RFC 7170
TEAP v1
Errata Fixes

Aggregated by Oleg Pekar, IETF 108, 31 July 2020

Contributors (alphabetical)

- Alan DeKok
- Eliot Lear
- Jouni Malinen
- Oleg Pekar
- Joseph Salowey
- Jorge Vergara

TLS 1.3

TEAP over TLS 1.3 is out of scope of this errata.

Should be updated by “TLS-based EAP types and TLS 1.3”

https://tools.ietf.org/html/draft-ietf-emu-tls-eap-types-01

Errata ID: 5127, 5128 – KDF calls ambiguity

Problem: calls to TLS-PRF function don't correspond to TLS-PRF definition

Solution: RFC 5246 (TLS 1.2) defines TLS-PRF(secret, label, seed) = P_<hash>(secret, label | seed)

Sections:
5.2. Intermediate Compound Key Derivations
5.4. EAP Master Session Key Generation

- RFC 5295 defines USRK = KDF(EMSK, key label | 0x00 | optional data | length)

  where length is a 2-octet unsigned integer in network byte order

- Align all TEAP key derivation functions with RFC 5295 where the length is included into KDF input:

  Define new TEAP-PRF (secret, key label, optional data, length) = TLS-PRF(secret, key label | 0x00 | optional data, length) =

  P_<hash>(secret, key label | 0x00 | optional data | length) of TLS 1.2 that generates 'length' octets of output

  where a single byte 0x00 is used for no optional data, length is a 2-octet unsigned integer in network byte order

- Then we have the following functions in TEAP RFC:
- IMSK = First 32 octets of TEAP-PRF(EMSK, "TEAPbindkey@ietf.org", 64) =

  TLS-PRF(EMSK, "TEAPbindkey@ietf.org" | 0x00 | 0x00 | 0x00 | 0x40)

  Note: we want to generate 64 octets with included length value '64' and take first 32 octets. Per RFC 5295, USRKs MUST be at least 64 octets in length.

- IMCK[j] = TEAP-PRF(S-IMCK[j-1], "Inner Methods Compound Keys", IMSK[j], 60) =

  TLS-PRF(S-IMCK[j-1], "Inner Methods Compound Keys" | 0x00 | IMSK[j] | 0x00 | 0x3C)

- MSK = TEAP-PRF(S-IMCK[j], "Session Key Generating Function", 64) =

  TLS-PRF(S-IMCK[j], "Session Key Generating Function" | 0x00 | 0x00 | 0x00 | 0x40)

- EMSK = TEAP-PRF(S-IMCK[j], "Extended Session Key Generating Function", 64) =

  TLS-PRF(S-IMCK[j], "Extended Session Key Generating Function" | 0x00 | 0x00 | 0x00 | 0x40)

Errata ID: 5765 – Make Authority-ID TLV optional

Problem: Authority-ID TLV is mandatory that contradicts to outer TLVs requirement to be optional

Solution: Per “4.3.1. Outer TLVs”, Outer TLVs MUST be marked as optional and Authority-ID TLV is always an outer TLV.

```
4.2.2.  Authority-ID TLV

 0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |M|R|          TLV Type           |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               ID...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+


   M

      ~~Mandatory, set to one (1)~~

      0 (Optional)


   R

      Reserved, set to zero (0)
```

Errata ID: 5767, 5845 – Send Intermediate-Result TLV upon completion of each EAP **authentication** inner method

Problem: some places don't clarify when to send Intermediate-Result TLV with regards to inner EAP method type and the number of inner methods

Solution: 3.3.1. EAP Sequences

~~If more than one method is going to be executed in the tunnel, then upon method completion,~~ Upon completion of each EAP authentication method in the tunnel, the server MUST send an Intermediate-Result TLV indicating the result […] If the result indicates success, the Intermediate-Result TLV MUST be accompanied by a Crypto-Binding TLV.

EAP authentication method is EAP type 4 or greater.

3.3.3. Protected Termination and Acknowledged Result Indication

The Crypto-Binding TLV and Intermediate-Result TLV MUST be included to perform cryptographic binding after each successful EAP authentication method in a sequence of one or more EAP methods.

