Domain 10 Application Security

Secure Development Lifecycle:

- Secure Software Development Lifecycle (SSDLC).
- These stages identifies key processes, tools, and design patterns to be implemented in successful DevSecOps programs.

Figure 25: Stages of the Secure Software Development Lifecycle (SSDLC)

1. **Secure Design and Architecture:**
   - New product features and changes will route through design activities.
   - **Without including security in the design phase,** security measures will be introduced with a higher operating impact and cost at deployment or runtime.
2. **Secure Coding**:
   - Coding security controls that rely on automation as tools: identify weaknesses and vulnerabilities.
   - Security vulnerabilities and weaknesses will be more expensive to fix later in the SSDLC.
3. **Continuous Build, Integration, and Testing:**
   - The tools and processes to security test the functionality of an application/product.
4. **Continuous Delivery and Deployment:**
   - Without including security in deployment, there is a risk that vulnerabilities and poor security practices weaken an application/product, exploiting it for attacks in production environments.
5. **Runtime Defense and Monitoring**

Threat Modeling:

- Used in risk management to identify, assess, and address potential security threats.
- By focusing attention on areas more vulnerable to attacks.
- STRIDE is an example of threat modeling (lists six categories of application threats): **Spoofing / Tampering / Repudiation /Information Disclosure / Denial of Service (DoS) / Elevation of Privilege.** = Understanding these threats allows for the identification of potential vulnerabilities.

Testing: Pre-Deployment

- **Automated security code review:**
  - Also called: Static Application Security Testing (SAST).
  - **White-box** testing approach that examines an application's source code to identify existing security flaws or vulnerabilities. / an automated way.
  - Looks for logic errors, examines spec implementation, and checks style guidelines, and security flaws. It also scans the code for hard-coded credentials, keys, tokens, and secrets not allowing those to enter the repository and potentially leak.
  - Can be prone to false positives.
- **Software Composition Analysis (SCA)**:
  - Auditing the external open-source components your software (libraries and system components).
  - Assist in creating a **Software Bill of Materials (SBOM)** which is required by certain standards, providing transparency on all components used in the software.
- **Secrets, Images, and IaaC templates scanning:**
  - Detect application secrets embedded in code and configuration files, scan images for vulnerabilities, and detect Infrastructure as Code policy violations.
  - Can be made pre-deployment or after.
  - Could be integrated with SCA and static application security testing (SAST) tools.

Testing: Post Deployment:

- **Dynamic Application Security Testing (DAST):**
  - **A black box** testing in which a tester examines a web application while it is running but has no knowledge of the application.
  - Help determine whether the application is vulnerable and could be susceptible to a real malicious attack.
- **Interactive Application Security Testing**:
  - IAST is a method of application security testing that tests the application while the application is run by an automated test, human, or any activity interacting with the application functionality.
  - A combination of SAST and DAST.
- **Penetration Testing:**
  - Conducted as a simulated cyberattack, penetration testing aims to exploit known vulnerabilities, testing the resilience of security measures and the software's ability to withstand attacks.
- **Bug Bounty Program**:
  - Offer monetary rewards to ethical hackers for successfully discovering and reporting a vulnerability or bug on the live application.

Secrets Management:

- Sensitive elements are stored, accessed, and managed securely.

- Secrets can be avoided by assigning IAM roles/identities to services.
- Image Generating and managing X.509 certificates, including the creation of a key pair, submission of a certificate signing request (CSR), and issuance by a certificate authority (CA).

