EMU
Internet-Draft
Intended status: Standards Track
Expires: May 20, 2021

M. Chen, Ed.
Li. Su
China Mobile
H. Wang
Huawei International Pte. Ltd.

November 16, 2020

Use Identity as Raw Public Key in EAP-TLS
draft-chen-emu-eap-tls-ibs-01

Abstract

This document specifies the use of identity as a raw public key in EAP-TLS both for TLS1.2 and TLS1.3, EAP-TLS for TLS1.2 is defined in RFC 5216 and EAP-TLS for TLS1.3 is defined in the draft draft-ietf-tls-dtls13. The protocol procedures of EAP-TLS-IBS will consistent with EAP-TLS's interactive process, Identity-based signature will be extended to support EAP-TLS's signature algorithms.

Table of Contents

1. Introduction

The Extensible Authentication Protocol(EAP) defined in RFC 3748[RFC3748] can provide support for multiple authentication methods. Transport Layer Security(TLS) provides for mutual authentication, integrity-protected ciphersuite negotiation, and exchange between two endpoints. The EAP-TLS defined in RFC 5216 [RFC5216] which combines EAP and TLS that apply EAP method to load TLS procedures.

Traditionally, TLS client and server public keys are obtained in PKIX containers in-band as part of the TLS handshake procedure and are validated using trust anchors based on a PKIX certification authority (CA). But there is another method, Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) are defined in RFC 7250[RFC7250], the document defines two TLS extensions client_certificate_type and server_certificate_type, which can be used as part of an extended TLS handshake when raw public keys are used. In the draft draft-ietf-emu-eap-tls13 reads certificates can be of any type supported by TLS including raw public keys. In

RFC7250[RFC7250] it assuming that an out-of-band mechanism is used to bind the public key to the entity presenting the key.

Digital signatures provide the functions of Sender reliability and Message integrity. A chain of trust for such signatures is usually provided by certificates, but in low-bandwidth and resource-constrained environments, the use of certificates might be undesirable. In comparison with the original certificate, the raw public key is fairly small. This document describes a signature algorithm using identity as a raw public key in EAP-TLS, instead of transmitting a full certificate in the EAP-TLS message, only public keys are exchanged between client and server, also known as EAP-TLS-IBS.

With the existing raw public key scheme, a public key and identity mapping table is required at server. This table usually established with offline method and may require additional efforts for establishment and maintenance, especially when the number of devices are huge. On the other hand, with IBS signature algorithm, it not only can take the advantage of raw public key, but also eliminates the efforts for the mapping table establishment and maintenance at the server side. Instead, a small table for CRL is enough for exclude revoked identity from accessing the network. A number of IBE and IBS algorithms have been standardized, such as ECCSI defined in RFC 6507[RFC6507].

IBC was first proposed by Adi Shamir in 1984. For an IBC system, a Key Management System (KMS) is required to generate keys for devices. The KMS choose its KMS Secret Authentication Key(KSAK) as the root of trust. A public parameter, KMS Public Authentication Key (KPAK) is derived from this secrete key and is used by others in verifying the signature. The signatures are generated by an entity with private keys obtained from the KMS. KMS is a trusted third party, users or devices can obtain private key using their identities from KMS. In IBS the private key is also known as Secret Signing Key(SSK). A sender can sign a message using SSK. The receiver can verify the signature with sender's identity and the KPAK.

This document is organized as follows: the second section defines the terms used in the text; the third section gives a brief overview of the IBS algorithms; the fourth section presents the example message flow and message format for EAP-TLS-IBS and follows by security consideration and IANA cosideration etc.

2. Terminology

The readers should be familiar with the terms defined in.

In addition, this document makes use of the following terms:

IBC: Identity-Based Cryptograph, it is an asymmetric public key cryptosystem.

IBS: Identity-based Signature, such as ECCSI.

PKI: Public Key Infrastructure, an infrastructure built with a public-key mechanism.

authenticator: The entity initiating EAP authentication.

peer: The entity that responds to the authenticator.

backend authenticator server: A backend authentication server is an entity that provides an authentication service to an authenticator. When used, this server typically executes EAP methods for the authenticator.

EAP server: The entity that terminates the EAP authentication method with the peer. In the case where no backend authentication server is used, the EAP server is part of the authenticator. In the case where the authenticator operates in pass-through mode, the EAP server is located on the backend authentication server.

3. Structure of the Raw Public Key Extension

To support the negotiation of using raw public between client and server, a new certificate structure is defined in RFC 7250[RFC7250]. It is used by the client and server in the hello messages to indicate the types of certificates supported by each side. When RawPublicKey type is selected for authentication, SubjectPublicKeyInfo which is a data structure is used to carry the raw public key and its cryptographic algorithm.

