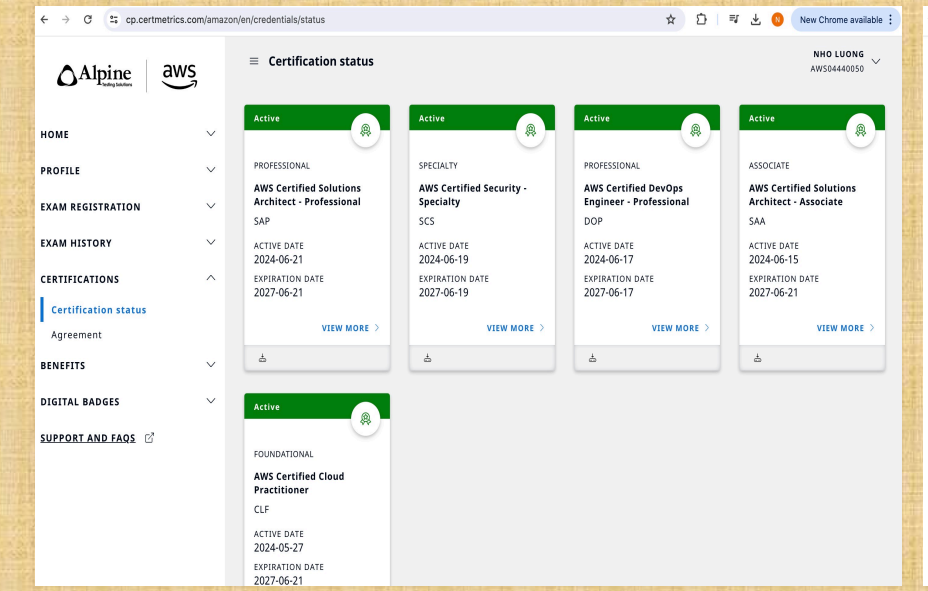
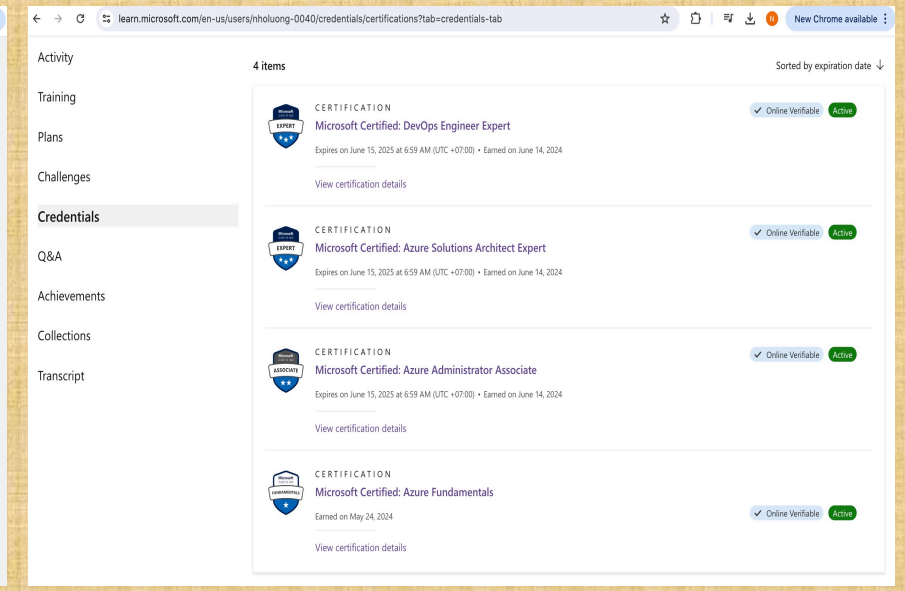
Managing Security Risk Technical Debt

Author: Nho Luong

Skill: DevOps Engineer Lead





The Why?

Author: Nho Luong
Skill: DevOps Engineer Lead

The Threat Landscape

- Local Area Networks
- First computer virus
- Boot sector viruses
- Create notoriety or cause havoc
- Slow propagation

1980s

- Internet Era
- Macro viruses
- Script viruses
- Create notoriety or cause havoc
- Faster propagation

1990s

- Broadband prevalent
- Spyware, Spam
- Phishing
- Botnets
- Rootkits
- Financial motivation
- Internet wide impact

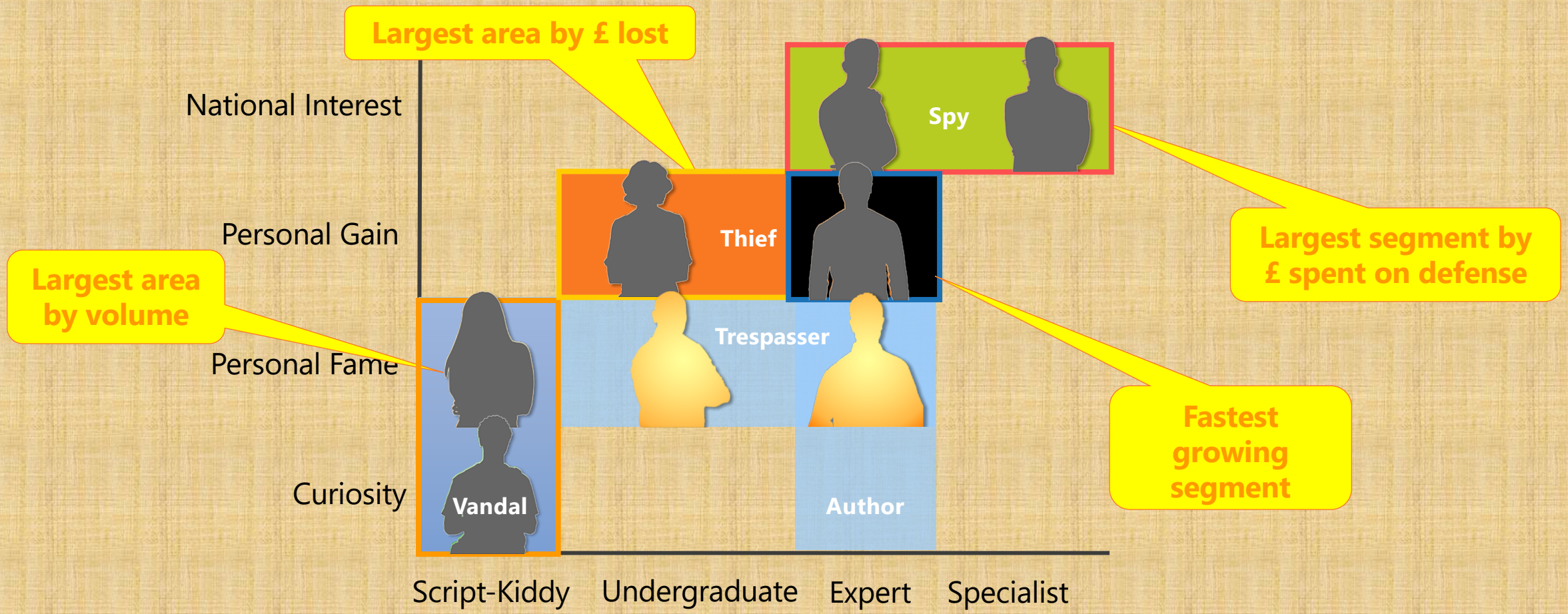
2000s

- Peer to Peer
- Social engineering
- Application attacks
- Financial motivation
- Targeted attacks
- Ransomware

2010s

Author: Nho Luong
Skill: DevOps Engineer Lead

Evolving Threats



Author: Nho Luong
Skill: DevOps Engineer Lead

The What?

Overview

Author: Nho Luong
Skill: DevOps Engineer Lead

How does IT secure the Business?

Firewalls

Penetration Tests

The Result?

- Only protect the boundary
- Firewalls are only part of a solution
- Organisations need to think more holistically
- IT can only protect against what they know and if developers don't tell them .....

Author: Nho Luong
Skill: DevOps Engineer Lead

The Problem is Software

"Malicious hackers don't create security holes; they simply exploit them. Security holes and vulnerabilities – the real root cause of the problem – are the result of bad software design and implementation."

Author: Nho Luong
Skill: DevOps Engineer Lead

A word from the analysts

“75 percent of hacks happen at the application”

Gartner “Security at the Application Level”|

"The conclusion is unavoidable: any notion that security is a matter of simply protecting the network perimeter is hopelessly out of date”

IDC and Symantec

“If only 50 percent of software vulnerabilities were removed prior to production … costs would be reduced by 75 percent”

Gartner “Security at the Application Level”

“The battle between hackers and security professionals has moved from the network layer to the Web applications themselves"

