NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Package Vulnerability Scanning

Cloud Infrastructure Engineering

Nanyang Technological University & Skills Union - 2022/2023

Course Content

- Learners will understand:
  - Concepts and importance of package security.
  - Knowledge of best practices for maintaining the security of packages.
- Learners will be able to:
  - Identify and remediate package vulnerabilities.
  - Familiarize with the various tools and techniques for scanning packages for vulnerabilities.

Q1: What is the purpose of Package Vulnerability Scanning?

Q2: What are the common tools used for Package Vulnerability Scanning?

Q3: How does Package Vulnerability Scanning help improve the security of a system?

Activity

Instructor

- Ask to use AWS use single region for all learner for easier monitoring

Package Vulnerability Scan?

What is Package Vulnerability Scan?

- Package vulnerability scanning is the process of analyzing software packages or dependencies for known vulnerabilities that could be exploited by attackers.
- It is an important aspect of software development and deployment because software packages often contain code from third-party libraries or open-source components, and these components can contain vulnerabilities that may go unnoticed by developers.

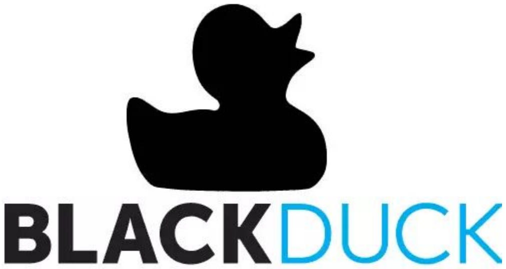
Goals of Package Vulnerability Scan

- The goal of package vulnerability scanning is to identify potential security risks early in the development process and address them before they can be exploited by attackers.
- By proactively identifying and addressing vulnerabilities, developers can ensure that their software is more secure and less vulnerable to attacks.

What kind of Package Vulnerability Scan tools?

- **Code libraries and dependencies**: Applications often rely on third-party libraries and dependencies, which can introduce security vulnerabilities. A vulnerability scan should analyze these libraries to identify any known vulnerabilities.
- **Input validation**: Applications that do not properly validate user input can be vulnerable to attacks such as SQL injection, cross-site scripting (XSS), and command injection.
- **Authentication and authorization**: Improper authentication and authorization can allow attackers to gain access to sensitive information or execute unauthorized actions. A vulnerability scan should assess authentication and authorization mechanisms to ensure they are secure and properly implemented.
- **Docker image scanning:** Docker images may contain vulnerable packages and third-party libraries. A vulnerability scan would be able to identify these vulnerabilities

Tools for Code Libraries & Dependencies



Tools for Input Validation & Authentication

Tools for Image Scanning

snyk

aqua

clair

A Container Image Security Analyzer by CoreOS

Twistlock

What tools are in AWS?

- **AWS CodeGuru Reviewer:** A machine learning-based tool that **analyzes your code** and **provides recommendations** for improving code quality, identifying potential security vulnerabilities, and adhering to best practices. CodeGuru Reviewer is available for Java, Python, and Go applications.

What tools are in AWS?

- **Amazon Inspector:** An automated security assessment service that **analyzes the behavior and configuration of your AWS resources** and **provides a detailed report** of any **vulnerabilities** and **security issues** found. Inspector includes rules packages for several common security frameworks, including CIS AWS Foundations, PCI DSS, and HIPAA.

What tools are in AWS?

- **Amazon GuardDuty:** A **threat detection service** that continuously monitors your AWS account for **suspicious activity** and provides alerts for **potential security threats**. GuardDuty uses machine learning and threat intelligence to identify threats, including vulnerabilities in your AWS resources and instances.

Amazon GuardDuty

What tools are in AWS?

- **AWS Security Hub:** A central hub for managing and monitoring security across your AWS accounts and resources. Security Hub integrates with other AWS security services and **provides a unified view of security findings and compliance status.**

Vulnerability Scanning Techniques

There are several ways for us to conduct vulnerability scanning:

- **Code Review**
- **Metadata Analysis**
- **Automated Scanning**
- **Package Signature Verification**
- **Threat Intelligence**
- **Penetration Testing**

Code Review

A manual code review of the source code for a **package to identify any security weaknesses or vulnerabilities**.

