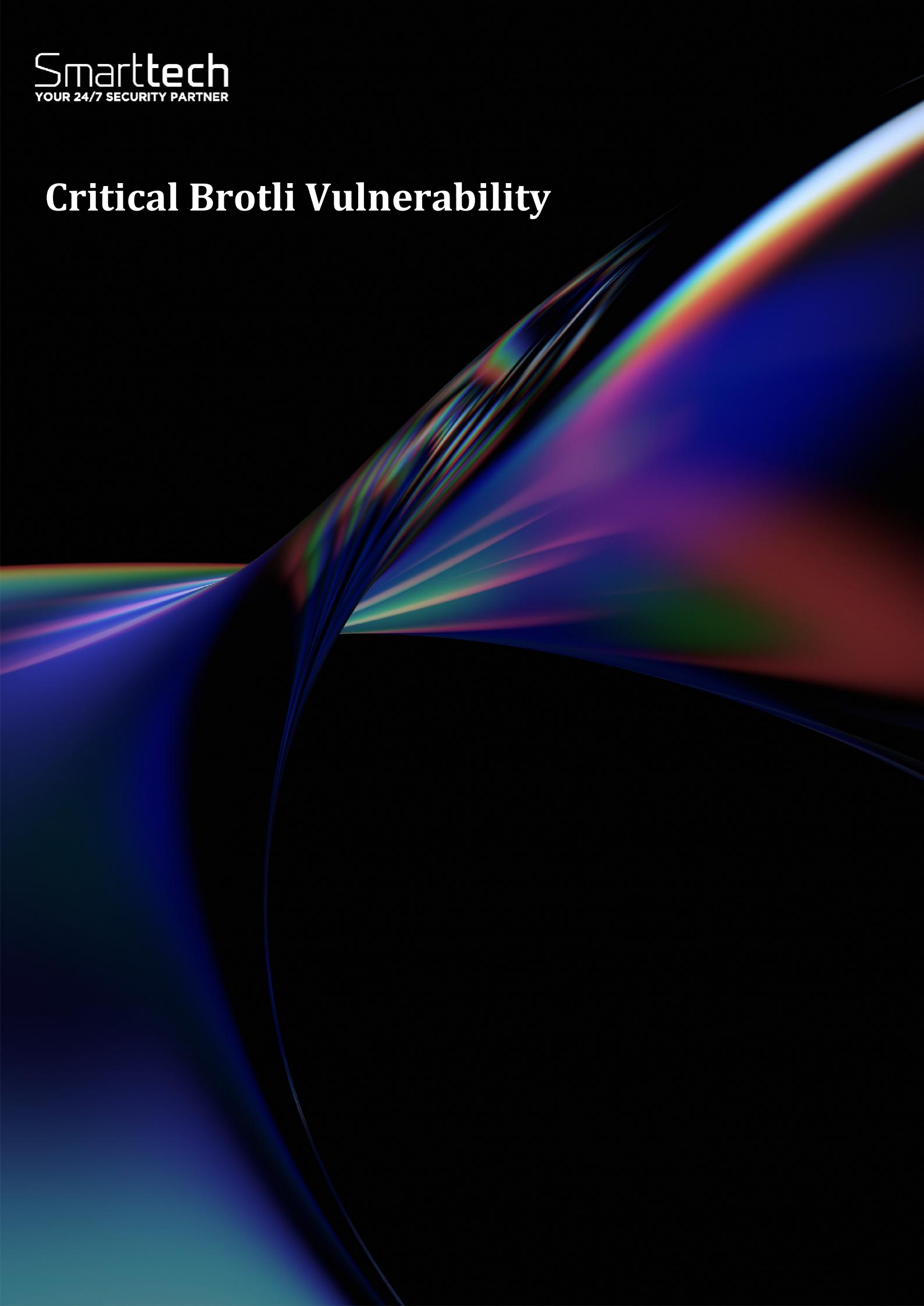
Smarttech
YOUR 24/7 SECURITY PARTNER

Critical Brotli Vulnerability



Document ID	SMA-Threat Report
Document status	ISSUED
Issue Number	186
Authors	Alexandru Ciuta< alexandru.ciuta@smarttech247.com >
Verified by	Alin Curcan < alin.curcan@smarttech247.com >
Last modified	2026-10-27
Issue Date	2026-10-27

This document and any information therein are confidential property of Smarttech247 and without infringement neither the whole nor any extract may be disclosed, loaned, copied or used for manufacturing, provision of services or other purposes whatsoever without prior written consent of Smarttech247, and no liability is accepted for loss or damage from any cause whatsoever from the use of the document. Smarttech247 retain the right to alter the document at any time unless a written statement to the contrary has been appended.

