

Boot Integrity Visibility

- Feature history for boot integrity visibility, on page 1
- Boot Integrity Visibility, on page 1
- How boot integrity visibility works, on page 2
- Image signing, on page 3
- Verify software image and hardware, on page 3
- Verify platform identity, on page 3
- Verify software integrity, on page 5
- Verify image signing, on page 6

Feature history for boot integrity visibility

This table provides release and platform support information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature name and description	Supported platform
Cisco IOS XE 26.2.1ea	Boot integrity visibility: Boot integrity visibility feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	Boot integrity visibility: This feature acts as a hardware trust anchor by validating the ROMMON software to ensure its integrity.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

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.

Your Cisco device supports the boot integrity visibility feature.

How boot integrity visibility works

Summary

When you copy a Cisco IOS XE software image onto your Cisco device, 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 Cisco build servers. The public release key is embedded within the ROMMON, enabling your device to validate the authenticity and integrity of the Cisco IOS XE software image before booting.

Workflow

The ROMMON follows these steps to verify a signed Cisco IOS XE software image when you boot the device:

1. Loads the Cisco IOS XE software image into the CPU memory.
2. Examines the Cisco IOS XE software package header.
3. Runs a nonsecure integrity check on the image to ensure there is no file corruption from disk or TFTP. The check uses a nonsecure SHA-1 hash.



Note This step checks for inadvertent corruption caused by disk, file transfer, or copying errors. It is not part of image code signing and does not 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 public release key.
6. Performs the Code Signing validation by calculating the SHA-512 hash of the Cisco IOS XE software package and compares it with the Code Signing signature. The signed package is now validated.
7. Examines the Cisco IOS XE software package header to validate the platform type and CPU architecture for compatibility.
8. Extracts the Cisco IOS XE software from the software package and boots the software.

Result



Note Image Code Signing validation occurs in steps 4, 5, and 6. This is a secure code signing check of the image using an SHA-512 hash, which 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 server, signature verification fails. Your device's ROMMON rejects the image and stops booting.

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

Image signing

Cisco build servers generate and digitally sign Cisco IOS XE software images at build time. An SHA-512 digest is calculated over the entire binary image, then signed using a Cisco-controlled RSA-2048 or ML-DSA private key. The device verifies the signature with the corresponding trusted public key, confirming the authenticity and integrity of the software image.

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

Verify platform identity

The following example displays the Secure Unique Device Identity (SUDI) chain in PEM format.

Encoded into the SUDI is the Product ID and Serial Number of each individual device such that the device can be uniquely identified on a network of thousands of devices.

- The first certificate is the Cisco Root CA 2048
- The second is the Cisco subordinate CA (ACT2 SUDI CA)

  Both certificates can be verified to match those published on https://www.cisco.com/security/pki/.
- The third is the SUDI certificate.