Network World

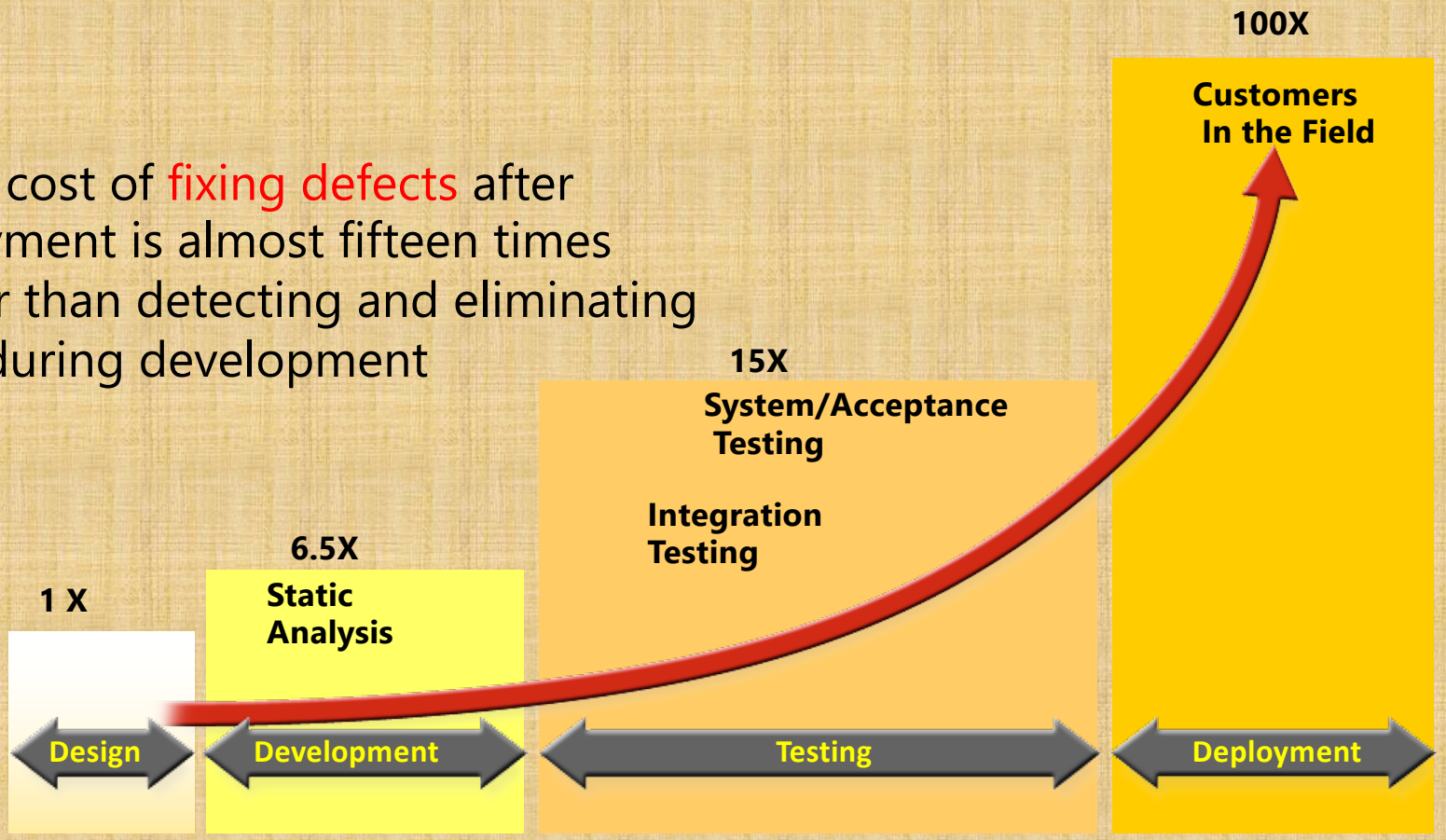
“64 percent of developers are not confident in their ability to write secure applications”

Microsoft Developer Research

Author: Nho Luong
Skill: DevOps Engineer Lead

Trauma of Reactive Security

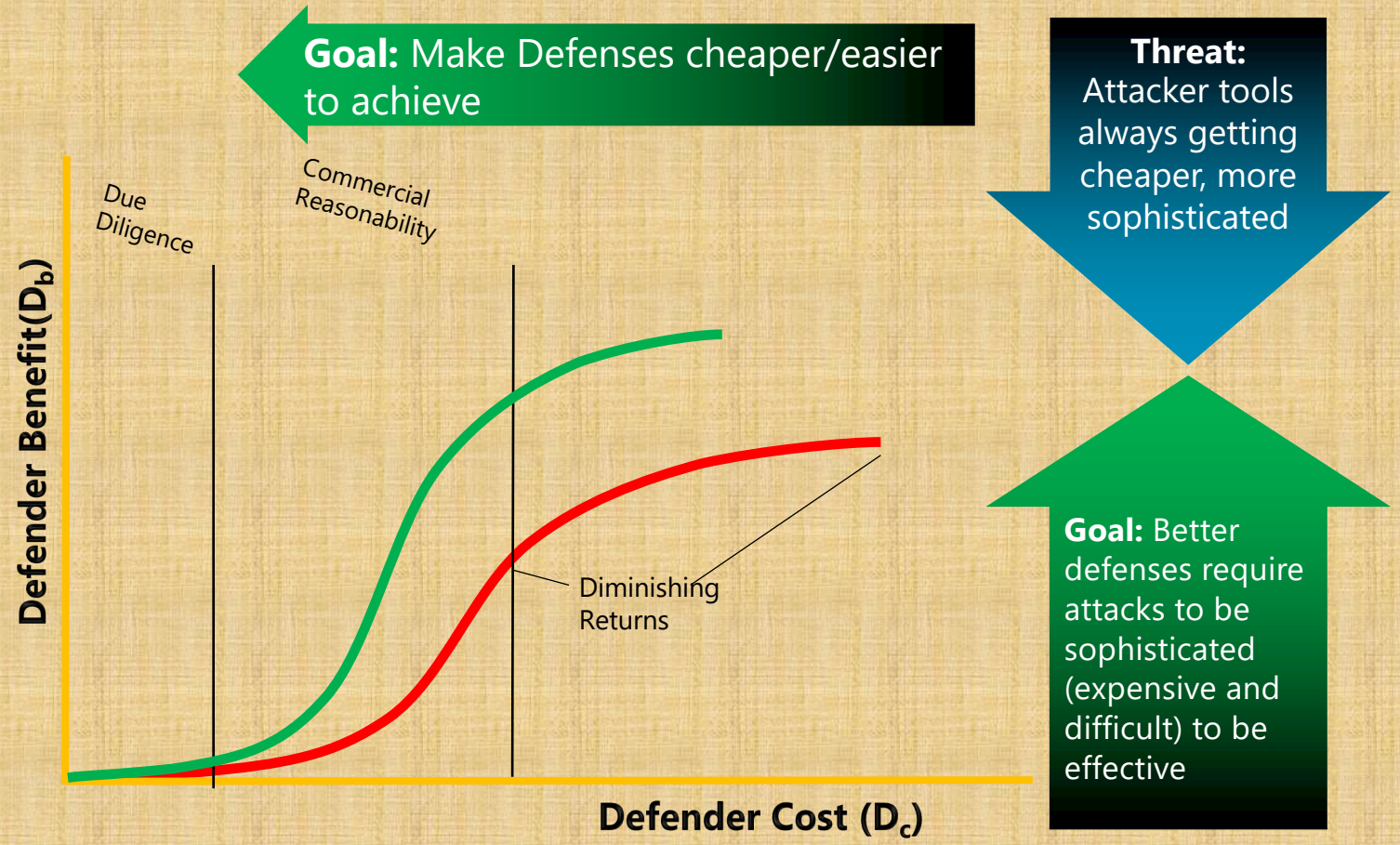
… the cost of fixing defects after deployment is almost fifteen times greater than detecting and eliminating them during development



Source IDC and IBM Systems Sciences Institute

Author: Nho Luong
Skill: DevOps Engineer Lead

Cybersecurity Economics



Author: Nho Luong
Skill: DevOps Engineer Lead

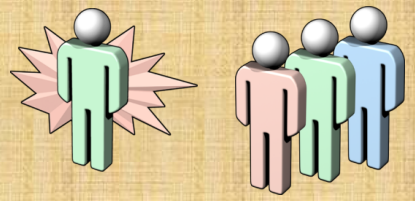
Ok So Why is Software Security poor?

- Security is seen as something that gets in the way of software functionality.
- Security is difficult to assess and quantify.
- Security is often not a primary skill or interest of software developers.
- Time spent on security is time not spent on adding new and interesting functionality.

- We have (more than enough) security technologies

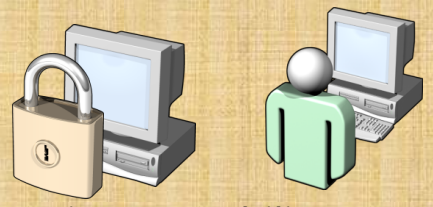
Author: Nho Luong
Skill: DevOps Engineer Lead

The What ? Theory - Summary



Attack v. Defend

- Attacker needs only one vulnerability
- Defender needs to secure all entry points
- Attackers have unlimited time
- Defender has time and cost constraints



Security vs. Usability

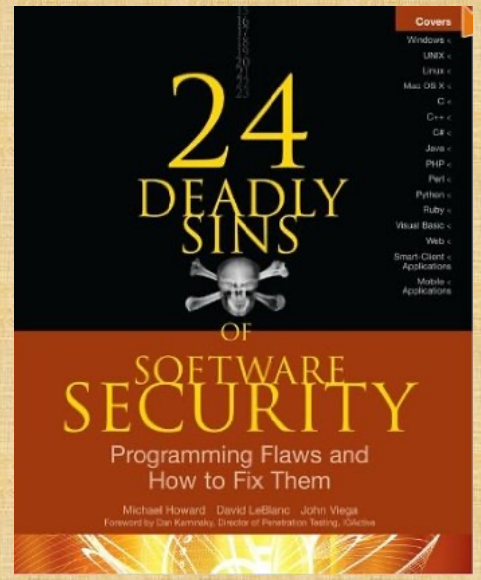
- Secure systems are more difficult to use
- Complexity is difficult to remember
- Users prefer simplicity and use

An Afterthought

- Management doesn't see business value
- Developers find it makes more work

- But addressing security just before release is **very** expensive

Author: Nho Luong
Skill: DevOps Engineer Lead



The What?

