Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 10, 2012

Y. Pettersen
Opera Software ASA

May 9, 2012

The TLS Multiple Certificate Status Request Extension
draft-ietf-tls-multiple-cert-status-extension-00

Abstract

This document defines the Transport Layer Security (TLS) Certificate Status Version 2 Extension to allow clients to specify and support multiple certificate status methods. Also defined is a new method based on the Online Certificate Status Protocol (OCSP) that servers can use to provide status information not just about the server's own certificate, but also the status of intermediate certificates in the chain.

Requirements Language

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].

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 November 10, 2012.

Copyright Notice

1. Introduction

The Transport Layer Security (TLS) Extension [RFC6066] framework defines, among other extensions, the Certificate Status Extension that clients can use to request the server's copy of the current status of its certificate. The benefits of this extension include a reduced number of roundtrips and network delays for the client to verify the status of the server's certificate and a reduced load on the certificate issuer's status response servers, thus solving a problem that can become significant when the issued certificate is presented by a frequently visited server.

There are two problems with the existing Certificate Status extension. First, it does not provide functionality to request the status information about intermediate Certification Authority (CA) certificates, which means the client has to request status information through other methods, such as CRLs, thus adding additional delay. Second, the current format of the extension and requirements in the TLS protocol prevents a client from offering the server multiple status methods.

Many Certification Authorities are now issuing intermediate CA certificates that not only specify a CRL Distribution Point [RFC5280], but also a URL for OCSP [RFC2560] Certificate Status requests. Given that client-cached CRLs are frequently out of date, using OCSP to access up-to-date status information about intermediate

CA certificates will be of great benefit to clients. The benefit to the issuing CA is less clear, as providing the bandwidth for the OCSP responder can be costly, especially for CAs with many high-traffic subscriber sites, and this cost is a concern for many CAs. There are cases where OCSP requests for a single high-traffic site caused significant network problems for the issuing CA.

For these reasons, it will be beneficial to use the TLS server to provide the certificate status information not just for the server certificate, but also for the intermediate CA certificates. This will reduce the roundtrips needed to complete the handshake by the client to just those needed for negotiating the TLS connection. Also, for the Certification Authorities, the load on their servers will depend on the number of certificates they have issued, not on the number of visitors to those sites.

For such a new system to be introduced seamlessly, it must be possible for clients to indicate support for the existing OCSP Certificate Status method, and a new multiple-OCSP mode.

Unfortunately, the definition of the Certificate Status extension only allows a single Certificate Status extension to be defined in a single extension record in the handshake, and the TLS Protocol only allows a single record in the extension list for any given extension. This means that it is not possible for clients to indicate support for new methods while still supporting older methods, which would cause problems for interoperability between newer clients and older servers. This will not just be an issue for the multiple status request mode proposed above, but also for any other future status methods that might be introduced. This will be true not just for the current PKIX infrastructure, but also for alternative PKI structures.

The solution to this problem is to define a new extension, status_request_v2, with an extended format that allows the client to indicate support for multiple status request methods. This is implemented by using a list of CertificateStatusRequestItem records in the extension record. As the server will select the single status method based on the selected cipher suite and the certificate presented, no significant changes are needed in the server's extension format.

2. Multiple Certificate Status Extension

2.1. New extension

The extension defined by this document is indicated by the "status_request_v2" in the ExtensionType enum, which uses the following value:

```
  enum {
    status_request_v2(XX) (65535)
  } ExtensionType;
```

[[EDITOR: The value used for status_request_v2 has been left as XX. This value will be assigned when this draft progresses to RFC.]]

2.2. Multiple Certificate Status Request record

Clients that support a certificate status protocol like OCSP may send the status_request_v2 extension to the server in order to use the TLS handshake to transfer such data instead of downloading it through separate connections. When using this extension, the "extension_data" field of the extension SHALL contain a CertificateStatusRequestList where:

```
struct {
  CertificateStatusType status_type;
  uint16 request_length; /* Length of request field in bytes */
  select (status_type) {
    case ocsp: OCSPStatusRequest;
    case ocsp_multi: OCSPStatusRequest;
  } request;
} CertificateStatusRequestItem;

enum { ocsp(1), ocsp_multi(YY), (255) } CertificateStatusType;

struct {
  ResponderID responder_id_list<0..2^16-1>;
  Extensions request_extensions;
} OCSPStatusRequest;

opaque ResponderID<1..2^16-1>;
opaque Extensions<0..2^16-1>;

struct {
  CertificateStatusRequestItem certificate_status_req_list<1..2^16-1>
} CertificateStatusRequestList
```

[[EDITOR: The value used for ocsp_multi has been left as YY. This value will be assigned when this draft progresses to RFC.]]

In the OCSPStatusRequestItem, the "ResponderIDs" provides a list of OCSP responders that the client trusts. A zero-length "responder_id_list" sequence has the special meaning that the responders are implicitly known to the server, e.g., by prior arrangement, or are identfied by the certificates used by the server. "Extensions" is a DER encoding of OCSP request extensions.

Both "ResponderID" and "Extensions" are DER-encoded ASN.1 types as defined in [RFC2560]. "Extensions" is imported from [RFC5280]. A zero-length "request_extensions" value means that there are no extensions (as opposed to a zero-length ASN.1 SEQUENCE, which is not valid for the "Extensions" type).

