TLS S. Santesson
Internet-Draft 3xA Security AB
Intended status: Standards Track H. Tschofenig
Expires: June 28, 2012 Nokia Siemens Networks
December 26, 2011

Transport Layer Security (TLS) Cached Information Extension
draft-ietf-tls-cached-info-11.txt

Abstract

Transport Layer Security (TLS) handshakes often include fairly static information, such as the server certificate and a list of trusted Certification Authorities (CAs). This information can be of considerable size, particularly if the server certificate is bundled with a complete certificate path (including all intermediary certificates up to the trust anchor public key).

This document defines an extension that omits the exchange of already available information. The TLS client informs a server of cached information, for example from a previous TLS handshake, allowing the server to omit the already available information.

Status of this Memo

This Internet-Draft will expire on June 28, 2012.

Copyright Notice

Table of Contents

1. Introduction

Transport Layer Security (TLS) handshakes often include fairly static information, such as the server certificate and a list of trusted Certification Authorities (CAs). This information can be of considerable size, particularly if the server certificate is bundled with a complete certificate path (including all intermediary certificates up to the trust anchor public key).

Optimizing the exchange of information to a minimum helps to improve performance in environments where devices are connected to a network with characteristics like low bandwidth, high latency and high loss rate. These types of networks exist, for example, when smart objects are connected using a low power IEEE 802.15.4 radio. For more information about the challenges with smart object deployments please see [I-D.iab-smart-object-workshop].

This specification defines a TLS extension that allows a client and a server to exclude transmission of cached information from the TLS handshake.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Cached Information Extension

This document defines a new extension type (cached_information(TBD)), which is used in client hello and server hello messages. The extension type is specified as follows.

```
      enum {
           cached_information(TBD), (65535)
      } ExtensionType;
```

The extension_data field of this extension, when included in the client hello, MUST contain the CachedInformation structure.

```
      enum {
           certificate_chain(1), trusted_cas(2) (255)
      } CachedInformationType;

      struct {
           CachedInformationType type;
           HashAlgorithm hash;
           opaque hash_value<1..255>;
      } CachedObject;

      struct {
           CachedObject cached_info<1..2^16-1>;
      } CachedInformation;
```

When the CachedInformationType identifies a certificate_chain, then the hash_value field MUST include a hash calculated over the certificate_list element of a server side Certificate message, excluding the three length bytes of the certificate_list vector.

When the CachedInformationType identifies a trusted_cas, then the hash_value MUST include a hash calculated over the certificate_authorities element of a server side CertificateRequest message, excluding the two length bytes of the certificate_authorities vector.

The hash algorithm used to calculate hash values is conveyed in the 'hash' field of the CachedObject element. This document defines the following hash algorithms:

- SHA-1: NIST FIPS PUB 180-3 [SHA]

- SHA-224: RFC 3874 [RFC3874]

- SHA-256: NIST FIPS PUB 180-3 [SHA]

- SHA-384: NIST FIPS PUB 180-3 [SHA]

- SHA-512: NIST FIPS PUB 180-3 [SHA]

This document establishes a registry for CachedInformationType types and additional values can be added following the policy described in Section 6.

4. Exchange Specification

Clients supporting this extension MAY include the "cached_information" extension in the (extended) client hello, which MAY contain zero or more CachedObject attributes.

Server supporting this extension MAY include the "cached_information" extension in the (extended) server hello, which MAY contain one or more CachedObject attributes. By returning the "cached_information" extension the server indicates that it supports caching of each present CachedObject that matches the specified hash value. The server MAY support other cached objects that are not present in the extension.

Note: Clients may need the ability to cache different values depending on other information in the Client Hello that modify what values the server uses, in particular the Server Name Indication [I-D.ietf-tls-rfc4366-bis] value.

Following a successful exchange of "cached_information" extensions, the server MAY send fingerprints of the cached information in the handshake exchange as a replacement for the exchange of the full data. Section 4.1 and Section 4.2 defines the syntax of the fingerprinted information.

The handshake protocol MUST proceed using the information as if it was provided in the handshake protocol. The Finished message MUST be calculated over the actual data exchanged in the handshake protocol. That is, the Finished message will be calculated over the hash values of cached information objects and not over the cached information that were omitted from transmission.

The server MUST NOT include more than one fingerprint for a single information element, i.e., at maximum only one CachedObject structure per replaced information is provided.

4.1. Fingerprint of the Certificate Chain

When an object of type 'certificate_chain' is provided in the client hello, the server MAY send a fingerprint instead of the complete certificate chain as shown below.