Classic Sins

Author: Nho Luong
Skill: DevOps Engineer Lead

Security Sins

- Sin 0 Complacency
- Sin 0.1 My Language/Run Time isn’t susceptible

Author: Nho Luong
Skill: DevOps Engineer Lead

Sin 1 : Buffer Overruns

- Occurs when a program allows input to write beyond the end of the allocated buffer
  - Program might crash or allow attacker to gain control
  - Still possible in languages like C#, Java since they use libraries written in C/C++ but more unlikely
  - Cause is failing to validate input
  - Can occur on stacks and heaps

Author: Nho Luong
Skill: DevOps Engineer Lead

Common Buffer Overflow Myths

- **Myth #1**: Buffer overflows only affect Microsoft platforms
- **Myth #2**: Buffer overflows cannot exist in managed languages (e.g. .NET and Java)
- **Myth #3**: Buffer overflows cannot be exploited on application heaps

Author: Nho Luong
Skill: DevOps Engineer Lead

Sin 4 : SQL Injection

- Takes user input
- Does not check user input for validity
- Uses user input data to query a database
- Uses string concatenation or string replacement to build the SQL query or uses the SQL Exec command

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;

SELECT * FROM Users WHERE UserId = 105 or 1=1
```

Author: Nho Luong
Skill: DevOps Engineer Lead

Common SQL Injection Myths

1. SQL injection applies only to applications that use Microsoft SQL Server
2. SQL injection applies only to Web-based applications
3. SQL injection can be remedied by using transport security protocols, such as SSL and IPSec
4. SQL injection only applies to non-authenticated users

Author: Nho Luong
Skill: DevOps Engineer Lead

Preventing SQL Injection

- Validate input
- Use SQL parameterized queries
- Use stored procedures
  - Do not use “exec @sql” construct
- Use SQL Execute-Only permission

