Stream: Internet Engineering Task Force (IETF)
RFC: 9919
Obsoletes: 5019
Category: Standards Track
Published: January 2026
ISSN: 2070-1721
Authors: 伊藤 忠彦 (T. Ito) *SECOM CO., LTD.* · C. Wilson *Apple, Inc.* · C. Bonnell *DigiCert, Inc.* · S. Turner *sn3rd*

RFC 9919
Updates to the Lightweight Online Certificate Status Protocol (OCSP) Profile for High Volume Environments

Abstract

This specification defines a profile of the Online Certificate Status Protocol (OCSP) that addresses the scalability issues inherent when using OCSP in large scale (high volume) Public Key Infrastructure (PKI) environments and/or in PKI environments that require a lightweight solution to minimize communication bandwidth and client- side processing.

This specification obsoletes RFC 5019. The profile specified in RFC 5019 has been updated to allow and recommend the use of SHA-256 over SHA-1.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions 4
3. OCSP Message Profile 4
  3.1. OCSP Request Profile 5
    3.1.1. OCSPRequest Structure 5
    3.1.2. Signed OCSPRequests 6
  3.2. OCSP Response Profile 6
    3.2.1. OCSPResponse Structure 6
    3.2.2. Signed OCSPResponses 7
    3.2.3. OCSPResponseStatus Values 8
    3.2.4. thisUpdate, nextUpdate, and producedAt 8
4. Client Behavior 9
  4.1. OCSP Responder Discovery 9
  4.2. Sending an OCSP Request 9
5. Ensuring an OCSPResponse Is Fresh 9
6. Transport Profile 10
7. Caching Recommendations 11
  7.1. Caching at the Client 11
  7.2. HTTP Proxies 11
  7.3. Caching at Servers 13
8. Security Considerations 13
  8.1. Replay Attacks 14
  8.2. Man-in-the-Middle Attacks 14
  8.3. Impersonation Attacks 14

8.4. Denial-of-Service Attacks 14
8.5. Modification of HTTP Header Fields 15
8.6. Request Authentication and Authorization 15
8.7. Use of SHA-1 for the Calculation of CertID Field Values 15
9. IANA Considerations 15
10. References 15
10.1. Normative References 15
10.2. Informative References 16
Appendix A. Differences from RFC 5019 17
Appendix B. Examples 17
B.1. Root Certification Authority Certificate 17
B.2. End-Entity Certificate 20
B.3. OCSP Responder Certificate 22
B.4. OCSP Request 26
B.5. OCSP Response 27
Acknowledgments 31
Authors' Addresses 32

1. Introduction

The Online Certificate Status Protocol [RFC6960] specifies a mechanism used to determine the status of digital certificates, in lieu of using Certificate Revocation Lists (CRLs). Since its definition in 1999, it has been deployed in a variety of environments and has proven to be a useful certificate status checking mechanism. (For brevity, the term "OCSP" is used herein to denote the verification of certificate status; however, it should be noted that this protocol is employed solely to ascertain the revocation status of a certificate.)

To date, numerous OCSP deployments have been implemented to provide timely and secure certificate status information, crucial for high-value electronic transactions and the handling of highly sensitive information, such as within the banking and financial sectors. Therefore, the requirement for an OCSP responder to respond in "real time" (i.e., generating a new OCSP response for each OCSP request) has been important. In addition, these deployments have operated in environments where bandwidth usage is not an issue and have run on client and server systems where processing power is not constrained.

As the use of PKI continues to grow and move into diverse environments, so does the need for a scalable and cost-effective certificate status mechanism. Although OCSP as currently defined and deployed meets the need of small to medium-sized PKIs that operate on powerful systems on wired networks, there is a limit as to how these OCSP deployments scale from both an efficiency and cost perspective. Mobile environments, where network bandwidth may be at a premium and client-side devices are constrained from a processing point of view, require the careful use of OCSP to minimize bandwidth usage and client-side processing complexity [OCSPMP].

PKI continues to be deployed into environments where millions if not hundreds of millions of certificates have been issued. In many of these environments, an even larger number of users (also known as relying parties) have the need to ensure that the certificate they are relying upon has not been revoked. As such, it is important that OCSP is used in such a way that ensures the load on OCSP responders and the network infrastructure required to host those responders are kept to a minimum.

This document addresses the scalability issues inherent when using OCSP in highly scaled PKI environments by defining a message profile and clarifying OCSP client and responder behavior that will permit:

1. OCSP response pre-production and distribution.
2. Reduced OCSP message size to lower bandwidth usage.
3. Response message caching both in the network and on the client.

It is intended that the normative requirements defined in this profile will be adopted by OCSP clients and OCSP responders operating in very large-scale (high-volume) PKI environments or PKI environments that require a lightweight solution to minimize bandwidth and client-side processing power (or both), as described above.

OCSP does not have the means to signal responder capabilities within the protocol. Thus, clients may need to use out-of-band mechanisms (e.g., agreed upon arrangements between operators of OCSP responders and OCSP clients) to determine whether a responder conforms to the profile defined in this document. Regardless of the availability of such out-of-band mechanisms, this profile ensures that interoperability will still occur between an OCSP client that fully conforms with [RFC6960] and a responder that is operating in a mode as described in this specification.

2. Conventions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. OCSP Message Profile

This section defines a subset of OCSPRequest and OCSPResponse functionality as defined in [RFC6960].

3.1. OCSP Request Profile

3.1.1. OCSPRequest Structure

A partial extract of the ASN.1 structure corresponding to the OCSPRequest with the relevant CertID as defined in [RFC6960] is provided here for convenience:

```
OCSPRequest     ::=     SEQUENCE {
   tbsRequest                  TBSRequest,
   optionalSignature   [0]     EXPLICIT Signature OPTIONAL }

TBSRequest      ::=     SEQUENCE {
   version             [0]     EXPLICIT Version DEFAULT v1,
   requestorName       [1]     EXPLICIT GeneralName OPTIONAL,
   requestList                 SEQUENCE OF Request,
   requestExtensions   [2]     EXPLICIT Extensions OPTIONAL }

Request         ::=     SEQUENCE {
   reqCert                     CertID,
   singleRequestExtensions     [0] EXPLICIT Extensions OPTIONAL }

CertID          ::=     SEQUENCE {
   hashAlgorithm       AlgorithmIdentifier,
   issuerNameHash      OCTET STRING, -- Hash of issuer's DN
   issuerKeyHash       OCTET STRING, -- Hash of issuer's public key
   serialNumber        CertificateSerialNumber }
```

OCSPRequests that conform to the profile in this document **MUST** include only one Request in the OCSPRequest.RequestList structure.

The CertID.issuerNameHash and CertID.issuerKeyHash fields contain hashes of the issuer's distinguished name (DN) and public key, respectively. OCSP clients that conform with this profile **MUST** use SHA-256, as defined in Section 2.2 of [RFC5754], as the hashing algorithm for the CertID.issuerNameHash and the CertID.issuerKeyHash values.

Older OCSP clients that provide backward compatibility with [RFC5019] use SHA-1, as defined in [RFC3174], as the hashing algorithm for the CertID.issuerNameHash and the CertID.issuerKeyHash values. However, these OCSP clients **MUST** transition from SHA-1 to SHA-256 as soon as practical.

Clients **MUST NOT** include the singleRequestExtensions structure.

Clients **SHOULD NOT** include the requestExtensions structure. If a requestExtensions structure is included, it is **RECOMMENDED** by this profile that the structure contain only the nonce extension (id-pkix-ocsp-nonce). See Section 5 for issues concerning the use of a nonce in high-volume OCSP environments.

3.1.2. Signed OCSPRequests

Clients **SHOULD NOT** send signed OCSPRequests. Responders **MAY** ignore the signature on OCSPRequests.

