Boot Integrity Visibility

Boot Integrity Visibility

Boot integrity visibility acts as a hardware trust anchor by validating the ROMMON software to ensure its integrity.

Boot integrity visibility enables Cisco platforms to make both platform identity and software integrity information visible and actionable. Platform identity refers to the unique identity assigned during manufacturing, ensuring each device can be reliably identified. Software integrity involves capturing boot integrity measurements, which help determine whether the platform has started up using trusted code.

Catalyst 9000 Series Switches support boot integrity visibility feature.

How boot integrity visibility works

Summary

When a Cisco IOS XE image is copied onto a Cisco 9000 Series Smart Switches , the ROMMON Boot ROM verifies the image using Cisco release keys. These release keys are public keys that correspond to the private release key securely stored on the Cisco servers. The public release key is embedded within the ROMMON, enabling the switch to validate the authenticity and integrity of the Cisco IOS XE image before booting.

Workflow

The ROMMON follows these steps when it verifies a signed Cisco IOS XE image during the bootup:

1. Loads the Cisco IOS XE image into the CPU memory.
2. Examines the Cisco IOS XE package header.
3. Runs a non-secure integrity check on the image to ensure that there is no unintentional file corruption from the disk or TFTP. This is performed using a non-secure SHA-1 hash.

Note This step is a non-secure check of the image which is intended to confirm the image against inadvertent corruption due to disk errors, file transfer errors, or copying errors. This is not part of the image code signing. This check is not intended to detect deliberate image tampering.

4. Copies the Cisco's RSA 2048-bit public release key from the ROMMON storage and validates that the Cisco's RSA 2048-bit public release key is not tampered.
5. Extracts the Code Signing signature (SHA-512 hash) from the package header and verifies it using Cisco's RSA 2048-bit public release key.
6. Performs the Code Signing validation by calculating the SHA-512 hash of the Cisco IOS XE package and compares it with the Code Signing signature. The Signed package is now validated.
7. Examines the Cisco IOS XE package header to validate the platform type and CPU architecture for compatibility.
8. Extracts the Cisco IOS XE software from the Cisco IOS XE package and boots it.

Result

Note Image Code Signing validation occurs in step 4, 5, and 6. This is a secure code signing check of the image using an SHA-512 hash that is encrypted with a 2048-bit RSA key. This check is intended to detect deliberate image tampering.

If the software is not generated by a Cisco build system, the signature verification fails. The device ROMMON rejects the image and stops booting.

If the signature verification is successfully, the device boots the image to the Cisco IOS XE runtime environment.

Image signing

The Cisco build servers generate the Cisco IOS XE images. The Cisco IOS XE image is digitally signed during the build time. An SHA-512 hash is generated over the entire binary image file, and then the hash is encrypted with a Cisco RSA 2048-bit private key.

Verify software image and hardware

This section describes how to retrieve the checksum record that was created during a switch bootup. Enter the following commands in privileged EXEC mode.

Table 1: Command to retrieve the checksum records

Command	Description
show platform sudi certificate [sign [nonce ***nonce*****]]**	Displays checksum record for the specific SUDI. - (Optional) **sign**: Show signature - (Optional) **nonce**: Enter a nonce value
show platform integrity [sign [nonce ***nonce*****]]**	Displays checksum record for boot stages. - (Optional) **sign**: Show signature - (Optional) **nonce**: Enter a nonce value