Venafi

SOLUTION BRIEF

# Securing Code Signing in CI/CD Pipelines

**Venafi CodeSign Protect enables DevOps teams to securely sign artifacts without slowing down their build process**

**Who should read this:**
InfoSec and engineering professionals who need to support secure DevOps code signing practices during development and deployment.

DevOps CI/CD pipelines are usually completely automated, running multiple times a day. Code signing the artifacts that are used and produced by these pipelines needs to be automated as well. Although the signing of code is usually automated in some fashion, the *securing* of code signing credentials, keys and certificates is not—leaving an organization vulnerable to a wide range of attacks.

In light of recent software supply chain attacks such as SolarWinds (2020) and ASUS (in 2018), it is imperative that code signing systems, as well as the credentials and signing keys they use, be protected and secured.

Venafi has developed a unique solution that secures code signing credentials, creates an irrefutable audit trail of all code signing operations, and does not impede on performance, scalability or speed of CI/CD pipelines.

This paper describes best practices for how to secure code signing with CI/CD pipelines, containers and orchestration systems and how Venafi CodeSign Protect does it.

## Challenges

### Secured Code Signing at DevOps Speed

A common problem with securing code signing for CI/CD pipelines is that authorizing a system to sign, waiting for a signing operation to complete or waiting for an approval to be obtained manually can slow down or stop an automated build. It is important to secure code signing without sacrificing speed and performance.

### Lack of Comprehensive Security Policy Across Build and Delivery Pipelines

As recent software supply chain attacks have shown, code signing only at the end of the software build pipeline is insufficient because attackers can "jump left" of this and insert malware earlier into the software build process.

### Does Not Support Developers

Development organizations use diverse software development environments including development platforms (Windows, Linux), software automation tools (Jenkins, Azure Pipelines), software languages (Java, C#, PowerShell scripts) and deliver for different platforms (mobile apps for Android or iOS, containers such as Docker images, programs that execute on Windows or Linux operating systems). When looking across an entire enterprise, a secured code signing solution must support the different needs of all enterprise development teams.

### Does Not Protect Code Signing Credentials

Developers often put sensitive code signing credentials, such as private code signing keys, in areas convenient and accessible to their build automation scripts. However, doing so puts these critical resources at risk for being misused or compromised.

## Securing Code Signing in CI/CD Pipelines

There are several popular pipeline management tools that are used to manage the process of building software including Jenkins, Azure Pipelines and GitLab. These technologies allow developers to move software through the pipeline from source code to production-ready releases. The ability to incorporate technologies, including source code analysis and malware scanning, makes build management and software delivery robust, repeatable and more secure.

Let's look at an example of the pieces involved when code signing is added to a Jenkins automated build:

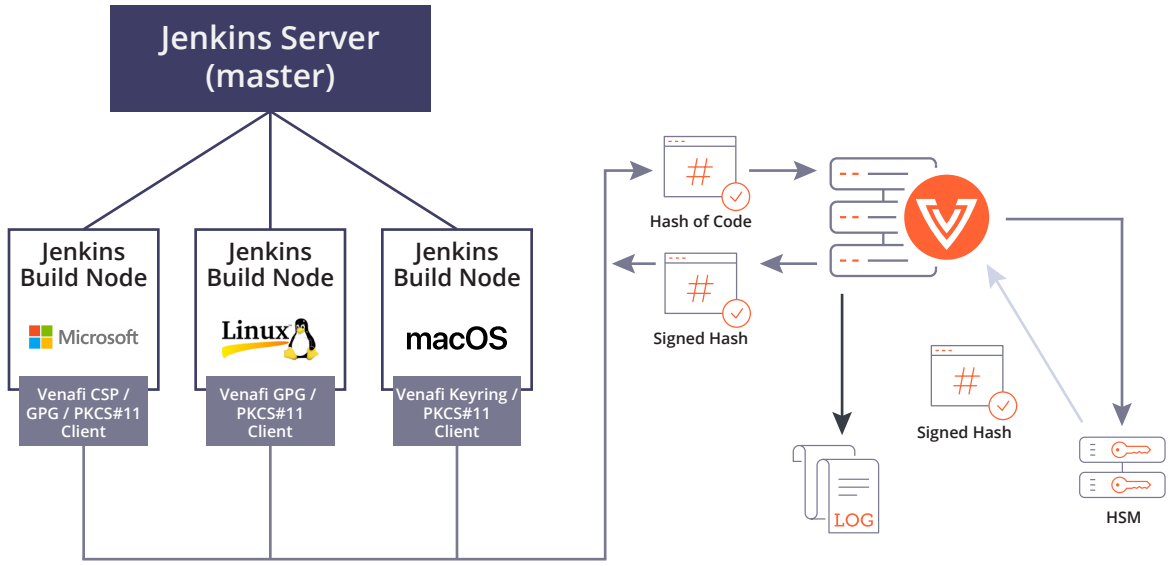


**Figure 1: Securing code signing processes within a Jenkins build pipeline**

Figure 1 illustrates how code signing can be incorporated into a Jenkins build pipeline. It's important to note that it does not matter if the Jenkins build nodes are static or ephemeral. However, there are security benefits when build nodes are ephemeral because they are not easily accessible, reducing the attack surface in the event of a security breach.

