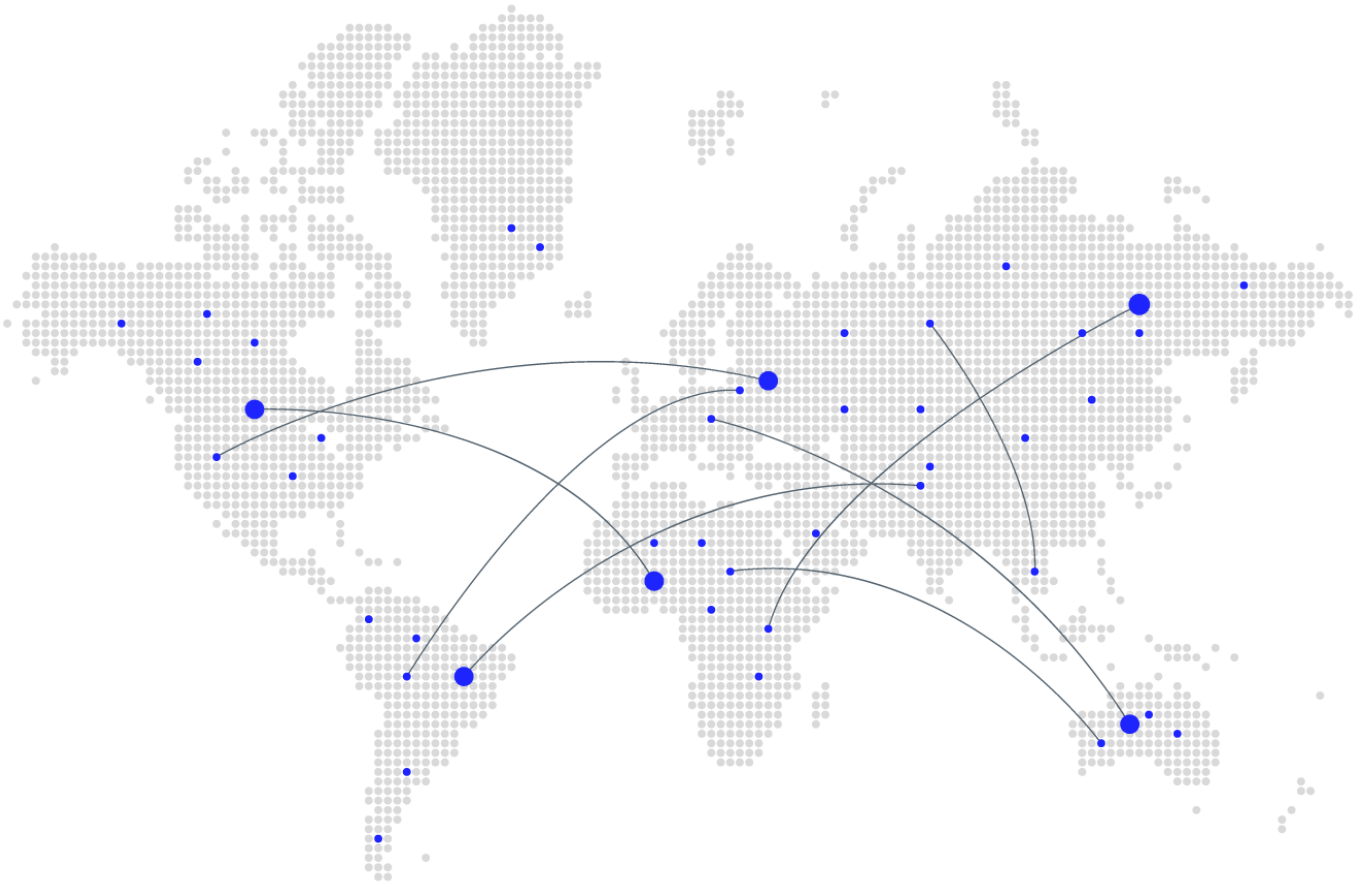
A Software Supply Chain Security Assessment of Prometheus



Chainguard

Published in collaboration with

CLOUD NATIVE COMPUTING FOUNDATION

Prometheus

Date

April 18, 2023

Overview

The Cloud Native Computing Foundation (CNCF) asked staff from Chainguard to assess the software supply chain security practices of Prometheus, the popular open-source monitoring and time-series database system. The assessment team attempted to apply the SLSA framework, described below, to prometheus/prometheus.