```
strSQL=“SELECT IsAdmin from tblUSERS WHERE Username=@username AND password=@password”;
SQLCommand cmd= new SQLCommand(strSQL,connection);
cmd.Parameters.AddWithValue(“@username”,username);
cmd.Parameters.AddWithValue(“@password”,password);
```

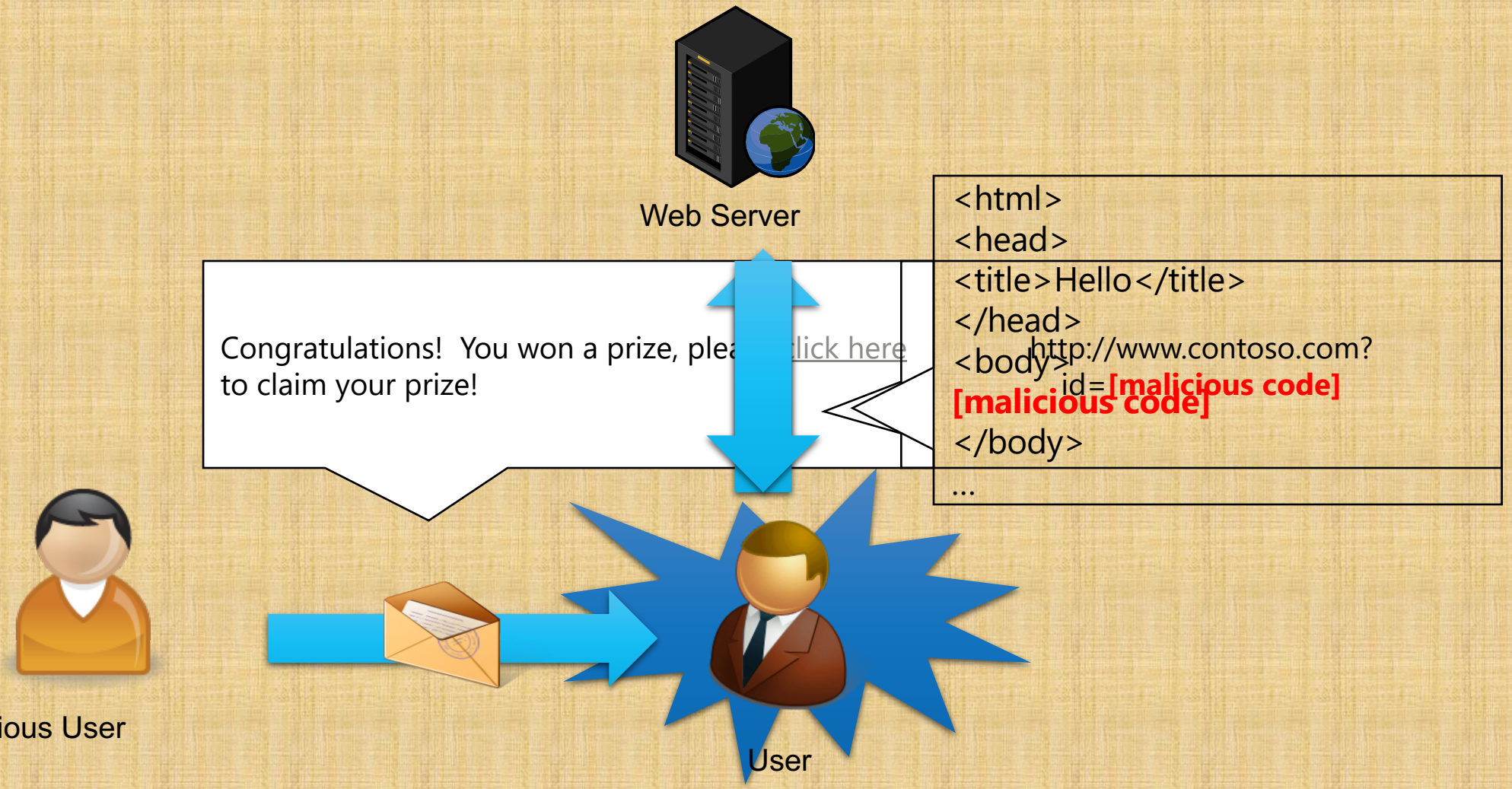
Author: Nho Luong
Skill: DevOps Engineer Lead

Sin 7 : Cross Site Scripting

- Somewhat misnamed, as crossing sites is not always necessary to exploit this bug
- In a nutshell
  - Web app takes input from a user
  - Input is stored or echoed back to the user
  - That’s it

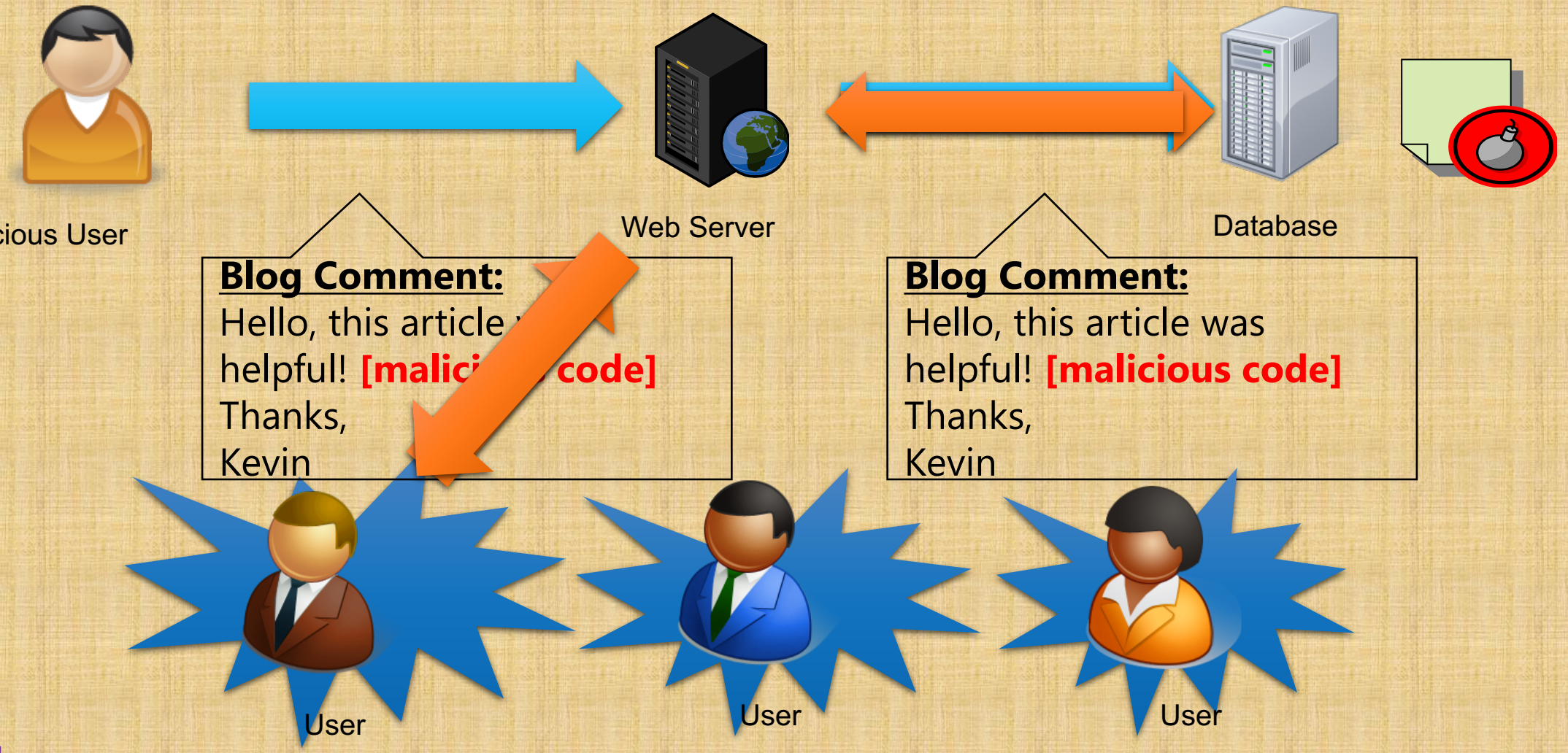
Author: Nho Luong
Skill: DevOps Engineer Lead

Type 1: Non-Persistent Cross-Site Scripting



Author: Nho Luong
Skill: DevOps Engineer Lead

Type 2: Persistent Cross-Site Scripting



Author: Nho Luong
Skill: DevOps Engineer Lead

Common Cross-Site Scripting Myths

- **Myth** - Cross-site scripting applies only to Web-based applications built on Microsoft technologies
- **Myth** - Cross-site scripting can be remedied by using transport security protocols, such as SSL and IPsec

Author: Nho Luong
Skill: DevOps Engineer Lead

Sin 13 : Information Leakage

- Attacker gets information, implicitly or explicitly, that could provide more information for the attacker to reach their goal
- Examples:
  - Name of server software, versions
  - Debugging information (e.g. left on in PHP)
  - Error messages that reveal code structure

Author: Nho Luong
Skill: DevOps Engineer Lead

How?

Threat Modelling

Author: Nho Luong
Skill: DevOps Engineer Lead

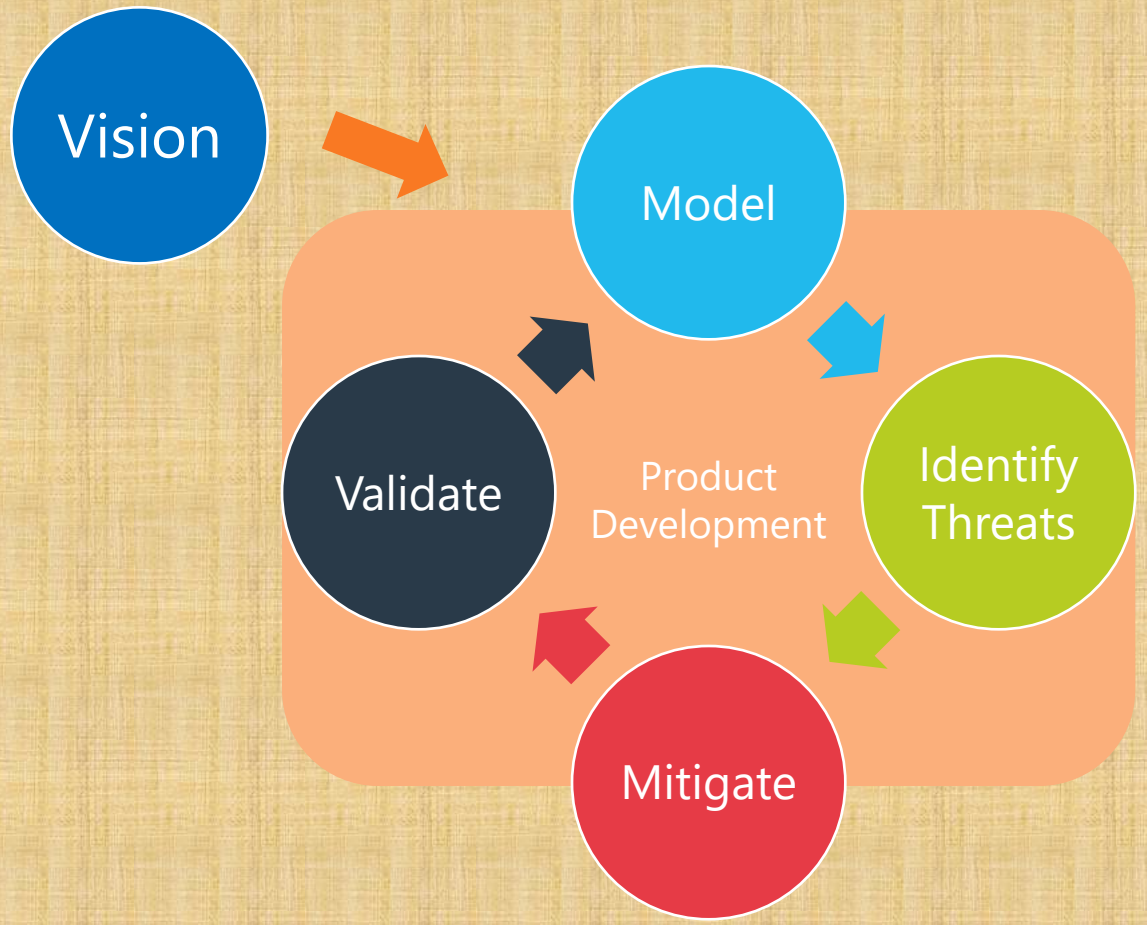
Terms

- **Threats**: What a malicious user may attempt in order to compromise a system
- **Vulnerability**: A defect in software that allows security policy to be violated.
- **Exploit**: A program that exercises a vulnerability.

- Threats and vulnerabilities are not the same thing:

Author: Nho Luong
Skill: DevOps Engineer Lead

Threat Modeling In a Nutshell

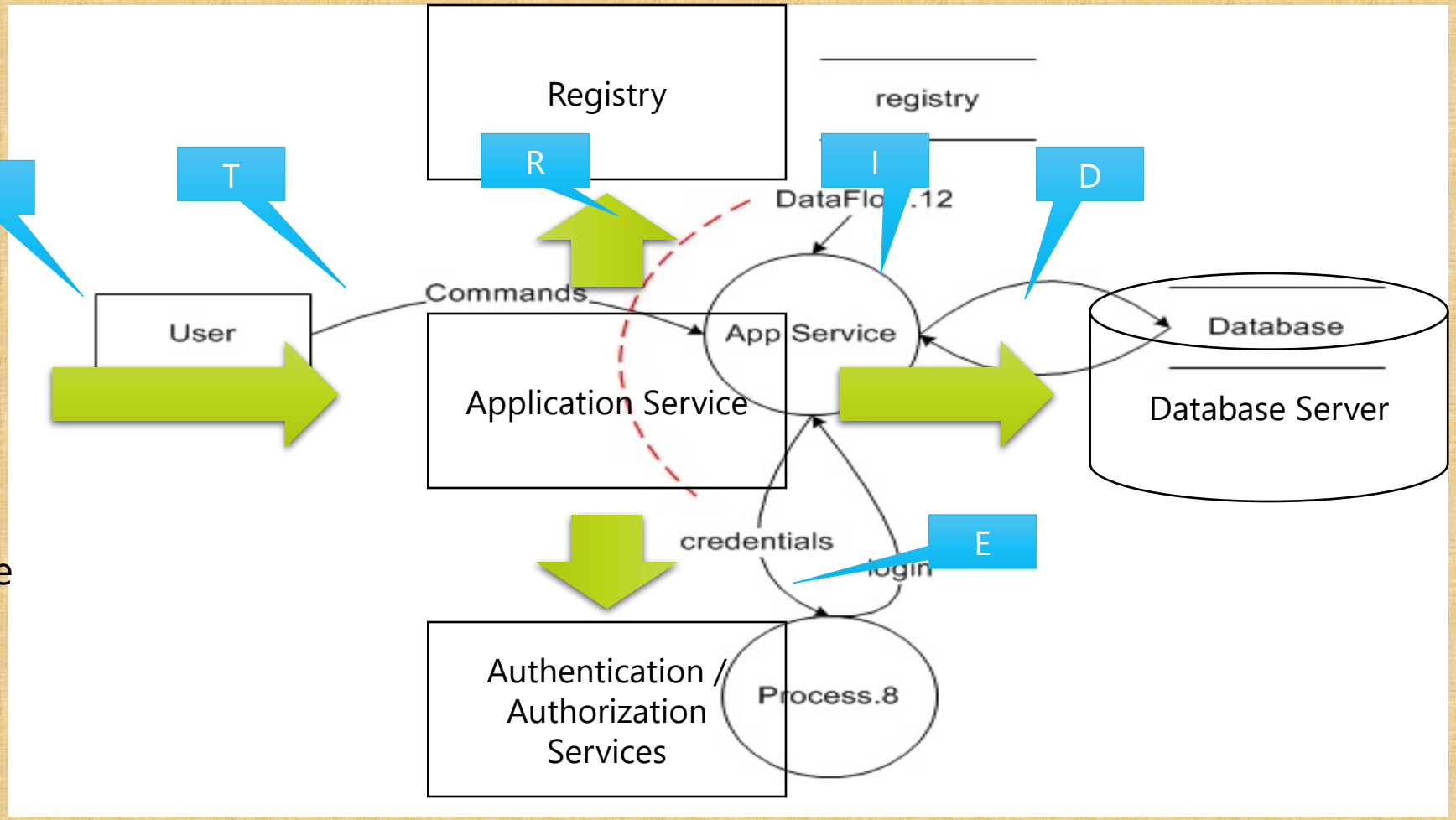


Author: Nho Luong
Skill: DevOps Engineer Lead

STRIDE Threat Types

Desired Property	Threat	Definition
Authentication	**S**poofing	Impersonating something or someone else
Integrity	**T**ampering	Modifying code or data without authorization
Non-repudiation	**R**epudiation	The ability to claim to have not performed some action against an application
Confidentiality	**I**nformation Disclosure	The exposure of information to unauthorized users
Availability	**D**enial of Service	The ability to deny or degrade a service to legitimate users
