Relay Attacks in Intra-handshake Attestation and Proposed Solution in Post-handshake Attestation

Muhammad Usama Sardar[1,2]

[1]TU Dresden, Germany

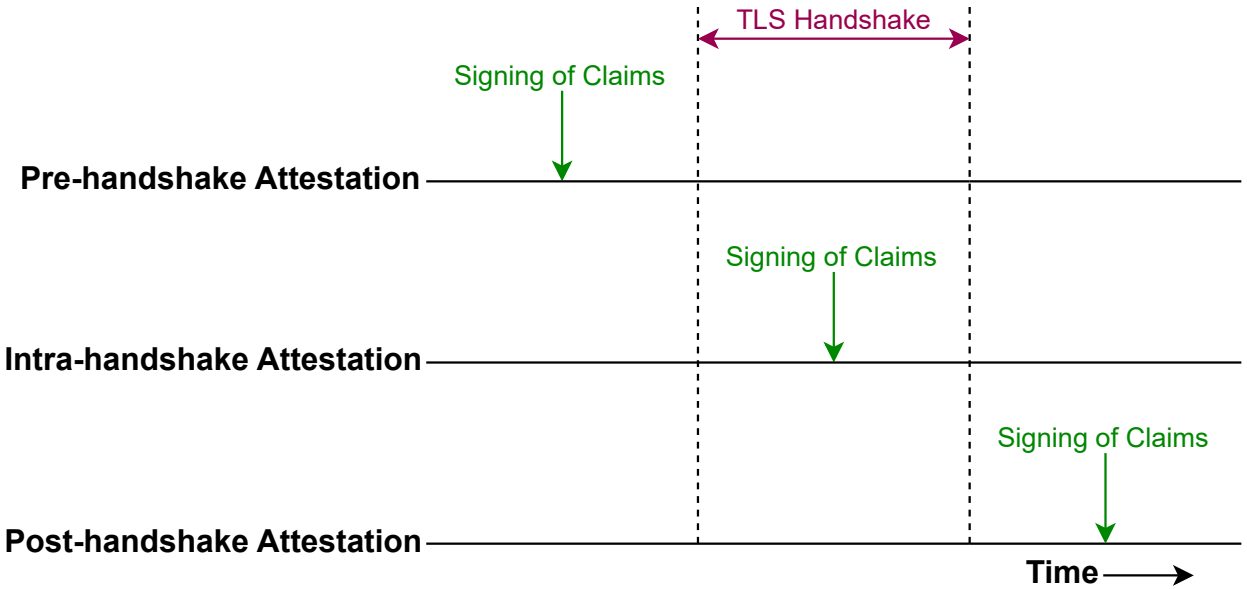
[2]Co-chair, Trusted Research Environment (TRE) Open Suite,
Global Alliance for Genomics and Health (GA4GH)

March 19, 2026

Outline

Categories of Attested TLS

- Related to work mainly being done in SEAT WG
  - Primarily a request to CFRG for recommendations and guidance
  - Follow-up of ML posts (relay attacks[1] and htsc vs. atsc[2])
- Temporal ordering of RA and TLS at Attester side



