Keyfactor
Tech Days
2023

Securing the Dev Ops CI/CD pipeline

Karthik Lalithraj

Director Solution Engineering (East)

Keyfactor

Signing Use Cases

Protect your infrastructure

Prevent unauthorized downloads and ransomware by allowing only trusted applications to run on your infrastructure.

Secure devices and firmware

Secure firmware and firmware updates with digital signatures and verification to prevent unauthorized device access.

Safeguard your software

Safeguard your brand image by ensuring that the software you publish and distribute is signed and that keys are protected.

Trust your containers

Establish trusted and digitally signed base images for containers and virtual machines (VMs) deployed in your environment.

Enable DevSecOps

Establish trust in your CI/CD pipeline, from digitally signing source code checked into repositories to signing packages post-build.

Prevent malicious macros

Prevent unauthorized or malicious macros on internal devices by ensuring that only signed macros are allowed to run.

Code signing maturity model

Level 0: No signing

- Does not currently sign software deliverables
- Vulnerable to unauthorized code access
- No auditability or integrity behind software

Level 1: Fragmented

- Signing software deliverables, but with fragmented tools
- Developers handle their own private keys, without any centralized control
- Signing keys are vulnerable to misuse or compromise

Level 2: Centralized

- Use a centralized platform for code signing policy, workflow, and auditability
- Protect sensitive signing keys in a secure HSM
- Not integrated with CI/CD processes or workflows, not all use cases covered

Level 3: CI/CD-integrated

- Signing containers, artifacts, and software deliverables
- Integrated with native signing tools and workflows
- Full auditability and governance over all signing processes

High risk → Low risk

Typical Risks and roadblocks and what we need in a potential solution

Unprotected keys

Sensitive signing keys are stored on flash drives, build servers, or workstations.

No auditability

And there's no clear audit trail or rules around who has access to keys or who signed what.

Need for speed

Developers just want something that's easy to use and won't change their process.

Protect sensitive keys

Signum generates and store code signing private keys within an HSM, hosted by Keyfactor.

Enforce rules and policies

Signum allows you to define access and usage policies for signing keys, with a full audit trail.

Integrate with native tools

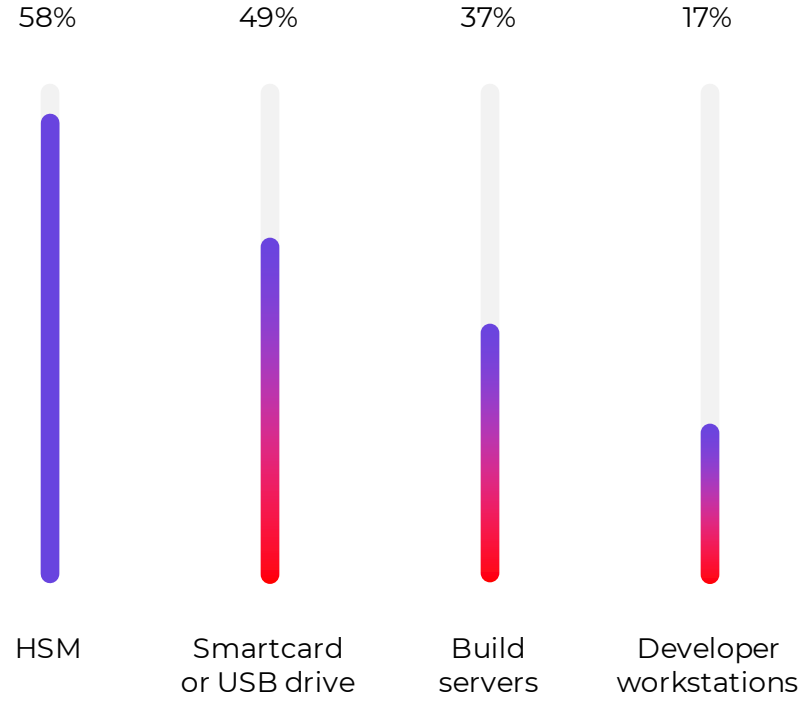
Signum integrates with native signing tools and drops into existing build and delivery processes.

