Cisco IOS XE and GuestShell CA Trust eMudhra Distrust Guide

Contents

Purpose

This document provides guidance for Cisco IOS® XE for the emergency distrust of eMudhra Root and Intermediate Certificate Authorities.

Use this guide to audit and manage CA trust for services whose configuration permits the affected hierarchies. Product applicability and required remediation are defined in the Cisco informational. Check the command support and service impact for your Cisco IOS XE release before changing trust.

Cisco informational: cisco-sa-emudhra-info-ScS22A1r

Affected eMudhra CA Certificates: Original and Cross-Signed

Inventory verified on 29 October 2026. Eight eMudhra issuing CA keys are affected: six for TLS and two for code signing. Each key appears in an original emSign-issued CA certificate and a Sectigo cross-signed CA certificate. The inventory therefore contains 16 certificates: 12 TLS and 4 code-signing certificates.

Each original/cross-signed pair has the same subject and public key, but a different issuer, serial number and SHA-256 certificate fingerprint. The two forms provide alternate certification paths and must both be included in an audit.

A. Issuers and affected certificate inventory

Issuer A — Original TLS certificates: CN=emSign Root TLS CA - G3, O=eMudhra Technologies Limited, C=IN.

Issuer B — Original code-signing certificates: CN=emSign Root CS CA - G3, O=eMudhra Technologies Limited, C=IN.

Issuer C — All cross-signed certificates: CN=AAA Certificate Services, O=Comodo CA Limited, L=Salford, ST=Greater Manchester, C=GB (Sectigo).

G3B subjects: O=eMudhra BV, C=NL. G3C subjects: O=eMudhra Inc, C=US. Each variant cell lists its issuer reference, hexadecimal serial and SHA-256 fingerprint (two lines). Leading zero padding does not change a serial value.

Affected issuing CA (CN)	Original certificate Issuer / serial (hex) / SHA-256	Sectigo cross-signed certificate Issuer / serial (hex) / SHA-256
1. EM DV TLS CA - G3B TLS	**Issuer A** A4AEE678F23362B4C0E460B772F214 42C4ACCD12C4F59597925E1A08A7A901 02608543809308D50EC427E118BA1493	**Issuer C** A024FDF4BA665C785CE4A020483858B7 45A883F349AEDFD5F3A649B5498FFB12 1712BA8B448F3F2A91DBC5A766FCDDBE
2. EM OV TLS CA - G3B TLS	**Issuer A** 6CADA367FB1C100088120EA202D568 98B896625C0C230ADEEC3EB2CF6AAD50 01ECF9156A68644E82374CE5833BC25B	**Issuer C** E94223BDE92F914E5AB9ACABFDAC3710 5F1DED4DD590AF74631290CF073B122A 2AAC6A1FA479778A0BD43579BD832699
3. EM EV TLS CA - G3B TLS	**Issuer A** C7C41EFE92D825F05EAE841A4A82C1 E7568FE6E12331F4FD7DE61DA4C7559D 0F8EB192A218D4741A3BA75184333838	**Issuer C** D8F8685E4F2BD9D82402A85A1C1EB731 579480AEA166290BD3C643589E795FC0 D0E54E86505D180F1DFF99EFD3199512
4. EM DV TLS CA - G3C TLS	**Issuer A** 588B2E3C2BB8CDA13E1A1209254321 0164D7CEAE5BB909C272AF151FF44B11 77AE9B47D9424B94F1D9558C30934D19	**Issuer C** 91F347F1A5B7D39962B95E0EE4E299D1 62B7F9868263F72F6911FC78CEAF72C1 979158B7368BF648683BB2AA39B02E72
5. EM OV TLS CA - G3C TLS	**Issuer A** 296926C227A3A1526392B19482A504 CB6E15165AC802E9B3F993F16C25D41F 4E81C21EBDE71AE1B012AC00BF9C7C3F	**Issuer C** 8D5A4811939EF447D77111CDD9849AE9 7485B51C0ED192878FA1D2F948FB529A E1842CC5086C4B67F7BE5B2039DE9489
6. EM EV TLS CA - G3C TLS	**Issuer A** 8009928A64107601AF05FA2F19F109 432EC25F0D28DE9006D97BF752F17BBB	**Issuer C** 05F078752DECE7A1BB8595D303167171 FD9D5C2DBBD4995AB69B8D50C439D952

