CE407 Secure Programming Week-2

Development Environment Security and Software Development Processes

Author: Dr. UÄŸur CORUH

Contents

List of Figures

List of Tables

1 CE407 Secure Programming

1.1 Week-2

1.1.0.1 Development Environment Security and Software Development Processes

Download PDF[1],DOCX[2], SLIDE[3], PPTX[4]

1.1.1 Outline

- Development Environment Security and Software Development Processes
- Software Development Process
  - Software Development Flow
  - Baseline the Configuration

[1] pandoc_ce407-week-2.en_doc.pdf
[2] pandoc_ce407-week-2.en_word.docx
[3] ce407-week-2.en_slide.pdf
[4] ce407-week-2.en_slide.pptx

    - Initiating and Classifying Changes
    - Approving and Releasing Changes
- Software Development Environments
    - Development Environment Security
    - Version Control Systems
    - Source Code Server Security
    - Office and Server Room Security

1.2 Software Development Flow and Change Management

1.2.1 1. Software Development Flow

1.2.1.1 Theoretical Explanation: Software development processes must be managed using structured workflows. Proper change management ensures the project's success. This typically involves version control systems, technical teams, and project management processes.

1.2.1.2 Application:

- **Task:** Start a simple software project and demonstrate how to manage change requests (RFC). Create an approval mechanism at each step and manage the project using a version control system.

1.2.2 2. Baseline the Configuration

1.2.2.1 Theoretical Explanation: Baselining a configuration ensures that all changes after a particular version are traceable. This is a fundamental step in development and change management processes.

1.2.2.2 Application:

- **Task:** Create a GIT repository and baseline the initial configuration settings. Set up a structure to track all changes made from this version onwards.

1.2.3 3. Initiate the Change

1.2.3.1 Theoretical Explanation: Change requests (RFCs) are made to add new features or fix bugs in the project. Before development begins, all requirements are defined and evaluated through technical meetings.

1.2.3.2 Application:

- **Task:** Create an RFC and simulate how it is communicated to the project team. Develop a scenario that shows how decisions are made through meetings and technical reviews.

1.2.4 4. Classify the Change

1.2.4.1 Theoretical Explanation: Change requests are classified based on cost, time, and technical requirements. If there are no financial or technical obstacles, the product owner grants approval for the technical team to implement the change.

1.2.4.2 Application:

- **Task:** Review a change request and manage the process that shows how it is classified and approved based on specific criteria.

1.2.5 5. Approve and Schedule the Change

1.2.5.1 Theoretical Explanation: Before development starts, the change request is approved, and a project plan is created. This plan includes sprints and task assignments.

1.2.5.2 Application:

- **Task:** Organize a sprint planning meeting and assign tasks according to the approved change request. Use project management tools (Jira, Trello, etc.) to organize this process.

1.2.6 6. Release the Change

1.2.6.1 Theoretical Explanation: The developed change is pushed to the production environment after testing is complete. At this stage, it is ensured that the change has been applied successfully.

1.2.6.2 Application:

- **Task:** Pull the developed change from the version control system and release it to the production environment. Record the steps and results during the release process.

1.2.7 7. Validate and Review the Change

1.2.7.1 Theoretical Explanation: After the change is released, it is validated to ensure it has been applied correctly and meets expectations. Both technical and user reviews are conducted.

1.2.7.2 Application:

- **Task:** Test the released change and gather user feedback. Verify whether the change meets the expectations.

1.3 Software Development Environments and Version Control Systems

1.3.1 1. Development Environments

1.3.1.1 Theoretical Explanation: Software development occurs across different environments: development, test, and production. Each environment requires different security measures and configurations.

1.3.1.2 Application:

- **Task:** Set up development and test environments. Create an application that demonstrates the different security configurations for each environment.

1.3.2 2. Version Control Systems

1.3.2.1 Theoretical Explanation: Version control systems (GIT, SVN, etc.) are used to track software development processes and allow changes to be reverted if necessary. Every change is recorded, and versions can be switched as needed.

1.3.2.2 Application:

- **Task:** Manage a software development process using GIT. Switch between branches and revert a change.

1.3.3 3. Development Site and Source Code Server Security

1.3.3.1 Theoretical Explanation: The physical and digital security of the development environment is critical. Securing source code servers and monitoring systems ensures the integrity of the software.

1.3.3.2 Application:

- **Task:** Demonstrate how to secure a source code server in a development environment. Set up encryption and access control systems practically.

1.3.4 4. Development Office and Server Room Security

1.3.4.1 Theoretical Explanation: Server rooms and development computers must be secured with access control, encryption, and physical security measures to protect the software.

1.3.4.2 Application:

- **Task:** Simulate access control for a server room. Configure security software on development computers and take precautions against potential attacks.

1.4 Weekly Summary and Next Week

1.4.1 This Week:

- **Software Development Flow and Change Management**
- **Configuration Baseline and Change Approval**
- **Development Environments and Version Control Systems**
- **Physical and Digital Security**

1.4.2 Next Week:

- **Data Security and Cryptography**
- **Secure Communication and Key Management**

$$End - Of - Week - 1$$