Secure Programming Techniques – An Overview

Prof. Zaheed Shaikh

GUIDE OVERVIEW

- Technology agnostic coding practices
- What to do, not how to do it
- Compact, but comprehensive checklist format
- Focuses on secure coding requirements, rather then on vulnerabilities and exploits
- Includes a cross referenced glossary to get developers and security folks talking the same language

CHECKLIST

- Data Validation
- Authentication and Password Management
- Authorization and Access Management
- Session Management
- Sensitive Information Storage or Transmission
- System Configuration Management
- General Coding Practices
- Database Security
- File Management
- Memory Management

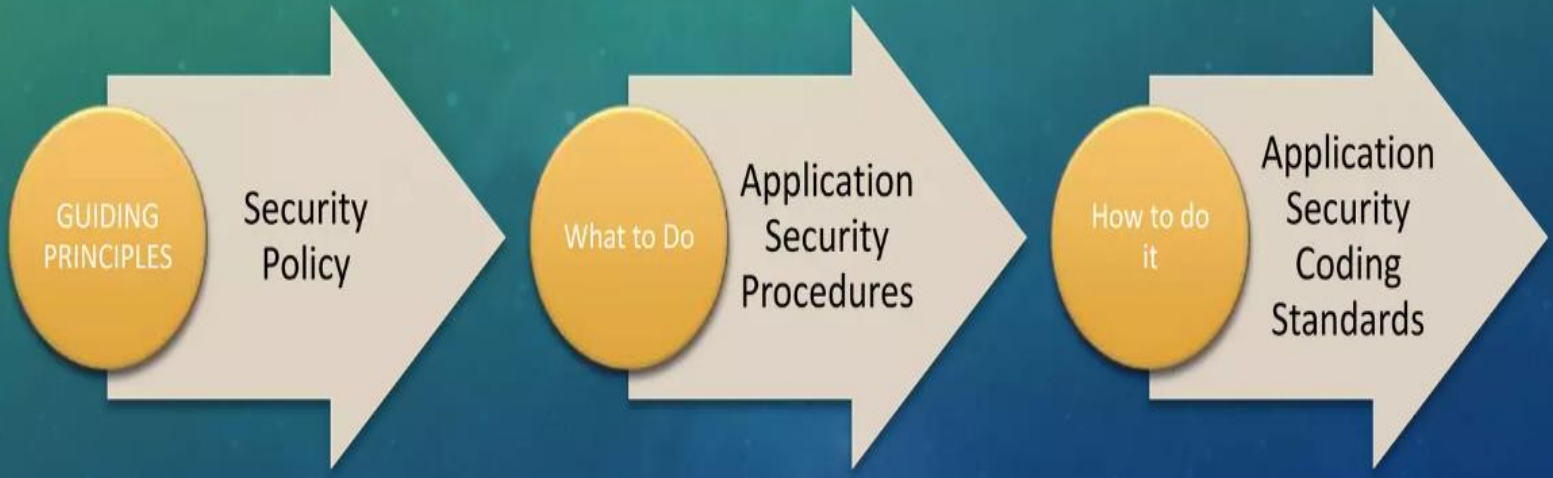
CHECKLIST PRACTICES

- Short and to the point.
- Straight forward "do this" or "don't do that"
- Does not attempt to rank the practices
- Some practices are conditional recommendations that depend on the criticality of the system or information
- The security implications of not following any of the practices that apply to the application, should be clearly understood

