Secure Evidence and Attestation Transport (SEAT) Working Group
Internet-Draft
Intended status: Standards Track
Expires: 23 April 2026

T. Fossati
Linaro
M. U. Sardar
TU Dresden
T. Reddy
Nokia
Y. Sheffer
Intuit
H. Tschofenig
H-BRS
I. Mihalcea
Arm Limited

20 October 2025

Remote Attestation with Exported Authenticators
draft-fossati-seat-expat-00

Abstract

This specification defines a method for two parties in a communication interaction to exchange Evidence and Attestation Results using exported authenticators, as defined in [RFC9261]. Additionally, it introduces the cmw_attestation extension, which allows attestation credentials to be included directly in the Certificate message sent during the Exported Authenticator-based post-handshake authentication. The approach supports both the passport and background check models from the RATS architecture while ensuring that attestation remains bound to the underlying communication channel.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://tls-attestation.github.io/exported-attestation/draft-fossati-seat-expat.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-fossati-seat-expat/.

Discussion of this document takes place on the SEAT Working Group mailing list (mailto:seat@ietf.org), which is archived at https://datatracker.ietf.org/wg/seat/about/. Subscribe at https://www.ietf.org/mailman/listinfo/seat/.

Source for this draft and an issue tracker can be found at https://github.com/tls-attestation/exported-attestation.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

There is a growing need to demonstrate to a remote party that cryptographic keys are stored in a secure element, the device is in a known good state, secure boot has been enabled, and that low-level software and firmware have not been tampered with. Remote attestation provides this capability.

More technically, an Attester produces a signed collection of Claims that constitute Evidence about its running environment(s). A Relying Party may consult an Attestation Result produced by a Verifier that has appraised the Evidence to make policy decisions regarding the trustworthiness of the Target Environment being assessed. This is, in essence, what [RFC9334] defines.

At the time of writing, several standard and proprietary remote attestation technologies are in use. This specification aims to remain as technology-agnostic as possible concerning implemented remote attestation technologies. To streamline attestation in TLS, this document introduces the cmw_attestation extension, which allows attestation credentials to be conveyed directly in the Certificate message during the Exported Authenticator-based post-handshake authentication. This eliminates reliance on real-time certificate issuance from a Certificate Authority (CA), reducing handshake delays while ensuring Evidence remains bound to the TLS session. The extension supports both the passport and background check models from the RATS architecture, enhancing flexibility for different deployment scenarios.

This document builds upon three foundational specifications:

* RATS (Remote Attestation Procedures) Architecture [RFC9334]: It defines how remote attestation systems establish trust between parties by exchanging Evidence and Attestation Results. These interactions can follow different models, such as the passport or the background check model, depending on the order of data flow in the system.

* TLS Exported Authenticators [RFC9261]: It offers bi-directional post-handshake authentication. Once a TLS connection is established, both peers can send an authenticator request message at any point after the handshake. This message from the server and the client uses the CertificateRequest and the ClientCertificateRequest messages, respectively. The peer receiving the authenticator request message can respond with an Authenticator consisting of Certificate, CertificateVerify, and Finished messages. These messages can then be validated by the other peer.

* RATS Conceptual Messages Wrapper (CMW) [I-D.ietf-rats-msg-wrap]: CMW provides a structured encapsulation of Evidence and Attestation Result payloads, abstracting the underlying attestation technology.

This specification introduces the cmw_attestation extension, enabling Evidence to be included directly in the Certificate message during the Exported Authenticator-based post-handshake authentication defined in [RFC9261].

2. Terminology

The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, NOT RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals as shown here.

The reader is assumed to be familiar with the vocabulary and concepts defined in [RFC9334] and [RFC9261].

"Remote attestation credentials", or "attestation credentials", is used to refer to both Evidence and attestation results, when no distinction needs to be made between them.

3. cmw_attestation Extension to the Authenticator's Certificate message

This document introduces a new extension, called cmw_attestation, to the Authenticator's Certificate message. This extension allows Evidence or Attestation Results to be included in the extensions field of the end-entity certificate in the TLS Certificate message.

As defined in Section 4.4.2 of [RFC8446], the TLS Certificate message consists of a certificate_list, which is a sequence of CertificateEntry structures. Each CertificateEntry contains a certificate and a set of associated extensions. The cmw_attestation extension MUST appear only in the first CertificateEntry of the Certificate message and applies exclusively to the end-entity certificate. It MUST NOT be included in entries corresponding to intermediate or trust anchor certificates. This design ensures that attestation information is tightly bound to the entity being authenticated.

