eIDAS Cryptographic Requirements for the Interoperability Framework

TLS and SAML

Version 1.2

31 August 2019

eIDAS Technical Specifications

Document created by:

eIDAS eID Technical Subgroup

Document adopted by:

eIDAS Cooperation Network

DOCUMENT HISTORY

Version	Date	Adopted by	Short Description of Changes
1.0	26 January 2016	eIDAS Cooperation Network	/
1.1	16 December 2016	eIDAS Cooperation Network	/
1.2	27 September 2019	eIDAS Cooperation Network	• Use of non-notified eID schemes • Identification of Relying parties • Fixes in attributes profiles and cryptographic requirements • Improvements in metadata handling

Table of Contents

1. Introduction

Within the eIDAS Interoperability Framework [1], communication between eIDAS nodes (i.e. eIDAS-Services and eIDAS-Connectors) is performed via the citizen's browser. Here, the content of the communication between eIDAS nodes is performed using cryptographically protected SAML messages. To secure the transport layer of this communication between these components and the citizen's browser, TLS is used.

This document specifies cryptographic requirements for the protection of the SAML communication as well as on the usage of TLS within this communication.

1.1. Key Words

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119. The key word "CONDITIONAL" is to be interpreted as follows:

CONDITIONAL: The usage of an item is dependent on the usage of other items. It is therefore further qualified under which conditions the item is REQUIRED or RECOMMENDED.

.

2. Requirements for TLS

Transport Layer Security (TLS) [2], formerly also known as Secure Socket Layer (SSL), is a protocol to protect the communication over the Internet, e.g. connection via HTTP (HTTPS). During TLS, a secure connection between the *Client* and the *Server* is negotiated.

As part of the establishment (*handshake*) of a TLS session, both parties negotiate the encryption and authentication algorithms (*cipher suite*) and the keys to be used during the session.

Within the context of the cross-border setting of the interoperability framework, the client is usually the citizen's browser, while the server is an eIDAS node. Here, the server has to authenticate to the client based on X.509 certificates.

In the following, minimal cryptographic requirements for the TLS building blocks are specified, i.e. the version of the TLS protocol, the cipher suites and crypto primitives as domain parameters, key lengths and certificates to be used by eIDAS nodes.

The security requirements given in following aim to provide a security level of at least 100 bit.

2.1. TLS VERSION

eIDAS nodes MUST use TLS 1.2. Prior TLS versions MUST NOT be accepted.

2.2. CIPHER SUITES

eIDAS nodes MUST only use cipher suites that provide perfect forward secrecy. It is RECOMMENDED to use a cipher suite from the following table for the TLS handshake.

<table>
<tr><th></th><th>Key agreement and authentication mechanisms</th><th></th><th>Encryption</th><th>Mode of operation</th><th>Hash</th></tr>
<tr><td rowspan="8">TLS_</td><td rowspan="2">ECDHE_ECDSA_</td><td rowspan="2">WITH_</td><td>AES_128_</td><td>CBC_
GCM_</td><td>SHA256</td></tr>
<tr><td>AES_256_</td><td>CBC_
GCM_</td><td>SHA384</td></tr>
<tr><td rowspan="2">ECDHE_RSA_</td><td rowspan="2">WITH_</td><td>AES_128_</td><td>CBC_
GCM_</td><td>SHA256</td></tr>
<tr><td>AES_256_</td><td>CBC_
GCM_</td><td>SHA384</td></tr>
<tr><td rowspan="3">DHE_RSA_</td><td rowspan="3">WITH_</td><td>AES_128_</td><td>CBC_
GCM_</td><td>SHA256</td></tr>
<tr><td rowspan="2">AES_256_</td><td>CBC_</td><td>SHA256</td></tr>
<tr><td>GCM_</td><td>SHA384</td></tr>
</table>

Table 1: Recommended Cipher Suites

If the usage of one of these cipher suites is not possible due to restrictions of the client's browser, an eIDAS node SHOULD also accept a cipher suite of the table above with *_SHA1 instead of *_SHA-256 or *_SHA384. However, eIDAS nodes MUST prefer the cipher suites given in table 1.

Other cipher suites than the ones listed above SHALL NOT be accepted by eIDAS nodes.

2.3. Elliptic curves

If Elliptic Curve Cryptography (ECC) is used for TLS, named curves MUST be used. It is RECOMMENDED to support the following named curves and to use one of them (if supported by the browser):

- BrainpoolP256r1, BrainpoolP384r1, BrainpoolP512r1 (cf. [3]);
- NIST Curve P-224, NIST Curve P-256, NIST Curve P-384, NIST Curve P-521 (cf. [4]).

Elliptic curves with a key length less than 224 bit MUST NOT be accepted.