Popular tools for Code Review:

- **SonarQube**: An open-source tool for continuous code quality inspection that includes code review and analysis features, such as detecting code smells, security vulnerabilities, and coding standards violations.
- **Crucible**: A code review tool from Atlassian that allows teams to review code changes, leave comments, and approve changes before they are merged. Crucible integrates with other Atlassian products, such as Jira and Bitbucket.

Metadata Analysis

The examination of package metadata, such as **package descriptions, change logs, and version history, to identify potential security vulnerabilities**.

Popular tools for Metadata Analysis:

- **AWS Config:** A service that enables you to assess, audit, and evaluate the configuration of your AWS resources. AWS Config can help identify potential vulnerabilities and compliance issues by analyzing metadata across your AWS environment.
- **Scout Suite:** An open-source tool for auditing the security and compliance of your AWS environment. Scout Suite analyzes metadata across your AWS resources and provides a detailed report of any vulnerabilities, misconfigurations, or compliance issues.

Automated Scanning

The use of automated tools, such as **static code analyzers and dynamic analysis tools**, to scan packages for vulnerabilities. These tools can **identify potential security weaknesses** in a matter of minutes, making them an effective method for quickly assessing the security of a large number of packages.

Popular tools for Automated Scanning:

- **Qualys:** A cloud-based vulnerability management platform that offers scanning and detection of vulnerabilities across networks, web applications, and cloud environments. Qualys offers real-time reporting and remediation guidance.
- **Burp Suite:** A web application security testing tool that includes a scanner for identifying vulnerabilities such as SQL injection, cross-site scripting, and file inclusion vulnerabilities. Burp Suite also includes a proxy for intercepting and modifying HTTP requests, and a suite of other tools for web application security testing.

Package Signature Verification

The use of digital signatures to verify the authenticity of a package and to ensure that it has not been modified since it was published.

Threat Intelligence

The use of threat intelligence sources to identify known vulnerabilities in packages and to determine the severity of the risks they pose.

Popular tools for Threat Intelligence:

- **NVD (National Vulnerability Database)**: Maintained by the US government's National Institute of Standards and Technology (NIST), the NVD is a database of known vulnerabilities in software packages. The NVD includes information about the severity of vulnerabilities, as well as recommendations for mitigation.

Penetration Testing

The use of simulated attacks to **identify security vulnerabilities in packages** and to determine how they can be exploited.

Popular tools for Penetration Testing:

- **Nessus:** A vulnerability scanner that can be used to identify vulnerabilities in operating systems, applications, and network devices. Nessus includes a database of known vulnerabilities and can be used to assess the severity of each vulnerability.