The assessment team concluded that the source and build portions of the Prometheus supply chain achieved SLSA (v0.1) level 3. The only area left to address is provenance published alongside all of the Prometheus artifacts which resulted in SLSA (v0.1) level 0. This SLSA assessment effort builds on the security work CNCF has been doing with independent security audits with OSTIF and fuzzing audits with ADA Logics and addresses a crucial aspect of security health in the software supply chain.

What is SLSA?

Supply-chain Levels for Software Artifacts (SLSA, pronounced *salsa*) is a framework for software supply chain integrity. With roots in Google's internal practices and now housed under the umbrella of the Open Source Security Foundation, SLSA defines levels of software supply chain security and a set of practices to achieve these levels.

Version 0.1 of SLSA (at the time of writing, a 1.0 specification has been announced) emphasizes a set of software supply chain security practices that deal with source code, the build process, and provenance. Further information on the 0.1 SLSA specification can be found here: https://slsa.dev/spec/v0.1/requirements

To be sure, SLSA does not cover all aspects of software supply chain security (notably vulnerability management among dependencies) and the framework is admittedly a work-in-progress. That said, the growing momentum behind SLSA and our belief that the application of this framework to the Prometheus organization could expose the strengths and weaknesses of SLSA motivated us to employ SLSA.

What is provenance?

One of the important artifacts that SLSA looks for is a provenance document. Provenance is more than just a signature for a specific software artifact. It is a document representing how a software artifact is built. Signatures on artifacts simply

state that whoever signed the artifact was in possession of it at one point. The purpose of a provenance document is to make an assertion about how an artifact was built and what dependencies were pulled into it. Provenance documents are meant to give consumers assurance that the artifact they are consuming was built where and how the authors claim it was built. For more information please see the slsa.dev site.

Assessment of the Prometheus Supply Chain

Prometheus is this section's shorthand for the main prometheus/prometheus project.

Current SLSA Practices of Prometheus

Prometheus is a popular CNCF project that already has strong security practices such as two-factor authentication for maintainers, one-person review for pull requests, and branch protection rules. The organization also has rules to lock down access for key repositories to maintainers and admins. While these are good security practices and the SLSA framework does take them into account, the following sections focus more on the SLSA specific practices.

Source

The Prometheus organization uses git and GitHub for their version control system, and has strict rules that maintainers are the only ones allowed to merge pull requests (PR's). All of the changes to the Prometheus source code are done transparently, with PR's getting reviewed and merged by strongly authenticated maintainers. Prometheus enforces a single maintainer for approvals on each PR.

Build

The Prometheus build and release process makes use of the GitHub Actions build service. The build is entirely scripted and described as code that sits in the git repository alongside the application code. Changes to the build script occur in the form of a PR that must be reviewed by at least one strongly authenticated maintainer before it is allowed to merge and take effect on the next release. The build environment provided by GitHub Actions is ephemeral in nature. Prometheus builds do make use of a cache but the cache is only used for CI testing and does not affect the release process.

The release process is kicked off by one of the maintainers when a commit is tagged and pushed to the Prometheus repository. The tagged commit sets off the release process in GitHub Actions, which pulls in build scripts from prometheus/promci to build and release the Prometheus artifacts. The GitHub Actions make use of a separate tool

(prometheus/promu) to generate and publish the final Prometheus artifacts. This method of breaking out the GitHub Actions scripts and helper utility is useful for reuse across the Prometheus ecosystem, but it does expand the supply chain dependencies for building the application.

The assessment team found no evidence of hermetic builds or reproducible build artifacts.

