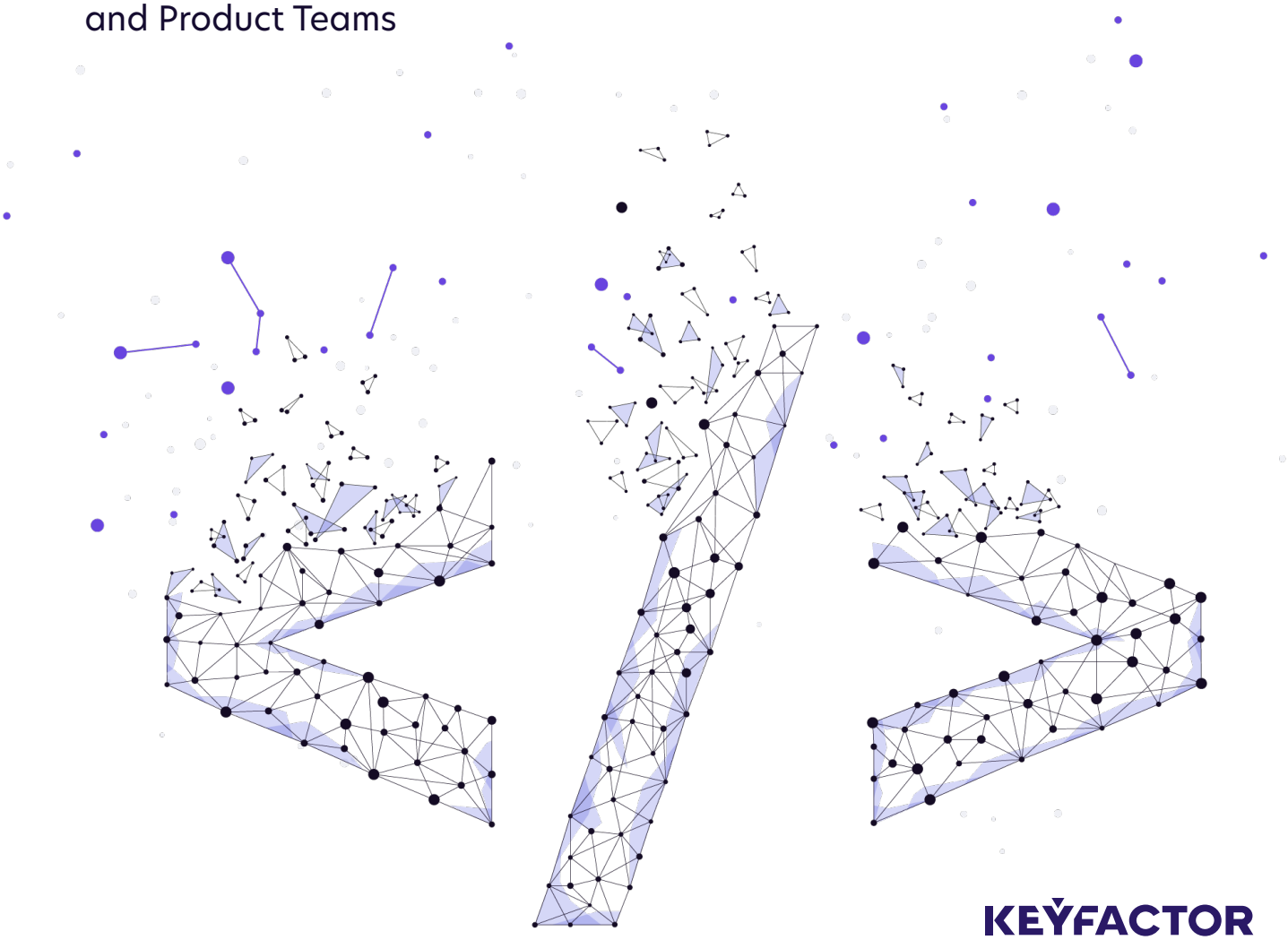
EBOOK

The Definitive Roadmap to Secure Code Signing

Practical Guidance for IT Security, DevOps, and Product Teams



KEYFACTOR

Table of Contents

Understand the Role of Code Signing 3

Identify Challenges 4

Know your Risks 5

Roadblocks to Success 7

The Roadmap to Secure Code Signing 7

- 01 Safeguard Private Keys 8
- 02 Secure Signing Operations 9
- 03 Integrate with DevOps 10
- 04 Monitor & Audit Compliance 11

Our Solutions 12

Understand the Role of Code Signing

Software and firmware developers must digitally sign code to establish trust and integrity in their products. With the rise of the Internet of Things (IoT) and explosive growth of software and mobile apps, the process of code signing is now more important than ever.