The cmw_attestation extension is only included in the Certificate message during Exported Authenticator-based post-handshake authentication. This ensures that the attestation credentials are conveyed within the Certificate message, eliminating the need for modifications to the X.509 certificate structure.

```
struct {
    opaque cmw_data<1..2^16-1>;
} CMWAttestation;
```

cmw_data: Encapsulates the attestation credentials in CMW format [I-D.ietf-rats-msg-wrap]. The cmw_data field is encoded using CBOR or JSON.

This approach eliminates the need for real-time certificate issuance from a Certificate Authority (CA) and minimizes handshake delays. Typically, CAs require several seconds to minutes to issue a certificate due to verification steps such as validating subject identity, signing the certificate, and distributing it. These delays introduce latency into the TLS handshake, making real-time certificate generation impractical. The cmw_attestation extension circumvents this issue by embedding attestation data within the Certificate message itself, removing reliance on external certificate issuance processes.

3.1. Negotiation of the cmw_attestation Extension

Negotiation of support cmw_attestation extension follows the model defined in Section 5.2 of [RFC9261].

Endpoints that wish to receive attestation credentials using Exported Authenticators MUST indicate support by including an empty cmw_attestation extension in the CertificateRequest or ClientCertificateRequest message. The presence of this empty extension indicates that the requester understands this specification and is willing to process an attestation credential in the peer’s Certificate message.

An endpoint that supports this extension and receives a request containing it MAY include the cmw_attestation extension in its Certificate message, populated with attestation data. If the cmw_attestation extension appears in a Certificate message without it having been previously offered in the corresponding request, the receiver MUST abort the authenticator verification with an "unsupported_extension" alert. As specified in Section 9.3 of [RFC8446], endpoints that do not recognize the cmw_attestation extension in a CertificateRequest or ClientCertificateRequest MUST ignore it and continue processing the message as if the extension were absent.

3.2. Usage in Exported Authenticator-based Post-Handshake Authentication

The cmw_attestation extension is designed to be used exclusively in Exported Authenticator-based post-handshake authentication as defined in [RFC9261]. It allows attestation credentials to be transmitted in the Authenticator’s Certificate message only in response to an Authenticator Request. This ensures that attestation credentials are provided on demand rather than being included in the initial TLS handshake.

To maintain a cryptographic binding between the Evidence and the authentication request, the cmw_attestation extension MUST be associated with the certificate_request_context of the corresponding CertificateRequest or ClientCertificateRequest message (from the Server or Client, respectively). This binding ensures that:

* The Evidence is specific to the authentication event and cannot be replayed across different TLS sessions.

* The Evidence remains tied to the cryptographic context of the TLS session.

3.3. Ensuring Compatibility with X.509 Certificate Validation

The cmw_attestation extension does not modify or replace X.509 certificate validation mechanisms. It serves as an additional source of authentication data rather than altering the trust model of PKI-based authentication. Specifically:

* Certificate validation (e.g., signature verification, revocation checks) MUST still be performed according to TLS [RFC8446] and PKIX [RFC5280].

* The attestation credentials carried in cmw_attestation MUST NOT be used as a substitute for X.509 certificate validation but can be used alongside standard certificate validation for additional security assurances.

* Implementations MAY reject connections where the certificate is valid but the attestation credentials is missing or does not meet security policy.

3.4. Applicability to Client and Server Authentication

The cmw_attestation extension is applicable to both client and server authentication in Exported Authenticator-based post-handshake authentication.

In TLS, one party acts as the Relying Party, and the other party acts as the Attester. Either the client or the server may fulfill these roles depending on the authentication direction.

The Attester may respond with either:

* Evidence (Background Check Model):

  - The Attester generates Evidence and includes it in the cmw_attestation extension to the Authenticator's Certificate message.

  - The Relying Party forwards the Evidence to an external Verifier for evaluation and waits for an Attestation Result.

  - The Relying Party grants or denies access, or continues or terminates the TLS session, based on the Verifier's Attestation Result.

* Attestation Result (Passport Model):

  - The Attester sends Evidence to a Verifier beforehand.

   - The Verifier issues an Attestation Result to the Attester.

   - The Attester includes the Attestation Result in the cmw_attestation extension to the Authenticator's Certificate message and sends it to the Relying Party.

   - The Relying Party validates the Attestation Result directly without needing to contact an external Verifier.

By allowing both Evidence and Attestation Results to be conveyed within cmw_attestation, this mechanism supports flexible attestation workflows depending on the chosen trust model.