In the case of the "id-pkix-ocsp-nonce" OCSP extension, [RFC2560] is unclear about its encoding; for clarification, the nonce MUST be a DER-encoded OCTET STRING, which is encapsulated as another OCTET STRING (note that implementations based on an existing OCSP client will need to be checked for conformance to this requirement).

The list of CertificateStatusRequestItem entries MUST be in order of preference.

A server that receive a client hello containing the "status_request_v2" extension MAY return a suitable certificate status response message to the client along with the server's certificate message. If OCSP is requested, it SHOULD use the information contained in the extension when selecting an OCSP responder and SHOULD include request_extensions in the OCSP request.

The server returns a certificate status response along with its certificate by sending a "CertificateStatus" message immediately after the "Certificate" message (and before any "ServerKeyExchange" or "CertificateRequest" messages). If a server returns a "CertificateStatus" message in response to a status_request_v2 request, then the server MUST have included an extension of type "status_request_v2" with empty "extension_data" in the extended server hello. The "CertificateStatus" message is conveyed using the handshake message type "certificate_status" as follows (see also [RFC6066]):

```
    struct {
      CertificateStatusType status_type;
      select (status_type) {
        case ocsp: OCSPResponse;
        case ocsp_multi: OCSPResponseList;
      } response;
    } CertificateStatus;

    opaque OCSPResponse<0..2^24-1>;

    struct {
      OCSPResponse ocsp_response_list<1..2^24-1>
    } OCSPResponseList
```

An "OCSPResponse" element contains a complete, DER-encoded [CCITT.X680.2002] OCSP response (using the ASN.1 [CCITT.X680.2002] type OCSPResponse defined in [RFC2560]). Only one OCSP response, with a length of at least one byte, may be sent for status_type "ocsp".

An "ocsp_response_list" contains a list of "OCSPResponse" elements, as specified above, each containing the OCSP response for the matching corresponding certificate in the server's Certificate TLS handshake message. That is, the first entry is the OCSP response for the first certificate in the Certificate list, the second entry is the response for the second certificate, and so on. The list MAY contain fewer OCSP responses than there were certificates in the Certificate handshake message, but there MUST NOT be more responses than there were certificates in the list. Individual elements of the list MAY have a length of 0 (zero) bytes, if the server does not have the OCSP response for that particular certificate stored, in which case, the client MUST act as if a response was not received for that particular certificate. If the client receives a "ocsp_response_list" that does not contain a response for one or more of the certificates in the completed certificate chain, the client SHOULD attempt to validate the certificate using an alternative retrieval method, such as downloading the relevant CRL; OCSP SHOULD in this situation only be used for the end entity certificate, not intermediate CA certificates, for reasons stated above.

Note that a server MAY also choose not to send a "CertificateStatus" message, even if it has received a "status_request_v2" extension in the client hello message and has sent a "status_request_v2" extension in the server hello message. Additionally, note that that a server MUST NOT send the "CertificateStatus" message unless it received either a "status_request" or "status_request_v2" extension in the client hello message and sent a corresponding "status_request" or

"status_request_v2" extension in the server hello message.

Clients requesting an OCSP response and receiving one or more OCSP responses in a "CertificateStatus" message MUST check the OCSP response(s) and abort the handshake, if the response is a revoked status or is otherwise not satisfactory with a bad_certificate_status_response(113) alert. This alert is always fatal.

[[Open issue: At least one reviewer has suggested that the client should treat an unsatisfactory (non-revoked) response as an empty response for that particular response and fall back to the alternative method described above]]

3. IANA Considerations

Section 2.1 defines the new TLS Extension status_request_v2 enum, which should be added to the ExtensionType Values list in the IANA TLS category after IETF Concensus has decided to add the value.

Section 2.2 describes a TLS CertificateStatusType Registry to be maintained by the IANA. CertificateStatusType values are to be assigned via IETF Review as defined in [RFC5226]. The initial registry corresponds to the definition of "ExtensionType" in Section 2.2.

4. Security Considerations

General Security Considerations for TLS Extensions are covered in [RFC5246]. Security Considerations for the particular extension specified in this document are given below. In general, implementers should continue to monitor the state of the art and address any weaknesses identified.

4.1. Security Considerations for status_request_v2

If a client requests an OCSP response, it must take into account that an attacker's server using a compromised key could (and probably would) pretend not to support the extension. In this case, a client that requires OCSP validation of certificates SHOULD either contact the OCSP server directly or abort the handshake.

Use of the OCSP nonce request extension (id-pkix-ocsp-nonce) may improve security against attacks that attempt to replay OCSP responses; see Section 4.4.1 of [RFC2560] for further details.

5. Acknowledgements

This document is based on [RFC6066] authored by Donald Eastlake 3rd.

6. Normative References

[CCITT.X680.2002]
International International Telephone and Telegraph Consultative Committee, "Abstract Syntax Notation One (ASN.1): Specification of basic notation", CCITT Recommendation X.680, July 2002.

[CCITT.X690.2002]
International International Telephone and Telegraph Consultative Committee, "ASN.1 encoding rules: Specification of basic encoding Rules (BER), Canonical encoding rules (CER) and Distinguished encoding rules (DER)", CCITT Recommendation X.690, July 2002.

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

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

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

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

[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, May 2008.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

Author's Address

Yngve N. Pettersen
Opera Software ASA

Email: yngve@opera.com