We live in a world that runs on code. Software permeates virtually every aspect of our lives, from the things we use each day to the critical infrastructure of our society. It's difficult to imagine a business today that doesn't depend on software in some way. As the IoT continues to grow, software only becomes more embedded in our physical world.

In a world where nothing is inherently trusted — from the software you deliver to end-users to the code and containers that run in your environment — the need to digitally sign and verify the authenticity of everything has reached a criticial point.

> **"When you sign a piece of code, you make a statement that the software came from your organization and that you stand behind it."**

Ted Shorter,
CTO & Co-Founder,
Keyfactor

Use Cases

Whether you build applications, deploy scripts and containers, or deliver over-the-air updates to your products, code signing is a critical step to protect your brand and ensure that code is authentic and untampered.

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.

Identify Challenges

Code signing is a powerful tool — if it's secure.

Signing code is a critical step in protecting your software supply chain, but it's more than just certificates and signatures. It's about ensuring only the right keys are used by the right developer to sign the right code and at the right time and place. Otherwise, it can open the door to malware, breaches, and supply chain attacks.

Need for speed

IT and application teams need quick, easy access to code signing keys to sign software, containers, and artifacts.

Lack of control

Security teams struggle to manage signing keys, who has access to them, and where they are stored.

Vulnerable keys

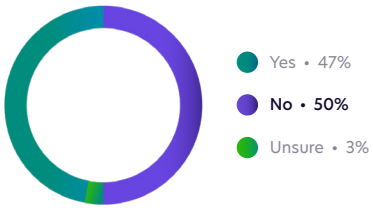
Keys are found on workstations, repo readme files, and build servers where they are susceptible to theft or misuse.

Here's the situation

The trust and integrity of code signing hinges entirely on the security of your keys. A single breach in this "chain of trust" can bring your entire business to a halt. Efforts to quickly revoke and re-issue certificates, notify customers, and push out a newly signed update are expensive.

50% of companies say they have no formal access controls for code signing keys.

Does your organization have a formal access control and approval process for code signing keys?



Keyfactor-Ponemon State of Machine Identity Management Report, 2022

Know Your Risks

Recent code signing attacks underscore the importance of managing reputational risk. Whether you consume software or sell it, all business leaders need to invest in the trust associated with their digital brand — and expect the same of their vendors.

Shadowhammer

Reported: 2019

Signing Breach

A well-known manufacturer of laptops and mobile phones unknowingly pushed malware to thousands of its customers for at least five months in a sophisticated software supply chain attack dubbed "Operation ShadowHammer." Hackers compromised two code signing certificates and pushed out signed malware through the company's Live Update Utility, inserting backdoors into at least 1 million devices.

Codecov

Reported: 2021

Supply Chain Attack

Attackers modified a script in Codecov, a popular code coverage tool, to export credentials to the attacker's server. One of Codecov's 29,000 customers confirmed that a subset of their CI pipeline used the affected Codecov component, and a private signing key used to sign their products was exposed.

BLISTER

Reported: 2021

Certificate Abuse

Cybersecurity researchers disclosed an widespread malware campaign that made use of valid code signing certificates issued by Sectigo to evade security defenses and deploy payloads to compromised systems.

Netfilter

Reported: 2021

Supply Chain Attack

Microsoft admitted to digitally signing a driver widely circulated in the gaming community, known as "Netfilter." The malicious actor submitted drivers for certification through the Windows Hardware Compatibility Program. The Netfilter driver communicated with Command and Control IP addresses in China.

Lazarus Group

Reported: 2022

Certificate Abuse

The North Korean state-sponsored group, known as Lazarus, best known for their role in the 2014 compromise of Sony Pictures, targetted Mac users with a signed executable disguised as a job description in a widespread social engineering campaign via LinkedIn. Code signing certificates have become a staple component for attacks by advanced persistent groups (APTs), including APT 27, APT 41, and Lazarus Group.