- Currently using ProVerif for formal analysis

[1] https://mailarchive.ietf.org/arch/msg/cfrg/NbxHIw9H_xpSYbgfO_n7lVIFeWs/

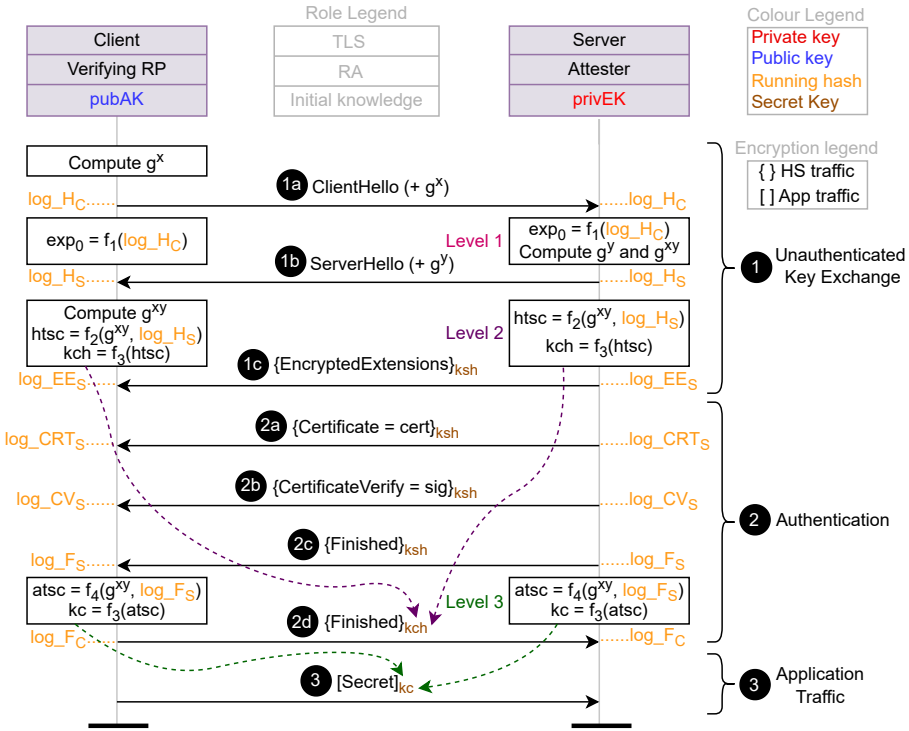
[2] https://mailarchive.ietf.org/arch/msg/cfrg/61SpXcv--YhcLbAvlcRog2J9S5s/

Binding Levels: Is Level 2 Sufficient?

Level	Secret	Handshake state
1	DH shared secret (g^{xy})	g^x from ClientHello and g^y from Server
2	Client's handshake traffic key (*htsc*)	complete ClientHello and ServerHello
3	Client's application traffic key (*atsc*)	ClientHello up to ServerFinished

- Request for CFRG recommendation/guidance: Is level 2 sufficient?

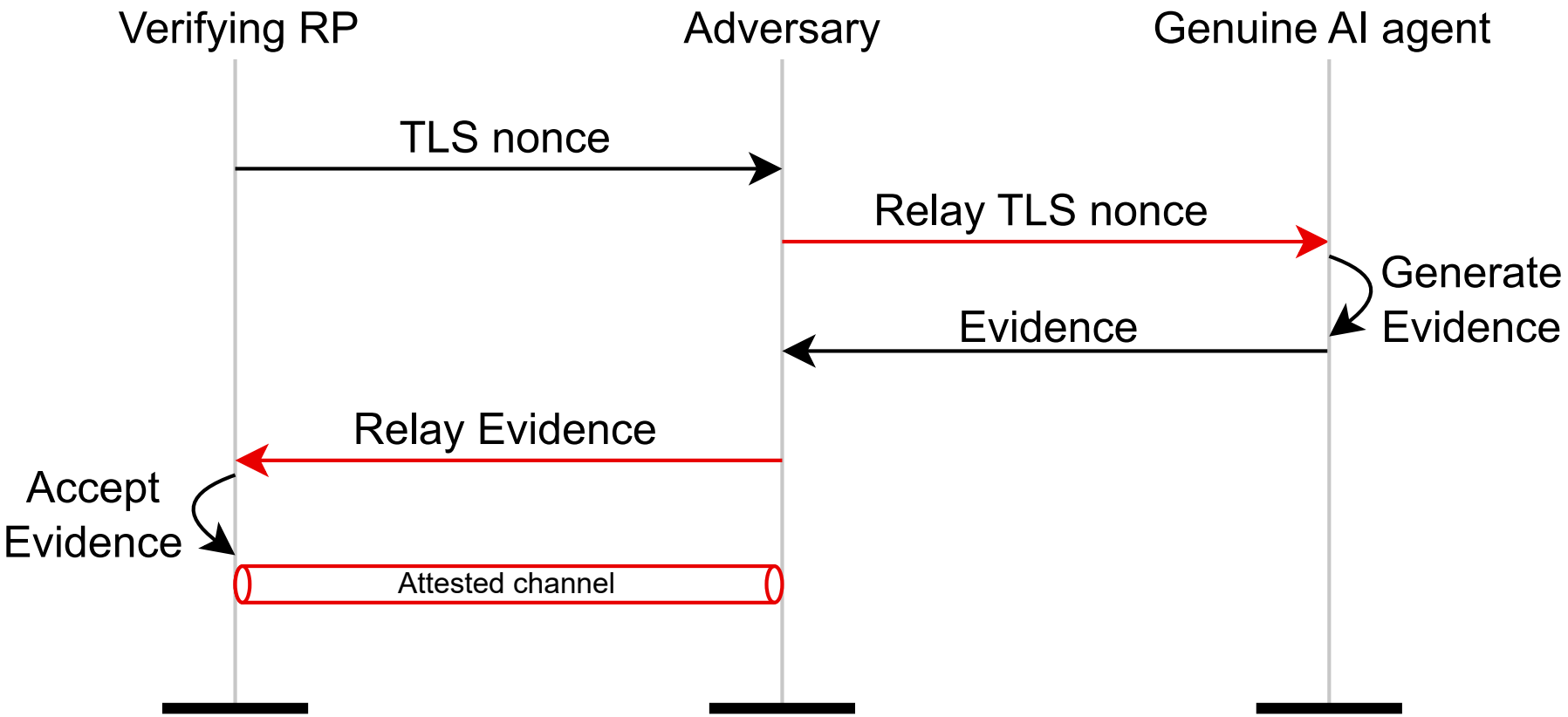
Is Binding to client_handshake_traffic_secret Sufficient?



