**Sprint 2 Plan: Kubernetes Cluster Security Testing and Built-in Vulnerability Scanning Tool Development**

Goal: Research Kubernetes cluster security testing methods, investigate the feasibility of a built-in vulnerability scanning tool, and initiate the development of the scanning tool.

Objective 1: Comprehensive Study of Kubernetes Cluster Security Testing

- Spend the first three days deeply exploring Kubernetes cluster security testing techniques.

- Analyze prominent security testing tools like kube-bench, kube-hunter, and others.

- Research strategies for penetration testing, vulnerability assessment, and configuration audits.

- Collect resources: security blogs, articles, and whitepapers on Kubernetes security practices.

Objective 2: Feasibility Evaluation for a Built-in Vulnerability Scanning Tool

- Allocate the next two days to assess the viability of creating a native vulnerability scanning tool within Kubernetes.

- Study existing solutions to understand their architecture and functionality.

- Identify potential challenges and benefits of integrating a scanning tool directly into Kubernetes.

- Gather resources: existing vulnerability scanning tools, Kubernetes extensibility documentation, relevant forum discussions.

Week 2: Tool Development Kick-off

Objective 3: Setting Up Development Environment

- Begin the first two days of the week by configuring the development environment.

- Choose programming languages and tools suitable for building the scanning tool.

- Set up a version control repository to track code changes.

- Gather resources: programming languages (Python, Go), development environment guides.

Objective 4: Initial Development of Vulnerability Scanning Tool

- Over the next three days, start the development process of the vulnerability scanning tool.

- Focus on a simple use case, such as scanning for outdated software versions in Kubernetes resources.

- Utilize Kubernetes APIs to interact with cluster resources and extract relevant information.

- Develop initial scanning logic and reporting functionality.

- Resources: Kubernetes API documentation, programming guides.

Objective 5: Progressive Tool Enhancement

- Allocate the first three days of the week to iteratively enhance the vulnerability scanning tool.

- Implement additional scanning criteria based on your research, such as Kubernetes misconfigurations and common vulnerabilities.

- Develop an alerting mechanism to notify users about detected vulnerabilities.

- Continuously test the tool on various Kubernetes clusters.

- Resources: Security best practices, programming resources, vulnerability databases.

Objective 6: Analysis and Documentation

- Dedicate the last two days of the sprint to analyzing the tool's performance and documenting progress.

- Review the effectiveness of the scanning tool in identifying vulnerabilities.

- Document the tool's features, limitations, and potential use cases.

- Prepare a progress report outlining the development journey, challenges faced, and accomplishments.

- Resources: Documentation templates, reporting guidelines.

Through this sprint, a robust understanding of Kubernetes cluster security testing will be established, while also initiating the development of a built-in vulnerability scanning tool. This work lays the foundation for enhancing Kubernetes security practices and potentially contributing a valuable tool to the Kubernetes ecosystem.