Lapsus$

Reported: 2022

Certificate Theft

The extortion group, known as Lapsus$, stole 1 terabyte of data from an American multinational technology company, which included two code signing certificates used to sign drivers and executables. Within days of the leak, security researchers found several hacking tools and malware components signed with the stolen certificates, including the well-known Colbalt Strike.

DiceyF

Reported: 2022

Malware Campaign

Researchers at Kaspersky observed a hacking group known as "DiceyF" targeting online gambling platform development studios and IT recruitment organizations. DiceyF steals code signing certificates to digitally sign malware and then distributes the signed malware via software distribution servers. Most targets were in Hong Kong and the Philippines, but there were also some in China and Vietnam.

Roadblocks to Success

The good news is that most independent software vendors (ISVs) and device manufacturers recognize the importance of code signing. The biggest challenge is how to implement it in a way that effectively meets the needs of both developers and IT security teams.

Speed vs Security

Security and PKI teams would prefer to isolate and lock down private keys, but developers need quick access to sign code and push it to production. The biggest problem is how to implement safeguards to prevent misuse of keys and certificates without impeding the productivity of your developers.

DevOps & Agile Development

DevOps practices have taken the IT world by storm. Fast and frequent incremental software builds are the name of the game. Any changes to the Software Development Lifecycle (SDLC) can introduce more risk than they aim to prevent. Security must adapt to existing DevOps workflows and signing processes.

Dispersed Dev Teams

Today's development teams collaborate across globally dispersed locations. When a remote team needs to sign code, the easy solution is to purchase a signing certificate. Once purchased, certificates are often left within reach of hackers and out of the purview of your security team on disparate developer workstations or build servers.

Sophisticated Threats

Software supply chain attacks are becoming increasingly frequent and sophisticated. Hackers, cybercriminals, and even state-sponsored attacks put the security and integrity of your software at risk. As the cost of code signing certificates on the underground market continues to rise, some attackers have taken a more direct approach.

Your Roadmap to Secure Code Signing Starts Here

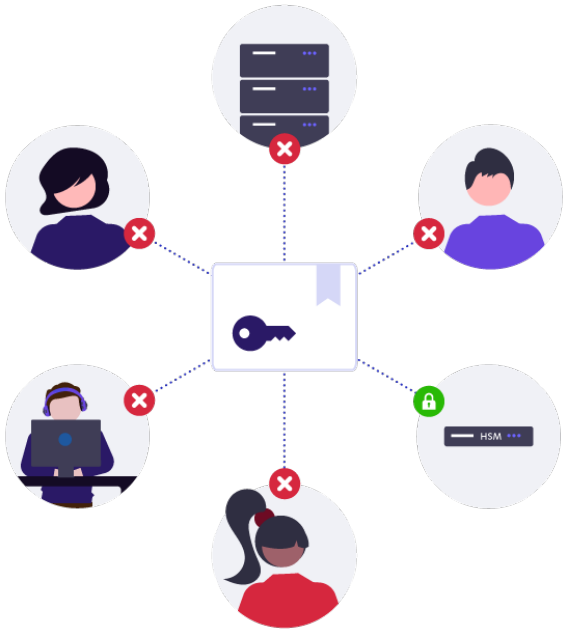
Enough theory. Let's dive into the four practical steps your organization can take to overcome these challenges and the right solution to help you get there.

01 Safeguard Private Keys

Private keys that developers use to sign code are invaluable to hackers. Once compromised, these keys can be used to sign virtually any code and distribute it to thousands of users. These types of attacks have become more and more frequent, as hackers seek to evade malware detection tools.

No industry or enterprise is immune — even trusted vendors in security and hardware have fallen prey to persistent attackers that successfully weaved through their network to find and compromise code signing keys, and used them against their own customers.

The burden to sign code often falls on developers that specialize in writing code, not securing keys. As a result, keys wind up in unsecured network locations such as developer workstations, build servers, and who knows where else. IT security teams are often left unaware of exactly how many code signing certificates they have or where they are stored.



Get a complete inventory of your landscape.

Digital certificates and keys used for code signing are high-value assets. Start first by taking an inventory of where your code signing keys live, who owns them, and how they are being used.

Store private keys in a certified HSM.

Hardware security modules (HSM) are the most effective way to ensure that your private keys remain under your control. Keys can be stored or generated inside the HSM and used to sign code anywhere without ever having to leave the hardware.

Centralize management of keys and certificates.

Disparate signing tools and workflows make it hard to keep track of signing keys. Ensure that private keys are stored and generated in a secure location, with centralized management and visibility over their usage.

