



Boot Integrity Visibility

- [Information About Boot Integrity Visibility, on page 1](#)
- [Verifying the Software Image and Hardware, on page 2](#)
- [Verifying Platform Identity and Software Integrity, on page 3](#)
- [Verifying Image Signing, on page 5](#)
- [Additional References for Boot Integrity Visibility, on page 7](#)
- [Feature History for Boot Integrity Visibility, on page 7](#)

Information About Boot Integrity Visibility

Boot Integrity Visibility allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity. Software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code.

During the boot process, the software creates a checksum record of each stage of the bootloader activities.

You can retrieve this record and compare it with a Cisco-certified record to verify if your software image is genuine. If the checksum values do not match, you may be running a software image that is either not certified by Cisco or has been altered by an unauthorized party.

Image Signing and Bootup

The Cisco build servers generate the Cisco IOS XE images. Cisco IOS XE images use the Abraxas image signing system to sign these images securely with the Cisco private RSA keys.

When you copy the Cisco IOS XE image onto a Catalyst 9000 Series Switch, Cisco's ROMMON Boot ROM verifies the image using Cisco release keys. These keys are public keys that correspond to the Cisco release private key that is stored securely on the Abraxas servers. The release key is stored in the ROMMON.

Catalyst 9000 Series Switches support boot integrity visibility feature. Boot integrity visibility serves as a hardware trust anchor which validates the ROMMON software to ensure that the ROMMON software is not tampered with.

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. The ROMMON verifies the signature using the Cisco public key. 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.

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.
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.



Note In above process, step 3 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.

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.

Verifying the Software Image and Hardware

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



Note On executing the following commands, 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.

The messages **% Error retrieving SUDI certificate** and **% Error retrieving integrity data** signify a real CLI failure.

Procedure

	Command or Action	Purpose
Step 1	<code>show platform sudi certificate [sign [nonce nonce]]</code>	Displays checksum record for the specific SUDI.

	Command or Action	Purpose
	<p>Example:</p> <pre>Device# show platform sudi certificate sign nonce 123</pre>	<ul style="list-style-type: none"> • (Optional) sign - Show signature • (Optional) nonce - Enter a nonce value
Step 2	<p>show platform integrity [sign [nonce]]</p> <p>Example:</p> <pre>Device# show platform integrity sign nonce 123</pre>	<p>Displays checksum record for boot stages.</p> <ul style="list-style-type: none"> • (Optional) sign - Show signature • (Optional) nonce - Enter a nonce value

Verifying Platform Identity and Software Integrity

Verifying 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 and 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.



