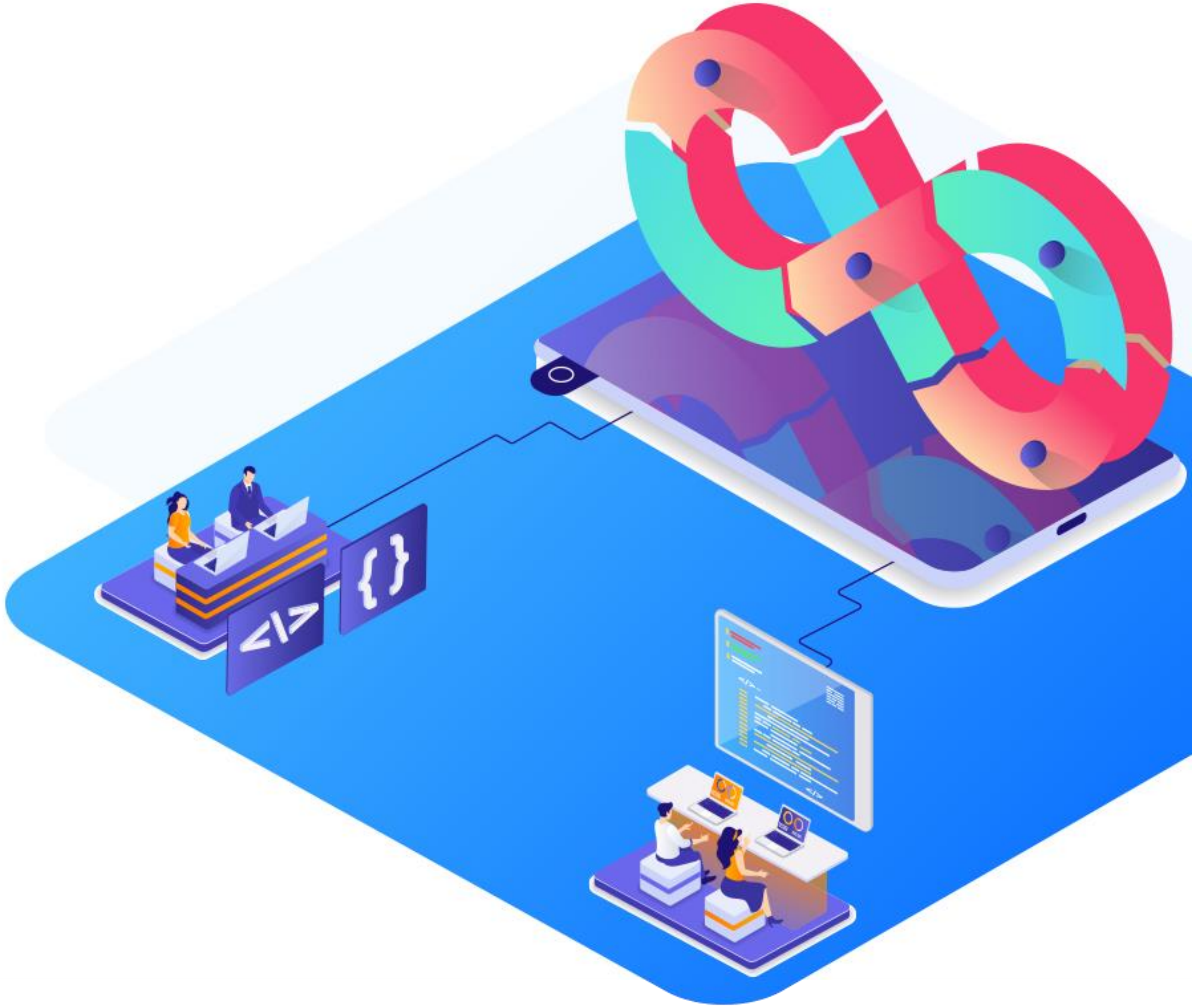
DevOps Foundations: Version Control and CI/CD with Jenkins



DevSecOps with Jenkins

Learning Objectives

By the end of this lesson, you will be able to:

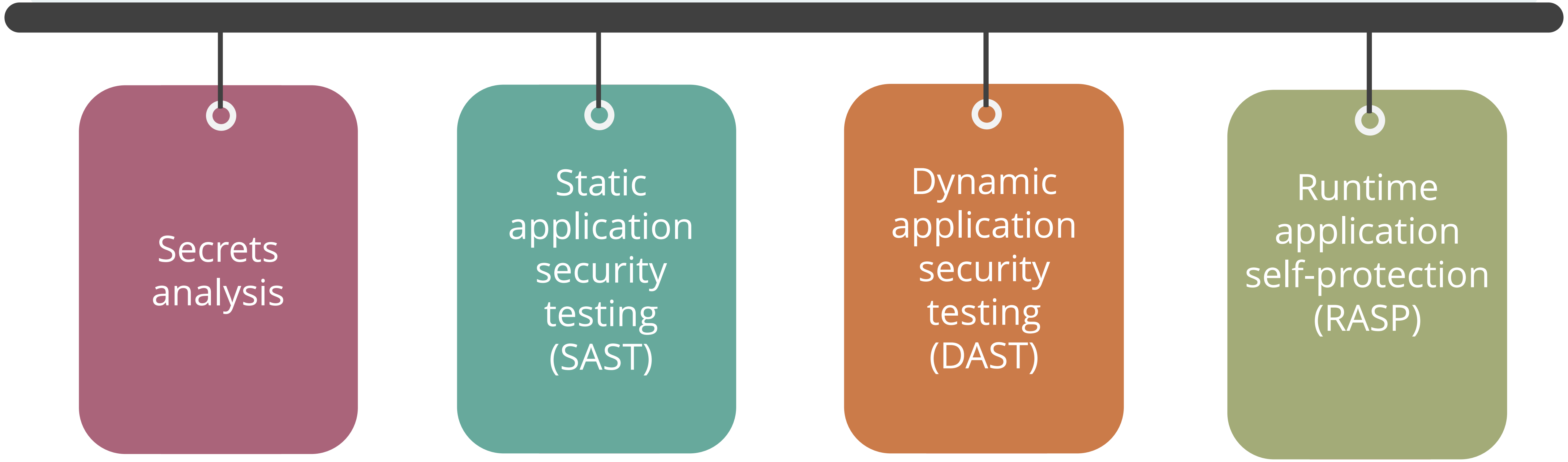
- Outline the various stages of DevSecOps to identify their appropriate applications in a secure software development lifecycle
- Apply static application security testing (SAST) techniques in a Jenkins CI/CD pipeline to identify potential vulnerabilities
- Integrate software composition analysis (SCA) tools with Jenkins to automatically scan for vulnerabilities in third-party libraries and dependencies in a continuous integration pipeline
- Construct a dynamic application security testing (DAST) setup using Jenkins to evaluate its effectiveness in simulating real-world attack scenarios on web applications

Getting Started with DevSecOps on Jenkins

DevSecOps Security Check Stages

DevSecOps integrates security throughout the entire software development lifecycle. This proactive approach helps to identify and fix vulnerabilities early, reducing the risk of security breaches.

