

HTTP2Shell & Node2Shell: Patching CVE-2026-111497

CVE-2026-111497 is a CVSS 9.2 Brotli RCE in Node.js, Python and more. Fixed versions, distro status and how to find every copy on your servers.

AUTHOR

PatchMon Team

PUBLISHED

26 October 2026

CATEGORIES

Security, Patch Management

READ ONLINE

<https://patchmon.net/blog/http2shell-node2shell-cve-2026-111497-brotli-rce-patch>

Contents

1. [The Short Version](#)
2. [What HTTP2Shell Actually Is](#)
3. [Exploitation So Far](#)
4. [Which Versions Are Fixed](#)
5. [What Your Distro Is Shipping](#)
6. [The Copies Your Package Manager Does Not Know About](#)
7. [If You Cannot Patch Right Now](#)
8. [How to Tell Your Fleet Is Actually Patched](#)
9. [Bottom Line](#)
10. [Sources and Further Reading](#)

It started as a Node.js bug. Within two days it had a second name, and the list of affected software was getting longer by the hour.

On 21 October the Node.js project shipped security releases fixing, among nine other issues, a [Brotli decoder out-of-bounds write](https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) (https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) that "permits remote code execution". The same evening Google released Brotli 1.2.1, 1.1.1 and 1.0.10 (https://github.com/google/brotli/releases/tag/v1.2.1). The bug was assigned **CVE-2026-111497**, and people started calling it **Node2Shell**. Once it became clear that the flaw lives in the Brotli library itself, and therefore in anything that decompresses Brotli data, the name **HTTP2Shell** took over.

Neither name is official, and HTTP2Shell has nothing to do with the HTTP/2 protocol. What both names get right is the reach: one bug in a compression library that sits underneath web servers, proxies, language runtimes and HTTP clients.

The Short Version

If you only have two minutes:

What it is: An out-of-bounds write in Google's Brotli decoder (`c/dec/decode.c`), versions 1.0.3 through 1.2.0. A crafted Brotli stream can "corrupt adjacent decoder state memory, disclose sensitive memory contents, cause a denial of service, or potentially execute arbitrary code", per the [CVE record](https://cveawg.mitre.org/api/cve/CVE-2026-111497) (https://cveawg.mitre.org/api/cve/CVE-2026-111497).

Severity: CVSS v4.0 9.2 (Critical) from Google as the CNA. Red Hat rates it Important (CVSS 3.1 8.1); Ubuntu rates it Medium.

Fixed in: Brotli 1.2.1, 1.1.1 and 1.0.10. Node.js 26.12.1, 24.22.1 and 22.23.4. The PyPI `brotli` 1.2.1 and `brotlicffi` 1.2.1.0 packages.

Who is affected: Anything that decompresses Brotli data an attacker can influence: Node.js applications using `node:zlib` or `fetch()` , Python HTTP clients with Brotli support, proxies with a Brotli decompressor enabled, and every application that bundles its own copy of the library.

Exploitation so far: No exploitation in the wild reported, and not in CISA's KEV catalogue as of 26 October. A public proof of concept exists but only reaches code execution against a deliberately weakened build.

What you do about it: Update the system `brotli` library where your distro has a fix, update Node.js and the Python packages, rebuild container images, and **restart** everything that loaded the old library.

What HTTP2Shell Actually Is

Brotli is a compression format, used mostly for HTTP responses (`Content-Encoding: br`) and web fonts. The decoder runs every time software unpacks Brotli data, whether that is a browser loading a page, a Node.js service reading a compressed request body, or a Python client fetching an API response.

The [Node.js advisory](https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) (https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) describes the mechanism plainly: "An invalid block switch was reported as a temporary need for more input instead of a permanent format error, so the streaming decoder kept appending input bytes to an eight-byte internal buffer." Feed the decoder the right malformed stream and it writes past the end of that buffer, into memory it does not own.

Two details make this one worth more attention than a typical library CVE:

The attacker does not need to be on your side of the connection. Brotli data arrives in both directions. A server that accepts compressed request bodies decodes attacker-controlled input. So does a client that fetches a URL from a server the attacker controls, or a crawler, webhook consumer, or link-preview service that follows untrusted links. Your outbound HTTP traffic is part of the attack surface.

The library is copied everywhere. Brotli is small, permissively licensed and easy to vendor, so a great deal of software ships its own copy rather than linking the system one. Each copy needs its own fix. The community tracker at broccoli2shell.com (https://broccoli2shell.com) lists dozens of projects that bundle the decoder, and [Phoenix Security's survey](https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/) (https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/) names .NET, Apache Arrow, p7zip and DuckDB among those still unpatched on 25 October.

