

TLS Working Group
Internet Draft
Intended status: Experimental

P. Urien
Telecom Paris
Ethertrust

18 March 2026

Expires: 17 September 2026

TLS for Secure Element Recursive Authentication
draft-urien-tls-se-xauth-01

Abstract

This document defines a recursive authentication architecture based on the TLS 1.3 pre-shared key (PSK) mode. In this context, TLS servers, typically hosted within secure elements (TLS-SE), realize procedures that compute TLS 1.3 PSK-binder and Handshake Secret. These procedures allow a client to authenticate to downstream TLS servers without directly possessing the corresponding PSKs. Authentication capabilities can therefore be delegated across multiple TLS servers while maintaining protection of the underlying secrets.

Requirements Language

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.

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 September 2026.

.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

Abstract.....	1
Requirements Language.....	1
Status of this Memo.....	1
Copyright Notice.....	2
1 Introduction.....	3
1.1 PSK Binder procedure.....	3
1.2 Handshake Secret Procedure.....	3
2 Recursive authentication with TLS-PSK.....	4
2.1 Overview.....	4
2.2 Recursive Authentication.....	4
2.3 Root Server.....	5
2.4 Crypto Provider.....	5
2.5 Target server.....	5
3 1-HOP session.....	6
4 2-HOPs session.....	6
5 n-HOPs session.....	7
6 IANA Considerations.....	7
7 Security Considerations.....	7
8 References.....	8
8.1 Normative References.....	8
8.2 Informative References.....	8
9 Authors' Addresses.....	9

1 Introduction

TLS1.3 [RFC8446] supports a basic key exchange mode based on pre-shared-key (PSK) with Diffie-Hellman over either finite fields or elliptic curves (EC)DHE, in short TLS1.3-PSK for PSK with (EC)DHE. According to [RFC8446] external PSKs MAY be provisioned outside of TLS [RFC9257].

TLS1.3-PSK server MAY be implemented in secure elements [TLS-SE]. The TLS1.3 client MAY use a TLS Identity Module [TLS-IM], that protects procedures dealing with PSK.

1.1 PSK Binder procedure

The Early Secret (ESK) is computed according to relation:

$$\text{ESK} = \text{HKDF-Extract}(\text{salt}=0s, \text{PSK}) = \text{HMAC}(\text{salt}=0s, \text{PSK})$$

The Binder Key (BSK) for outside provisioning is computed according to the relation:

$$\text{BSK} = \text{Derive-Secret}(\text{ESK}, \text{"ext binder"}, \text{""})$$

The Finished External Key (FEK) is computed according to the relation:

$$\text{FEK} = \text{KDF-Expand-Label}(\text{BSK}, \text{"finished"}, \text{""}, \text{Hash.length})$$

For Derive-Secret procedures, "" is equivalent to the value hash(empty), whose size is hash-length.

The PSK Binder is computed as:

$$\text{PSK-Binder} = \text{HMAC}(\text{FEK}, \text{transcript hash})$$

PSK-Binder is included in the Client Hello message.

We name binder(transcript hash) or binder(T) the procedure that computes the PSK-Binder. Because PSK relies on psk-identity inserted in the ClientHello message, we note by binder(ID,T) this relationship.

1.2 Handshake Secret Procedure

The Derived Secret (DSK) is computed according to the relation:

$$\text{DSK} = \text{Derive-Secret}(\text{ESK}, \text{"derived"}, \text{""}).$$

The Handshake Secret (HS) is computed as:

$$\text{HS} = \text{HKDF-Extract}(\text{salt}=\text{DSK}, (\text{EC})\text{DHE})$$

We name derive(DHE) the procedure that computes the Handshake Secret. Because PSK relies on psk-identity inserted in the ClientHello message, we note by derive(ID,DHE) this relationship.

The recovery of HS needs the knowledge of both PSK and (EC)DHE. Therefore, even if PSK has low entropy, it is extremely difficult to retrieve the HS value.

2 Recursive authentication with TLS-PSK

2.1 Overview

In traditional TLS-PSK deployments the client must possess the PSK locally in order to compute the PSK-Binder and the Handshake Secret. Distributing PSKs to clients introduces operational challenges and security risks, especially in large distributed systems.