4. Architecture

The cmw_attestation extension enables attestation credentials to be included in the Certificate message during Exported Authenticator-based post-handshake authentication, ensuring that attestation remains bound to the TLS session.

However, applications using this mechanism still need to negotiate the encoding format (e.g., JOSE or COSE) and specify how attestation credentials are processed. This negotiation can be done via application-layer signaling or predefined profiles. Future specifications may define mechanisms to streamline this negotiation.

Upon receipt of a Certificate message containing the cmw_attestation extension, an endpoint MUST take the following steps to validate the attestation credentials:

* Background Check Model:

   - Verify Integrity and Authenticity: The Evidence must be cryptographically verified against a known trust anchor, typically provided by the hardware manufacturer.

   - Ensure Certificate Binding and Freshness: The Evidence must be explicitly associated with the certificate_request_context in the authenticator request to ensure relevance, freshness, and protection against replay.

   - Evaluate Security Policy Compliance: The Evidence must be evaluated against the Relying Party's security policies to determine if the attesting device and the private key storage meet the required criteria.

* Passport Model:

- Verify the Attestation Result: The Relying Party MUST check that the Attestation Result is correctly signed by the issuing authority and that it meets the Relying Party’s security requirements.

By integrating cmw_attestation directly into the Certificate message during Exported Authenticator-based post-handshake authentication, this approach reduces latency and complexity while maintaining strong security guarantees.

In the following examples, the server possesses an identity certificate, while the client is not authenticated during the initial TLS exchange.

Figure 1 shows the passport model while Figure 2 illustrates the background-check model.

```
Client                   Server                   Verifier
  |                        |                         |
  |  Regular TLS Handshake |                         |
  |    (Server-only auth)  |                         |
  |<---------------------->|                         |
  |                        |                         |
  |  ... time passes ...   |                         |
  |                        |                         |
  | Authenticator Request  |                         |
  | (ClientCertificateReq) |                         |
  |<-----------------------|                         |
  |                        |                         |
  |                  Sends Evidence                  |
  |------------------------------------------------->|
  |                 Gets Attestation result          |
  |<-------------------------------------------------|
  | Exported Authenticator(|                         |
  | Certificate with       |                         |
  | cmw_attestation,       |                         |
  | CertificateVerify,     |                         |
  | Finished)              |                         |
  |----------------------->|                         |
```

Figure 1: Passport Model with Client as Attester

Figure 2 shows an example using the background-check model.

```
  Client              Attester                 Server          Verifier
    |                   |                        |                  |
    |  Regular TLS Handshake (Server-only auth)  |                  |
    |<------------------------------------------>|                  |
    |                   |                        |                  |
    |   ... time passes ...                      |                  |
    |                   |                        |                  |
    | Authenticator Request (ClientCertReq)      |                  |
    |<-------------------------------------------|                  |
    |                   |                        |                  |
    | Request Evidence  |                        |                  |
    |------------------>|                        |                  |
    | Attestation       |                        |                  |
    | Evidence          |                        |                  |
    |<------------------|                        |                  |
    | Exported Authenticator(Certificate with    |                  |
    | cmw_attestation                            |                  |
    | CertificateVerify,                         |                  |
    | Finished)                                  |                  |
    |------------------------------------------->|                  |
    |                   |                        | Send Evidence    |
    |                   |                        |----------------->|
    |                   |                        | Attestation      |
    |                   |                        | Result           |
    |                   |                        |<-----------------|
    |                   |                        |                  |
```

Figure 2: Background Check Model with a Separate Client-Side Attester

4.1. API Requirements for Attestation Support

To enable attestation workflows, implementations of the Exported Authenticator API MUST support the following:

1. Authenticator Generation

   * The API MUST support the inclusion of attestation credentials within the Certificate message provided as input.

2. Context Retrieval

   * The certificate_request_context MUST be provided in all cases to ensure proper validation of Evidence.

   * The receiving endpoint MUST use the "get context" API to retrieve the certificate_request_context associated with the exported authenticator as attestation-based authentication requires strict enforcement of the request context. This ensures that the freshness of Evidence can be verified.

3. Authenticator Validation

   * The API MUST verify that the Evidence within the Certificate message is cryptographically valid and bound to the certificate_request_context.

5. Security Considerations

This document inherits the security considerations of [RFC9261] and [RFC9334]. The integrity of the exported authenticators must be guaranteed, and any failure in validating Evidence SHOULD be treated as a fatal error in the communication channel. Additionally, in order to benefit from remote attestation, Evidence MUST be protected using dedicated attestation keys chaining back to a trust anchor. This trust anchor will typically be provided by the hardware manufacturer.