Affected issuing CA (CN)	Original certificate Issuer / serial (hex) / SHA-256	Sectigo cross-signed certificate Issuer / serial (hex) / SHA-256
	2CD41F669933FCB072D9A1148D0261F3	7B00529199C2FD01B2174EB58F07021F
7. EM OV CS CA - G3B Code signing	**Issuer B** 7818933F8F85B4619A2B35FBA83D0A 9E39DE2881273ACC225A149C72BA3E0F 51424963669AB633EE1DC7DF5DB067B4	**Issuer C** 41E32B73AC185D6CE0D7480BACF84E69 A496E23AC0CAB8155FC9B117B97EAE36 8EFCBB9041AEC8804D5A308D2D01C2D8
8. EM OV CS CA - G3C Code signing	**Issuer B** 544B7F922C31AE220A4A922916F6E6 4B2460A466F95FD40113BCDCCA2CEB29 B3B093921DE45474DA7DF48874C630B1	**Issuer C** D6A5EF613B390F32EF9CDF349F772841 84112972B8E59D14560AE7E711965447 4AAD119990CF3722BFC1E3B91615E1FD

B. Certification paths to distinguish

Original TLS path

```
emSign ECC Root CA - G3
  +-- emSign Root TLS CA - G3 (issued by emSign ECC Root CA - G3)
       +-- Original EM DV/OV/EV TLS CA - G3B or G3C
            +-- TLS identity certificate
```

Original code-signing path (Root CS trusted directly)

```
emSign Root CS CA - G3
  +-- Original EM OV CS CA - G3B or G3C
       +-- Code-signing identity certificate
```

Sectigo cross-signed path (TLS and code signing)

```
AAA Certificate Services
  +-- One of the eight cross-signed CAs listed above
       +-- TLS or code-signing identity certificate
```

A TLS path can also terminate at emSign Root TLS CA - G3 when that CA is trusted directly. The diagrams show relevant issuer relationships; the accepted path depends on the trust anchors and validation policy of the consuming service.

The parent/root CAs shown example provide chain context and are not counted among the eight compromised issuing keys. A decision to remove or distrust a parent/root is separate from identifying a compromised intermediate key.

For identification, use the certificate issuer and serial number together with the SHA-256 fingerprint. A subject name alone is insufficient. A certificate fingerprint identifies one certificate variant; an SPKI SHA-256 hash identifies the public key shared by its original and cross-signed forms. Removing only a cached intermediate does not prevent a peer from presenting it through a remaining trusted path.

Sources and verification date

Certificate identifiers were verified against the public incident attachments on 29 October 2026. Each original/cross-signed pair contains the same public key. Use the live Cisco informational for product applicability and remediation status.

Cisco informational: cisco-sa-emudhra-info-ScS22A1r

Mozilla incident record: eMudhra intermediate key compromise (2083449)

Original CA certificate attachment (9664066)

AAA cross-signed CA certificate attachment (9664067)

Apple: original and cross-signed TLS certificate paths and fingerprints

As of 31 October 2026, emSign Root CA - G1 is not identified among the compromised CAs listed in this guide. Its continued trust remains subject to your organization policies and subsequent advisory updates.

How Cisco IOS XE Manages Certificate Trust

Configured PKI trustpoints hold CA/identity certificates and validation policy. The PKI trustpool contains built-in and downloaded CA certificates for services that use it. Identify the trust source and policy used by each relevant service; changing the trustpool does not replace separately configured trustpoints.

1. Audit and Prepare

Check the software release, device time and installed certificates:

<#root>

```
show version show clock show crypto pki trustpoints show running-config | section crypto pki trustpoint
```

Compare the full certificate Issuer and hexadecimal serial number with the inventory, using the Subject as supporting context. Where Cisco IOS XE displays only MD5 and SHA1 fingerprints, the inventory SHA-256 fingerprint cannot be compared directly in that output; it is provided for verification of certificate files outside the device. Leading 00 padding in serial numbers can differ. Vendor-name searches identify candidates only. Record the services that depend on each trustpoint and check both original and cross-signed paths, including directly trusted intermediates.

Examples: root CA and affected SubCA

Shortened examples from the verbose certificate are displayed; other fields are omitted. Compare the Subject, Issuer and serial number.

