Individual Submission
Internet-Draft
Intended status: Experimental
Expires: 22 May 2026

Y. Filimonov
Independent

18 November 2025

Outer-Inner TLS (OI-TLS)
draft-filimonov-oitls-00

Abstract

Outer-Inner TLS (OI-TLS) hides TLS ClientHello metadata (SNI, ALPN, cipher list, JA3) from on-path DPI by splitting a TLS session into two layers. The client first establishes an outer TLS 1.3 tunnel to an entry node with no SNI, then tunnels an ordinary TLS handshake for the backend inside that encrypted channel. This document describes the architecture, signaling, deployment considerations, and security trade-offs, together with laboratory evidence demonstrating the technique.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 22 May 2026.

Copyright Notice

Table of Contents

1. Introduction

DPI appliances routinely block HTTPS connections based on TLS ClientHello contents such as SNI, ALPN, cipher lists, or JA3 fingerprints ([RFC8446]). OI-TLS inserts an entry node between the client and backend. On-path observers see only a normal TLS 1.3 connection to an IP address, whereas the true ClientHello is tunneled inside the outer channel.

2. Motivation and Relationship to Existing Work

Encrypted Client Hello (ECH) [I-D.ietf-tls-esni] protects SNI within a single TLS handshake but requires clients, DNS, and origin servers to support ECH and to deploy DNSHPKE keys. OI-TLS targets operators that can deploy a terminating entry proxy in front of unmodified backends; only the entry node needs changes. OI-TLS also hides the entire ClientHello (cipher list, JA3, ALPN), not only SNI. Unlike domain-fronting techniques, OI-TLS does not rely on CDN misconfiguration; entry nodes are explicit infrastructure.

3. Terminology

* OuterTLS TLS 1.3 session between client and entry node, no SNI, certificate bound to entry IP.

* InnerTLS TLS 1.3 (or 1.2) session between client and backend, carried as application data inside OuterTLS.

* Entry Node Frontend proxy that terminates OuterTLS, parses the inner ClientHello, selects a backend, and relays InnerTLS records.

* Backend Origin server that terminates the InnerTLS connection.

4. Architecture

Client --OuterTLS--> Entry Node --InnerTLS--> Backend.

1) DNS discovery: client queries ‘_oitls.example.com TXT "v=1 ttl_outer=10"‘ or an HTTPS RR ‘example.com HTTPS ... oi-tls=1‘ to learn that OI-TLS is supported. 2) OuterTLS: client connects to the entry IP (A/AAAA answer) with TLS 1.3, no SNI, ordinary extensions. 3) InnerTLS: client generates a standard ClientHello with the real SNI and transmits it as encrypted application data. 4) Session lifetime: outer tunnel normally stays up for the session, but entry nodes MUST enforce an advertised TTL and drop tunnels where the inner handshake never starts. After the inner handshake completes, entry nodes MAY close the outer tunnel.

5. DNS Signaling

Clients SHOULD retrieve ‘_oitls‘ TXT/HTTPS records using DoH/DoT ([RFC8484], [RFC7858]) to hide the signal from DPI; UDP queries still work but leak metadata. The TXT record may carry parameters such as outer TTL or ALPN hints. The entry IP is taken from the original A/AAAA answer. If no OI-TLS record exists, the client falls back to plain TLS.

6. Deployment Considerations

* Entry Integration: OI-TLS can run within HAProxy/nginx modules or standalone proxies. The entry node inspects only the first inner record.

* Backend Requirements: none—backends see an ordinary TLS session and can host HTTP/1.1, HTTP/2, WebSocket, etc.

* Certificate Validation: OuterTLS MUST use trusted certificates (e.g., ACME for entry IPs). DNS discovery SHOULD use trusted DoH/DoT.

* OuterTLS TTL: entry nodes MUST enforce an advertised TTL (e.g., 510s) and drop tunnels with no inner ClientHello. After the inner handshake, outer tunnels MAY be closed.

* Rate Limiting: operators SHOULD cap concurrent OuterTLS sessions and inner handshakes to mitigate CPU-based DDoS.

7. Implementation Status

Reference code and Docker labs are available at https://github.com/EvrkMs/OI-TLS. The labs include a baseline HTTPS flow where DPI observes SNI and an OI-TLS flow where DPI's SNI extractor fails. These labs demonstrate the logic and are not production SDKs; real deployments require separate client/entry implementations.

8. Security Considerations

OI-TLS hides ClientHello metadata but does not defend against IP blocking, volumetric DDoS, or backend compromise. Entry nodes become trusted points of failure; if compromised they reveal all SNI values. Proper certificate validation and DoH/DoT are required to avoid MITM. Entry nodes must be protected via Anycast, load balancing, and rate limiting.

9. IANA Considerations

This document makes no requests of IANA.

10. References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC7858] Hu, Z., "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[I-D.ietf-tls-esni] Rescorla, E., "TLS Encrypted Client Hello", 2024, <https://datatracker.ietf.org/doc/draft-ietf-tls-esni/>.

Author's Address

Filimonov Yaroslav Aleksandrovich
Independent
Email: evrk.msn@gmail.com