Introduction

● Who am I?
  ○ Zak Dehlawi

● Why am I here?
  ○ Talk to you about Secure Development Lifecycles (SDL) and WebApp security
Introduction

- **PhD Student**
  - UW Information School
  - Advising Committee
    - Dr. Barbara Endicott-Popovskiy
    - Dr. Jochen Scholl
    - Dr. Yoshi Kohno

- **Education**
  - Johns Hopkins M.S. Security Informatics
  - UW CSE and PoliSci bachelor degrees
Introduction

- Senior Security Engineer
  - Security Innovation, Inc.
    - Cool place, come work there with me
  - Primary Tasks:
    - Threat Modeling
    - Secure Development Lifecycle
    - Penetration Testing
Outline

- SDL
- OWASP
- A1-Injection
- A2-Broken Authentication and Session Management
- A3-Cross-Site Scripting (XSS)
- A6-Sensitive Data Exposure
● Why a Secure Development Lifecycle?
  ○ Reduces the total number of vulnerabilities
  ○ Addresses compliance requirements
  ○ Reduce the cost of development
    ■ Fixes later in development cycle are more costly to address

Source: Microsoft SDL - Benefits, NIST May 2002
• Phases
  ○ Training (Regularly scheduled)
  ○ Requirements
  ○ Design
  ○ Implementation
  ○ Verification
  ○ Release
  ○ Response (Post-Release)
Phases

- Training (Regularly scheduled)
  - Developers, Testers, PMs, Architects
- Requirements
  - Establish security requirements (Compliance + regulation)
- Design
  - Security architecture review
  - Attack surface analysis
  - Threat modeling
● Phases
  ○ Implementation
    ■ Security code reviews
    ■ Static code analysis
  ○ Verification
    ■ Penetration testing
    ■ Fuzz testing
  ○ Release
    ■ Incident response plan
    ■ Black-box penetration testing
  ○ Response (Post-Release)
    ■ Execute incident response plan
● OWASP Top 10-2013 (Select few)
  ○ A1-Injection
  ○ A2-Broken Authentication and Session Management
  ○ A3-Cross-Site Scripting (XSS)
  ○ A6-Sensitive Data Exposure
OWASP: A1-Injection

● Types:
  ○ SQL
    ■ Update, delete, read arbitrarily from database
      ● Little Bobby Tables
        ○ Robert'); DROP TABLE Students;--
  ○ OS
    ■ Execute arbitrary OS or interpreter commands
  ○ XML
    ● XML Bombs with inline DTD
    ● XML External Entity attacks
  ○ JSON
    ● Define arbitrary entities
  ○ etc.
OWASP: A1-Injection

● Mitigations
  ○ SQL
    ■ Use ORMs and DALs
    ■ Use parameterized queries
  ○ OS
    ■ Never eval or execute user supplied input
  ○ XML
    ■ Disable inline DTD in the XML parser
      ● Default in most parsers now
  ○ JSON
    ■ Use CSRF tokens
  ○ etc.
OWASP: A1-Injection

WHITELIST

ALL THE THINGS
OWASP: A2-Broken Authentication and Session Management

- **Types:**
  - Session Fixation
  - Session tokens are weakly generated
  - Session tokens are not protected by SSL/TLS

- **Mitigations**
  - Issue new sessions upon login
  - Use cryptographically secure random number generators
    - Or make sure your framework is using one
  - Use HTTPS and mark cookies as Secure
OWASP: A3-Cross-Site Scripting

- **Types:**
  - Stored
  - Reflected
  - DOM based
  - Includes HTML injection
    - Favorite test is to use `<marquee>` tag

- **Mitigations**
  - Escape untrusted input
    - Frameworks have tools for that
OWASP: A3-Cross-Site Scripting

WHITELIST

ALL THE THINGS
OWASP: A6-Sensitive Data Exposure

● Types:
  ○ Personally Identifiable Information
  ○ Credit Cards
  ○ Passwords

● Mitigations
  ○ Encrypt in database
    ■ Attackers can steal encryption keys
  ○ Use SSL/TLS for transmission
  ○ DON’T STORE IT!
    ■ Use OpenID
    ■ Email based authentication
Password Storage

Thou Shalt NOT:
- Store plaintext passwords
- Encrypt passwords
- Use vanilla SHA1, SHA512, MD5, etc.

Thou Shalt:
- Use password storage mechanism
  - bcrypt, scrypt, PBKDF2
- Use a unique salt per password
Information Security Careers

- 0% unemployment rate
- Federal government is hiring
- Corporate world is hiring
- Pays pretty well
- Information security is fun
- You get to be a cyber-warrior
Contact Information

- Zak Dehlawi
  - zdehlawi@securityinnovation.com
  - zaxim@uw.edu

Questions!?? and Brainstorm!!1