Note On executing the following command, you might see the message **% Please Try After Few Seconds** displayed on the CLI. This does not indicate a CLI failure, but indicates setting up of underlying infrastructure required to get the required output. We recommend waiting for a few minutes and then try the command again

```
Device# show platform sudi certificate sign

-----BEGIN CERTIFICATE-----
MIIDITCCAgmgAwIBAgIJAZozWHjOFsHBMA0GCSqGSIb3DQEBCwUAMC0xDjAMBgNV
BAoTBUNpc2NvMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwOTkwIBcNMTYwODA5
MjA1ODI4WhgPMjA5OTA4MDkyMDU4MjhaMC0xDjAMBgNVBAoTBUNpc2NvMRswGQYD
VQQDExJDaXNjbyBSb290IENBIDIwOTkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAw
ggEKAoIBAQDTtuM1fg0+9Gflik4axlCK1I2fb3ESCL8+tk8kOXlhfrJ/zlfRbe60
xRP0iUGMKWKBj0IvvWFf4AW/nyzCR8ujTt4a11Eb55SAKXbXYQ7L4YMg+lmZmg/I
v3GJEc3HCYU0BsY8g9LuLMvqwiNmAwM2jWzNq0EPArt/F6RiQKq6Ta3e7VIfDZ7J
65OA2xASA2FrSe9Vj97KpQReDcm6G7cqFH5f+CrdQ4qwAa4zWNyM3kOpUb637DNd
9m+n6WECyc/IUD+2e+yp21kBZIKH7JvDpu2U7NBPfr52mFX8AfCZgkXV69bp+iYf
saH1DvXIfPpNp93zGKUSXxEj4w881t2zAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIB
BjAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBQ4lVcPNCNO86EmILoUkcdBiB2j
WzANBgkqhkiG9w0BAQsFAAOCAQEAjeKZo+4xd05TFtq99nKnWA0J+DmydBOnPMwY
lDrKfBKe2wVu5AJMvRjgJIoY/CHVPaCOWH58UTqfji95eUaryQ/s36RKrBgMMlwr
WNItxE625PHuaN6EjD1WdWiRMZ2hy8F4FCKz5hgUEvN+PUNZwsPnpU6q3Ay0+11T
4TriwCV8kJx3cWu0NvTypYCCXMscSfLFQR13bo+1z6XNm30SecmrxkmQBVMqjCZM
VvAxhxW1iGnYdPRQuNqt0xITzCSERqg3QVVqYnFJUkNVN6j0dmmMVKZh17HgqLnF
PKkmBlNQ9hQcNM3CSzVvEAK0CCEo/NJ/xzZ6WX1/f8Df1eXbFg==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEZzCCA0+gAwIBAgIJCmR1UkzYYXxiMA0GCSqGSIb3DQEBCwUAMC0xDjAMBgNV
BAoTBUNpc2NvMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwOTkwIBcNMTYwODEx
MjAyODA4WhgPMjA5OTA4MDkyMDU4MjdaMDExHzAdBgNVBAMTFkhpZ2ggQXNzdXJh
bmNlIFNVREkgQ0ExDjAMBgNVBAoTBUNpc2NvMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEAvdzeSWdDI61RZDYRvA6JqaRvQyy6Dx1WaqI82UeKR4ZRn0ef
xMGvp4c88/VMS8WSjQO1qolMfMxqHkcSiFBOULx6Trquw4TrEf9sIuzvgJvDaEa8
IllXPwtPtNqZEIWi8jlinz2uGam93KuGPcioHfruzbDKWHL/HWFGYMgz+OKwhD3J
4NRySknQvUovfV8eWLeVOqW8rbnG3TZxv5VexOiK4jL3ObvsQPuAWUwUoo7nuFlE
GTG/VCeyCe/H8+afIScbZOkI9xejtckflnBYFVCyFxzm2H3YZatb6ohbyRXLtOPj
T3SJ+OOoYMlSLd28z727LpRbFFLGYhyWxEXDuQIDAQABo4IBgjCCAX4wDgYDVR0P
AQH/BAQDAgEGMBIGA1UdEwEB/wQIMAYBAf8CAQAwfwYIKwYBBQUHAQEEczBxMEEG
CCsGAQUFBzAChjVodHRwczovL3d3dy5jaXNjby5jb20vc2VjdXJpdHkvcGtpL2Nl
cnRzL2NyY2EyMDk5LmNlcjAsBggrBgEFBQcwAYYgaHR0cDovL3BraWN2cy5jaXNj
