Secure Software Design and Engineering

Introduction

Dr. Zubair Ahmad

Life with Cybersecurity??

About me!

Zubair Ahmad

Education

- Ph.D. in Computer Science (Cyber Security) - University of Venice Italy (2020-2024)
- Visiting Scholar - CISPA Helmholtz Center for Information Security Germany
- European Parliament – EU AI Act 2023
- OPLSS Summer School Uni of Oregon and Boston Uni USA 2021

Research Interests

- Web Security and Privacy
- Data Privacy and Protection
- Internet and Web Measurements
- EU Compliance regulations, GDPR
- Internet of Things

More about me -->https://zahmaad.github.io/

Schedule

- **When?**

  - Will share soon on webpage

- **Where?**

  - Here!
  - LH6 NAB

- **What?**

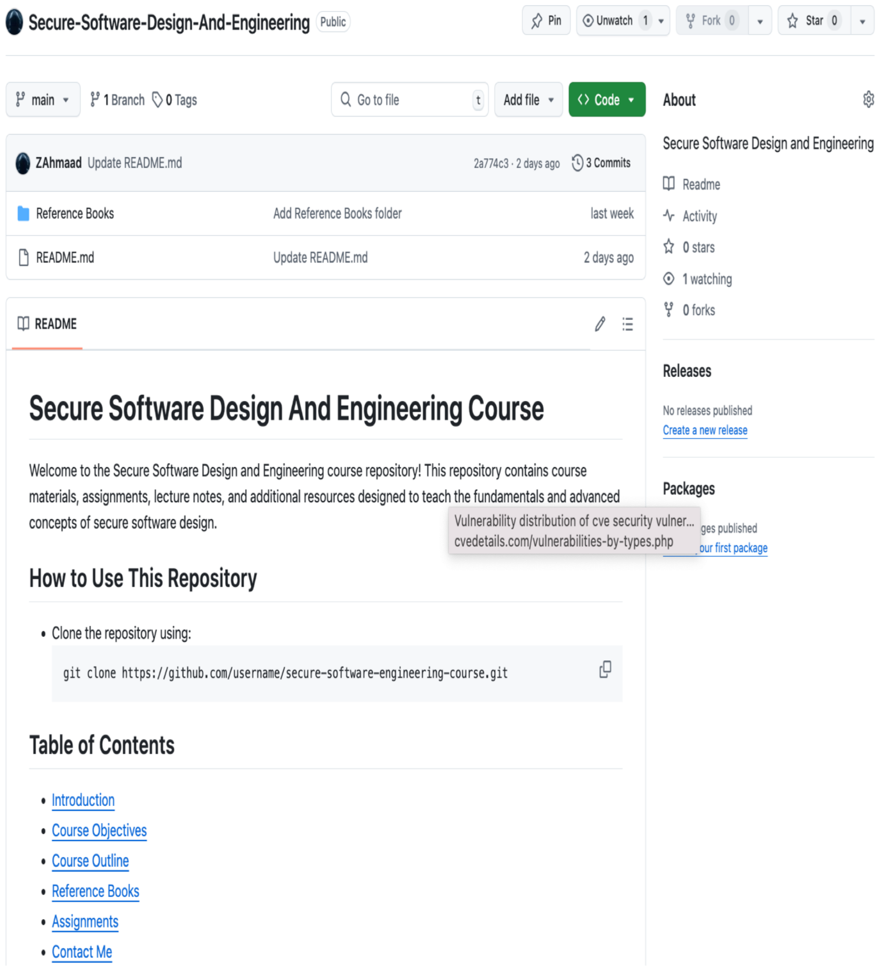
  - Lecturers and exercises
  - Quizzes/Assignments/ Projects
  - Mid/Final Exams

- **Attendance?**

  - Active Attendance
  - **Dead Bodies.**
  - **Active Minds**
  - Mobiles in hands -> Mark as absent
  - 80% mandatory

Webpage

- Lectures/ Slides
- Books
- Assignments/Project
- News
- Labs Material



ZAhmaad Update README.md 2a774c3 · 2 days ago 3 Commits

Reference Books	Add Reference Books folder	last week
README.md	Update README.md	2 days ago

Secure Software Design and Engineering

Secure Software Design And Engineering Course

Welcome to the Secure Software Design and Engineering course repository! This repository contains course materials, assignments, lecture notes, and additional resources designed to teach the fundamentals and advanced concepts of secure software design.

