CY321/CY321L	Secure Software Design and Engineering Lab (1 CH)	Knowledge Profile: WK3	Focus: PBL	CyS

Pre-Requisite: Cybersecurity Principles and Concepts

Instructor: Jazia Sajid
Office: F11, FCSE, GIK Institute
Email: Jazia.Sajid@giki.edu.pk
Office Hours: 8:00am ~ 05:00 pm

Course Introduction

This course provides a foundational understanding of secure software development, emphasizing integrating security throughout the Software Development Life Cycle (SDLC). Students will learn secure coding practices, risk management, threat modeling, and security testing. Through hands-on labs and projects, they will apply security frameworks, design secure systems, and address real-world vulnerabilities. By the end of the course, students will be equipped with the skills to build secure, compliant software systems.

Lab Contents

Lab 1: Security in the SDLC

Lab 2: Risk Management

Lab 3: Secure Design Principles and Threat Modeling

Lab 4: Implementing Security Frameworks

Lab 5: Security Mechanisms and Access Control

Lab 6: Identifying and Mitigating Vulnerabilities

Lab 7: Implementing Defensive Coding

Lab 8: Secure Coding Implementation

Lab 9: Security Testing Tools

Lab 10: Static and Dynamic Analysis

Lab 11: Implementing Authentication Protocols

Lab 12: Securing APIs

OEL: Open-Ended Lab

Mapping of CLOs and PLOs

Sr. No	Course Learning Outcomes[+]	WA PLOs	SA PLOs	Taxonomy level (Psychomotor Domain)
CLO_1	Integrate security measures into various phases of the Software Development Life Cycle (SDLC) to ensure secure software development practices.	PLO 2	Problem Analysis	P3 (Guided Response)
CLO_2	Analyze, design, and implement secure software solutions using frameworks, tools, and techniques to identify and mitigate potential security vulnerabilities.	PLO 5	Modern Tool Usage	P3 (Guided Response)
CLO_3	Conduct security testing and implement tools to detect vulnerabilities, mitigate risks, and ensure secure development through secure coding practices and defensive mechanisms.	PLO 3	Design/Development	P3 (Guided Response)

[+]Upon successful completion of this course, the student will be able to:

CLO Assessment Mechanism

Assessment Items	CLO1	CLO2

Lab Performance	50%	-
Midterm	50%	-
PBL	-	30%
Final	-	70%

Grading

Assessment Items	Percentage
Lab Performance	40%
Midterm Exam	20%
Project/PBL	10%
Final Exam	30%

Text and Reference Books

- Computer Kohnfelder, Loren. Designing Secure Software: A Guide for Developers. O'Reilly Media, 2021.
- SAFECode. Fundamental Practices for Secure Software Development. March 2018.
- Nebelwelt. Secure Software Development Practices.
- Olmsted, Aspen. Security-Driven Software Development. 2020.
- Pressman, Roger S. Software Engineering: A Practitioner's Approach. 9th ed., McGraw-Hill Higher Education, 2010.
- National Institute of Standards and Technology (NIST). Software Security Framework. NIST Special Publication 800-218.
- National Institute of Standards and Technology (NIST). NIST SP 800-218: Secure Software Development Framework (SSDF).

Administrative Instruction

- According to institute policy, 80% attendance is *mandatory* to appear in the final examination.
- Assignments must be submitted as per the instructions mentioned in the assignments.
- In any case, there will be no retake of (scheduled/surprise) quizzes.
- For queries, kindly follow the office hours to avoid any inconvenience.

Computer Usage

Wireshark , IriusRisk , Stride , CodeQl. Postman.

Lab Breakdown

Week 1: Lab 1 - Security in the SDLC

- Integrate security into SDLC phases and document requirements in Jira.
- Map security considerations to SDLC phases using case studies.

Week 2: Lab 2 - Risk Management

- Use IriusRisk to design and analyze a secure login system.
- Apply secure design principles and generate threat model reports.

Week 3: Lab 3 - Secure Design Principles and Threat Modeling

- Conduct STRIDE analysis, risk matrix, and mitigation plans using Excel.
- Compare STRIDE and OCTAVE frameworks and apply NIST standards.

Week 4: Lab 4 - Implementing Security Frameworks

- Apply COBIT, SABSA, and Zachman frameworks to align goals with controls.
- Use tools to map SABSA components and implement COBIT controls.

Week 5: Lab 5 - Security Mechanisms and Access Control

- Develop RBAC, authentication, and authorization mechanisms.
- Test access control policies using tools like Postman and OpenLDAP.

Week 6: Lab 6 - Identifying and Mitigating Vulnerabilities

- Simulate attacks (SQL injection, buffer overflow) using Nessus.
- Document findings, propose mitigations, and discuss real-world impacts.

Week 7: Lab 7 - Implementing Defensive Coding

- Write secure code, fix vulnerabilities, and follow OWASP guidelines.
- Use Burp Suite to detect and resolve security issues.

Week 8: Lab 8 - Secure Coding Implementation

- Develop secure login modules using bcrypt and validate inputs.
- Identify vulnerabilities with ESLint and CodeQL.

Week 9: Lab 9 - Security Testing Tools

- Conduct testing with OWASP ZAP and static analysis using SonarQube.
- Analyze vulnerability reports and log issues in Jira.

Week 10: Lab 10 - Static and Dynamic Analysis

- Perform static analysis with SonarQube and dynamic testing with OWASP ZAP.
- Simulate attacks like SQL injection and XSS to fix vulnerabilities.

Week 11: Lab 11 - Implementing Authentication Protocols

- Design authentication protocols and analyze traffic with Wireshark.
- Set up Kerberos for mutual authentication and implement password hashing.

Week 12: Lab 12 - Securing APIs

- Develop secure REST APIs using Flask or Express.js with OAuth2.
- Test APIs using Postman and secure data with HTTPS.

Week-13: Lab 13. Project/ Open-Ended Lab

Version	SPRING 2025
Revision Date	15-01-2024
Knowledge Group (KG) Head	Prof. Dr. Ghulam Abbas
KG Head Signature