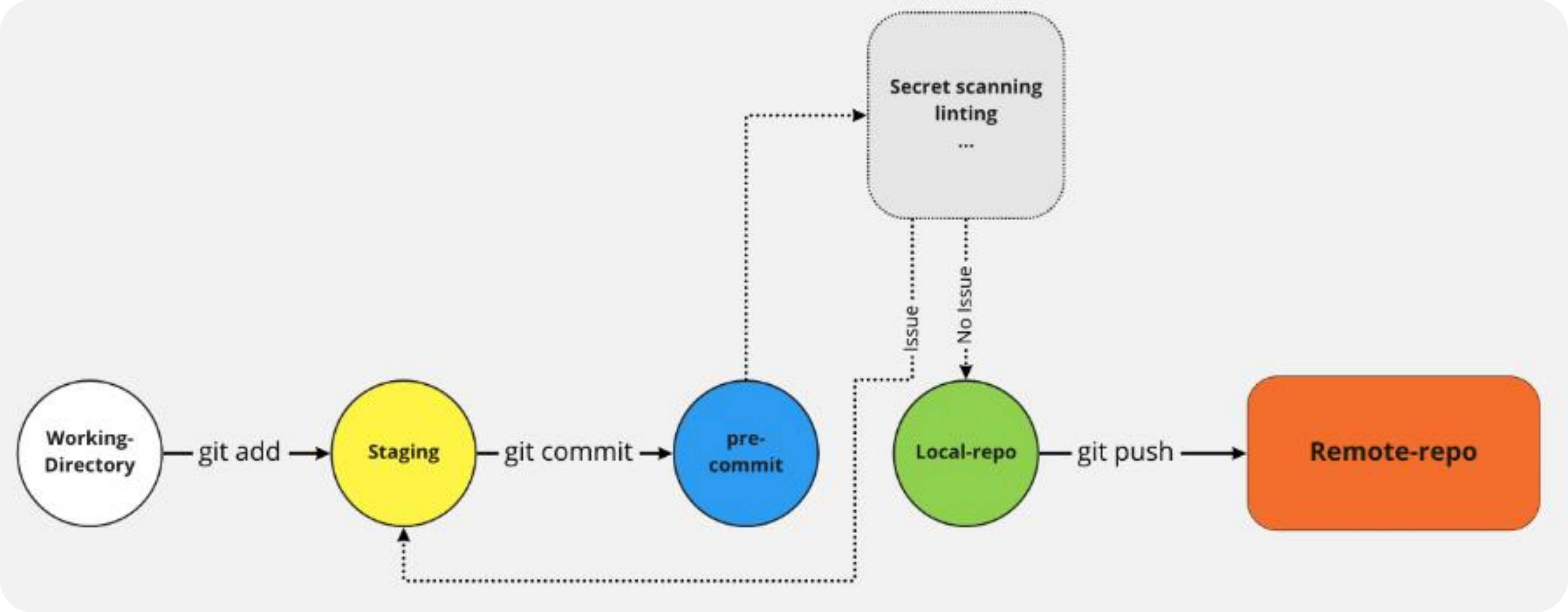
Following are the stages to implement DevSecOps to ensure security throughout the development process:



- Secrets analysis
- Static application security testing (SAST)
- Dynamic application security testing (DAST)
- Runtime application self-protection (RASP)

What Is Secrets Analysis?

Secrets analysis in the DevSecOps pipeline specifically focused on identifying and preventing the exposure of sensitive information like API keys, passwords, and other credentials.



Commonly, vulnerabilities are discovered while pushing code to remote repositories, so secret scanning should be performed before remote repository pushes.

Source: https://owasp.org/www-project-devsecops-guideline/latest/01a-Secrets-Management

What Is Static Application Security Testing (SAST)?

It is a security technique used to identify vulnerabilities in software by analyzing its source code before it is compiled.

SAST tools can be integrated with the IDE to perform the following:

- **Code scan:** Examine the source code of the application to search for potential security weaknesses
- **Vulnerability detection:** Identify SQL injection flaws, cross-site scripting (XSS) vulnerabilities, buffer overflows, and insecure authentication practices
- **Early bug detection:** Allow developers to catch and address security issues as they write the code

What Is Dynamic Application Security Testing (DAST)?

DAST is a **black-box** testing that detects runtime security vulnerabilities and weaknesses of an application by injecting malicious payloads.

DAST tools provide scans for the client side and server side of the application to detect:

- **Input or output validation:** Simulate real-world attacks on a web application
- **Authentication issues:** Simulate brute-force attacks where attackers try many password combinations to gain unauthorized access
- **Server configuration mistakes:** Identify vulnerabilities that arise due to the server misconfigurations

What Is Runtime Application Self-Protection (RASP)?

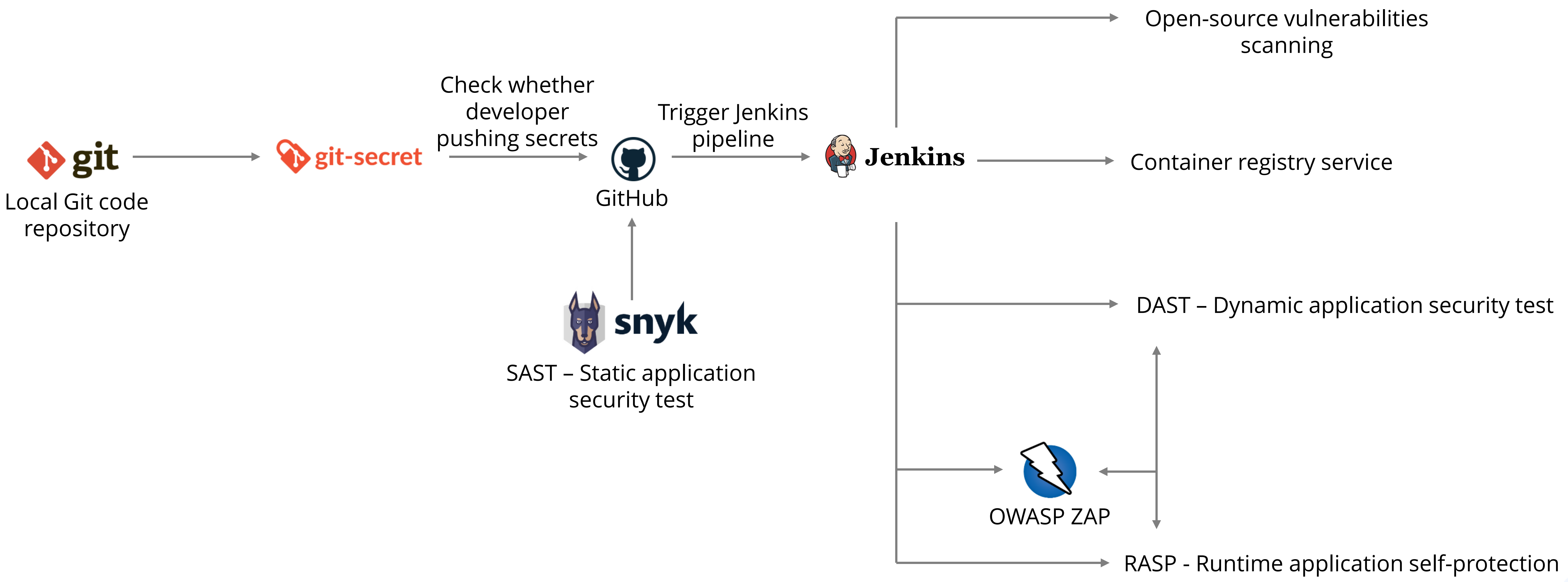
RASP is a technology that works on a server during the runtime of an application. These tools are designed to detect real-time attacks on an application.

