

Keyfactor Secure Code Signing Module

The demand for trust in today's uber-connected digital society is unprecedented. Consumers of software require guaranteed proof that the application they are using is legitimate. Secure code signing validates the author of the software and proves that the code has not been altered or tampered with after it was signed. Trusted code signing certificates are used to verify authenticity, but what is preserving the integrity of those certificates? Code signing certificates can be sold or used to create signed malware. Developers must take extreme care in protecting private keys mapped to code signing certificates to avoid complications. A streamlined, secure code signing process safeguards your business and provides inherent trust to your software consumers.

Why Securing Code Signing Certificates Matter

KEYFACTOR™ COMMAND FOR THE ENTERPRISE

Protection from breach. Signing certificates are valuable assets to hackers. Having an organization's signing certificates allows a hacker to create trusted versions of malware that appear to have been created by the authentic development organization.

Operating system requirements. Development teams are often required to sign software to support the installation. Operating systems like Windows will warn users if software or drivers are not signed.

Globally disperse development team operations. Today's development teams are widely comprised of members located in globally dispersed offices. When a remote team needs to sign developed software, the easy solution is to purchase a trusted signing certificate. Once implemented, the regional development team may leave the code signing certificate on a disparate PC or laptop, introducing risk. Placing the certificate in such location leaves it within easy reach of a hacker or to be forgotten. Both scenarios pose risk of breach or outage.

KEYFACTOR CONTROL[™] FOR INTERNET OF THINGS (IoT)

CAN-bus insecurities. Many large machines such as light-duty vehicles have underlying communication busses that are inherently insecure. Vehicles primarily utilize a bus technology called CAN-bus. This communication bus was developed at a time when there was no notion of securing such communication networks – as such the bus is

still unsecure today. Changing the bus in favor of a more secure communication network technology such as TCP/IP base communications is largely recognized by the automotive industry as long overdue. However, the cost to the industry to switch that technology will be in the trillions of dollars.

Code-signed firmware to secure ECUs. Code-signed firmware and software keep malicious code from being loaded onto vehicle electronic control units (ECU) unless the code is cryptographically trusted with a verifiable trust chain.

Given the fact that the underlying communication cannot be secured, the automotive industry has turned to utilizing cryptographically signed secure firmware and software (code-signed) to deploy on the vehicle ECUs. The now famous 2014 Jeep Hack changed the firmware on the Jeep's CAN-bus to take over and remotely control the vehicle in motion.

There are many examples of industrial systems that are similar to CAN-bus that both use an insecure communications bus system and also want to take advantage of the business opportunities that IoT technologies provide.

Examples of insecure, well-entrenched bus systems include:

- **CAN-bus.** Automotive, Trucking, Heavy Equipment, Rail, Busses
- BAC-Net. Building, HVAC, Refrigeration, Lighting
- Modbus. Industrial control and SCADA
- DNP. Utility systems and SCADA
- Profibus. Industrial Control system

KEÝFACTOR

Securely Supporting Globally Diverse DevSecOps

HOW IT WORKS

The Keyfactor Secure Code Signing Module locates and transfers all code signing certificates from Enterprise network locations (including all networked PC, storage, and thumb drives) to a secure vault. Once inside *the certificates never leave the vault*. A user with appropriate access presents the code to be signed to the module where it's signed and returned to the user. Access controls are in place to ensure that only those with the right privileges can sign software and firmware.

FIGURE 1. KEYFACTOR CODE SIGNING MODULE | WORKFLOW



STEP 1



Development manager grants a time-bound access to a trusted developer to sign a piece of developed software or firmware. Developer presents software to the Keyfactor Secure Code Signing Module.

KEY FEATURES

- Available on-premise or in the cloud
- Code signing certificate discovery across the entire
 Enterprise
- Once inside, certificates never leave secure vault
- Secure vault is available globally for utilization by a globally dispersed development team
- Process and workflow are provided to keep access to these valuable assets to only essential personnel
- Audit trail provides a full picture of all activities related to code signing



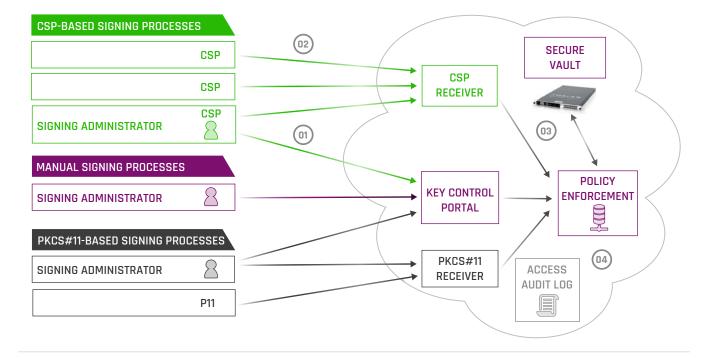
STEP 3

Developer performs code signing [either firmware for IoT, or Cryptographic Service Provider (CSP) from Windows].



STEP 4

Development manager has the ability to audit the entire process from start to finish.



ABOUT KEÝFACTOR

Keyfactor[™], formerly Certified Security Solutions (CSS), is a leading provider of secure digital identity management solutions that enables organizations to confirm authenticity, and ensure the right things are interacting in the right ways in our connected world.

CONTACT US

- keyfactor.com
- 216.785.2990

© 2018 Keyfactor | All Rights Reserved