The SubjectPublicKeyInfo structure is defined in Section 4.1 of RFC 5280 [PKIX][RFC5280] and not only contains the raw keys, such as the public exponent and the modulus of an RSA public key, but also an algorithm identifier. The algorithm identifier can also include parameters. The structure of SubjectPublicKeyInfo is shown in Figure 1:

```
SubjectPublicKeyInfo  ::=  SEQUENCE  {
          algorithm              AlgorithmIdentifier,
          subjectPublicKey       BIT STRING  }

     AlgorithmIdentifier   ::=  SEQUENCE  {
          algorithm              OBJECT IDENTIFIER,
          parameters             ANY DEFINED BY algorithm OPTIONAL }
```

Figure 1: SubjectPublicKeyInfo ASN.1 Structure

The algorithms identifiers are Object Identifier(OIDs), AlgorithmIdentifier is also data structure with two fields, OID represent the cryptographic algorithm used with raw public key, such as ECCSI, parameters are the necessary parameters associated with the algorithm.

In the case of IBS algorithm, the User's identity is the raw public key which can be represented by "subjectPublicKey", when ECCSI is used as the Identity-based signature algorithm, then "algorithm" is for ECCSI, and "parameters" is the parameters needed in ECCSI.

So far, IBS has the following four algorithms, the following table is the corresponding table of Key type and OID.

Key Type	Document	OID
ISO/IEC 14888-3 IBS-1	ISO/IEC 14888-3: IBS-1 mechanism	1.0.14888.3.0.7
ISO/IEC 14888-3 IBS-2	ISO/IEC 14888-3: IBS-2 mechanism	1.0.14888.3.0.8
ISO/IEC 14888-3 ChineseIBS(SM9)	ISO/IEC 14888-3: ChineseIBS mechanism	1.2.156.10197.1.302.1
Elliptic Curve-Based Signatureless For Identitiy-based Encryption (ECCSI)	Section 5.2 in RFC 6507	1.3.6.1.5.5.7.6.29

Table 1: Algorithm Object Identifiers

In the draft draft-wang-tls-raw-public-key-with-ibc, there extend signature scheme with IBS algorithm which indicated in the client's "signature_algorithms" extension. The SignatureScheme data structure also keep pace with the section 4.

4. EAP-TLS using raw public keys

This section describes EAP-TLS-IBS both in the case of TLS1.2 and TLS1.3, each section contains EAP-TLS and EAP-TLS using raw public keys full message authentication, and finally give the example when using IBS.

4.1. EAP TLS1.2 Client and Server Handshake Behavior

4.1.1. raw public keys TLS exchange

As described in [RFC7250][RFC7250], the document intrudoces the use of raw public keys in TLS/DTLS, the basic raw public key TLS exchange will appear as follows, Figure 2 shows the client_certificate_type and server_certificate_type extensions added to the client and server hello messages. An extension type MUST NOT appear in the ServerHello unless the same extension type appeared in the corresponding ClientHello, defined in RFC5246[RFC5246].

The server_certificate_type extension in the client hello indicates the types of certificates the client is able to process when provided by the server in a subsequent certificate payload.

The client_certificate_type and server_certificate_type extensions sent in the client hello each carry a list of supported certificate types, sorted by client preference. When the client supports only one certificate type, it is a list containing a single element. Many types of certificates can be used, such as RawPublicKey, X.509 and OpenPGP.

Figure 2: Basic Raw Public Key TLS Exchange

4.1.2. EAP-TLS handshake in TLS1.2

As described in [RFC3748] [RFC3748], the EAP-TLS conversation will typically begin with the authenticator and the peer negotiating EAP. The authenticator will then typically send an EAP-Request/Identity packet to the peer, and the peer will respond with an EAP-Response/ Identity packet to the authenticator, containing the peer's user-Id. The authenticator MAY act as a pass-through device, with the EAP packets received from the peer being encapsulated for transmission to a backend authentication server.