RASP tools provide scans for the server side of the application to detect:

- **Deployment vulnerabilities:** Integrate into the application or its runtime environment and enable real-time protection
- **Broken authentication:** Identify weak authentication processes that can allow attackers to gain unauthorized access to applications
- **Incoming traffic vulnerabilities:** Monitor incoming traffic to servers and APIs without impacting application performance

DevSecOps with Jenkins Architecture

The following shows the architecture of a DevSecOps pipeline using Jenkins to identify and fix security vulnerabilities:

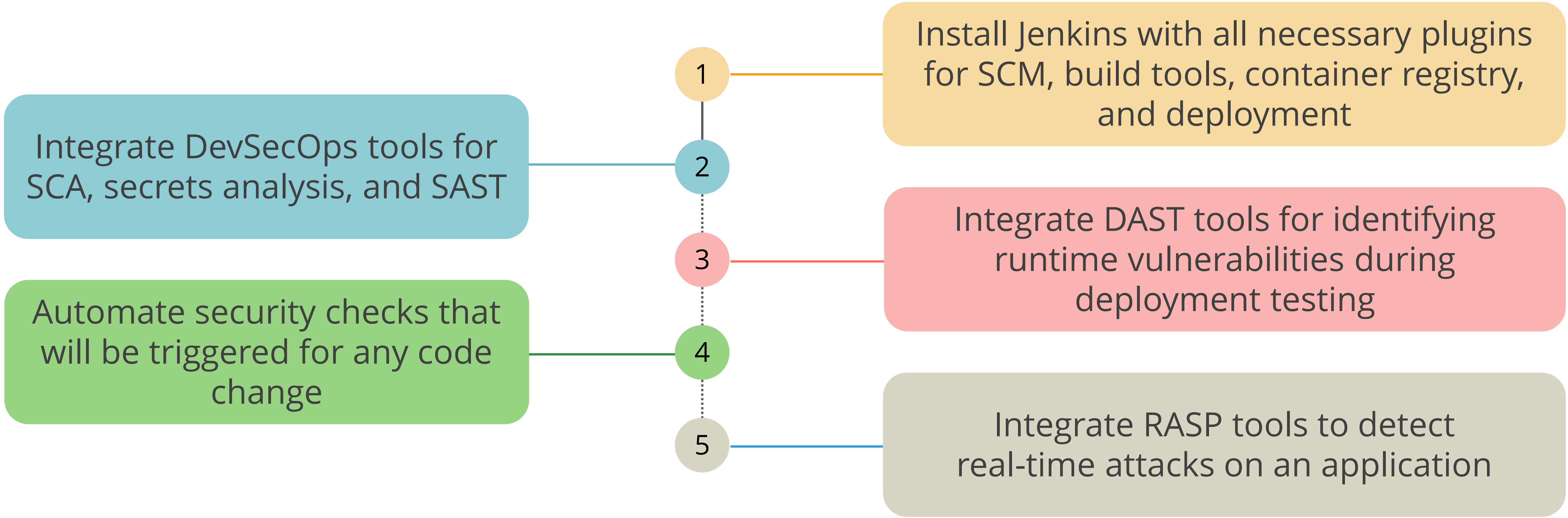


git
Local Git code repository
git-secret
Check whether developer pushing secrets
GitHub
Trigger Jenkins pipeline
snyk
SAST – Static application security test
Jenkins
Open-source vulnerabilities scanning
Container registry service
DAST – Dynamic application security test
OWASP ZAP
RASP - Runtime application self-protection

This ensures comprehensive security checks at multiple development and deployment stages, using tools like **git-secret**, **Snyk**, and **OWASP ZAP** to maintain a secure application environment.

DevSecOps with Jenkins Architecture

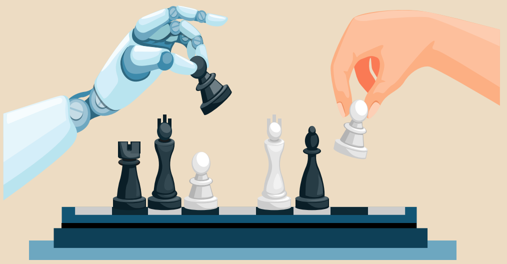
The following shows the implementation of DevSecOps tools to create a secure and automated Jenkins CI/CD pipeline:



1. Install Jenkins with all necessary plugins for SCM, build tools, container registry, and deployment
2. Integrate DevSecOps tools for SCA, secrets analysis, and SAST
3. Integrate DAST tools for identifying runtime vulnerabilities during deployment testing
4. Automate security checks that will be triggered for any code change
5. Integrate RASP tools to detect real-time attacks on an application

Case Study

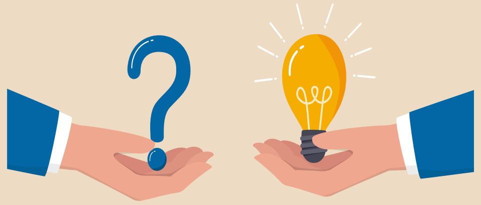
Streamlining security and innovation by implementing DevSecOps with Jenkins



Challenges

- Gainsight, a leading customer success platform with a vast user base, faced a challenge in balancing rapid development with robust security practices.
- Their existing development process lacked automation and integration, hindering collaboration and slowing down releases.
- Additionally, they were facing difficulties in ensuring the code quality with security throughout the development lifecycle.

Case Study



Solution

- Gainsight’s engineering team decided to implement a smarter, faster DevSecOps platform using Jenkins.
- They utilized Jenkins as an open-source automation server that provided a central hub to integrate various development and security tools.
- They adopted an Infrastructure as Code (IaC) approach to manage the entire platform configuration along with robust DevSecOps practices at an early stage of development.

Case Study

Outcome

- Enhanced security by integrating security tools at each phase of development within the Jenkins pipeline
- Increased efficiency by enabling automation of repetitive tasks, streamlining the development lifecycle
- Scalable infrastructure was achieved by leveraging Jenkins' Infrastructure as Code approach, resulting in a 95% code-scalable architecture.

Gainsight's lead DevOps engineer emphasizes the pivotal role of Jenkins: "*Jenkins is the epicenter of DevSecOps in our organization.*"

Quick Check

Your company is building a new e-commerce platform and wants to implement DevSecOps practices. Which of the following stages is the most crucial security check to perform before deploying the code to production using Jenkins?

A. RASP (Runtime application self-protection)

B. Secrets analysis

C. DAST (Dynamic application security testing)

D. SAST (Static application security testing)

Automating SAST in Jenkins with Snyk

What Is Snyk?

Snyk is a developer security platform that helps software development teams find and fix vulnerabilities across various aspects of their applications.

It can proactively help find and fix vulnerabilities for **NPM, Maven, NuGet, RubyGems, containers, and infrastructure as code.**

Why Use Snyk?

Snyk being an open-source security tool offers the following security scanning capabilities:

- **Static Application Security Testing (SAST):** Snyk can scan the source code for static vulnerabilities, ensuring the application is secure.

- **Dynamic Application Security Testing (DAST):** It can scan the runtime for vulnerabilities in the application.

- **Container security testing:** It ensures containers are secure on-premises, in the cloud, and while using Kubernetes.

- **Infrastructure security testing:** It can scan infrastructure code for vulnerabilities.