Securing the Keys within the CI/CD framework

Digital Signing Use cases

Securing the sensitive private keys

Question: Where are code signing keys kept in
your organization?



Question: Do you have formal access controls and policies for the use of code signing keys?

50%

No, we don't

47%

Yes, we do

*3% unsure

Question: How are the code signing keys distributed?

Native Integration

Digital Signing Use cases

Use of native signing tools

Native interfaces

Integrate with native signing tools via KSP / PKCS11 interfaces.

Authentication

Integrate with your identity provider (Okta, Azure AD, etc.).

- **SignTool**
  dll, exe, msi, .sys, powershell etc.
- **JarSigner**
  jar, war, tar etc.
- **Office Macros**
  Excel and Word macros created in VBA
- **Container Signing**
  cosign
- **HLK Signing**
  hlk studio

Microsoft

Java

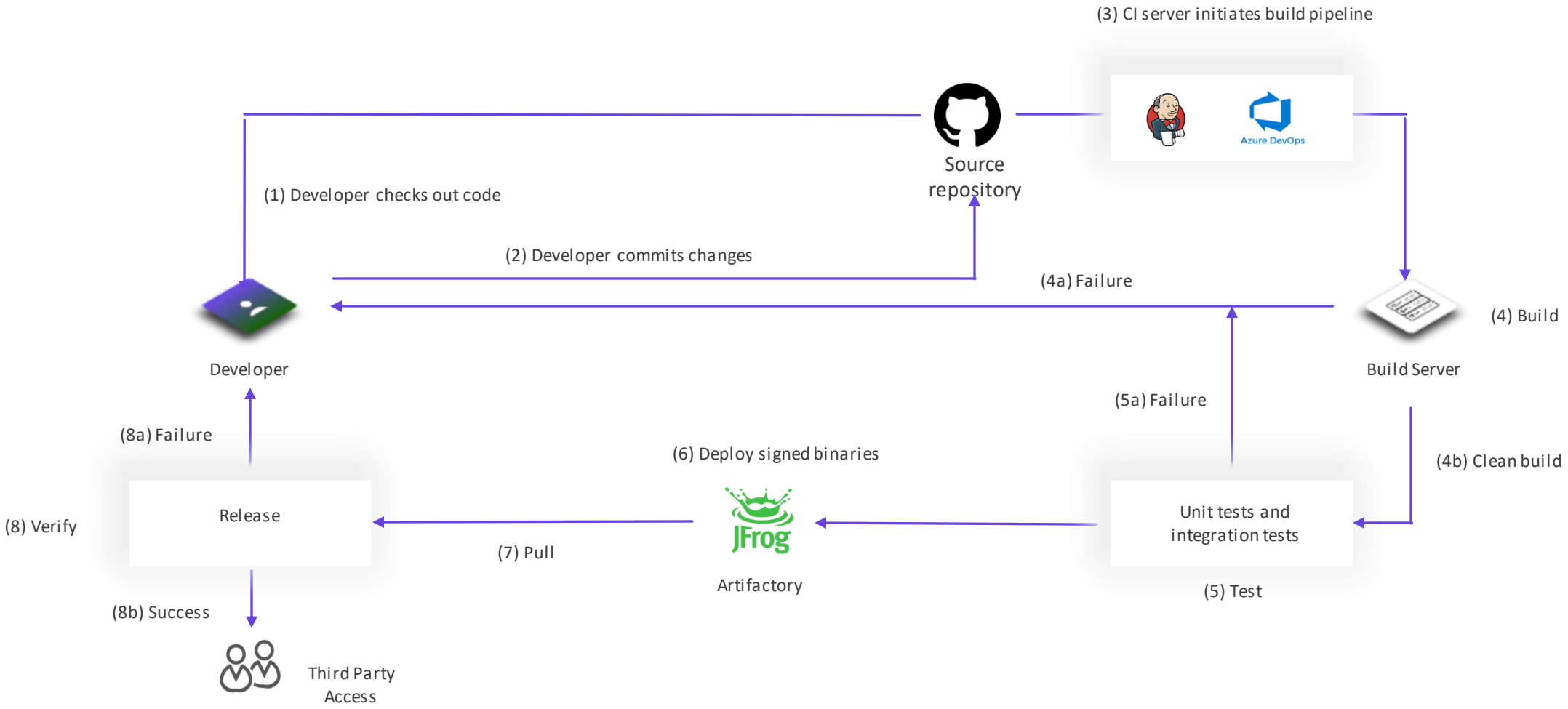
Visual Studio

sigstore cosign

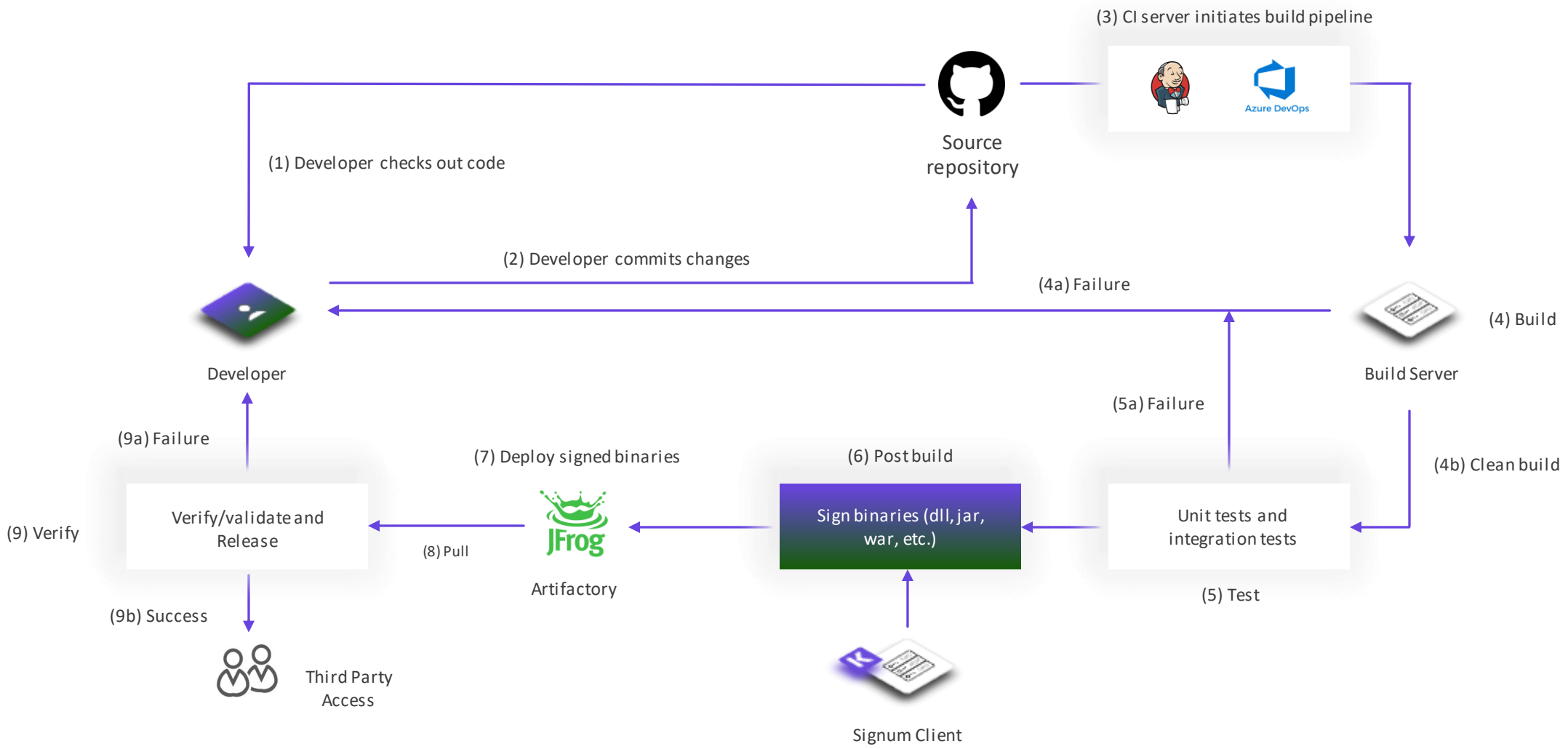
OpenSSL

Integration with CI/CD Pipelines

AS-IS CI/CD pipeline

