### Data Protection

#### Best Practice
- Use HTTPS everywhere (CWE-106)
- Use HTTP Strict Transport Security (HSTS) (CWE-163)
- Use server-provided privacy options (CWE-404)

#### CME in Description
- CWE-106
- CWE-163
- CWE-404

#### Description
- HTTPS should be used for your entire application. If you have to use an HTTP redirect, make sure the "Allow Automatic" option is set to true and use it for all pages as well as in pages after the user is authenticated. Otherwise, if web services are used to communicate critical information, XML encryption should be employed.
- This HTTP Strict Transport Security (HSTS) header should be used from the very beginning.
- Set up server-provided privacy options to inform crawlers on which files should not be indexed.

#### Example
- "Strict-Transport-Security: max-age=31536000; include-subdomains; preload"

#### Authentication

#### Best Practice
- Don’t hardcode credentials
- Develop a strong password policy
- Implement access control based on user data and/or context
- Use tokens to ensure that the account is valid before returning a response

#### CME in Description
- CWE-106
- CWE-111
- CWE-163
- CWE-404

#### Description
- Never allow credentials to be stored directly within the application code.
- Develop a strong password policy for all users of the system.
- These systems are often based on stored access permission to specific resources. You should always use authentication to check user credentials as the primary method of validation. User credentials should never be used as JavaScript objects or sent through HTTP requests.
- Example: Use tokens to ensure that the account is valid before returning a response

#### Example
- "Content-Security-Policy: frame-src 'self'"
Why should you have an application security program?

- **LinkedIn**
  LinkedIn reported that the password hashes of 6.5 million members were extracted from their servers, a leading predictor used in a password cracking strategy, allowing unauthorized passwords to be hashed using SHA1. It was estimated that 6 million passwords were reverse within a few hours due to an estimated 35% of breaches, according to the information about these security vulnerabilities and how to prevent mistakes that occur while software is being developed and deployed.

- **AT&T**
  AT&T reported that the email addresses of 14,620,000 users were extracted from its website, including the addresses of some notable celebrities. The email addresses were made up a unique part within the URL, which could be modified to access another user’s email address.

- **Heartland Payment Systems**
  Heartland Payment Systems announced that a SQL Injection vulnerability resulted in the loss of an estimated 130 million credit card numbers. Reports were later released that up to 47,000 instances of 31 million on breached-reliances.

- **SANS AppSec Poster Contributors:**
  - SANS AppSec Poster Contributors: Jason Lam, Gregory Leonard, StH.Developer Training

What reasons should you fund an application security program?

- **Dynamic Analysis**
  Identifying vulnerabilities in a business application using automated scanning:
  - IBM AppScan, HP WebInspect, Acunetix WebScan, Qualys, Q-Test

- **Fuzz Testing**
  Testing technique that provides an application with an average of 15.4 million per incident in the U.S.:
  - Set a penetration test, fuzz the application’s source code for vulnerabilities using customised scripts.

- **Code Reviews**
  Recurse source code changes with a subject-matter expert for security implications:
  - Planfully search the application’s source code for vulnerabilities using customised scripts.

- **Static Analysis**
  Automated scanning of an application’s source code to locate vulnerabilities in the application.
  - HP Fortify, IBM AppScan Source, HP WebInspect, Qualys, Cenzic

- **Component Lifecycle Management**
  Implement a solution to find the use of third-party components with known published vulnerabilities. This phase typically involves taking an inventory of the components being used by an application and monitoring vulnerability intelligence feeds.
  - Static, Microsoft SDL, NIST 800-64, BSIMM, Open SAMM

- **SDLC Integration**
  Ensuring security is integrated into all phases of the development process through continuous integration, development, and security collaboration.
  - Microsoft SDL, NIST 800-53, NIST 800-25, Open SAMM

- **Metrics & Reporting**
  Provide dashboards and scorecards with qualitative feedback about the application security program and operational health of the application.
  - Windows Container

- **Application Security Training**
  Actual software security training that your team needs to prevent the pinpoint software security awareness training that your team needs to prevent vulnerabilities found in these scans should fail the build and require immediate patching before deployment into a testing environment.
  - Jenkins
  - Bitbucket
  - Travis

- **Continuous Integration**
  Building security into continuous integration starts with creating security-specific unit tests for critical sections of code such as authentication, password management, validation routines, and access control. Execute tests, security scanning, and automated testing (e.g., static and dynamic analysis) before deployment. Tools like OWASP Top 10, and vulnerability databases. Faked tests and high-fidelity vulnerabilities found in these scans should fail the build and require immediate patching before deployment into a testing environment.
  - Jenkins
  - Bitbucket
  - Travis

- **Continuous Delivery**
  Wiring automated security testing into the continuous delivery process allows to deploy static and dynamic testing to be completed. The scan results are fed through pre-defined acceptance criteria, with failures resulting in an automatic failure and deployment.
  - Jenkins
  - GitHub
  - Artifactory

- **Infrastructure as Code**
  Writing code to manage the server’s infrastructure, configuration, and deployment to a production environment. Steth container configuration code and data in version control to track the history changes made to the environment. Perform security review of the manifests and cookbooks to ensure hardened baseline requirements are met.
  - Puppet
  - Ansible
  - AWS

- **Container Security**
  Containerized applications images and an application’s dependency should be quickly installed and executed. Use signed certificates verify the application’s environment allows new servers to be quickly configured. Store infrastructure and data in version control to track the history changes made to the environment. Perform security review of the manifests and cookbooks to ensure hardened baseline requirements are met.
  - Docker
  - Kubernetes
  - Windows Container

- **Security Assessment**
  Ensuring that development and security teams perform periodic and a security assessment prior to publishing a mobile app.

- **Client-side Data Storage**
  Avoid storing sensitive information in a mobile app:
  - Use SQLite, SQLCipher, and/or Hyperscript, for storing sensitive information.

- **Key Cryptography**
  Encrypt sensitive data stored on the device by storing an additional layer of encryption with a symmetric encryption/decryption algorithm. Do not hard-code encryption keys or store them unprotected on the file system in locations such as a local file system, which can be easily stolen devices or use malware to easily steal devices or use malware to

- **Transport Layer Security**
  Mobile web services that support mobile applications are vulnerable to the same types of threats that blast web protocols. When applications are used to transmit data securely, to ensure that sensitive data is transmitted, encrypted data and session tokens are used to ensure data privacy and obtain data encryption. Performing certification chain validation, using approved protocols and ciphers, and ciphering proven techniques is seen in some mobile applications.

- **Server-side Security**
  Mobile web services that support mobile applications are vulnerable to the same types of threats that blast web protocols. When applications are used to transmit data securely, to ensure that sensitive data is transmitted, encrypted data and session tokens are used to ensure data privacy and obtain data encryption. Performing certification chain validation, using approved protocols and ciphers, and ciphering proven techniques is seen in some mobile applications.