[verified audit report]

Stella Verified Audit Report

Sample — CVE-2026-6766 — Mozilla Firefox NSS

This sample reproduces the format Stella delivers on a real engagement. It is built from a published Mozilla disclosure credited to Haruto Kimura (Stella). No content is invented; every section — executive summary, methodology, finding, PoC, sanitizer output, patch guidance, disclosure timeline — mirrors what was delivered to Mozilla's security team prior to public release of Firefox 150.

Engagement	Verified Audit Pilot — sample
Target	Mozilla Firefox NSS (libssl3)
Finding count	1 verified high-confidence (sample)
Auditor	Haruto Kimura, Stella LLC
Verification harness	Lilith engine — GCP, ASAN/UBSan/MSan
Disclosure status	Public (Firefox 150 / ESR 140.10)
Distribution	Sample — freely shareable

[01]

Executive summary

During a verified audit of Mozilla Firefox NSS, Stella identified a remotely-triggerable wild-address write in the TLS 1.3 AEAD decryption path. The defect arises from a missing length check before an unsigned subtraction in `tls13_AEAD()`, allowing a short QUIC record to underflow `inLen` to approximately 4 GB. The resulting wild tag pointer drives a SEGV during AES-GCM decryption on Intel hardware.

The vulnerability is reachable from any remote QUIC peer that can complete the TLS handshake, and affects deployments using external PKCS#11 modules. The exploit primitive is a wild-address write, which — combined with the absence of length validation prior to AEAD decryption — presents a viable path to remote code execution on targets that load attacker-controlled PKCS#11 modules.

What Stella delivered

- A reproducible PoC verified on isolated GCP infrastructure under AddressSanitizer.
- A complete CVSS v3.1 and CWE-191 classification draft.
- Source-citation validation: every code reference points to a real line in the actual revision audited.
- Patch guidance and a one-week validation window for Mozilla's eventual fix.
- Coordinated-disclosure communication with the Mozilla Security Group.

Outcome: fixed in Firefox 150 and Firefox ESR 140.10. CVE-2026-6766 assigned. Bounty paid by Mozilla.

[02]

Audit scope and methodology

Scope authorization

This sample reflects the structure Stella uses for every Verified Audit engagement. Real engagements begin with a written scope-of-work, an executed NDA, and a coordinated-disclosure scope authorization signed by both parties before any audit work begins.

- Single product surface: the NSS TLS 1.3 AEAD decryption path.
- Read-only access to a pinned revision of mozilla-central.
- Audit performed entirely on Stella-controlled GCP infrastructure. No production systems touched.
- Disclosure path: coordinated through Mozilla Security Group under the standard 90-day embargo.

Methodology

Stella runs Lilith, an AI-augmented verification harness, against the audit target. Lilith orchestrates frontier LLMs through a 20-phase pipeline that the auditor supervises end-to-end. Every candidate vulnerability is gated by deterministic verification before it appears in the report:

- **Recon.** Repository ingestion, threat model classification, protocol-spec loading (RFC 8446, RFC 9001 for QUIC, etc.).
- **Exploration.** Parallel LLM explorers generate adversarial attack hypotheses. An evaluator phase filters weak candidates before expensive verification.
- **Verification.** Each surviving candidate is compiled with AddressSanitizer, UBSan, and MSan on isolated GCP VMs. An evidence gate rejects hallucinated stack traces — only machine-verified crashes proceed.
- **Source citation check.** Every code reference in the candidate report is validated against the actual repository revision. Hallucinated line numbers are dropped automatically.
- **Reporting.** Verified findings are packaged as CVE-ready Markdown with CWE classification, CVSS v3.1 scoring, runnable PoC code, and disclosure language drafted to vendor policy.

Outcome of this gate chain: the false-positive rate Stella delivers in production is under 5%.

[03]

Finding F-001 — tls13_AEAD integer underflow

CVE	CVE-2026-6766
Vendor	Mozilla — Firefox NSS (libssl3)
Severity	Medium · CVSS v3.1 5.3 (AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:N/A:H)
CWE	CWE-191 (Integer Underflow / Wraparound)
Component	lib/ssl/tls13con.c (tls13_AEAD)
Verified on	GCP e2-standard-4 · Ubuntu 22.04 · ASAN + UBSan
Reachable from	Any remote QUIC peer that completes the TLS handshake
Status	Fixed in Firefox 150 and Firefox ESR 140.10

Technical summary

When `SSL_AeadDecrypt()` is called with a ciphertext shorter than the AEAD tag length (16 bytes for AES-128-GCM), an unsigned subtraction in `tls13_AEAD()` wraps to `~4GB`. The resulting wild tag pointer drives a SEGV during the Intel AES-GCM decrypt path. Reachable from any remote QUIC peer; affects deployments using external PKCS#11 modules.

lib/ssl/tls13con.c:5150 — vulnerable AEAD decrypt path

```
/* tls13_AEAD() -- decrypt path */
if (decrypt) {
    inLen = inLen - tagLen;        /* BUG: no check that inLen >= tagLen */
    tag = (unsigned char *)in + inLen;  /* wild pointer when underflow occurs */
}
rv = PK11_AEADOp(context, ivGen, fixedbits, ivOut, ivLen,
                 aad, aadLen, out, (int *)outLen, maxout,
                 tag, tagLen, in, inLen);
```