Authorization	**E**levation of Privilege	The ability of a user to elevate their privileges with an application without authorization

Threat Modeling Illustrated



Spoofing
Tampering
Repudiation
Information Disclosure
Denial of Service
Elevation of Privilege

Author: Nho Luong
Skill: DevOps Engineer Lead

Use DREAD to Prioritise Threats

	Definition
Damage Potential	How great can the damage be?
Reproducibility	How easy is it to get the potential attack to work?
Exploitability	How much effort and experience is required to mount the attack?
Affected Users	If the threat were exploited, how many users will be effected?
Discoverability	The probability the threat will be found

$Risk_{DREAD}$: (D + R + E + A + D) / 5

Author: Nho Luong
Skill: DevOps Engineer Lead

How? Threat Modelling - Summary

- Perform regular threat modelling
- Use a consistent process
- Prioritise actions based on the business view to risk

Author: Nho Luong
Skill: DevOps Engineer Lead

How?

Mitigations

Author: Nho Luong
Skill: DevOps Engineer Lead

Mitigations

Approaches to threat mitigation (in order of preference):

- Redesign
- Use standard mitigations
- Use unique mitigations
- Accept risk in accordance with policies

Author: Nho Luong
Skill: DevOps Engineer Lead

Examples of Standard Mitigations

Threat	Example Standard Mitigations
Spoofing	IPsec Digital signatures Message authentication codes Hashes
Tampering	ACLs Digital signatures Message Authentication Codes
Repudiation	Strong Authentication Secure logging and auditing
Information Disclosure	Encryption ACLs
Denial of Service	ACLs Quotas High availability designs
Elevation of Privilege	ACLs Group or role membership Input validation

Author: Nho Luong
Skill: DevOps Engineer Lead

Attack Surface Reduction

- **Attack Surface**: Any part of an application that is accessible by a human or another program
  - Each one of these can be potentially exploited by a malicious user
- **Attack Surface Reduction**: Minimize the number of exposed attack surface points a malicious user can discover and attempt to exploit

Author: Nho Luong
Skill: DevOps Engineer Lead

Defense In Depth

- Assume that software and hardware will fail at some point
  - Trusted applications: security and privacy features and mechanisms
- To many applications today can be compromised when single, and often only, layer of defense is breached (firewall)
- If one defense layer is breached, what other defense layers (if any) provide additional protection to the application?

Author: Nho Luong