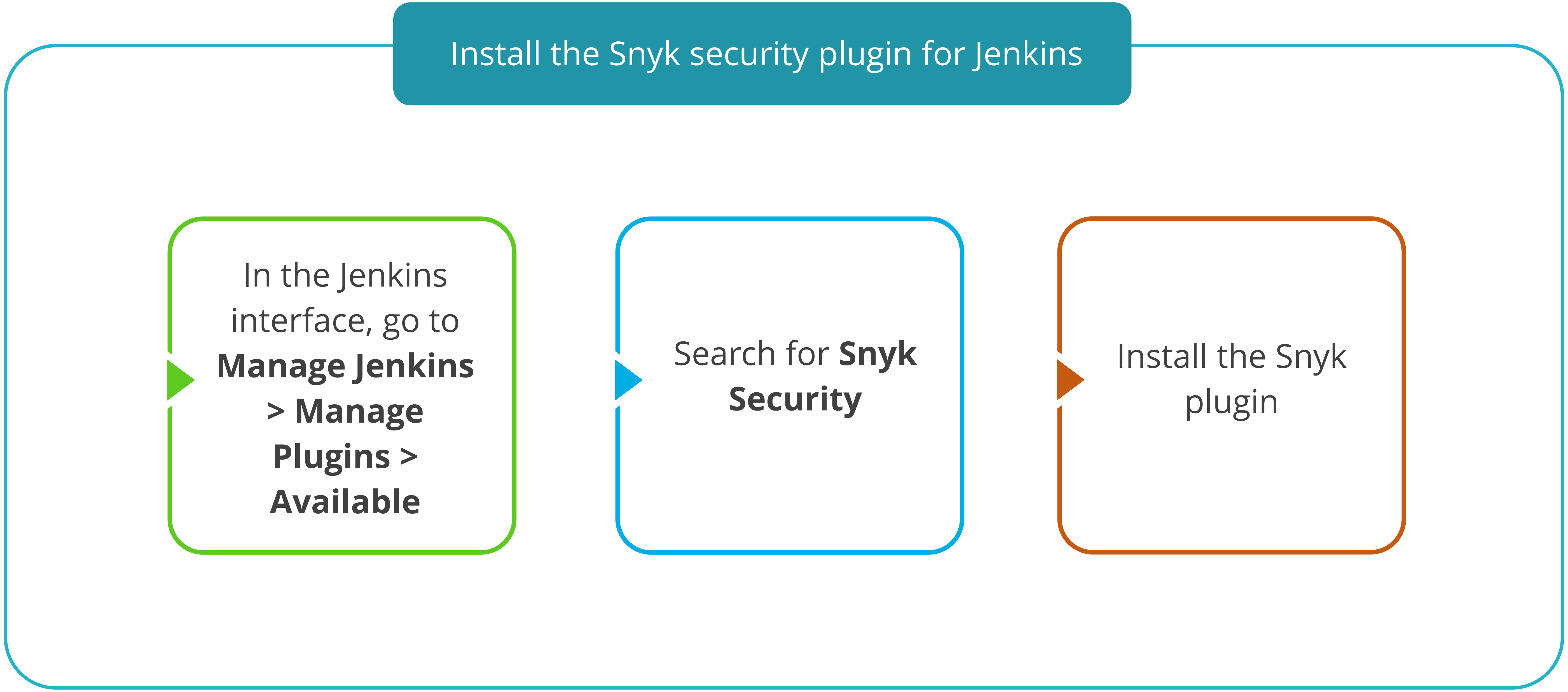
- **Actionable fix advice:** It identifies vulnerabilities and provides clear and actionable advice on how to fix them throughout the development process.

Companies Using Snyk

snowflake
reddit
The Telegraph
Spotify®
ATLASSIAN
AUSTRALIA POST

How to Integrate Snyk with Jenkins

Following are the steps to integrate Snyk with Jenkins:



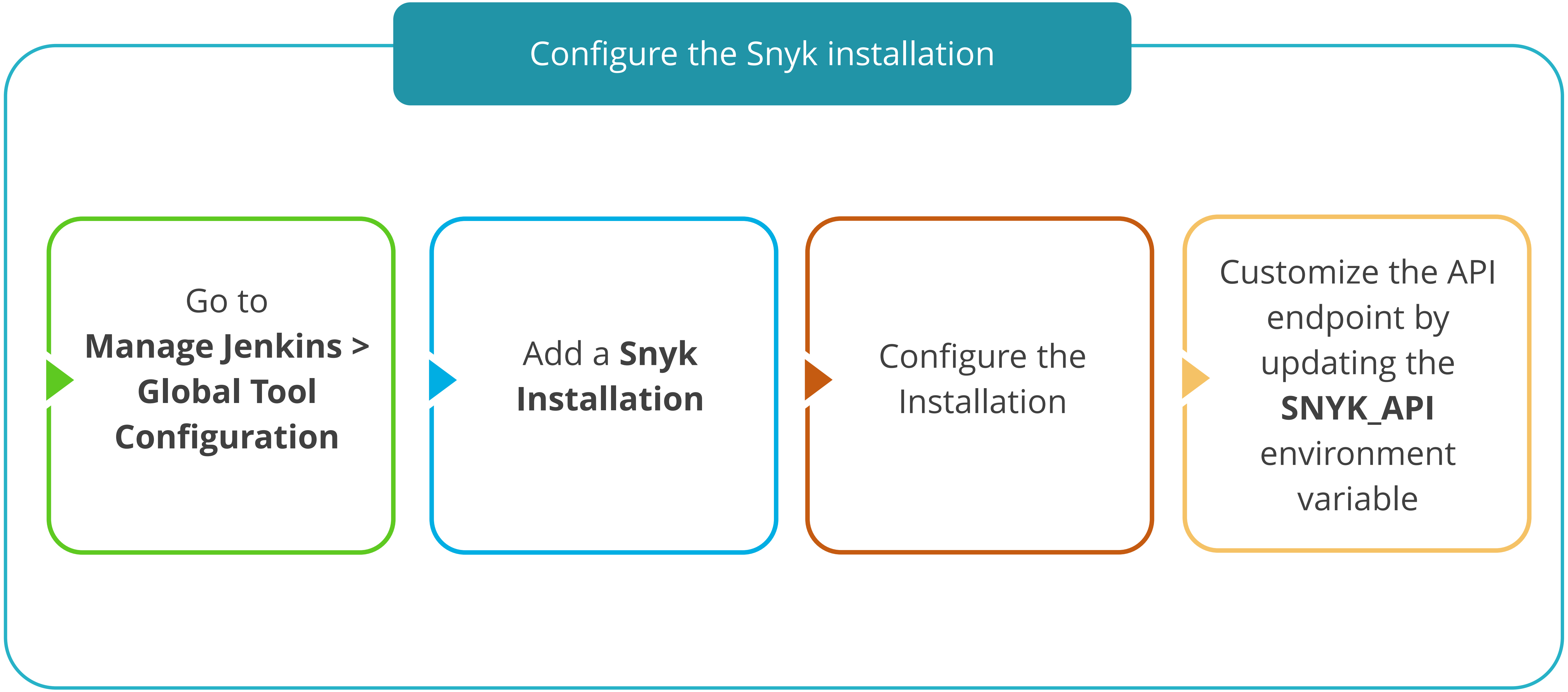
Install the Snyk security plugin for Jenkins