- `htsc`: used for encryption of clientFinished message (2d).
  - Irrelevant for security goals
  - Server not yet authenticated at this point
- `atsc`: used for encryption of application data (client's secret, e.g., decryption key)
  - Relevant for security goals

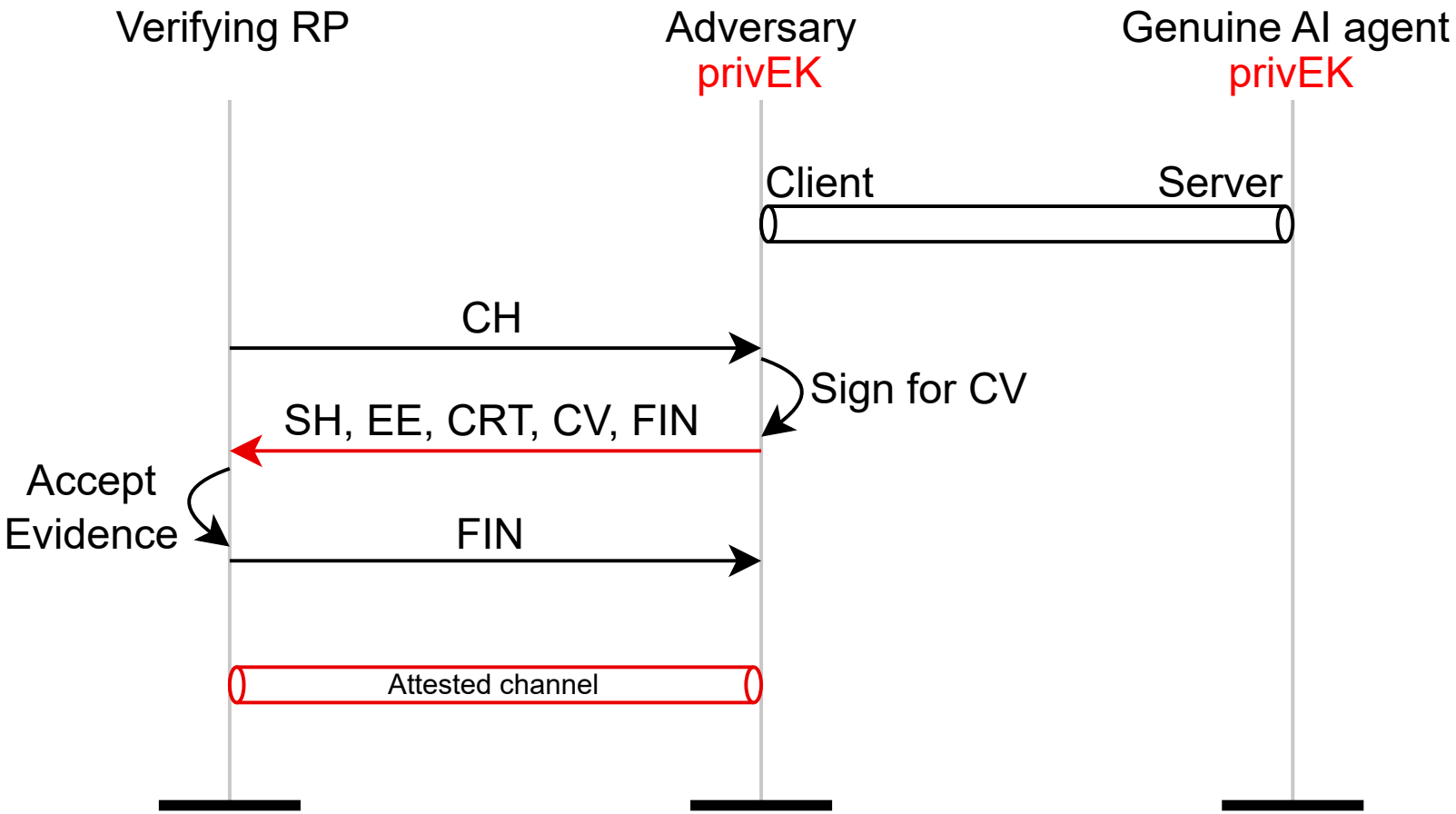
Why Binding Evidence to TLS Client Nonce is Insufficient?