The original handshake message syntax is defined in RFC 5246 [RFC5246] and has the following structure:

```
      opaque ASN.1Cert<1..2^24-1>;

      struct {
          ASN.1Cert certificate_list<0..2^24-1>;
      } Certificate;
```

By using the extension defined in this document the following information is sent:

```
   struct {
                  CachedObject ASN.1Cert<1..2^24-1>;
   } Certificate;
```

The opaque ASN.1Cert structure is replaced with the CachedObject structure defined in this document.

Note: [I-D.wouters-tls-oob-pubkey] allows a PKIX certificate containing only the SubjectPublicKeyInfo instead of the full information typically found in a certificate. Hence, when this specification is used in combination with [I-D.wouters-tls-oob-pubkey] and the negotiated certificate type is RawPublicKey then the TLS server sends the hashed Certificate element that contains a ASN.1Cert with the mentioned raw public key.

4.2. Fingerprint for Trusted CAs

When a hash for an object of type 'trusted_cas' is provided in the client hello, the server MAY send a fingerprint instead of the complete certificate authorities information as shown below.

The original handshake message syntax is defined in RFC 5246 [RFC5246] and has the following structure:

```
      opaque DistinguishedName<1..2^16-1>;

      struct {
          ClientCertificateType certificate_types<1..2^8-1>;
          SignatureAndHashAlgorithm
            supported_signature_algorithms<2^16-1>;
          DistinguishedName certificate_authorities<0..2^16-1>;
      } CertificateRequest;
```

By using the extension defined in this document the following information is sent:

```
      struct {
         ClientCertificateType certificate_types<1..2^8-1>;
         SignatureAndHashAlgorithm
           supported_signature_algorithms<2^16-1>;
         CachedObject DistinguishedName<1..2^16-1>;
      } CertificateRequest;
```

The opaque DistinguishedName structure is replaced with the CachedObject structure defined in this document.

5. Security Considerations

The hash algorithm used in this specification is required to have reasonable random properties in order to provide reasonably unique identifiers. There is no requirement that this hash algorithm must have strong collision resistance.

Caching information in an encrypted handshake (such as a renegotiated handshake) and sending a hash of that cached information in an unencrypted handshake might introduce integrity or data disclosure issues as it enables an attacker to identify if a known object (such as a known server certificate) has been used in previous encrypted handshakes. Information object types defined in this specification, such as server certificates, are public objects and usually not sensitive in this regard, but implementers should be aware if any cached information are subject to such security concerns and in such case SHOULD NOT send a hash over encrypted data in en unencrypted handshake.

6. IANA Considerations

6.1. New Entry to the TLS ExtensionType Registry

IANA is requested to add an entry to the existing TLS ExtensionType registry, defined in RFC 5246 [RFC5246], for cached_information(TBD) defined in this document.

6.2. New Registry for CachedInformationType

IANA is requested to establish a registry for TLS CachedInformationType values. The first entries in the registry are

- certificate_chain(1)

- trusted_cas(2)

The policy for adding new values to this registry, following the terminology defined in RFC 5226 [RFC5226], is as follows:

- 0-63 (decimal): Standards Action

- 64-223 (decimal): Specification Required

- 224-255 (decimal): reserved for Private Use

6.3. New Registry for HashAlgorithm

IANA is requested to establish a registry for HashAlgorithm values and to populate the registry with an initial set of values listed in Section 3.

The policy for adding new values to this registry, following the terminology defined in RFC 5226 [RFC5226], is as follows:

- 0-63 (decimal): Standards Action

- 64-223 (decimal): Specification Required

- 224-255 (decimal): reserved for Private Use

7. Acknowledgments

The author acknowledges input from many members of the TLS working group.

We would like to thank Paul Wouters for his feedback and Nikos Mavrogiannopoulos for his document review in December 2011.

8. References

8.1. Normative References

[I-D.ietf-tls-rfc4366-bis]
3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", draft-ietf-tls-rfc4366-bis-12 (work in progress), September 2010.

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

[RFC3874] Housley, R., "A 224-bit One-way Hash Function: SHA-224", RFC 3874, September 2004.

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

[SHA] "Federal Information Processing Standards Publication (FIPS PUB) 180-3, Secure Hash Standard (SHS)", October 2008.

8.2. Informative References

[I-D.iab-smart-object-workshop]
Tschofenig, H. and J. Arkko, "Report from the 'Interconnecting Smart Objects with the Internet' Workshop, 25th March 2011, Prague", draft-iab-smart-object-workshop-06 (work in progress), October 2011.

[I-D.wouters-tls-oob-pubkey]
Wouters, P., Gilmore, J., Weiler, S., Kivinen, T., and H. Tschofenig, "TLS out-of-band public key validation", draft-wouters-tls-oob-pubkey-02 (work in progress), November 2011.

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

Authors' Addresses

Stefan Santesson
3xA Security AB
Scheelev. 17
Lund 223 70
Sweden

Email: sts@aaa-sec.com

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at