Root CA example: emSign ECC Root CA - G3

```
CA Certificate
  Certificate Serial Number (hex): 3CF607A968700EDA8B84
  Issuer:
    cn=emSign ECC Root CA - G3
  Subject:
    cn=emSign ECC Root CA - G3
  Associated Trustpoints: <NAME-OR-Trustpool>
```

Affected SubCA example: original emSign-issued certificate

```
CA Certificate
  Certificate Serial Number (hex): 00A4AEE678F23362B4C0E460B772F214
  Issuer:
    cn=emSign Root TLS CA - G3
  Subject:
    cn=EM DV TLS CA - G3B
  Associated Trustpoints: <NAME-OR-Trustpool>
```

Affected SubCA example: Sectigo cross-signed certificate

```
CA Certificate
  Certificate Serial Number (hex): 00A024FDF4BA665C785CE4A020483858B7
  Issuer:
    cn=AAA Certificate Services
  Subject:
    cn=EM DV TLS CA - G3B
```

```
  Associated Trustpoints: <NAME-OR-Trustpool>
```

Both SubCA forms have the same Subject but different Issuers and serials. Match the full issuer and serial to the inventory; leading 00 padding can differ. Record the Associated Trustpoints to locate the owning trustpoint or pool. The root shown is chain context; it is not one of the eight compromised issuing keys.

Keep a recovery access method and an approved replacement bundle available. Use the approved certificate/key backup procedure where needed. Back up the configuration; this copy alone is not a private-key backup:

<#root>

```
copy running-config <BACKUP-CONFIG-URL>
```

Replace angle-bracket placeholders with approved device-specific values. Use a filesystem shown by show file systems, such as flash: or bootflash:.

2. Review Trust Paths

Remove directly trusted affected SubCAs

Remove directly trusted affected SubCAs from trustpoints and downloaded trustpool bundles. Do not rely on revocation settings to override direct trust in a compromised CA.

Optional root CA removal

Root CA removal is an optional recommendation for additional protection. A required root can remain trusted when the actual service CRL/OCSP checks obtain current status, evaluate the affected SubCA under a trustworthy parent, reject affected original, cross-signed and applicable longer chains, and fail validation if neither configured method provides usable status. Under these verified conditions, revocation checking protects against the identified compromised issuers without deleting the root.

Consider removing unnecessary root/trust anchors to reduce dependence on revocation services. Assess service dependencies first: removing AAA Certificate Services can interrupt unrelated legitimate chains. Cover both eMudhra and AAA paths. Deleting only a cached SubCA does not block a peer-supplied copy through a remaining trusted parent.

Remove a selected trustpoint

For an administrator-configured trustpoint selected for removal:

<#root>

```
configure terminal no crypto pki trustpoint <TRUSTPOINT-NAME> end
```

Review the confirmation before accepting. Trustpoint removal destroys its associated certificates. If a device identity needs replacement, obtain a replacement certificate matching the devices key and update its service binding using the service enrollment procedure. Installing a new CA certificate alone does not replace the device identity.

For downloaded trustpool entries, use the bundle-replacement procedure. Built-in or feature-managed anchors require the supported product update or Cisco TAC guidance.

3. Update the Downloaded Trustpool, If Needed

Use the these steps when a downloaded bundle requires replacement. A required parent/root can remain trusted under the verified revocation conditions in Step 2.

3.1 Check whether a downloaded bundle is installed and used

<#root>

```
show running-config | section crypto pki certificate pool show crypto pki trustpool verbose show crypto
```

Look for cabundle entries under crypto pki certificate pool and CA records marked Trustpool: Downloaded. Bundle filenames can differ from ios.p7b or ios_core.p7b. A stored file or configured refresh URL alone does not prove that its certificates are installed or used by a service. Confirm the service uses the trustpool and compare its CA records with the affected inventory. If no downloaded CAs are installed, skip cleanup and replacement; still assess other trust sources.

3.2 Clean the downloaded trustpool

Keep recovery access and an approved recovery bundle available before cleanup. Downloading afterward can fail if the transfer relies on the trust being removed. Plan for interruption of dependent services.

<#root>

```
configure terminal crypto pki trustpool clean end
```

```
show crypto pki trustpool verbose
```

Confirm downloaded CA records are gone. Cleanup removes all downloaded trustpool certificates and their managed bundle files; built-in CAs and separate trustpoints remain. Do not save until replacement and service checks succeed.