2.4. Certificates

Until 2017, eIDAS nodes MUST use extended validation certificates or qualified website certificates[1] for TLS. As of 2018, newly issued TLS certificates for eIDAS nodes MUST be qualified.

2.5. Domain parameters, keys and random numbers

eIDAS nodes SHALL use the following key lengths for ephemeral keys during the TLS handshake:

Protocol	*Minimal Key length*
ECDH	256
DH	2048

Table 2: Key lengths for TLS

To provide acceptable security of a TLS session, it is REQUIRED that random numbers (for nonces or ephemeral keys) to be used within the TLS handshake are generated with cryptographically secure random number generators that provide sufficient entropy (according to the security level of 100 bits).

2.6. Additional Requirements and Recommendations

If the client indicates support for signature algorithms with a SHA-2 hash function for the signature algorithm to be used, eIDAS nodes SHALL use this hash function for signatures within the handshake (and not SHA-1).

[1] Cf. Article 45 of the eIDAS Regulation

Furthermore, the following recommendations for the deployment of TLS are given:

- TLS compression SHOULD NOT be used.
- The heartbeat extension SHOULD NOT be used (cf. [5]).
- If a CBC-based cipher suite is used, it is RECOMMENDED to first encrypt and then authenticate the data to be transmitted. Hence, eIDAS nodes SHOULD support and use the Enc-then-MAC extension according to [6].
- Session Renegotiation SHOULD NOT be used.
- eIDAS nodes SHOULD NOT use a truncated HMAC (cf. truncated_hmac extension, [7]).

3. Requirements for SAML

The Security Assertion Markup Language (SAML) is a XML framework for the exchange of authentication information that is used within the eIDAS Interoperability Framework for the communication between eIDAS nodes.

SAML is used, to protect

- confidentiality of the person identification data,
- authenticity/integrity of the person identification data, and
- secure identification of communication end-points.

SAML uses XML Encryption and XML Signatures based on authentication via X.509 certificates according to [8].

Furthermore, eIDAS-Connectors and eIDAS-Proxy-Services need to provide metadata about their node. This is done via signed SAML Metadata, that has to be validated up to a published trust anchor.

In the following, we specify security requirements for the deployment of SAML that provide a security level of at least 120 bit. Besides the provision of a suitable security level, the proposal also aims to achieve interoperability between different eIDAS nodes.

3.1. General Requirements

The following rules MUST apply to the SAML communication between eIDAS nodes:

- SAML request and SAML response messages MUST be signed by the sending party.
- The signature of a SAML assertion is OPTIONAL.
- The (signed) SAML assertion within the SAML response message MUST be encrypted.

Ephemeral keys or random numbers (for nonces or generation of ephemeral keys) SHALL be used only once. It is REQUIRED that random numbers to be used within SAML are generated with cryptographically secure random number generators that provide sufficient entropy (according to the security level of 120 bits).

3.1.1. Hash functions

In XML Encryption/Signature, hash function are used for different purposes (e.g. key derivation, signatures). The following hash functions MUST be supported.

Algorithm	Minimal output length
SHA-2	256

Table 3: Hash function for SAML

Other hash functions than listed above SHALL NOT be used or accepted.

3.2. XML Encryption with SAML

To protect the confidentiality of data, a hybrid crypto system is used. The content MUST be encrypted via symmetric cryptography (Content Encryption) and the corresponding symmetric key (Session Key) MUST be randomly generated for each transmission. A static public key of the receiver MUST be used to encrypt the session key (Key Encryption).

3.2.1. Content Encryption

For content encryption, algorithms of the following list MUST be supported:

- http://www.w3.org/2009/xmlenc11#aes128-gcm
- http://www.w3.org/2009/xmlenc11#aes256-gcm

Additionally, the following algorithms MAY be supported:

- http://www.w3.org/2009/xmlenc11#aes192-gcm

Other algorithms than listed above SHALL NOT be used or accepted for content encryption.

3.2.2. Key Encryption

To encrypt the session key for content encryption, either key transport or key agreement mechanisms MUST be used. eIDAS-Services MUST support both key encryption mechanisms. In case of key transport, the session key is asymmetrically encrypted using the public key of the receivers X.509 certificate. If key agreement is applied, the sender derives a symmetric key pair by means of a ECDH key agreement using an ephemeral key pair and the static public key contained in the receivers X.509 certificate. The derived symmetric key is then used to wrap the session key.

3.2.2.1 Methods for Key Transport

The following methods and key lengths MUST be supported for key transport. One of the methods MUST be used.