Threat Reports are reports created by Smarttech247 based on high and critical severity vulnerabilities that may have a high potential to be exploited in the wild i.e. vulnerabilities that are present in most used products by companies and do not have an auto-update option or they are usually not automatically updated in case that could lead to some service disruption. This report is usually created as soon as the vulnerability is released, therefore we strongly recommend that the information is reviewed, tests are performed and patches are applied before the first proof-of-concept is released.
Even though certain vulnerabilities may not have an active exploit in the wild at the time that we report on them, we take into consideration the wider risk and the impact it could have on systems, should an exploit like that be available after a while. Our duty is to report them on time and we recommend enterprises that, in order to keep critical business systems protected, they should consider, on average, ten working days to check whether or not the new vulnerability affects them, and if so, to implement actions in order to remove the risk.

Overview:

A critical out-of-bounds write in the Google Brotli decoder (CVE-2026-111497, also tracked as Node2Shell, HTTP2Shell and Broccoli2Shell) lets an unauthenticated remote attacker corrupt memory with a crafted Brotli stream. The impact ranges from a process crash and memory disclosure to remote code execution.

RISK:

Government:

- Large and medium government entities: **Critical**
- Small government entities: **High**

Businesses:

- Large and medium business entities: **Critical**
- Small business entities: **High**

SYSTEMS AFFECTED:

Any software that decompresses attacker-controlled Brotli data with Google Brotli 1.0.3 through 1.2.0 is affected. Patching Node.js fixes only the copy of the decoder that Node.js bundles. Bun, Python packages, Envoy, the Ruby gem, distribution libraries and compiled executables each need their own update.

Product	Affected	Fixed version / action
Google Brotli	1.0.3 to 1.0.9, 1.1.0, 1.2.0	1.0.10, 1.1.1, 1.2.1
Node.js	22.x before 22.23.4, 24.x before 24.22.1, 26.x before 26.12.1. EOL lines 20.x, 23.x and 25.x are treated as affected.	22.23.4, 24.22.1, 26.12.1. No patch for EOL lines: migrate. No vendor workaround.
Bun	1.4.3 and earlier	1.4.4. Rebuild executables created with bun build --compile.
Python Brotli, brotlicffi	Brotli 1.2.0 and earlier, brotlicffi 1.2.0.x and earlier	Brotli 1.2.1, brotlicffi 1.2.1.0
Envoy and Istio	Where the Brotli decompressor is configured	Envoy 1.40.0, 1.39.3, 1.38.6, 1.37.8, 1.36.12. Istio 1.31.2, 1.30.6, 1.29.9.
Microsoft .NET	.NET 8, 9, 10 and 11 RC, where Brotli decompression is enabled. .NET Framework is not affected.	8.0.33, 9.0.22, 10.0.14, 11 RC3 (out-of-band release, 25 October 2026)

Product	Affected	Fixed version / action
Palo Alto Networks PAN-OS	PA-Series and VM-Series with Brotli inspection and TLS/SSL decryption enabled on the dataplane (see below)	Security fixes are available for several PAN-OS versions, with additional releases scheduled through 30 October 2026. Apply the relevant fixed version where available; otherwise, implement the recommended workaround.
Prisma Cloud Application Security Transporter Client	Before 0.2.9	0.2.9
Red Hat Enterprise Linux	RHEL 7, 8, 9 and 10 (brotli package). Firefox, Thunderbird, Node.js and JBoss Core Services brotli also listed.	Errata pending. Rebuild RHEL and UBI-based images after release.
Debian	bookworm and trixie	brotli 1.0.9-2+deb12u1 (bookworm), 1.1.0-2+deb13u1 (trixie)
Alpine	Edge, 3.21, 3.23, 3.24	brotli 1.2.1-r0 (1.1.1-r0 on 3.21)
Amazon Linux 2023	brotli package. The nodejs package has no fix planned.	Update brotli. Fix Node.js through the runtime.
Ruby brotli gem	Before 0.9.1	0.9.1 or 0.9.2
Java brotli4j	Before 1.24.0 (per CISA)	1.24.0. We could not confirm this against a vendor advisory.

Other bundled copies (for example PHP extensions, DuckDB, ClickHouse, Arrow, MuPDF and 7-Zip) are reported by a single tracker and are unverified.