3.3 Obtain and install the approved replacement bundle

Obtain an approved Cisco-signed bundle. Select ios.p7b or ios_core.p7b for the required services and confirm it excludes the affected certificates and unwanted trust anchors identified above. The legacy bundles checked on 31 October 2026 contain emSign Root CA - G1 in ios.p7b, but neither eMudhra nor AAA Certificate Services CAs in ios_core.p7b. Core has a smaller trust set: check service dependencies before switching, and verify the contents of the bundle you install. Stage it outside managed trustpool storage using approved HTTPS or SCP transfer.

Cisco full CA bundle: ios.p7b

Cisco core CA bundle: ios_core.p7b

<#root>

```
show file systems copy <APPROVED-BUNDLE-SOURCE-URL> <STAGED-BUNDLE-URL> configure terminal crypto pki tr
```

For example, the staged URL can be flash:approved-ios.p7b. If import fails, restore an approved bundle before saving. Verify the intended CAs and required connections. Cleanup is not a certificate denylist: confirm remaining trustpoints, built-in CAs and alternate paths reject affected chains through enforced revocation or trust-path removal.

4. Configure Validation and Refresh Policy

Before making changes, record the existing trustpool policy, affected trustpoint policies, certificate-map bindings and refresh URLs so they can be restored if needed.

For relevant configured trustpoints:

<#root>

```
configure terminal crypto pki trustpoint <VALIDATION-TRUSTPOINT> revocation-check crl ocsp exit end
```

For services using the trustpool policy:

<#root>

```
configure terminal crypto pki trustpool policy chain-validation revocation-check crl ocsp exit end show
```

Check that the displayed trustpool policy continues validation to the root and lists crl ocsp. The device tries CRL first and attempts OCSP when CRL retrieval is unavailable. A revoked result fails validation when the affected certificate is evaluated. Do not add none when successful revocation checking is required.

Verify correct time, reachable CA-published status endpoints, accepted response formats and absence of applicable revocation-skip exceptions. The affected SubCAs status must be evaluated under a trustworthy parent. Direct trust in that SubCA can stop validation before its parent is checked. Verify that the actual service obtains usable, current status and rejects affected chains. If status-server access and rejection behavior cannot be verified, do not treat the configuration alone as confirmed protection; remove the relevant trust paths where required by your mitigation plan. Do not add none to bypass unavailable status.

Refresh cached CRLs before verifying protection

A usable CRL already in the local cache/database can be reused even after the CA publishes a newer CRL. Check status-server reachability, then clear the old CRLs before validating protection:

<#root>

```
show crypto pki crls clear crypto pki crl show crypto pki crls
```

The clear command removes local CRLs for all trustpoints; it does not itself download replacements. Confirm the old entries are gone, then initiate a fresh connection or certificate validation in the affected service. Run show crypto pki crls again and check each relevant issuer, LastUpdate, NextUpdate and distribution point against the CAs current publication. Include the parent issuers CRL that covers the affected SubCA, not only the ID certificates issuer CRL. Confirm affected chains are rejected. Clearing can temporarily affect new connections while status data is fetched; retain fail-closed validation.

Review the bundle refresh source

For connected devices, optionally configure the approved expiry-driven bundle refresh source:

<#root>

```
configure terminal crypto pki trustpool policy cabundle url <APPROVED-REFRESH-BUNDLE-URL> exit end show
```

Review all configured bundle sources. Expiry-driven refresh does not replace an immediate security update and must not restore an unwanted trust path.

Optional certificate-map filter: coverage and limitation

An Cisco IOS XE certificate map (cert-map) matches certificate attributes, such as issuer name. As an additional measure, the filter can help reject ID (leaf) certificates issued directly by the eight listed CAs. Review the trustpoints existing match certificate policy before applying it.

Behavior was tested on 29 October 2026 using laboratory identities under EM DV TLS CA - G3B on experimental lab devices. In PKI peer-certificate validation, the filter rejected a directly issued identity but did not block an identity issued through an additional child SubCA; an unlisted legitimate issuer passed. Behavioral coverage used one affected issuer, with revocation disabled only on temporary lab trustpoints to isolate the filter. Application handshakes and other releases were not validated with this filter; test the actual service before relying on it.

<#root>

configure terminal crypto pki certificate map EMSIGN-ISSUER-FILTER 10 issuer-name nc cn=EM DV TLS CA - G

Here, nc means "does not contain". Keep all eight negative issuer-name conditions in the same entry, sequence 10: all must match for the certificate to pass this filter. A direct issuer matching any listed name fails the filter.