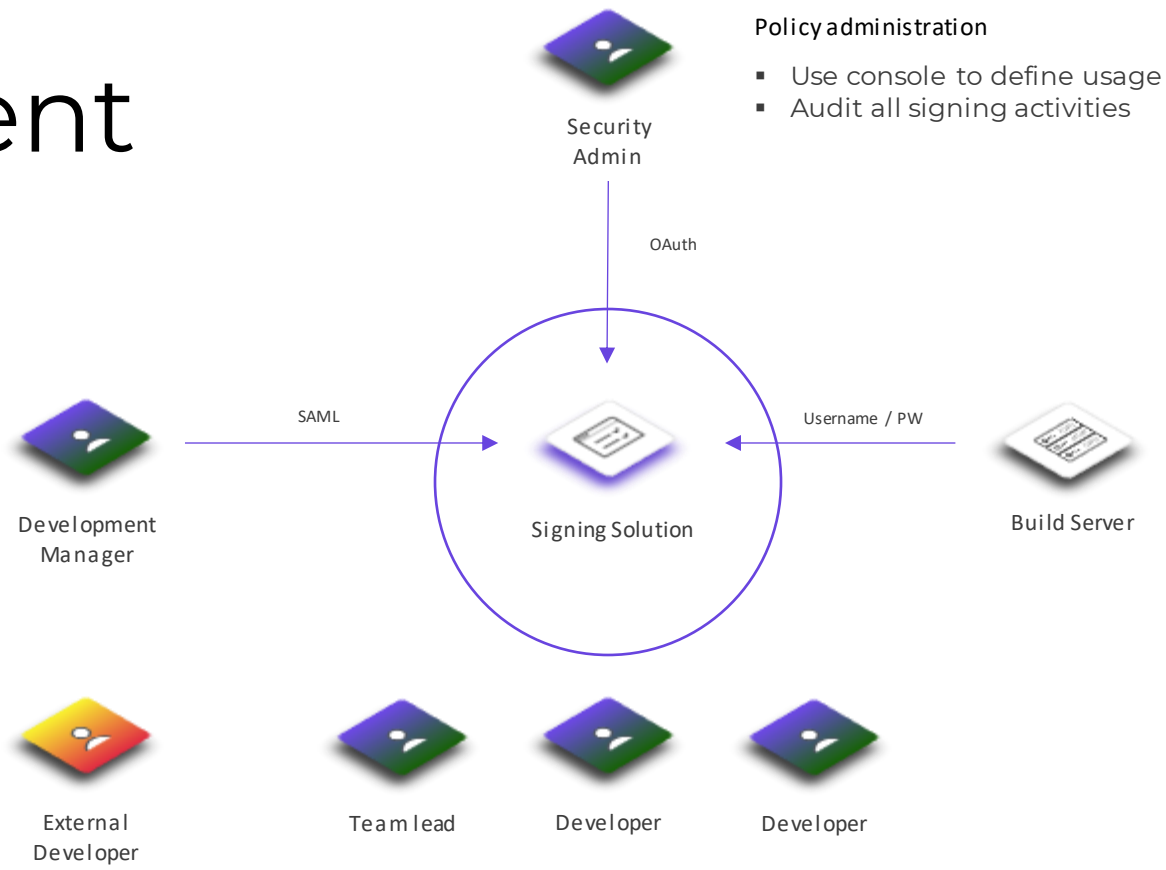


TO-BE CI/CD pipeline



Applying Policy as part of a CI/CD Pipeline

Policy enforcement



Policy administration

- Use console to define usage policies
- Audit all signing activities

Dev Mgt Policy

- Can sign using any application
- Can access all dev certificates

Build Server Policy

- Can sign only using SignTool
- Can access all dev certificates

External Dev Policy

- Can sign only using SignTool
- Can access only external certs
- Can only sign 8 am – 5 pm EST