In the Jenkins interface, go to **Manage Jenkins > Manage Plugins > Available**

Search for **Snyk Security**

Install the Snyk plugin

How to Integrate Snyk with Jenkins



Configure the Snyk installation

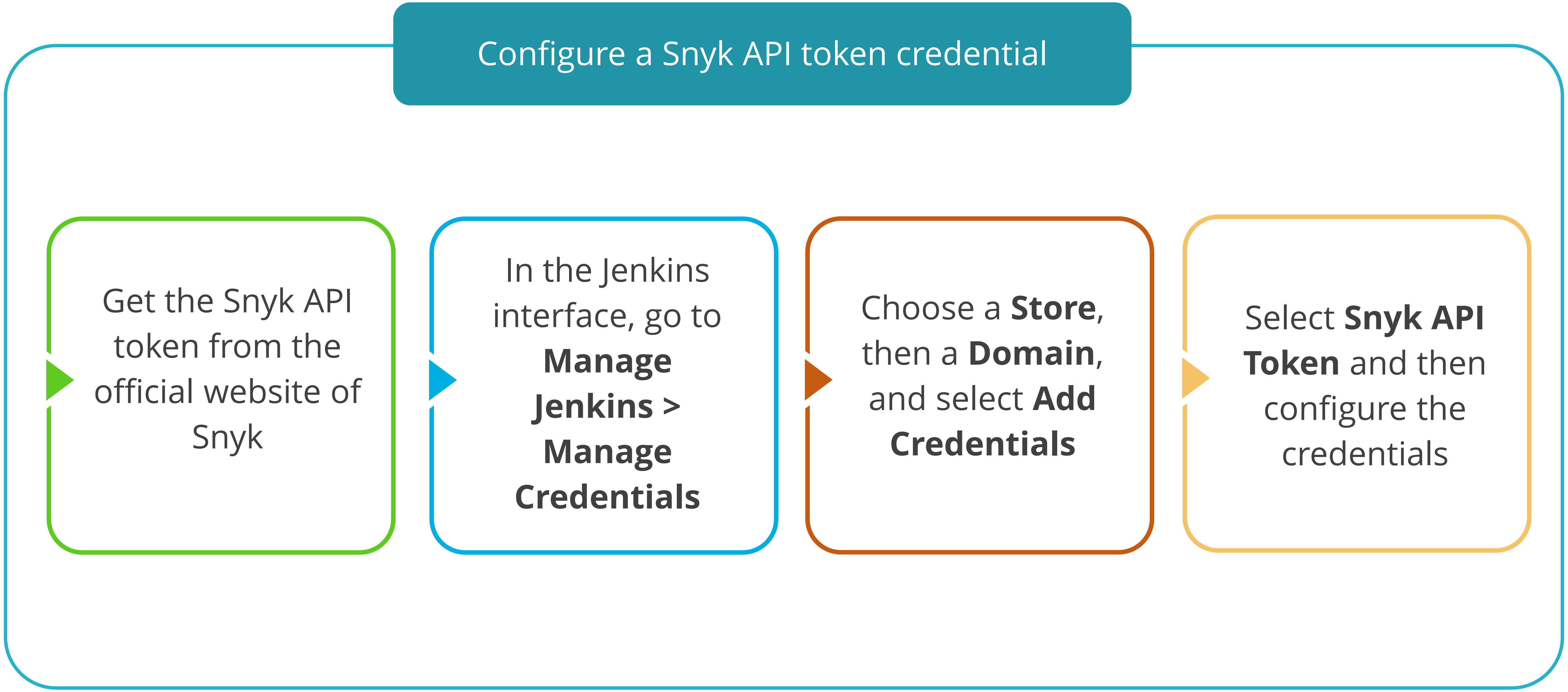
Go to **Manage Jenkins > Global Tool Configuration**

Add a **Snyk Installation**

Configure the Installation

Customize the API endpoint by updating the **SNYK_API** environment variable

How to Integrate Snyk with Jenkins



Configure a Snyk API token credential

Get the Snyk API token from the official website of Snyk

In the Jenkins interface, go to **Manage Jenkins > Manage Credentials**

Choose a **Store**, then a **Domain**, and select **Add Credentials**

Select **Snyk API Token** and then configure the credentials

How to Integrate Snyk with Jenkins

Add Snyk security to the project

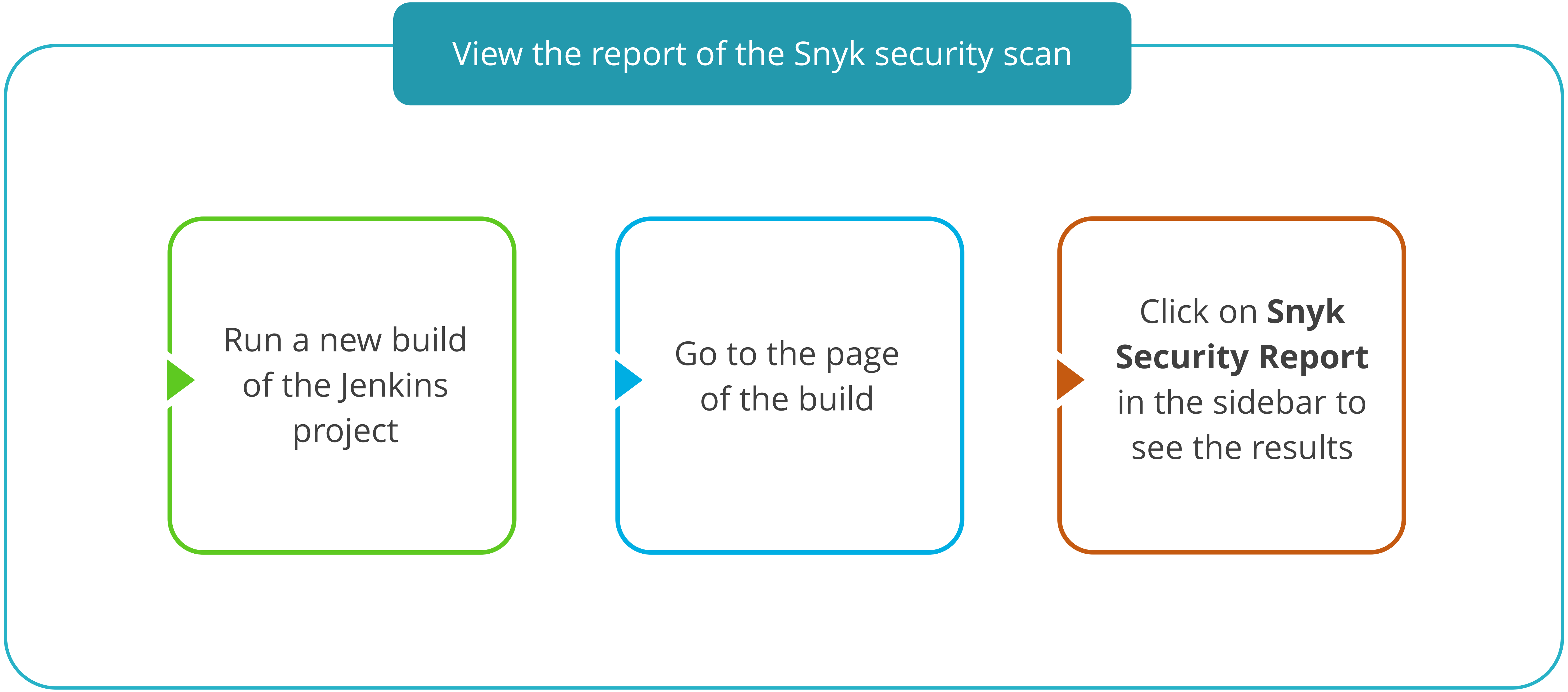
Freestyle project

- Select a project
- Go to **Configure**
- Under Build, select **Add build step > Invoke Snyk Security Task**

Pipeline project

- Use the **snykSecurity** step with the required parameters as part of the pipeline script

How to Integrate Snyk with Jenkins



View the report of the Snyk security scan

Run a new build of the Jenkins project

Go to the page of the build

Click on **Snyk Security Report** in the sidebar to see the results

Assisted Practice

Setting up Snyk for SAST in Jenkins **Duration: 20 Min.**

Problem statement:

You have been assigned a task to demonstrate the setup of the Snyk plugin in Jenkins for Static Application Security Testing (SAST), to automatically detect vulnerabilities in their codebase during development, thereby enhancing application security before deployment.

Outcome:

By completing this demo, you will gain proficiency in setting up the Snyk plugin in Jenkins to perform SAST scans. You will configure Snyk as a SAST tool and set up a Jenkins job to perform these scans.