Important All the CLI outputs provided here are intended only for reference. The output differs based the configuration of the device.

```
Device# show platform sudi certificate sign nonce 123
-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIQX/h7KcT03I1CoxW1aMmt/zANBgkqhkiG9w0BAQUFADA1
MRYwFAYDVQQKEw1DaXNjbyBTeXN0ZW1zMRSwGQYDVQQDExJDaXNjbyBSb290IENB
IDIwNDgwHhcNMDQwNTE0MjAxNzEyWhcNMjkwNTE0MjAyNTQyWjAlMRYwFAYDVQK
Ew1DaXNjbyBTeXN0ZW1zMRSwGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgwggEg
MA0GCSqGSIb3DQEBAQUAA4IBDQAwggEIAoIBAQCwmrmrp68Kd6ficia0ZmKUEIhH
xmJVhEAyv8CrLqUccda8bnuoqrpu0hWISEWdovyd0M5j0AmaHBKeN8hF570YQXJ
FcjPfto1YYmUQ6iEqDGYeJu5Tm8sUxJsZr2tKyS7McQr/4NEb7Y9JHcJ6r8qqB9q
VvYgDxFU14FlpyXOWWqCze+36ufijXWLBvLdT6ZeYpzPEApk0E5tzivMW/VgpSdH
jWn0f84bcN5wGyDwbs2mAag8EtKpP6BrXruOIIt6ke0la06g58QBdKhtCytKmg9l
Eg6CTy5j/e/rmxxrbU6YTYK/CdfHbBcl1HP7R2RQgYCUTOG/rksc35LTLgXfAgED
o1EwTzALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUUJ/PI
FR5umgIJFq0roIlgX9p7L6owEAYJKwYBBAGCNxUBBAMCAQAwDQYJKoZIhvcNAQEF
BQADggEBAJ2dhISjQa18dwy3U8pORFbi71R803UXHOjgxxkLtv5MOhmBvrbW7hmW
Yqpao2TB9k5UM8Z3/sUcuVdJcr18JOagxEu5sv4dEX+5wW4q+ffY0vhN4TauYuX
cB7w4ovXsNgOnbFp1iqRe6lJT37mjpXYgyC8lWhJdTsD9i7rp77rMKSSh0T8lasz
Bvt9YArEtIpjsJyp8qS5UwGH0GikJ3+r/+n6yUA4iGe00caEb1fJU9u6ju7AQ7L4
CYNu/2bPPu8Xs1gYJQk0XuPL1hS27PKSb3TkL4Eq1ZKR4OCXPDJoBYVL0fdX41Id
kxpUnwVwwEpxYB5DC2Ae/qPOgRnhCzU=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEPDCCAySgAwIBAgIKYQlufQAAAAAADANBgkqhkiG9w0BAQUFADA1MRYwFAYD
VQQKEw1DaXNjbyBTeXN0ZW1zMRSwGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgw
```

```

HhcNMTEwNjMwMtc1NjU3WhcNMjkwNTE0MjAyNTQyWjAnMQ4wDAYDVQKKEwVdAXNj
bzEVMBMGAlUEAxMMQUNUMiBTvURJIENBmIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBcGKAQEA0m5l3THIx9tN/hS5qR/6UZRpdd+9aE2JbFknjht6gfHKd477Aks
5XAtUs5oxDYVt/zEbslZq3+LR6grqKQVv6JYvH05UYLBqCj38s76NLk53905Wzp
9pRcmRCPuX+a6tHF/qRuOiJ44mdeDYZo3qPCpxzprWJDPclM4iYKHumMQMqmgmg+
xghHiooWS80BOcdiynEbeP5rZ7qRuewKmpl1TiI3WdBNjZjnpfjg66F+P4SaDkGb
BXDgJl3oVeF+EyFWLrFjJ97fL2+8oauV43Qrvnf3d/GfqXj7ew+z/sXlXtEOjSXJ
URsYMEj53Rdd9tJwHky8neapszS+r+kdVQIDAQABO4IBWjCCAVYwCwYDVR0PBAQD
AgHGMB0GAlUdDgQWBBRI2PHxwnDVW7t8cwmTr7i4MAP4fzAfBgNVHSMEGDAWgBQn
88gVhm6aAgkWrSugiWbf2nsvqjBDBGNVHR8EPDA6MDiGnQA0hjJodHRWOi8vd3d3
LmNpc2NvLmNvbS9zZWN1cm10eS9wa2kvY3JsL2NyY2EyMDQ4LmNybDBQBggrBgEF
BQcBAQREMEIwQAYIKwYBBQUHMAKGNh0dHA6Ly93d3cuY2l2Y28uY29tL3NlY3Vy
aXR5L3BraS9wb2xpY2l1cy9pbmRleC5odG1sMBIGAlUdEwEB/wQIMAYBAf8CAQAwDQYJ
KozIhvcNAQEFBQADggEBAGh1qc1r9tx4hzWgDERm371yeuEmqcIfi9b9+GbMSJbi
ZHc/CcCl0lJu0a9zTXA9w47H9/t6leduGxb4WeLxcwCiUgVftCa51Ik1t8nNbcKY
/4dwlex+7amATUQO4QggIE67wVlPu6bgAE3Ja/nRS3xKYsnj8H5TehimBSv6TECI
i5jUhoWryAK4dVo8hCkjEkzu3ufBTJapnv89g9OE+H3VKM4L+/KdkUO+52djFKn
hyl47d7cZR4DY4LIuFM2P1As8YyJzoNpK/urSRI14WdIlpl1r1nH7KND15618yfvP
0IFJZBGrooCRBjOSwFv8cpWCbmWdPaCQT2nwIjTfY8c=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIDfTCCAmWgAwIBAgIEAwQD7zANBgkqhkiG9w0BAQsFADANMQ4wDAYDVQKKEwVD
aXNjbzEVMBMGAlUEAxMMQUNUMiBTvURJIENBMB4XDTE4MDkyMzIyMzIwNl0xDTI5
MDUxNDIwMjU0MvowATEnMCUGAlUEBRMeUElEOkM5NjAwLmNvVUC0xIFNOokNBVDIy
MzZMMFE5MQ4wDAYDVQKKEwVdAXNjZzEYMBYGA1UECzMPQUNULTIgtG10ZSBTVURJ
MRQwEgYDVQDEwtDOTYwMCI1TVVATMTCCASiWdQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBANsh0jcvgh1pdOjP9KnffDnDc/zEHDzbCTWpJi2FZcsaSE5jvq6CUqc4
MYpNAZU2Jym7NSD8iQbMXwnCtoL64QtXqEfhRYmc4d5o933M7GwpEH0I7HUSbo/
Fxy7JbMGPgAkY7rKsYENiNK2hiR7Q207X2BidOKknEuofWdJMNyMaZgLYLOHbJ
5oXaORxhUy3VRaxN16qI7kyxuugg2LcAbZ539sRXe8JtHyK811URNsGmiQ0S17pS
idGmrJJOpeHA0EUVTZqEYn3z+Nw9uxLVszu6+hEJYlqfI+Yef0DbVZl1cy5r/jf
yNdGuGKvd5agvgCly8aYmZa3P+D5S8sCAwEAAANvMG0wDgYDVR0PAQH/BAQDAgXg
MAwGAlUdEwEB/wQMAAwTQYDVR0RBEYwRKBBCBgkrBgEEAQkVAgOgNRMzQ2hpcElE
PVUxUk5TVEl3TVRjd05qTFBQUFwZndBQUFBQUFBQUFBQUFBQUFBQUhUhtSlU9MAOG
CSqGS1b3DQEBCwUAA4IBAQCrpHo/CUyk5Hs/asIcYw0ep8KocSkn8qamyd4owD
e/MGJW9Bs5f09IEbILWPdytCCS21SyJbxz2HvVDzdxQdxjDwUNiWuu3dWMMXN/i67
yuCGM+1A1AAG5dT61NgWYHh+YzsZm9eoq1+4NM+JumXWsnzAK8rSy+dSpBxqFsbQ
E00lPsaK7y2h8gs+XrV9x+D48OZQkTRXpxhJfiWvs+EbdgsAM/vBxTaoTJpVmXWN
Cmcj9X52Xl3i4MdOUXocZLO2kh6JSgOYGkFeZifJ0idVmFAf0cJ6+cEF6bSxAqBL
veel+8LmeiE/209h6qGHPPDacCaXA2oJCDHveAt8iPTG
-----END CERTIFICATE-----

