transparency Requirements (e.g., breach notification, SOX, GDPR)
- **Risk Treatment (i.e., avoid, modify, share, retain)**
- Different Risk Frameworks and Metrics for Risk Management
- Assessment of Risk Environment (e.g., service, vendor, infrastructure)

Data Ownership

- Owner – is often the creator of the information/data
  - Can categorize and determines classification or sensitivity level
- Custodian – Is the keeper of the information from a technical perspective
  - Ensures that CIA is maintained
- Steward - Manages the data and metadata from a business perspective
  - Ensures compliance (standards/controls) and data quality
- Processor – performs data input and runs batch jobs
- Chief privacy officer - ensures privacy of all data in the entire organization

Risk Treatment

- Risk acceptance (retain)
  - Do not implement any safeguards
  - Justification in writing is often required
- Risk avoidance
  - Choose not to undertake actions that introduce risk
- Risk transference/sharing
  - Pass the risk to a third-party, such as an insurance company or a cloud service provider
- Risk mitigation (modify)
  - Implement safeguards that will eliminate or reduce risk exposure - risk may exist, but impact is reduced

6.5 Understand Outsourcing and Cloud Contract Design

- **Business Requirements (e.g., Service Level Agreement (SLA), Master Service Agreement (MSA), Statement of Work (SOW))**
- Vendor Management
- Contract Management (e.g., right to audit, metrics, definitions, termination, litigation, assurance, compliance, access to cloud/data, cyber risk insurance)
- Supply-Chain Management (e.g., ISO/IEC 27036)

Service Level Agreements (SLA) and Master Service Agreement (MSA)

- The CSP must realize that the use of contractual agreements such as hosting/connection agreements and Service Level Agreements (SLAs) are used to allocate shared responsibility and risk among both cloud providers and cloud consumers
- An SLA defines the precise responsibilities of the provider and sets customer expectations
- Also clarifies the support system (service desk) response to problems or outages for an agreed level of service (based on support plan)
- The liability for the failure of one or more controls and the realization of risk can be appropriately documented and understood by all involved parties
- **The SLA is also called a Master Service Agreement (MSA)**

Master Service Agreement (MSA)

- As part of due diligence in your BCP, you should confirm any/all expectations with the candidate service provider and ensure that they are documented in your MSA/SLAs
- A master service agreement (MSA) is a contract two parties enter into during a service transaction
- This agreement details the expectations of both parties
- The goal of a master service agreement is to make the contract process faster
- It also should make future contract agreements simpler

Elements of SLA and MSA

- Confidentiality
- Delivery requirements
- Dispute resolution
- Geographic locations
- Intellectual property rights
- Limitations of liability
- Payment terms
- Venue of law
- Warranties
- Work standards

Statement of Work (SOW)

- A statement of work (SoW) is an agreement that establishes the expectations for a project or program and aligning the team(s) involved
- Details should clarify price, cost, timeline, deliverables, process, expectations of requirements, invoicing schedules, and much more, depending on the scope and breadth of the project
- Basically, a SoW is a document of agreement between a client and service or agent defining the scope and details of a project
- It is among the first documents you will use to establish the framework of a project before entering the planning and execution stages