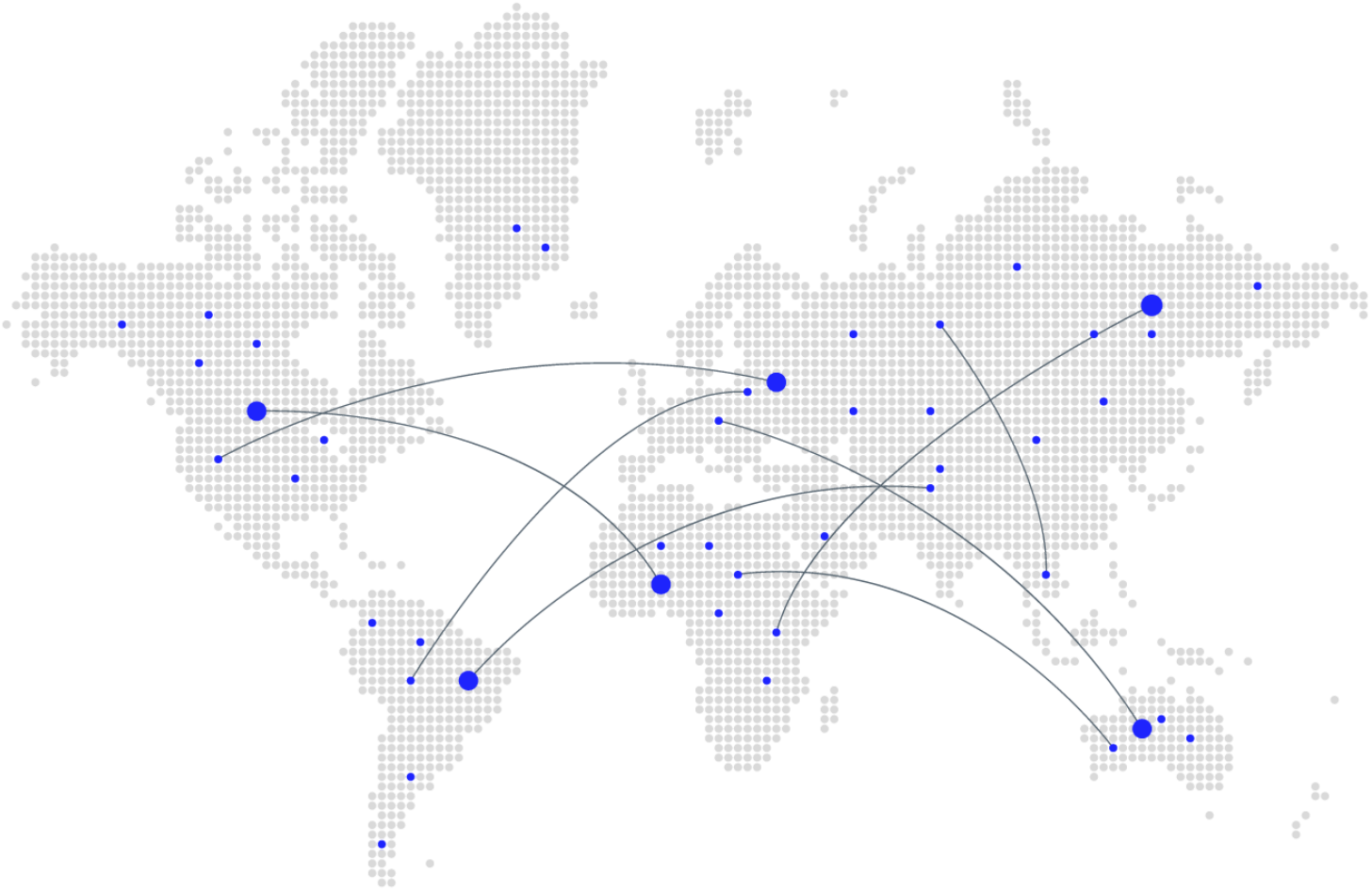
A Software Supply Chain Security Assessment of Argo CD



Chainguard

Published in collaboration with

CLOUD NATIVE COMPUTING FOUNDATION

argo

Date

April 18, 2023

Overview

The Cloud Native Computing Foundation (CNCF) asked staff from Chainguard to assess the software supply chain security practices of Argo CD, the popular open-source continuous deployment project for Kubernetes. The assessment team attempted to apply the SLSA framework, described below, to argoproj/argo-cd.

The assessment team concluded that the source, build, and provenance portions of the Argo CD supply chain achieved SLSA (v0.1) level 3. The Argo CD maintainers were already in the process of improving their SLSA levels by adding provenance and signing throughout their release objects. At the time of writing provenance exists for the Argo CD artifacts. More on this in the Assessment of the Argo CD Supply Chain section.