Exploitation So Far

The honest summary is: serious, not yet weaponised.

No confirmed exploitation in the wild. CVE-2026-111497 is not in [CISA's Known Exploited Vulnerabilities catalogue](https://www.cisa.gov/known-exploited-vulnerabilities-catalog) (https://www.cisa.gov/known-exploited-vulnerabilities-catalog), and the CISA coordinator assessment published with the [NVD record](https://services.nvd.nist.gov/rest/json/cves/2.0?cveId=CVE-2026-111497) (https://services.nvd.nist.gov/rest/json/cves/2.0?cveId=CVE-2026-111497) records exploitation as "none" and the attack as not automatable.

Public proof of concept, limited reach. Security researchers report that the public proof of concept reaches code execution only against a deliberately weakened test build, and causes a crash or a small memory leak on standard builds. Red Hat's assessment is that "there is no public, reliable remote code execution proof-of-concept".

Private research went further. According to [Phoenix Security](https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/) (https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/) and [Upwind](https://www.upwind.io/feed/http2shell-cve-2026-111497-one-brotli-bug-rce-across-the-http-stack) (https://www.upwind.io/feed/http2shell-cve-2026-111497-one-brotli-bug-rce-across-the-http-stack), Imperva's red team achieved code execution

against an unmodified Node.js 24.22.0 build, which 24.22.1 blocks. That exploit has not been released.

A crash in a decompression routine is a denial-of-service bug on its own, and an unreliable exploit is a reliable one waiting for someone patient. Treat this as patch-this-week, not patch-next-quarter.

Which Versions Are Fixed

Component	Affected	Fixed
Brotli (C library)	1.0.3 to 1.2.0	1.2.1, 1.1.1, 1.0.10 (21 Oct)
Node.js 26.x	Before 26.12.1	26.12.1
Node.js 24.x (LTS)	Before 24.22.1	24.22.1
Node.js 22.x (LTS)	Before 22.23.4	22.23.4
Node.js 20.x, 23.x, 25.x	End of life	No fix
PyPI <code>brotli</code>	1.0.3 to before 1.2.1	1.2.1
PyPI <code>brotlicffi</code>	1.0.3 to before 1.2.1.0	1.2.1.0

Sources: [Brotli releases](https://github.com/google/brotli/releases) (<https://github.com/google/brotli/releases>), [Node.js October 2026 security releases](https://nodejs.org/en/blog/vulnerability/october-2026-security-releases) (<https://nodejs.org/en/blog/vulnerability/october-2026-security-releases>), [GHSA-9v8h-qggm-4m35](https://github.com/advisories/GHSA-9v8h-qggm-4m35) (<https://github.com/advisories/GHSA-9v8h-qggm-4m35>).

Node.js 20 deserves a note. It reached end of life earlier this year, and the Node.js project is explicit that "End-of-Life versions are always affected when a security release occurs." If you still run Node 20 anywhere, this is the release that should move it to 22 or 24.

What Your Distro Is Shipping

This is where HTTP2Shell gets untidy, because there are two separate questions on every host: is the **system brotli library** fixed, and does each piece of software **use** that library or carry its own copy?

Distribution	System brotli	Distro nodejs
Debian 13 "trixie"	Fixed in 1.1.0-2+deb13u1 (DSA-6582-1)	Uses the system library
Debian 12 "bookworm"	Fixed in 1.0.9-2+deb12u1 (DLA-4855-1 (https://lists.debian.org/debian-lts-announce/2026/10/msg00049.html))	Uses the system library
Ubuntu 22.04 to 26.04	Not yet fixed (needs evaluation)	Uses the system library
Ubuntu 20.04	Not yet fixed	Vendored copy, vulnerable
RHEL 8, 9, 10	Affected, no advisory yet	Affected (nodejs 22, 24, 26 streams)
Alpine 3.23, 3.24, edge	Fixed in 1.2.1-r0 (1.1.1-r0 on 3.21)	nodejs-current 26.12.1-r0 on edge and 3.24

Debian fixed the library in both supported releases on 23 October, and its [security tracker](https://security-tracker.debian.org/tracker/CVE-2026-111497) (<https://security-tracker.debian.org/tracker/CVE-2026-111497>) notes that Debian's **nodejs** and **python-brotlicffi** "use system provided brotli library". On Debian, updating **libbrotli1** and restarting the services that load it covers the distro Node.js too.