by5jb20vcGtpL29jc3AwHwYDVR0jBBgwFoAUOJVXDzQjTvOhJiC6FJHHQYgdo1sw
UgYDVR0gBEswSTBHBgorBgEEAQkVAR4AMDkwNwYIKwYBBQUHAgEWK2h0dHA6Ly93
d3cuY2lzY28uY29tL3NlY3VyaXR5L3BraS9wb2xpY2llcy8wQwYDVR0fBDwwOjA4
oDagNIYyaHR0cDovL3d3dy5jaXNjby5jb20vc2VjdXJpdHkvcGtpL2NybC9jcmNh
MjA5OS5jcmwwHQYDVR0OBBYEFOpro7nBE5d+G/s6jWhgBzlfh0j6MA0GCSqGSIb3
DQEBCwUAA4IBAQBcqYEOgAHhGWKndwM901XX2Enh4hjXR5avDg7G/f6Tb9H509dt
QW+AeZGEghhwUrw1EeG79tHkncAe+m+64xMC1ttyI1RSyn8rBqQYkXnnCRbtF/Nw
pQe5fjvdeIFWJhUI16TOt/ZlkNnWnLsUU1alZmN+J/FhSr8VTJWGRM9gY8hefH8f
5U7LMiDXxsFVHB7R6KGNjvtawrl6W6RKp2dceGxEIIvMahgMWWHHiWOQAOtVrHuE
NEjYR/7klLLwdgQF/NNCA2z47pSfMFnBcr8779GqVIbBTpOP2E6+1pBrE2jBNNoc
uBG1fgvh1qtJUdBbTziAKNoCo4sted6PW2/U
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIE8jCCA9qgAwIBAgIKB2EQhTQQhWJhIDANBgkqhkiG9w0BAQsFADAxMR8wHQYD
VQQDExZIaWdoIEFzc3VyYW5jZSBTVURJIENBMQ4wDAYDVQQKEwVDaXNjbzAgFw0y
NjA0MTYwODI5MzVaGA8yMDk5MDgwOTIwNTgyNlowbDEnMCUGA1UEBRMeUElEOkM5
NTUwLTk2TDREIFNOOkZETzI5NDEwTkZLMQ4wDAYDVQQKEwVDaXNjbzEYMBYGA1UE
CxMPQUNULTIgTGl0ZSBTVURJMRcwFQYDVQQDEw5RNUdHLVlSUEYtNE5OVTCCASIw
DQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMPgMraZ0Jd37GVAcAwU6BgRU53E
l88DM1SKv31JkQ5rt5RpnpdzDIBFfeAp3He9Q42ITWaXHoVCFQNhZnBkI4NOeuRo
UPWiRmgZdZ0cLvrpQJ8HkrDvXTBavLLLSNP72clvKjDJvf/O/kDuoo1C4cpwKh0E
```

```
a3PxFdGFveHstE8uda1G29KpXpSLTUNHIiDNZWdy8stFGNqUopjKA/J1MULyOlTI
iRQlwrlegyqAc1FKWux28DnOEBiMkbdcQTVeb/oO6FI1X+c1KZTOBNJoUl+goGDk
VuijxKqSEJVSbX4/IOebPHYRaW4YoxmebBGjGbk9D9Dl99VQVBgGXmMyrY0CAwEA
AaOCAc0wggHJMA4GA1UdDwEB/wQEAwIF4DAMBgNVHRMBAf8EAjAAMB8GA1UdIwQY
MBaAFOpro7nBE5d+G/s6jWhgBzlfh0j6MIIBZwYDVR0RBIIBXjCCAVqgUAYKKwYB
BAEJFQMEAqBCE0A5NzNGQTQ5ODE5RTE0NTY1NjYwNzFBQzVFMTU0RjI1ODI2RTM2
RjFGOTkzM0M3MEM5NjAwNDVGRjEyOTRGQUU4oHAGCisGAQQBCRUDBASgYhNgMUI2
Mjk3NEY0MDExNzA0MDgyMzQ3NEZBNkEyRjg3REZGMTNGQjM1MjRDQ0EwNDkxQzQy
NDY4NjkwNjVEQzIyQjA5M0E0MkUzMzY5MkQwRTRFMEE5NUNGMjNCQzE1QUFEoFAG
CisGAQQBCRUDBAOgQhNAMDAwMDEwMDAwMDA0MTQyMDgyMUREMTlEMDAwMDAwMDAw
MDAwMDAwMDAwMDAwMDAwMDAwMDA0QjUyNTk1MKBCBgkrBgEEAQkVAgOgNRMz
Q2hpcElEPUx2VzZpVlJaam9zRGs3L0xSOVBmdmdBQUFBQUFBQUFBQUFBQUFB
QUE9MB0GA1UdDgQWBBTr19uzsiktLCPintxH66NV1TbxVjANBgkqhkiG9w0BAQsF
AAOCAQEArX2KKkI4E7Z7unNFFkiUNcyDegqNbZTqXgmcEpPkz3ZdFphhRfN4vnmN
KKGkU+WY6/y53FrjtueXuVIdU5MChKeTkuJsMOr+PbXeS3cnhapyvEzzaL4b/oi5
DkSXw5MyzcLh5T7ZVQ5sHO54F5MXIZW0/EMREf9ir5uTZs/iTGoe6pYW46ZmlHtq
H2t2SbOXHYpX1+aZOKuVU37dduLoMUKJNFVAeEDU3HgEUZxQuc94M6vVhUj7K+0X
jPbCAuG0v6DUOQVv6nMPEHdofPMRcMyoexzLk7jAL/8GwZlRJftmcyH4x329c0Yl
Z0ahgllmAN9Q8V8UY25KOeUzDkAyWA==
-----END CERTIFICATE-----
Signature version: 1
Signature:
```

