Common Authentication Technology Next Generation F. Schmaus
Internet-Draft C. Egger
Intended status: Experimental University of Erlangen-Nuremberg
Expires: May 15, 2018 November 11, 2017

The Hashed Token SASL Mechanism
draft-schmaus-kitten-sasl-ht-03

Abstract

This document specifies the family of Hashed Token SASL mechanisms, which are meant to be used for quick re-authentication of a previous session. The Hashed Token SASL mechanism's authentication sequence consists of only one round-trip. This is achieved by the usage of short-lived, exclusively ephemeral hashed tokens. It further provides hash agility, mutual authentication and is secured by channel binding.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This specification describes the the family of Hashed Token (HT) Simple Authentication and Security Layer (SASL) [RFC4422] mechanisms. The HT mechanism is designed to be used with short-lived, exclusively ephemeral tokens, called SASL-HT tokens, and allow for quick, one round-trip, re-authentication of a previous session.

Further properties of the HT mechanism are 1) hash agility, 2) mutual authentication, and 3) being secured by channel binding.

Clients are supposed to request SASL-HT tokens from the server after being authenticated using a "strong" SASL mechanism like SCRAM [RFC5802]. Hence a typical sequence of actions using HT may look like the following:

```
      A) Client authenticates using a strong mechanism (e.g., SCRAM)
      B) Client requests secret SASL-HT token
      C) Service returns SASL-HT token
         <normal client-server interaction here>
      D) Connection between client and server gets interrupted,
         for example because of a WiFi <-> GSM switch
      E) Client resumes previous session using HT and token from C)
      F) Service revokes the sucessfully used SASL-HT token
         [goto B]
```

The HT mechanism requires an accompanying, application protocol specific, extension, which allows clients to requests a new SASL-HT token (see Section 5). One example for such an application protocol specific extension based on HT is [XEP-ISR-SASL2]. This XMPP [RFC6120] extension protocol allows, amoungst other things, B) and C),

Since the SASL-HT token is not salted, and only one hash iteration is used, the HT mechanism is not suitable to protect long-lived shared secrets (e.g. "passwords"). You may want to look at [RFC5802] for that.

1.1. Conventions and Terminology

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 [RFC2119].

1.2. Applicability

Because this mechanism transports information that should not be controlled by an attacker, the HT mechanism MUST only be used over channels protected by Transport Layer Security (TLS, see [RFC5246]), or over similar integrity-protected and authenticated channels. Also the application protoocl specific extension which requests a new SASL-HT token SHOULD only be used over similar protected channels.

In addition, when TLS is used, the client MUST successfully validate the server's certificate ([RFC5280], [RFC6125]).

The family of HT mechanisms is not applicable for proxy authentication, since they can not carry a authorization identity string (authzid).

2. The HT Family of Mechanisms

Each mechanism in this family differs by the choice of the hash algorithm and the choice of the channel binding [RFC5929] type.

A HT mechanism name is a string beginning with "HT-" followed by the capitalized name of the used hash, followed by "-", and suffixed by one of 'ENDP' and 'UNIQ'.

Hence each HT mechanism has a name of the following form:

```
                        HT-<hash-alg>-<cb-type>
```

Where <hash-alg> is the capitalized "Hash Name String" of the IANA "Named Information Hash Algorithm Registry" [iana-hash-alg] as specified in [RFC6920], and <cb-type> is one of 'ENDP' or 'UNIQ' denoting the channel binding type. In case of 'ENDP', the tls-server-end-point channel binding type is used. In case of 'UNIQ', the tls-unique channel binding type is used. Valid channel binding types are defined in the IANA "Channel-Binding Types" registry [iana-cbt] as specified in [RFC5056].

CBT	Channel Binding Type
ENDP	tls-server-end-point
UNIQ	tls-unique

Mapping of CBT to Channel Bindings

The following table lists the HT SASL mechanisms registered by this document.

Mechanism Name	HT Hash Algorithm	Channel-binding unique prefix
HT-SHA-512-ENDP	SHA-512	tls-server-end-point
HT-SHA-512-UNIQ	SHA-512	tls-unique
HT-SHA3-512-ENDP	SHA3-512	tls-server-end-point
HT-SHA-256-UNIQ	SHA-256	tls-unique

Defined HT SASL mechanisms

3. The HT Authentication Exchange

The mechanism consists of a simple exchange of exactly two messages between the initiator and responder.