If the OCSPRequest is signed, the client **SHALL** specify its name in the OCSPRequest.requestorName field; otherwise, clients **SHOULD NOT** include the requestorName field in the OCSPRequest. OCSP responders **MUST** handle unsigned OCSP requests that contain the requestorName field, as if the requestorName field were absent.

3.2. OCSP Response Profile

3.2.1. OCSPResponse Structure

A partial extract of the ASN.1 structure corresponding to the OCSPResponse with the relevant CertID as defined in [RFC6960] is provided here for convenience:

```
OCSPResponse ::= SEQUENCE {
   responseStatus         OCSPResponseStatus,
   responseBytes          [0] EXPLICIT ResponseBytes OPTIONAL }

ResponseBytes ::=       SEQUENCE {
   responseType   OBJECT IDENTIFIER,
   response       OCTET STRING }

The value for response SHALL be the DER encoding of
BasicOCSPResponse.

BasicOCSPResponse       ::= SEQUENCE {
   tbsResponseData      ResponseData,
   signatureAlgorithm   AlgorithmIdentifier,
   signature            BIT STRING,
   certs            [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL }

ResponseData ::= SEQUENCE {
   version              [0] EXPLICIT Version DEFAULT v1,
   responderID              ResponderID,
   producedAt               GeneralizedTime,
   responses                SEQUENCE OF SingleResponse,
   responseExtensions   [1] EXPLICIT Extensions OPTIONAL }

SingleResponse ::= SEQUENCE {
   certID                       CertID,
   certStatus                   CertStatus,
   thisUpdate                   GeneralizedTime,
   nextUpdate         [0]       EXPLICIT GeneralizedTime OPTIONAL,
   singleExtensions   [1]       EXPLICIT Extensions OPTIONAL }
```

Responders **MUST** generate a BasicOCSPResponse as identified by the id-pkix-ocsp-basic OID. Clients **MUST** be able to parse and accept a BasicOCSPResponse. OCSPResponses that conform to this profile **SHOULD** include only one SingleResponse in the ResponseData.responses structure but **MAY** include additional SingleResponse elements if necessary to improve response pregeneration performance or cache efficiency and to ensure backward compatibility. For instance, to provide support to OCSP clients that do not yet support the use of SHA-256 for CertID hash calculation, the OCSP responder **MAY** include two SingleResponses in a BasicOCSPResponse. In that BasicOCSPResponse, the CertID of one of the SingleResponses uses SHA-1 for the hash calculation, and the CertID in the other SingleResponse uses SHA-256. OCSP responders **SHOULD NOT** distribute OCSP responses that contain CertIDs that use SHA-1 if the OCSP responder has no clients that require the use of SHA-1. Operators of OCSP responders may consider logging the hash algorithm used by OCSP clients to inform their determination of when it is appropriate to obsolete the distribution of OCSP responses that employ SHA-1 for CertID field hashes. See Section 8.7 for more information on the security considerations for the continued use of SHA-1.

The responder **SHOULD NOT** include responseExtensions. As specified in [RFC6960], clients **MUST** ignore unrecognized non-critical responseExtensions in the response.

In the case where a responder does not have the ability to respond to an OCSP request containing an option not supported by the responder, it **SHOULD** return the most complete response it can. For example, in the case where a responder only supports pre-produced responses and does not have the ability to respond to an OCSP request containing a nonce, it **SHOULD** return a response that does not include a nonce.

Clients **SHOULD** attempt to process a response even if the response does not include a nonce. See Section 5 for details on validating responses that do not contain a nonce. See also Section 8 for relevant security considerations.

Responders that do not have the ability to respond to OCSP requests that contain an unsupported option such as a nonce **MAY** forward the request to an OCSP responder capable of doing so.

The responder **MAY** include the singleResponse.singleResponse extensions structure.

3.2.2. Signed OCSPResponses

Clients **MUST** validate the signature on the OCSPResponse.

If the response is signed by a delegate of the issuing certification authority (CA), a valid responder certificate **MUST** be referenced in the BasicOCSPResponse.certs structure.

It is **RECOMMENDED** that the OCSP responder's certificate contain the id-pkix-ocsp-nocheck extension, as defined in [RFC6960], to indicate to the client that it need not check the certificate's status. In addition, it is **RECOMMENDED** that neither an OCSP authorityInfoAccess (AIA) extension nor cRLDistributionPoints (CRLDP) extension be included in the OCSP responder's certificate. Accordingly, the responder's signing certificate **SHOULD** be relatively short-lived and renewed regularly.

Clients **MUST** be able to identify OCSP responder certificates using the byKey field and **SHOULD** be able to identify OCSP responder certificates using the byName field of the ResponseData.ResponderID [RFC6960] choices.

Older responders that provide backward compatibility with [RFC5019] **MAY** use the byName field to represent the ResponderID but should transition to using the byKey field as soon as practical.

Newer responders that conform to this profile **MUST** use the byKey field to represent the ResponderID to reduce the size of the response.

3.2.3. OCSPResponseStatus Values

As long as the OCSP infrastructure has authoritative records for a particular certificate, an OCSPResponseStatus of "successful" will be returned. When access to authoritative records for a particular certificate is not available, the responder **MUST** return an OCSPResponseStatus of "unauthorized". As such, this profile extends the [RFC6960] definition of "unauthorized" as follows:

The response "unauthorized" is returned in cases where the client is not authorized to make this query to this responder or the responder is not capable of responding authoritatively.

For example, OCSP responders that do not have access to authoritative records for a requested certificate, such as those that generate and distribute OCSP responses in advance and thus do not have the ability to properly respond with a signed "successful" yet "unknown" response, will respond with an OCSPResponseStatus of "unauthorized". Also, in order to ensure the database of revocation information does not grow unbounded over time, the responder **MAY** remove the status records of expired certificates. Requests from clients for certificates whose record has been removed will result in an OCSPResponseStatus of "unauthorized".

Security considerations regarding the use of unsigned responses are discussed in [RFC6960].

3.2.4. thisUpdate, nextUpdate, and producedAt

When pre-producing OCSPResponse messages, the responder **MUST** set the thisUpdate, nextUpdate, and producedAt times as follows:

thisUpdate: The time at which the status being indicated is known to be correct.

nextUpdate: The time at or before which newer information will be available about the status of the certificate. As described in Section 2.4 of [RFC6960], this field is optional. However, this field **MUST** be included in the profile specified in this document to help clients cache responses. See Section 7 for additional information on caching.

producedAt: The time at which the OCSP response was signed.

> Note: The values of thisUpdate, nextUpdate, and producedAt are set as described in Section 2.5 of [RFC6960], and in many cases, the value of thisUpdate and producedAt are the same.

For the purposes of this profile, ASN.1-encoded GeneralizedTime values, such as thisUpdate, nextUpdate, and producedAt, **MUST** be expressed Greenwich Mean Time (Zulu) and **MUST** include seconds (i.e., times are YYYYMMDDHHMMSSZ), even where the number of seconds is zero. GeneralizedTime values **MUST NOT** include fractional seconds.

4. Client Behavior

4.1. OCSP Responder Discovery

Clients **MUST** support the authorityInfoAccess extension as defined in [RFC5280] and **MUST** recognize the id-ad-ocsp access method. This enables CAs to inform clients how they can contact the OCSP service.

In the case where a client is checking the status of a certificate that contains both an authorityInformationAccess (AIA) extension pointing to an OCSP responder and a cRLDistributionPoints extension pointing to a CRL, the client **SHOULD** attempt to contact the OCSP responder first. Clients **MAY** attempt to retrieve the CRL if no OCSPResponse is received from the responder after a locally configured timeout and number of retries.

4.2. Sending an OCSP Request

To avoid needless network traffic, applications **MUST** verify the signature of signed data before asking an OCSP client to check the status of certificates used to verify the data. If the signature is invalid or the application is not able to verify it, an OCSP check **MUST NOT** be requested.