3.8.3. Server Unauthenticated Provisioning Mode

Upon successful completion of the EAP authentication method in Phase 2, the peer and server exchange a Crypto-Binding TLV to bind the inner method with the outer tunnel and ensure that a man-in-the-middle attack has not been attempted.

4.2.11. Intermediate-Result TLV

The Intermediate-Result TLV provides support for acknowledged intermediate Success and Failure messages ~~between multiple inner EAP methods within EAP.~~ upon completion of each inner EAP authentication method.

4.2.13. Crypto-Binding TLV

It MUST be included with the Intermediate-Result TLV to perform cryptographic binding after each successful EAP authentication method in a sequence of one or more EAP methods, before proceeding with another inner EAP method.

Errata ID: 5767, 5845 – Intermediate-Result TLV/Crypto-binding TLV - clarifications on usage

Send after each inner method	Intermediate-Result TLV	Crypto-binding TLV
One inner method	Yes	Yes
Multiple inner methods	Yes	Yes
Basic password authentication	Yes	No
No inner method	No	Yes (Zero—MSK)

→ Yes (Zero—MSK)

4.2.11. An Intermediate-Result TLV indicating success MUST be accompanied by a Crypto-Binding TLV

3.3.3. A successful TEAP Phase 2 conversation MUST always end in a successful Crypto-Binding TLV and Result TLV exchange. A TEAP server may initiate the Crypto-Binding TLV and Result TLV exchange without initiating any EAP conversation in TEAP Phase 2

Problem: when basic password authentication is used together with inner EAP method in TEAP tunnel then there's an Intermediate-Result TLV after it but no Crypto-Binding TLV.
Possible solution (mixed with Errata 5844, see below), can simplify the implementation:

3.3.2. Optional Password Authentication
If the peer wishes to participate in password authentication, then it responds with a Basic-Password-Auth-Resp TLV. Upon receiving the response, the server MUST indicate~~s~~ the success or failure of the exchange using an Intermediate-Result TLV and Crypto-Binding TLV.

Errata ID: 5844 – Intermediate-Result TLV in Basic Password Authentication

Problem: missing explicit requirement to send Intermediate-Result TLV after basic password authentication

Solution:

3.3.2. Optional Password Authentication

If the peer wishes to participate in password authentication, then it responds with a Basic-Password-Auth-Resp TLV. Upon receiving the response, the server MUST indicate~~s~~ the success or failure of the exchange using an Intermediate-Result TLV.

3.6. Error Handling

3. The Intermediate-Result TLVs carry success or failure indications of the individual EAP methods or basic password authentication in TEAP Phase 2. Errors within the EAP conversation in Phase 2 are expected to be handled by individual EAP methods.

4.2.11. Intermediate-Result TLV

The Intermediate-Result TLV provides support for acknowledged intermediate Success and Failure messages between multiple inner EAP methods or basic password authentication within EAP.

4.2.13. Crypto-Binding TLV

The Crypto-Binding TLV MUST be exchanged and verified before the final Result TLV exchange, regardless of whether there is an inner EAP method or basic password authentication or not.

Errata ID: 5844 (cont.) – Intermediate-Result TLV is missing in Appendix C

Problem: missing appearance of Intermediate-Result TLV in examples of basic password authentication
Solution:

C.1. Successful Authentication

```
                          <- Crypto-Binding TLV (Request),
                             Intermediate-Result TLV (Success),
                             Result TLV (Success),
                             (Optional PAC TLV)

   Intermediate-Result-TLV (Success),
   Crypto-Binding TLV(Response),
   Result TLV (Success),
   (PAC-Acknowledgement TLV) ->
```

C.2. Failed Authentication

```
                                <- Basic-Password-Auth-Req TLV, Challenge

   Basic-Password-Auth-Resp TLV, Response with both
   username and password) ->

                                <- Intermediate-Result TLV (Failure),
                                   Result TLV (Failure)
   Intermediate-Result TLV (Failure),
   Result TLV (Failure) ->
```

Errata ID: 5768 – Compound MAC in CMK has variable length per hash algorithm

Problem 1: Compound MAC field length in Crypto-Binding TLV is defined statically as 20 octets but it depends on TLS MAC function (e.g. if it is based on SHA-1 then the length is 20 octets, SHA-256 – 32 octets etc.)