This document introduces a recursive authentication model in which TLS1.3 PSK servers perform PSK-related cryptographic computations on behalf of the client. The client invokes these procedures during the TLS1.3 handshake without learning the underlying PSKs.

After authenticating to one TLS secure element server [TLS-SE], the client uses delegated cryptographic services from that server to authenticate to additional TLS-SE servers. Authentication relationships therefore form a recursive chain across multiple servers.

2.2 Recursive Authentication

The schema below illustrates recursive authentication with two TLS1.3 servers, server k and server k+1. Two procedures binder, which computes the PSK-Binder value, and derive, which computes the Handshake Secret are realized by server k and used for connection to server k+1. It should be noticed that server k generates Handshake Secret for server k+1, and therefore COULD decrypt further data exchanges between client and server k+1.

Client	Server k	Server k+1
=== TLS Channel Established ===		
----binder(IDk+1,T)----->		
<---PSK-Binder value-----		
----ClientHello(IDk+1,PSK-Binder)----->		
<---ServerHello-----		
Compute (EC)DHE		
----derive(IDk+1,DHE)----->		
<---HandshakeSecret value-----		
----Close TLS session----->		
<---Server Encrypted Extensions-----		
<---Server Finished-----		
----Client Finished----->		
===== TLS Channel Established =====		

2.3 Root Server

The root server (RS) is the first server to which client is connected; it provides binder and derive procedures to be used for connection to the second server (server 2). According to [TLS-SE] root server MAY be implemented in secure elements [ISO7816].

2.4 Crypto Provider

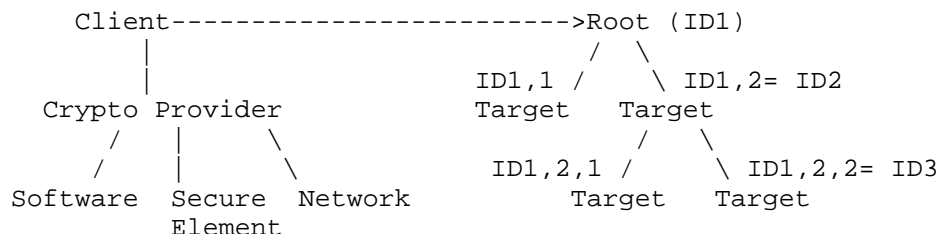
We call Crypto Provider (CP) the entity that knows the pre-shared-key to be used with root server (server 1), and therefore that is able to compute binder(T) and derive(DHE) procedures.

Crypto Provider could be realized in multiple computing environments such as:

- Classical software programs using PSK value to compute binder and derive procedures. PSK is exposed in software memory, and cloning is possible.
- Tamper resistant computing devices such as secure elements defined by [ISO7816] standards. The IETF draft [TLS-IM] defines ISO7816 interfaces for binder and derive procedures computed within secure elements. This environment ensures strong key protection, non-extractable secrets, and portable identity.
- Remote call of procedures hosted in network device. For example the IETF draft [RACS] defines interfaces running over a TLS stack that enable remote accesses to secure elements implementing binder and derive functions as defined by [TLS-IM]. These resources can be used over the TLS protocol by the client, which is identified by an X509 certificate. This provides hardware-backed keys, and targets Virtual Machine environments with PKI support.

2.5 Target server

Recursive Authentication creates a tree of servers, the root node is the root server, while other are referred as target servers. The client uses a crypto provider for connection to root. According to [TLS-SE] target server MAY be implemented in secure elements [ISO7816].



3 1-HOP session

A 1-HOP session establishes a session between a client and a root server.