Provenance

The assessment team found no evidence of provenance within the Prometheus repository. This result is not unexpected because SLSA is still being adopted across the industry through utilities such as the slsa-github-generator.

SLSA Assessment

The Prometheus SLSA assessment resulted in table 1 below. Each category of requirements (source, build, provenance, and contents of provenance) are logically separated. The SLSA level associated with each control (i.e., each row) can be found in the columns. The green check marks indicated evidence of compliance with a control; red boxes indicate the assessment team's inability to find evidence of compliance with a control.

	SLSA Levels			
Source Requirements	*1*	*2*	*3*	*4*
Version controlled	✓	✓	✓	✓
Verified history	✓	✓	✓	✓
Retained indefinitely	✓	✓	✓	✓
Two-person reviewed				
Build Requirements	*1*	*2*	*3*	*4*
Scripted build	✓	✓	✓	✓
Build service	✓	✓	✓	✓
Build as code	✓	✓	✓	✓
Ephemeral environment	✓	✓	✓	✓
Isolated	✓	✓	✓	✓
Parameterless	✓	✓	✓	✓
Hermetic				
Reproducible				
Provenance	***1***	***2***	***3***	***4***

Available	■	■	■	■
Authenticated		■	■	■
Service generated		■	■	■
Non-falsifiable			■	■
Dependencies complete				■
Contents of Provenance	***1***	***2***	***3***	***4***
Identifies artifact	■	■	■	■
Identifies builder	■	■	■	■
Identifies build instructions	■	■	■	■
Identifies source code		■	■	■
Identifies entry point			■	■
Includes all build parameters			■	■
Includes all transitive dependencies				■
Includes reproducible info				■
Includes metadata	■	■	■	■

Table 1. SLSA Assessment Results for Prometheus

Note: A checkmark inside a green box indicates the existence of the practice. A red box without the checkmark indicates that the assessment team did not find evidence of this practice.

SLSA Assessment Justifications

Source requirements

Version controlled
The Prometheus repository uses git and GitHub to manage source code.

Verified history
The Prometheus repository uses git and GitHub to manage source code.

Retained indefinitely
The Prometheus repository uses git and GitHub to manage source code.

Two-person reviewed
The assessment team found evidence of only one reviewer required for each pull request.

Build requirements

Scripted build

The Prometheus team uses GitHub Actions to script their build process.

Build service
The Prometheus team uses GitHub Actions as the build service.

Build as code
The Prometheus team manages the GitHub Actions build scripts in the git repository with the code for Prometheus as well as a shared GitHub Actions repository.

Ephemeral environment
The Prometheus team does not maintain their own build nodes, instead they use GitHub Actions build agents which are provisioned for each build.

Isolated
The Prometheus team does not use a build cache across releases. Builds cannot influence each other.

Parameterless
The Prometheus build and release process is parameterless. It's set in motion by tagging and pushing a commit, then there is no manual intervention.

Hermetic
The assessment team found no evidence of hermetic builds.

Reproducible
The assessment team found no evidence of reproducible builds.

Provenance requirements

Available
The assessment team found no evidence of provenance within the Prometheus repository. For this reason, the rest of the provenance requirement justifications are omitted.

Result and Recommendations

The assessment of the Prometheus software supply chain yielded a result of **SLSA (v0.1) Level 0.**

The assessment of Prometheus yielded SLSA Level 3 for both Source and Build sections. The assessment did not find, however, provenance for the published artifacts, which resulted in SLSA Level 0 for Provenance. Provenance is an important piece of the supply chain that allows the consumers of an artifact to verify its authenticity. The assessment team recommends that the Prometheus maintainers implement provenance generation within the prometheus build infrastructure. The assessment team also recommends that the Prometheus maintainers implement provenance generation throughout the rest of the projects under the Prometheus organization. Finally, once the SLSA v1.0 specification is out, the assessment team recommends that the Prometheus maintainers review the spec and ensure they are in compliance with any changes.