How to Use This Repository

- Clone the repository using:

```
git clone https://github.com/username/secure-software-engineering-course.git
```

Table of Contents

- Introduction
- Course Objectives
- Course Outline
- Reference Books
- Assignments
- Contact Me

https://github.com/ZAhmaad/Secure-Software-Design-And-Engineering

Grading Policy

Assessment Items	Percentage
Quizzes	15%
Assignment/Project	15%
Midterm Exam	30%
Final Exam	40%

Assignments- Project-Quizzes

- A number of assignments/project and quizzes will be taken
- Announced and/or unannounced quizzes

- Github
- Overleaf

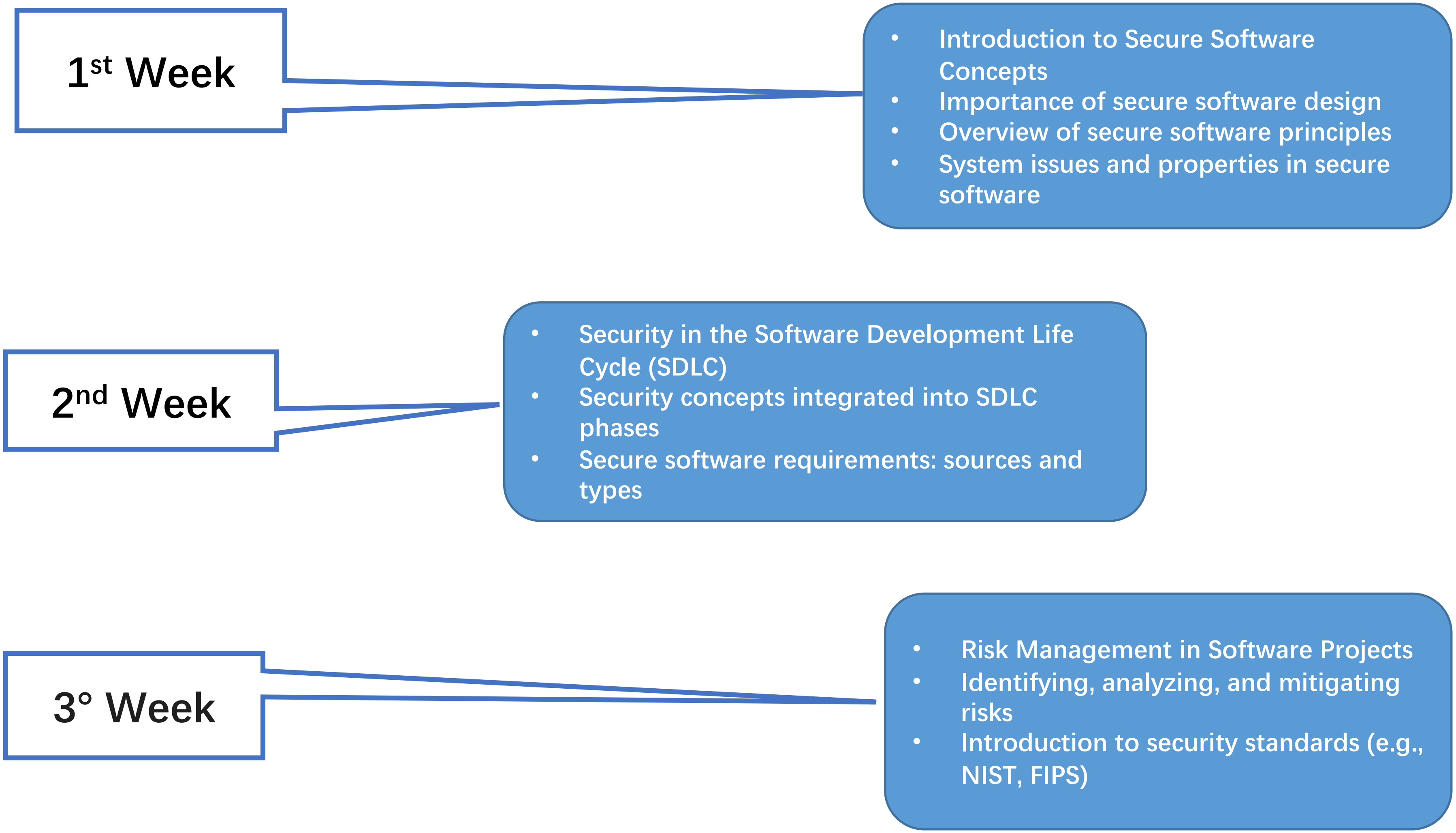
Project/Assignments

- Python
- JavaScript

What Should you expect in this course?