Note: Refer to the demo document for detailed steps

Assisted Practice: Guidelines

Steps to be followed:

1. Configure Snyk as a SAST scan tool
2. Create and configure a Jenkins job for Snyk integration
3. Manage Snyk API and Jenkins credentials
4. Configure the Jenkins job for scanning

ASSISTED PRACTICE

Quick Check

You're building a Jenkins pipeline for an application and want to include Snyk security checks. How would you integrate Snyk for **automated code analysis** to detect vulnerabilities before deployment?

A. Use the **Post-build Actions** section of the pipeline to trigger a SAST scan with Snyk

B. Configure the Snyk Jenkins plugin within the code pipeline to scan the codebase during the build stage

C. Manually run a SAST scan with Snyk after every build completes

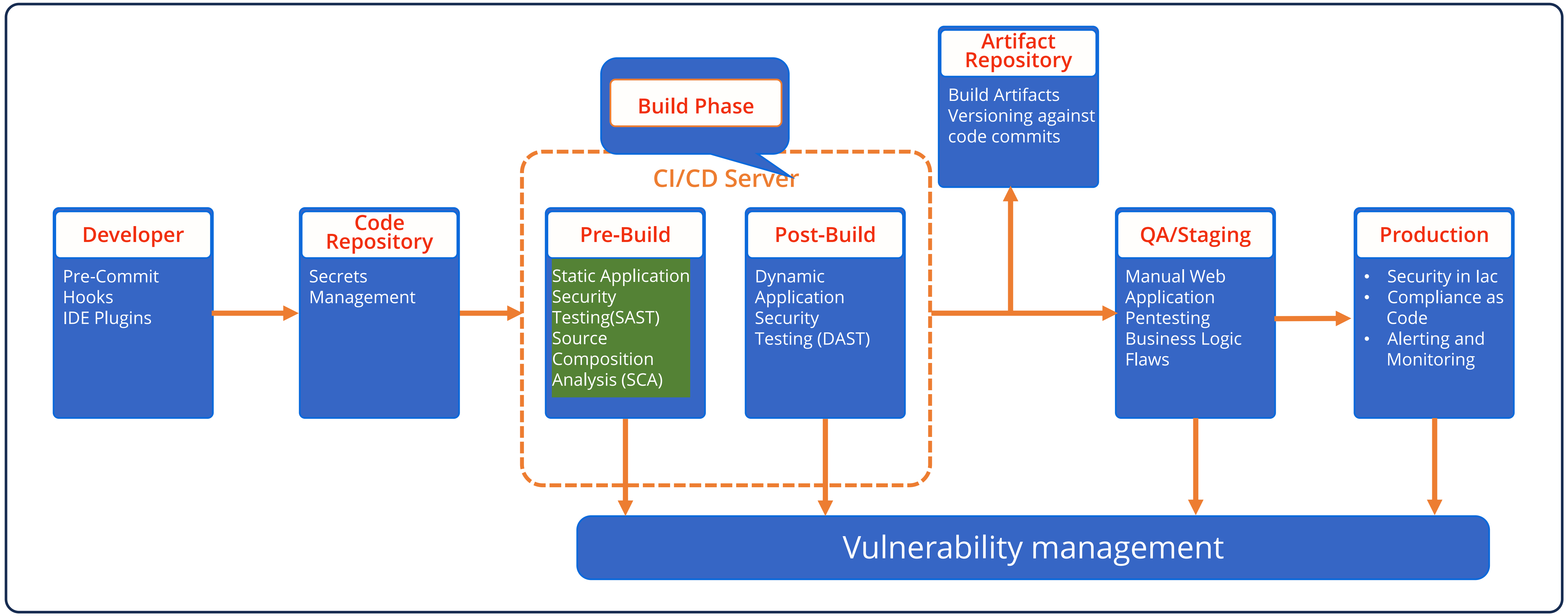
D. Integrate Snyk with the container registry to scan container images during deployment

Software Composition Analysis in Jenkins

Overview of Software Composition Analysis

Software Composition Analysis (**SCA**) testing manages risks from third-party and open-source components in software projects, enhancing development and security management.

SCA process in SDLC



Developer
Pre-Commit Hooks
IDE Plugins

Code Repository
Secrets Management

CI/CD Server

Build Phase

Pre-Build
Static Application Security Testing(SAST)
Source Composition Analysis (SCA)

Post-Build
Dynamic Application Security Testing (DAST)

Artifact Repository
Build Artifacts Versioning against code commits

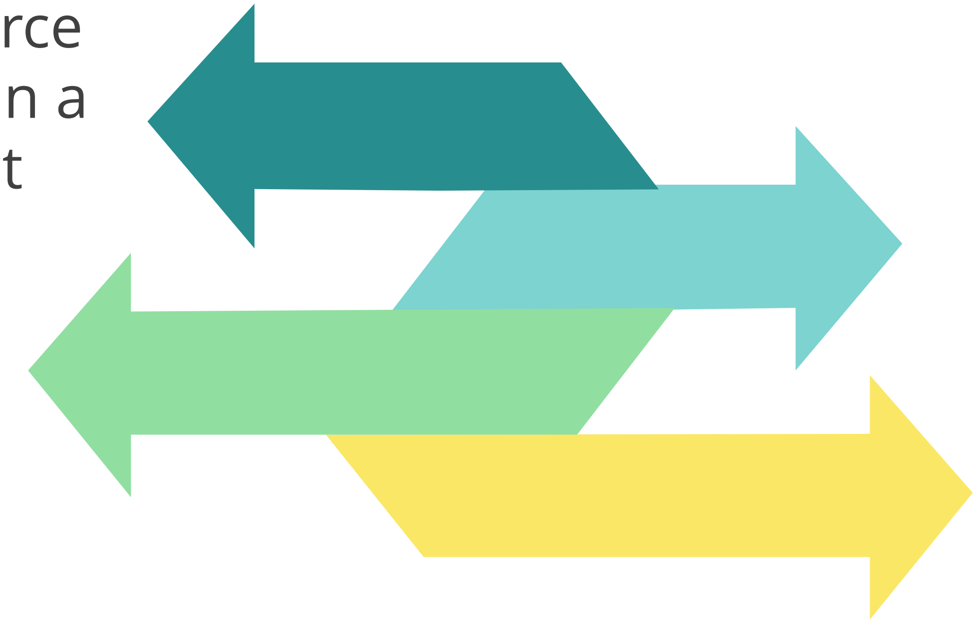
QA/Staging
Manual Web Application Pentesting
Business Logic Flaws

Production
- Security in Iac
- Compliance as Code
- Alerting and Monitoring

Vulnerability management

Why Is SCA Important?