App Team 1 Policy

- Can sign using SignTool or jarsigner
- Can access only App 1 certificate

Auditing Requirements

Digital Signing Use cases

Audit requirements and Key Attestation

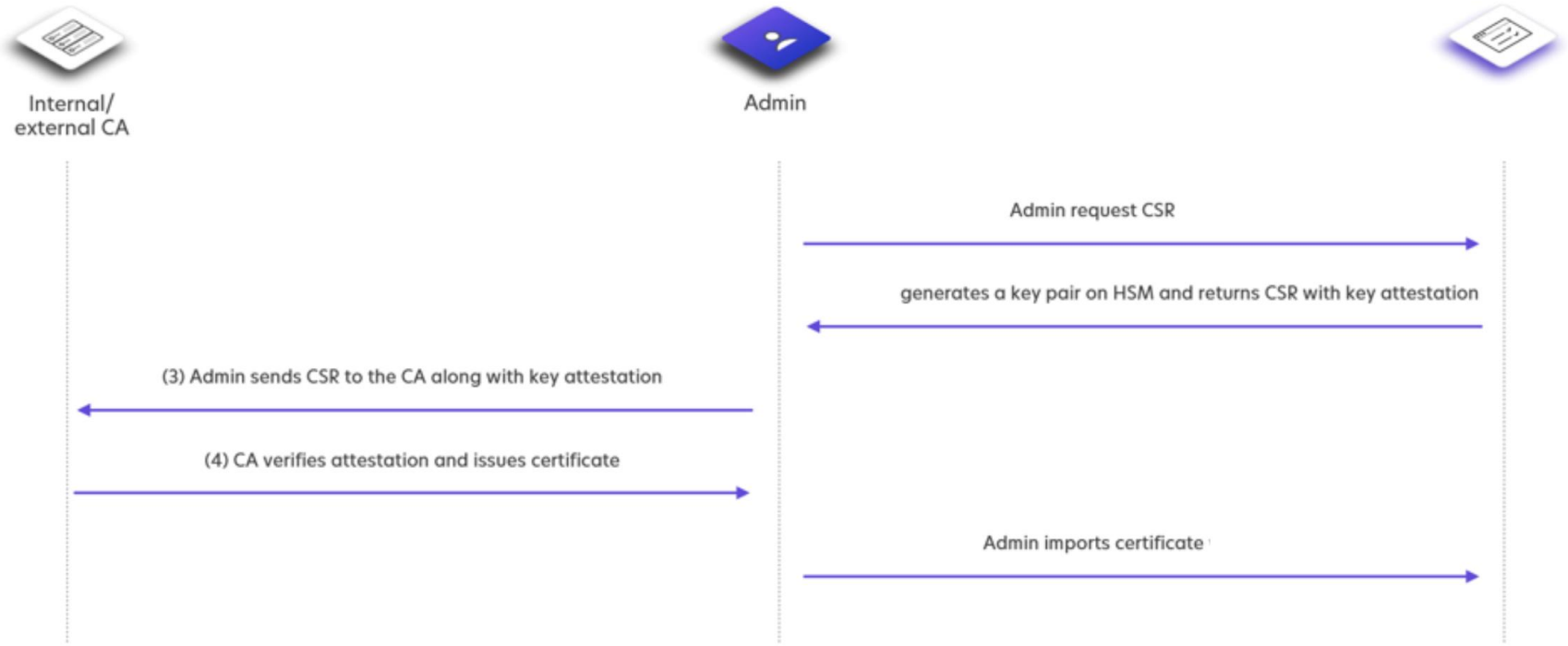
Event logs

Get full visibility into all signing activities with detailed logs.