- AI agent use case from SEAT WG (AI agent is TLS Server)
- Relay attack (abstracted): Adversary can relay nonce to a genuine Attester



Why Binding Evidence to Server's Public Key is Insufficient?

- No protection if privEK is accessible to some entity other than an AI agent (e.g., provisioning at runtime) or leaked via vulnerability in code
- Adversary gets signed Evidence from genuine AI agent



Relay Attacks Acknowledged by Cocos AI[3]

- Cocos AI uses "Attestation nonce ‖ Server's public key"
- Are there crypto mechanisms that can help?

Limitations and the Relay Attack

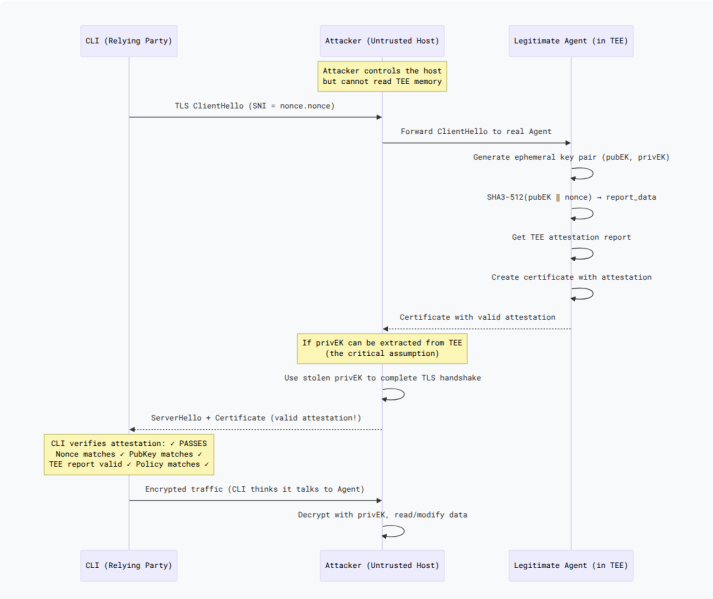
The Formal Analysis

Sardar et al. analyzed all known intra-handshake attestation implementations using ProVerif, a symbolic security analysis tool. Their key findings, presented at IETF SEAT meeting 124 and documented in draft-usama-seat-intra-vs-post:

1. **All** analyzed binding mechanisms fail to achieve even Level 1 binding (correlation of Evidence to the shared DH secret).
2. Any binding that involves the server's public key requires the **additional assumption** that the server's private key does not leak.
3. The extension of TLS with attestation in these implementations does not bring the security benefit one might expect from a purely theoretical perspective.

