0-DAY NO CVE HIGH PUBLIC

OpenLiteSpeed HTTP/3 skips Host-vs-SNI mTLS bind

OpenLiteSpeed 1.9.3 still binds TCP Host-vs-SNI mTLS after 1.8.3.1. HTTP/3 does not. QuicStream inherits HioCrypto::verifyContext that always returns true. Gadget labbed. No live OLS process. No CVE yet.

NAME OpenLiteSpeed HTTP/3 skips Host-vs-SNI mTLS bind	TYPE 0-day analysis
CVE n/a	CVE RISK high
DISCLOSURE STATUS public	VENDOR OpenLiteSpeed (LiteSpeed Technologies)
AFFECTED OpenLiteSpeed 1.9.3 (edbe7f7) HTTP/3 on a shared HTTPS listener with clientVerify Require on one name. TCP still 403s. clientVerify defaults None. HTTP/3 defaults on.	PUBLISHED 12 Oct 2026
UPDATED 12 Oct 2026	TAGS 0-day, openlitespeed, http3, mtls, privilege-leftover, cwe-287, cwe-295, unauthenticated

TL;DR

Lab files and proof-of-concept **here**.

01 TCP got the pointer compare. HTTP/3 got `return true` .

I am @abraxas_null. The proof of concept is on GitHub: **abraxas/openlitespeed-h3-mtls-verifycontext** (loopback client). The lab stack is `lab/` : Dockerfile, docker-compose.yml, run.sh, poc.py, gadget.cpp. Authorized lab only. It talks to loopback.

This is **OpenLiteSpeed v1.9.3** (`edbe7f7db12577e905291955a6d416200942fb8b`). No CVE yet. CWE-287 / CWE-295. 7.4 High (`CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N`). Remote privilege leftover of the 1.8.3.1 Host-vs-SNI mTLS bind. HTTP/3. Not a live OpenLiteSpeed process. Last line `how=gadget` .

1.8.3.1 closed a USENIX Security 2025 class of bug: handshake SNI picks one `SSL_CTX` , Host picks another vhost, and Client Verification Require on the Host vhost was not bound to the handshake. After `matchVHost()` , OLS now asks the session crypto whether this is that vhost's context. `SslConnection` answers with pointer equality. `QuicStream` does not override the question. The base `HioCrypto::verifyContext` always returns true. HTTP/3 enablement defaults on. `clientVerify` itself defaults to None.

Same leftover class as 1.8.3.1. Different transport. The TCP path still 403s.

02 After Caddy and lighttpd, the next HTTP server

I had just finished unpublished Crit/High hunts of Caddy v2.11.7 and lighttpd 1.4.85. Caddy came back empty at this bar. Named GHSAs on that pin already held: Windows `%5c` , `copy_headers` , FastCGI Unicode and case-fold `split_path` , mTLS fail-open, the CVE-2026-30852 bypass. Default admin is still localhost. Caddy's missing-CA fail-open is not this bug. `configOneCert` errors if CA load fails. lighttpd

on Unix was the same kind of empty. The leftover that survived there is native `_WIN32` staticfile `..%20/`. Unix is not that class.

The next HTTP server was OpenLiteSpeed. C++ origin, lsphp in the same install, HTTP/2 and HTTP/3 in the same process, a history of request-smuggling and cookie-role patches, and a typical production stack that is WordPress plus LiteSpeed Cache. That is a different animal from Caddy's Go listeners and lighttpd's small Unix static server.

I only wanted Critical or High. Local is in-scope. Medium stays on the floor unless it chains. A finding on this hunt needed a leftover of a named 1.8.x / 1.9.x guard, unauth mutating admin, CGI/lsphp RCE, path traversal off the docroot, or a privilege control that still 403s the request the docs told operators to block and then forwards the blocked bytes anyway.

Two leftovers survived. G2 is HTTP/1 duplicate Cookie skipping the 1.8.2.1 `litespeed_role=` 403. That is the typical production stack: lsphp, official Hub images, CyberPanel, WordPress-on-OLS. This post is G1. Mixed-mTLS-on-shared-443 is a minority layout. That is why 1.8.3.1 existed. It is also why AC:H sits on the vector.

03 Pinning v1.9.3

Latest stable. Not an RC. Release log dates 1.9.3 to 2026-09-29. GitHub's tag object showed up a day later. HEAD landed on `edbe7f7db12577e905291955a6d416200942fb8b`. `dist/VERSION` says `1.9.3`. That is the pin.

