Web App Security

Presented by

Sagar
Java Consultant

App Security

- Application security in Java involves implementing various measures to protect web applications from vulnerabilities and attacks.

- **Practices and considerations**

  - **Input Validation**: Validate all user inputs to prevent injection attacks.

  - **Authentication and Authorization**:
    - Implement strong authentication mechanisms (e.g., multi-factor authentication) and enforce proper authorization checks to ensure that users only access resources they are allowed to.
  - **Session Management**:
    - Securely manage session tokens and cookies. Use secure session management practices such as setting HttpOnly and Secure flags on cookies, implementing session timeout mechanisms

App Security

- **Secure Communication**: Use HTTPS/TLS to encrypt data transmitted between clients and servers to prevent eavesdropping and man-in-the-middle attacks.
- **Error Handling**: Implement proper error handling and logging
- **Database Security**: Use parameterized queries or prepared statements to prevent SQL injection attacks
- **Secure Dependencies**: Manage dependencies carefully and keep them up-to-date to avoid using libraries with known vulnerabilities

The role of the container in Security

Container Role :

In the context of web applications, the term "container" typically refers to the runtime environment where the application runs.

- For Java web applications, this often means a servlet container like Apache Tomcat, Jetty, or a Java EE application server like WildFly, WebLogic, or WebSphere.

- The container plays a crucial role in ensuring the security of the deployed applications.

- **Isolation**: Containers provide isolation between different applications running on the same server. Each application typically runs within its own context (e.g., web application directory structure, class loader.

App Security

Request Handling and Filtering:

Containers often provide built-in mechanisms for handling requests and filtering inputs.

For example, servlet containers can handle HTTP requests and responses,and they often include filters that can be configured to intercept requests for purposes such as input validation, logging, or security checks.

Resource Management:

Containers manage system resources such as threads, memory, and network connections used by deployed applications

Integration with Security Frameworks:

Many containers integrate with security frameworks and standards such as Basic Authentication, Java EE security APIs, Spring Security, OAuth, and others.

Java Security Authentication Models

1. **HTTP Basic Authentication:**

Description:

HTTP Basic Authentication is a simple authentication scheme built into the HTTP protocol.

It involves sending the username and password encoded as a Base64 string with each request.

The app need not always be a web app. A stand-alone app might use java.net API **Demo

2. Form-Based Authentication:

Description:

Form-Based Authentication involves presenting a login form to the user, which collects username and password.

Upon submission, the credentials are validated on the server-side, and if correct, the user gains access to the application. **Demo

*Refer RequestDispatcher & Servlet Filters Apps

Web Data Encryption

- Data is encrypted as one of the security considerations.

The Encryption API has
java.security.MessageDigest
java.security.NoSuchAlgorithmException classes.

MessageDigest has methods - **update()** – to update the given string data into byte array[].

digest() - returns the converted byte array into encrypted form

** Demo