The Relay Attack Scenario

Here is the concrete attack that the formal analysis proves is possible:

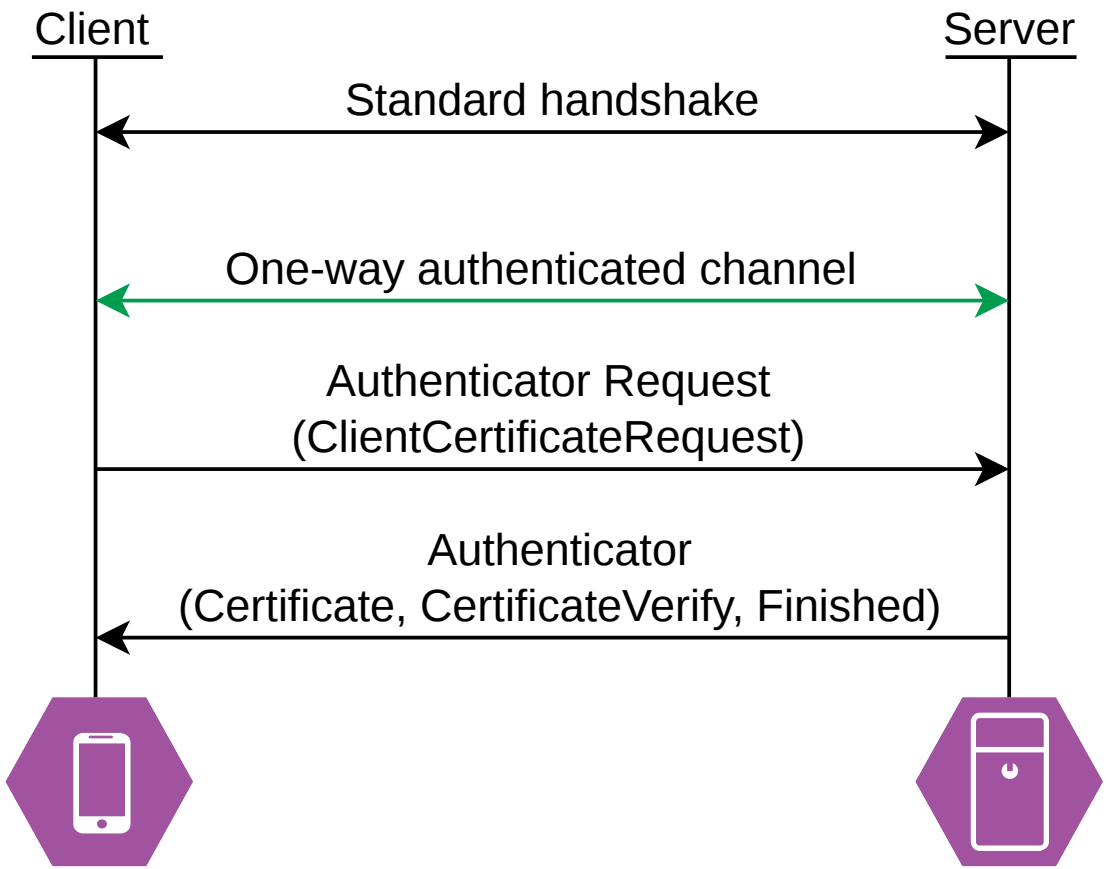


[3] https://web.archive.org/web/20260227160554/https://www.ultraviolet.rs/blog/tee-tls-privacy/

Outline

Exported Authenticators[4] (RFC 9261)