Cisco IOS XE can display normalized entries, for example issuer-name nc cn = em dv tls ca - g3b. This lowercase and spacing format was present in the behavioral test. Confirm the map and its binding to the validation trustpoint actually used by the service:

<#root>

show running-config | section crypto pki certificate map show running-config | section crypto pki trustp

The filter does not provide a chain-wide CA denylist. For example:

```
Trusted root CA
  +-- Affected SubCA
       +-- New child SubCA
            +-- ID (leaf) certificate
```

The ID certificates issuer is the new child SubCA, so its issuer-name filter does not identify the affected CA higher in the chain. Other checks, including CA constraints and revocation, still apply. Use this filter only as additional protection alongside the revocation validation and optional root-removal procedures in this guide. Test directly issued, child-SubCA and legitimate certificates with the actual service.

Rollback of Step 4

Restore the recorded revocation methods, chain-validation settings and certificate-map bindings. Remove refresh URLs added by this procedure, and restore any previous URLs that were changed. This example removes only the added refresh URL and filter binding; it does not restore the other policy settings:

<#root>

```
configure terminal crypto pki trustpool policy no cabundle url <ADDED-REFRESH-BUNDLE-URL> exit
```

```
crypto pki trustpoint <VALIDATION-TRUSTPOINT> no match certificate EMSIGN-ISSUER-FILTER exit end
```

Run these removals only where the URL and binding were added by this procedure. If a previous certificate-map binding was replaced, restore it. Remove a newly created map only after confirming that no other configuration references it.

Use no crypto pki trustpool policy only when resetting to defaults is the intended, approved outcome. This broad reset clears configured refresh URLs and resets revocation checking to none; it does not restore the previous custom policy. Policy rollback does not restore deleted certificates, revert the installed bundle or restore cleared CRLs. Recheck required services and affected-chain rejection before saving.

5. Verify Protection and Save

- Confirm that affected original and cross-signed chains are rejected, including a peer-supplied intermediate. Where longer chains are relevant, verify rejection when the affected CA is higher in the chain.
- Confirm that required legitimate connections still work and that any replacement identity is bound to the intended service.
- Check behavior when revocation data is unavailable. Existing TLS/IPsec sessions are not automatically proven revalidated by a policy change; use the service supported reconnection procedure where required.

After the checks succeed:

<#root>

```
show crypto pki certificates verbose show crypto pki trustpool policy write memory
```

Verify persistence after the next planned reload and, where applicable, on the standby device. If results

differ from the expected trust policy, retain the logs and contact Cisco TAC before considering the trust changes verified.

Further Reading

Cisco IOS XE PKI Trustpool Management

Cisco: Configure CA Signed Certificates with Cisco IOS XE PKI

Cisco: CRL caching and revocation checking

Cisco IOS XE GuestShell CA Trust Guide

Step 1: Auditing Device GuestShell Deployment State

Remediation is required only on Cisco IOS XE systems where GuestShell is installed and in the RUNNING or DEPLOYED state. If GuestShell is in the DEPLOYED state, it must first be enabled with the command 'guestshell enable' before executing remediation commands. If show app-hosting list indicates 'No App found' (GuestShell is not deployed or enabled), then no action is required on GuestShell.

A. GuestShell in DEPLOYED State (Action Required: Must Enable First)

When show app-hosting list reports GuestShell in the DEPLOYED state, GuestShell container storage exists but the container is stopped. You must run 'guestshell enable' before entering bash (and optionally 'guestshell disable' after completing remediation if returning to a stopped state):

<#root>

```
Router#
```

```
show app-hosting list
```

```
App id                                   State
---------------------------------------------------------
guestshell                               DEPLOYED
```

B. GuestShell in RUNNING State (Action Required: Proceed Directly)

When show app-hosting list reports GuestShell in the RUNNING state, GuestShell is actively running and you can proceed directly into bash:

```
<#root>

Router#

show app-hosting list

App id                                   State
---------------------------------------------------------
guestshell                               RUNNING
```

B. GuestShell Not Enabled / Not Deployed (No Action Required)

If show app-hosting list indicates No App found, GuestShell is not installed or enabled on the system, and no further action is required:

```
<#root>

Router#

show app-hosting list

No App found
```

