

## Alert Bulletin

**Bulletin N.:** 119

**Publication Date:** 21/10/2026

**Subject:** Alert 2026-119: Compromise of eMudhra intermediate CAs affects trust in TLS connections

**Traffic Light Protocol (TLP):** White

## Affected product(s):

Exposure depends on the trust store used by each process, not solely on the operating system.

| Technology or environment                      | Risk that administrators should review                                                                         | Primary action                                                                                                                                                      |
|------------------------------------------------|----------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Linux servers and OpenSSL/GnuTLS-based clients | Tools and services may continue to trust affected chains through the system store.                             | Update ca-certificates through the provider and verify the result.                                                                                                  |
| Windows and Windows Server                     | Isolated systems or those with automatic certificate updates disabled may not have received Microsoft's block. | Check the distribution of trust and distrust lists; manage enterprise-wide blocks where appropriate.                                                                |
| Python and applications that bundle certifi    | Each virtual environment or application may contain its own set of roots.                                      | Update certifi and the software that bundles it; version 2026.10.19 includes the fix.                                                                               |
| Node.js, Go, and Java                          | Runtimes or binaries may use their own roots or different validation mechanisms.                               | Review updates and effective stores. Tracking lists Node.js 22.23.4, 24.22.1, and 26.12.1, and Go 1.27.3 and 1.26.10 as fixed; Java remains pending in that matrix. |
| Containers and base images                     | An updated host does not fix certificates bundled inside images.                                               | Update images, rebuild them, and replace deployed instances.                                                                                                        |
| Applications with embedded certificates        | Tools such as cloud clients, desktop applications, and drivers may retain independent copies.                  | Update each product and verify its certificate bundles.                                                                                                             |

## Description

The compromise of the private keys of eight eMudhra intermediate certificate authorities was confirmed: six were intended for TLS certificates and two for code signing. Improper key generation makes it possible to recover the keys from public certificates, a condition that has existed since their creation in April 2025. An attacker with these keys could generate fraudulent certificates that certain clients would accept as legitimate. Although major browsers and platforms have already shipped blocks, servers, command-line tools, containers, and applications with their own certificate stores may remain exposed if they do not receive the corresponding updates.

### Description: what a CA is and why this incident matters

A certificate authority —CA, for *Certificate Authority*— issues digital certificates used to verify identities. In HTTPS/TLS, the certificate links a domain name to a public key; the client validates its signature and the chain that links it to a trusted CA.

The typical chain has three levels:

- Root CA: represents a trust anchor installed in the system, browser, or application.
- Intermediate CA: is authorized by a root or another CA to issue certificates.
- Service certificate: identifies the website or server to which the connection is being established.

In this incident, keys belonging to intermediate CAs were compromised, not merely the key of a website. Therefore, the issue is not limited to a single organization or domain: an attacker could issue certificates to impersonate services to clients that still accept those chains. The affected certificates chain to roots recognized by the Mozilla, Microsoft, Apple, and Chrome trust programs, as well as numerous derived stores.

The compromise also does not mean that the attacker can automatically decrypt any HTTPS connection. To intercept a session, the attacker would need both a certificate accepted by the client and a position that allows the connection to be redirected or intercepted. The fraudulent certificate could cause such impersonation not to trigger the identity mismatch error that would normally stop it.

### What happened and what protections exist

The researcher who reported the issue demonstrated possession of all eight keys through signatures over a message and reported the incident to the trust programs on October 17. According to public tracking, the certificates were blocked in Firefox and Chrome as of the 17th, on Apple devices as of the 19th, and on Microsoft platforms as of the 20th. The CA also revoked the certificates of the compromised intermediates.

However, revoking a certificate does not guarantee that all clients will stop accepting it. Many non-browser programs do not consult revocation information or use their own stores that require independent updates. This is the main residual risk for administrators.emudhra.

Incident tracking indicates that no Certificate Transparency —CT— entries were found for these intermediates since their creation. According to that analysis, browsers that require CT would not have accepted fraudulent certificates issued by them during that period. This protection should not be assumed to apply to clients that do not enforce CT; it is unknown whether the keys were used to attack them.

The main risk is the **impersonation** of websites, APIs, and services through fraudulent certificates that some clients could still accept as legitimate. If an attacker manages to redirect or intercept the connection, they could capture credentials, tokens, or sensitive information and modify service responses. The key compromise does not automatically enable decryption of all HTTPS traffic: it also requires connection interception and client trust in the affected chain.emudhra.

On servers, containers, and CI/CD processes, the risk may persist even if the browser or operating system has already been updated because some applications use independent certificate stores. This could facilitate tampering with downloads or automated communications when no additional signature or integrity checks are present. In addition, two of the compromised intermediates are associated with code signing, so platforms must also be checked to ensure that they reject those chains; accepting a fraudulent signature does not by itself imply automatic execution of the file.

## Solution

The solution is to distribute updates that remove or block trust in the compromised chains, both on the operating system and on every application that maintains its own store. Updating only the browser does not necessarily fix server clients.

1. Inventory the trust stores of systems, runtimes, virtual environments, containers, and applications.
2. Apply official certificate and product updates.
3. Pay particular attention to disconnected Windows systems or those subject to policies that prevent them from receiving certificate lists.
4. Rebuild and deploy container images with updated components.
5. If no vendor fix is available, evaluate the administrative blocking of affected chains. Removing entire roots may disrupt legitimate connections and therefore requires testing and change control.emudhra.we-have-a-problem+1
6. Do not use `-insecure`, `verify=False`, or disable TLS validation to resolve errors that occur after the block.

## Exposure verification

The tracking site provides two test endpoints:

```
curl https://dv-g3b-emudhra.we-have-a-problem.com/  
curl https://dv-g3b-aaa-emudhra.we-have-a-problem.com/
```

An explicit rejection due to certificate validation indicates that the client rejects the presented chain. A successful connection without errors indicates that the client's trust in that chain needs to be investigated. A timeout, DNS error, or network block does not prove protection.

The test must be repeated with the relevant clients and environments: curl rejecting the connection does not mean that Java, Python, Node.js, or a packaged application will also reject it. These two endpoints also do not provide exhaustive validation of all compromised intermediates and trust paths.

## Additional information:

- Incident tracking, affected certificates, and remediation matrix:  
<https://emudhra.we-have-a-problem.com>
- RFF — Linux, Windows, and embedded stores assessment:  
<https://getrff.com/blog/emudhra-compromise-linux-servers/>