The optional RSA 2048 signature is across the three certificates, the signature version and the user-provided nonce.

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

Cisco management solutions are equipped with the ability to interpret the above output. However, a simple script using OpenSSL commands can also be used to display the identity of the platform and to verify the signature, thereby ensuring its Cisco unique device identity.

```
linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:C9300-24P SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite SUDI/CN=C9300-24P
```

Verify software integrity

The following example displays the checksum record for the boot stages. The hash measurements are displayed for each of the three stages of software successively booted. These hashes can be compared against Cisco-provided reference values. An option to sign the output gives a verifier the ability to ensure the output is genuine and is not altered. A nonce can be provided to protect against replay attacks.

Note Boot integrity hashes are not MD5 hashes. If you run **verify /md5 cisco9k_iosxe.17.18.01.SPA.bin** command for the bundle file, the hash will not match.

The following is a sample output of the **show platform integrity sign** command in install mode.

This output includes measurements of each installed package file.

```
Device# show platform integrity sign

Platform: C9550-96L4D
Boot 0 Version: MP2002R07.1120260302.3ea7c865
Boot 0 Hash: 8ADA8820CF37AAF1ECFEAB5BD75415D87589D72AE4A544880E0531A000EA980E
Boot Loader Version: 26.1.1.A%26.1.1.A
```

```
Boot Loader Hash:
608EEFD92E95C13C48C9A4742F4E31D636D08E3E4F1D8A3B45A8FD57EBA28FD8B33508522D88370761A9A5063C6A3962E3239323EAC18C8AF162B55165FFA057
OS Version: BLD_V262_THROTTLE_LATEST_20260709_003125
OS Hashes:
cisco9k_iosxe.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.bin:
9EC3AEB72E8A8DC5609368E4CFFACD35F55F02A42045AAC36C3E2A590CBC0F6E9B6904355C5A90E398E72DC2E3F9E8F665515C0BEBC04E8D54FA459993F64949
cisco9k-rpbase.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
0B0072BA547C21C15C33E3D469EB32CDA9F9A86667E0F62E9BE0341DEBB1B432F48A4F62C30C82DDA9658C966E0A8D61AD155D67E69A60B4A6142D35C08DE741
cisco9k-wlc.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
AA01F3AC8A6C7E4DEFC1A29C733D609BC9AC51F8E1B8B60880CB717A9EB712243E83B56C9E821F403AB4C9F731FB332DFF8129C587E16876FE5134F2081780D8
cisco9k-srdriver.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
893283C5B038B79E17C42D99092C6A73F4C231B1463FF8843EA935A5ED2FA9056F9A101A8B0C68D640122833D4D011AB22B566158780111755446048569D9D62
cisco9k-webui.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
46EE0653035B37F98FD1FFE639D56C165C70AC376802FA1AA72D6645362F0E7167A54A59AAA560294D5E52EDAF81BB7CFCCA40348E6910A0EBD23E71FDB8E023
cisco9k-guestshell.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
E592AE468794F0C22FE28337462FE22C423733A13F672216C8C682B542706B5F9318FAF20066771E7BFEA89C9683A84D4BCB622454C7E51CCABB430BC63F716A
cisco9k-lni.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
D3247E6820E5DF343FDCD673850706FA27CBE825393C31CFDFACEB0F3CD523764BF09F306B99180208DAFFC4E2F596E23EF894D63F4D18B0646B0275E1209620
cisco9k-cc_srdriver.BLD_V262_THROTTLE_LATEST_20260709_003125.SSA.pkg:
9851AE6F435841B7DF5586FE8DA34929FF4C3122150FE7992EA1030AC3D2765ED5F51F87419BCD86090907557C3030D89C4FA271EC0237501CA9A8940C1D134B
PCR0: 7244B76E2DF81C6585BC9FE5303A3B0734C0CB9752D097FC27546C935FBEFA5D
PCR8: 36E69ED3949CA95A9B77D41A02B97ECAB9AEFCD87963160EFDEAD4C8B738FA67
Signature version: 1
Signature:
[redacted]
```

