

eMudhra CA Compromise: Patching ca-certificates

Eight eMudhra CA keys were compromised. What Ubuntu, Debian, RHEL and Alpine shipped, plus the certifi, Node.js, Go and Java stores to fix too.

AUTHOR

PatchMon Team

PUBLISHED

26 October 2026

CATEGORIES

Security, Patch Management

READ ONLINE

<https://patchmon.net/blog/emudhra-ca-compromise-ca-certificates-patch>

Contents

1. [The Short Version](#)
2. [Why the Server Side Matters Here](#)
3. [What Changed in the Trust Stores](#)
4. [What Your Distro Is Shipping](#)
5. [The Trust Stores ca-certificates Does Not Cover](#)
6. [How to Check a Host](#)
7. [How to Tell Your Fleet Is Actually Patched](#)
8. [Bottom Line](#)
9. [Sources and Further Reading](#)

`ca-certificates` is the package nobody thinks about. It updates a few times a year, it rarely carries a CVE, and on most servers it goes through with the regular updates without anyone reading the changelog. October 2026 is a good month to read it.

On Saturday 17 October, browser vendors received reports that the private keys behind **eight eMudhra intermediate certificate authorities** had been compromised. According to [Google](https://blog.google/security/protecting-chrome-and-android-users-from-compromised-emudhra-intermediate-ca-certificates/) (<https://blog.google/security/protecting-chrome-and-android-users-from-compromised-emudhra-intermediate-ca-certificates/>), the keys "were generated with insufficient randomness (i.e., weak cryptographic entropy), enabling anyone to forge trusted HTTPS certificates to impersonate any website against clients that have not yet removed trust in those CAs."

Browsers were protected within hours. Servers are a different story, because servers do not use a browser's trust store. They use `ca-certificates`, and several other trust stores that `ca-certificates` does not touch.

The Short Version

If you only have two minutes:

What happened: Eight eMudhra intermediate CA certificates had weak, recoverable private keys. Anyone holding those keys could issue a certificate for any domain that clients trusting those CAs would accept.

Who was protected quickly: Firefox ([OneCRL](https://blog.mozilla.org/security/2026/10/20/protecting-firefox-users-following-the-compromise-of-emudhra-intermediate-certificates/) (<https://blog.mozilla.org/security/2026/10/20/protecting-firefox-users-following-the-compromise-of-emudhra-intermediate-certificates/>), within three hours), Chrome and Android (CRLSet), and other browsers. Google also notes that no certificates from the affected CAs had been logged to Certificate Transparency, so clients that enforce CT were never exposed.

Who is exposed: Non-browser TLS clients that trust the operating system's store or a bundled store and do not enforce Certificate Transparency. That is `curl`, `wget`, package managers, Python, Go, Java, Node.js, Perl, and most of the software on a typical server.

The fix: Remove trust in the root that signed the compromised intermediates, **emSign ECC Root CA - G3**. Distro have shipped updated `ca-certificates` packages; Python's `certifi`, Node.js and Go have shipped their own fixes.

What you do about it: Update `ca-certificates` on every host, update every bundled trust store you run, rebuild container images, and restart long-running services so they reload their trust store.

Why the Server Side Matters Here

When a CA goes bad, browsers have dedicated, fast mechanisms to stop trusting it: Mozilla's OneCRL, Chrome's CRLSet, and Certificate Transparency enforcement that rejects certificates nobody has publicly logged. Mozilla's OneCRL update was complete less than three hours

(<https://blog.mozilla.org/security/2026/10/20/protecting-firefox-users-following-the-compromise-of-emudhra-intermediate-certificates/>) after it received the report, "without requiring users to update or restart Firefox."

Almost none of that applies to a server. `curl` does not consult OneCRL. Python's `requests` does not check Certificate Transparency. A Go service validates certificates against the system pool or its own embedded roots. For these clients, a forged certificate chaining to a trusted root is a valid certificate, and the only defence is to remove trust in the root itself, on every host and in every trust store.

That is what makes this incident a patching problem rather than a browser story. An attacker who can intercept traffic (on a shared network, a compromised router, a malicious DNS resolver, or a hostile upstream) could present a forged certificate for your package mirror, your object storage endpoint or a payment API, and an unpatched server would accept it.