1.9.3 is the header/URL pass, the CGI hardening, the update-script SHA-256, lsquic 4.10.0. If a leftover of 1.8.3.1 still fires here, it is not a stale minor.

04 The skip list is half the hunt

Before you look for leftovers, you write down what is already public so you do not "discover" a CVE with a press release. On 1.9.3 the interesting ones were already closed:

- CVE-2026-104474 `lsup.sh` unverified `autoupdate/` as root. 1.9.3 stages into `admin/update` mode 0700 and `verify_package_sha256` before `./install.sh`.
- CVE-2026-31386 WebAdmin command injection as admin through 1.9.0. Ext App `path` now goes through `check_cmd_invalid_str` then `SUExec::buildArgv` / `execve`, not `/bin/sh -c`.
- 1.9.0 HTTP/2 request smuggling, HTTP/3 engine remote exploits, `UnsafeAllow3F`. Request-line unpack rejects. Vendored lsquic is 4.10.0. Rewrite capture `?` re-encodes to `%3F` unless the operator opts in.
- 1.9.1 / 1.9.2 general code hardening and H2/H3 protocol and header validation.
- 1.8.3.1 TE then CL, and TCP mTLS Host-vs-SNI. Both orders of Transfer-Encoding plus Content-Length are 400. TCP `SslConnection::verifyContext` is pointer equality on the handshake CTX. That TCP bind is the control this leftover walks around, not the leftover.
- 1.8.2.1 block `litespeed_role` cookie. Single Cookie is still 403. That is G2.
- Older: CVE-2024-31617 chunked before 1.8.1, CVE-2023-40518 header validation before 1.7.18, CVE-2018-19791 Range amplification, the 2022 search-path LPE / RCE chain, WebAdmin URL and External App LPEs.

CVE-2026-49975 HTTP/2 Bomb: vendor says OLS is not vulnerable. LiteSpeed Cache WordPress CVEs are a different tree. I am not re-filing the plugin. I am not re-filing the 1.8.3.1 TCP Host bypass.

05 Twenty surfaces, default Example empty

I split the tree. Request parser. CGI / lsphp / FastCGI. Proxy. WebSocket. HTTP/2. HTTP/3 / lsquic. Rewrite `%3f`. WebAdmin. `lsup.sh`. Chunked leftover of CVE-2024-31617. Staticfile / context. Auth / htpasswd. SSI. ACME / TLS leftover of 1.8.3.1. Namespace. Internal redirect. Node.js. Cache / recaptcha / Lua. Range leftover of CVE-2018-19791. 1.8.x / 1.9.x patch siblings.

Most of that is OpenLiteSpeed doing what 1.9.3 said it would do.

TE other than exactly `chunked` is 501. Duplicate TE, CL, Host, Authorization are 400. `Proxy` is 400 (HTTPoxy). `() {` in a value is 400 (shellshock). URI decode then `GPath::clean` fail-closes `/../`. SCRIPT_FILENAME is the real file from the stat walk. Range parse saturates at INT64_MAX. CGI uid-0 env strip plus lscgid v2 keyed MAC. CONTINUATION exclusive, 64 KiB cap. Rapid reset trips HTTP/2 off. WebAdmin mutating PHP still wants `CAuthorizer::Authorize()` plus CSRF.

ACME HTTP-01 does not write challenge files. `mod_acme` is in-memory RFC 8555 stateless echo. Built-in ACME runs `acme.sh --stateless`. Token alphabet is base64url, no `/`, no `.`. Unknown SNI on TCP keeps the listener cert (`sslStrictSni` default 0) and the TCP mTLS Host mismatch still 403s. Missing CA at config time fail-closes: `configOneCert` errors if `clientVerify` is set and `setCALocation` fails.

Stock `dist/conf/httpd_config.conf.in` listens on `*:%HTTP_PORT%` (install default 8088), maps Example, auto-starts `lsapi:lsphp` for suffix `php`, `quicEnable 1`, cache check-only. Example vhost is static HTML plus `/cgi-bin`. `clientVerify` default is None. WebAdmin binds `*:7080` with password auth. Stock Example listener is cleartext `secure 0`.

Default Example unpublished Crit/High: **NONE**. Named 1.9.x guards hold. I am not filing "port 8088 serves index.html." I am not filing "QUIC is on." I am filing the missing override on the crypto object HTTP/3 actually uses.

