Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2013

S. Josefsson
SJD AB
C. Latze
Swisscom

July 12, 2012

SASL Mechanism Family for External Authentication: EXTERNAL-* draft-josefsson-sasl-external-channel-05

Abstract

This document describes a way to perform client authentication in the Simple Authentication and Security Layer (SASL) framework by referring to the client authentication provided by an external security layer. We specify a SASL mechanism family EXTERNAL-* and one instance EXTERNAL-TLS that rely on the Transport Layer Security (TLS) protocol. This mechanism differs to the existing EXTERNAL mechanism by alleviating the a priori assumptions that servers and clients needs somehow negotiate out of band which secure channel that is intended. This document also discuss the implementation of authorization decisions.

See <http://josefsson.org/external-channel/> for more information.

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 http://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 January 13, 2013.

Copyright Notice

Table of Contents

1. Introduction

The EXTERNAL mechanism, described in Appendix A of [RFC4422] allows a client to request the server to use credentials established by means external to the mechanism to authenticate the client. The external means may be, for instance, TLS [RFC5246] or IP Security [RFC4301] services.

The EXTERNAL mechanism requires some a priori agreement between the client and the server regarding which external channel, and consequently which external credentials, should be used for authentication. In practice this has often meant that the EXTERNAL mechanism is only used when there is tight out of band interaction between the server administration and client user. This has impacted the interoperability of the EXTERNAL mechanism.

The EXTERNAL-* mechanism family, specified in this document, is similar to the EXTERNAL mechanism in that it relies on an external channel to perform the client authentication. However, EXTERNAL-* provides a way for the client to provide an identifier of the external channel that is intended to provide the client credentials. The intention is that the server need not rely on a priori arrangement to identify the secure channel that was used, but can automatically find the intended channel and re-use its credentials for the SASL authentication. Further, upon successful authentication, the client knows that the server used credentials from the indicated security channel.

In the EXTERNAL-* mechanism family, the external channel is identified through the SASL mechanism name.

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 [RFC2119].

2. Use Cases

Depending on the application, in addition to authenticating a user it is also important to authenticate the device the user is logged in to. Assuming that the user and the device ID consist of an X.509 certificate, on way to authenticate a user and a device is to establish a secure tunnel based on the device's certificate. The user certificate will then be used to authenticate the user within that tunnel. Although this solution works nicely with today's authentication protocols it comes with a certain complexity since it requires a tunnel-in-tunnel setup. It would be better to end up with only one secure tunnel while still being able to use both

certificates. Another point is that the authorization decision might be based on both authentications. The user is only allowed to access certain resources if it uses a certain machine.

One real world scenario of this use case the so called bring-your-own-device (BYOD) initiatives. In BYOD, a company allows employees to bring their own hardware to access the company's infrastructure. This is risky since they still want to make sure that only this employee can access the infrastructure. Therefore the company could issue a device certificate for this device as well as a user certificate for the employee in order to make sure that only this employee can access the network with his device.

This scheme might be extended on even more than two identities.

The EXTERNAL-TLS mechanism provides means to implement this scheme.

3. Specification of EXTERNAL-* Mechanism Family

The name of the mechanism family is "EXTERNAL-".

The mechanism family does not provide a security layer. It provides similar functionality by relying on an external channel.

The mechanism is capable of transferring an authorization identity string. If the authorization identity string is empty, the client is requesting to act as the identity the server has associated with the client's credentials. If the authorization identity string is non-empty, the client is requesting to act as the identity represented by the string.

The client is expected to send data first in the authentication exchange. Where the client does not provide an initial response data in its request to initiate the authentication exchange, the server is to respond to the request with an empty initial challenge and then the client is to provide its initial response.

The client sends the initial response containing a UTF-8 [RFC3629] encoding of the requested authorization identity string.

The authorization identity is non-empty when the client is requesting to act as the identity represented by the (non-empty) string. The authorization identity is empty when the client is requesting to act as the identity the server associates with the external authentication credentials.

The syntax of the initial response is specified as a value of the

<extern-initial-resp> production detailed below using the Augmented Backus-Naur Form (ABNF) [RFC5234] notation.

```
external-initial-resp = authz-id-string

authz-id-string       = *( UTF8-char-no-nul )
UTF8-char-no-nul      = UTF8-1-no-nul / UTF8-2 / UTF8-3 / UTF8-4
     ;; where the UTF8-2, UTF8-3, and UTF8-4 productions are
     ;; as defined in RFC 3629.

UTF8-1-no-nul         = %x01-7F
```

There are no additional challenges and responses.

Hence, the server is to return the outcome of the authentication exchange.

The external security channel to use is implied by the SASL mechanism name. The channel has to be uniquely identifiable at both cliend and server side. This means that mechanisms registered in this family MUST detail which channel should be chosen if there are layered channels of the same type.