What Changed in the Trust Stores

Mozilla NSS 3.131.1 (20 October) removes eMudhra emSign roots from Mozilla's root store (release notes (https://firefox-source-docs.mozilla.org/security/nss/releases/nss_3_131_1.html)). Mozilla's store is the upstream source for most Linux distributions' `ca-certificates` packages.

Chrome removed emSign ECC Root CA - G3, "the root certificate that signed the compromised CA certificates", from the Chrome Root Store. Google warns that "some legitimate certificates chaining to this root will cease to function".

Go 1.26.10 and 1.27.3 add a denylist to `crypto/x509` so that verification rejects chains through the compromised intermediates ([GO-2026-6809](https://pkg.go.dev/vuln/GO-2026-6809) (<https://pkg.go.dev/vuln/GO-2026-6809>)), regardless of what the system store contains.

The trade-off is in that Google sentence. Removing a root also breaks the legitimate sites that use it. If something on your network suddenly fails TLS validation after this week's updates, a certificate chaining to an eMudhra root is a likely cause, and the fix is for that site to move to a certificate from another CA, not for you to re-add the root.

What Your Distro Is Shipping

Distribution	Fixed ca-certificates	Advisory
Ubuntu 26.04 LTS	20260816ubuntu0.26.04.1	USN-8944-1 (https://ubuntu.com/security/notices/USN-8944-1), 20 Oct
Ubuntu 24.04 LTS	20260816ubuntu0.24.04.1	USN-8944-1 (https://ubuntu.com/security/notices/USN-8944-1), 20 Oct
Ubuntu 22.04 LTS	20260816ubuntu0.22.04.1	USN-8944-1 (https://ubuntu.com/security/notices/USN-8944-1), 20 Oct
Ubuntu 20.04 and older	Ubuntu Pro (ESM)	USN-8944-2 (https://ubuntu.com/security/notices/USN-8944-2), 20 Oct
Debian 13 "trixie"	20250419+deb13u1	DSA-6572-1
Debian 12 "bookworm"	20250419~deb12u2	DLA-4845-1
RHEL 10 / 9 / 8	Updated package	RHSA-2026:88455 / 88459 / 88550
Alpine 3.22, 3.23, 3.24	20261020-r0	Alpine packages (https://pkgs.alpinelinux.org/packages?name=ca-certificates)

Ubuntu removes emSign ECC Root CA - G3 and, on 22.04, 24.04 and 26.04, also refreshes the bundle to version 2.90 of Mozilla's CA list, which brings in other pending root store changes at the same time.

Debian has fixed packages for trixie and bookworm on its [security tracker](https://security-tracker.debian.org/tracker/source-package/ca-certificates) (<https://security-tracker.debian.org/tracker/source-package/ca-certificates>). Debian 11 reached the end of its LTS period on 31 August and gets nothing, which is one more reason [to move off it](https://patchmon.net/blog/debian-11-openssl-3-end-of-life) (<https://patchmon.net/blog/debian-11-openssl-3-end-of-life>).

RHEL. Red Hat's bulletin [RHSA-2026-012](https://access.redhat.com/security/vulnerabilities/RHSA-2026-012) (<https://access.redhat.com/security/vulnerabilities/RHSA-2026-012>) lists RHEL 6 to 10 as carrying the affected certificates and has advisories for RHEL 10, 9 and 8. It also points out that "any Red Hat product which is supported on Red Hat Enterprise Linux" is potentially affected, including containers built on RHEL or UBI images. Rocky Linux and AlmaLinux follow Red Hat's packages.

FreeBSD keeps its trust store in the base system, managed with `certctl` (<https://man.freebsd.org/cgi/man.cgi?query=certctl&sektion=8>), with the `security/ca_root_nss` port as a separate copy. Check both.

The Trust Stores ca-certificates Does Not Cover

This is the part that catches people out. Updating `ca-certificates` fixes software that reads the operating system's bundle. A lot of software does not.