06 The bind that 1.8.3.1 added

After `matchVHost()`, if the Host-selected vhost wants client verify, the session crypto must prove it is that vhost's `SSL_CTX`:

```cpp
if (pVHost->isSslClientAuth() && getCrypto())
{
    if (getCrypto()->verifyContext(pVHost->getSslContext()) == false)
    {
        LS_DBG_L(getLogSession(), "VHost [%s] require SSL client authentication,
                 pVHost->getName());
        return SC_403;
    }
}
```

`httpsession.cpp:1655-1662`. `HttpVHost::isSslClientAuth` is `m_pSSLCtx ? m_pSSLCtx->getVerifyMode() : 0`. Mode 2 is `SSL_VERIFY_PEER | SSL_VERIFY_FAIL_IF_NO_PEER_CERT | SSL_VERIFY_CLIENT_ONCE`.

TCP answers with a pointer:

```cpp
return SSL_get_SSL_CTX(m_ssl) == ctx->get();
```

`sslconnection.cpp:673-675`. Handshake SNI to vhost A (no mTLS) then Host of vhost B (mTLS required) is 403 on HTTP/1.1 and HTTP/2 over TCP.

HTTP/3 does not have that override. `QuicStream` is `HioStream` plus `HioCrypto` and only overrides `getEnv`. Base implementation:

```cpp
```

```
virtual bool verifyContext(SslContext *ctx)
{   return true;    }
```

`hiocrypto.h:49-50`. `QuicEngine::onNewStream` sets `pConnInfo→m_pCrypto = pStream`. `HttpSession::onInitConnected` copies that pointer. The 1.8.3.1 `if` therefore runs, calls `verifyContext`, and never 403s. `CLIENT_CERT` on QUIC is hardcoded 0. Handshake still uses the SNI-selected `SSL_CTX`, so SNI of the mTLS vhost can still demand a client cert at TLS time. SNI of a sibling name without client verify does not. Host / `:authority` then selects the mTLS vhost with the always-true bind.

QUIC is on unless turned off: `QUIC_ENABLED_BY_DEFAULT 1`, listener `enableQuic` default 1, vhost `vhssl` `enableQuic` default 1. `clientVerify` itself defaults 0. Same non-default mTLS requirement as 1.8.3.1. Same default-on HTTP/3 that 1.8.3.1 did not cover.

07 Why a gadget, not a live OLS process

I did not start `openlitespeed`. A live HTTP/3 mTLS lab wants a shared HTTPS listener, two names, a CA, a client cert on one path and not the other, and a QUIC stack that actually completes a handshake against the pin. That is a real lab. It is also a different lab from proving the pin functions do what the source says.

The leftover is the missing override. The gadget is those functions: `HioCrypto::verifyContext` always true, `QuicStream` has no override, `quicengine` installs the stream as crypto, `SslConnection` pointer equality, `HttpSession` 403 only when `verifyContext` is false. `gadget.cpp` is a faithful copy. The oracle speaks HTTP on `127.0.0.1:19200`. Pin snippets live in `lab/pin-src/`. The last line still reads SUCCESS. `how=gadget` is in that line so nobody can pretend a live OLS binary answered.

TCP mismatch in the gadget is 403. H3 mismatch is 200 plus `OLS-H3-MTLS-VERIFYCONTEXT-WITNESS`. Matching SNI on TCP is 200, not leftover. Public vhost on

HTTP/3 with matching SNI is 200, not leftover. That is the whole argument.

08 Wrong turns, in the order I made them

Treating Caddy mTLS fail-open as this bug. Caddy GHSA missing-CA is a different class. OLS `configOneCert` errors if CA load fails. The leftover is Host-vs-SNI after a successful handshake CTX, not a missing CA.

Re-filing 1.8.3.1. TCP still 403s. The skip list item is `SslConnection::verifyContext`. The leftover is `QuicStream` inheriting `return true`.

Treating default Example as in-scope. Example `clientVerify` is None. Stock listener is cleartext. This leftover needs Require on one name of a shared SSL listener. Default Example unpublished Crit/High is still NONE.

Waiting on a live OLS HTTP/3 stack before the pin functions could be shown. The leftover is the missing virtual. The gadget is those functions. Claiming a live OLS SUCCESS would be a lie. `how=gadget` is the honest last line.

Treating `quicEnable` default 1 as the bug. HTTP/3 on is intended. The bug is the bind that 1.8.3.1 added on TCP and did not add on the crypto object HTTP/3 actually installs.