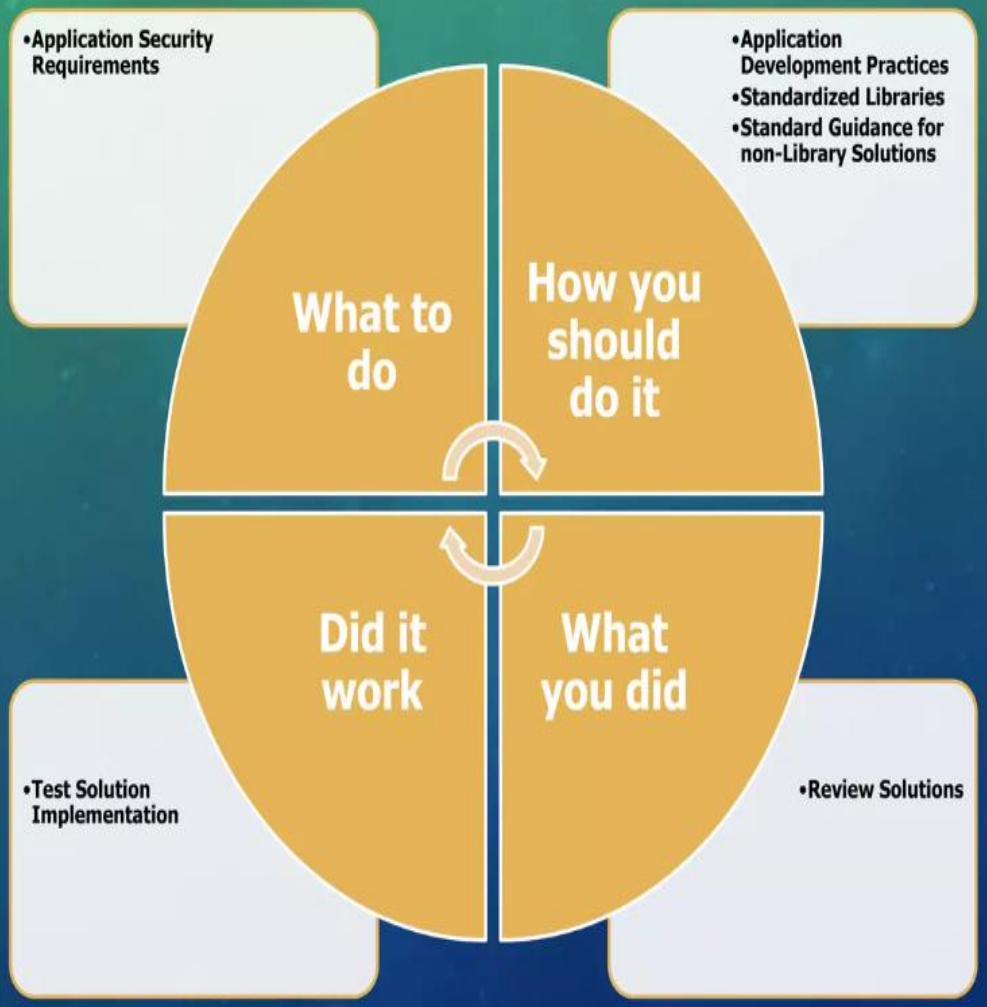
EXTRACT - DATABASE SECURITY

- Use strongly typed parameterized queries. Parameterized queries keep the query and data separate through the use of placeholders. The query structure is defined with place holders and then the application specifies the contents of each placeholder.
- Utilize input validation and if validation fails, do not run the database command.
- Ensure that variables are strongly typed.
- Escape meta characters in SQL statements.
- The application should use the lowest possible level of privilege when accessing the database.
- Use secure credentials for database access.
- Do not provide connection strings or credentials directly to the client. If this is unavoidable, encrypted them.
- Use stored procedures to abstract data access.
- Turn off any database functionality (e.g., unnecessary stored procedures or services).
- Eliminate default content.
- Disable any default accounts that are not required to support business requirements.
- Close the connection as soon as possible.
- The application should connect to the database with different credentials for every trust distinction (e.g., user, read-only user, guest, administrators).

DEVELOPING GUIDANCE DOCUMENTS

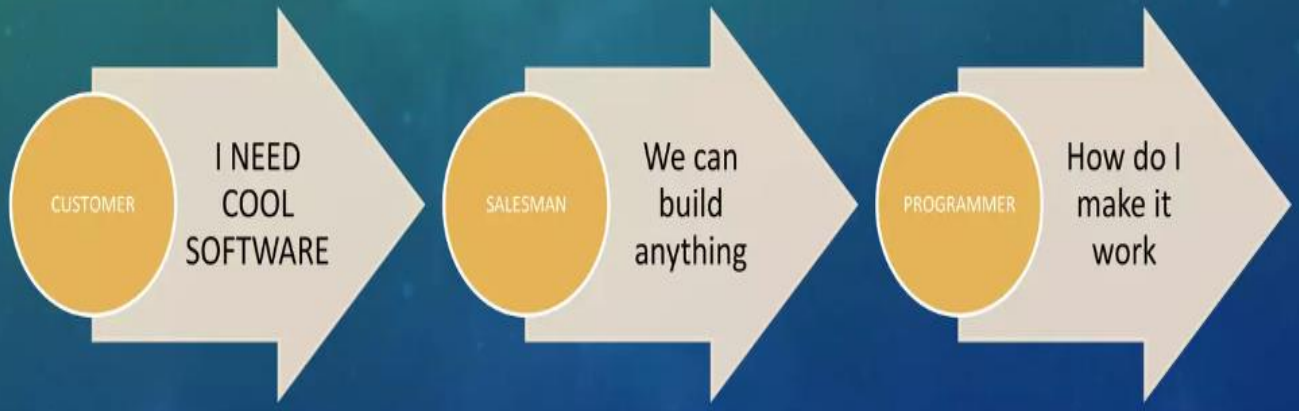


SUPPORT SECURE DEVELOPMENT LIFECYCLE



CONTRACTED DEVELOPMENT

- Identify security requirements to be added to outsourced software development projects.
- Include them in the RFP and Contract



SUMMARY

- Makes it easier for development teams to quickly understand secure coding practices
- Assists with defining requirements and adding them to policies and contracts
- Provides a context and vocabulary for interactions with security staff
- Serves as an easy desk reference

A SECURE DEVELOPMENT FRAMEWORK

- Implement a secure software development lifecycle
  - OWASP CLASP Project
- Establish secure coding standards
  - OWASP Development Guide Project
- Build a re-usable object library
  - OWASP Enterprise Security API (ESAPI) Project
- Verify the effectiveness of security controls
  - OWASP Application Security Verification Standard (ASVS) Project)
- Establish secure outsourced development practices including defining security requirements and verification methodologies in both the RFP and contract
  - OWASP Legal Project