Palo Alto Networks PAN-OS

Versions	Affected	Unaffected
PAN-OS 12.2	< 12.2.4-h1	>= 12.2.4-h1
PAN-OS 12.1	< 12.1.4-h13 < 12.1.7-h11 < 12.1.10-h3	>= 12.1.4-h13 (ETA: 10/29) >= 12.1.7-h11 >= 12.1.10-h3 (ETA: 10/30)
PAN-OS 11.2	< 11.2.4-h23 < 11.2.7-h23 < 11.2.10-h17 < 11.2.13-h4 < 11.2.16-h1	>= 11.2.4-h23 (ETA: 10/30) >= 11.2.7-h23 (ETA: 10/29) >= 11.2.10-h17 >= 11.2.13-h4 (ETA: 10/27) >= 11.2.16-h1 (ETA: 10/28)
PAN-OS 11.1	None	All

PAN-OS 10.2	None	All
Prisma Cloud - Application Security Transporter Client	< 0.2.9	>= 0.2.9

Palo Alto Networks is not aware of any malicious exploitation of this issue.

- Exposure requires both Brotli inspection and TLS/SSL decryption on the dataplane. Brotli encoding is disabled by default from PAN-OS 11.2.4 and is unsupported before that.
- Check exposure with: show system setting brotli (exposed if Enabled: yes). On PAN-OS 12.1.2 and later, check Device > Setup > Content-ID > Allow Brotli Encoding.
- Workaround: set system setting brotli-encoding disable or untick Allow Brotli Encoding and commit. Customers with Threat Prevention get partial coverage from Threat ID 97686 (content version 9165 and later). Disabling Brotli inspection reduces visibility into Brotli-encoded traffic until the fix is installed.

TECHNICAL SUMMARY:

CVE-2026-111497 (CWE-787 out-of-bounds write, CWE-252 unchecked return value): Brotli decoder block-switch handling

Brotli streams are split into blocks of three kinds: literals, insert-and-copy commands and distances. When a kind declares only one block type, a block switch is invalid and the decoder must reject it. The vulnerable helper returned the same plain false for that case and for the normal case where the decoder runs out of input. The caller treated the invalid switch as a request for more input and kept appending bytes to an eight-byte internal buffer that never drains, so writes land outside the buffer. According to ELLIO's analysis, an attacker can choose a destination within about 2 GB of the ring buffer and write 16 bytes of their own content there. The upstream fix (commit 0ed4268a) returns a dedicated error, BROTLI_DECODER_ERROR_FORMAT_BLOCK_SWITCH, so the decoder stops on an invalid stream.

Not every system that includes Brotli is automatically vulnerable to remote attack. Exploitation requires the application to decompress attacker-controlled Brotli data, such as inbound request bodies with Content-Encoding: br, responses fetched from untrusted servers, or untrusted compressed files. A system that bundles the library but never passes untrusted data to the decoder has negligible exposure, though it should still be patched.

How untrusted Brotli reaches the decoder

- **Inbound request bodies** sent with Content-Encoding: br. This is the most dangerous path because any unauthenticated client can use it. Express 5.0.0 to 5.3.0 with express.json() or express.raw(), body-parser 2.0.0 to 2.3.0, and Koa with @koa/bodyparser decompress the body before any route handler runs.
- **Outbound responses** decoded automatically by built-in fetch() (undici), axios, got, SuperAgent and server-side fetches in frameworks such as Next.js. This matters wherever an attacker can influence the URL, for example webhooks, link previews, crawlers and agent tools.
- **Direct calls** to node:zlib Brotli functions on client-supplied data.
- **Python:** requests, urllib3, httpx[brotli], aiohttp and Scrapy decode based on the server's Content-Encoding, so installing a vulnerable Brotli package is enough to be exposed.

- **Microsoft .NET:** ASP.NET Core request decompression, HttpClient automatic response decompression and System.IO.Compression Brotli APIs. None is enabled by default.

Impact beyond RCE

- A failed exploit attempt usually crashes the worker, so repeated attempts cause denial of service. A service with no restart supervisor can stay down after the second request.
- Where core dumps are enabled (for example systemd-coredump on a stock Ubuntu host), each crash can write another visitor's session cookies, bearer tokens or key material to disk. Containers usually write no core file.
- Python Brotli and brotlicffi, the PHP streaming API and the Ruby gem can hang on the malicious stream without raising an error. The PHP streaming case spins at 100% CPU. This looks like a stalled job, not a failure.
- Do not disable ASLR as a mitigation. ELLIO measured a shell after about five attempts with ASLR on, and none in 60 attempts with it off, because a fixed layout lets an attacker measure once and reuse it.
- process.versions.brotli reports 1.2.0 on both patched and unpatched Node.js builds, because the fix was cherry-picked without a version bump. Check the Node.js release version instead, or the OS package version where Node.js links the system library.