Similarly, an application **MUST** validate the signature on certificates in a chain before asking an OCSP client to check the status of the certificate. If the certificate signature is invalid or the application is not able to verify it, an OCSP check **MUST NOT** be requested. Clients **SHOULD NOT** make a request to check the status of expired certificates.

5. Ensuring an OCSPResponse Is Fresh

In order to ensure that a client does not accept an out-of-date response that indicates a "good" status when in fact there is a more up-to-date response that specifies the status of "revoked", a client must ensure the responses they receive are fresh.

In general, two mechanisms are available to clients to ensure a response is fresh. The first uses nonces, and the second is based on time. In order for time-based mechanisms to work, both clients and responders **MUST** have access to an accurate source of time.

Because this profile specifies that clients **SHOULD NOT** include a requestExtensions structure in OCSPRequests (see Section 3.1), clients **MUST** be able to determine OCSPResponse freshness based on an accurate source of time. Clients that opt to include a nonce in the request **SHOULD NOT** reject a corresponding OCSPResponse solely on the basis of the nonexistent expected nonce but **MUST** fall back to validating the OCSPResponse based on time.

Clients that do not include a nonce in the request **MUST** ignore any nonce that may be present in the response.

Clients **MUST** check for the existence of the nextUpdate field and **MUST** ensure the current time, expressed in GMT time as described in Section 3.2.4, falls between the thisUpdate and nextUpdate times. If the nextUpdate field is absent, the client **MUST** reject the response.

If the nextUpdate field is present, the client **MUST** ensure that it is not earlier than the current time. If the current time on the client is later than the time specified in the nextUpdate field, the client **MUST** reject the response as stale. Clients **MAY** allow configuration of a small tolerance period for acceptance of responses after nextUpdate to handle minor clock differences relative to responders and caches. This tolerance period should be chosen based on the accuracy and precision of time synchronization technology available to the calling application environment. For example, Internet peers with low latency connections typically expect NTP time synchronization to keep them accurate within parts of a second; higher latency environments or where an NTP analogue is not available may have to be more liberal in their tolerance (e.g., allow one day difference).

See the security considerations in Section 8 for additional details on replay and man-in-the-middle attacks.

6. Transport Profile

OCSP clients can send HTTP-based OCSP requests using either the GET or POST method. The OCSP responder **MUST** support requests and responses over HTTP. When sending requests that are less than or equal to 255 bytes in total (after encoding) including the scheme and delimiters (http://), server name and base64-encoded OCSPRequest structure, clients **MUST** use the GET method (to enable OCSP response caching). OCSP requests larger than 255 bytes **SHOULD** be submitted using the POST method. In all cases, clients **MUST** follow the descriptions in Appendix A.1 of [RFC6960] when constructing these messages.

When constructing a GET message, OCSP clients **MUST** base64-encode the OCSPRequest structure according to Section 4 of [RFC4648]. Clients **MUST NOT** include whitespace or any other characters that are not part of the base64 character repertoire in the base64-encoded string. Clients **MUST** properly URL-encode the base64-encoded OCSPRequest according to [RFC3986]. OCSP clients **MUST** append the base64-encoded OCSPRequest to the URI specified in the AIA extension [RFC5280]. For example:

```
http://ocsp.example.com/MEowSDBGMEQwQjAKBggqhkiG9w0CBQQQ7sp6GTKpL2dA
deGaW267owQQqInESWQD0mGeBArSgv%2FBWQIQLJx%2Fg9xF8oySYzol80Mbpg%3D%3D
```

In response to properly formatted OCSPRequests that are cachable (i.e., responses that contain a nextUpdate value), the responder will include the binary value of the DER encoding of the OCSPResponse preceded by the following HTTP [RFC9110] [RFC9111] header fields.

```
Content-type: application/ocsp-response
Content-length: < OCSP response length >
Last-modified: < producedAt HTTP-date >
ETag: "< strong validator >"
Expires: < nextUpdate HTTP-date >
Cache-control: max-age=< n >, public, no-transform, must-revalidate
Date: < current HTTP-date >
```

See Section 7.2 for details on the use of these HTTP header fields.

7. Caching Recommendations

The ability to cache OCSP responses throughout the network is an important factor in high volume OCSP deployments. This section discusses the recommended caching behavior of OCSP clients and HTTP proxies and the steps that should be taken to minimize the number of times that OCSP clients "hit the wire". In addition, the concept of including OCSP responses in protocol exchanges (aka stapling or piggybacking), such as has been defined in TLS, is also discussed.

7.1. Caching at the Client

To minimize bandwidth usage, clients **MUST** locally cache authoritative OCSP responses (i.e., a response with a signature that has been successfully validated and that indicates an OCSPResponseStatus of "successful").

Most OCSP clients will send OCSPRequests at or near the nextUpdate time (when a cached response expires). To avoid large spikes in responder load that might occur when many clients refresh cached responses for a popular certificate, responders **MAY** indicate when the client should fetch an updated OCSP response by using the cache- control:max-age directive. Clients **SHOULD** fetch the updated OCSP response on or after the max-age time. To ensure that clients receive an updated OCSP response, OCSP responders **MUST** refresh the OCSP response before the max-age time.

7.2. HTTP Proxies

The responder **SHOULD** set the HTTP header fields of the OCSP response in such a way as to allow for the intelligent use of intermediate HTTP proxy servers. See [RFC9110] and [RFC9111] for the full definition of these HTTP header fields and the proper format of any date and time values.

HTTP Header Field	Description
Date	The date and time at which the OCSP responder generated the HTTP response.

HTTP Header Field	Description
Last-Modified	This value specifies the date and time at which the OCSP responder last modified the response. This date and time will be the same as the thisUpdate timestamp in the request itself.
Expires	Specifies how long the response is considered fresh. This date and time will be the same as the nextUpdate timestamp in the OCSP response itself.
ETag	A string that identifies a particular version of the associated data. It is **RECOMMENDED** by this profile that the ETag value be the ASCII HEX representation of the SHA-256 hash of the OCSPResponse structure.
Cache-Control	Contains a number of caching directives. • max-age = < n > - where n is a time value later than thisUpdate but earlier than nextUpdate. • public - makes normally uncachable response cachable by both shared and nonshared caches. • no-transform - specifies that a proxy cache cannot change the type, length, or encoding of the object content. • must-revalidate - prevents caches from intentionally returning stale responses.

Table 1: HTTP Header Fields

OCSP responders **MUST NOT** include the "Pragma: no-cache", "Cache- Control: no-cache", or "Cache-Control: no-store" HTTP header fields in authoritative OCSP responses.

OCSP responders **SHOULD** include one or more of these HTTP header fields in non-authoritative OCSP responses.

For example, assume that an OCSP response has the following timestamp values:

```
    thisUpdate = March 19, 2023 01:00:00 GMT
    nextUpdate = March 21, 2023 01:00:00 GMT
    productedAt = March 19, 2023 01:00:00 GMT
```

and that an OCSP client requests the response on March 20, 2023 01:00:00 GMT. In this scenario, the HTTP response may look like this:

```
    Content-Type: application/ocsp-response
    Content-Length: 1000
    Date: Mon, 20 Mar 2023 01:00:00 GMT
    Last-Modified: Sun, 19 Mar 2023 01:00:00 GMT
    ETag: "97df3588b5a3f24babc3851b372f0ba7
          1a9dcdded43b14b9d06961bfc1707d9d"
    Expires: Tue, 21 Mar 2023 01:00:00 GMT
    Cache-Control: max-age=86000,public,no-transform,must-revalidate
    <...>
```

OCSP clients **MUST NOT** include a no-cache HTTP header field in OCSP request messages, unless the client encounters an expired response, which may be a result of an intermediate proxy caching stale data. In this situation, clients **SHOULD** resend the request specifying that proxies should be bypassed by including an appropriate HTTP header field in the request (i.e., Pragma: no-cache or Cache-Control: no-cache).