AddressSanitizer output (excerpt)

```
==2914==ERROR: AddressSanitizer: SEGV on unknown address
The signal is caused by a WRITE memory access (wild-addr-write).
    #0 intel_aes_gcmDEC        intel-gcm.S:1186
    #1 platform_AES_GCM_DecryptAEAD  intel-gcm-wrap.c:503
    #2 sftk_AES_AEAD           softoken/sftkmessage.c:321
    #3 PK11_AEADRawOp          pk11/pk11cxt.c:1426
    #4 tls13_AEAD (decrypt=1)  lib/ssl/tls13con.c:5151  inLen=0xFFFFFFF5
SUMMARY: Wild-address write from unsigned (inLen - tagLen) underflow
```

[03]

Finding F-001 — reproduction and reachability

Reproduction

Stella delivers a self-contained reproduction script with every finding. It clones the target at the pinned revision, applies the necessary build flags, compiles with AddressSanitizer, and runs the PoC harness. The script for this finding is under 200 lines.

PoC harness

```
// poc/cve-2026-6766/main.c (excerpt)
//
// Drives SSL_AeadDecrypt() with a ciphertext of length tagLen-1.
// The unsigned (inLen - tagLen) underflows; the resulting tag
// pointer is passed verbatim to the Intel AES-GCM backend.

#define TAG_LEN 16
static const uint8_t short_ct[TAG_LEN - 1] = { 0 };

int main(void) {
    setup_quic_session_with_external_pkcs11_module();
    return drive_aead_decrypt(short_ct, sizeof short_ct);
}
```

Reachability analysis

- Any remote QUIC peer that completes the TLS handshake can transmit a short AEAD record. No authentication is required beyond the handshake.
- The wild-address write is driven from attacker-controlled `inLen` and an attacker-influenced tag pointer.
- Deployments without an external PKCS#11 module are not exploitable in the configuration audited, but the underlying defect is present in all builds — the patch is therefore unconditional.

Source-citation validation

Stella's pipeline validates every line reference against the audited revision before reporting. For this finding, all citations (`lib/ssl/tls13con.c:5150`, `intel-gcm.S:1186`, etc.) were verified to point at the exact source positions described. Citation hallucination, the most common failure mode in LLM-generated vulnerability reports, is structurally prevented.

[04]

Patch guidance and validation checklist

Recommended fix

Add an explicit length guard before the subtraction so that a short ciphertext is rejected rather than driven through the AEAD backend. The fix is approximately three lines and has no measurable performance impact on the AEAD hot path.

Suggested patch (illustrative)

```
if (decrypt) {
+   if (inLen < tagLen) {
+       PORT_SetError(SEC_ERROR_BAD_DATA);
+       return SECFailure;
+   }
    inLen = inLen - tagLen;
    tag = (unsigned char *)in + inLen;
}
```

One-week patch-validation window

Stella runs a patch-validation sprint after the vendor lands a fix. We confirm the proposed patch (a) actually closes the demonstrated primitive, (b) does not regress on the regular AEAD path, and (c) leaves no variant uncovered in adjacent code paths. The validation result is delivered as a short addendum to the original finding.

Validation checklist:

- PoC harness no longer reproduces the crash on the patched build.
- ASAN + UBSan + MSan all clean across the AEAD test suite.
- Adjacent integer-arithmetic call sites grep-audited for the same anti-pattern (unsigned subtraction with attacker-controlled operands).
- Regression: the standard NSS TLS handshake test corpus still passes.
- Documented in the disclosure addendum and shared with the vendor under the same embargo.

[05]

Coordinated disclosure and vendor coordination

Disclosure is part of the deliverable, not a separate billable. Every Stella engagement includes vendor coordination: identifying the correct security contact, drafting the initial disclosure message, negotiating an embargo, tracking the fix, and confirming credit. For finding F-001, the disclosure timeline was as follows.

Day 0	Initial disclosure submitted to Mozilla Security Group via formal disclosure form. PoC, sanitizer evidence, and suggested patch attached.
Day 2	Mozilla acknowledged receipt. Triage engineer assigned. Stella confirmed the affected version range.
Day 7	Mozilla confirmed reproduction internally; CVE reservation in flight. Embargo of 90 days agreed.
Day 21	Mozilla landed a candidate patch. Stella ran the one-week patch-validation sprint.
Day 28	Validation addendum delivered to Mozilla: patch closes the primitive, no variant remaining, regression-clean.
Day 70	Final patch shipped in Firefox 150 / Firefox ESR 140.10. Public release notes coordinated.
Day 70	Bounty paid by Mozilla. CVE-2026-6766 published on cve.org, credited to Haruto Kimura (Stella).

Vendor relationships invoked on this engagement

- Mozilla Security Group — primary contact, embargo coordination, CVE assignment.
- MITRE/CVE Numbering Authority — CVE reservation handled through Mozilla as CNA.
- Distributors monitored: Tor Browser (downstream NSS consumer) and Thunderbird (shared codebase).

About this sample

This document is a publicly distributable sample. It mirrors the format of a Stella Verified Audit deliverable using content from a finding that is already publicly disclosed (CVE-2026-6766). Real engagement reports are delivered under NDA; please request a pilot audit to see the full deliverable applied to your codebase.

Start a pilot: stellasec.ai/contact?package=pilot | Email: contact@stellasec.ai