Protocol/Algorithm	*Minimal key length*
http://www.w 3.org/2001/04/xmlenc#rsa-oaep-mgf1p	3072
http://www.w3.org/2009/xmlenc11#rsa-oaep	3072

Table 4: Algorithms for key transport

Other algorithms or key lengths than listed above SHALL NOT be accepted for key transport.

3.2.2.2 Methods for Key Agreement

The following methods and key lengths MUST be supported for encryption of session keys via key agreement. One of the methods MUST be used.

Protocol/Algorithm	*Minimal key length*
Key agreement	
http://www.w3.org/2009/xmlenc11#ECDH-ES	256
Key wrapping algorithm	
http://www.w3.org/2001/04/xmlenc#kw-aes128	128
http://www.w3.org/2001/04/xmlenc#kw-aes256	256

Table 5: Key Agreement Algorithms for SAML

For derivation of the symmetric key for key encryption a hash function as specified above MUST be used. The sender MUST generate a new ephemeral public key for each new message to be sent.

However, the receiver MUST support additional ephemeral data (KA-Nonce element) to be included into the derivation of the key encryption key.

3.3. SIGNATURES FOR SAML AND SAML METADATA

This section specifies the requirements for signatures of SAML assertions, SAML messages and SAML Metadata.

3.3.1. Signature Alghoritms

The following algorithms MUST be supported for the generation/verification of signatures. One of the algorithms MUST be used.

Algorithm	*Minimal key length*
RSASSA-PSS [9]	
http://www.w3.org/2007/05/xmldsig-more#sha256-rsa-MGF1 http://www.w3.org/2007/05/xmldsig-more#sha384-rsa-MGF1 http://www.w3.org/2007/05/xmldsig-more#sha512-rsa-MGF1	3072
ECDSA [9]	
http://www.w3.org/ 2001/04/xmldsig-more#ecdsa sha256 http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha384 http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha512	256

Table 6: Signature Algorithms for SAML

Within the signature algorithm, a hash function as specified above MUST be used. Other algorithms or key lengths than listed above SHALL NOT be accepted.

3.4. ELLIPTIC CURVES

If Elliptic Curve Cryptography (ECC) is used within SAML, only named curves MUST be used. It is RECOMMENDED

to support the following named curves:

- BrainpoolP256r1, BrainpoolP384r1, BrainpoolP512r1 ([10]);
- NIST Curve P-256, NIST Curve P-384, NIST Curve P-521 ([11])

Elliptic curves with a key length less than 256 bit SHALL NOT be accepted.

3.5. X.509 CERTIFICATES

3.5.1. Certificates for SAML Metadata

The algorithms and keys for the signature of certificates within the certificate chain to verify the SAML Metadata MUST fulfill the following requirements.

Algorithm	*Minimal key length*	*Minimal hash length*
RSASSA-PSS [12]	3072 Bit	256 Bit
ECDSA [13]	256 Bit	256 Bit

Table 7: Signature of Certificates

3.5.2. Certificates for SAML Communication

The authoritative references for X.509 certificates used for protection of SAML messages are the key representation elements in SAML metadata (cf. [14]). X.509 Certificates used for SAML communication SHOULD contain information on the key usage within the key usage extension. For different purposes (encryption vs signature), different certificates containing different keys SHALL be used.

References

[1] eIDAS Technical Subgroup: eIDAS Interoperability Architecture

[2] IETF: RFC 5246: T. Dierks, E. Rescorla: The Transport Layer Security (TLS) Protocol Version 1.2

[3] IETF: RFC 7027: J. Merkle, M. Lochter: Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)

[4] IETF: RFC 4492: S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, B. Moeller: Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)

[5] IETF: RFC 6520: R. Seggelmann, M. Tuexen, M. Williams, Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension, 2012

[6] IETF: RFC 7366: P. Gutmann, Encrypt-then-MAC for Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS), 2014

[7] D. Eastlake, Transport Layer Security (TLS) Extensions: Extension Definitions, 2011

[8] IETF: RFC 5280: D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

[9] IETF: RFC 6931: D. Eastlake, Additional XML Security Uniform Resource Identifiers, 2013

[10] IETF: RFC 5639: M. Lochter, J. Merkle, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation

[11] IETF: RFC 5480: S. Turner, D.Brown, K. Yiu, R. Housley, T. Polk, Elliptic Curve Cryptography Subject Public Key Information, 2009

[12] IETF: RFC 4055: J, Schaad, B. Kaliski, R. Housley, Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, 2005

[13] IETF: RFC 5758: Q. Dang, S. Santesson, K. Moriarty, D. Brown, T.Polk, Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA, 2010

[14] eIDAS Technical Subgroup: eIDAS Message Format

eIDAS Cryptographic Requirements for the Interoperability Framework - Version 1.2

2019 – 9 pages