Skill: DevOps Engineer Lead

Least Privilege

- Assume that all applications can and will be compromised
- If an application is compromised, then the potential damage that the malicious person can inflict is contained and minimized accordingly

Author: Nho Luong
Skill: DevOps Engineer Lead

Secure Defaults

- Deploy applications in more secure configurations by default.
- Helps to better ensure that customers get safer experience with your application out of the box, not after extensive configuration
- It is up to the user to reduce security and privacy levels

Author: Nho Luong
Skill: DevOps Engineer Lead

Input Validation Tips

- “All input is evil, until proven otherwise”
- Validate input that crosses trust boundaries
- Validate inputs against expected data:
  - Format
  - Length
  - Type
  - Range

Author: Nho Luong
Skill: DevOps Engineer Lead

Compiler Defenses

- Run-time defenses mostly from buffer overflows and other malicious attempts to control execution flow
- Defensive code is automatically added by compiler, not developer
- Compiler defenses do not fix security vulnerabilities
  - Should not be considered a “silver bullet”

Author: Nho Luong
Skill: DevOps Engineer Lead

Banned APIs

- Incorrect use can lead to serious vulnerabilities
- Effective way to help reduce risk
- The Microsoft SDL Banned APIs list is not a static list
- Also applicable to non-Microsoft platforms
- Useful when source code is available

https://msdn.microsoft.com/en-us/library/bb288454.aspx

Author: Nho Luong
Skill: DevOps Engineer Lead

Not All Cryptographic Standards Are Safe!

- Banned by Microsoft SDL for all new applications:
  - MD4, MD5, RC4, DES, SHA1

- Approved by Microsoft SDL for all new applications:
  - Advanced Encryption Standard (AES) or Rijndael
  - SHA 256 or higher

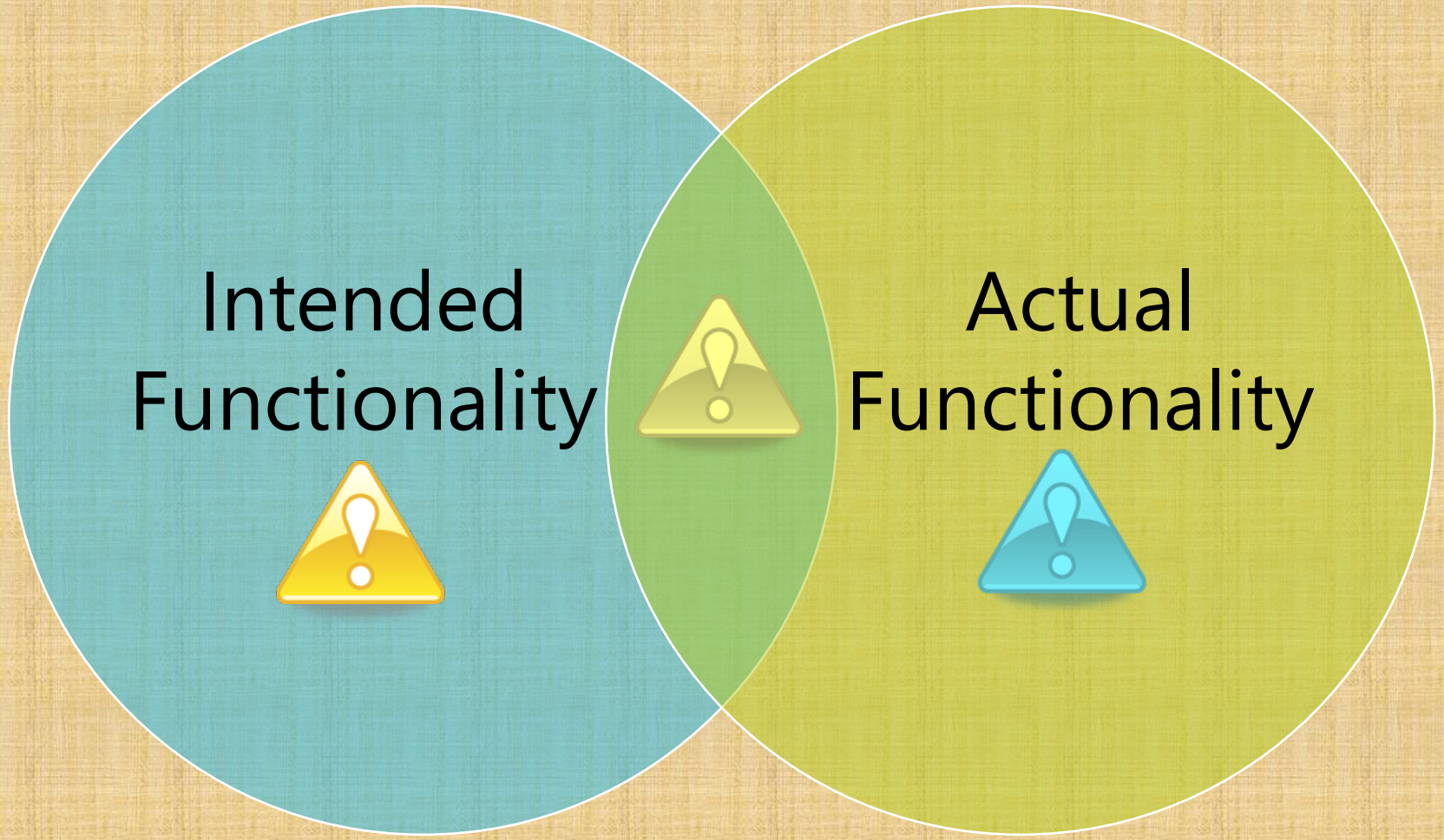
Author: Nho Luong
Skill: DevOps Engineer Lead

How?

Testing and Analysis

Author: Nho Luong
Skill: DevOps Engineer Lead

Functional Testing vs. Secure Testing



Author: Nho Luong
Skill: DevOps Engineer Lead

How Experts view Secure Testing

"The symptoms of security vulnerabilities are very different from those of traditional bugs"

"How To Break Software Security"

"The designers and the specifications might outline a secure design, the developers might be diligent and write secure code, but it's the testing process that determines whether the product is secure in the real world"

"Writing Secure Code, Second Edition"

Author: Nho Luong
Skill: DevOps Engineer Lead

Security Testing Tips

