Internet Engineering Task Force (IETF)
Request for Comments: 6961
Category: Standards Track
ISSN: 2070-1721

Y. Pettersen

June 2013

The Transport Layer Security (TLS) Multiple Certificate Status Request Extension

Abstract

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

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6961.

1. Introduction

The Transport Layer Security (TLS) Extension [RFC6066] framework defines, among other extensions, the Certificate Status extension (also referred to as "OCSP stapling") 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 Certificate Revocation Lists (CRLs), introducing further delays. Second, the current format of the extension and requirements in the TLS protocol prevent a client from offering the server multiple status methods.

Many CAs are now issuing intermediate CA certificates that not only specify the publication point for their CRLs in a CRL Distribution Point [RFC5280] but also specify a URL for their OCSP [RFC6960] server in Authority Information Access [RFC5280]. Given that client-cached CRLs are frequently out of date, clients would benefit from using OCSP to access up-to-date status information about intermediate CA certificates. 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.

Clients will benefit from the TLS server providing certificate status information regardless of type, not just for the server certificate but also for the intermediate CA certificates. Combining the status checks into one extension 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. Additionally, using this extension significantly reduces privacy concerns around the clients informing the certificate issuer about which sites they are visiting.

For such a new system to be introduced seamlessly, clients need to be able 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 [RFC5246] 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 [RFC5280] 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 using a list of CertificateStatusRequestItemV2 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.

1.1. 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].

1.2. Presentation Language

This document defines protocol structures using the same conventions and presentation language as defined in Section 4 of [RFC5246].

2. Multiple Certificate Status Extension

2.1. New Extension

The extension defined by this document is indicated by "status_request_v2" in the ExtensionType enum (originally defined by [RFC6066]), which uses the following value:

```
enum {
  status_request_v2(17), (65535)
} ExtensionType;
```

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 (defined in Section 7.4.1.4 of [RFC5246]) of the extension SHALL contain a CertificateStatusRequestListV2 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;
} CertificateStatusRequestItemV2;

enum { ocsp(1), ocsp_multi(2), (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 {
  CertificateStatusRequestItemV2
                    certificate_status_req_list<1..2^16-1>;
} CertificateStatusRequestListV2;
```

In the OCSPStatusRequest (originally defined by [RFC6066]), the "ResponderID" 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 identified by the certificates used by the server. "Extensions" is a DER encoding [X.690] of the OCSP request extensions, and if the server supports the forwarding of OCSP request extensions, this value MUST be forwarded without modification.

Both "ResponderID" and "Extensions" are DER-encoded ASN.1 types as defined in [RFC6960]. "Extensions" is imported from [RFC5280]. A zero-length "request_extensions" value means that there are no extensions (as opposed to a DER-encoded zero-length ASN.1 SEQUENCE, which is not valid for the "Extensions" type).

Servers that support a client's selection of responders using the ResponderID field could implement this selection by matching the responder ID values from the client's list with the ResponderIDs of known OCSP responders, either by using a binary compare of the values or a hash calculation and compare method.

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). This has been clarified in [RFC6960].

The items in the list of CertificateStatusRequestItemV2 entries are ordered according to the client's preference (favorite choice first).

A server that receives 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 (originally defined by [RFC6066]) immediately after the "Certificate" message (Section 7.4.2 of [RFC5246]) (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" (defined in [RFC6066]) as follows (updated from the definition in [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 (originally defined by [RFC6066]) contains a complete, DER-encoded OCSP response (using the ASN.1 [X.680] type OCSPResponse defined in [RFC6960]). 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 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 other unacceptable responses (as determined by client policy) with a bad_certificate_status_response(113) alert. This alert is always fatal.

If the OCSP response received from the server does not result in a definite "good" or "revoked" status, it is inconclusive. A TLS client in such a case MAY check the validity of the server certificate through other means, e.g., by directly querying the certificate issuer. If such processing still results in an inconclusive response, then the application using the TLS connection will have to decide whether to close the connection or not. Note that this problem cannot be decided by the generic TLS client code without information from the application. If the application doesn't provide any such information, then the client MUST abort the connection, since the server certificate has not been sufficiently validated.

An example of where the application might wish to continue is with EAP-TLS (Extensible Authentication Protocol - TLS), where the application can use another mechanism to check the status of a certificate once it obtains network access. In this case, the application could have the client continue with the handshake, but it MUST NOT disclose a username and password until it has fully validated the server certificate.

3. IANA Considerations

Section 2.1 defines the new TLS extension status_request_v2 (17) enum, which has been added to the "ExtensionType Values" list in the IANA "Transport Layer Security (TLS) Extensions" registry.

Section 2.2 describes a TLS CertificateStatusType registry that is now maintained by IANA. The "TLS Certificate Status Types" registry has been created under the "Transport Layer Security (TLS) Extensions" registry. CertificateStatusType values are to be assigned via IETF Review as defined in [RFC5226]. The initial registry corresponds to the definition of "CertificateStatusType" in Section 2.2.

Value	Description	Reference
0	Reserved	[RFC6961]
1	ocsp	[RFC6066] [RFC6961]
2	ocsp_multi	[RFC6961]
3-255	Unassigned	

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 [RFC6960] for further details.

This extension allows the client to send arbitrary data to the server. The server implementers need to handle such data carefully to avoid introducing security vulnerabilities.

The security considerations of [RFC6960] apply to OCSP requests and responses.

5. Acknowledgements

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

6. References

6.1. Normative References

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

[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.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, June 2013.

[X.680] ITU-T Recommendation X.680 (2008) | ISO/IEC 8824-1:2008, "Abstract Syntax Notation One (ASN.1): Specification of basic notation", November 2008.

[X.690] ITU-T Recommendation X.690 (2008) | ISO/IEC 8825-1:2008, "ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", November 2008.

6.2. Informative References

[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.

Author's Address

Yngve N. Pettersen

EMail: yngve@spec-work.net