SCA testing is primarily used to:



Identify open-source components within a software project

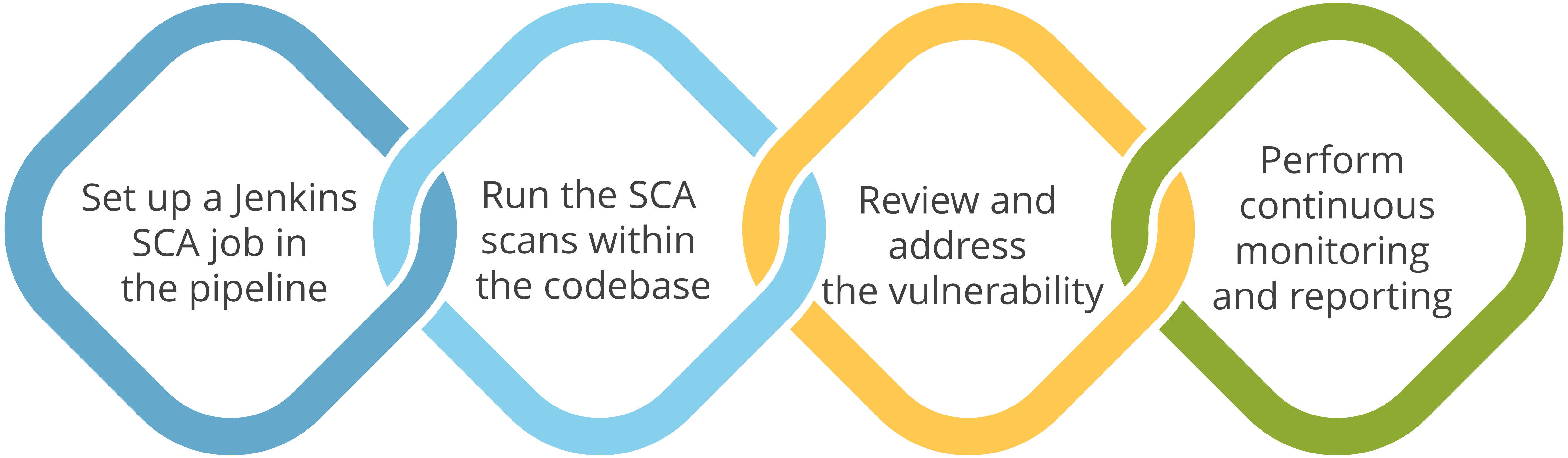
Identify the security vulnerabilities of third-party tool integration

Detect license compliance issues

Manage outdated libraries or frameworks

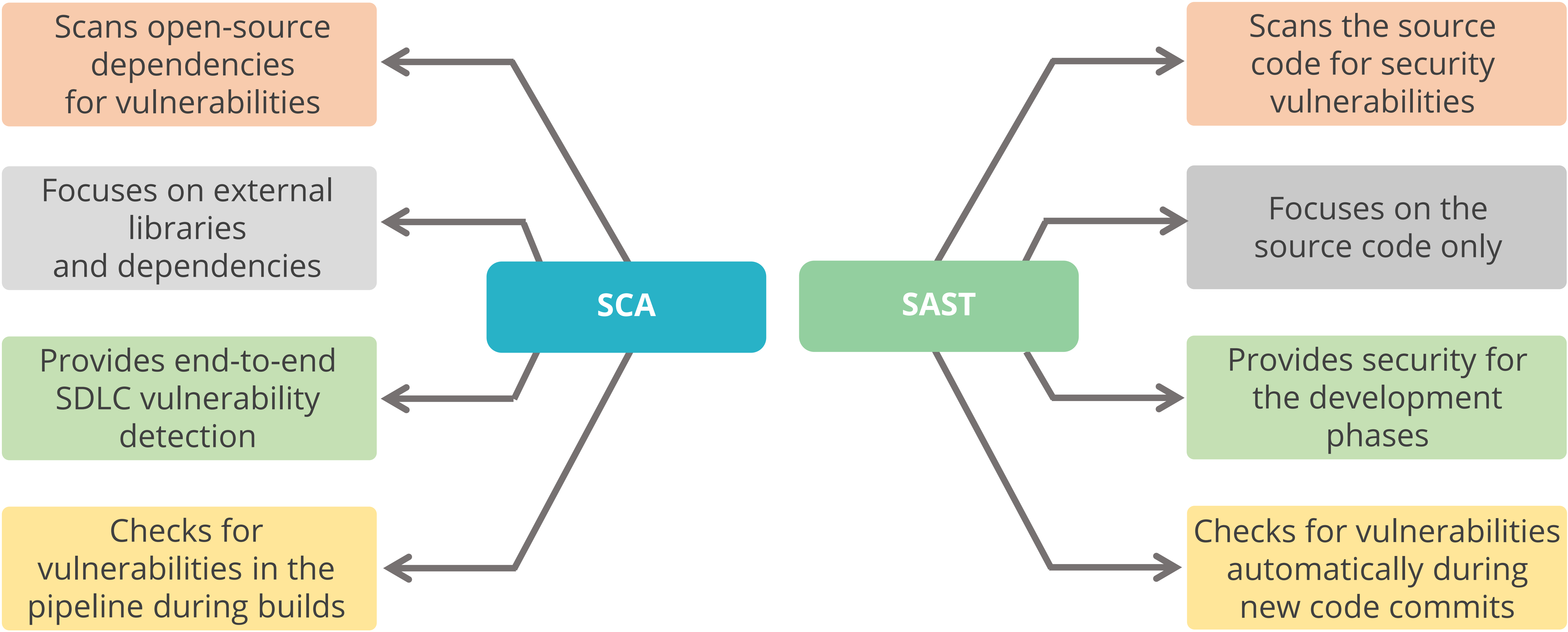
How SCA Works?

SCA scan is integrated within a Jenkins pipeline to perform scheduled scans at various stages typically either before or after builds. The following shows a typical SCA scan routine in a CI/CD pipeline:



Set up a Jenkins SCA job in the pipeline
Run the SCA scans within the codebase
Review and address the vulnerability
Perform continuous monitoring and reporting

SCA vs. SAST



SCA

Scans open-source dependencies for vulnerabilities

Focuses on external libraries and dependencies

Provides end-to-end SDLC vulnerability detection

Checks for vulnerabilities in the pipeline during builds

SAST

Scans the source code for security vulnerabilities

Focuses on the source code only

Provides security for the development phases

Checks for vulnerabilities automatically during new code commits

Benefits of Integrating SCA with Jenkins

Early detection of third-party vulnerabilities

Enhanced operational efficiency

Timely checks for license compliance to avoid legal issues

Detailed reporting and traceability of the security issues

Automated security checks during every builds in the pipeline

SCA Scanning Tools

The following are the popular SCA scanning tools that can be integrated within a Jenkins pipeline:

snyk

BLACKDUCK
BY SYNOPSYS®

SOOS

debricked
by opentext™

mend.io

Assisted Practice

Integrating SCA tools into Jenkins for enhanced vulnerability detection

Duration: 15 Min.

Problem statement:

You have been assigned a task to automate SCA scans by integrating the Snyk plugin with Jenkins, enhancing the efficiency of vulnerability detection within Jenkins build jobs.

Outcome:

By completing this demo, you will gain proficiency in setting up the Snyk plugin in Jenkins to perform SCA scans. You will configure Snyk and set up a Jenkins job to perform an SCA scan for a Java application.

Note: Refer to the demo document for detailed steps

Assisted Practice: Guidelines

Steps to be followed:

1. Install Snyk and Maven plugins in the Jenkins
2. Configure the Maven and Snyk installations
3. Create a new Jenkins pipeline job

Quick Check

Your development team is building a new web application that relies on several open-source libraries. You're concerned about the potential vulnerabilities in these libraries. Which Jenkins integration can most effectively address this concern?

A. Static code analysis tool

B. SCA plugin

C. Code review workflow

D. Manual vulnerability scanning

Automating DAST in Jenkins Using OWASP ZAP

OWASP ZAP as a DAST Tool

OWASP ZAP is a free and open-source web application security scanner, developed and actively maintained by a dedicated international team of developers.

It is used to find vulnerabilities in web applications, identify misconfigurations, and test web functionality. It supports various protocols such as HTTP, HTTPS, SOAP, REST, FTP, and FTPS.