Regardless of which approach you take, the idea is to isolate the ability to sign on specific nodes. The Venafi CodeSign Client is agnostic to the overall CI/CD delivery technology. In the above architecture, Jenkins could be substituted with Azure, GitLab, Bamboo, Ansible and others. The Venafi CodeSign Protect client would be configured in the same way in all cases.

You need to address the following important factors when integrating code signing into your pipeline:

1. How is permission granted to the node responsible for code signing?
2. Is this permission and node short-lived, or is it a static system in your pipeline?
3. Can you set controls like code signing approval, audit and revocation of signing in that automated pipeline?

For cases like signing macOS and iOS software, there are not currently methods to containerize this capability because code signing must occur on a macOS system. Since a container is out of the question, organizations must introduce a static system into their pipeline that is responsible for the signing of those binaries. This approach can be combined with other signing capabilities that may be ephemeral in nature. For the static system, the CodeSign Protect client is setup with a system account, and keys for that system are typically restricted so they can only sign based on requests from that specific system. This prevents unauthorized requests from other systems and ensures no unauthorized copies of the keys exist.

In our macOS/iOS example, there are two options for providing the CodeSign Protect client the ability to sign. The first is to install the Venafi Client and authorize a system account to sign on that system. This is considered a persistent installation and signing capability on that system. With this type of setup, it's recommended that signing operations be required to get "approval." Within CodeSign Protect, keys can have approval workflows set, and in this case, pre-approval before each signing operation would be an ideal extra security control to prevent unauthorized use. The second option is to add or revoke the grant on the signing system with each use. A secrets management system would be used to have the end system obtain a grant for signing operations as part of the build/signing process. In the case of Jenkins above, the Jenkins build script could use the secrets management in Jenkins to obtain encrypted credentials to get the grant necessary to sign code, and then revoke that grant once signing is completed.

Many organizations are moving to short-lived containers and want to sign both source code built in those containers, as well as the containers themselves. First, let's look at an example of building a container that can build and sign code inside the container when it is run.

## Using Containers for Signing

Using containers for building software is now just about as common as for deploying software, and the list of container and container orchestration technologies grows on a regular basis. Regardless of whether Docker, Podman, Buildah or another container technology is being used, the approach is consistent across the technologies when you are talking about enabling Venafi CodeSign Protect within a container. In some cases, a CI/CD pipeline will create an ephemeral workload for signing code, and in some cases the system that will be used for signing is a static long-lived system. Let's look at some basics on using an ephemeral container.

In this case we will use Docker as the example. For ephemeral workloads in general, we want to follow industry best practices:

1. Start with a known trusted base image
2. Either select an image with the signing tools already available or add them during the build phase of your container. In the example below, we install openjdk as part of the container build process.
3. Add the Venafi client during the build phase—do not build in credentials or the grant and token used for signing. This should be obtained or passed in at runtime, not during the build process. In the example below, we break this rule on the last line to simplify the example but explain some better processes below.
4. Destroy the grant/token when stopping the running container, either in the container or as part of the decommission statements.

A best practice would be to use a secrets management engine and pass the secret to a RUNNING container (not in the build). A secrets management engine, such as the one used in Kubernetes or HashiCorp Vault, the credentials needed for CodeSign Protect would never exist in a config file or the file system. The secrets engine could either hold the grant used for signing, or a username and password could be obtained and used to obtain the grant.

```
FROM alpine:3.2
RUN apk --update add openjdk7-jre
CMD ["/usr/bin/java", "-version"]
ADD /host/abs/path/to/venafi-codesigningclients-20.4.0-linux-x86_64.rpm /venafi-
codesigningclients-20.4.0-linux-x86_64.rpm
RUN rpm -i venafi-codesigningclients-20.4.0-linux-x86_64.rpm
RUN /opt/venafi/codesign/bin/trust-pkcs11 -force -
hsmurl:https://tpp.venafidemo.com/vedhsm
RUN /opt/venafi/codesign/bin/pkcs11config seturl -
authurl:https://tpp.venafidemo.com/vedauth -hsmurl:https://tpp.venafidemo.com/vedhsm
```

**Figure 2: This is a simple example of having a container built that would have a Java SDK to compile code and the Venafi CodeSign Protect product to sign that code.**