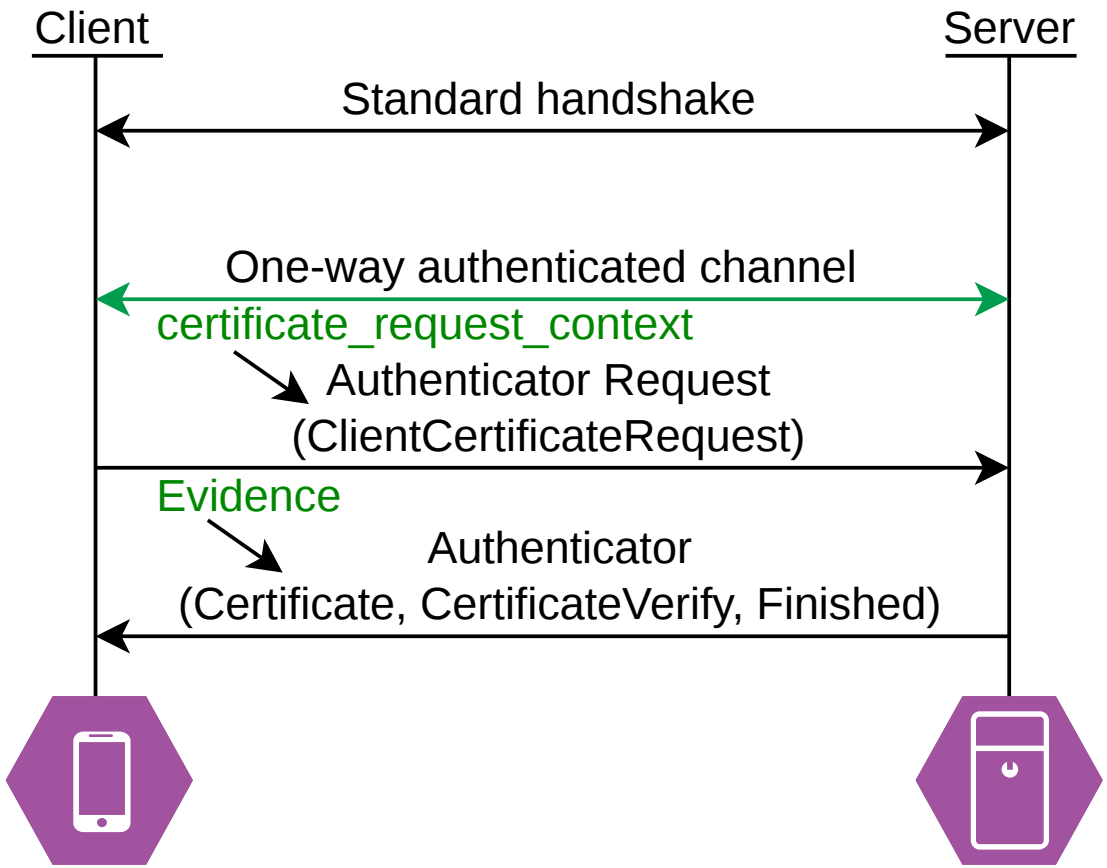
- ClientCertificateRequest: Same as CertificateRequest in RFC8446
- Key for HMAC of Finished using Exported Keying Material (EKM)



[4] Sullivan, *Exported Authenticators in TLS*, 2022.

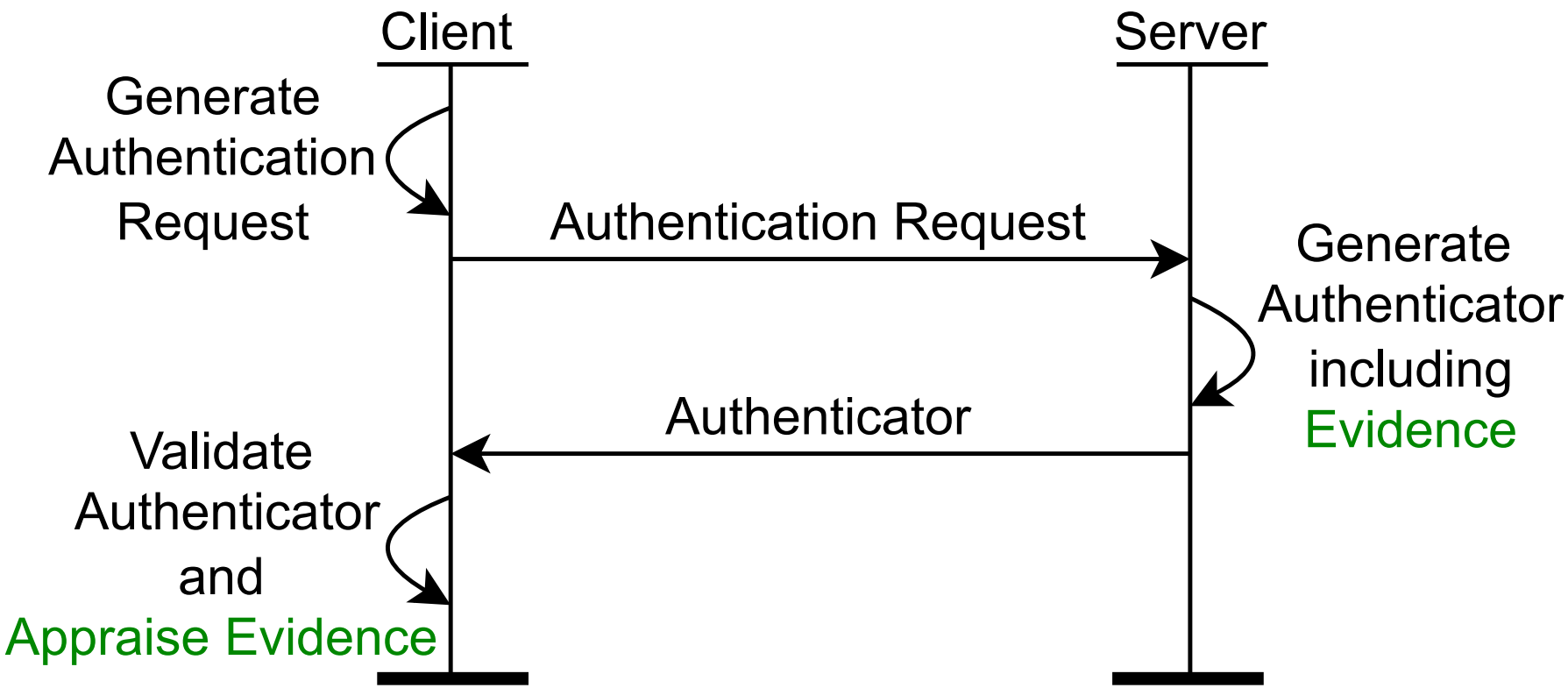
Post-handshake Attestation[5]

