SEAT
Internet-Draft
Intended status: Informational
Expires: 17 September 2026

J. Zhang
H. Labiod
Huawei Technologies France S.A.S.U.

16 March 2026

Balancing Security and Deployability in the Selection of Attested TLS Protocol
draft-zhang-seat-selection-00

Abstract

This document analyzes the selection of Attested Transport Layer Security (aTLS) protocols, among pre-handshake, intra-handshake, and post-handshake aTLS protocols, focusing on the trade-off between theoretical security strength and practical deployability. The goal is to enable flexible, context-aware deployment of endpoint attestation while maintaining compatibility with existing infrastructure.

Table of Contents

1. Introduction 2
2. Terminology 3
3. The Theoretical Upper Bound of Security 3
4. The Practical Lower Bound of Deployment 4
5. Selection Analysis 4
5.1. The "Binding" Gap 4
5.2. The "Middlebox" Gap 5
5.3. The "Latency" Gap 5
6. Evaluation of Attestation Phases 5
6.1. Pre-Handshake Attestation 5
6.2. Intra-Handshake Attestation 6
6.3. Post-Handshake Attestation 6
7. Context-Dependent Selection 7
7.1. General Internet Deployment 7
7.2. Controlled Environments and Integrated Protocols 7
7.3. Protocols provide easy attestation integration 7
7.4. Niche Case: Pre-Handshake for Network Admission Control 7
8. IANA Considerations 8
9. Security Considerations 8
10. References 8
10.1. Normative References 8
10.2. Informative References 8
11. Acknowledgements 8
Authors' Addresses 8

1. Introduction

Attested TLS (aTLS) enables a TLS server to provide cryptographic proof of its endpoint behavior, including configuration, identity, and software integrity. This is essential for high-assurance use cases such as zero-trust architectures, compliance verification, and supply chain security.

However, the timing and method of attestation delivery significantly affect both security strength and network deployability. This document considers three models:

Pre-handshake: Attestation sent before TLS begins.

Intra-handshake: Attestation embedded in TLS handshake messages.

Post-handshake: Attestation sent after handshake completion.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Attester: The entity that generates attestation evidence.

Verifier: The entity that validates the attestation.

Relying Party: The client or service relying on the attestation result.

TLS Exporter: A TLS 1.3 mechanism to derive keying material from the handshake.

Middlebox: Network devices that inspect or modify TLS traffic (e.g., firewalls, load balancers).

3. The Theoretical Upper Bound of Security

In an ideal world, aTLS should achieve the following security properties:

Cryptographic Binding: The attestation evidence MUST be cryptographically bound to the TLS session. Specifically, the evidence should cover the TLS handshake transcript (e.g., ClientHello...ServerHello). This prevents a "Man-in-the-Middle" (MITM) from stripping the attestation or replaying a valid attestation from a different session.

Freshness: The attestation must be fresh, preventing replay attacks. Ideally, the nonce used in the attestation is derived from the TLS handshake.

Zero-Knowledge (Optional Privacy): The protocol should allow for selective disclosure, proving properties (e.g., "I am running patched Linux") without revealing the exact measurement hash, if desired.

Formal Verification: The protocol composition (TLS + Attestation) should be formally verified to ensure no security degradation (e.g., no downgrade attacks).

The Upper Bound implies a tight integration where the attestation is signed over the TLS handshake state, effectively making the attestation a mandatory extension of the Finished messages.

4. The Practical Lower Bound of Deployment

In the real world, protocols face harsh constraints that define the "Lower Bound" of what is acceptable:

Middlebox Compatibility: Enterprise networks, firewalls, and TLS terminators often inspect or modify TLS handshakes. A protocol that breaks these middleboxes (e.g., by adding large custom extensions that are stripped) will fail to deploy.

Library Fragmentation: Not all clients and servers use the latest OpenSSL/BoringSSL. The solution must ideally be backward compatible or implementable as a modular extension.

Hardware Asymmetry:

-High-End: Servers with TPMs 2.0 or Intel SGX/TDX can perform complex signing operations during the handshake.

-Low-End: IoT devices or highly loaded servers may experience significant latency if the handshake is blocked waiting for hardware signing.

Operational Complexity: Verifying attestation requires a complex infrastructure (Verifier, Policy Engine). If the protocol requires the TLS server to act as the Verifier for the client, it increases server load and state management complexity.

Passive Observation: In some deployment models (e.g., mutual TLS), the verifier might be an offline entity analyzing logs rather than the immediate peer, requiring the attestation to be accessible post-handshake.

The Lower Bound dictates that the protocol must be resilient to extension stripping, must not add unacceptable latency, and must allow for a "pass-through" mode where the TLS stack is unaware of the attestation semantics.

5. Selection Analysis

5.1. The "Binding" Gap

The primary tension lies in Binding.