The exchange fails if

- the client has not established its credentials via the indicated external channel,

- the client's credentials are inadequate,

- the client provided an empty authorization identity string and the server is unwilling or unable to associate an authorization identity with the client's credentials,

- the client provided a non-empty authorization identity string that is invalid per the syntax requirements of the applicable application protocol specification,

- the client provided a non-empty authorization identity string representing an identity that the client is not allowed to act as, or

- the server is unwilling or unable to provide service to the client for any other reason.

Otherwise the exchange is successful. When indicating a successful outcome, additional data is not provided.

4. Specification of EXTERNAL-TLS Mechanism

The purpose of the EXTERNAL-TLS mechanism is to refer to the authentication completed by an already negotiated TLS [RFC5246] protocol. This covers potentially both client and server authentication. The typical scenario is that applications enable TLS protection of the application protocol using a STARTTLS-like functionality, performs whatever client and server authentication necessary within the TLS session, and then proceeds to the EXTERNAL-TLS mechanism negotiation.

Usually the TLS channel will have only one TLS handshake, but multiple TLS handshakes (i.e., one initial TLS handshake followed by re-negotiations) MAY be used to establish multiple authentications. Implementations MUST only use credentials established securely with the TLS Renegotiation Extension [RFC5746]. The set of credentials relevant to EXTERNAL-TLS authentication starts with the inner-most TLS channel and includes each additional credential negotiated outside of the current TLS channel when that channel was negotiated using TLS Renegotiation Extension.

For example, if an application opens up a TLS channel and starts SASL negotiation, and if that communication happens to be sent over a TLS-based VPN, the intended channel is the TLS channel opened by the application. Only the credentials established by the application TLS handshake is relevant.

The server MUST NOT advertise the EXTERNAL-TLS mechanism if the client did not provided any supported form of client-side authentication in the TLS channel, e.g., X.509 client certificate, OpenPGP client key [RFC6091], or SRP [RFC5054]. The client MUST only request the EXTERNAL-TLS if it wishes to re-use the TLS client credentials for the SASL application.

5. Making Authorization Decisions

The server may use any mechanism to make authorization decisions. For illustration, we want to give some ideas on how this may work in practice. This section is not normative.

Typically external channels will not use authentication identities that can be used by the application protocol that uses an instance of the SASL EXTERNAL-* mechanism. Thus, a mapping is normally required. There may be mappings from the external credential to a set of permitted identifiers, and a "default" identifier can be provided in the mapping table if the client do not specify a particular authorization identity.

For example, when mapping from X.509 credentials used in TLS connections to simple usernames, a table stored on the server can contain hex-encoded hashes of client X.509 certificates and a set of usernames.

```
aef3a7835277a28da831005c2ae3b919e2076a62 simon jas admin
d2fc512490a15036460b5489401439d6da5407fa joe
```

The server could extract a successfully authenticated X.509 client certificate from the TLS stack, hash it and look it up in the mapping table. Each of the usernames given would be permitted authorization identities. The first username given may be the default username if the client does not provide an authorization identity.

When mapping from multiple re-negotiated TLS handshakes, the server could extract all successfully authenticated X.509 client certificates from the TLS stack, hash them, concatenate them and look the concatenation string up in the mapping table. The following shows an example where a first TLS handshake has been negotiated to authenticate the client's machine and the second re-negotiated TLS handshake was used to authenticate the user.

```
    da831005c2ae3b919d2fc512490a15036460b548\
    d2fc512490a15036460b5489401439d6da5407fa carolin@tux
```

When mapping from OpenPGP credentials used in TLS [RFC6091], the mapping table could consist of verified OpenPGP fingerprints and a set of permitted usernames, such as the following table.

```
0424D4EE81A0E3D119C6F835EDA21E94B565716F simon jas admin
A4D94E92B0986AB5EE9DCD755DE249965B0358A2 werner
90A79E2FC6F4AAB5B604974FE15DD857B15C37D1 nikos
```

When SRP authentication with TLS [RFC5054] is used, the username provided may be the same as the application username, and no mapping would be necessary.

6. Examples

This section provides examples of EXTERNAL-TLS authentication exchanges. The examples are intended to help the readers understand the above text. The examples are not definitive. The Application Configuration Access Protocol (ACAP) [RFC2244] is used in the examples because ACAP sends the SASL tokens without additional encoding.