The following syntax specifications use the Augmented Backus-Naur form (ABNF) notation as specified in [RFC5234].

3.1. Initiator First Message

The HT mechanism starts with the initiator-msg, send by the initiator to the responder. The follwing lists the ABNF syntax for the initiator-msg:

```
initiator-msg = authcid-length authcid-data initiator-hashed-token
```

```
authcid-length = 2OCTET

authcid-data = 1*OCTET

initiator-hashed-token = 1*OCTET
```

The initiator-msg starts with an unsigned 16-bit integer in big endian. It denotes length of the authcid-data, which contains the authentication identity. Before sending the authentication identity string the initiator SHOULD prepare the data with the UsernameCasePreserved profile of [RFC8265].

The authcid-data is followed by initiator-hashed-token. The value of the initiator-hashed-token is defined as follows:

```
initiator-hashed-token := HMAC(token, "Initiator" || cb-data)
```

HMAC() is the function defined in [RFC2104] with H being the selected HT hash algorithm, 'cb-data' represents the data provided by the selected channel binding type, and 'token' are the UTF-8 [RFC3629] encoded octets of the SASL-HT token string which acts as shared secret between initiator and responder.

The initiator-msg MAY be included in TLS 1.3 0-RTT early data, as specified in [I-D.ietf-tls-tls13]. If this is the case, then the initiating entity MUST NOT include any further appliction protocol payload in the early data besides the HT initiator-msg and potential required framing of the SASL profile. The responder MUST abort the SASL authentication if the early data contains additional application protocol payload.

> TODO: It should be possible to exploit TLS 1.3 early data for "0.5" RTT resumption of the application protocol's session. That is, on resumption the initiating entity MUST NOT send any application protocol payload together with first flight data, besides the HT initiator-msg. But if the responding entity is able to verify the TLS 1.3 early data, then it can send additional application protocol payload right away together with the "resumption successful" response to the initiating entity.
>
> TODO: Add note why HMAC() is always involved, even if HMAC() is usually not required when modern hash algorithms are used.

3.2. Initiator Authentication

Upon receiving the initiator-msg, the responder calculates itself the value of initiator-hashed-token and compares it with the received value found in the initiator-msg. If both values are equal, then the

initiator has been successfully authenticated. Otherwise, if both values are not equal, then authentication MUST fail.

If the responder was able to authenticate the initiator, then the used token MUST be revoked immediately.

3.3. Final Responder Message

After the initiator was authenticated the responder continues the SASL authentication by sending the responder-msg to the initiator.

The ABNF for responder-msg is:

```
responder-msg = 1*OCTET
```

The responder-msg value is defined as follows:

```
responder-msg := HMAC(token, "Responder" || cb-data)
```

The initiating entity MUST verify the responder-msg to achieve mutual authentication.

4. Compliance with SASL Mechanism Requirements

This section describes compliance with SASL mechanism requirements specified in Section 5 of [RFC4422].

1. "HT-SHA-256-ENDP", "HT-SHA-256-UNIQ", "HT-SHA-3-512-ENDP" and "HT-SHA-3-512-UNIQ".

2. Definition of server-challenges and client-responses:

   a HT is a client-first mechanism.

   b HT does send additional data with success (the responder-msg).

3. HT is not capable of transferring authorization identities from the client to the server.

4. HT does not offer any security layers (HT offers channel binding instead).

5. HT does not protect the authorization identity.

5. Requirements for the Application-Protocol Extension

It is REQUIRED that the application-protocol specific extension provides a mechanism to request a SASL-HT token in form of a Unicode string. The returned token MUST have been newly generated by a cryptographically secure random number generator and MUST contain at least 128 bit of entropy.

It is RECOMMENDED that the the protocol allows the requestor to signal the name of the SASL mechanism which he intends to use with the token. If a token is used with a different mechanism than the one which was signalled upon requesting the token, then the authentication MUST fail. This allows to pin the token to a SASL mechanism, which increases the security because it makes it impossible for an attacker to downgrade the SASL mechanism.