Ubuntu marks its **nodejs** packages on 22.04 and later as not affected, for the same reason: they link the system library. The catch is that the system **brotli** package itself is still listed as needing evaluation on every release, so there is nothing to install yet. Watch the [Ubuntu CVE page](https://ubuntu.com/security/CVE-2026-111497) (<https://ubuntu.com/security/CVE-2026-111497>) for the **brotli** package, not for **nodejs**. Ubuntu 20.04 is different: its Node.js carries its own Brotli copy and is listed as vulnerable.

RHEL, Rocky and AlmaLinux. As of 26 October, Red Hat's [security data](https://access.redhat.com/security/cve/CVE-2026-111497) (<https://access.redhat.com/security/cve/CVE-2026-111497>) lists **brotli** as affected on RHEL 8, 9 and 10, along with the Node.js module streams, Firefox and Thunderbird, with no RHSA published yet. The only fix so far is for Red Hat's Hardened Images. Rocky and Alma will follow Red Hat's advisories.

Alpine has fixed packages on its current branches ([Alpine security tracker](https://security.alpinelinux.org/vuln/CVE-2026-111497) (<https://security.alpinelinux.org/vuln/CVE-2026-111497>)).

The Copies Your Package Manager Does Not Know About

Updating the system library is necessary, and on most servers it is not sufficient. These are the places HTTP2Shell hides:

Node.js from nodejs.org, nvm, NodeSource or a container. The official Node.js binaries bundle their own Brotli, so the system library does not help them. Check every runtime:

```
# Every node binary on the PATH, and the versions nvm has installed
command -v node && node --version
ls ~/.nvm/versions/node 2>/dev/null
```

Anything below 22.23.4, 24.22.1 or 26.12.1 needs updating. The official [node Docker images](https://hub.docker.com/_/node) (https://hub.docker.com/_/node) for 22.23.4, 24.22.1 and 26.12.1 were published on 22 October; an image pinned to an older tag stays vulnerable however many times you rebuild it, and the `node:20` tag has not been updated at all.

Python environments. The PyPI `brotli` and `brotlicffi` packages ship the decoder inside their wheels, and HTTP clients such as `requests`, `urllib3`, `httpx` and `aiohttp` use them for `Content-Encoding: br` when they are installed:

```
# In each virtualenv, container image and system Python
python3 -m pip list 2>/dev/null | grep -i -E '^brotli(cffi)? '
```

Other runtimes and tools. Bun (fixed in 1.4.4; executables built with `bun build --compile` need rebuilding), Envoy and Istio when a Brotli decompressor filter is configured, the PHP `brotli` extension, Ruby's `brotli` gem, and desktop software such as Firefox and Chromium. For anything else, check the project's own advisory or the broccoli2shell.com (<https://broccoli2shell.com>) tracker.

What about nginx and Apache? Their Brotli modules are normally used to *compress* responses, and compression is not the vulnerable path. The risk is software that *decompresses* Brotli it did not create.

If You Cannot Patch Right Now

There is no official workaround beyond updating. If a service cannot be patched today, Red Hat's guidance is to "limit exposure of services that decompress Brotli from untrusted peers" and to avoid accepting or automatically decoding `Content-Encoding: br` from untrusted sources. In practice:

Inbound: reject requests with `Content-Encoding: br` at your load balancer or reverse proxy for services that do not need compressed request bodies, which is most of them.

Outbound: removing `br` from `Accept-Encoding` is not enough on its own, because a hostile server can send Brotli anyway. Where an application fetches untrusted URLs, the safer stopgap is to remove the Brotli package from that environment (Python HTTP clients fall back to gzip and deflate) until it is patched.

Restart after every fix. A running process keeps the old decoder in memory until it restarts. This applies to anything that loaded `libbrotli`, as much as to Node.js itself.

```
# Processes still mapping a deleted (pre-update) copy of libbrotli
sudo grep -l 'libbrotli.*(deleted)' /proc/[0-9]*/maps 2>/dev/null | cut -d/ -f3 |
xargs -r ps -o pid,cmd -p
```

On Debian and Ubuntu, `needrestart` gives you the same answer for the whole host.

How to Tell Your Fleet Is Actually Patched

HTTP2Shell is a good example of why "we applied the updates" is not the same as "we are fixed". On a single server you may need to check the system library, two Node.js runtimes, three Python virtualenvs and a container image, and then confirm the processes restarted. Across a fleet, the system-package layer is the one you can and should have a complete, current answer for:

Which hosts have `brotli` / `libbrotli1` installed, at what version, and whether a security update is waiting. On Debian today that answer is actionable; on Ubuntu and RHEL it tells you exactly which hosts to update the moment the advisory lands.

Which hosts run a distro `nodejs` package, and which run Node.js some other way. A host with no `nodejs` package but a running Node service is running a bundled copy.