Choose a flexible solution.

Choose a solution that is easy to deploy and scale as new use cases emerge. Determine whether a hardware, software, or cloud-based solution is best suited for your use case, and find a solution that can support your deployment requirements.

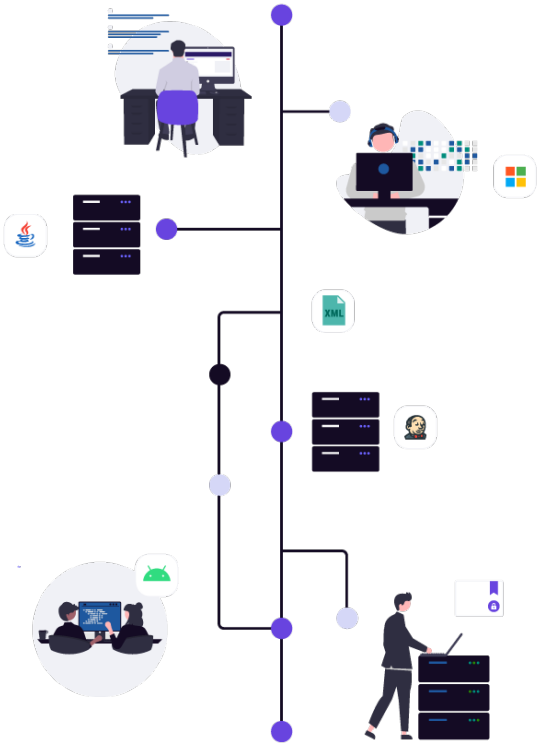
02 Secure Signing Operations

If your build or signing process is breached, a malicious actor doesn't need to steal your keys. By gaining access to a build server or developer workstation with access to code signing infrastructure, hackers can simply submit malware to be signed and distributed without detection.

Better known as "software supply chain attacks," these threats are even more difficult to detect, because they often involve either an insider or an attacker with direct access to code signing. Even the high-tech companies with sophisticated security teams were unable to detect breaches in their code signing infrastructure for months.

Attackers will always find the path of least resistance. Storing private keys in an HSM reduces the risk of key theft, but you can bet that determined threat actors will find another way.

It's absolutely critical to ensure that only the right developers can sign the right code — at the right time. By allowing only authorized users to sign and approve code, you can ensure that even if a hacker does breach your network, they can't infiltrate your code signing infrastructure.



Define roles and seperate duties.

Establish separate roles for those authorized to submit code for signing and those authorized to approve signing requests. Dividing these duties will help ensure that only trusted users can sign code, and avoid unauthorized signing.

Enforce code signing policies.

Ensure that even authorized developers are only granted access to sign code within specific parameters, such as which signing tools they can use, when they can sign code, or from which device. Look for solutions that can help you enforce these policies.

Segment test and release signing.

If code is signed during the development and testing stages, certificates should be distinct from those used in production signing. Ensure that development keys are not linked to the same root of trust as keys used to sign production code.

Authenticate before signing.

Ensure that users and devices are authenticated before they are allowed to sign code, artifacts, containers and other critical assets.

03 Integrate with DevOps

It's no knock against developers to say that security isn't their top priority. To move fast, application teams need to focus on delivering features and fixes, not handling sensitive keys. Security teams are under increasing pressure to protect sensitive secrets in DevOps environments, without disrupting build and release workflows.

Responsibility to secure code signing keys shouldn't fall on engineers or developers — they simply need to sign code efficiently. It's about finding the right balance between the security teams need for control and auditability, with the need for other teams to move efficiently.

Formalize your code signing process.

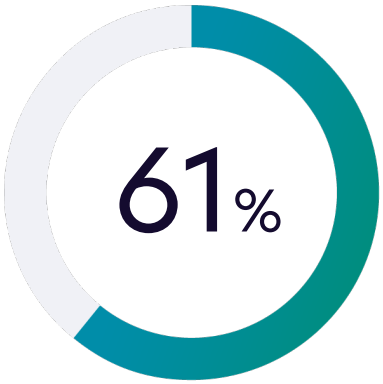
Document, track, and rigorously follow the steps required to sign code as part of your software development lifecycle (SDLC). Define what checks and verifications (i.e. QA, pen tests, virus scans, static analysis, etc.) must be performed before signing.

