Trivy Scan Guide

Trivy Introduction

Trivy is a comprehensive and versatile security scanner designed to identify security issues across various targets.

For more information see: https://github.com/aquasecurity/trivy

Targets (what Trivy can scan):

- Container Image
- Filesystem
- Git Repository (remote)
- Virtual Machine Image
- Kubernetes
- AWS

Scanners (what Trivy can find there):

- OS packages and software dependencies in use (SBOM)
- Known vulnerabilities (CVEs)
- IaC issues and misconfigurations
- Sensitive information and secrets
- Software licenses

In this guide, we will be using the filesystem scanning feature to detect vulnerabilities, secrets, and software licenses. If a different target type is more suitable for your needs, please refer to the Trivy documentation available online.

Known Limitations:

- Trivy will not correctly scan for software licenses in .Net projects that use project files (e.g., `ExampleProject.csproj`) to track dependencies.
- In this case, please generate a **transitive** list of dependencies using the appropriate .Net command (such as `dotnet list package --include-transitive`), and send the list to us.
- Alternatively, collect all the project files and send those to us for further analysis.

Getting Started

Trivy is available through most common distribution channels. You can find a full list of installation options on the Installation page. Trivy is updated regularly. We recommend downloading the latest binary from https://github.com/aquasecurity/trivy/releases/latest/.

For optimal scanning, we suggest organizing all of your software packages, including dependencies, into subfolders under a main folder named “package-scan.”

Important: Trivy scans can be CPU-intensive, so we recommend proceeding with caution. It’s advisable to avoid running scans on production systems or to schedule them outside of peak processing times.

Linux Installation

In this example, we will download the ***trivy_0.51.2_Linux-64bit.deb*** package and install it using the ***apt*** installer. Note that you must have an Internet connection to perform these steps.

1. Log into the shell of your Linux system.

2. Download the Trivy package with the command:

```
wget
https://github.com/aquasecurity/trivy/releases/download/v0.51.2/trivy_0.51.2_Linux-64bit.deb
```

3. Use apt to install the package, trivy_0.51.2_Linux-64bit.deb, with this command:

```
sudo apt install ./trivy_0.51.2_Linux-64bit.deb
```

Windows Installation

For Windows systems, download the Windows zip release from https://github.com/aquasecurity/trivy/releases/latest/ and extract the zip file.

Running Project Scans

Trivy can be used to scan both Linux and Microsoft Windows-based systems directly from the command line.

Scan Durations:
Execution times vary based on CPU performance. The following are general guidelines:

- **Known vulnerabilities (CVEs)**: Execution time should be under 1 minute for 75,000 files.
- **Sensitive information and secrets**: Execution time should be under 5 minutes for 75,000 files.
- **Software licenses**: Execution time should be between 10 to 25 minutes for 75,000 files.

Linux Project Scans

We will be performing multiple scan types on the “package-scan” folder and subfolders. **Please note that each command is a single line, no carriage returns**.

1. Scan for known vulnerabilities (CVEs) with the command:

```
sudo trivy fs --timeout 30m --scanners vuln -o trivy_fs_vuln.txt -d ./package-scan 2>&1 | tee trivy_vuln.log
```

2. Scan for sensitive information and secrets with the command:

```
sudo trivy fs --timeout 30m --scanners secret -o trivy_fs_secret.txt -d ./package-scan 2>&1 | tee trivy_secrets.log
```

3. Scan for software licenses with the command:

```
sudo trivy fs --timeout 30m --scanners license --license-full -o trivy_fs_license-full.txt -d ./package-scan 2>&1 | tee trivy_license-full.log
```

Windows Project Scans

We will perform multiple scan types on the “package-scan” folder and its subfolders. These commands must be run in Microsoft PowerShell from the directory where the Trivy Windows zip release was extracted, containing the `trivy.exe` file. **Please note that each command is a single line, no carriage returns**.

1. Scan for known vulnerabilities (CVEs) with the command:

```
cmd /c 'trivy fs --timeout 30m --scanners vuln -o trivy_fs_vuln.txt -d c:\package-scan 2>&1' |
tee trivy_vuln.log
```

2. Scan for sensitive information and secrets with the command:

```
cmd /c 'trivy fs --timeout 30m --scanners secret -o trivy_fs_secret.txt -d c:\package-scan
2>&1' | tee trivy_secrets.log
```

3. Scan for software licenses with the command:

```
cmd /c 'trivy fs --timeout 30m --scanners license --license-full -o trivy_fs_license-full.txt
-d c:\package-scan 2>&1' | tee trivy_license-full.log
```

Collecting the Results

When the scans are complete, please upload the resulting output files (txt and log).

Alternatively, you can send the files in an email being certain to compress with a password in order to protect any sensitive data in transit.

Tracking the Use of OSS and COTS Components

Tracking the use of open source software (OSS) and commercial off-the-shelf (COTS) components in commercial products is critical for several reasons, encompassing legal, technical, and security aspects. Specifically, it is important to ensure that intellectual property (IP) is not contaminated or compromised, that there are no significant license compliance-related financial issues, that there is not substantial technical debt due to outdated components, and that there is minimal risk of supply-chain security risks.