- Secure Software Development Process
- Detect and Mitigate Insecure Programming Practices
- Utilize Software Security Tools
- Design and Conduct Security Testing

What We will learn?



1st Week

- Introduction to Secure Software Concepts
- Importance of secure software design
- Overview of secure software principles
- System issues and properties in secure software

2nd Week

- Security in the Software Development Life Cycle (SDLC)
- Security concepts integrated into SDLC phases
- Secure software requirements: sources and types

3° Week

- Risk Management in Software Projects
- Identifying, analyzing, and mitigating risks
- Introduction to security standards (e.g., NIST, FIPS)

What We will learn?

4th Week

- Security principles and secure design considerations
- Designing secure design principles
- Design processes and best practices (e.g., OWASP guides)
- Security methodologies (e.g., STRIDE, DREAD, OCTAVE)

5th Week

- Secure Software Frameworks
- Introduction to frameworks (e.g., Zachman, COBIT, SABSA)
- Practical examples of applying frameworks

6° Week

- Risk Management in Software Projects
- Identifying, analyzing, and mitigating risks
- Introduction to security standards (e.g., NIST, FIPS)

What We will learn?

7th Week

- Common Software Vulnerabilities
- Overview of vulnerabilities (e.g., buffer overflow, SQL injection)
- Exploitation techniques and impacts

8th Week

- Defensive Coding Practices
- Concepts and techniques for defensive coding
- Avoiding vulnerabilities and polymorphic malware attacks

9° Week

- Secure Software Implementation and Coding
- Best practices for secure coding
- Software development methodologies

What We will learn?

10th Week

- Secure Software Testing
- Testing concepts: functional, non-functional, and security testing.
- Security testing methodologies and tools.
- Static and Dynamic analysis

11th Week

- Software Security Testing and Acceptance
- Software acceptance criteria from a security perspective.
- Testing against common attack vectors

12° Week

- Authentication Protocols
- Common Pitfalls Ways to Analyze Protocols
- Login-only protocols
- Mutual authentication with Key Distribution Center

What We will learn?

13th Week

- Secure Modern Web Development Browser Security Mechanism
- Building blocks for secure modern web applications.
- The future of user authentication on the web

14th Week

- Web security in the real world
- Client-side attacks and defenses
- Server-side attacks and defenses

15° Week

- Emerging trends in secure software engineering (e.g., AI security)
- Recap of major concepts covered
- Final Q&A and discussions

A Quick Starter Scenario (1)

High Value Messaging System

Bank Own Staff

Physical Attack

ATM Machines

Website and Mobile App

A Quick Starter Scenario (2)

Patient record systems - Research

"show me all males born in 1953 who were treated for atrial fibrillation on October 19th 2003" should be enough to target former Prime Minister Tony Blair, who was rushed to hospital that day to be treated for an irregular heartbeat

Safety usability

Safety usability failures are estimated to kill about as many people as road traffic accidents

