ENTRUST

SECURING A WORLD IN MOTION

Code Signing Best Practices

The biggest issue with code signing is protecting the private signing key associated with the code signing certificate.

Exposing private keys can result in significant security risks, potentially leading to unauthorized access to sensitive data, financial losses, and damage to a company's reputation or brand.

Risks associated with exposed private keys include:

- Compromised data confidentiality and integrity
- Unauthorized system access
- Theft of intellectual property
- Distribution of malware
- Potential for regulatory compliance violations

Consider the following code signing best practices:

1. **Minimize access to private keys.**
    - Allow minimal connections to computers with keys.
    - Minimize the number of users who have key access.
    - Use physical security controls to reduce access to keys.

2. **Protect private keys with cryptographic hardware products.**
    - Use cryptographic hardware that does not allow export of the private key to software where it could be attacked.
    - Use minimum FIPS 140-2 Level 2 or Common Criteria EAL4+ hardware security module (HSM).
      - An HSM or service may be provided by the certification authority (CA), a cloud provider, or a subscriber key protection service.
      - If you purchase an HSM, evaluate the key management capabilities, including key generation, storage, and backup; and choose an HSM that separates duties, dual control, and key rotation.
      - The HSM should support auditing and monitoring capabilities, which allow tracking and logging of all signing operations and can detect any unusual activity.

3. **Timestamp code.**
    - All code signatures should be timestamped, allowing code to be verified after the certificate has expired or been revoked.

- If your code signing private key signs suspect code, your code signing certificate will be revoked before the time of the suspect code signature; timestamping allows all code signing before the revocation time to remain trusted.
- Your CA should provide a timestamp service.

4. **Understand the difference between test-signing and release-signing.**
    - Test-signing private keys and certificates requires fewer security access controls than production code signing private keys and certificates.
    - Test-signing certificates can be self-signed or come from an internal test CA.
    - Test certificates must chain to a completely different root certificate than the root certificate used to sign publicly released products; this precaution helps ensure that test certificates are trusted only within the intended test environment.
    - Establish a separate test code signing infrastructure to test-sign pre-release builds of software.

5. **Authenticate code to be signed.**
    - Any code submitted for signing should be strongly authenticated before it is signed and released.
    - Implement a code-signing submission and approval process to prevent signing of unapproved or suspect code.
    - Consider scanning code before signing or distribution to ensure suspect code is not distributed.
    - Log all code-signing activities for auditing and/or incident-response purposes.

6. **Virus-scan code before signing.**
    - Code signing does not confirm the safety or quality of the code; it confirms the publisher and whether the code has been changed.
    - Take care when incorporating code from other sources.
    - Implement virus-scanning to help improve the quality of the released code.

7. **Do not overuse any private key (distribute risk with multiple private keys and certificates).**
    - If suspect code is found, then publishers may want to display a User Account Control dialog box; if your key has signed suspect code, then code the certificate with the key compromise revocation reason.
    - If suspect code was signed before good code was signed, then revoking the certificate may impact the good code as well; changing keys and certificates often will help to reduce or avoid this conflict.

8. **Note the following regarding the revocation of compromised certificates:**
    - Report any key compromise or signed malware to your CA.
    - Compromised keys or signed malware of suspect code requires the code signing certificate to be revoked.

- If all signed code has been timestamped, you can select a revocation date that is before the date of the compromise, meaning code signed before the revocation date may not be impacted.