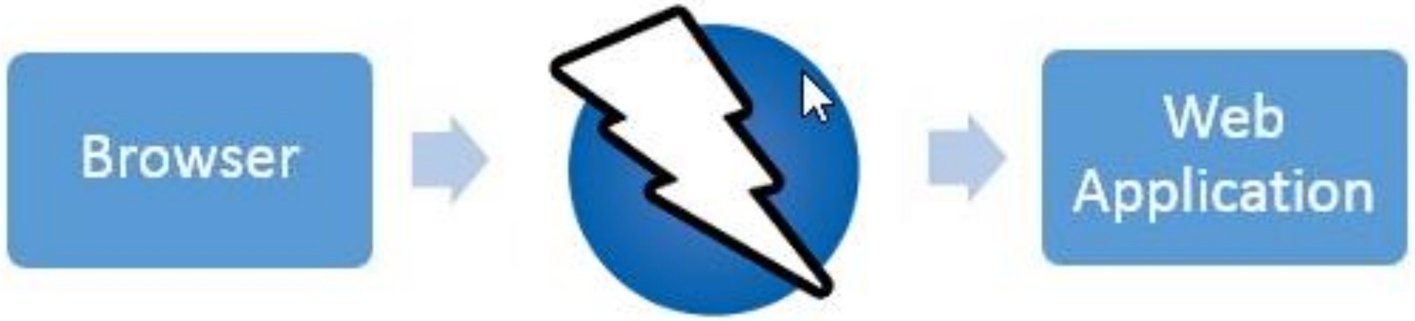
OWASP ZAP as a DAST Tool

ZAP simulates the behavior of an attacker on a web application to function as a DAST tool. Following are ways in which ZAP ensures the dynamic security of an application:

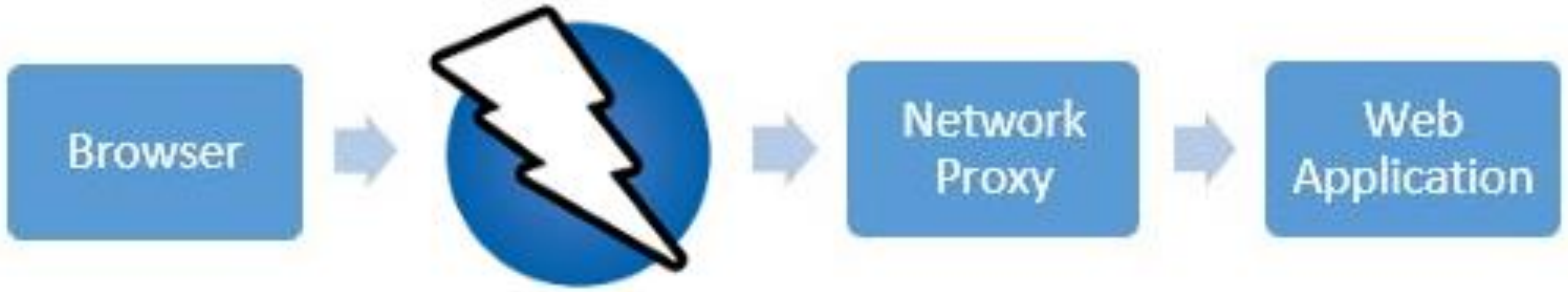
- ZAP crawls the web application with its spider and scans each page to attack them as attackers.
- It tests targeted vulnerabilities, including SQL injection and XSS, by injecting malicious payloads.
- It passively monitors application traffic while acting as a proxy.
- It integrates well with CI/CD pipelines, enabling automated security testing.

How Does OWASP ZAP Work?

ZAP acts as a *man-in-the-middle proxy* placed between the tester's browser and the web application. It intercepts and inspects the messages sent between the browser and the web application.



If there is another network proxy connected between the browser and the web application, ZAP can also be configured to connect to that proxy.



Source: https://www.zaproxy.org/getting-started/

DAST Scan in Jenkins Pipeline Using ZAP

OWASP ZAP tool can be integrated with a Jenkins CI/CD pipeline to automate dynamic application security testing (DAST) for the software development process.

- Automated vulnerability assessment
- Penetration testing
- Security testing from an end-user
- Code review

Following are the specific outcomes that can expected from DAST with ZAP in the Jenkins CI/CD pipeline:

- Security scan reports
- Vulnerability categorization
- Improved secure collaboration

Assisted Practice

Installing and configuring ZAP plugin on Jenkins

Duration: 15 Min.

Problem statement:

You have been assigned a task to install and configure the OWASP ZAP plugin on Jenkins to automate security testing of web applications during the build process.

Outcome:

By completing this demo, you will gain proficiency in setting up the OWASP ZAP plugin in Jenkins to perform DAST scans. You will install and configure OWASP ZAP and set up a Jenkins pipeline job to integrate the vulnerability scan tool.

Note: Refer to the demo document for detailed steps

Assisted Practice: Guidelines

Steps to be followed:

1. Configure OWASP ZAP tool in Jenkins
2. Create a Jenkins pipeline job to integrate the vulnerability scan tool

Assisted Practice

Implementing OWASP ZAP DAST scan in Jenkins pipeline

Duration: 15 Min.

Problem statement:

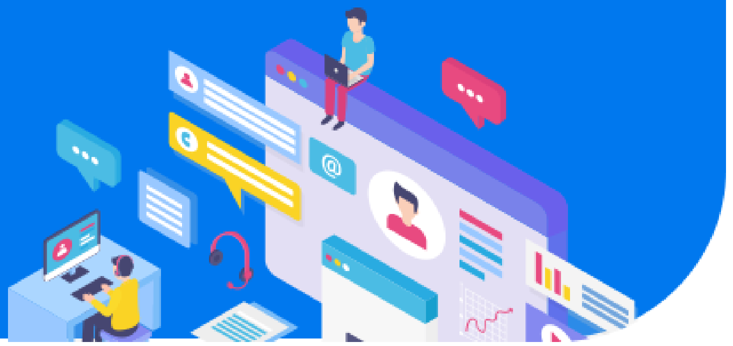
You have been assigned a task to implement the ZAP DAST tool using the Jenkins declarative pipeline to automate code scans using the Jenkins build job.

Outcome:

By completing this demo, you will gain proficiency in using the OWASP ZAP plugin in Jenkins to perform DAST scans. You will implement the ZAP DAST tool using the Jenkins declarative pipeline to automate code scans using the Jenkins build job.

Note: Refer to the demo document for detailed steps

Assisted Practice: Guidelines



Steps to be followed:

1. Create a Jenkins pipeline job to integrate the vulnerability scan tool

Quick Check

You've integrated OWASP ZAP with Jenkins to automate DAST scans within your CI/CD pipeline. Which of the following is an advantage of this approach?

A. Developers can skip security testing entirely because ZAP automates everything.

B. ZAP replaces the need for code reviews, as it can identify all vulnerabilities.

C. Security testing is seamlessly integrated into the development process, enabling early vulnerability detection.

D. OWASP ZAP becomes the single source of truth for application security.

Key Takeaways

- The stages to implement DevSecOps throughout the software development process include secrets analysis, SAST, SCA, DAST, and RASP.
- Secrets analysis in the DevSecOps pipeline specifically focuses on identifying and preventing the exposure of sensitive information.
- Static application security testing (SAST), also known as static code analysis, is a security technique used to identify vulnerabilities in software by analyzing its source code.
- DAST is a black-box testing, that detects runtime security vulnerabilities and weaknesses of an application by injecting malicious payloads.

Key Takeaways

- Software composition analysis (SCA) testing manages risks from third-party and open-source components in software projects.
- OWASP ZAP is a free and open-source web application security scanner which can simulate the behavior of an attacker on a web application to function as a DAST tool.

Thank You