7.3. Caching at Servers

In some scenarios, it is advantageous to include OCSP response information within the protocol being utilized between the client and OCSP responder. Including OCSP responses in this manner has a few attractive effects.

First, it allows for the caching of OCSP responses on the OCSP responder, thus lowering the number of hits.

Second, it enables certificate validation in the event the client is not connected to a network and thus eliminates the need for clients to establish a new HTTP session with the OCSP responder.

Third, it reduces the number of round trips the client needs to make in order to complete a handshake.

Fourth, it simplifies the client-side OCSP implementation by enabling a situation where the client need only the ability to parse and recognize OCSP responses.

This functionality has been specified as an extension to the TLS protocol [RFC9846] in Section 4.4.2 of [RFC9846] but can be applied to any client-server protocol.

It is **RECOMMENDED** by this profile that both TLS clients and servers implement the certificate status request extension mechanism for TLS.

Further information regarding caching issues can be obtained from [RFC3143].

8. Security Considerations

The following considerations apply in addition to the security considerations addressed in Section 5 of [RFC6960].

8.1. Replay Attacks

Because the use of nonces in this profile is optional, there is a possibility that an out-of-date OCSP response could be replayed, thus causing a client to accept a good response when in fact there is a more up-to-date response that specifies the status of "revoked". In order to mitigate this attack, clients **MUST** have access to an accurate source of time and ensure that the OCSP responses they receive are sufficiently fresh.

Clients that do not have an accurate source of date and time are vulnerable to service disruption. For example, a client with a sufficiently fast clock may reject a fresh OCSP response. Similarly, a client with a sufficiently slow clock may incorrectly accept expired valid responses for certificates that may in fact be revoked.

Future versions of the OCSP protocol may provide a way for the client to know whether the responder supports nonces or does not support nonces. If a client can determine that the responder supports nonces, it **MUST** reject a reply that does not contain an expected nonce. Otherwise, clients that opt to include a nonce in the request **SHOULD NOT** reject a corresponding OCSPResponse solely on the basis of the nonexistent expected nonce but **MUST** fall back to validating the OCSPResponse based on time.

8.2. Man-in-the-Middle Attacks

To mitigate risk associated with this class of attack, the client **MUST** properly validate the signature on the response.

The use of signed responses in OCSP serves to authenticate the identity of the OCSP responder and to verify that it is authorized to sign responses on the CA's behalf.

Clients **MUST** ensure that they are communicating with an authorized responder by the rules described in Section 4.2.2.2 of [RFC6960].

8.3. Impersonation Attacks

The use of signed responses in OCSP serves to authenticate the identity of OCSP responder.

As detailed in [RFC6960], clients must properly validate the signature of the OCSP response and the signature on the OCSP response signer certificate to ensure an authorized responder created it.

8.4. Denial-of-Service Attacks

OCSP responders **SHOULD** take measures to prevent or mitigate denial- of-service attacks. As this profile specifies the use of unsigned OCSPRequests, access to the responder may be implicitly given to everyone who can send a request to a responder, and thus the ability to mount a denial-of-service attack via a flood of requests may be greater. For example, a responder could limit the rate of incoming requests from a particular IP address if questionable behavior is detected.

8.5. Modification of HTTP Header Fields

Values included in HTTP header fields, as described in Sections 6 and 7, are not cryptographically protected; they may be manipulated by an attacker. Clients **SHOULD** use these values for caching guidance only and ultimately **SHOULD** rely only on the values present in the signed OCSPResponse (Section 4.2.2.1 of [RFC6960]). Clients **SHOULD NOT** rely on cached responses beyond the nextUpdate time.

8.6. Request Authentication and Authorization

The suggested use of unsigned requests in this environment removes an option that allows the responder to determine the authenticity of incoming requests. Thus, access to the responder may be implicitly given to everyone who can send a request to a responder. Environments where explicit authorization to access the OCSP responder is necessary can utilize other mechanisms to authenticate requestors or restrict or meter service.

8.7. Use of SHA-1 for the Calculation of CertID Field Values

Although the use of SHA-1 for the calculation of CertID field values is not of concern from a cryptographic security standpoint, the continued use of SHA-1 in an ecosystem requires that software that interoperates with the ecosystem maintain support for SHA-1. This increases implementation complexity and potential attack surface for the software in question. Thus, the continued use of SHA-1 in an ecosystem to maintain interoperability with legacy software must be weighed against the increased implementation complexity and potential attack surface.

9. IANA Considerations

This document has no IANA actions.

10. References

10.1. Normative References

[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>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC5019] Deacon, A. and R. Hurst, "The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments", RFC 5019, DOI 10.17487/RFC5019, September 2007, <https://www.rfc-editor.org/info/rfc5019>.

[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>.

[RFC5754] Turner, S., "Using SHA2 Algorithms with Cryptographic Message Syntax", RFC 5754, DOI 10.17487/RFC5754, January 2010, <https://www.rfc-editor.org/info/rfc5754>.

[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, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

[RFC9111] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Caching", STD 98, RFC 9111, DOI 10.17487/RFC9111, June 2022, <https://www.rfc-editor.org/info/rfc9111>.

[RFC9846] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 9846, DOI 10.17487/RFC9846, January 2026, <https://www.rfc-editor.org/info/rfc9846>.

10.2. Informative References

[OCSPMP] Open Mobile Alliance, "OCSP Mobile Profile V1.0", <https://www.openmobilealliance.org>.

[RFC3143] Cooper, I. and J. Dilley, "Known HTTP Proxy/Caching Problems", RFC 3143, DOI 10.17487/RFC3143, June 2001, <https://www.rfc-editor.org/info/rfc3143>.

[RFC3174] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001, <https://www.rfc-editor.org/info/rfc3174>.

[RFC9500] Gutmann, P. and C. Bonnell, "Standard Public Key Cryptography (PKC) Test Keys", RFC 9500, DOI 10.17487/RFC9500, December 2023, <https://www.rfc-editor.org/info/rfc9500>.

Appendix A. Differences from RFC 5019

This document obsoletes [RFC5019]. [RFC5019] defines a lightweight profile for OCSP that makes the protocol more suitable for use in high-volume environments. The lightweight profile specifies the mandatory use of SHA-1 when calculating the values of several fields in OCSP requests and responses. In recent years, weaknesses have been demonstrated with the SHA-1 algorithm. As a result, SHA-1 is increasingly falling out of use even for non-security-relevant use cases. This document obsoletes the lightweight profile as specified in [RFC5019] to instead recommend the use of SHA-256 where SHA-1 was previously required. An OCSP client compliant with [RFC5019] is still able to use SHA-1, but the use of SHA-1 may become obsolete in the future.

Substantive changes to RFC 5019:

- Section 3.1.1 requires new OCSP clients to use SHA-256 to support migration for OCSP clients.
- Section 3.2.2 requires new OCSP responders to use the byKey field and support migration from byName fields.
- Section 6 clarifies that OCSP clients **MUST NOT** include whitespace or any other characters that are not part of the base64 character repertoire in the base64-encoded string.

Appendix B. Examples

B.1. Root Certification Authority Certificate

This is a self-signed certificate for the certification authority that issued the end-entity certificate and OCSP-delegated responder example certificates below.