New technology

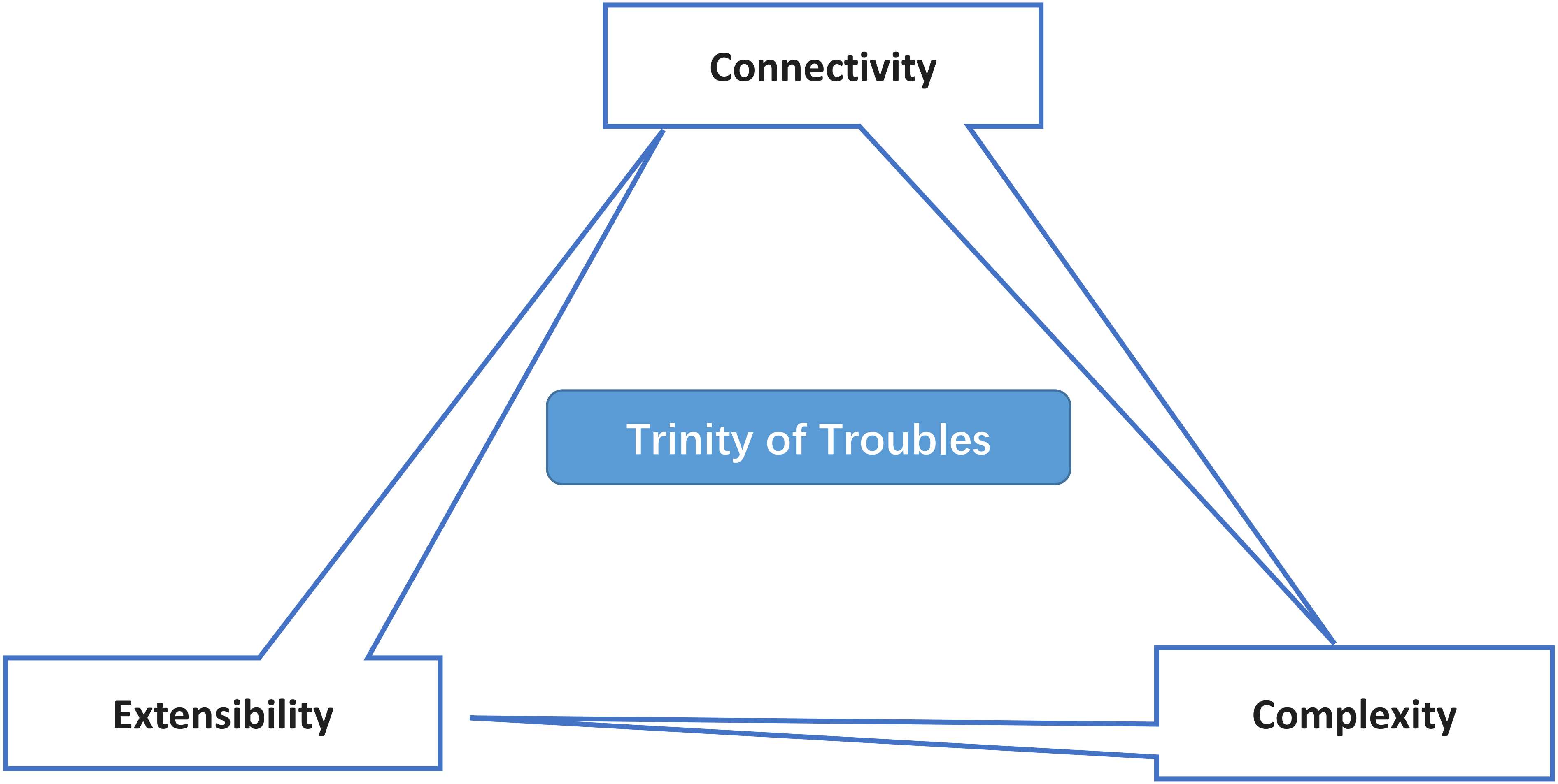
Several hospitals in Britain had machines infected by the Wannacry malware in May 2017, they closed down their networks to limit further infection,