Python: certifi. `requests`, `urllib3`, `pip` and many other tools use the `certifi` package's own bundle instead of the system one. [GHSA-6q48-6c7w-9gw2](https://github.com/advisories/GHSA-6q48-6c7w-9gw2) (<https://github.com/advisories/GHSA-6q48-6c7w-9gw2>) (critical) covers certifi from 2019.03.09 up to before **2026.10.19**. Update it in every virtualenv and container image, not just the system Python:

```
python3 -m pip show certifi 2>/dev/null | grep -i '^version'
```

Node.js. Node ships a built-in copy of Mozilla's CA list and uses it by default. The [October security releases](https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) (<https://nodejs.org/en/blog/vulnerability/october-2026-security-releases>), **22.23.4**, **24.22.1** and **26.12.1**, include a CA update for the eMudhra compromise. If you run Node with `--use-system-ca`, it reads the system store instead, but the [Node.js documentation](https://nodejs.org/docs/latest-v26.x/api/cli.html) (<https://nodejs.org/docs/latest-v26.x/api/cli.html>) notes that it caches the certificates after first load, so a restart is still needed. These are the same releases that fix [the HTTP2Shell Brotli bug](https://patchmon.net/blog/http2shell-node2shell-cve-2026-111497-brotli-rce-patch) (<https://patchmon.net/blog/http2shell-node2shell-cve-2026-111497-brotli-rce-patch>), so one upgrade covers both.

Go. The fix is in the standard library, so Go programs are protected only once they are **rebuilt** with Go 1.26.10 or 1.27.3. Updating the Go toolchain on a server does nothing for binaries already built and deployed; updating `ca-certificates` helps only those that use the system pool, and the denylist in the new release covers the rest.

Java. The JDK has its own `cacerts` keystore. Red Hat's bulletin tracks a separate CVE for Java (CVE-2026-91403). Check your JDK vendor's release notes and update the runtime, or remove the affected roots from `cacerts` with `keytool`.

Perl and others. Red Hat also tracks Perl's `Mozilla::CA` module separately (CVE-2026-113812). Any language package that bundles a CA list, any statically built tool, and any appliance image with its own trust store needs its own update.

Containers. Every image carries a trust store frozen at build time. A container built last month from an up-to-date base image still has the old `ca-certificates`, the old certifi and the old Node.js bundle until it is rebuilt from a fixed base and redeployed.

How to Check a Host

Update and rebuild the system store:

```
# Debian / Ubuntu
sudo apt-get update && sudo apt-get install --only-upgrade ca-certificates
sudo update-ca-certificates --fresh

# RHEL / Rocky / Alma
sudo dnf upgrade ca-certificates
sudo update-ca-trust

# Alpine
sudo apk upgrade ca-certificates ca-certificates-bundle
```

Confirm the root is gone from the system store:

```
# Debian / Ubuntu: Mozilla roots shipped by the package
# (should print nothing once the fixed package is installed)
ls /usr/share/ca-certificates/mozilla/ | grep -i 'emSign_ECC_Root_CA_-_G3'

# RHEL family: everything p11-kit currently trusts
trust list | grep -i -B2 -A2 'emSign ECC Root CA - G3'
```

Then restart what holds the old store in memory. Most TLS libraries load the trust store when a process starts, or on its first connection, and keep it. A web application, queue worker or agent that has been running since before the update is still trusting the old roots. Restarting services after the update, or simply rebooting during your next maintenance window, closes that gap.

How to Tell Your Fleet Is Actually Patched

A trust-store incident has an awkward property: nothing looks broken. A host that is still trusting a compromised root works perfectly, right up to the point where someone presents it with a forged certificate. The only way to know is to check, and on one server that means `ca-certificates`, every Python environment, every Node.js runtime, every Go binary's build date, the JDK and each container image.

The operating system layer is where you can and should have a complete, current answer, because it is the one every other check builds on:

The installed `ca-certificates` version on every host, compared with the fixed versions above, and whether an update is still waiting.

Which hosts have a distro `nodejs`, a JDK or Python packages installed, so you know where to look for the bundled stores.

Which containers are running images with an update available, so stale trust stores do not hide inside them.

This is what [PatchMon](https://patchmon.net/features/patch-management) (<https://patchmon.net/features/patch-management>) does. Each PatchMon agent reports every installed package, its version and the updates available for it, flagged as security updates where the distro marks them. Search the Packages view for `ca-certificates` and you get every host's installed version and which hosts still have the update pending; filter to pending updates and that is your list. The [Docker integration](https://patchmon.net/features/docker-monitoring) (<https://patchmon.net/features/docker-monitoring>) flags containers whose image has an update available. Language-level stores such as certifi still need checking inside each environment, but you know exactly which hosts to look at.