The key pair for the certification authority is the "testECCP521" key from Section 2.3 of [RFC9500].

```
-----BEGIN CERTIFICATE-----
MIICKDCCAYqgAwIBAgIBATAKBggqhkjOPQQDBDA4MQswCQYDVQQGEwJYWDEUMBIG
A1UECgwLQ2VydHMgJ3IgVXMxEzARBgNVBAMMCklzc3VpbmcgQ0EwHhcNMjQwNDAy
MTIzNzQ3WhcNMjUwNDAyMTIzNzQ3WjA4MQswCQYDVQQGEwJYWDEUMBIGA1UECgwL
Q2VydHMgJ3IgVXMxEzARBgNVBAMMCklzc3VpbmcgQ0EwgZswEAYHKoZIzj0CAQYF
K4EEACMDgYYABAHQ/XJXqEx0f1YldcBzhdvr8vUr6lgIPbgv3RUx2KrjzIdf8C/3
+i2iYNjrYtbS9dZJJ44yFzagYoy7swMItuYY2wD2KtIExkYDWbyBiriWG/Dw/A7F
quikKBc85W8A3psVfB5cgsZPVi/K3vxKTCj200LPPvYW/ILTO3KFySHyvzb92KNC
MEAwHQYDVR0OBBYEFI7CFAlgduqQOOk5rhttUsQXfZ++MA8GA1UdEwEB/wQFMAMB
Af8wDgYDVR0PAQH/BAQDAgIEMAoGCCqGSM49BAMEA4GLADCBhwJBbr/1SJiHCgXG
EJ7R+3er1LdWqrdZHgtCwyT7+wFBIJmVswEiom2LGh/oMuu5mD+u/+o1m07vmmZj
/+ipGp8TIwkCQgCoZ4bHte6XkFm7hUXascLN7vkv7qKwXyTsCvIDpEDTRCX8dUFe
73jGebitkumRHjVhlBJLo7n3FMJrFHNoeblMbw==
-----END CERTIFICATE-----
```

```
0 552: SEQUENCE {
  4 394:   SEQUENCE {
  8   3:     [0] {
 10   1:       INTEGER 2
       :       }
 13   1:     INTEGER 1
 16  10:     SEQUENCE {
 18   8:       OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :       }
 28  56:     SEQUENCE {
 30  11:       SET {
 32   9:         SEQUENCE {
 34   3:           OBJECT IDENTIFIER countryName (2 5 4 6)
 39   2:           PrintableString 'XX'
       :           }
       :         }
 43  20:       SET {
 45  18:         SEQUENCE {
 47   3:           OBJECT IDENTIFIER organizationName (2 5 4 10)
 52  11:           UTF8String 'Certs 'r Us'
       :           }
       :         }
 65  19:       SET {
 67  17:         SEQUENCE {
 69   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 74  10:           UTF8String 'Issuing CA'
       :           }
       :         }
       :       }
 86  30:     SEQUENCE {
 88  13:       UTCTime 02/04/2024 12:37:47 GMT
103  13:       UTCTime 02/04/2025 12:37:47 GMT
       :       }
118  56:     SEQUENCE {
120  11:       SET {
122   9:         SEQUENCE {
124   3:           OBJECT IDENTIFIER countryName (2 5 4 6)
129   2:           PrintableString 'XX'
       :           }
       :         }
133  20:       SET {
135  18:         SEQUENCE {
137   3:           OBJECT IDENTIFIER organizationName (2 5 4 10)
142  11:           UTF8String 'Certs 'r Us'
       :           }
       :         }
155  19:       SET {
157  17:         SEQUENCE {
159   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
164  10:           UTF8String 'Issuing CA'
       :           }
       :         }
       :       }
176 155:     SEQUENCE {
179  16:       SEQUENCE {
181   7:         OBJECT IDENTIFIER ecPublicKey (1 2 840 10045 2 1)
190   5:         OBJECT IDENTIFIER secp521r1 (1 3 132 0 35)
```

```
       :          }
 197 134:        BIT STRING
       :          04 01 D0 FD 72 57 A8 4C 74 7F 56 25 75 C0 73 85
       :          DB EB F2 F5 2B EA 58 08 3D B8 2F DD 15 31 D8 AA
       :          E3 CC 87 5F F0 2F F7 FA 2D A2 60 D8 EB 62 D6 D2
       :          F5 D6 49 27 8E 32 17 36 A0 62 8C BB B3 03 08 B6
       :          E6 18 DB 00 F6 2A D2 04 C6 46 03 59 BC 81 8A B8
       :          96 1B F0 F0 FC 0E C5 AA E8 A4 28 17 3C E5 6F 00
       :          DE 9B 15 7C 1E 5C 82 C6 4F 56 2F CA DE FC 4A 4C
       :          28 F6 D3 42 CF 3E F6 16 FC 82 D3 3B 72 85 C9 21
       :          F2 BF 36 FD D8
       :        }
 334  66:      [3] {
 336  64:        SEQUENCE {
 338  29:          SEQUENCE {
 340   3:            OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 345  22:            OCTET STRING, encapsulates {
 347  20:              OCTET STRING
       :                8E C2 14 09 60 76 EA 90 38 E9 39 AE 1B 6D 52 C4
       :                17 7D 9F BE
       :              }
       :            }
 369  15:          SEQUENCE {
 371   3:            OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 376   1:            BOOLEAN TRUE
 379   5:            OCTET STRING, encapsulates {
 381   3:              SEQUENCE {
 383   1:                BOOLEAN TRUE
       :                }
       :              }
       :            }
 386  14:          SEQUENCE {
 388   3:            OBJECT IDENTIFIER keyUsage (2 5 29 15)
 393   1:            BOOLEAN TRUE
 396   4:            OCTET STRING, encapsulates {
 398   2:              BIT STRING 2 unused bits
       :                '100000'B (bit 5)
       :              }
       :            }
       :          }
       :        }
       :      }
 402  10:    SEQUENCE {
 404   8:      OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :      }
 414 139:    BIT STRING, encapsulates {
 418 135:      SEQUENCE {
 421  65:        INTEGER
       :          6E BF F5 48 98 87 0A 05 C6 10 9E D1 FB 77 AB D4
       :          B7 56 AA B7 59 1E 0B 42 C3 24 FB FB 01 41 20 99
       :          95 B3 01 22 A2 6D 8B 1A 1F E8 32 EB B9 98 3F AE
       :          FF EA 35 9B 4E EF 9A 66 63 FF E8 A9 1A 9F 13 23
       :          09
 488  66:        INTEGER
       :          00 A8 67 86 C7 B5 EE 97 90 59 BB 85 45 DA B1 C2
       :          CD EE F9 2F EE A2 B0 5F 24 EC 0A F2 03 A4 40 D3
       :          44 25 FC 75 41 5E EF 78 C6 79 B8 AD 92 E9 91 1E
       :          35 61 94 12 4B A3 B9 F7 14 C2 6B 14 73 68 79 B9
```

```
         :          4C 6F
         :        }
         :      }
         :    }
```

B.2. End-Entity Certificate

This is an end-entity certificate whose status is requested and returned in the OCSP request and response examples below.

