Security Development Lifecycle

Matthew Remmel,
Matt Lorentz

Project Overview

- UI-Labs - Project Owners / Developers
- Rolls Royce - Potential Customer
- Been in development for ~3 years
- Rolls Royce wanted recommendation on functionality and security status.

Technology Stack

- Angular App up front with PHP serving pages in some places
- Java Spring REST backend
- PostgresSQL for the database.
- Deployed on Azure

Security Audit Process

- White Box Evaluation
- Automated Scans
- Manual attacking via proxy
- Automatic and manual source code analysis
- Generally just thinking like an attacker

Discovered Issues

- Committed credentials in (public) Github
- SQL Injection vulnerabilities in several endpoint
- Arbitrary CRUD operations on anyones documents
- No server side validation
- Timing attack on file upload
- Attacking document versioning

Broken Development Process

- Broken build process - can't update dependencies
- Not able to run project locally (at the time)
- Pumping out new features without addressing tech debt and known issues

Security Foundations

Some Terminology

- Adversary (threat agent)
- Attack
- Countermeasure
- Risk
- Security Policy
- System Resource (Asset)
- Threat
- Vulnerability

CIA-AA Triad

- Confidentiality
- Integrity
- Availability
- Authenticity
- Accountability

Security Challenges

- Not as easy as it seems
- Battle of the wits
- Little perceived benefit until security failure occurs
- Constant monitoring and maintenance
- Often an afterthought
- Security undermines convenience often

Security Development Lifecycle

Weakest Link

- **Application Layer is the weakest point**
- 75% of money spent on infrastructure. 3/4 of attacks target applications
- Only 1/3 of developers are confident in their code

A Step Forward

- Integrate a Security Development Lifecycle
- Train developers in secure coding practices
- Incorporate Threat Modeling, Secure Coding Techniques, Secure Code Review, and Security Focused Testing into the development process.

SDL-IT & SDLC

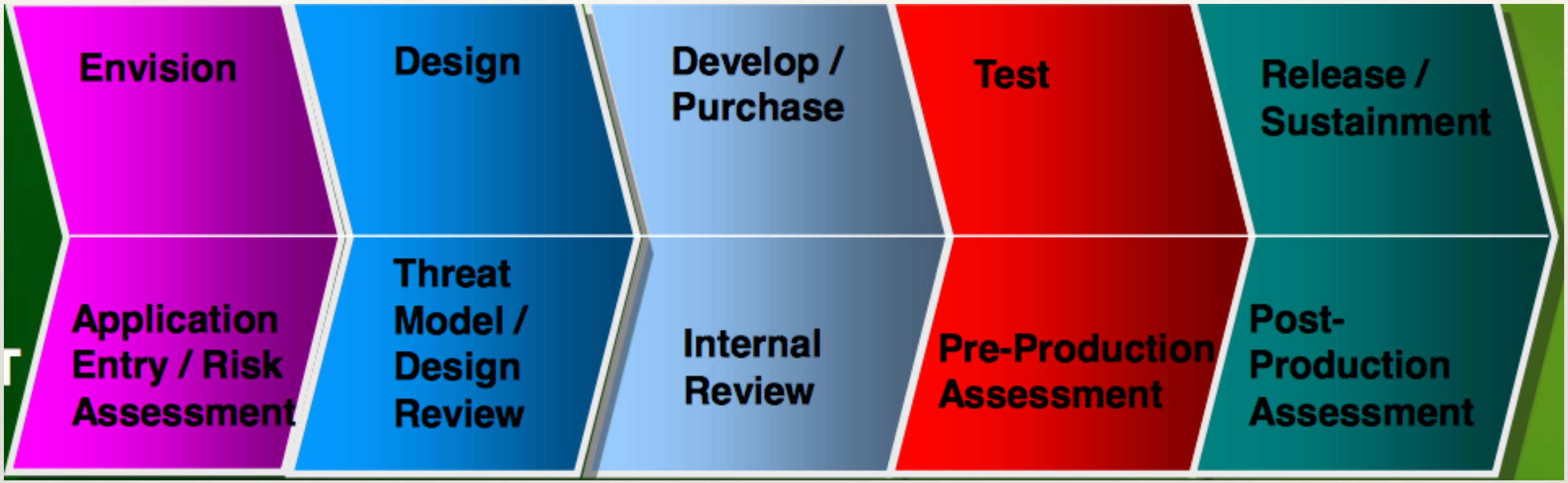


Image: https://www.owasp.org/images/d/d0/OWASP_SDL-IT.pdf

Risk Assessment

- Happens during the envisioning stage
- Application Inventory
- Determine Application Risk Categorization
  - High Risk
  - Medium Risk
  - Low Risk