1. License Compliance and Associated Liabilities

- **License Compliance**: OSS components come with a variety of licenses, each with its own set of obligations. Some licenses, like the GNU General Public License (GPL), require that derivative works be distributed under the same license, which can have significant implications for a commercial product. Non-compliance with OSS licenses can lead to legal disputes, financial penalties, and even the requirement to release proprietary code as open source.
- **Liabilities of Non-Compliance**: Failing to comply with OSS licenses can result in litigation, damage to a company's reputation, and loss of intellectual property rights. Additionally, violations of commercial licenses (if the product is using COTS components) can lead to breach of contract issues, resulting in financial penalties or loss of access to critical software components.

2. Technical Debt from Outdated Components

- **Technical Debt**: Using outdated OSS or COTS components increases technical debt, which is the cost associated with maintaining and updating these components in the future. Over time, as newer versions of these components are released, the gap between the outdated components and the latest versions widens, making it increasingly difficult and costly to upgrade. This can also lead to compatibility issues with other software, resulting in slower development cycles and higher maintenance costs.
- **Long-term Impact**: The longer outdated components remain in use, the more complex and risky the upgrade process becomes, potentially requiring significant refactoring of the codebase. This technical debt can hinder innovation and slow down the release of new features, impacting the competitiveness of the product.

3. Security Risks from Outdated OSS and COTS Components

- **Security Risks**: Outdated OSS and COTS components often contain published vulnerabilities that can be readily exploited by attackers. If these components are not regularly updated, they create security holes in the software, which can lead to data breaches, unauthorized access, and other cyber threats.
- **Compliance and Legal Ramifications**: Many industries have strict regulations regarding data security and privacy. Failing to address security vulnerabilities in a timely manner can result in non-compliance with these regulations, leading to fines, legal action, and loss of customer trust.
- **Cost of Breaches**: The cost of a security breach can be enormous, not just in terms of financial losses but also in terms of damage to the company's reputation and customer relationships. Regularly tracking and updating OSS and COTS components is a critical aspect of a robust security strategy.

4. Other Risks Mitigated by Consistent Tracking

- **IP Infringement**: Proper tracking ensures that all OSS and COTS components are correctly attributed and that no intellectual property (IP) is inadvertently infringed upon. This can prevent potential lawsuits and the associated costs of IP disputes.
- **Operational Continuity**: Some OSS or COTS components may become obsolete, with no ongoing support or updates. Tracking usage helps to identify these components early, allowing for proactive replacement or migration to supported alternatives, ensuring the long-term viability of the software.
- **Supply Chain Risks**: Commercial software often relies on a chain of third-party components. If one component in this chain becomes compromised, it can affect the entire product. Consistent tracking helps to maintain visibility into these dependencies, allowing for quicker responses to issues such as supply chain attacks.
- **Product Quality and Stability**: Keeping track of OSS and COTS components allows for more structured and planned upgrades, reducing the likelihood of unexpected bugs or stability issues when integrating newer versions.

Security Reassurance: Why Trivy Does Not Upload Your Source Code

One common concern when using open-source scanning tools is the risk of exposing proprietary source code by inadvertently uploading it to the cloud. Trivy addresses this concern by running entirely locally on your machine. No data, including your source code, is transmitted externally to any third-party server or open-source project.

To further ensure the security of your code, Trivy scans can be performed in an offline environment. For example, running the scan on a standalone, isolated laptop that has no network connection guarantees that the data stays entirely within your control. The results, which are output as a simple text file, can then be transferred using a secure method, such as a USB stick. Once the scan is complete, you can wipe the machine, ensuring that no traces of the scan or the code remain.

This method ensures that your proprietary source code remains protected while still allowing the organization to benefit from Trivy's robust scanning capabilities.

Rationale for Performing a Trivy Scan

Using Trivy is essential for maintaining both security and compliance within your software stack. Open-source components are widely used in modern applications, but they can introduce risks. For example, open-source licenses like GPL may require you to make your entire codebase publicly available if not properly managed. By scanning with Trivy, you can identify and address these risks in advance, ensuring that your organization complies with all relevant licenses and avoids any costly or unexpected liabilities.

Moreover, Trivy plays a key role in identifying security vulnerabilities in outdated open-source libraries. Open-source components with known vulnerabilities are often targeted by attackers. If you don't regularly update these components, you leave your system open to exploitation. Trivy helps you stay ahead of these risks, ensuring that all components are up-to-date and secure.

Conclusion

Tracking the use of OSS and COTS components in commercial products is not just a best practice; it is essential for maintaining legal compliance, minimizing technical debt, securing the product, and ensuring operational continuity. Regular and thorough tracking enables companies to mitigate a wide range of risks, from legal liabilities to security vulnerabilities, and supports the ongoing health and competitiveness of the software product.

Additional References

- Troubleshooting https://aquasecurity.github.io/trivy/latest/docs/references/troubleshooting/
- Alternative targets https://aquasecurity.github.io/trivy/latest/docs/
- Trivy Documentation https://aquasecurity.github.io/trivy/latest/