The key pair for the end-entity certificate is the "testECCP256" key from Section 2.3 of [RFC9500].

```
-----BEGIN CERTIFICATE-----
MIIB2zCCATygAwIBAgIEAarwDTAKBggqhkjOPQQDBDA4MQswCQYDVQQGEwJYWDEU
MBIGA1UECgwLQ2VydHMgJ3IgVXMxEzARBgNVBAMMCklzc3VpbmcgQ0EwHhcNMjQw
NDAyMTIzNzQ3WhcNMjUwNDAyMTIzNzQ3WjAcMRowGAYDVQQDDBF4bi0tMThqNGQu
ZXhhbXBsZTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABEIlSPiPt4L/teyjdERS
xyoeVY+9b3O+XkjpMjLMRcWxbEzRDEy41bihcTnpSILImSVymTQl9BQZq36QpCpJ
QnKjUDBOMB0GA1UdDgQWBBRbcKeYF/ef9jfS9+PcRGwhCde71DAfBgNVHSMEGDAW
gBSOwhQJYHbqkDjpOa4bbVLEF32fvjAMBgNVHRMBAf8EAjAAMAoGCCqGSM49BAME
A4GMADCBiAJCAIot8SYNFkScrcsY5T81HSmNzhP/0GC87N3WI849CN0qmNa0nMXW
8HnDKGR5nv/D9x+T8uLMBlpFUWmHQmXAJPN8AkIBW8A0XsiyPJyZfaZieODmtnoI
obZP+eTLNWkGUFL6uCtLtQmYtrXpLAJfvkE6WYVqCUl495Kx9l6M9TBLK5X6V3w=
-----END CERTIFICATE-----
```

```
0 475: SEQUENCE {
  4 316:   SEQUENCE {
  8   3:     [0] {
 10   1:       INTEGER 2
       :       }
 13   4:     INTEGER 27979789
 19  10:     SEQUENCE {
 21   8:       OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :       }
 31  56:     SEQUENCE {
 33  11:       SET {
 35   9:         SEQUENCE {
 37   3:           OBJECT IDENTIFIER countryName (2 5 4 6)
 42   2:           PrintableString 'XX'
       :           }
       :         }
 46  20:       SET {
 48  18:         SEQUENCE {
 50   3:           OBJECT IDENTIFIER organizationName (2 5 4 10)
 55  11:           UTF8String 'Certs 'r Us'
       :           }
       :         }
 68  19:       SET {
 70  17:         SEQUENCE {
 72   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 77  10:           UTF8String 'Issuing CA'
       :           }
       :         }
       :       }
 89  30:     SEQUENCE {
 91  13:       UTCTime 02/04/2024 12:37:47 GMT
106  13:       UTCTime 02/04/2025 12:37:47 GMT
       :       }
121  28:     SEQUENCE {
123  26:       SET {
125  24:         SEQUENCE {
127   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
132  17:           UTF8String 'xn--18j4d.example'
       :           }
       :         }
       :       }
151  89:     SEQUENCE {
153  19:       SEQUENCE {
155   7:         OBJECT IDENTIFIER ecPublicKey (1 2 840 10045 2 1)
164   8:         OBJECT IDENTIFIER prime256v1 (1 2 840 10045 3 1 7)
       :         }
174  66:       BIT STRING
       :         04 42 25 48 F8 8F B7 82 FF B5 EC A3 74 44 52 C7
       :         2A 1E 55 8F BD 6F 73 BE 5E 48 E9 32 32 CC 45 C5
       :         B1 6C 4C D1 0C 4C B8 D5 B8 A1 71 39 E9 48 82 C8
       :         99 25 72 99 34 25 F4 14 19 AB 7E 90 A4 2A 49 42
       :         72
       :       }
242  80:     [3] {
244  78:       SEQUENCE {
246  29:         SEQUENCE {
248   3:           OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
```

```
 253  22:           OCTET STRING, encapsulates {
 255  20:             OCTET STRING
        :               5B 70 A7 98 17 F7 9F F6 37 D2 F7 E3 DC 44 6C 21
        :               09 D7 BB D4
        :             }
        :           }
 277  31:         SEQUENCE {
 279   3:           OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 284  24:           OCTET STRING, encapsulates {
 286  22:             SEQUENCE {
 288  20:               [0]
        :               8E C2 14 09 60 76 EA 90 38 E9 39 AE 1B 6D 52 C4
        :               17 7D 9F BE
        :               }
        :             }
        :           }
 310  12:         SEQUENCE {
 312   3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 317   1:           BOOLEAN TRUE
 320   2:           OCTET STRING, encapsulates {
 322   0:             SEQUENCE {}
        :             }
        :           }
        :         }
        :       }
        :     }
 324  10:   SEQUENCE {
 326   8:     OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
        :     }
 336 140:   BIT STRING, encapsulates {
 340 136:     SEQUENCE {
 343  66:       INTEGER
        :         00 8A 2D F1 26 0D 16 44 9C AD CB 18 E5 3F 35 1D
        :         29 8D CE 13 FF D0 60 BC EC DD D6 23 CE 3D 08 DD
        :         2A 98 D6 B4 9C C5 D6 F0 79 C3 28 64 79 9E FF C3
        :         F7 1F 93 F2 E2 CC 06 5A 45 51 69 87 42 65 C0 24
        :         F3 7C
 411  66:       INTEGER
        :         01 5B C0 34 5E C8 B2 3C 9C 99 7D A6 62 78 E0 E6
        :         B6 7A 08 A1 B6 4F F9 E4 CB 35 69 06 50 52 FA B8
        :         2B 4B B5 09 98 B6 B5 E9 2C 02 5F BE 41 3A 59 85
        :         6A 09 49 78 F7 92 B1 F6 5E 8C F5 30 4B 2B 95 FA
        :         57 7C
        :       }
        :     }
        :   }
```

B.3. OCSP Responder Certificate

This is a certificate for the OCSP-delegated response that signed the OCSP response example below.