The first example shows use of EXTERNAL-TLS with an empty

authorization identity. In this example, the initial response is not sent in the client's request to initiate the authentication exchange.

```
S: * ACAP (SASL "GSSAPI")
C: a001 STARTTLS
S: a001 OK "Begin TLS negotiation now"
<TLS negotiation, further commands are under TLS layer>
S: * ACAP (SASL "GSSAPI" "PLAIN" "EXTERNAL-TLS")
C: a002 AUTHENTICATE "EXTERNAL-TLS"
S: + ""
C: + ""
S: a002 OK "Authenticated"
```

The second example shows use of EXTERNAL-TLS with an authorization identity of "simon". In this example, the initial response is sent with the client's request to initiate the authentication exchange. This saves a round-trip.

```
S: * ACAP (SASL "GSSAPI")
C: a001 STARTTLS
S: a001 OK "Begin TLS negotiation now"
<TLS negotiation, further commands are under TLS layer>
S: * ACAP (SASL "GSSAPI" "PLAIN" "EXTERNAL-TLS")
C: a002 AUTHENTICATE "EXTERNAL-TLS" {5+}
C: simon
S: a002 NO "Cannot assume requested authorization identity"
```

Note how the server rejects the authentication attempt with an authorization-related error message. Presumably the client credentials presented in the TLS session does not give the client authority to assume the identity of "simon".

The third example shows use of EXTERNAL-TLS with multiple re-negotiated TLS handshakes. The first TLS negotiation could have been authenticated with a device certificate, and the TLS re-negotiation could have been authenticated with a user certificate. Furthermore, an authorization identity of "carolin@tux" is used.

```
S: * ACAP (SASL "GSSAPI")
C: a001 STARTTLS
S: a001 OK "Begin TLS negotiation now"
<TLS negotiation, further commands are under TLS layer>
<TLS re-negotiation, further commands are under TLS layer>
S: * ACAP (SASL "GSSAPI" "PLAIN" "EXTERNAL-TLS")
C: a002 AUTHENTICATE "EXTERNAL-TLS"
S: + ""
C: + carolin@tux
S: a002 OK "Authenticated"
```

7. IANA Considerations

The IANA is requested to add to the SASL mechanisms registry the following entry.

> Subject: Registration of SASL mechanism family EXTERNAL-*
> SASL family name (or prefix for the family): EXTERNAL-
> Security considerations: [THIS-DOC]
> Published specification (recommended): [THIS-DOC]
> Person & email address to contact for further information:
> Simon Josefsson <simon@josefsson.org>
> Intended usage: COMMON
> Owner/Change controller: Simon Josefsson <simon@josefsson.org>

IANA will register new SASL mechanism names under the "EXTERNAL-" namespace on a First Come First Served basis, as defined in [RFC5226]. IANA has the right to reject obviously bogus registration requests, but will perform no review of claims made in the registration form.

Registration of a SASL mechanism under the "EXTERNAL-" namespace is requested by filling in the same template used in [RFC4422] using a name prefixed with "EXTERNAL-".

While this registration procedure does not require expert review, authors of SASL mechanisms are encouraged to seek community review and comment whenever that is feasible. Authors may seek community review by posting a specification of their proposed mechanism as an Internet-Draft. SASL mechanisms intended for widespread use should be standardized through the normal IETF process, when appropriate.

8. Security Considerations

The security of external channel is critical to the security of this mechanism. It is important that the client authentication provided by the security channel is securely bound to any confidentiality or integrity services that protects the security channel.

The EXTERNAL-* mechanism family does not authenticate clients itself, it relies on implementation to perform the authentication as part of the external channel. Care must be taken to ensure that the client credential has been authenticated, rather than just blindly accepted as part of a leap-of-faith setup.

9. Acknowledgements

Significant amount of text in this document is copied from SASL [RFC4422].

The document was improved by discussion in the SASL Working Group between Chris Newman, Philip Guenther, Alexey Melnikov, Hallvard B Furuseth, Nicolas Williams, Sam Hartman, Jeffrey Hutzelman, and Kurt Zeilenga.

Further fruitful discussions took place with Paul Sangster and Gloria Serrao.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC4422] Melnikov, A. and K. Zeilenga, "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5746] Rescorla, E., Ray, M., Dispensa, S., and N. Oskov, "Transport Layer Security (TLS) Renegotiation Indication Extension", RFC 5746, February 2010.

10.2. Informative References

[RFC2244] Newman, C. and J. Myers, "ACAP -- Application Configuration Access Protocol", RFC 2244, November 1997.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC5054] Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, "Using the Secure Remote Password (SRP) Protocol for TLS Authentication", RFC 5054, November 2007.

[RFC6091] Mavrogiannopoulos, N. and D. Gillmor, "Using OpenPGP Keys for Transport Layer Security (TLS) Authentication", RFC 6091, February 2011.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

Authors' Addresses

Simon Josefsson
SJD AB

Email: simon@josefsson.org

Carolin Latze
Swisscom

Email: carolin.latze@swisscom.com