- No change in TLS handshake protocol
- Example: TLS Server as RATS Attester



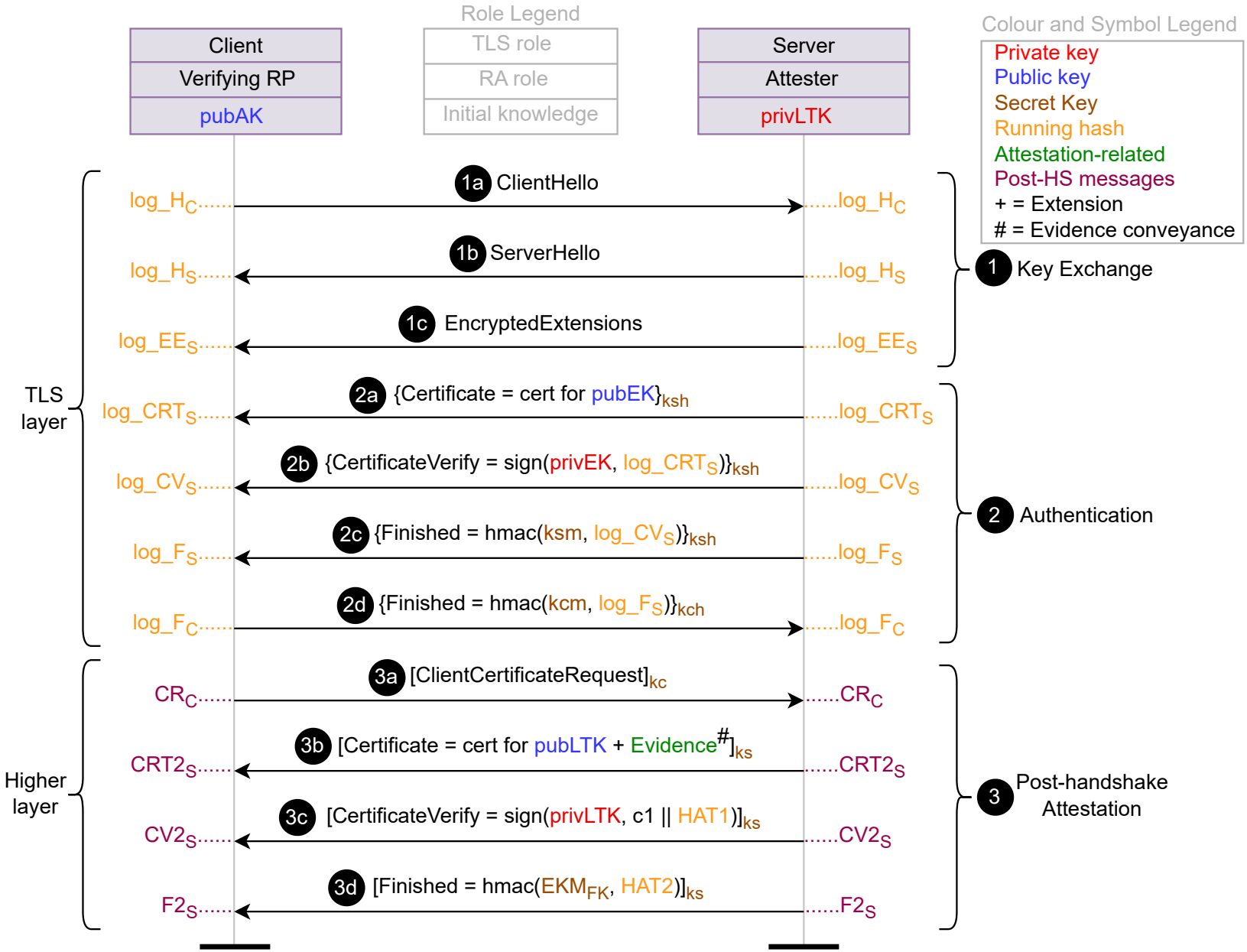
[5] Sardar, Moustafa, and Aura, "Identity Crisis in Confidential Computing: Formal Analysis of Attested TLS", 2026.