Patient record systems

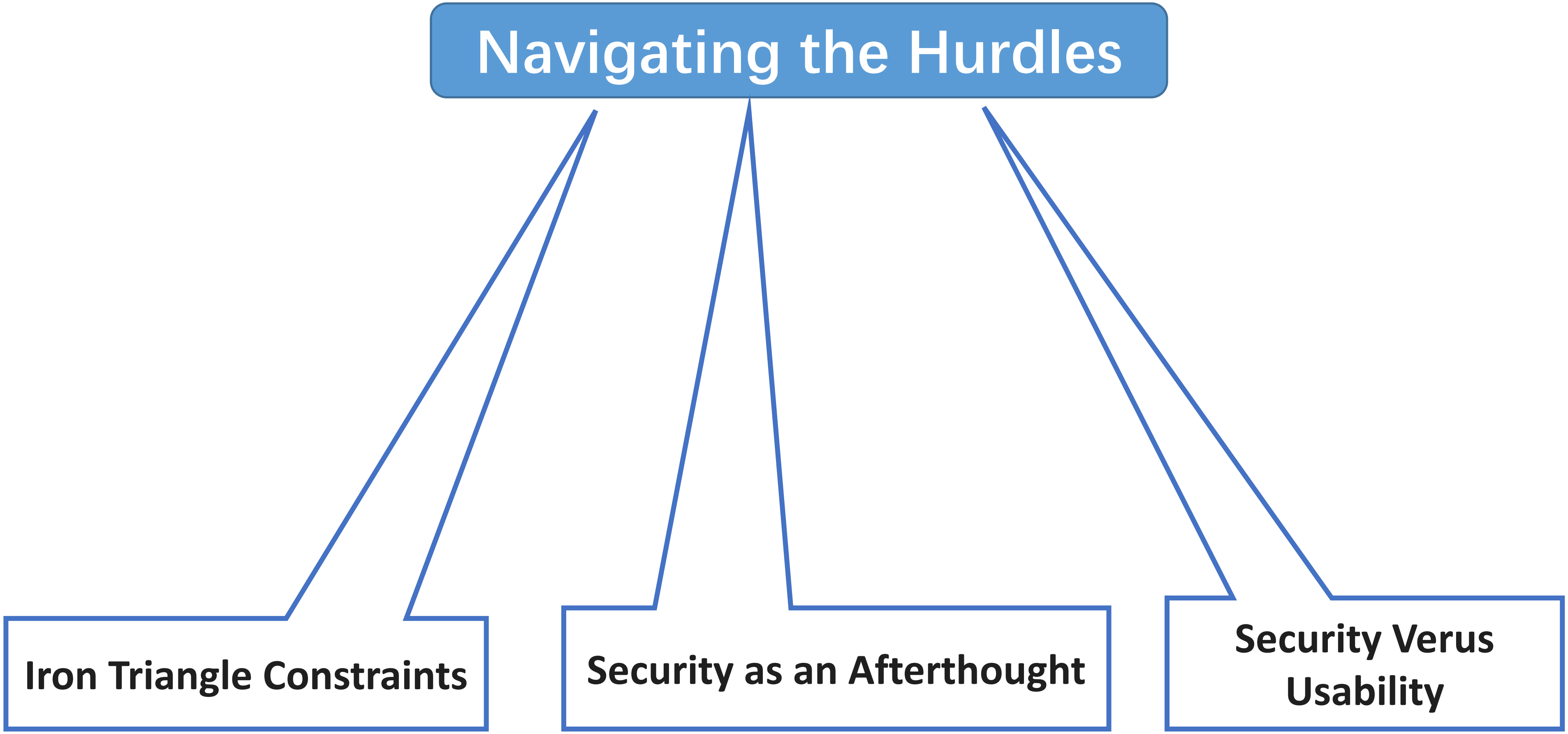
A Quick Starter Scenario (3)

By 2015, President Obama’ s council of advisers on science and technology was predicting that pretty soon every inhabited space on earth would have microphones that were connected to a small number of cloud service providers

Why the problem is growing?