- Single Rule of Security Testing: there are no rules!
- “Malicious users wouldn’t try to do that” - Yes they will!
- Malicious users will try to circumvent application client components
- Malicious users will try to compromise application dependencies

Author: Nho Luong
Skill: DevOps Engineer Lead

Penetration Testing

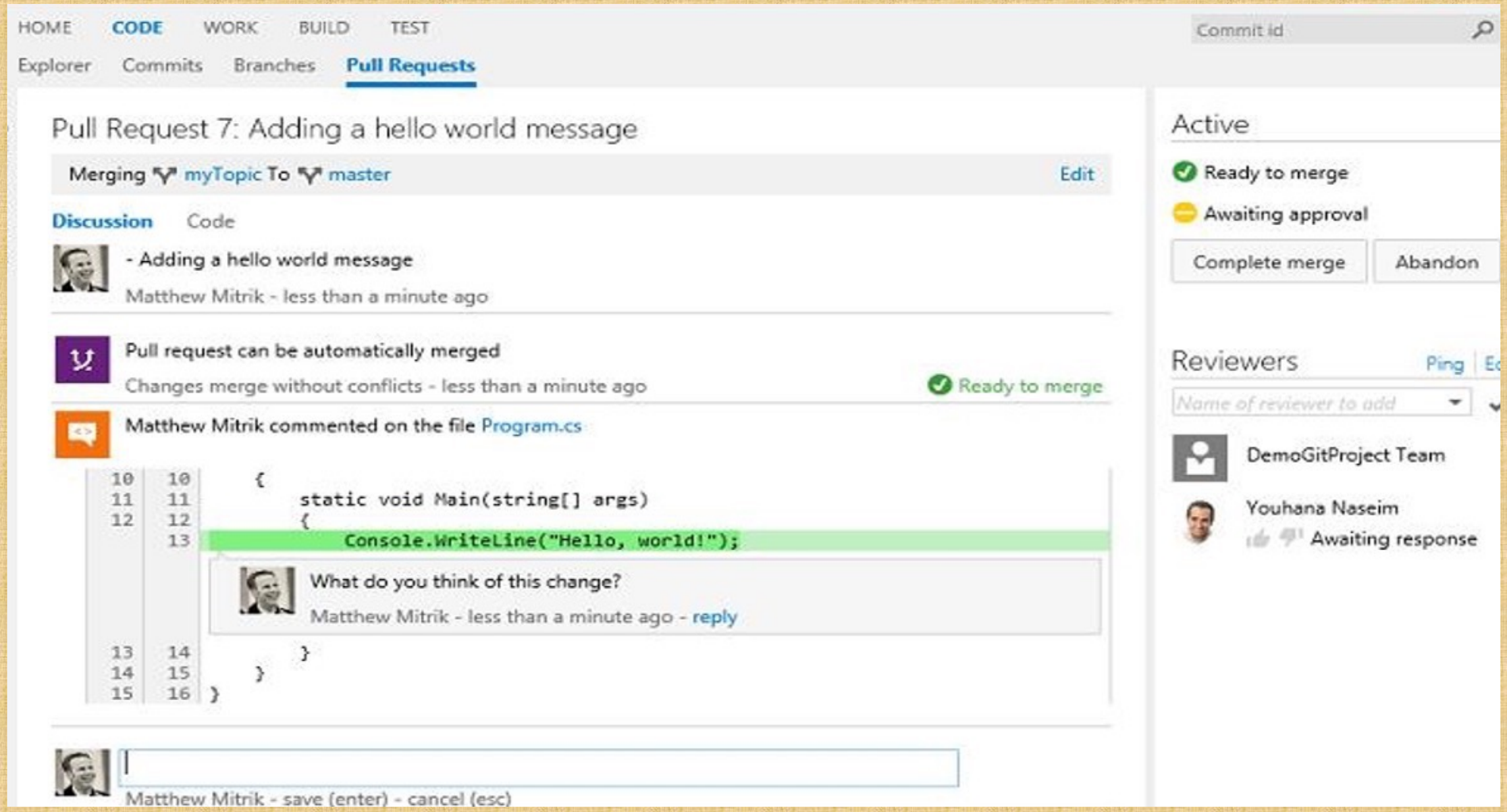
- Goal is to simulate actual attacks against an application and measure how well the application is able to withstand such attacks
- Attack the application as a malicious user by using:
  - Manual techniques
  - Attack tools
- Penetration testing provides a security posture snapshot only at the specific time of testing

Author: Nho Luong
Skill: DevOps Engineer Lead

Code Review

- Manual review of the source code of an application to identify common code weaknesses
- Context-aware analysis
- Should be performed for all high-priority code
- Very effective, but also very labor-intensive
- Automation tools to assist code reviewers are available

Author: Nho Luong
Skill: DevOps Engineer Lead



Author: Nho Luong
Skill: DevOps Engineer Lead

Release Management

- Make sure settings and configurations are correctly secured.
- Use automation where possible
- Have an incident response plan ready
- Keep a release archive

Author: Nho Luong
Skill: DevOps Engineer Lead

Incident Response

- Minimize response time for an active incident to the time to execute response plan
- Update planning with a retrospective
  - What worked?
  - What didn't work?
  - What should we do next time?

- THIS IS TOO OFTEN FORGOTTEN…

Author: Nho Luong
Skill: DevOps Engineer Lead

Implementing SDL in VSTS

Author: Nho Luong
Skill: DevOps Engineer Lead

Is there a TFS SDL Template?

- No, there isn't one….
- You need to grow your own. Key features will be…
  - A means to mark issues/features as security related
  - Potentially automated work item creation on events, or tools to generate work upon demand e.g. start of sprint
    - Thread model session for new features
    - Secure Code reviews
  - Static analysis in automated build

Author: Nho Luong
Skill: DevOps Engineer Lead

The first layer - Visual Studio

- Use the built in tools for the first layer of protection
  - Unit-Testing
  - Code Analysis (FxCop)
  - StyleCop
  - SonarLint (VS 2015 and later)

Author: Nho Luong
Skill: DevOps Engineer Lead

The second layer – VSTS/TFS

- Combined Visual Studio with VSTS or TFS
  - Continuous Integration
  - Code Review via Pull Requests if using Git
  - Gated Checkin if using TFVC