Key attestation

Provide proof to your Public CA that keys are generated in an HSM.

Comply with CA/B Forum requirement that code signing keys are generated and stored in an HSM.



Keyfactor Signum

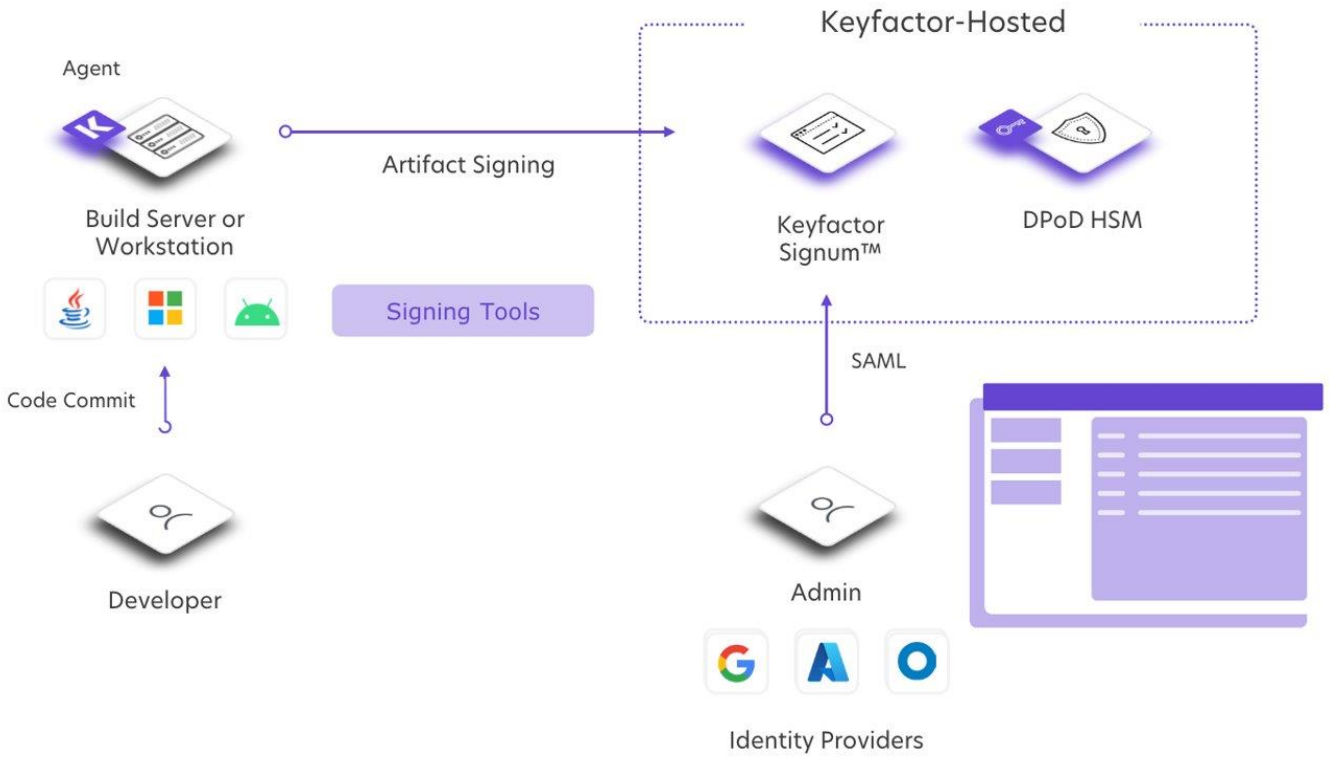
Shameless Plug

Solution architecture:
How it works

Native signing

Agent for local signing via native tools (e.g., SignTool, jarsigner, CoSign, Linux Penguin etc.).

- Native, transparent signing
- SAML/OAuth agent authentication



Built-in HSM

Private signing keys are stored securely in a FIPS 140-2 Level 3 HSM.

Admin Console

Keyfactor Signum™ console for audit trail, policy controls, roles and approvals.

Thank You

Questions?