Verify image signing

The following example displays the secure code signing check of the image during bootup using an SHA-512 hash

```
switch:boot
system has been configured to boot from device configuration.
Preparing to autoboot. [Press Ctrl-C to interrupt]  0
boot: attempting to boot from
[tftp://10.106.16.20/auto/tftp-blr-users1/kbalucha/cisco9k_iosxe.26.02.01.SPA.bin]
tftp:
 blocksize : 8192
 server    : 10.106.16.20
 path      : auto/tftp-blr-users1/kbalucha/cisco9k_iosxe.26.02.01.SPA.bin
 (ascii)   : auto/tftp-blr-users1/kbalucha/cisco9k_iosxe.26.02.01.SPA.bin
 url      : tftp://10.106.16.20/auto/tftp-blr-users1/kbalucha/cisco9k_iosxe.26.02.01.SPA.bin
tftp: preparing TFTPv4 GUID vector
network: enabling IPv4 static configuration
network: enabling IPv4 static configuration

h/w (environment):
 interface : eth0
 mac       : 14:BC:68:D3:7C:00
n/w (environment):
 ip        : 10.106.21.16
 mask      : 255.255.255.0
 gateway   : 10.106.21.1

h/w:
 interface : eth0 (Ethernet)
 mac       : 14:BC:68:D3:7C:00
n/w (ip v4):
 ip        : 10.106.21.16
 mask      : 255.255.255.0
 route(s)  : 0.0.0.0 -> 10.106.21.0/255.255.255.0
           : 10.106.21.1 -> 0.0.0.0/0.0.0.0
```

```
n/w (ip v6):
 ip(s)     : FE80::16BC:68FF:FED3:7C00/64
 route(s)  : :: -> FE80::/64

tftp v4:
 server    : 10.106.16.20
 file      : auto/tftp-blr-users1/kbalucha/cisco9k_iosxe.26.02.01.SPA.bin
 blocksize : 8192
tftp: preparing TFTPv4 function vector
tftp v4:
 station   : 10.106.21.16
 netmask   : 255.255.255.0
 gateway   : 10.106.21.1
 server    : 10.106.16.20
tftp: configured TFTPv4 protocol
tftp: value(s): 0
tftp: value(s): 0 8192
Option#1  : tsize
Value     : 0
Option#2  : blksize
Value     : 8192
tftp: requesting 2 option(s)
tftp: received 2 option(s)
Option#1  : tsize
Value     : 1258756397
Option#2  : blksize
Value     : 8192
tftp: TFTPv4 option 'blksize' value '8192'
!
!
!
<<output truncated>>
```