In the case where the EAP-TLS mutual authentication is successful, defined in RFC5216 [RFC5216], the conversation will appear as follows:

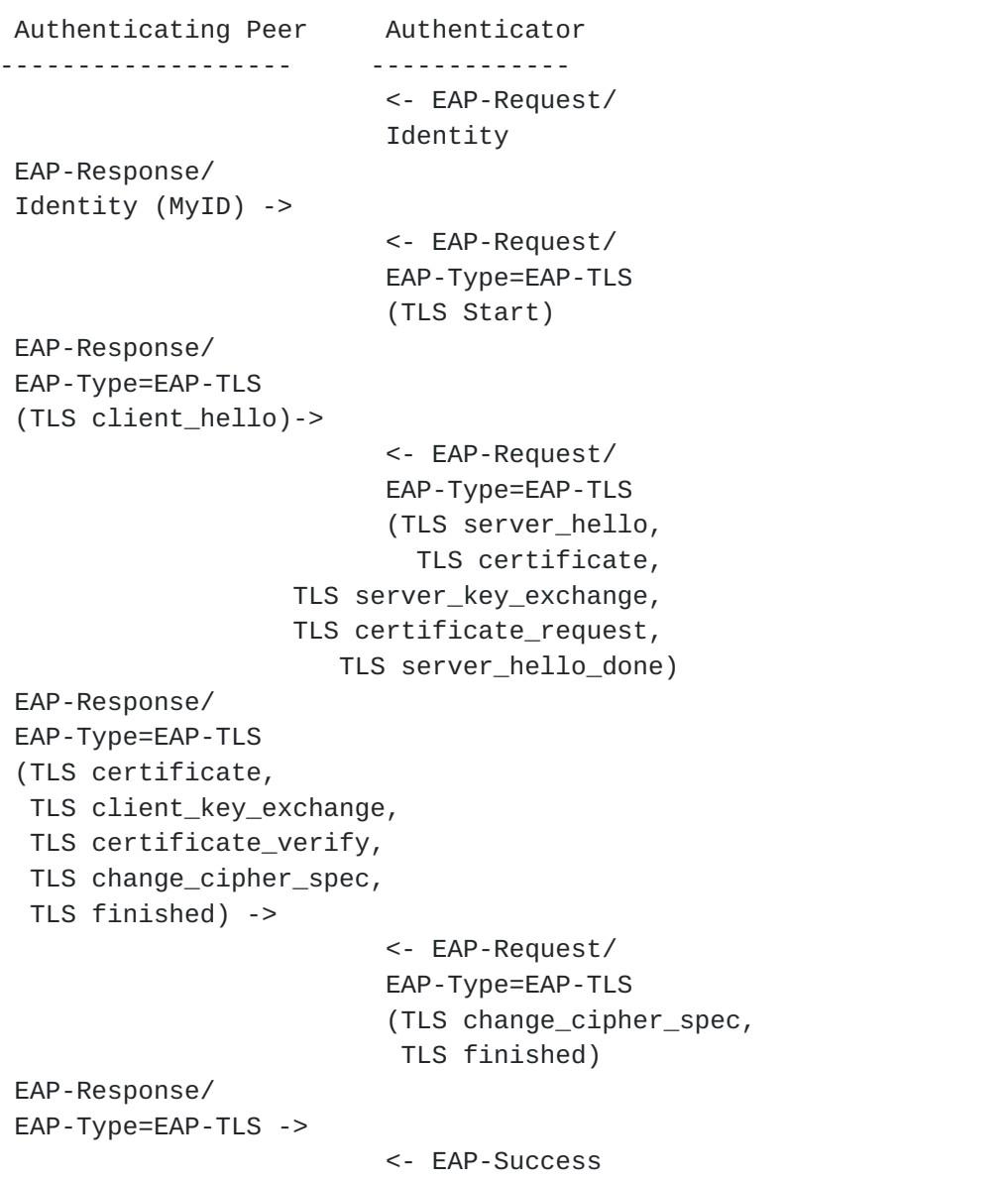


Figure 3: EAP-TLS authentication procedure with TLS1.2 handshake

4.1.3. raw public keys EAP-TLS exchange

This section describes EAP-TLS extend using raw public keys, the procedures is as follows, In the discussion, we will use the term "EAP server" to denote the ultimate endpoint conversing with the peer.