Challenges



Navigating the Hurdles

Iron Triangle Constraints

Security as an Afterthought

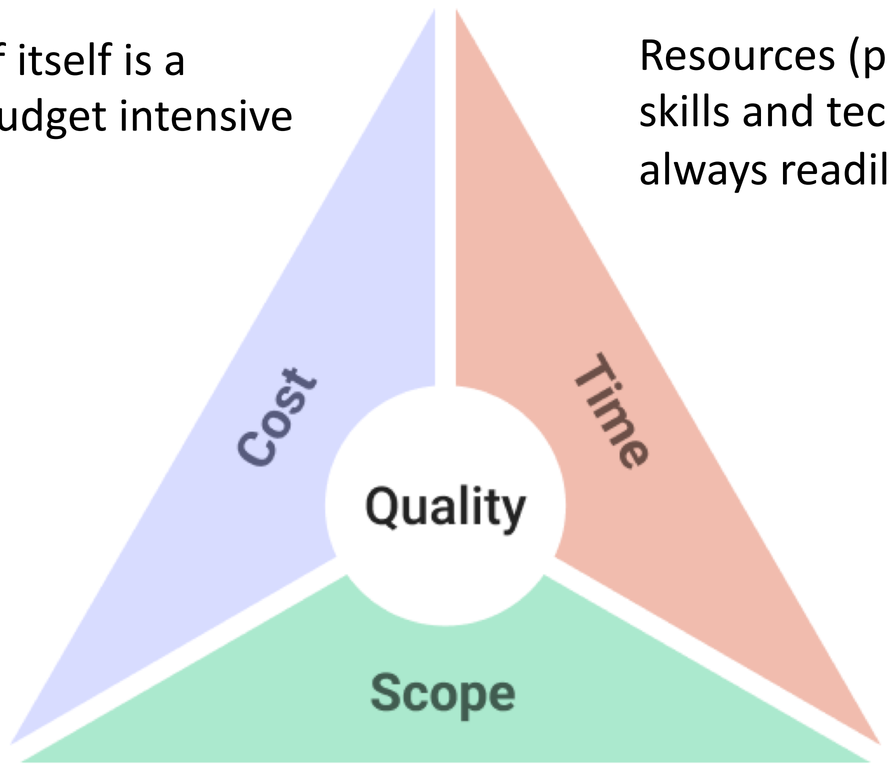
Security Verus Usability

Iron Triangle Constraints

The attacker has the upper hand

Software development in and of itself is a resource, schedule (time) and budget intensive process

Resources (people) with appropriate skills and technical knowledge are not always readily available and are costly.

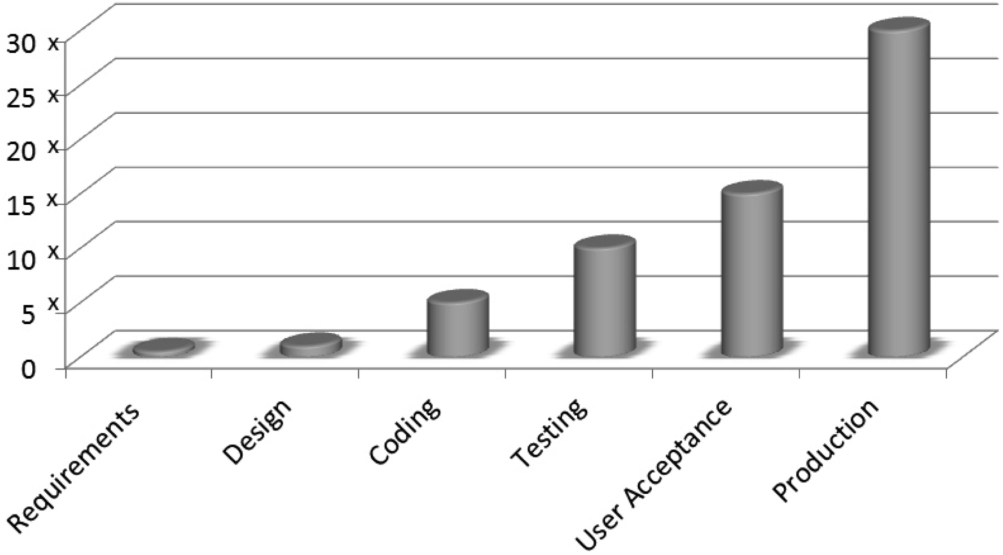


Security as an Afterthought

- Security does not add any business value

- Secure features are built into the software, instead of being added on at a later stage,

Relative Cost of Software Defects



Security Versus Usabilty

More Security = Less Usability

- Secure features is viewed as rendering the software to become very complex, restrictive and unusable

- For example, the human resources organization needs to be able to view payroll data of employees and the software development team has been asked to develop a web application that the human resources personnel can access

Quality = Security???

- A software product that is secure will add to the quality of that software but the inverse is not always necessarily true

- Security functionality in the vendor’s software does not make it secure

Quality is high with lack of **Security**

E-commerce Website

High-Quality but Not Secure:
Imagine an e-commerce website that:

- Loads pages quickly.
- Has an intuitive user interface.
- Processes orders without crashing.

But

- Passwords are stored in plaintext.
- There's no HTTPS encryption for data in transit.

Simple But Confusing Terms

Secrecy

An engineering term that refers to the effect of the mechanisms used to limit the number of principals who can access information, such as cryptography or computer access controls.

Confidentiality

An obligation to protect some other person or organisation secrets if you know them

Privacy

The ability and/or right to protect your personal information and extends to the ability and/or right to prevent invasions of your personal space (the exact definition of which varies from one country to another).

Trusted system or component is one whose failure can break the security policy

Trust

A trustworthy system or component is one that wont fail.