Threat Model / Design Review

- Happens during design review
- Provides a consistent methodology for objectively evaluating threats
- Review application design to verify compliance with security standards and best practices
- Verify application meets security principles

Internal Review

- Happens during development / code review
- Review security checklists and policies
- Team conducts code review and attack and penetration tests

Pre-Production Assessment

- Happens during (beta) testing phase
- Low Risk Applications
  - Host level scan
- High/Medium Risk Applications
  - Host Level Scan
  - White Box Code Review

Post-Production Assessment

- Happens after production deployment
- Host level scan

SDL Resources

- Mitre - Common weaknesses & vulnerabilities
- OWASP - www.owasp.org
- Microsoft SDL - https://www.microsoft.com/en-us/sdl/
- Static Code Analysis - Snyk, FindBugs, ect
- Security Checklists
- Threat Modeling - Adam Shostack
    - Elevation of Privilege Cards
- Secure Design Principles (next slide)

Secure Design Principles

- Economy of Mechanism - Keep it simple
- Fail-safe Defaults - Permission rather than exclusion
- Complete Mediation - Caching causes headaches
- Open Design - Obfuscation != security
- Separation of Privilege - Similar to least privilege
- Least Privilege - Need to know basis

- Least Common Mechanism - (A principle of kernel design)
- Psychological Acceptability - Keep it usable
- Isolation - Reduce pivoting ability
- Encapsulation - Restrict communication between domains
- Modularity - Integrate security without coupling
- Layering - Defense in depth
- Least Astonishment - Transparent security

Threat Modeling

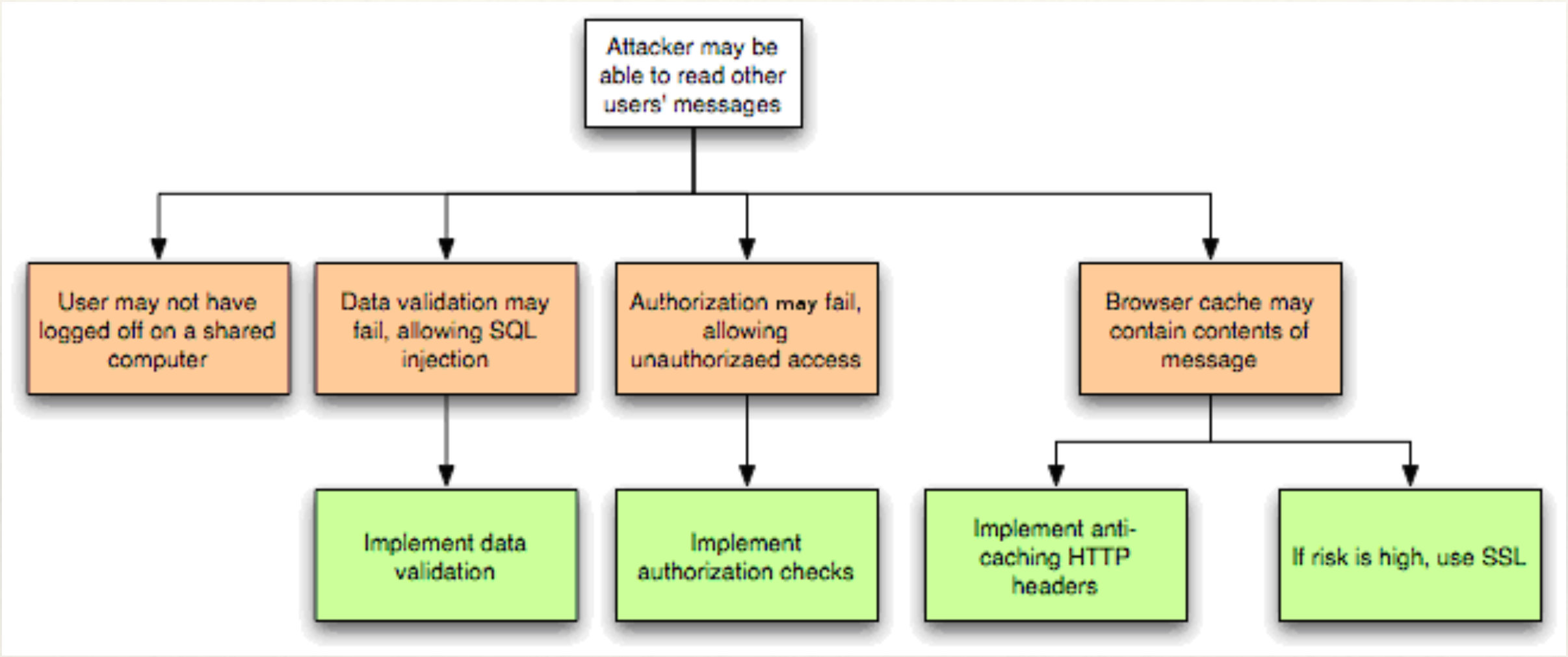
- Decompose the application
- Determine and rank threats
- Determine countermeasures and mitigation

