# Secure Coding

# Overview

- Cybersecurity has risen to the top priority discussion items, and it is the subject of the US-Russia presidential communications. The number of ransomware attacks doubled in the past year, and other attacks are on the rise.
- This course teaches a comprehensive approach to cybersecurity. It starts with threat modeling, creating the lay of the land. It then continues with common attacks, with the principles of designing secure multi-layer systems, and goes into the details of secure coding for the target languages.
- Also included are securing runtime environments and modern security frameworks.

# Audience

- Developers, team leads, project managers

# Skill Level

- Introductory - Intermediate

# Duration

- Three days

# Format

- Lectures and hands on labs. (50% - 50%)

# Prerequisites

- Recommended: Cybersecurity awareness
- Comfortable developing code in the target environment

# Lab environment

- Zero Install: There is no need to install software on students' machines!
- A lab environment in the cloud will be provided for students.

# Students will need the following

- A reasonably modern laptop with unrestricted connection to the Internet. Laptops with overly restrictive VPNs or firewalls may not work properly.
  - A checklist to verify connectivity will be provided
- Chrome browser

# Detailed outline

# Threat modeling

- STRIDE attack classification
- Security terminology
- Threat modeling
- CVSS attack assessment
- Labs on threat modeling

# Common attacks

- Cross site scripting
- Malicious file execution
- Session hijacking
- Encryption
- Unsecured direct object reference
- Failure to authorize/hidden URLs
- Cross site request forgery (CSRF)

# Secure design

- Security at high level, all the way from testing, deployment, and maintenance
  - Start from non-functional requirements
- Layered design concepts
- Object layer
- Persistence layer
- Presentation layer

# Countermeasures

- Validation
- Validation controls
- Strong typing
- Regular expressions
- White list
- Scrubbing
- Black list
- Encoding
- CAPTCHA
- Honey pots
- Avoiding SQL injection
- Parametrizing queries/Prepared statements
- Stored procedures
- Entity Frameworks/Hibernate
- Avoiding cross site request forgeries

# Modern security frameworks

- Introduction to modern frameworks
  - Vault
  - Consul
  - Anthos
- Modern security design patterns
  - Dynamic secrets
  - Automatic credential rotation
  - Cubbyhole response wrapping
  - Encryption as a service
- Where to go from here

# Authorization and Authentication

- SSO (at least high-level)
- Spring security
- .NET authentication (just mention)
- Basic & Digest
- Forms
- Windows authentication (just mention)
- JAAS and other Java authentication services
- Authorization
- Password security
- Brute force attacks
- Password resets
- Secret questions/answers
- SSL/TLS

# Session security

- Perfect Secrecy
- Asymmetric and symmetric encryption
- Session IDs
- Policies
- Hijacking/Fixation Attacks

# Framework architecture

- Threading
- Privileges
- Audits/Logs
- Secure coding
- Encryption services
- Static code analysis
- Securing the API (both publishing and consuming API)
- JWT
- Dynamic code analysis (e.g. with Spotbugs)

# Securing the runtime environment

- Spring boot
- .NET (mention)
- Code Access
- GAC
- Strong named assemblies
- CLR
- Security Zones
- Permissions
- Security policy

# Security future

- Zero-trust networks
- Artificial intelligence
- Quantum computing / cryptography