Trustworthy

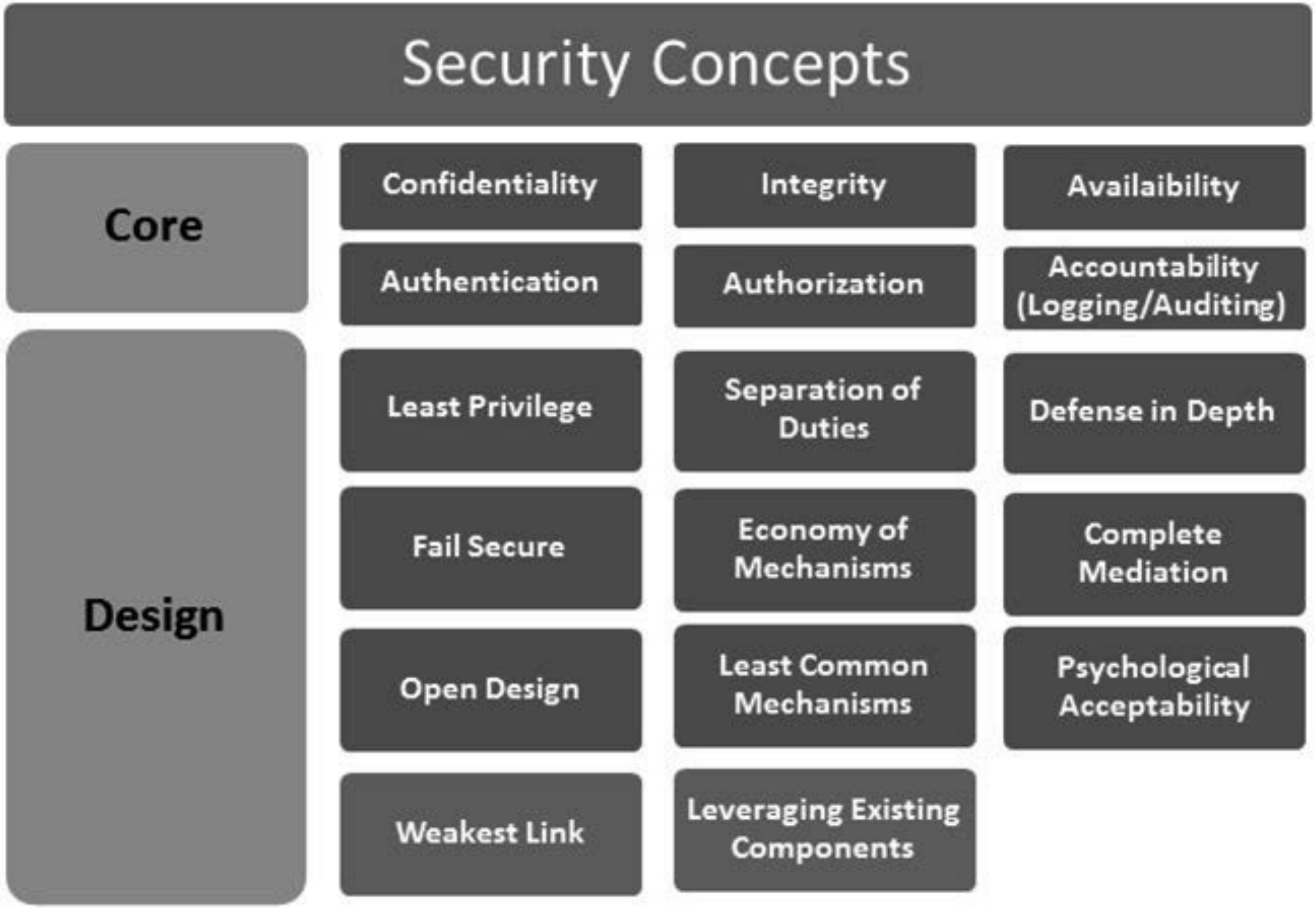
Simple But Confusing Terms

Trust

Trustworthy

The following example illustrates the difference: if an NSA employee is observed in a toilet stall at Baltimore Washington International airport selling key material to a Chinese diplomat, then (assuming his operation was not authorized) we can describe him as trusted but not trustworthy. I use the NSA definition that a trusted system or component is one whose failure can break the security policy, while a trustworthy system or component is one that wont fail

What makes Software Secure?



Security Concepts

Core	Confidentiality	Integrity	Availaibility
	Authentication	Authorization	Accountability (Logging/Auditing)
Design	Least Privilege	Separation of Duties	Defense in Depth
	Fail Secure	Economy of Mechanisms	Complete Mediation
	Open Design	Least Common Mechanisms	Psychological Acceptability
	Weakest Link	Leveraging Existing Components	

Looking Forward

zahmaad.github.io