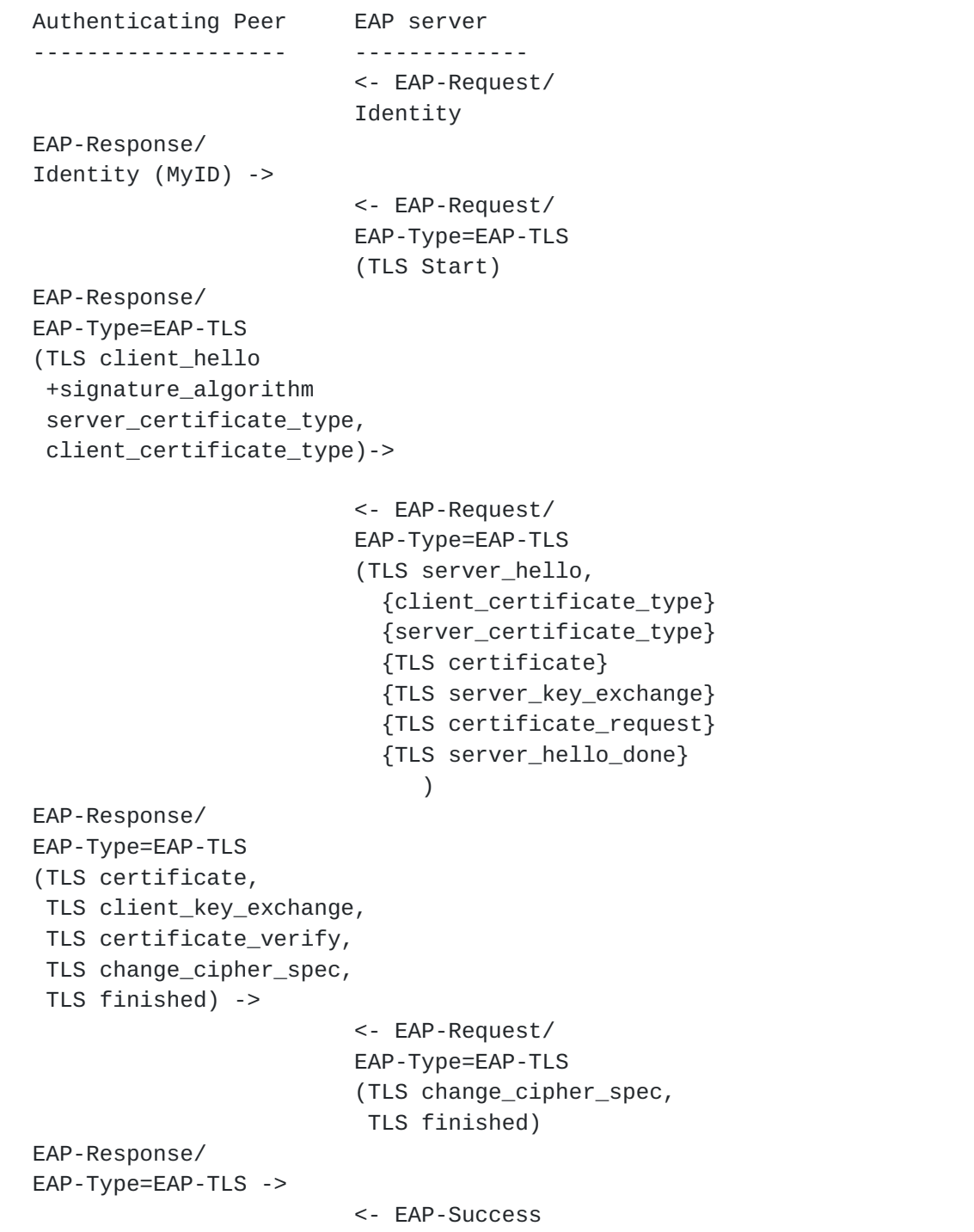


Figure 4: EAP-TLS extend raw public keys authentication procedure with TLS1.2 handshake

4.1.4. EAP-TLS1.2-IBS example

In this example, both the TLS client and server use ECCSI for authentication, and they are restricted in that they can only process ECCSI signature algorithm. As a result, the TLS client sets both the server_certificate_type and the client_certificate_type extensions to be raw public key; in addition, the client sets the signature algorithm in the client hello message to be eccsi_sha256.