## Securing Early and Often

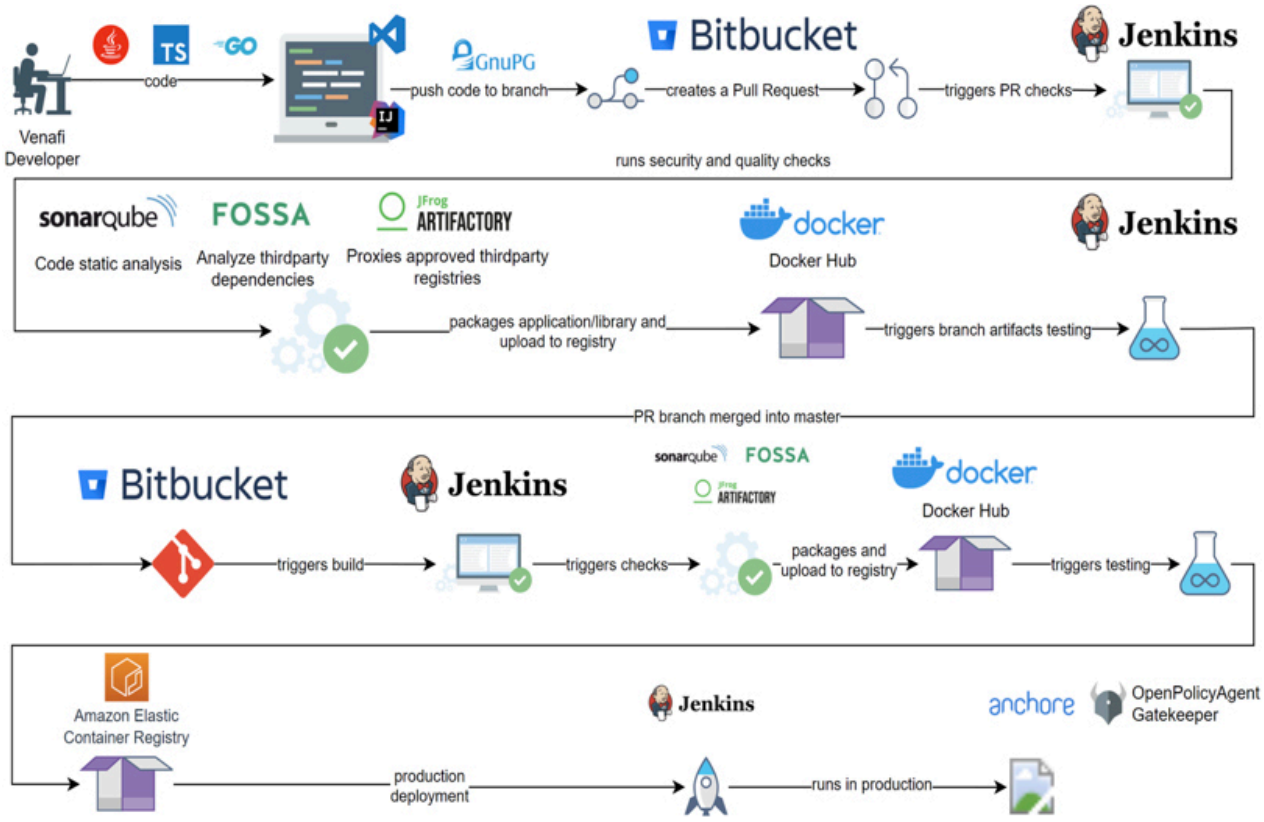


**Figure 3: Secure early and often during a CI/CD pipeline**

As Figure 3 summarizes, you need to code sign artifacts throughout the software development process, not just at the end, to protect your CI/CD pipelines. Artifacts such as source code and intermediate software libraries should be signed to ensure authenticity and prove they have not been tampered with.

## Venafi CodeSign Protect

Venafi CodeSign Protect secures enterprise code signing processes by providing centralized and secure key storage along with role-based policy enforcement. Providing code signing-as-a-service, it reduces the burden on development teams by integrating with the tools and processes they already use.

Venafi CodeSign Protect is specifically architected to support secure DevOps build and delivery pipelines by including such features as:

- Use of standard interfaces that allows automation of all code signing activities in CI/CD pipelines
- Ability to digitally sign individually signed artifacts, such as source code at stages throughout the CI/CD pipeline
- High-performance, available and scalable for large software development organizations
- Enterprisewide visibility for InfoSec teams

To find out more about how Venafi CodeSign Protect can help make your software build and delivery pipelines more secure, visit **venafi.com/platform/codesign-protect**.

## Trusted by

**5 OF THE 5** Top U.S. Health Insurers
**5 OF THE 5** Top U.S. Airlines
**3 OF THE 5** Top U.S. Retailers
**3 OF THE 5** Top Accounting/Consulting Firms
**4 OF THE 5** Top Payment Card Issuers
**4 OF THE 5** Top U.S. Banks
**4 OF THE 5** Top U.K. Banks
**4 OF THE 5** Top S. African Banks
**4 OF THE 5** Top AU Banks

Venafi is the cybersecurity market leader in identity management for machines. From the ground to the cloud, Venafi solutions automate the lifecycle of identities for all types of machines—from physical devices to software applications, APIs and containers. With more than 30 patents, Venafi delivers innovative solutions for the most demanding, security-conscious organizations in the world. **To learn more, visit venafi.com**