Progetto di SSDLC - SecDevOps

Progetto di Sviluppo Sicuro per progetto Java/C# con Pipeline di CI/CD

Obiettivo

Implementare un semplice processo di sviluppo sicuro per un'ipotetica azienda tramite la configurazione di una pipeline di CI/CD con Jenkins o strumenti equivalenti (GitHub Actions, TravisCI, ecc.) per automatizzare la build e le verifiche di qualità e di sicurezza del codice sorgente.

Attività

1. **Configurazione dell'ambiente**
    - È possibile configurare l'ambiente in due modi:
        - Installazione dei diversi strumenti direttamente sul sistema operativo oppure su Macchina Virtuale.
        - Installazione tramite utilizzo di container Docker (aggiunge un grado di complessità aggiuntivo, che verrà valutato positivamente).
2. **Scelta di un repository open source**
    - Analizzare il funzionamento del codice sorgente e scegliere un progetto Java o C# tra quelli proposti al termine del presente documento.
3. **Configurazione della Pipeline CI/CD**
    - **Installazione di Jenkins**: Assicurarsi che Jenkins sia installato e in esecuzione.
    - **Configurazione del Repository**: Configurare il repository Git contenente il codice.
    - **Definizione della Pipeline**:
        - **Stage 1 - Check-out del codice**: Recupero del codice sorgente dal repository.
        - **Stage 2 - Build**: Utilizzare Maven o Gradle (Java), oppure NuGet (C#) per compilare il codice.
        - **Stage 3 - Scansione SAST (Static Application Security Testing) del codice**: Integrare SonarQube e altri strumenti per l'analisi statica del codice.

        - **Stage 4** - **Scansione SCA (Software Composition Analysis)**: Integrare Dependency Check, Dependency Track o altri strumenti per l'analisi delle librerie di terze parti utilizzate.
        - **Stage 5** - **Check dei gate di qualità e sicurezza (quality gate check)**: Interrompere la pipeline in caso di fallimento dei requisiti minimi imposti sui risultati delle scansioni di qualità e di sicurezza del codice.
        - **Stage 6 - Archiviazione locale**: Archiviazione degli artefatti di build (ad esempio archiviazione di un JAR sulla pipeline Jenkins) qualora siano stati superati i gate di qualità / sicurezza.
        - **Stage 7 - Notifica**: Notifica dei risultati delle scansioni di sicurezza ad un ipotetico team di sicurezza.
4. **[Facoltativo] Risoluzione delle vulnerabilità**
    - Dare evidenza del rimedio di 2 delle vulnerabilità più critiche, effettuando una scansione del codice originale e comparandola con quella del codice modificato.
        - Impostare il Gate in maniera proporzionale alle vulnerabilità riscontrate durante la prima scansione, in maniera tale che risolvendo le vulnerabilità definite in precedenza, il codice modificato superi i controlli del Gate.

Redazione del Report

Il report deve includere una spiegazione dettagliata e sequenziale delle operazioni svolte durante lo sviluppo e la configurazione della pipeline.

Struttura del Report

1. **Introduzione**
2. **Configurazione della Pipeline di CI/CD**
    - Descrizione dei passaggi per configurare la pipeline (es: pipeline Jenkins).
    - Strumenti integrati nella pipeline per l'analisi statica del codice (es. SonarQube, SpotBugs, ...).
    - Strumenti integrati nella pipeline per l'analisi della sicurezza delle librerie di terze parti utilizzate.
    - Spiegazione del funzionamento della pipeline e dei controlli di sicurezza.
    - Dettagli su come è stato configurato il gate di sicurezza.

- Dare evidenza delle modifiche fatte al codice, al fine di risolvere le 2 vulnerabilità più critiche mitigate (se svolto).

3. **Analisi delle Vulnerabilità**
    - Risultati delle scansioni di sicurezza effettuate dalla pipeline.
    - Per almeno 10 vulnerabilità trovate, includere:
        - **Descrizione della vulnerabilità**: Dettagli sulla natura della vulnerabilità.
        - **Prova della rilevazione**: Immagini o snippet di output che dimostrano la rilevazione.
        - **Classificazione OWASP TOP 10**: Quando applicabile, indicare la classificazione secondo OWASP.
        - **Gravità e Impatti**: Valutazione della gravità e delle possibili conseguenze.
        - **Fix del Codice**: Suggerimenti per la correzione della vulnerabilità.

Repository codice sorgente

Scegliere uno dei repository elencati.

Progetti Java

- https://github.com/jaygajera17/E-commerce-project-springBoot/tree/master2
- https://github.com/shashirajraja/onlinebookstore

Progetti C#

- https://github.com/vijaythapa333/ONLINE-BRANDING-SYSTEM-PP-
- https://github.com/vijaythapa333/anystore?tab=readme-ov-file