This specification assumes that the Hardware Security Module (HSM) or Trusted Execution Environment (TEE) is responsible for generating the key pair and producing either Evidence or attestation results, which is included in the Certificate Signing Request (CSR) as defined in [I-D.ietf-lamps-csr-attestation]. This attestation enables the CA to verify that the private key is securely stored and that the platform meets the required security standards before issuing a certificate.

5.1. Security Guarantees

Note that as a pure cryptographic protocol, attested TLS as-is only guarantees that the identity key used for TLS handshake is known by the confidential environment, such as confidential virtual machine. A number of additional guarantees must be provided by the platform and/or the TLS stack, and the overall security level depends on their existence and quality of assurance:

* The identity key used for TLS handshake is generated within the trustworthy environment, such as Trusted Platform Module (TPM) or TEE.

* The identity key used for TLS handshake is never exported or leaked outside the trustworthy environment.

* For confidential computing use cases, the TLS protocol is implemented within the confidential environment, and is implemented correctly, e.g., it does not leak any session key material.

* The TLS stack including the code that performs the post-handshake phase must be measured.

* There must be no other way to initiate generation of evidence except from signed code.

These properties may be explicitly promised ("attested") by the platform, or they can be assured in other ways such as by providing source code, reproducible builds, formal verification etc. The exact mechanisms are out of scope of this document.

5.2. Using the TLS Connection

Remote attestation in this document occurs within the context of a TLS handshake, and the TLS connection remains valid after this process. Care must be taken when handling this TLS connection, as both the client and server must agree that remote attestation was successfully completed before exchanging data with the attested party.

Session resumption presents special challenges since it happens at the TLS level, which is not aware of the application-level Authenticator. The application (or the modified TLS library) must ensure that a resumed session has already completed remote attestation before the session can be used normally, and race conditions are possible.

5.3. Evidence Freshness

The Evidence carried in cmw_attestation does not require an additional freshness mechanism, such as a nonce [RA-TLS] or timestamp, since freshness is inherently provided by the certificate_request_context in the authenticator request.

The evidence presented in this protocol is valid only at the time it is generated and presented. To ensure that the attested peer remains in a secure state, remote attestation may be re-initiated periodically. In the current protocol, this can be achieved by initiating a new Exported Authenticator-based post-handshake authentication exchange, which will generate a new certificate_request_context to maintain freshness.

6. Privacy Considerations

6.1. Client as Attester

In this section, we are assuming that the Attester is a TLS client, representing an individual person. We are concerned about the potential leakage of privacy-sensitive information about that person, such as the correlation of different connections initiated by them.

In background-check model, the Verifier not only has access to detailed information about the Attester’s TCB through Evidence, but it also knows the exact time and the party (i.e., the RP) with whom the secure channel establishment is attempted [RA-TLS]. The privacy implications are similar to OCSP [RFC6960]. While the RP may trust the Verifier not to disclose any information it receives, the same cannot be assumed for the Attester, which generally has no prior relationship with the Verifier. Some ways to address this include:

* Attester-side redaction of privacy-sensitive evidence claims,

* Using selective disclosure (e.g., SD-JWT [I-D.ietf-oauth-selective-disclosure-jwt] with EAT [RFC9711]),

* Co-locating the Verifier role with the RP,

* Utilizing privacy-preserving attestation schemes (e.g., DAA [I-D.ietf-rats-daa]), or

* Utilizing Attesters manufactured with group identities (e.g., Requirement 4.1 of [FIDO-REQS]).

The last two also have the property of hiding the peer’s identity from the RP.

Note that the equivalent of OCSP "stapling" involves using a passport topology where the Verifier’s involvement is unrelated to the TLS session.

6.2. Server as Attester

For the case of the TLS server as the Attester, the server can ask for client authentication and only send the Evidence after successful client authentication. This limits the exposure of server’s hardware-level Claims to be revealed only to authorized clients.

7. IANA Considerations

// Note to RFC Editor: in this section, please replace RFCthis with the RFC number assigned to this document and remove this note.