Problem 2: Missing explicit declaration of TEAP method type octet in Crypto-Binding TLV

Solution:

4.2.13. Crypto-Binding TLV

EMSK Compound MAC

The EMSK Compound MAC field ~~is 20 octets~~. This can be the Server MAC (B1_MAC) or the Client MAC (B2_MAC). The computation of the MAC is described in Section 5.3.

MSK Compound MAC

The MSK Compound MAC field ~~is 20 octets~~. This can be the Server MAC (B1_MAC) or the Client MAC (B2_MAC). The computation of the MAC is described in Section 5.3.

The length in octets of each EMSK and MSK Compound MAC fields is equal to MAC function output length in octets.

5.3. Computing the Compound MAC

~~2 The EAP Type sent by the other party in the first TEAP message.~~

2 A single octet 0x37 (TEAP type). It helps in protection against cross protocol attacks if Crypto-Binding TLV was reused.

Errata ID: 5770 – Keep both IMCK[j] derived from inner method MSK and EMSK

Problem: the description of key derivation based on inner method MSK and EMSK need to be clarified

Solution: 5.2. Intermediate Compound Key Derivations

For each inner method and if no inner method was conducted IMSK, IMCK, S-IMCK and CMK are generated. IMSK can be derived from inner method MSK or EMSK, or Zero-MSK if it was no inner method. Since the sender doesn't know before the receiver's response which IMSK is negotiated according to sender and receiver common capabilities, the sender needs to calculate all four keys derived from both MSK and EMSK.

[Note: the schema is from the original RFC and not yet affected by the other errata fixes]

S-IMCK[0] = session_key_seed

For j = 1 to n-1 do

IMCK-MSK[j] = TLS-PRF(S-IMCK[j-1], "Inner Methods Compound Keys", IMSK-MSK[j], 60)

S-IMCK-MSK[j] = first 40 octets of IMCK-MSK[j]

CMK-MSK[j] = last 20 octets of IMCK-MSK[j]

IMCK-EMSK[j] = TLS-PRF(S-IMCK[j-1], "Inner Methods Compound Keys", IMSK-EMSK[j], 60)

S-IMCK-EMSK[j] = first 40 octets of IMCK-EMSK[j]

CMK-EMSK[j] = last 20 octets of IMCK-EMSK[j]

5.3. Computing the Compound MAC

The Compound MAC computation is as follows:

CMK-MSK = CMK-MSK[j]

Compound-MAC-MSK = MAC(CMK-MSK, BUFFER)

CMK-EMSK = CMK-EMSK[j]

Compound-MAC-EMSK = MAC(CMK-EMSK, BUFFER)

Errata ID: 5775 – Explicitly specify Zero-MSK

Problem 1: missing description on key generation when no inner method was conducted

Problem 2: missing explicit declaration of Zero-MSK usage in Crypto-Binding TLV

Solution:

4.2.13. Crypto-Binding TLV

Flags

The Flags field is four bits. Defined values include

0 Only Zero-MSK Compound MAC is present at MSK Compound MAC field

1 EMSK Compound MAC is present

2 MSK Compound MAC is present

3 Both EMSK and MSK Compound MAC are present

5.2. Intermediate Compound Key Derivations

If the ith inner method does not generate an EMSK or MSK, then Zero-MSK is used as IMSKi is set to zero (e.g., IMSKi = 32 octets of 0x00s). If no inner method was conducted the derivation cycle below is done once and Zero-MSK is used as IMSKi.

If an inner method specifications implies generation of MSK or EMSK – Zero-MSK MUST not be used. Using Zero-MSK when the inner method generates MSK or EMSK MUST be handled as an invalid Crypto-Binding TLV with a fatal error.

Errata ID: 6157 – Request-Action TLV can be sent either by the server or the peer

Problem: Request-Action TLV processing is bound to the server while can be done by the peer also

Solution:

4.2.9. Request-Action TLV

Status

The Status field is one octet. This indicates the result if the ~~server does not process the action requested by the peer~~ party who receives this TLV does not process the action.