The key pair for the OCSP responder certificate is the "testECCP384" key from Section 2.3 of [RFC9500].

```
-----BEGIN CERTIFICATE-----
MIICSzCCAa6gAwIBAgIBATAKBggqhkjOPQQDBDA4MQswCQYDVQQGEwJYWDEUMBIG
A1UECgwLQ2VydHMgJ3IgVXMxEzARBgNVBAMMCklzc3VpbmcgQ0EwHhcNMjQwNDAy
MTIzNzQ3WhcNMjUwNDAyMTIzNzQ3WjA8MQswCQYDVQQGEwJYWDEUMBIGA1UECgwL
Q2VydHMgJ3IgVXMxFzAVBgNVBAMMDk9DU1AgUmVzcG9uZGVyMHYwEAYHKoZIzj0C
AQYFK4EEACIDYgAEWwkBuIUjKW65GdUP+hqcs3S8TUCVhigr/soRsdla27VHNK9X
C/grcijPImvPTCXdvP47GjrTlDDv92Ph1o0uFR2Rcgt3lbWNprNGOWE6j7m1qNpI
xnRxF/mRnoQk837Io4GHMIGEMB0GA1UdDgQWBBQK46D+ndQldpi163Lrygznvz31
8TAfBgNVHSMEGDAWgBSOwhQJYHbqkDjpOa4bbVLEF32fvjAMBgNVHRMBAf8EAjAA
MA4GA1UdDwEB/wQEAwIHgDATBgNVHSUEDDAKBggrBgEFBQcDCTAPBgkrBgEFBQcw
AQUEAgUAMAoGCCqGSM49BAMEA4GKADCBhgJBFCqM1gpsZcd0Zd8RW8H/+L4OIbTa
GtpT2QY0pd6JBw91lFqNCxj+F1k9XJrKSQAVVAa/b3JaZOsRrH6vihlO3MYCQUkL
C0mmLubTRDH2v+6A1aycIVKIpR3G6+PuaD2Um3PSF7FElkoU4NYkbl1SH/8FzbDy
/LCBhih25e7hAtyg/XsI
-----END CERTIFICATE-----
```

```
0 587: SEQUENCE {
  4 430:   SEQUENCE {
  8   3:     [0] {
 10   1:       INTEGER 2
       :       }
 13   1:     INTEGER 1
 16  10:     SEQUENCE {
 18   8:       OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :       }
 28  56:     SEQUENCE {
 30  11:       SET {
 32   9:         SEQUENCE {
 34   3:           OBJECT IDENTIFIER countryName (2 5 4 6)
 39   2:           PrintableString 'XX'
       :           }
       :         }
 43  20:       SET {
 45  18:         SEQUENCE {
 47   3:           OBJECT IDENTIFIER organizationName (2 5 4 10)
 52  11:           UTF8String 'Certs 'r Us'
       :           }
       :         }
 65  19:       SET {
 67  17:         SEQUENCE {
 69   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 74  10:           UTF8String 'Issuing CA'
       :           }
       :         }
       :       }
 86  30:     SEQUENCE {
 88  13:       UTCTime 02/04/2024 12:37:47 GMT
103  13:       UTCTime 02/04/2025 12:37:47 GMT
       :       }
118  60:     SEQUENCE {
120  11:       SET {
122   9:         SEQUENCE {
124   3:           OBJECT IDENTIFIER countryName (2 5 4 6)
129   2:           PrintableString 'XX'
       :           }
       :         }
133  20:       SET {
135  18:         SEQUENCE {
137   3:           OBJECT IDENTIFIER organizationName (2 5 4 10)
142  11:           UTF8String 'Certs 'r Us'
       :           }
       :         }
155  23:       SET {
157  21:         SEQUENCE {
159   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
164  14:           UTF8String 'OCSP Responder'
       :           }
       :         }
       :       }
180 118:     SEQUENCE {
182  16:       SEQUENCE {
184   7:         OBJECT IDENTIFIER ecPublicKey (1 2 840 10045 2 1)
193   5:         OBJECT IDENTIFIER secp384r1 (1 3 132 0 34)
```

```
       :         }
 200  98:       BIT STRING
        :         04 5B 09 01 B8 85 23 29 6E B9 19 D5 0F FA 1A 9C
        :         B3 74 BC 4D 40 95 86 28 2B FE CA 11 B1 D9 5A DB
        :         B5 47 34 AF 57 0B F8 2B 72 28 CF 22 6B CF 4C 25
        :         DD BC FE 3B 1A 3A D3 94 30 EF F7 63 E1 D6 8D 2E
        :         15 1D 91 72 0B 77 95 B5 8D A6 B3 46 39 61 3A 8F
        :         B9 B5 A8 DA 48 C6 74 71 17 F9 91 9E 84 24 F3 7E
        :         C8
        :       }
 300 135:     [3] {
 303 132:       SEQUENCE {
 306  29:         SEQUENCE {
 308   3:           OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
 313  22:           OCTET STRING, encapsulates {
 315  20:             OCTET STRING
        :               0A E3 A0 FE 9D D4 25 76 98 B5 EB 72 EB CA 0C E7
        :               BF 3D F5 F1
        :             }
        :           }
 337  31:         SEQUENCE {
 339   3:           OBJECT IDENTIFIER authorityKeyIdentifier (2 5 29 35)
 344  24:           OCTET STRING, encapsulates {
 346  22:             SEQUENCE {
 348  20:               [0]
        :               8E C2 14 09 60 76 EA 90 38 E9 39 AE 1B 6D 52 C4
        :               17 7D 9F BE
        :               }
        :             }
        :           }
 370  12:         SEQUENCE {
 372   3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
 377   1:           BOOLEAN TRUE
 380   2:           OCTET STRING, encapsulates {
 382   0:             SEQUENCE {}
        :             }
        :           }
 384  14:         SEQUENCE {
 386   3:           OBJECT IDENTIFIER keyUsage (2 5 29 15)
 391   1:           BOOLEAN TRUE
 394   4:           OCTET STRING, encapsulates {
 396   2:             BIT STRING 7 unused bits
        :               '1'B (bit 0)
        :             }
        :           }
 400  19:         SEQUENCE {
 402   3:           OBJECT IDENTIFIER extKeyUsage (2 5 29 37)
 407  12:           OCTET STRING, encapsulates {
 409  10:             SEQUENCE {
 411   8:               OBJECT IDENTIFIER ocspSigning (1 3 6 1 5 5 7 3 9)
        :               }
        :             }
        :           }
 421  15:         SEQUENCE {
 423   9:           OBJECT IDENTIFIER ocspNoCheck (1 3 6 1 5 5 7 48 1 5)
 434   2:           OCTET STRING, encapsulates {
 436   0:             NULL
        :             }
        :           }
        :         }
        :       }
        :     }
438  10:   SEQUENCE {
440   8:     OBJECT IDENTIFIER ecdsaWithSHA512 (1 2 840 10045 4 3 4)
        :     }
450 138:   BIT STRING, encapsulates {
454 134:     SEQUENCE {
457  65:       INTEGER
        :         14 2A 8C D6 0A 6C 65 C7 74 65 DF 11 5B C1 FF F8
        :         BE 0E 21 B4 DA 1A DA 53 D9 06 34 A5 DE 89 07 0F
        :         75 94 5A 8D 0B 18 FE 17 59 3D 5C 9A CA 49 00 15
        :         54 06 BF 6F 72 5A 64 EB 11 AC 7E AF 8A 19 4E DC
        :         C6
524  65:       INTEGER
        :         49 0B 0B 49 A6 2E E6 D3 44 31 F6 BF EE 80 D5 AC
        :         9C 21 52 88 A5 1D C6 EB E3 EE 68 3D 94 9B 73 D2
        :         17 B1 44 96 4A 14 E0 D6 24 6E 5D 52 1F FF 05 CD
        :         B0 F2 FC B0 81 86 28 76 E5 EE E1 02 DC A0 FD 7B
        :         08
        :       }
        :     }
        :   }
```

B.4. OCSP Request

This is a base64-encoded OCSP request for the end-entity certificate above.

```
MGEwXzBdMFswWTANBglghkgBZQMEAgEFAAQgOplGd1aAc6cHv95QGGNF5M1hNNsI
Xrqh0QQl8DtvCOoEIEdKbKMB8j3J9/cHhwThx/X8lucWdfbtiC56tlw/WEVDAgQB
qvAN
```

```
 0  97: SEQUENCE {
   2  95:   SEQUENCE {
   4  93:     SEQUENCE {
   6  91:       SEQUENCE {
   8  89:         SEQUENCE {
  10  13:           SEQUENCE {
  12   9:             OBJECT IDENTIFIER sha-256 (2 16 840 1 101 3 4 2 1)
  23   0:             NULL
        :             }
  25  32:           OCTET STRING
        :             3A 99 46 77 56 80 73 A7 07 BF DE 50 18 63 45 E4
        :             CD 61 34 DB 08 5E BA A1 D1 04 25 F0 3B 6F 08 EA
  59  32:           OCTET STRING
        :             47 4A 6C A3 01 F2 3D C9 F7 F7 07 87 04 E1 C7 F5
        :             FC 96 E7 16 75 F6 ED 88 2E 7A B6 5C 3F 58 45 43
  93   4:           INTEGER 27979789
        :           }
        :         }
        :       }
        :     }
        :   }
```