Scoring it below High because mixed-mTLS-on-shared-443 is uncommon. AC:H is already in the vector. 1.8.3.1 existed because that layout is real. A leftover of a High control is still High, then you say what the lab actually was.

A reverse shell. Theatre. The oracle is a witness string in a gadget JSON body on loopback.

Calling G1 the typical leftover. lsphp plus LSCWP is what people actually run. That is G2. This post is the other one.

09 What an attacker can do

On a shared HTTPS listener with HTTP/3 left at default on, and Client Verification Require set on one vhost while another name on that IP does not require a client cert:

Complete a QUIC handshake using SNI (or no SNI) for a CTX that does not have `SSL_VERIFY_FAIL_IF_NO_PEER_CERT`. Send HTTP/3 with `:authority` of the mTLS vhost. `matchVHost` selects that vhost, `isSslClientAuth()` is true, `QuicStream` `verifyContext` returns true, request is served without a client certificate.

PRIVILEGE	METHOD /
Serve the mTLS vhost on HTTP/3 without a client cert (integrity of the mTLS bind)	SNI publ `mtls.ex` `QuicStr`
Same Host/SNI mismatch on TCP/TLS	`SslConn` pointer e
Public vhost on HTTP/3 with matching SNI	no client
Default Example vhost with `clientVerify` None	no mTLS
WebAdmin / OS RCE / file read	this lefto

Closed on this leftover: default Example HTML vhost, WebAdmin, unauthenticated OS RCE, and claiming a live OLS SUCCESS. Mixed-mTLS-on-shared-443 is a minority layout. CyberPanel and WordPress-on-OLS almost never require client certs. `quicEnable` defaults on. `clientVerify` defaults off.

I am not going to print a curl you can paste at someone else's `:443`. The missing override is the useful part. If you own the origin, run the proof of concept against loopback.

10 Lab

PLAIN TEXT Copy

```
cd lab && ./run.sh
```

Or from a fresh clone:

PLAIN TEXT Copy

```
git clone https://github.com/abraxas/openlitespeed-h3-mtls-verifycontext
cd openlitespeed-h3-mtls-verifycontext/lab
./run.sh
```

Loopback gadget `http://127.0.0.1:19200`. Compose project `openlitespeed-h3-mtls-verifycontext`. Pin `v1.9.3`. Witness `OLS-H3-MTLS-VERIFYCONTEXT-WITNESS`. Last line:

PLAIN TEXT Copy

```
SUCCESS openlitespeed-h3-mtls-verifycontext tcp-mismatch=403 h3-mismatch=200 pin-
```

Bind it to loopback. The loopback client is `lab/run.sh` with a banner on it.

11 The fix

Override `verifyContext` on the QUIC crypto object so Host-selected `clientVerify` Require still binds to the handshake CTX and a verified peer cert. Fail closed if the stream cannot prove that. Do not leave the `HioCrypto` base `return true` on the HTTP/3 path.

Until then, the 1.8.3.1 sentence is only true for TCP. HTTP/3 clients that complete a handshake against a CTX without client verify can still name the mTLS vhost in `:authority`. Turn HTTP/3 off on listeners that share names with a Require vhost. Or do not share those names.

Same pin leftover: openlitespeed-dup-cookie-litespeed-role.

12 References

- Proof of concept: abraxas/openlitespeed-h3-mtls-verifycontext · openlitespeed-h3-mtls-verifycontext-Abraxas-Labs.py
- Lab: `lab/` · Dockerfile · docker-compose.yml · run.sh · poc.py · gadget.cpp
- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-287 · CWE-295 · 7.4 High
- Tree: litespeedtech/openlitespeed v1.9.3 (`edbe7f7db12577e905291955a6d416200942fb8b`) · `hiocrypto.h` always true · `SslConnection` pointer equality · `QuicStream` no override · `quicengine` install · `HttpSession` 403 bind

- 1.8.3.1 TCP Host-vs-SNI bind · same pin: openlitespeed-dup-cookie-litespeed-role · OLS 1.9.x release log

Abraxas Labs

Independent security research by abraxas.

Analysis of 0-days, n-days, malware, and the

leftovers that still matter.

GitHub **@abraxas_null** **abraxas.null@proton.me** **abraxaslabs.tech**

Independent research. Private analysis by arrangement. No ads. abraxas.null@proton.me · For defensive awareness and authorized testing only.

© 2026 Abraxas Labs