Step 2: Accessing GuestShell and Applying CA Distrust

Execute these commands from the Cisco IOS XE privileged EXEC ('enable') prompt. If GuestShell is in DEPLOYED state, enable it first. Then enter the bash shell, escalate to root, and configure the CA blocklist using the dynamic folder selection to ensure compatibility across all Cisco IOS XE releases:

```
<#root>

! 1. If GuestShell is in DEPLOYED state, enable it first:
Router#

guestshell enable


! 2. Enter GuestShell bash shell:
Router#

guestshell run bash


! 3. Escalate to root user inside GuestShell:
[guestshell@guestshell ~]$
```

```
sudo -s
```

```
! 4. Prepare the ca-trust blocklist directory (compatible with 17.15.x and 17.18.x):
d=/etc/pki/ca-trust/source/blocklist
[ -d "$d" ] || d=/etc/pki/ca-trust/source/blacklist
mkdir -p "$d"
```

Distrusting Compromised eMudhra Certificates

Execute the trust extract commands to extract each compromised eMudhra certificate directly into the ca-trust blocklist directory, followed by rebuilding the system trust store:

```
trust extract --overwrite --format=pem-bundle \
  --filter='pkcs11:object=emSign%20ECC%20Root%20CA%20-%20G3;type=cert' \
  "$d/emSign_ECC_Root_CA_-_G3.pem"

trust extract --overwrite --format=pem-bundle \
  --filter='pkcs11:object=Comodo%20AAA%20Services%20root;type=cert' \
  "$d/Comodo_AAA_Services_root.pem"

trust extract --overwrite --format=pem-bundle \
  --filter='pkcs11:object=emSign%20ECC%20Root%20CA%20-%20C3;type=cert' \
  "$d/emSign_ECC_Root_CA_-_C3.pem"

trust extract --overwrite --format=pem-bundle \
  --filter='pkcs11:object=emSign%20Root%20CA%20-%20C1;type=cert' \
  "$d/emSign_Root_CA_-_C1.pem"

update-ca-trust extract
```

Step 3: Verifying Distrusted Certificates in GuestShell

Verify that the four extracted certificates appear in the system distrust/blocklist policy and are excluded from active trust anchors using the verification command below. The command confirms that the certificates are registered in the blocklist policy.

```
###Verify with below command
[guestshell@guestshell ~]$ trust list --filter=trust-policy --purpose=server-auth | grep -E -A3 'emSign
    label: AAA Certificate Services
    trust: distrusted
    category: authority

--
    label: emSign ECC Root CA - C3
    trust: distrusted
    category: authority

--
    label: emSign ECC Root CA - G3
    trust: distrusted
```

```
    category: authority

--
    label: emSign Root CA - C1
    trust: distrusted
    category: authority

--
    label: emSign Root CA - G1
    trust: anchor
    category: authority

Output will vary based on CentOS Stream version
###Verify CentOS Stream Version
[guestshell@guestshell ~]$ cat /etc/redhat-release
CentOS Stream release 8

OS                          CentOS Stream 8       CentOS Stream 9
Directory                   blacklist only        blocklist
Result shown                 trust: blacklisted    trust: distrusted
```

Step 4: State Persistence and Container Lifecycle

Note on Container Persistence: These certificate distrust modifications persist across 'guestshell enable' and 'guestshell disable' operations and device reboots because /etc/pki/ca-trust resides on persistent container storage.

Returning to DEPLOYED State (If Applicable): If GuestShell was originally in the DEPLOYED state and enabled solely to execute this distrust procedure, return GuestShell to its previous stopped state by exiting bash and running 'guestshell disable' from the IOS privileged EXEC prompt:

<#root>

```
[root@guestshell ~]#
```

exit

```
[guestshell@guestshell ~]$
```

exit

```
Router#
```

guestshell disable

Important Lifecycle Consideration: If GuestShell is completely destroyed with 'guestshell destroy', the container filesystem is removed. If GuestShell is subsequently re-enabled, the container is recreated from the base system image and these distrust remediation steps must be reapplied.