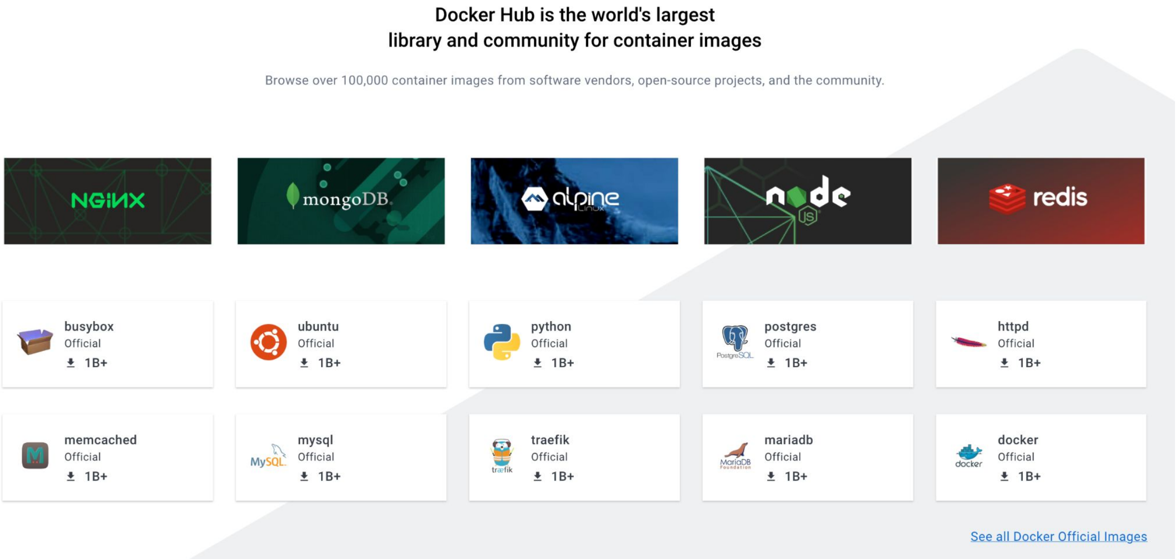
Best Practices

Best Practices Summarised

- Securing the Supply Chain
- Continuous Monitoring
- Updating Packages
- Documenting Vulnerabilities
- Remediating Vulnerabilities
- Implementing Access Controls
- Testing
- Encryption

Securing the Supply Chain

- Ensuring that packages used by an organization are obtained from trusted sources and are free from tampering.



Docker Hub is the world's largest
library and community for container images

Browse over 100,000 container images from software vendors, open-source projects, and the community.

busybox
Official
1B+

ubuntu
Official
1B+

python
Official
1B+

postgres
Official
1B+

httpd
Official
1B+

memcached
Official
1B+

mysql
Official
1B+

traefik
Official
1B+

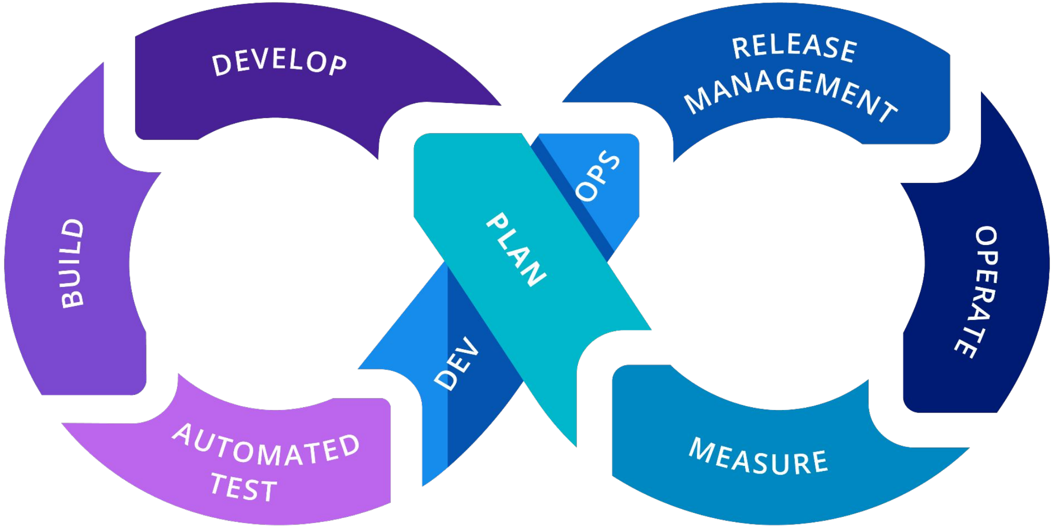
mariadb
Official
1B+

docker
Official
1B+

See all Docker Official Images

Continuous Monitoring

- Regularly scanning packages for vulnerabilities and monitoring for new threats to ensure that packages remain secure over time.