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 Argo CD 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 Argo CD Supply Chain

Argo CD is this section's shorthand for the main argoproj/argo-cd project.

Current SLSA Practices of Argo CD

Argo CD 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 Argo 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 Argo CD source code are done transparently, with PR's getting reviewed and merged by strongly authenticated maintainers. Argo CD enforces a single maintainer for approvals on each PR.

Build

The Argo CD 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. Changes to the build script occur in the form of a PR, which 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. Argo CD builds also do not share a cache and are isolated from each other meaning one build cannot influence another.

The release process happens in a couple steps. First, a GitHub Action takes two parameters as input: the target branch and version of software to build. These input parameters generate a pull request that must be reviewed and merged by a maintainer. The final build and release process is kicked off by one of the maintainers when a commit is tagged and pushed to the Argo CD repository. The tagged commit sets off the release process in GitHub Actions, which generates all the Argo CD artifacts, signs the artifacts, generates provenance, then finally publishes all artifacts along with their signatures and build provenance.

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

Provenance

As of Argo CD release v2.7.0-rc1 provenance is available for all published Argo CD artifacts. The provenance is generated within the Argo CD builds using the slsa-github-generator GitHub Action. Provenance data is generated for both the binary releases as well as the container images. The provenance for the binaries lives in the GitHub release and the provenance for the container images lives in the registry. Generating provenance data with the slsa-github-generator Action provides SLSA Level 3 compliance for provenance documents.

SLSA Assessment

The Argo CD 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 Argo CD

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 Argo CD repository uses git and GitHub to manage source code.

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

Retained indefinitely
The Argo CD 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 Argo CD team uses GitHub Actions to script their build process.

Build service
The Argo CD team uses GitHub Actions as the build service.

Build as code
The Argo CD team manages the GitHub Actions build scripts in the git repository with the code for Argo CD.

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

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

Parameterless
The Argo CD release process has one step with input parameters, however this step simply generates a pull request which must be reviewed by the maintainers before merging. The rest of the Argo CD release process is parameterless.

Hermetic
The assessment team found no evidence of hermetic builds.

Reproducible
After speaking with the Argo CD maintainers the assessment team concluded the Argo CD builds are not reproducible.

Provenance requirements

Available
As of Argo CD v2.7.0-rc1 the Argo CD team publishes provenance data with all release artifacts.

Authenticated
The provenance generated by the Argo CD build system uses the SLSA provenance generator which makes use of the Sigstore signing service to sign the provenance and store the signature data in a transparency log.

Service generated
The Argo CD team uses GitHub Actions build service to generate the provenance data.

Non-falsifiable
The provenance generated by the Argo CD build system uses the SLSA provenance generator which makes use of the Sigstore signing service to sign the provenance and store the signature data in a transparency log.

Dependencies complete
The assessment team found no evidence of complete dependency list in the provenance.

Requirements on the contents of the provenance

Identifies artifact
The Argo CD provenance documents identify each artifact in the release with a SHA256 hash.

Identifies builder
The Argo CD provenance documents identify the builder as GitHub Actions.

Identifies build instructions
The Argo CD provenance documents identify the build instructions as the following GitHub Action Workflow: `.github/workflows/release.yaml` for the cli artifacts and `.github/workflows/image.yaml` for the images.

Identifies source code
The Argo CD provenance documents identify the Argo CD source code used for the build. The identification is observed and added by the provenance generator itself.

Identifies entry point
The Argo CD provenance documents identify the entry point as the following GitHub Action Workflow: `.github/workflows/release.yaml` for the cli artifacts and `.github/workflows/image.yaml` for the images.

Includes all build parameters
The Argo CD provenance documents identify an empty build parameter list.

Includes all transitive dependencies
The assessment team found no evidence of transitive dependencies in the Argo CD provenance document. However, the Argo CD maintainers publish an SBOM alongside the provenance document which contains transitive dependencies. While the SBOM may not satisfy the letter of the SLSA specification, it satisfies the spirit of identifying transitive dependencies.

Includes reproducible info
The Argo CD provenance documents contain a boolean and identifies that the build is not reproducible.

Includes metadata
The Argo CD provenance documents do contain metadata identifying the build.

Result and Recommendations

The assessment of the Argo CD software supply chain yielded a result of **SLSA Level 3.**

The software supply chain practices displayed by the Argo CD team were enough to achieve SLSA Level 3. The assessment team recommends that the Argo CD maintainers keep up the good work with Argo CD and spread the generation of provenance data to other repositories within the Argo Organization. Once the SLSA v1.0 specification is released, the assessment team recommends that the Argo CD maintainers review the spec and ensure they are in compliance with any changes.