  - SonarQube integration

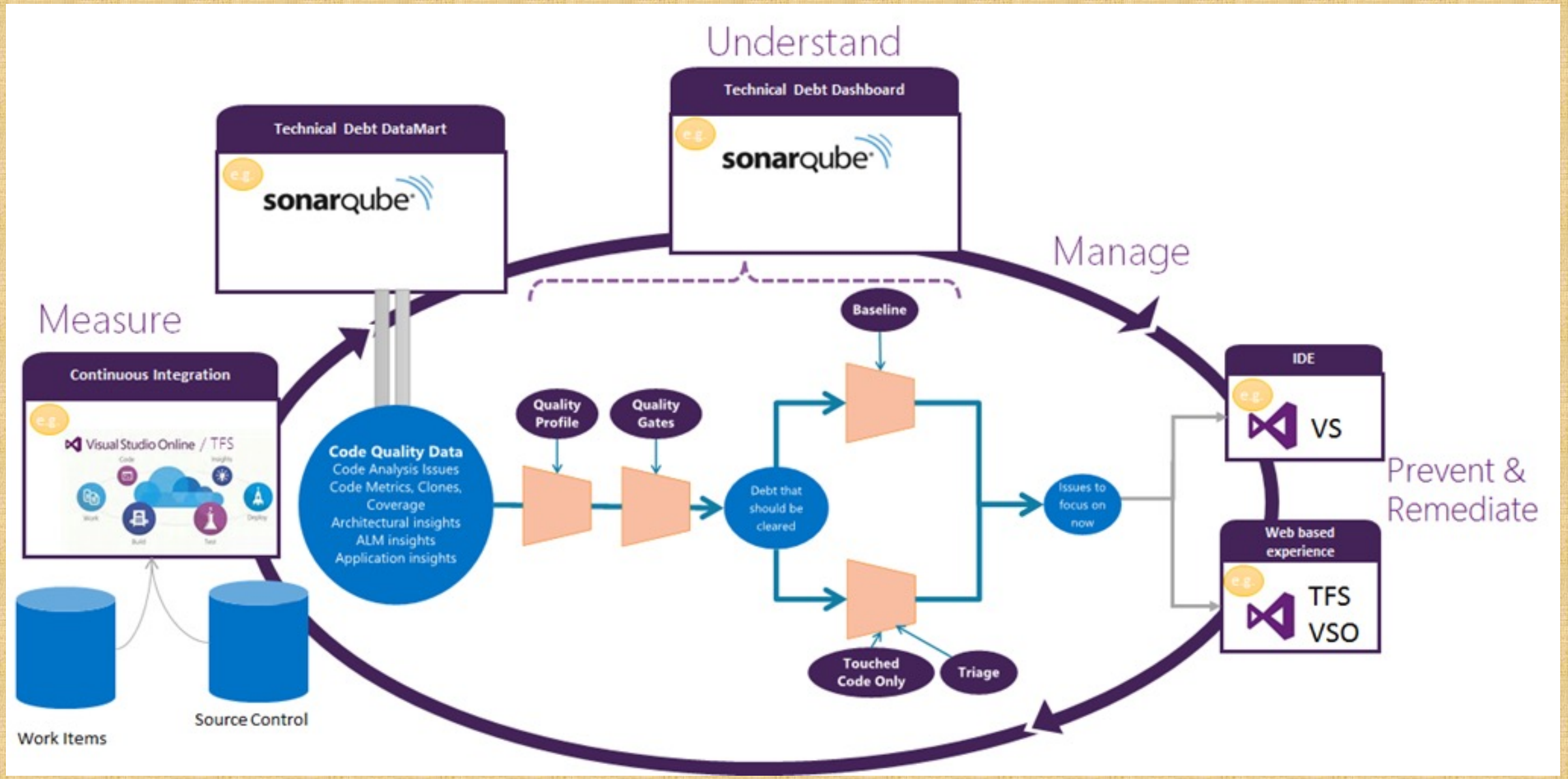
Author: Nho Luong
Skill: DevOps Engineer Lead

Introducing SonarQube

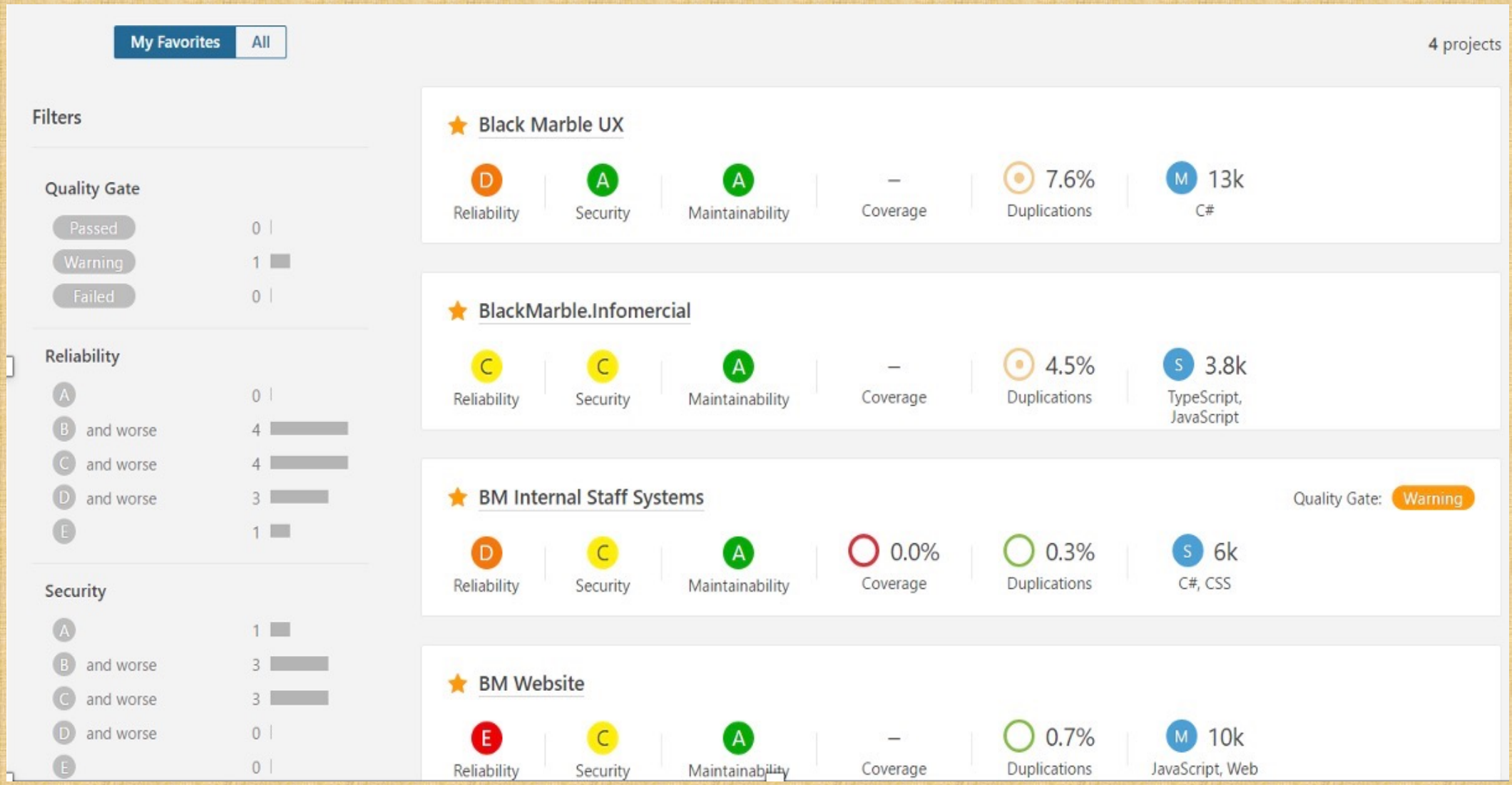
To quote the product homepage…

"The SonarQube® platform is an open source quality management platform, dedicated to continuously analyzing and measuring the technical quality of source code, from project portfolio down to the method level. To understand what the platform tracks and why it's important, take a look at the Developers' Seven Deadly Sins."

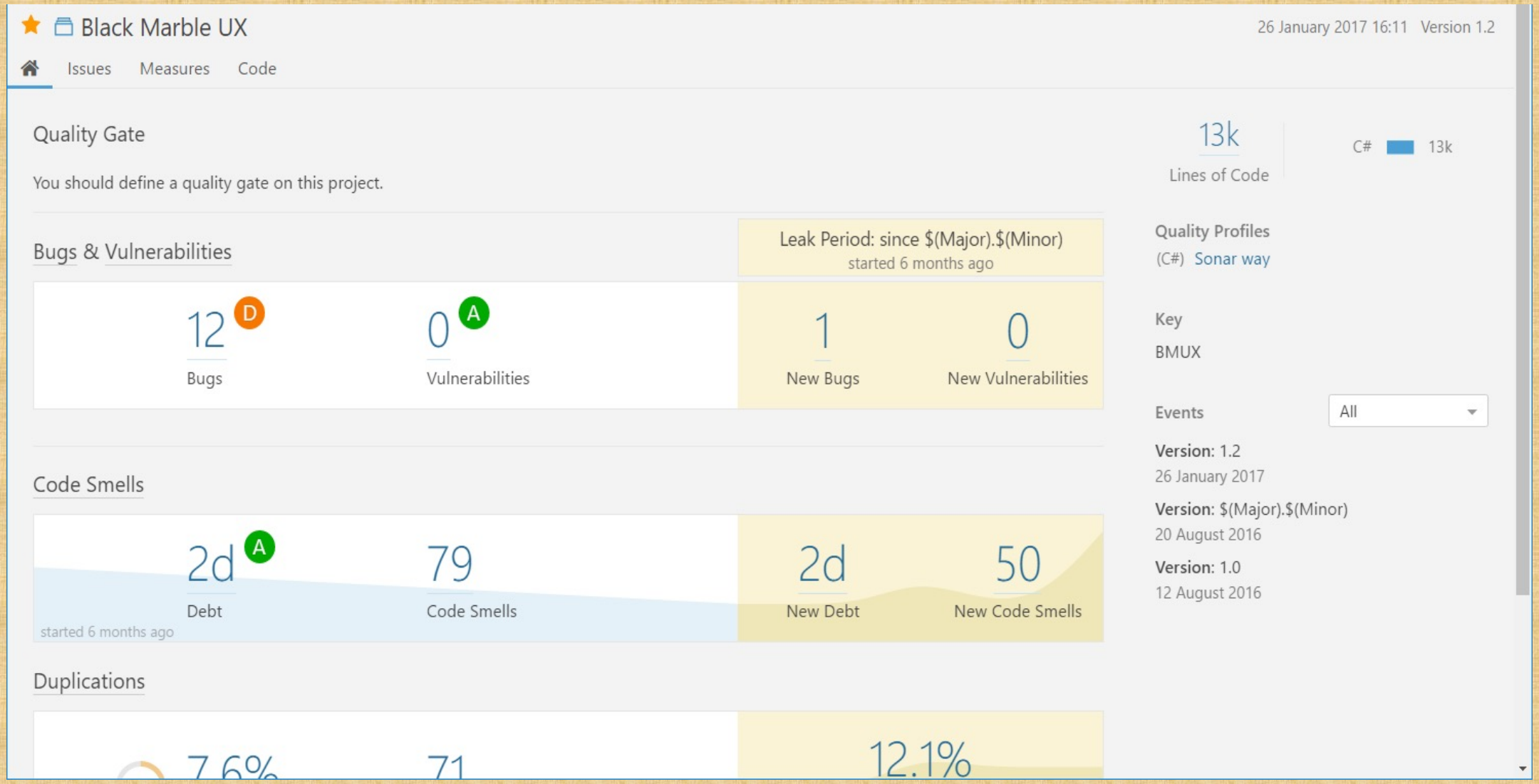
Author: Nho Luong
Skill: DevOps Engineer Lead



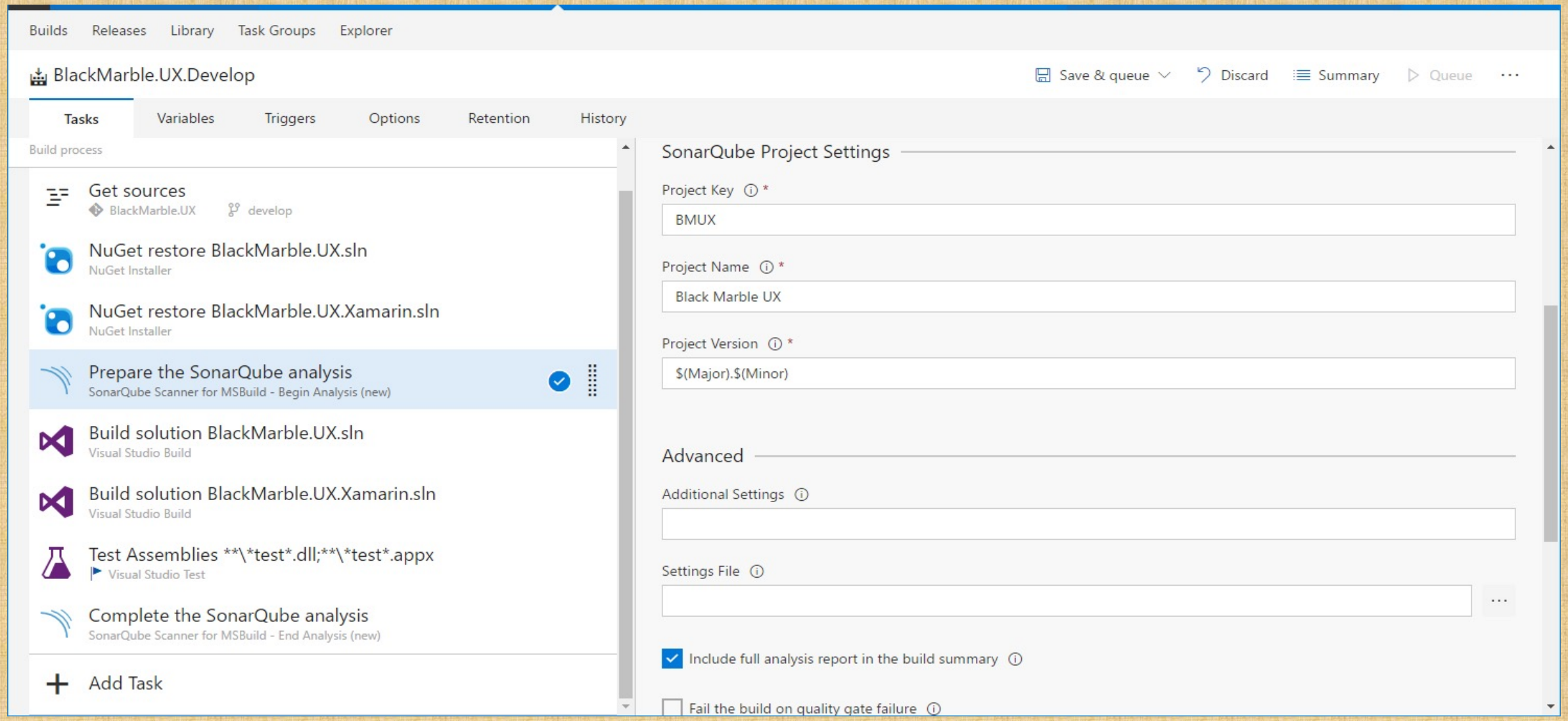
Author: Nho Luong
Skill: DevOps Engineer Lead



Author: Nho Luong
Skill: DevOps Engineer Lead



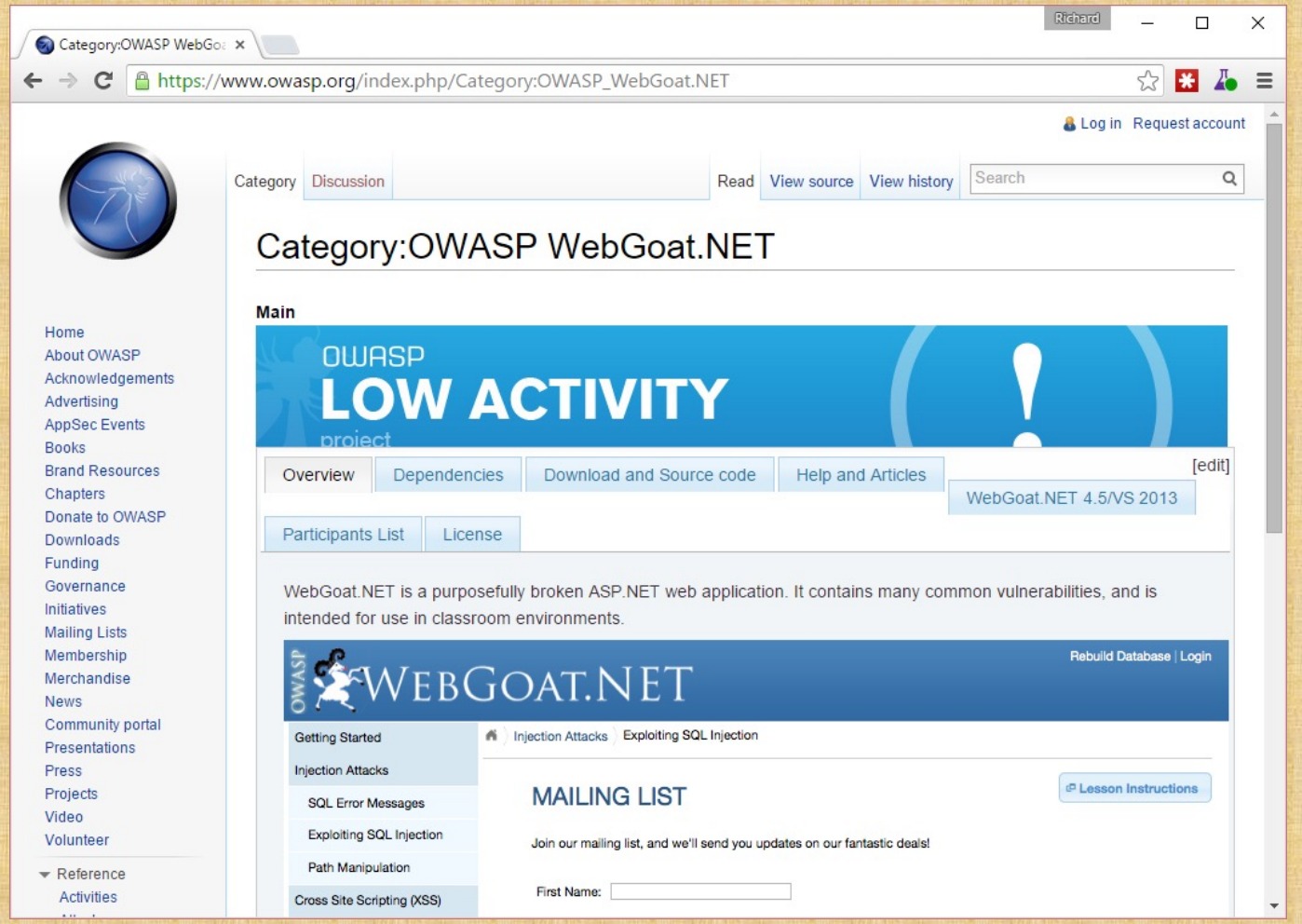
Author: Nho Luong
Skill: DevOps Engineer Lead



Author: Nho Luong
Skill: DevOps Engineer Lead

Whatever your process you need to test it

- Have a sample that tests your process find issues
- OWASP Open Web Application Security Project - WebGoat Bad Bank
- Useful to provide processes during audit



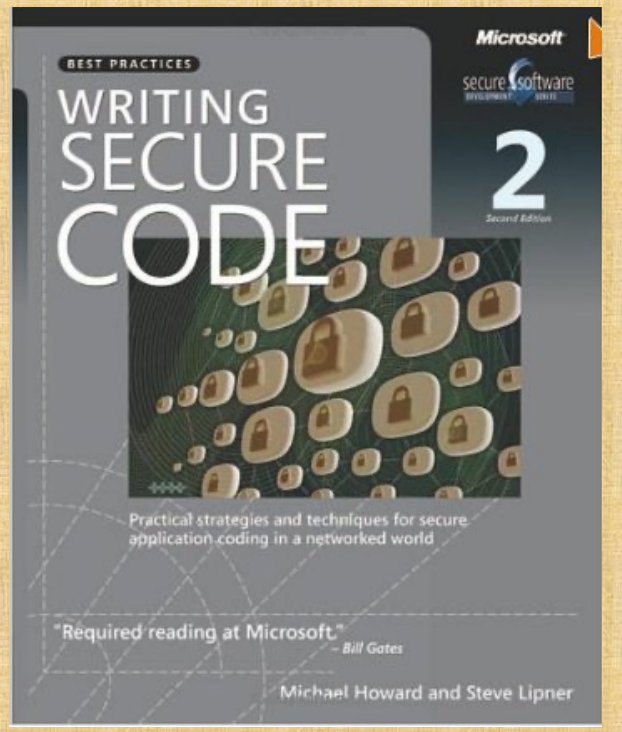
Author: Nho Luong
Skill: DevOps Engineer Lead

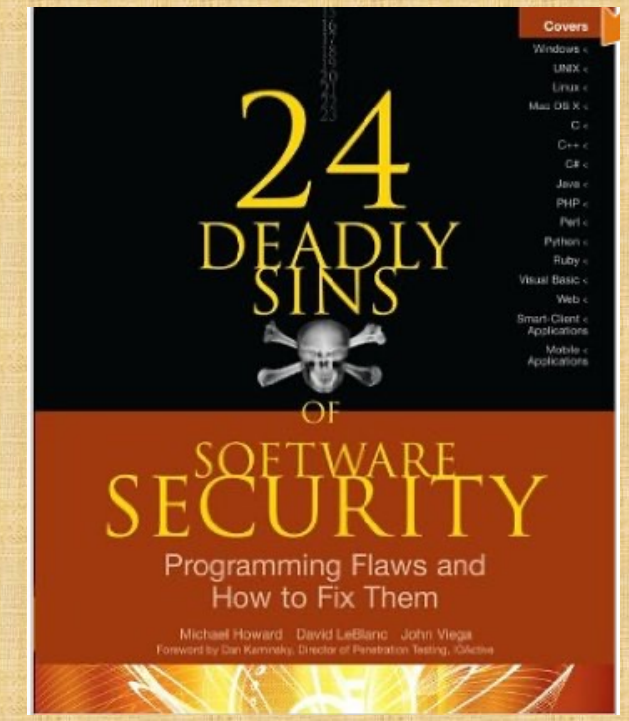
Summary

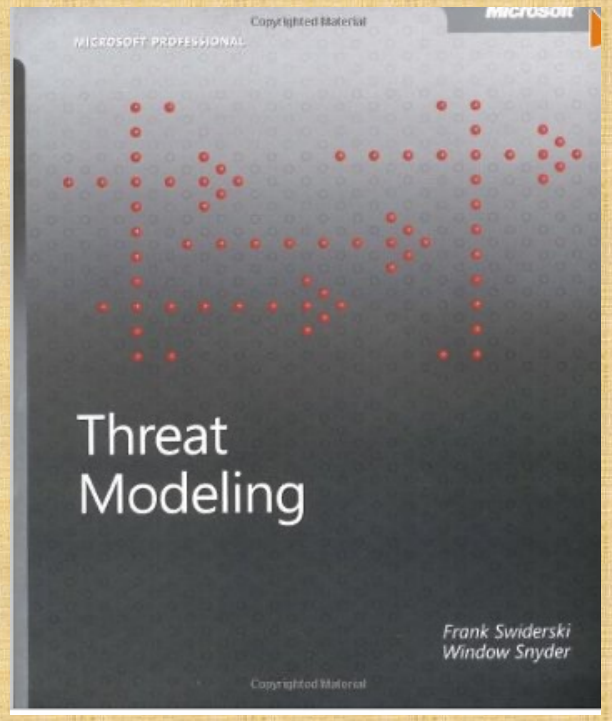
- If you wait until an application is already in production to make it secure, you are too late
- Develop a thoroughly considered policy exception tracking process
- Education is crucial to the success of a security program
- Security is an ongoing, always changing concern
- Threat model your work!
  - Start early
  - Track changes

Author: Nho Luong
Skill: DevOps Engineer Lead

Books to read







Author: Nho Luong
Skill: DevOps Engineer Lead

Resources

- Open Web Application Security Project (OWASP)
  - https://www.owasp.org
- SonarQube
  - http://www.sonarqube.org
- Microsoft ‘Writing Secure Code’
  - https://msdn.microsoft.com/en-us/security/aa570401.aspx
- Microsoft ‘The Trustworthy Computing Security Development Lifecycle’
  - https://msdn.microsoft.com/en-us/library/ms995349.aspx

Author: Nho Luong
Skill: DevOps Engineer Lead

Call to Action

Black Marble SDLC Consultancy

- Bespoke On-site Workshops:
  Assess your current situation and needs
- Technology Demonstrator:
  Deploy the tools for you to use

Ongoing Consultany:
Automate your own projects and build best-practice

**Length varies based on scale and ALM/DevOps maturity*

Author: Nho Luong
Skill: DevOps Engineer Lead

Thank You

Author: Nho Luong
Skill: DevOps Engineer Lead