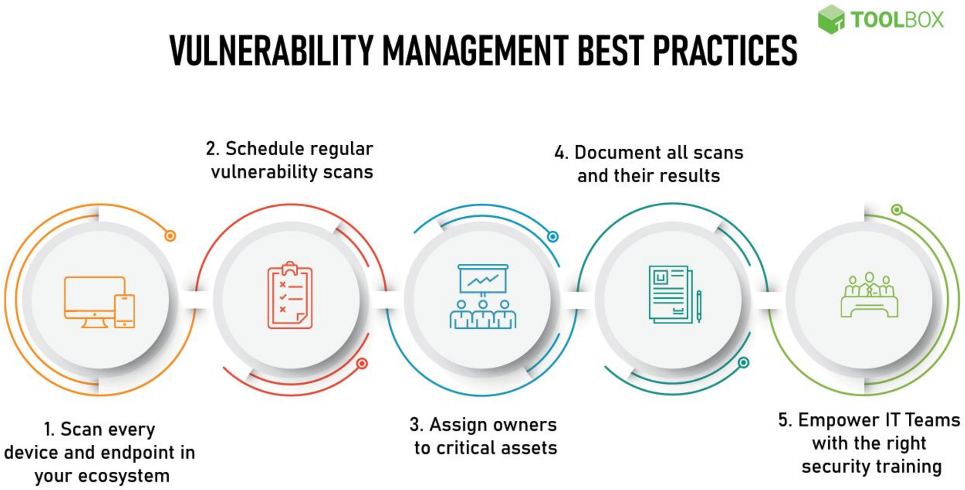


Updating Packages

- Regularly updating packages to ensure that security vulnerabilities are patched in a timely manner.

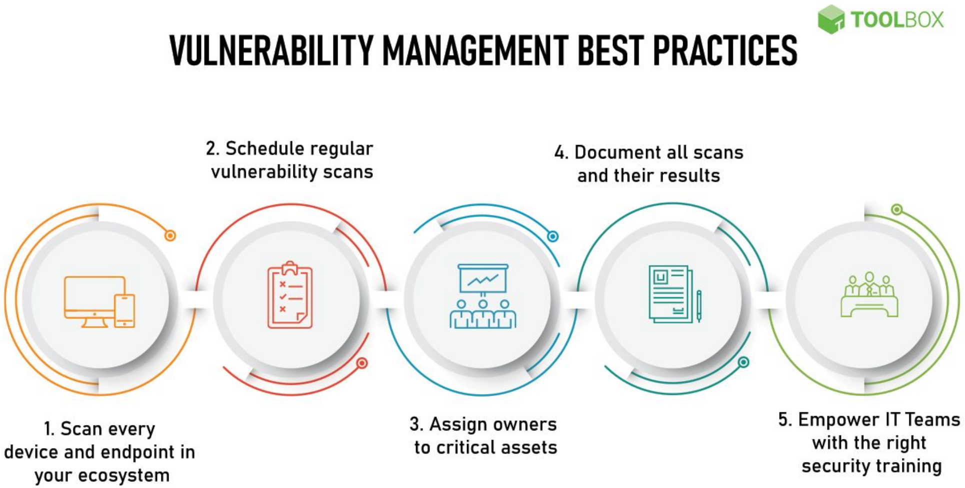
Documenting Vulnerabilities

- Maintaining a comprehensive record of the vulnerabilities that have been identified in packages, including details on the impact and remediation steps taken.


TOOLBOX
VULNERABILITY MANAGEMENT BEST PRACTICES
1. Scan every device and endpoint in your ecosystem
2. Schedule regular vulnerability scans
3. Assign owners to critical assets
4. Document all scans and their results
5. Empower IT Teams with the right security training

Remediating Vulnerabilities

- Taking action to address security vulnerabilities in packages, such as updating, patching, or mitigating the vulnerability.