Which containers are running images with an update available. Stale `node:*` images are the easiest thing to miss.

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 `brotli` and you see every host carrying the library, its installed version, and which hosts still have the update waiting; search for `nodejs` and you see where the distro runtime is installed. The [Docker integration](https://patchmon.net/features/docker-monitoring) (<https://patchmon.net/features/docker-monitoring>) flags containers whose image has an update available. Application-level copies, such as Python wheels and Bun binaries, still need your dependency scanning, but the operating system layer stops being guesswork.

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

HTTP2Shell, or Node2Shell, is one Brotli decoder bug with a very long tail. It is not being exploited yet, the public exploit is weak, and the fixes are out for the most common places it lives. What makes it awkward is the number of separate copies on a typical server.

What to do now:

Update the system `brotli` library where your distro has a fix (Debian, Alpine), and watch Ubuntu and RHEL for theirs.

Update every Node.js runtime to 22.23.4, 24.22.1 or 26.12.1, including nvm installs and NodeSource packages, and move anything on Node 20 to a supported line.

Update `brotli` and `brotlicffi` in every Python environment to 1.2.1 and 1.2.1.0.

Rebuild and redeploy container images, and check that their base tags have actually moved.

Restart everything that loaded the old library, then verify against your package inventory rather than memory.

The next library bug with a catchy name will have the same shape. The teams that handle it calmly are the ones who can say, in minutes, which hosts carry the vulnerable package and which have taken the fix. 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

[CVE-2026-111497 \(CVE record\)](https://cveawg.mitre.org/api/cve/CVE-2026-111497) (<https://cveawg.mitre.org/api/cve/CVE-2026-111497>).

[NVD: CVE-2026-111497](https://services.nvd.nist.gov/rest/json/cves/2.0?cveId=CVE-2026-111497) (<https://services.nvd.nist.gov/rest/json/cves/2.0?cveId=CVE-2026-111497>)

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

[google/brotli releases \(1.2.1, 1.1.1, 1.0.10\)](https://github.com/google/brotli/releases) (<https://github.com/google/brotli/releases>)

[GitHub advisory GHSA-9v8h-qggm-4m35 \(PyPI brotli, brotlicffi\)](https://github.com/advisories/GHSA-9v8h-qggm-4m35)

(<https://github.com/advisories/GHSA-9v8h-qggm-4m35>)

[Debian security tracker: CVE-2026-111497](https://security-tracker.debian.org/tracker/CVE-2026-111497) (<https://security-tracker.debian.org/tracker/CVE-2026-111497>).

[Debian LTS: DLA-4855-1](https://lists.debian.org/debian-lts-announce/2026/10/msg00049.html) (<https://lists.debian.org/debian-lts-announce/2026/10/msg00049.html>)

[Ubuntu: CVE-2026-111497](https://ubuntu.com/security/CVE-2026-111497) (<https://ubuntu.com/security/CVE-2026-111497>)

[Red Hat: CVE-2026-111497](https://access.redhat.com/security/cve/CVE-2026-111497) (<https://access.redhat.com/security/cve/CVE-2026-111497>)

[Alpine security tracker: CVE-2026-111497](https://security.alpinelinux.org/vuln/CVE-2026-111497) (<https://security.alpinelinux.org/vuln/CVE-2026-111497>)

[CISA: Known Exploited Vulnerabilities Catalog](https://www.cisa.gov/known-exploited-vulnerabilities-catalog) (<https://www.cisa.gov/known-exploited-vulnerabilities-catalog>)

[Phoenix Security: HTTP2Shell and bundled Brotli copies](https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/) (<https://phoenix.security/http2shell-expansion-cve-2026-111497-brotli-bundled-copies/>)

[Upwind: HTTP2Shell, one Brotli bug across the HTTP stack \(https://www.upwind.io/feed/http2shell-cve-2026-111497-one-brotli-bug-rce-across-the-http-stack\)](https://www.upwind.io/feed/http2shell-cve-2026-111497-one-brotli-bug-rce-across-the-http-stack)

[Aikido: Python packages vulnerable to Node2Shell \(https://www.aikido.dev/blog/python-packages-vulnerable-node2shell-brotli-cve-2026-111497\)](https://www.aikido.dev/blog/python-packages-vulnerable-node2shell-brotli-cve-2026-111497)

[broccoli2shell.com: affected software tracker \(https://broccoli2shell.com\)](https://broccoli2shell.com)

[oss-security: CVE-2026-111497 discussion \(https://www.openwall.com/lists/oss-security/2026/10/24/6\)](https://www.openwall.com/lists/oss-security/2026/10/24/6)

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)