Theory: The attestation signature MUST cover the ClientHello.random and ServerHello.random.

Practice: Generating this signature inside the TLS state machine is difficult for many libraries.

Selection Decision: The group should prefer mechanisms that use the TLS Exporter [RFC5705] to bridge this gap. This allows the attestation to be generated "outside" the strict handshake code but still be cryptographically bound to the session, satisfying the upper bound while respecting the lower bound of library modularity.

5.2. The "Middlebox" Gap

Theory: We control the whole stack.

Practice: The path contains F5 load balancers, legacy proxies, and SaaS inspection tools.

Selection Decision: Sending attestation evidence as Application Data (post-handshake) is the most robust against the middlebox "lower bound." However, to mitigate the security risk of MITM stripping, the protocol MUST implement a "fallback detection" mechanism. If the client expects attestation and does not receive it within the first N bytes, it MUST abort.

5.3. The "Latency" Gap

Theory: Attestation is instant.

Practice: TPM signing takes 50ms-200ms; SGX quoting is variable.

Selection Decision: To prevent blocking the TLS handshake (which hurts performance), Asynchronous Attestation should be supported. The connection can be established, and data can flow (perhaps restricted), while the attestation is being validated in the background. This lowers the performance impact to the practical floor.

6. Evaluation of Attestation Phases

6.1. Pre-Handshake Attestation

* *Description*: Attestation sent before TLS handshake begins.

* *Pros*:

  - Allows early rejection of untrusted servers.

- Can be used in stateless environments.

* *Cons*:

- No cryptographic binding to TLS session.

- Vulnerable to downgrade or substitution attacks.

- Requires out-of-band channel or non-standard port.

* *Verdict*: Not recommended due to weak session binding.

6.2. Intra-Handshake Attestation

* *Description*: Attestation embedded in TLS handshake messages (e.g., via extensions or Certificate).

* *Pros*:

- Strongest security: fully bound to handshake transcript.

- Enables mutual authentication with attestation.

- No added latency.

* *Cons*:

- High risk of *middlebox interference* (e.g., stripping unknown extensions).

- Difficult to deploy on the open Internet.

- May require protocol-specific integration.

* *Verdict*: High security, low deployability. Suitable only in controlled environments.

6.3. Post-Handshake Attestation

* *Description*: Attestation sent after handshake completion via application data.

* *Pros*:

- No middlebox issues (uses standard TLS record layer).

- Easy to implement and deploy.

- Compatible with TLS 1.3 and widely deployed stacks.

* *Cons*:

- Requires explicit binding to handshake (e.g., via exporter).

- Slight latency cost (one extra message).

* *Verdict*: Best balance of security and deployability.

7. Context-Dependent Selection

7.1. General Internet Deployment

For public-facing services (e.g., HTTPS, APIs), *post-handshake attestation* like[draft-fossati-seat-early-attestation] is REQUIRED due to widespread middlebox deployment and legacy infrastructure, and is recommended for broad interoperability.

7.2. Controlled Environments and Integrated Protocols

In enterprise networks, data centers, or IoT deployments, where: middleboxes are under administrative control, TLS stacks are upgradable, and Security is paramount, *Intra-handshake attestation* MAY be used when network environment allows.

7.3. Protocols provide easy attestation integration

Certain protocols, for example the LAKE Protocol [LAKE-RA], provide native support for attestation integration without requiring protocol modifications. In such cases, *Intra-handshake attestation is RECOMMENDED*. This approach enables zero-cost attestation, optimal cryptographic binding, and protocol-level trust composition.

7.4. Niche Case: Pre-Handshake for Network Admission Control

In architectures featuring a validating gateway (e.g., a Zero Trust Network Access gateway):

1. The client attests to the gateway via a separate protocol (e.g., EST over CoAP, or a custom UDP protocol).

2. The gateway validates the attestation.

3. The gateway issues a short-lived Attestation Token (e.g., a psk_identity for TLS-PSK).

4. The client initiates the TLS handshake to the internal server, presenting the Token.

5. The server validates the Token to authorize the session.

Pre-handshake attestation is RECOMMENDED for gateway-based NAC, provided the Token is cryptographically bound to the TLS session.

8. IANA Considerations

TODO

9. Security Considerations

TODO

10. References

10.1. Normative References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", August 2018.

[RFC5705] Rescorla, E., "Keying Material Exporters for TLS", March 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, March 1997.

10.2. Informative References

[draft-fossati-seat-early-attestation] draft-fossati-seat-early-attestation, Using Attestation in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)

[LAKE-RA] draft-ietf-lake-ra, Remote attestation over EDHOC

11. Acknowledgements

TODO

Authors' Addresses

Jun Zhang
Huawei Technologies France S.A.S.U.
Email: junzhang1@huawei.com

Houda Labiod
Huawei Technologies France S.A.S.U.
Email: houda.labiod@huawei.com