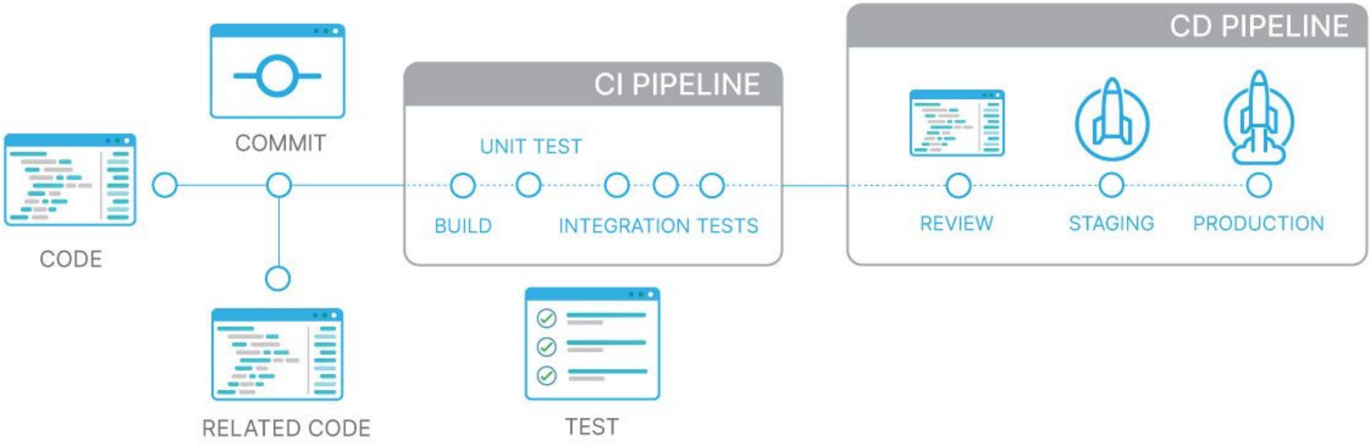

TOOLBOX
VULNERABILITY MANAGEMENT BEST PRACTICES
1. Scan every device and endpoint in your ecosystem
2. Schedule regular vulnerability scans
3. Assign owners to critical assets
4. Document all scans and their results
5. Empower IT Teams with the right security training

Implementing Access Controls

- Restricting access to packages and the systems that use them to only authorized personnel.

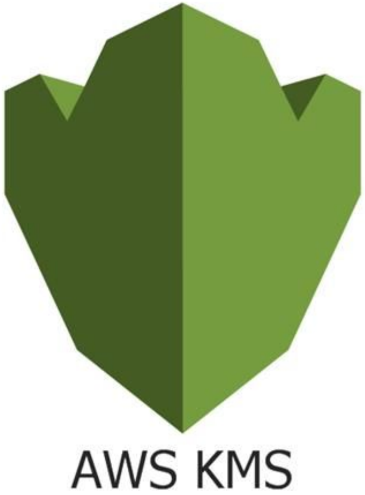
Testing

- Regularly testing packages for security vulnerabilities and conducting penetration testing to identify any weaknesses that may exist.



Encryption

- Encrypting sensitive data that is stored in packages to protect it from unauthorized access.



Best Practices Summarised

- Securing the Supply Chain
- Continuous Monitoring
- Updating Packages
- Documenting Vulnerabilities
- Remediating Vulnerabilities
- Implementing Access Controls
- Testing
- Encryption

Hands-on with Snyk

Pre-Requisites

- Have npm installed on your local machine

What is Snyk?

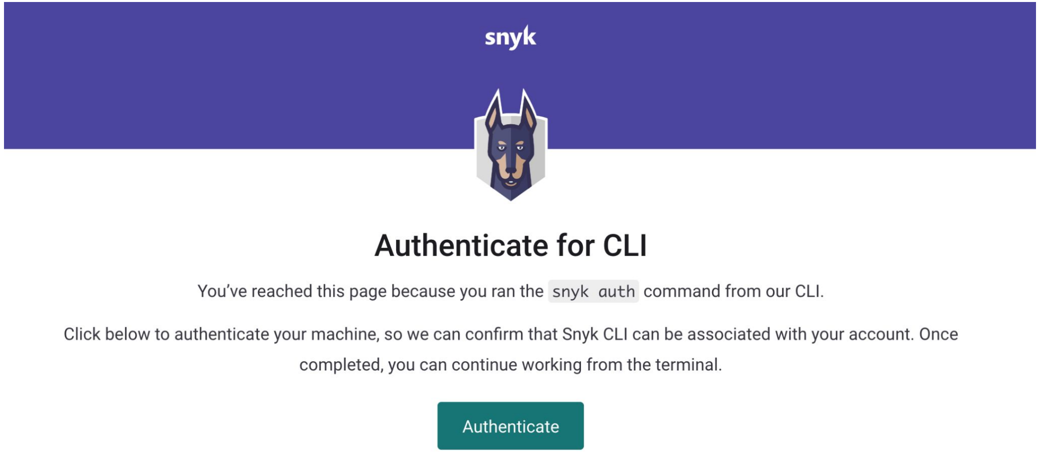
snyk
Vulnerability DB
Docs
My account
Dashboard
Reports
Projects
Integrations
Settings

Steps to install Synk

- Run the below command:
  - sudo npm install -g snyk

Steps to authenticate Snyk

- Run the below command:
  - snyk auth
- Authenticate using an email or github & click “Authenticate”