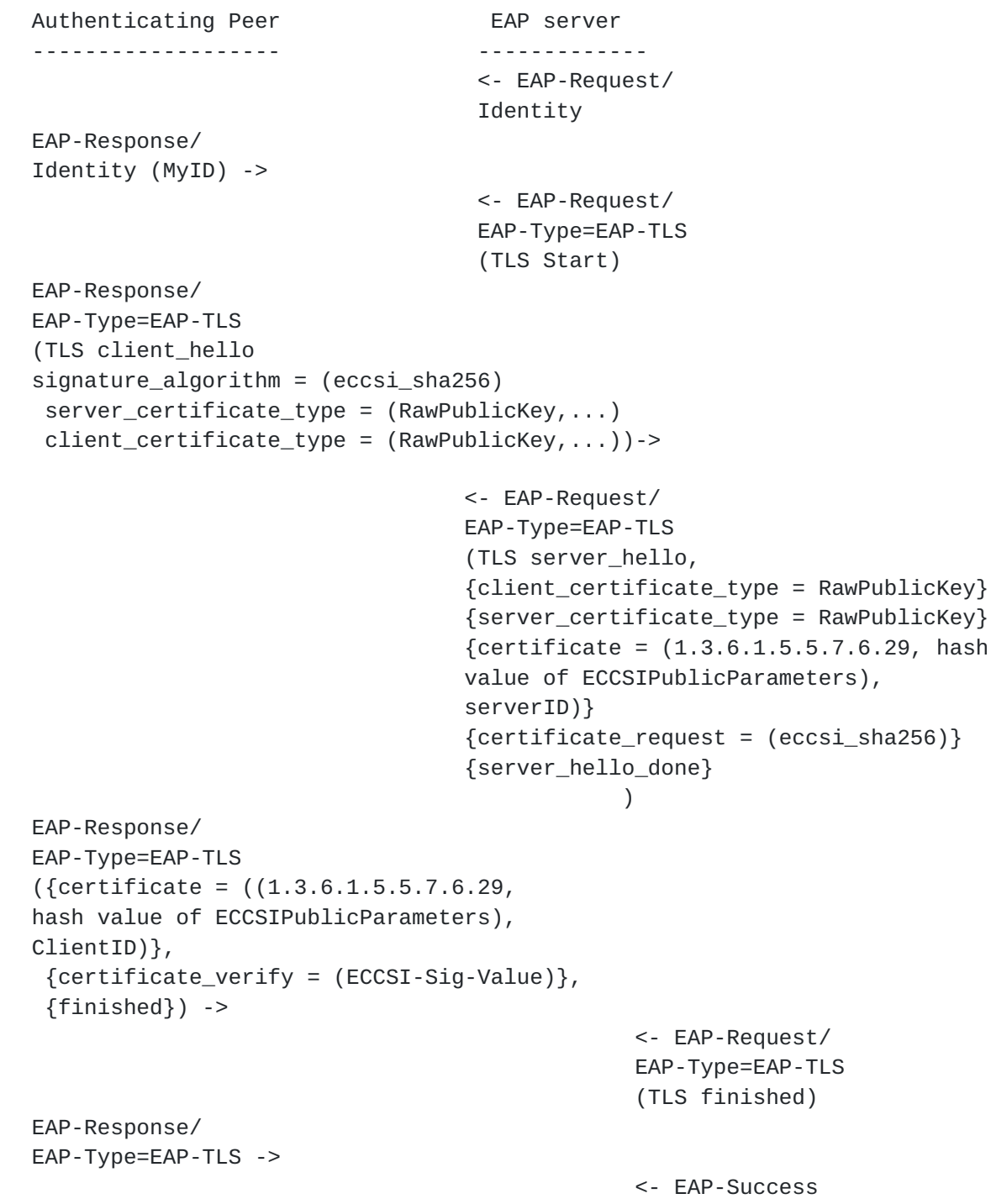


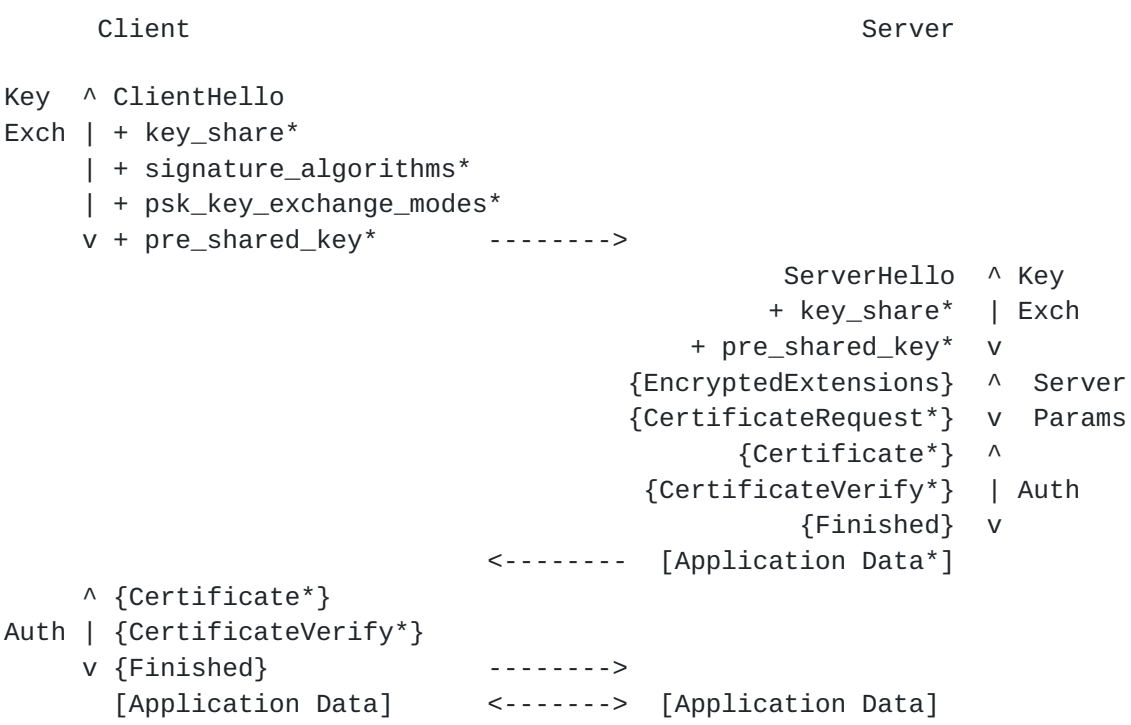
Figure 5: EAP-TLS1.2-IBS example

4.2. EAP TLS1.3 Client and Server Handshake Behavior

TLS1.3 defined in RFC8446, as TLS 1.3 is not directly compatible with previous versions, all versions of TLS incorporate a versioning mechanism which allows clients and servers to interoperably negotiate a common version if one is supported by both peers. when make the discussion on EAP-TLS using raw public keys we also make a different with TLS1.2, This section is for EAP-TLS1.3 handshake behavior using raw public keys and give example for EAP-TLS-IBS.

4.2.1. TLS1.3 handshake

TLS1.3 is more secure than TLS1.2 in preventing eavesdropping, tampering, and message forgery. The handshake can be thought of having three phases: Key Exchange, Server Parameters and Authentication. The message flow for full TLS handshake is as follows.



+ Indicates noteworthy extensions sent in the previously noted message.

* Indicates optional or situation-dependent messages/extensions that are not always sent.

{} Indicates messages protected using keys derived from a [sender]_handshake_traffic_secret.

[] Indicates messages protected using keys derived from [sender]_application_traffic_secret_N.

Figure 6: Message Flow for Full TLS1.3 Handshake

4.2.2. EAP-TLS1.3 handshake procedure

EAP-TLS mutual authentication in the case of TLS1.3. defined in the draft-ietf-emu-eap-tls13. TLS 1.3 provides significantly improved security, privacy, and reduced latency when compared to earlier versions of TLS. EAP-TLS with TLS 1.3 further improves security and privacy by mandating use of privacy and revocation checking.