Minimize changes to the SDLC.

Take a collaborative approach to ensure that security and development teams achieve mutual goals to protect keys without disrupting the SDLC. It's important to find solutions that integrate with your processes, not the other way around.

Cover distributed dev teams.

Development teams today are often distributed across remote locations. Any code signing process must allow developers to sign code from anywhere, without requiring those developers to store keys locally on their machines.



Say theft or misuse of keys and digital certificates is a serious concern

Keyfactor-Ponemon State of Machine Identity Management Report, 2022

Integrate with signing tools and workflows.

Bottom line — if security is easy, developers will adopt it. Ensure that your code signing process supports a multitude of signing tools, workflows, and formats, and determine whether signing should be performed locally or server-side.

04 Monitor & Audit Compliance

Implementing security is one thing, keeping a clean audit log and staying compliant with best practices is another. Code signing security is a continuous process, not a set-and-forget deployment. Certificates inevitably expire, keys and algorithms weaken over time, and threats continue to evolve. Security teams must be able to identify these risks quickly and effectively respond.

Monitor code signing operations.

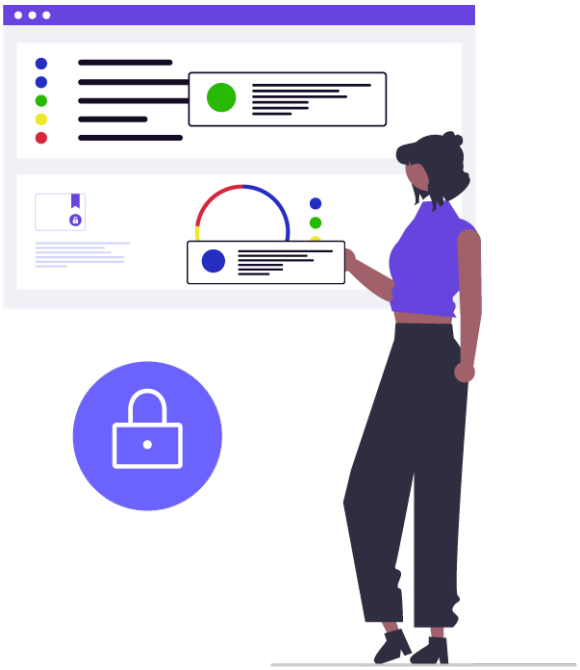
By monitoring code signing requests, authorizations, and signatures in real-time, security teams can more effectively detect anomalous activity and respond within minutes, rather than months.

Log and audit key usage.

Keep a comprehensive audit log of who used code signing keys, when, and who authorized the action. Review logs regularly for suspicious activity and ensure that these logs cannot be tampered with.

Include code signing certificates in CLM.

All digital certificates in your organization should be governed by certificate lifecycle management (CLM), including Extended Validation and Standard Code Signing Certificates - with provisions for how they are requested, issued, renewed or revoked.



Find the right fit for your use case

Solve all of your code signing challenges — whether you want to run in the cloud or on-premise, leverage platform-native signing tools or APIs, sign hundreds of times a day or millions — find the right fit for your use cases with Keyfactor.

Our Solutions

Protect the integrity and authenticity of code, software, and containers across your software supply chain. Keyfactor provides flexible and secure digital signing solutions, ensuring that signing is effortless for developers and easy to manage for security.

Keyfactor Signum

Policy-driven code signing as as a service

Let us handle it

Signum is delivered as a fully cloud-hosted code signing as a service solution with a built-in cloud HSM.

Sign with native tools

Signum integrates with platform-native signing tools, such as SignTool, Jarsigner, Cosign, and more.

Policy driven

Signum uses a built-in policy engine to define granular access and usage policies for code signing keys.

Learn More

Keyfactor SignServer

High performance, API-based signing engine

Deploy your way

SignServer is available as a turnkey software appliance, hardware appliance, or in the cloud.

Sign with APIs

SignServer is a centralized, server-side signing solution that enables signing via web interface or API.

High performance

SignServer is a signing and time-stamping engine intended for high-volume signing workloads.

Learn More

KEYFACTOR

Keyfactor is the machine and IoT identity platform for modern enterprises. The company helps security teams manage cryptography as critical infrastructure by simplifying PKI, automating certificate lifecycle management, and enabling crypto-agility at scale.

Contact Us

- www.keyfactor.com
- +1.216.785.2946