Overview of Post-handshake Flow



Outline

Detailed Protocol Design With PoP of privEK and privLTK



Proposal for Authenticator Request

- Let *na* represent the attestation nonce, then:

$$certificate\_request\_context = na \tag{1}$$

Proposal for Authenticator

- Certificate message with CMW extension for Evidence

$$EKM_{RD} = TLS\text{-}Exporter(\text{"}attestationRD\text{"}, na, L) \tag{2}$$

$$\begin{aligned} rdata &= Hash(pubEK \mid\mid EKM_{RD}) \\ ev &= sign(privAK, rdata||dev\_state) \end{aligned} \tag{3}$$

- CertificateVerify message

$$EKM_{HC} = TLS\text{-}Exporter(\text{"}attestationHC\text{"}, na, L) \tag{4}$$

$$\begin{aligned} AT1 &= EKM_{HC} \mid\mid CR_C \mid\mid CRT2_S \\ HAT1 &= Hash(AT1) \end{aligned} \tag{5}$$

- Finished message

$$EKM_{FK} = TLS\text{-}Exporter(\text{"}attestationFK\text{"}, na, L) \tag{6}$$

$$HAT2 = Hash(AT1 \mid\mid CV2_S) \tag{7}$$

Core Idea for Relay Attack Protection

- Exporters for Certificate, CertificateVerify, and Finished messages bind them all to the connection
- Evidence bound to a fully authenticated server
- Running hash to protect integrity of post-handshake messages
- Correlation properties in ProVerif
- Does RG recommend computational proof (using CryptoVerif)?

Welcome any feedback!

Links to Resources

- Wiki page
  - `github.com/EuroProofNet/ProgramVerification/wiki/AttestedTLS`
- Formal proof of insecurity of pre- and intra-handshake attestation
  - `github.com/CCC-Attestation/formal-spec-id-crisis`
- Attestation in Arm CCA and Intel TDX
  - `github.com/CCC-Attestation/formal-spec-TEE`
- Technical Concepts
- Validation of TLS 1.3 Key Schedule
- General Approach

Key References

Sardar, Muhammad Usama, Mariam Moustafa, and Tuomas Aura. "Identity Crisis in Confidential Computing: Formal Analysis of Attested TLS". In: *Proceedings of the 21st ACM ASIA Conference on Computer and Communications Security (ACM ASIACCS 2026)*. New York, NY, USA: ACM, 2026. URL: https://www.researchgate.net/publication/398839141_Identity_Crisis_in_Confidential_Computing_Formal_Analysis_of_Attested_TLS.

Sullivan, Nick. *Exported Authenticators in TLS*. RFC 9261. July 2022. DOI: 10.17487/RFC9261. URL: https://www.rfc-editor.org/info/rfc9261.