If you cannot update immediately

Only an update fully remediates this. These steps reduce exposure to specific paths and do not make a system safe.

- Drop or reject Content-Encoding: br on inbound request bodies at the reverse proxy, WAF or load balancer where clients do not need it. This closes the request-body path only.
- Block or allow-list destinations for server-side fetches, and avoid automatic Brotli decoding of responses from untrusted servers.
- Do not call zlib.brotliDecompress() or createBrotliDecompress() on untrusted data.
- On PAN-OS, disable Brotli inspection until the fixed release is installed.
- Restrict who can read core dumps, and shorten retention.

Recommendations:

Identify vulnerable systems and applications

- Review systems running Node.js, Bun, Python Brotli libraries, Microsoft .NET, Envoy, Istio, Palo Alto PAN-OS and other affected software.
- Identify applications that process Brotli-compressed data from external or untrusted sources.
- Check application dependencies, containers and bundled libraries, as updating the operating system alone may not address all vulnerable components.

Apply the latest security updates

- Upgrade affected software to the fixed versions in the relevant vendor advisories. Prioritise internet-facing applications, APIs, reverse proxies and services that process untrusted compressed data, ideally within 24 hours.
- Node.js 20.x, 23.x and 25.x are end of life and will not receive a fix. Plan migration to a supported release.
- Rebuild and redeploy applications containing bundled vulnerable components,

including self-contained executables and container images.
- Verify remediation using the Node.js release version or the OS package version, not process.versions.brotli, which still reports 1.2.0 on patched builds. Confirm with vulnerability scans and software inventories.

Implement temporary mitigations where patching is delayed
- Restrict or disable Brotli decompression of untrusted data until updates can be applied.
- Where operationally feasible, configure reverse proxies, WAFs or load balancers to reject requests with Content-Encoding: br. This closes the request-body path only.
- Restrict server-side applications from fetching and decompressing content from untrusted destinations.
- For affected Palo Alto Networks firewalls, disable Brotli inspection until the fixed PAN-OS release is installed.

Monitor for potential exploitation attempts
- Review web-server and application logs for br-encoded POST requests from unauthenticated clients, especially very large bodies or short bursts of mixed-size bodies.
- Investigate unexpected crashes, memory-related errors, excessive CPU consumption and abnormal service restarts.
- Monitor for Node.js or other application processes spawning shells, or making unexpected outbound connections.
- Correlate WAF, EDR and network alerts to identify potentially malicious activity.

Strengthen application and infrastructure security
- Apply least-privilege permissions to application services to limit the impact of exploitation.
- Segment internet-facing applications from critical internal systems.
- Restrict access to application logs, crash dumps and core files, which may contain credentials or session data.
- Do not run circulating PoC code outside an isolated lab. Some public exploit repositories may be malware aimed at defenders.

References:

- https://nodejs.org/en/blog/vulnerability/october-2026-security-releases
- https://github.com/google/brotli/releases/tag/v1.2.1
- https://security.paloaltonetworks.com/CVE-2026-111497
- https://devblogs.microsoft.com/dotnet/dotnet-october-2026-oob-security-updates/
- https://access.redhat.com/security/vulnerabilities/RHSB-2026-015
- https://ellio.tech/en/blog/node2shell-cve-2026-111497-rce-on-stock-nodejs-builds/
- https://www.cyberkendra.com/2026/10/node2shell-cve-2026-111497-exploited-in-the-wild.html
- https://www.rapid7.com/blog/post/etr-cve-2026-111497-critical-out-of-bounds-write-in-the-node-js-brotli-decoder/
- https://www.wiz.io/blog/cve-2026-111497-brotli-rce-node-js
- https://www.sysdig.com/blog/node2shell-cve-2026-111497-detecting-the-node-js-brotli-rce
- https://www.aikido.dev/blog/python-packages-vulnerable-node2shell-brotli-cve-2026-111497
- https://broccoli2shell.com/
- https://www.cve.org/CVERecord?id=CVE-2026-111497

CVE: CVE-2026-111497

Smarttech
YOUR 24/7 SECURITY PARTNER

www.smarttech247.com