6. Security Considerations

To be secure, the HT mechanism MUST be used over a TLS channel that has had the session hash extension [RFC7627] negotiated, or session resumption MUST NOT have been used.

It is RECOMMENDED that implementations peridically require a full authentication using a strong SASL mechanism which does not use the SASL-HT token.

It is of vital importance that the SASL-HT token is generated by a cryptographically secure random generator. See [RFC4086] for more information about Randomness Requirements for Security.

7. IANA Considerations

IANA has added the following family of SASL mechanisms to the SASL Mechanism registry established by [RFC4422]:

```
To: iana@iana.org
Subject: Registration of a new SASL family HT

SASL mechanism name (or prefix for the family): HT-*
Security considerations:
  Section FIXME of draft-schmaus-kitten-sasl-ht
Published specification (optional, recommended):
  draft-schmaus-kitten-sasl-ht-XX (TODO)
Person & email address to contact for further information:
IETF SASL WG <kitten@ietf.org>
Intended usage: COMMON
Owner/Change controller: IESG <iesg@ietf.org>
Note: Members of this family MUST be explicitly registered
using the "IETF Review" [@!RFC5226] registration procedure.
Reviews MUST be requested on the Kitten WG mailing list
<kitten@ietf.org> (or a successor designated by the responsible
Security AD).
```

8. References

8.1. Normative References

[I-D.ietf-tls-tls13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-21 (work in progress), July 2017.

[iana-cbt] Williams, N., "IANA Channel-Binding Types", 2010, <https://www.iana.org/assignments/channel-binding-types/channel-binding-types.xhtml>.

[iana-hash-alg] Williams, N., "IANA Named Information Hash Algorithm Registry", 2010, <https://www.iana.org/assignments/named-information/named-information.xhtml#hash-alg>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[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>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC4422] Melnikov, A., Ed. and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, DOI 10.17487/RFC4422, June 2006, <https://www.rfc-editor.org/info/rfc4422>.

[RFC5056] Williams, N., "On the Use of Channel Bindings to Secure Channels", RFC 5056, DOI 10.17487/RFC5056, November 2007, <https://www.rfc-editor.org/info/rfc5056>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[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>.

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, DOI 10.17487/RFC5929, July 2010, <https://www.rfc-editor.org/info/rfc5929>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6920] Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B., Keranen, A., and P. Hallam-Baker, "Naming Things with Hashes", RFC 6920, DOI 10.17487/RFC6920, April 2013, <https://www.rfc-editor.org/info/rfc6920>.

[RFC7627] Bhargavan, K., Ed., Delignat-Lavaud, A., Pironti, A., Langley, A., and M. Ray, "Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension", RFC 7627, DOI 10.17487/RFC7627, September 2015, <https://www.rfc-editor.org/info/rfc7627>.

[RFC8265] Saint-Andre, P. and A. Melnikov, "Preparation, Enforcement, and Comparison of Internationalized Strings Representing Usernames and Passwords", RFC 8265, DOI 10.17487/RFC8265, October 2017, <https://www.rfc-editor.org/info/rfc8265>.

8.2. Informative References

[RFC5802] Newman, C., Menon-Sen, A., Melnikov, A., and N. Williams, "Salted Challenge Response Authentication Mechanism (SCRAM) SASL and GSS-API Mechanisms", RFC 5802, DOI 10.17487/RFC5802, July 2010, <https://www.rfc-editor.org/info/rfc5802>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <https://www.rfc-editor.org/info/rfc6120>.

[XEP-ISR-SASL2]
Schmaus, F., "XEP-XXXX: Instant Stream Resumption", 2017, <http://geekplace.eu/xeps/xep-isr-sasl2/xep-isr-sasl2.html>.

Appendix A. Acknowledgments

This document benefited from discussions on the KITTEN WG mailing list. The authors would like to specially thank Thijs Alkemade, Sam Whited and Alexey Melnikov for their comments on this topic. Also thanks to Alexander Wuerstlein, who came up with the idea to pin the token to a SASL mechanism for increased security.

Authors' Addresses

Florian Schmaus
University of Erlangen-Nuremberg

Email: schmaus@cs.fau.de

Christoph Egger
University of Erlangen-Nuremberg

Email: egger@cs.fau.de