```

```

Signature version: 1
Signature:

```

```

-----BEGIN-----
-----END-----

```

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 sudi_id.pem -subject -noout
subject= /serialNumber=PID:C9600-SUP-1 SN:CAT2239L06B/CN=C9600-SUP-1-70b3171eaa00

```



```

boot: reading file packages.conf
#
Performing Integrity Check ...
boot: parsed image from conf file: cat9k-rpboot.17.02.01.SSA.pkg

```

```

Loading image in Verbose mode: 1

```

```

Image Base is: 0x100099000
Image Size is: 0x2C83487
Package header rev 3 structure detected
Package type:30001, flags:0x0
IsoSize = 0
Parsing package TLV info:
000: 0000000900000001D4B45595F544C565F - KEY_TLV_
010: 5041434B4147455F434F4D5041544942 - PACKAGE_COMPATIB
020: 494C49545900000000000090000000B - ILITY
030: 4652555F52505F54595045000000009 - FRU_RP_TYPE
040: 000000184B45595F544C565F5041434B - KEY_TLV_PACK
050: 4147455F424F4F54415243480000009 - AGE_BOOTARCH
060: 0000000E415243485F693638365F5459 - ARCH_i686_TY
070: 5045000000000009000000144B45595F - PE KEY_
080: 544C565F424F4152445F434F4D504154 - TLV_BOARD_COMPAT
090: 0000000900000010424F4152445F6361 - BOARD_ca
0A0: 74396B5F545950450000000900000018 - t9k_TYPE
0B0: 4B45595F544C565F43525950544F5F4B - KEY_TLV_CRYPTO_K
0C0: 4559535452494E470000000900000004 - EYSTRING

TLV: T=9, L=29, V=KEY_TLV_PACKAGE_COMPATIBILITY
TLV: T=9, L=11, V=FRU_RP_TYPE
TLV: T=9, L=24, V=KEY_TLV_PACKAGE_BOOTARCH
TLV: T=9, L=14, V=ARCH_i686_TYPE
TLV: T=9, L=20, V=KEY_TLV_BOARD_COMPAT
TLV: T=9, L=16, V=BOARD_cat9k_TYPE
TLV: T=9, L=24, V=KEY_TLV_CRYPTO_KEYSTRING
TLV: T=9, L=4, V=none
TLV: T=9, L=11, V=CW_BEGIN=$$
TLV: T=9, L=17, V=CW_FAMILY=$cat9k$
TLV: T=9, L=74, V=CW_IMAGE=$cat9k-rpboot.17.02.01.SSA.pkg$
TLV: T=9, L=20, V=CW_VERSION=$17.2.01$
IOS version is 17.2.1
TLV: T=9, L=53, V=CW_FULL_VERSION=$17.2.01.0.869.1580816579..Amsterdam$
TLV: T=9, L=52, V=CW_DESCRIPTION=$Cisco IOS Software, IOS-XE Software$
TLV: T=9, L=9, V=CW_END=$$
Found DIGISIGN TLV type 12 length = 392
RSA Self Test Passed

Expected hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F

Obtained hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F
Sha512 Self Test Passed
Found package arch type ARCH_i686_TYPE

```

```
Found package FRU type FRU_RP_TYPE
Performing Integrity Check ...
```

```
RSA Signed DEVELOPMENT Image Signature Verification Successful.
```

Additional References for Boot Integrity Visibility

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	<i>Command Reference (Catalyst 9600 Series Switches)</i>

Feature History for Boot Integrity Visibility

This table provides release and related information for features explained in this module.

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

Release	Feature	Feature Information
Cisco IOS XE Gibraltar 16.11.1	Boot Integrity Visibility	Boot Integrity Visibility allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to <http://www.cisco.com/go/cfn>.

