# Bug 2442232 (CVE-2026-3099) - CVE-2026-3099 libsoup: Libsoup: Authentication bypass via digest authentication replay attack

**Keywords:** Security

**Status:** NEW

**Alias:** CVE-2026-3099

**Product:** Security Response

**Component:** vulnerability

**Version:** unspecified

**Hardware:** All

**OS:** Linux

**Priority:** medium

**Severity:** medium

**Target Milestone:** ---

**Assignee:** Product Security DevOps Team

**QA Contact:**

**Docs Contact:**

**URL:**

**Whiteboard:**

**Depends On:** 2442233 2442234 2442235

**Blocks:**

**TreeView+** depends on / blocked

**Reported:** 2026-02-24 07:35 UTC by OSIDB Bzimport

**Modified:** 2026-04-27 07:12 UTC (History)

**CC List:** 2 users (show)

**Fixed In Version:**

**Clone Of:**

**Environment:**

**Last Closed:**

**Embargoed:**

---

**Attachments** (Terms of Use)

OSIDB Bzimport 2026-02-24 07:35:07 UTC Description

```
DESCRIPTION

Here, I had found another flaw in Libsoup's server-side digest
authentication implementation. The SoupAuthDomainDigest class
fails to track issued nonces or enforce the incrementing
nonce-count (nc) attribute required by the Digest Auth
standard.
The validation logic in check_hex_urp performs a purely
mathematical verification. It recomputes the expected hash
without verifying if the nonce was actually generated by the
server or if the nc value is being reused. This allows an
attacker who captures a single valid Authorization header to
replay it indefinitely, bypassing authentication and accessing
```

protected resources as the victim.

VULNERABILITY DETAILS

The vulnerability exists in libsoup/server/soup-auth-domain-digest.c due to a "stateless" design choice that violates security standard.
-Stateless Validation: The function check_hex_urp validates the response hash but fails to check against a store of active, issued nonces.
-Missing Nonce-Count Check (The Main Flaw): RFC 7616 explicitly states that the server MUST verify that the nc (nonce-count) value increases for each request using the same nonce.
In soup-auth-domain-digest.c (around lines 265-), the code parses the nonce count:

```
nonce_count = strtoul (nc, NULL, 16);
if (nonce_count <= 0)
return FALSE;
```

Here, it do checks that nonce_count is positive, but it never compares it to a previously seen value for that nonce. It simply accepts any positive integer, allowing attackers to reuse nc=00000001 infinitely.
-Insecure Nonce Generation: Nonces are generated using time(0) and the message pointer address. They are not cryptographically signed (HMAC) and are never expired by the server, creating an infinite window for replay attacks.

Note

You need to log in before you can comment on or make changes to this bug.