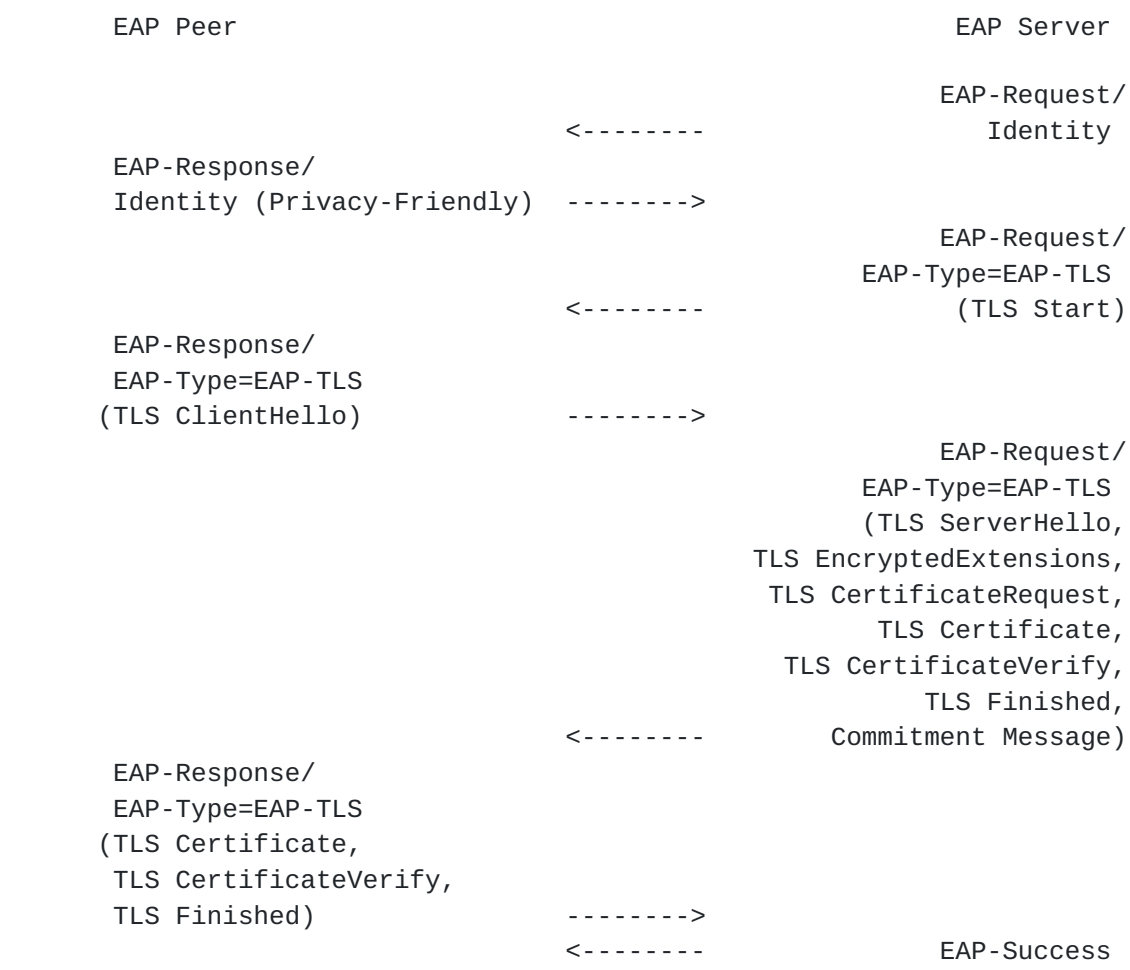


Figure 7: EAP-TLS mutual authentication with TLS1.3 handshake

4.2.3. raw public keys EAP-TLS1.3 exchange

This section describes EAP-TLS1.3 extend using raw public keys, the procedures is as follows, both client and server have the extension "key_share", the "key_share" extension contains the endpoint's cryptographic parameters. the "signature_algorithm" extension contains the signature algorithm and hash algorithms the client and server can support for the new signature algorithms specific to the IBS algorithms. When IBS is chosen as signature algorithm, the server need to indicated the required IBS signature algorithms int the signature_algorithm extension within the CertificateRequest.

```
     Authenticating Peer     EAP server
     -------------------     -------------
                             <- EAP-Request/
                             Identity
     EAP-Response/
     Identity (MyID) ->
                             <- EAP-Request/
                             EAP-Type=EAP-TLS
                             (TLS Start)
     EAP-Response/
     EAP-Type=EAP-TLS
     (TLS client_hello
      +key_share
      +signature_algorithm
      server_certificate_type,
      client_certificate_type)->

                             <- EAP-Request/
                             EAP-Type=EAP-TLS
                             (TLS server_hello,
                               +key_share
                               {EncryptedExtensions}
                               {client_certificate_type}
                               {server_certificate_type}
                               {certificate}
                               {CertificateVerify}
                               {certificateRequest}
                               {Finished}
                               [Application Data]
                                  )
     EAP-Response/
     EAP-Type=EAP-TLS
     ({certificate}
      {CertificateVerify}
      {Finished}
      [Application Data]) ->
                             <- EAP-Success
                                  [Application Data]
```

Figure 8: EAP-TLS1.3 authentication procedure with raw public keys

4.2.4. EAP-TLS1.3-IBS example

When the EAP server receives the client hello, it processes the message. Since it has an ECCSI raw public key from the KMS, it indicates that it agrees to use ECCSI and provides an ECCSI key by placing the SubjectPublicKeyInfo structure into the Certificate

payload back to the client, including the OID, the identity of server, ServerID, which is the public key of server also, and hash value of KMS public parameters. The client_certificate_type indicates that the TLS server accepts raw public key. The TLS server demands client authentication, and therefore includes a certificate_request, which requires the client to use eccsi_sha256 for signature. A signature value based on the eccsi_sha256 algorithm is carried in the CertificateVerify. The client, which has an ECCSI key, returns its ECCSI public key in the Certificate payload to the server, which includes an OID for the ECCSI signature. The example of EAP-TLS1.3-IBS is as follows:

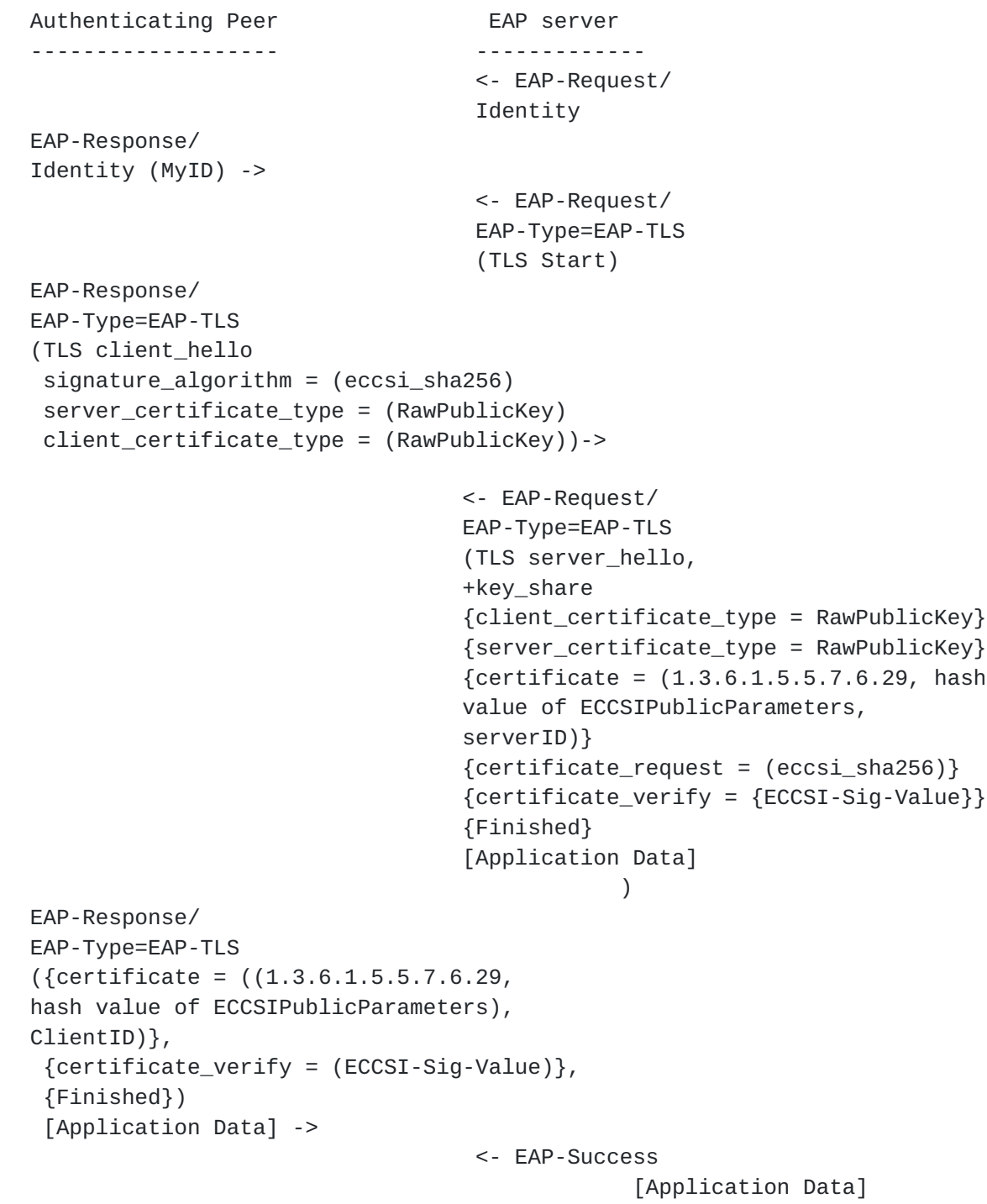


Figure 9: EAP-TLS1.3-IBS example

5. Security Considerations

TBD

6. IANA Considerations

This document registers the following item in the "Method Types" registry under the "extensible Authentication Protocol(EAP) Registry" heading.

Value	Description
TBD	EAP-TLS-IBS

7. Acknowledgement

TBD

8. References

8.1. Normative References

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <https://www.rfc-editor.org/info/rfc5216>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[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/info/rfc5280>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <https://www.rfc-editor.org/info/rfc7250>.

8.2. Informative references

[RFC6507] Groves, M., "Elliptic Curve-Based Certificateless Signatures for Identity-Based Encryption (ECCSI)", RFC 6507, DOI 10.17487/RFC6507, February 2012, <https://www.rfc-editor.org/info/rfc6507>.

Authors' Addresses

Meiling Chen (editor)
China Mobile
32, Xuanwumen West
BeiJing, BeiJing 100053
China

Email:
chenmeiling@chinamobile.com

Li Su
China Mobile

32, Xuanwumen West

BeiJing

100053

China

Email:
suli@chinamobile.com

Haiguang Wang
Huawei International Pte. Ltd.
11 North Buona Vista Dr, #17-08
Singapore 138589
SG

Phone: +65 6825 4200
Email: wang.haiguang1@huawei.com