Client	Crypto Provider	Root Server
----- binder(T, ID1)----->		
<----- PSK-Binder value ----		
----- ClientHello(ID1, PSK-Binder)----->		
<----- ServerHello -----		
Compute DHE		
----- derive(DHE, ID1) ---->		
<----- Handshake Secret-----		
<----- Server Encrypted Extensions -----		
<----- Server Finished -----		
----- Client Finished ----->		
===== Secure AEAD Channel Established =====		

4 2-HOPs session

Client	Crypto Provider	Root Server	Target Server
-- binder(T1, ID1)--->			
<--PSK-Binder value--			
---ClientHello(ID1, PSK-Binder)----->			
<--ServerHello-----			
---derive(DHE1, ID1)->			
<--HS1 value-----			
<--Server Encrypted Extensions-----			
<--Server Finished-----			
---Client Finished----->			
===== TLS Channel Established =====			
---binder(T2, ID2)----->			
<--PSK-Binder value -----			
---ClientHello(ID2, PSK-Binder)----->			
<--ServerHello-----			
---derive(DHE2, ID2)----->			
<--HS2 Value-----			
<--Close Session----->			
<--Server Encrypted Extensions-----			
<--Server Finished-----			
---Client Finished----->			
===== TLS Channel Established =====			

A 2-HOPs session establishes a session with a target server. The root server typically acts as a gateway that checks the client rights.

An example of 2-HOPs client, is available at [TLSSE-Client].

5 n-HOPs session

Client establishes a first session 1 with root server, using a crypto provider that computes $\text{binder}(\text{ID}_1, \text{T}_1)$ and $\text{derive}(\text{ID}_1, \text{DHE}_1)$.

Client establishes a session 2 with target 1 server, using $\text{binder}(\text{ID}_2, \text{T}_2)$ and $\text{derive}(\text{ID}_2, \text{DHE}_2,)$ provided par root server. Client closes session with root server.

Client establishes a session $k+1$ with target k server, using $\text{binder}(\text{ID}_{k+1}, \text{T}_{k+1})$ and $\text{derive}(\text{ID}_{k+1}, \text{DHE}_{k+1})$ provided par target $k-1$ server. Client closes session with target $k-1$.

Client establishes a session n with target $n-1$ server, using $\text{binder}(\text{ID}_n, \text{T}_n)$ and $\text{derive}(\text{ID}_n, \text{DHE}_n)$ provided par target $n-2$ server. Client closes session with target $n-2$.

At depth k in the tree, ID_k can be expressed with k indexes:
 $\text{ID}_k = \text{ID}(1, \text{I}_2, \dots, \text{I}_k)$,
where I_j (j between 2 and k) indicates the child rank of node $j-1$, between 1 and number of child.

6 IANA Considerations

This draft does not require any action from IANA.

7 Security Considerations

Recursive Authentication MAY be realized with classical software TLS-PSK server, but in these platforms PSK is exposed in software memory, and cloning is possible.

Recursive Authentication within TLS-SE servers has the followings security benefits :

- PSK Protection :PSKs remain protected inside secure elements and MUST NOT be exposed to clients.

- Delegated Authentication: Authentication capabilities are delegated through controlled cryptographic procedures rather than through direct key distribution.

TLS for Secure Element Recursive Authentication

March 2026

- Compartmentalization: Each TLS-SE server manages its own PSKs and delegation policies. Compromise of one server does not automatically compromise unrelated servers.
- Revocation: Removing a server from the authentication chain prevents further delegation to downstream servers.

8 References

8.1 Normative References

- [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/info/rfc2119>>
- [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/info/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/info/rfc8446>.
- [RFC9257] HOUSLEY, R., HOYLAND, J., SETHI, M., et al. Rfc 9257: Guidance for external pre-shared key (psk) usage in tls. 2022.
- [ISO7816] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts", The International Organization for Standardization (ISO).

8.2 Informative References

- [TLS-IM] "Identity Module for TLS Version 1.3", draft-urien-tls-im-10.txt, January 2024.
- [RACS] "Remote APDU Call Secure (RACS)", draft-urien-core-racs-19.txt, February 2024
- [TLS-SE] "Secure Element for TLS Version 1.3", draft-urien-tls-se-08, December 2024
- [TLSSE-Client] "2-Hops Recursive Authentication Client", <https://github.com/purien/pnhsm>

9 Authors' Addresses

Pascal Urien
Telecom Paris - Ethertrust
19 place Marguerite Perey
91120 Palaiseau
France
Email: Pascal.Urien@telecom-paris.fr