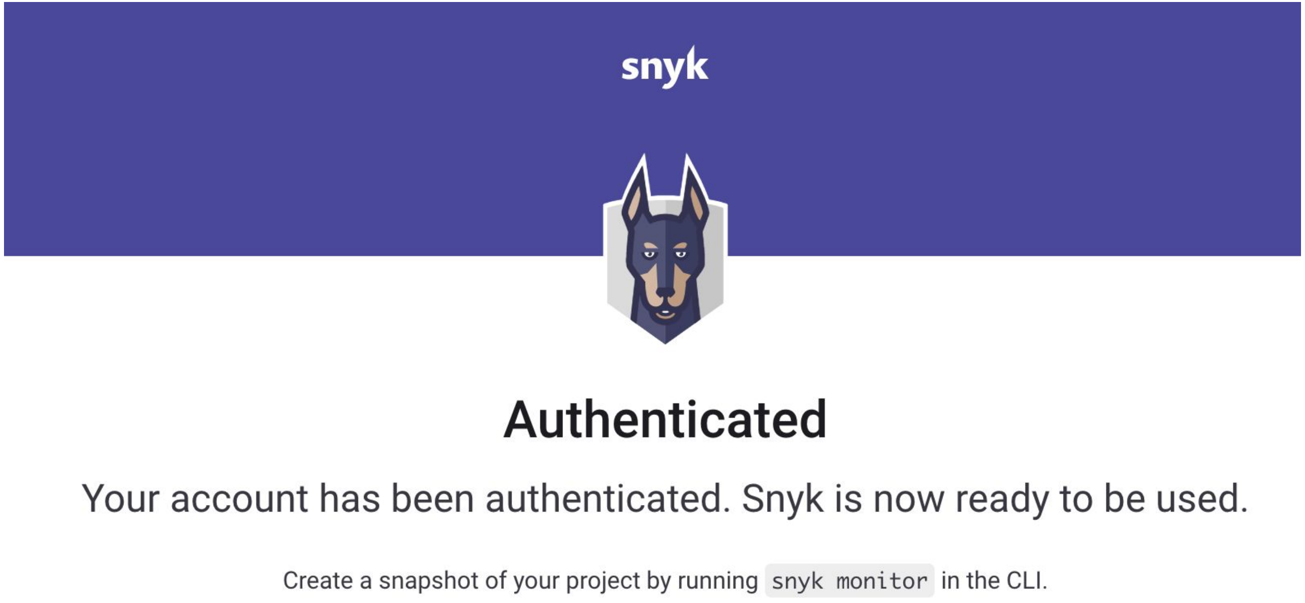
Authenticate for CLI

You’ve reached this page because you ran the `snyk auth` command from our CLI.

Click below to authenticate your machine, so we can confirm that Snyk CLI can be associated with your account. Once completed, you can continue working from the terminal.

Steps to authenticate Snyk

- Once done, you will see this page



Authenticated

Your account has been authenticated. Snyk is now ready to be used.

Create a snapshot of your project by running `snyk monitor` in the CLI.

Steps to test your repo with Snyk

- Navigate to your github repository and run the command below
  - snyk test
  - sudo snyk test

Continuously monitor with Snyk

- Once you have fixed any issues found, you can continuously monitor for vulnerabilities using
  - snyk monitor
  - sudo snyk monitor

Bonus Video

snyk
Developer Security Platform
Snyk Code
Snyk Open Source
Snyk Container
Snyk IaC
Developer Experience
Application Intelligence
Security Intelligence
Empowerment
Extensibility
Governance

Assignment

Embed into CI/CD pipeline

- Your task today would be to include Snyk or any third-party package vulnerability tools in your CI/CD pipeline.
- Explore how you can do this efficiently given the existing github actions file.
- Some guide:
  - https://docs.snyk.io/integrations/ci-cd-integrations/github-actions-integration
  - https://snyk.io/blog/building-a-secure-pipeline-with-github-actions/
  - https://dev.to/adafycheng/ci-with-snyk-using-github-actions-5f51

Activity

Learner:

- Clean up AWS.
- Remove/delete/terminate all service/ resources that you created.

Instructor

- Clean up AWS.
- Remove/delete/terminate all service/ resources that you created.
- Check the AWS account after learner clean up.

What’s Next?