7.1. TLS Extension Type Registration

IANA is requested to register the following new extension type in the "TLS ExtensionType Values" registry [IANA.tls-extensiontype-values]:

Value	Extension Name	TLS 1.3	DTLS-Only	Recommended	Reference
TBD	cmw_attestation	CT	N	Yes	RFCthis

Table 1

7.2. TLS Flags Extension Registry

IANA is requested to add the following entry to the "TLS Flags" extension registry established by [I-D.ietf-tls-tlsflags]:

* Value: TBD1

* Flag Name: CMW_Attestation

* Messages: CH, EE

* Recommended: Y

* Reference: RFCthis

8. References

8.1. Normative References

[I-D.ietf-rats-msg-wrap]
Birkholz, H., Smith, N., Fossati, T., Tschofenig, H., and D. Glaze, "RATS Conceptual Messages Wrapper (CMW)", Work in Progress, Internet-Draft, draft-ietf-rats-msg-wrap-19, 17 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-msg-wrap-19>.

[I-D.ietf-tls-tlsflags]
Nir, Y., "A Flags Extension for TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-tlsflags-16, 14 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-tlsflags-16>.

[IANA.tls-extensiontype-values]
IANA, "Transport Layer Security (TLS) Extensions", <https://www.iana.org/assignments/tls-extensiontype-values>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

[RFC9261] Sullivan, N., "Exported Authenticators in TLS", RFC 9261, DOI 10.17487/RFC9261, July 2022, <https://www.rfc-editor.org/rfc/rfc9261>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

8.2. Informative References

[FIDO-REQS]
Peirani, B. and J. Verrept, "FIDO Authenticator Security and Privacy Requirements", March 2025, <https://fidoalliance.org/specs/fido-security-requirements/>.

[I-D.ietf-lamps-csr-attestation]
Ounsworth, M., Tschofenig, H., Birkholz, H., Wiseman, M., and N. Smith, "Use of Remote Attestation with Certification Signing Requests", Work in Progress, Internet-Draft, draft-ietf-lamps-csr-attestation-21, 5 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-lamps-csr-attestation-21>.

[I-D.ietf-oauth-selective-disclosure-jwt]
Fett, D., Yasuda, K., and B. Campbell, "Selective Disclosure for JWTs (SD-JWT)", Work in Progress, Internet-Draft, draft-ietf-oauth-selective-disclosure-jwt-22, 29 May 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-selective-disclosure-jwt-22>.

[I-D.ietf-rats-daa]
Birkholz, H., Newton, C., Chen, L., Giannetsos, T., and D. Thaler, "Direct Anonymous Attestation for the Remote Attestation Procedures Architecture", Work in Progress, Internet-Draft, draft-ietf-rats-daa-08, 3 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-daa-08>.

[RA-TLS] Sardar, M. U., Niemi, A., Tschofenig, H., and T. Fossati, "Towards Validation of TLS 1.3 Formal Model and Vulnerabilities in Intel’s RA-TLS Protocol", November 2024, <https://ieeexplore.ieee.org/document/10752524>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/rfc/rfc6960>.

[RFC9711] Lundblade, L., Mandyam, G., O’Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", RFC 9711, DOI 10.17487/RFC9711, April 2025, <https://www.rfc-editor.org/rfc/rfc9711>.

[TLS-Ext-Registry]
IANA, "Transport Layer Security (TLS) Extensions", November 2023, <https://www.iana.org/assignments/tls-extensiontype-values>.

Acknowledgements

We would like to thank Chris Patton for his proposal to explore RFC 9261 for attested TLS. We would also like to thank Eric Rescorla, Paul Howard, and Yogesh Deshpande for their input.

Post-handshake vs Intra-handshake Privacy

From the view of the TLS server, post-handshake attestation offers better privacy than intra-handshake attestation when the server acts as the Attester. In intra-handshake attestation, due to the inherent asymmetry of the TLS protocol, a malicious TLS client could potentially retrieve sensitive information from the Evidence without the client’s trustworthiness first being established by the server. In post-handshake attestation, the server can ask for client authentication and only send the Evidence after successful client authentication.

Document History

-03

* Expanded security considerations, in particular added security guarantees

* Added privacy considerations

* Corrected Figure 1

Authors’ Addresses

Thomas Fossati
Linaro
Email: thomas.fossati@linaro.org

Muhammad Usama Sardar
TU Dresden
Email: muhammad_usama.sardar@tu-dresden.de

Tirumaleswar Reddy
Nokia
Bangalore
Karnataka
India
Email: k.tirumaleswar_reddy@nokia.com

Yaron Sheffer
Intuit
Email: yaronf.ietf@gmail.com

Hannes Tschofenig
University of Applied Sciences Bonn-Rhein-Sieg
Germany
Email: Hannes.Tschofenig@gmx.net

Ionut Mihalcea
Arm Limited
Email: ionut.mihalcea@arm.com