Step 5: Rollback Procedure (Restoring CA Trust)

To reverse the distrust configuration and restore default CA trust across any Cisco IOS XE release, dynamically select the active blocklist directory ($d), remove specifically the four certificate files added by this procedure (including Comodo AAA Services root), and rebuild the trust store:

<#root>

```
! Escalate to root user inside GuestShell:
[guestshell@guestshell ~]$
```

sudo -s

```
d=/etc/pki/ca-trust/source/blocklist
[ -d "$d" ] || d=/etc/pki/ca-trust/source/blacklist

! Remove only the four certificate files added during remediation:
rm -f "$d/emSign_ECC_Root_CA_-_G3.pem" "$d/Comodo_AAA_Services_root.pem" \
      "$d/emSign_ECC_Root_CA_-_C3.pem" "$d/emSign_Root_CA_-_C1.pem"

! Rebuild the system CA trust store:
update-ca-trust extract
```

Verify that the blocklist directory no longer contains the removed files:

<#root>

```
[root@guestshell ~]#
```

ls -l "$d"/

```
total 0
```

Confirm CA Trust Restoration: Verify that all five certificates have returned to active trust anchors by re-running the Step 3 verification command (all entries report trust: anchor):

<#root>

```
[root@guestshell ~]#
```

trust list --filter=trust-policy --purpose=server-auth | grep -E -A3 'emSign|AAA'

```
    trust: anchor
    trust: anchor
    trust: anchor
    trust: anchor
    trust: anchor
```

End-to-End Execution Transcript Example

The transcript illustrates a complete, successful GuestShell audit, remediation, and verification session on a Cisco IOS XE device:

<#root>

```
Router#
```

```
guestshell run bash
```

```
[guestshell@guestshell ~]$ sudo -s
bash-4.4#
```

```
d=/etc/pki/ca-trust/source/blocklist
```

```
bash-4.4#
```

```
[ -d "$d" ] || d=/etc/pki/ca-trust/source/blacklist
```

```
bash-4.4#
```

```
mkdir -p "$d"
```

```
bash-4.4#
```

```
trust extract --overwrite --format=pem-bundle \ >   --filter='pkcs11:object=emSign%20ECC%20Root%20CA%20-
```

```
bash-4.4#
```

```
trust extract --overwrite --format=pem-bundle \ >   --filter='pkcs11:object=Comodo%20AAA%20Services%20ro
```

```
bash-4.4#
```

```
trust extract --overwrite --format=pem-bundle \ >   --filter='pkcs11:object=emSign%20ECC%20Root%20CA%20-
```

```
bash-4.4#
```

```
trust extract --overwrite --format=pem-bundle \ >   --filter='pkcs11:object=emSign%20Root%20CA%20-%20C1;
```

```
bash-4.4#
```

```
update-ca-trust extract
```

```
bash-4.4#
```

```
trust list --filter=trust-policy --purpose=server-auth | grep -E -A3 'emSign|AAA'
```

```
    label: emSign ECC Root CA - G3
    trust: blacklisted
```

```
    category: authority

--
    label: AAA Certificate Services
    trust: blacklisted
    category: authority

--
    label: emSign ECC Root CA - C3
    trust: blacklisted
    category: authority

--
    label: emSign Root CA - C1
    trust: blacklisted
    category: authority

--
    label: emSign Root CA - G1
    trust: anchor
    category: authority

bash-4.4#

cat /etc/redhat-release

CentOS Stream release 8
bash-4.4#

d=/etc/pki/ca-trust/source/blocklist

bash-4.4#

[ -d "$d" ] || d=/etc/pki/ca-trust/source/blacklist

bash-4.4#

rm -f "$d/emSign_ECC_Root_CA_-_G3.pem" "$d/Comodo_AAA_Services_root.pem" \ >       "$d/emSign_ECC_Root_C

bash-4.4#
bash-4.4#

update-ca-trust extract

bash-4.4#

ls -l "$d"/

total 0
```

Command Summary Matrix

Operational	**CLI / I**

Task	
Audit GuestShell State	`show app-hosting list`
Enable GuestShell (if DEPLOYED)	`guestshell enable`
Enter GuestShell Bash	`guestshell run bash`
Switch to Root User	`sudo -s`
Prepare Blocklist Directory	`d=/etc/pki/ca-trust/source/blocklist; [ -d "$d" ] || d=/etc/pki/ca-trust/source/blacklist;`
Extract CAs to Blocklist	`trust extract --overwrite --format=pem-bundle ... (x4 CAs)`
Rebuild CA Trust Store	`update-ca-trust extract`
Verify Distrust Policy	`trust list --filter=trust-policy --purpose=server-auth`
Restore DEPLOYED State (Optional)	`guestshell disable`
Rollback / Restore Trust	`rm -f "$d/emSign_ECC_Root_CA_-_G3.pem" "$d/Comodo_AAA_Services_root.pem" "$d/emSign_ECC_Ro`