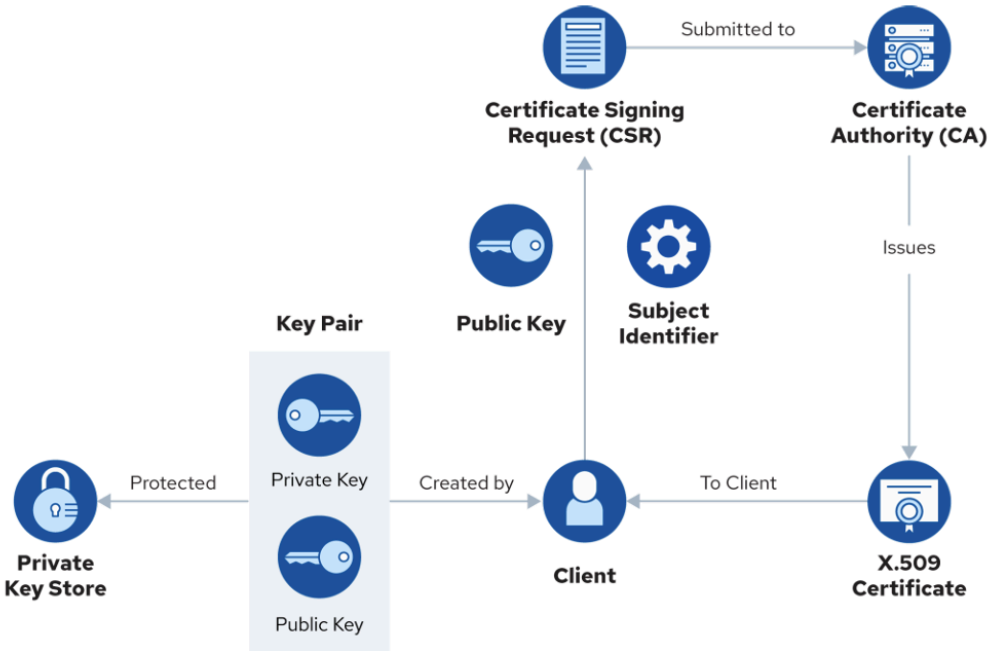

Figure 26: IAM and Secrets Management Process

DevOps & DevSecOps:

DevOps:

- Software development (Dev) + IT operations (Ops) : to speed up the software development lifecycle (SDLC) and deliver updates frequently.
- Collaboration and automation between developers and operations professionals to build and deploy code more efficiently.

DevSecOps:

- An extension of DevOps.
- Integrates security practices throughout the entire software development lifecycle. DevOps and DevSecOps = the automated deployment pipeline, which continuously executes security checks, scans, and tests.

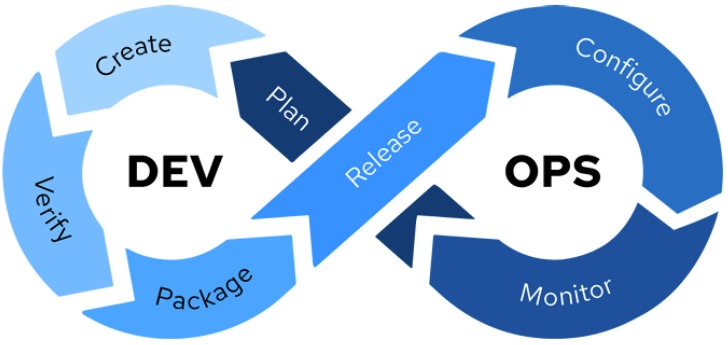

Figure 27: DevSecOps CI/CD Cycle

CI/CD Pipelines and ShiftLeft:

- "Shift Left" is the phrase used to indicate that "security" should move to earlier phases in the SSDLC to ensure secure-by-design and secure-by-default products. (ensuring each phase of the SSDLC, starting from inception and planning, is viewed through the lens of security.)
- "Shift Left" = Cost-effective compared to "bolting-on security" .
- **Continuous Integration (CI)**: Developers frequently merge their code changes into a shared repository. Pre-deployment automated security tests are required in this phase (e.g., SAST).
- **Continuous Deployment (CD):** it is automatically deployed to testing or staging environments. Ensures that code changes are delivered quickly and consistently. Post-deployment automated security tests are required in this phase (e.g., DAST).

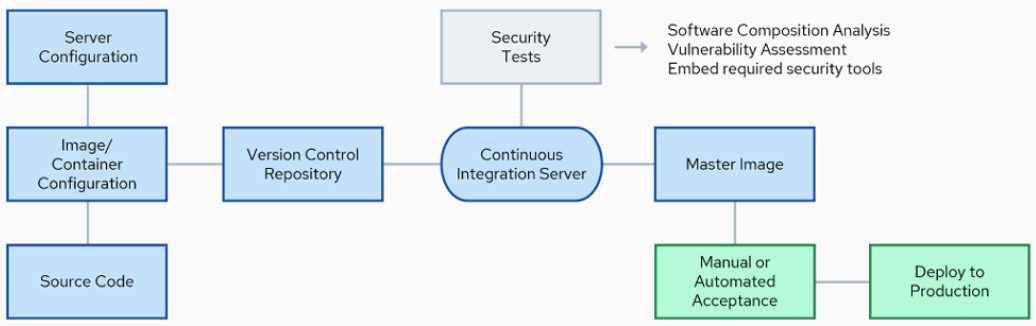

Figure 28: Secure Image Deployment Pipeline Process

- **WAF** tends to secure regular website HTTP traffic against known attacks
- **API gateways** address specific API security concerns including basic authentication and authorization actions, rate limiting, and request/response transformations.