Secure Coding Standards

# Security Principles

Defense in Depth

Positive Security Model

Fail Securely

Principle of Least Privilege

Separation of Duties

Security by Obscurity

Do not trust the client

# Access Control

Access control has three significant parts: ensuring the identity of a user (Authentication), determining what rights they have in the application (Authorization), and maintaining some level of state as to the user’s identity (Session Management).

## Authentication

Users must have individual user accounts with unique usernames.

Accounts should integrate with Active Directory when possible.

The login page of a web application, the page before the login page, and all logged in pages must use TLS.

All pages and resources must require authentication, unless explicitly intended to be public.

Applications should use the default authentication for their platform whenever possible.

Authentication controls must be enforced on a trusted system (e.g. on the server rather than the client)

Applications must only use HTTP POST to transmit credentials

Passwords must be at least 8 characters in length, and incorporate two alternative characters.

Stored user accounts must use PBKDF2 for password hashing.

“Remember me” features must only remember the username, not the password.

Login functions must not inform the user which of the username or password is incorrect, but instead use a generic message.

Web Services must use a hashed message authentication code for authentication.

## Authorization

The application must store authorization information (e.g. ‘role’) in an artifact that is inaccessible to the user.

The application should use non-enumerable values to refer to objects, if the reference is user editable.

Authorization systems must fail securely.

The application must force authorization controls on every request.

Authorization requests must be performed on the server, not the client.

## Session Management

When possible, all applications should use the default authorization for their platform.

Session identifiers must always be created on the server, not the client.

Logout functionality must delete the session id stored on the client, and the session information on the server.

Session identifiers must be stored as a cookie when the client is a web browser.

Web services must not store state information

Session identification must be supplemented by utilizing per-session strong random tokens or parameters, in order to mitigate cross-site request forgery attacks.

The application must disallow concurrent sessions with the same username.

The session timeout should be as short as is reasonable.

The application must generate a new session identifier after a successful login, if one was assigned for anonymous access.

# Content Management

Content management is dealing with the encoding and validation of data in the system.

## Input Validation

All input validation must FLTR: check Format, Length, Time, and Range against known good values.

All validation of data must occur on the server.

Character sets should be specified for all sources of input.

Data must be encoded into a common character set before validation.

Where possible, all input should be validated against a whitelist of known good values.

Where possible, the developer should use a standard, tested library for each type of input validation.

## Output Encoding

The application must contextually encode all data returned to the client that originated outside of the application’s trust barrier.

The application must contextually sanitize all output of un-trusted data to queries for SQL, XML and LDAP, and for operating system commands.

Where possible, the developer should use a standard, tested library for each type of output encoding.

# Browser Interaction

Browser interaction defines configuration and modification of the HTTP request/Response pairs, at the server or browser level.

## HTTP

The server must have all unused HTTP verbs disabled.

The headers should include an X-Frame-Options header unless the site is designed to be framed.

HTTP errors must use custom pages that do not disclose the web server version.

## JavaScript

The application must authorize and validate all AKAX requests.

All JSON or XML responses must be correctly encoded.

All Access-Control-Allow-Origin headers are to be set to appropriate domains.

Third-party JavaScript Libraries must be kept up-to-date on every release.

All externally hosted JavaScript libraries must be hosted by trusted providers.

The application must not trust data stored in browser storage.

The application must not store sensitive information in browser storage.

# Exception Management

Exception Management is the practice of handling system errors correctly.

## Error Handling

Applications must use structured error handling.

Error messages must only expose the minimum amount of information to the user in order to solve the problem.

Error pages must not reflect a user editable string on the page.

Error handling logic dealing with access control should deny access by default.

Applications must implement generic error controls and use custom error pages.

Applications must release resourced if an error occurs.

## Logging

An application must use a standardized logging system.

Applications must log successful and failed authentication attempts, multifactor authentication, lockouts, access to sensitive data, and account creation or deactivation.

Applications must not log sensitive data, credentials, or unhashed session information.

Log entries must include timestamp, userid, source ip, type of event, result of event, error codes, and the session ID in a hashed form.

# Cryptography

All cryptographic functions used to protect secrets from the application user must be implemented on the server.

Master secrets and cryptographic keys must be securely protected from unauthorized access.

All random numbers, random file names, random GUIDs, and random strings should be generated using the cryptographic module’s approved random number generator when these random values are intended to be un-guessable.

## Encryption

Data should be encrypted with Chacha20-Poly1305 or AES-GCM.

Symmetric encryption key length must be at least 256 bits.

Asymmetric encryption should use NaCl if possible. Avoid RSA and move to elliptic curve based libraries if NaCl isn’t available on your platform.

## Hashing

APIs must be secured using a Hashed Message Authentication Protocol (HMAC) with SHA-2.

Passwords must be hashed using scrypt, bcrypt, or PBKDF2.

## Secure Communication

All communication must always use TLS 1.1 or 1.2.

# System Configuration

## File Management

System configuration files must have restricted access.

Security configuration must be human readable for audit.

All files must be tracked in a source control system.

Uploaded files must be stored in a non-addressable location

The application must not pass directory or file paths in user editable variables.

Uploaded files should be scanned for malware.

## Deployment

Deployment must be accomplished by a scripted system.

Deployment scripts must only publish files that are absolutely required to run the application.

## Server Setup

Applications must have separate servers for Development, Test, and Production

Servers must not run as a privileged system account.

Servers must be running the latest version of software.