https://www.owasp.org/index.php/Application_Threat_Modeling

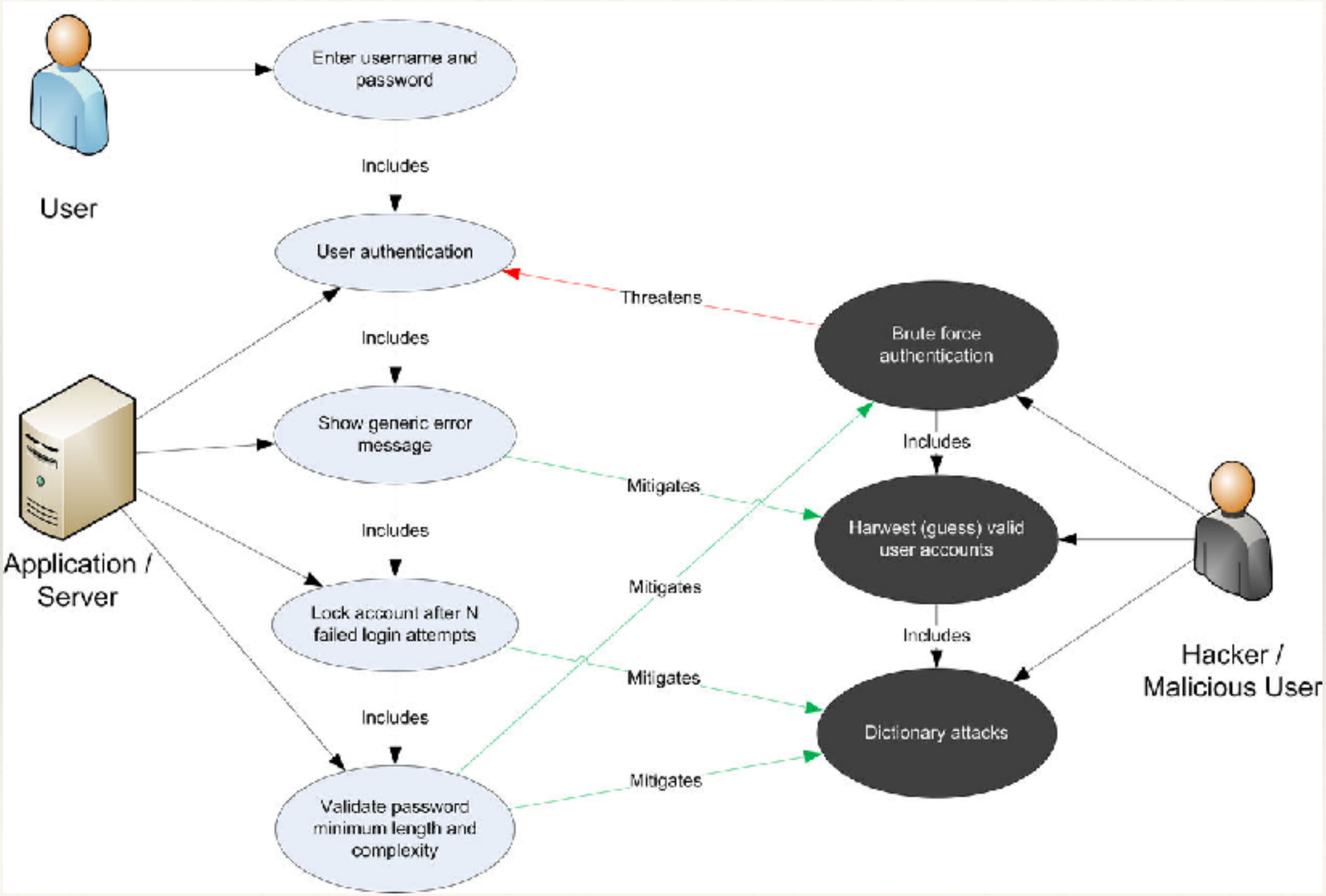
Threat Modeling Tools

- Diagrams
- STRIDE - Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege
- DREAD - Damage Potential, Reproducibility, Exploitability, Affected Users, Discoverability

Example Diagram



Example Diagram



Demo

Disclaimer: Do not use any tools against anything you don't have explicit (preferably written) permission to attack. Computer fraud and abuse penalties are very severe.

Final Thoughts

- The need for security is obvious
- Delivering quality software means delivering **adequately** secure software
- Continuous improvement process of people, processes, and tools
- Security first attitude