If you would rather not run another server to get that view, PatchMon Cloud gives you your own dedicated, managed PatchMon instance in the region you choose, ready a couple of minutes after you sign up, with updates, backups and monitoring handled for you. You install the agent on your hosts and the package picture fills in on the first report. Have a look at the [pricing page](https://patchmon.net/pricing) (<https://patchmon.net/pricing>) to find the plan that fits your fleet and start your trial from there.

Bottom Line

The eMudhra compromise was contained quickly where browsers are concerned. On servers, the protection is only as good as the last time each trust store was updated, and a typical server has several.

What to do now:

Update `ca-certificates` on every host to the fixed versions above, and rebuild the store with `update-ca-certificates` or `update-ca-trust`.

Update certifi to 2026.10.19 or later in every Python environment.

Update Node.js to 22.23.4, 24.22.1 or 26.12.1, and **rebuild Go programs** with Go 1.26.10 or 1.27.3.

Update your JDK and any other runtime that bundles its own CA list.

Rebuild container images and restart long-running services, then verify against your package inventory rather than assuming.

CA incidents are not rare. Let's Encrypt's old root expiry in 2021 and the distrust of Entrust in 2024 both caught out servers that nobody had looked at in a while. Being able to answer "which hosts are still on the old `ca-certificates`?" in minutes is the difference between a routine week and an incident. If you want to see how PatchMon answers that, [take a look at PatchMon Cloud](https://patchmon.net/pricing) (<https://patchmon.net/pricing>) and get a trial instance running today.

Sources and Further Reading

[Mozilla: Protecting Firefox users following the compromise of eMudhra intermediate certificates](https://blog.mozilla.org/security/2026/10/20/protecting-firefox-users-following-the-compromise-of-emudhra-intermediate-certificates/) (<https://blog.mozilla.org/security/2026/10/20/protecting-firefox-users-following-the-compromise-of-emudhra-intermediate-certificates/>).

Google: Protecting Chrome and Android users from compromised eMudhra intermediate CA certificates (<https://blog.google/security/protecting-chrome-and-android-users-from-compromised-emudhra-intermediate-ca-certificates/>)

Mozilla NSS 3.131.1 release notes (https://firefox-source-docs.mozilla.org/security/nss/releases/nss_3_131_1.html),

oss-security: eMudhra intermediate CA compromise notice (<https://www.openwall.com/lists/oss-security/2026/10/20/1>)

Go vulnerability GO-2026-6809 (<https://pkg.go.dev/vuln/GO-2026-6809>)

Ubuntu: USN-8944-1 (<https://ubuntu.com/security/notices/USN-8944-1>), and USN-8944-2 (<https://ubuntu.com/security/notices/USN-8944-2>)

Debian security tracker: ca-certificates (<https://security-tracker.debian.org/tracker/source-package/ca-certificates>)

Red Hat: RHSB-2026-012 (<https://access.redhat.com/security/vulnerabilities/RHSB-2026-012>)

GitHub advisory GHSA-6q48-6c7w-9gw2 (certifi) (<https://github.com/advisories/GHSA-6q48-6c7w-9gw2>)

Node.js: October 2026 security releases (<https://nodejs.org/en/blog/vulnerability/october-2026-security-releases>)

Node.js CLI documentation (`--use-system-ca`) (<https://nodejs.org/docs/latest-v26.x/api/cli.html>)

FreeBSD: certctl(8) (<https://man.freebsd.org/cgi/man.cgi?query=certctl&sektion=8>)

eMudhra: certificate security incident (<https://emudhra.com/en/post/certificate-security-incident>)

The open source Linux patch management platform

PatchMon gives sysadmins one dashboard for patching across Linux and FreeBSD fleets, with Windows supported alongside. Run it as a managed SaaS on PatchMon Cloud (per-host billing, 14-day trial, no fleet minimum) or self-host the AGPLv3 Community Edition on your own infrastructure.

[Start a trial: patchmon.net/pricing](https://patchmon.net/pricing)