B.5. OCSP Response

This is a base64-encoded OCSP response for the end-entity certificate above.

```
MIIDnwoBAKCCA5gwggOUBgkrBgEFBQcwAQEEggOFMIIDgTCBsKIWBBQK46D+ndQl
dpi163Lrygznvz318RgPMjAyNDA0MDIxMjM3NDdaMIGEMIGBMFkwDQYJYIZIAWUD
BAIBBQAEIDqZRndWgHOnB7/eUBhjReTNYTTbCF66odEEJfA7bwjqBCBHSmyjAfI9
yff3B4cE4cf1/JbnFnX27YguerZcP1hFQwIEAarwDYAAGA8yMDI0MDQwMzEyMzc0
N1qgERgPMjAyNDA0MTAxMjM3NDdaMAoGCCqGSM49BAMDA2kAMGYCMQDRmVmiIb4D
m9yEXiv2XtoeQi6ftpjLmlBqqRIi+3htfF/OyjdHnFuh38cQKYqqrWYCMQDKiPct
Vu7SQs587d2ZBEHQH20j5AFiGGsbI1b3+C9ZK6NIzgD6DnWlDwpSfilEarOgggJT
MIICTzCCAkswggGuoAMCAQICAQEwCgYIKoZIzj0EAwQwODELMAkGA1UEBhMCWFgx
FDASBgNVBAoMC0NlcnRzICdyIFVzMRMwEQYDVQQDDApJc3N1aW5nIENBMB4XDTI0
MDQwMjEyMzc0N1oXDTI1MDQwMjEyMzc0N1owPDELMAkGA1UEBhMCWFgxFDASBgNV
BAoMC0NlcnRzICdyIFVzMRcwFQYDVQQDDA5PQ1NQIFJlc3BvbmRlcjB2MBAGByqG
SM49AgEGBSuBBAAiA2IABFsJAbiFIyluuRnVD/oanLN0vE1AlYYoK/7KEbHZWtu1
RzSvVwv4K3IozyJrz0wl3bz+Oxo605Qw7/dj4daNLhUdkXILd5W1jaazRjlhOo+5
tajaSMZ0cRf5kZ6EJPN+yKOBhzCBhDAdBgNVHQ4EFgQUCuOg/p3UJXaYtety68oM
57899fEwHwYDVR0jBBgwFoAUjsIUCWB26pA46TmuG21SxBd9n74wDAYDVR0TAQH/
BAIwADAOBgNVHQ8BAf8EBAMCB4AwEwYDVR0lBAwwCgYIKwYBBQUHAwkwDwYJKwYB
BQUHMAEFBAIFADAKBggqhkjOPQQDBAOBigAwgYYCQRQqjNYKbGXHdGXfEVvB//i+
DiG02hraU9kGNKXeiQcPdZRajQsY/hdZPVyaykkAFVQGv29yWmTrEax+r4oZTtzG
AkFJCwtJpi7m00Qx9r/ugNWsnCFSiKUdxuvj7mg9lJtz0hexRJZKFODWJG5dUh//
Bc2w8vywgYYoduXu4QLcoP17CA==
0 927: SEQUENCE {
  4   1:   ENUMERATED 0
  7 920:   [0] {
 11 916:     SEQUENCE {
 15   9:       OBJECT IDENTIFIER ocspBasic (1 3 6 1 5 5 7 48 1 1)
 26 901:       OCTET STRING, encapsulates {
 30 897:         SEQUENCE {
 34 176:           SEQUENCE {
 37  22:             [2] {
 39  20:               OCTET STRING
       :               0A E3 A0 FE 9D D4 25 76 98 B5 EB 72 EB CA 0C E7
       :               BF 3D F5 F1
       :               }
 61  15:             GeneralizedTime 02/04/2024 12:37:47 GMT
 78 132:             SEQUENCE {
 81 129:               SEQUENCE {
 84  89:                 SEQUENCE {
 86  13:                   SEQUENCE {
 88   9:                     OBJECT IDENTIFIER
       :                       sha-256 (2 16 840 1 101 3 4 2 1)
 99   0:                     NULL
       :                     }
101  32:                   OCTET STRING
       :               3A 99 46 77 56 80 73 A7 07 BF DE 50 18 63 45 E4
       :               CD 61 34 DB 08 5E BA A1 D1 04 25 F0 3B 6F 08 EA
135  32:                   OCTET STRING
       :               47 4A 6C A3 01 F2 3D C9 F7 F7 07 87 04 E1 C7 F5
       :               FC 96 E7 16 75 F6 ED 88 2E 7A B6 5C 3F 58 45 43
169   4:                   INTEGER 27979789
       :                   }
175   0:                 [0]
177  15:                 GeneralizedTime 03/04/2024 12:37:47 GMT
194  17:                 [0] {
196  15:                   GeneralizedTime 10/04/2024 12:37:47 GMT
       :                   }
       :                 }
       :               }
       :             }
213  10:           SEQUENCE {
215   8:             OBJECT IDENTIFIER
       :               ecdsaWithSHA384 (1 2 840 10045 4 3 3)
       :             }
225 105:           BIT STRING, encapsulates {
228 102:             SEQUENCE {
230  49:               INTEGER
       :               00 D1 99 59 A2 21 BE 03 9B DC 84 5E 2B F6 5E DA
       :               1E 42 2E 9F B6 98 CB 9A 50 6A A9 12 22 FB 78 6D
       :               7C 5F CE CA 37 47 9C 5B A1 DF C7 10 29 8A AA AD
       :               66
281  49:               INTEGER
       :               00 CA 88 F7 2D 56 EE D2 42 CE 7C ED DD 99 04 41
       :               D0 1F 6D 23 E4 01 62 18 6B 1B 23 56 F7 F8 2F 59
       :               2B A3 48 CE 00 FA 0E 75 A5 0F 0A 52 7E 29 44 6A
       :               B3
       :               }
       :             }
332 595:           [0] {
336 591:             SEQUENCE {
340 587:               SEQUENCE {
344 430:                 SEQUENCE {
348   3:                   [0] {
350   1:                     INTEGER 2
       :                     }
353   1:                   INTEGER 1
356  10:                   SEQUENCE {
358   8:                     OBJECT IDENTIFIER
       :                       ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :                     }
368  56:                   SEQUENCE {
370  11:                     SET {
372   9:                       SEQUENCE {
374   3:                         OBJECT IDENTIFIER countryName (2 5 4 6)
379   2:                         PrintableString 'XX'
       :                         }
       :                       }
383  20:                     SET {
385  18:                       SEQUENCE {
387   3:                         OBJECT IDENTIFIER
       :                           organizationName (2 5 4 10)
392  11:                         UTF8String 'Certs 'r Us'
       :                         }
       :                       }
405  19:                     SET {
407  17:                       SEQUENCE {
409   3:                         OBJECT IDENTIFIER commonName (2 5 4 3)
414  10:                         UTF8String 'Issuing CA'
       :                         }
       :                       }
       :                     }
426  30:                   SEQUENCE {
428  13:                     UTCTime 02/04/2024 12:37:47 GMT
443  13:                     UTCTime 02/04/2025 12:37:47 GMT
       :                     }
458  60:                   SEQUENCE {
460  11:                     SET {
462   9:                       SEQUENCE {
464   3:                         OBJECT IDENTIFIER countryName (2 5 4 6)
469   2:                         PrintableString 'XX'
       :                         }
       :                       }
473  20:                     SET {
475  18:                       SEQUENCE {
477   3:                         OBJECT IDENTIFIER
       :                           organizationName (2 5 4 10)
482  11:                         UTF8String 'Certs 'r Us'
       :                         }
       :                       }
495  23:                     SET {
497  21:                       SEQUENCE {
499   3:                         OBJECT IDENTIFIER commonName (2 5 4 3)
504  14:                         UTF8String 'OCSP Responder'
       :                         }
       :                       }
       :                     }
520 118:                   SEQUENCE {
```

```
522  16:                     SEQUENCE {
524   7:                       OBJECT IDENTIFIER
       :                         ecPublicKey (1 2 840 10045 2 1)
533   5:                       OBJECT IDENTIFIER
       :                         secp384r1 (1 3 132 0 34)
       :                       }
540  98:                     BIT STRING
       :               04 5B 09 01 B8 85 23 29 6E B9 19 D5 0F FA 1A 9C
       :               B3 74 BC 4D 40 95 86 28 2B FE CA 11 B1 D9 5A DB
       :               B5 47 34 AF 57 0B F8 2B 72 28 CF 22 6B CF 4C 25
       :               DD BC FE 3B 1A 3A D3 94 30 EF F7 63 E1 D6 8D 2E
       :               15 1D 91 72 0B 77 95 B5 8D A6 B3 46 39 61 3A 8F
       :               B9 B5 A8 DA 48 C6 74 71 17 F9 91 9E 84 24 F3 7E
       :               C8
       :                     }
640 135:                   [3] {
643 132:                     SEQUENCE {
646  29:                       SEQUENCE {
648   3:                         OBJECT IDENTIFIER
       :                           subjectKeyIdentifier (2 5 29 14)
653  22:                         OCTET STRING, encapsulates {
655  20:                           OCTET STRING
       :               0A E3 A0 FE 9D D4 25 76 98 B5 EB 72 EB CA 0C E7
       :               BF 3D F5 F1
       :                           }
       :                         }
677  31:                       SEQUENCE {
679   3:                         OBJECT IDENTIFIER
       :                           authorityKeyIdentifier (2 5 29 35)
684  24:                         OCTET STRING, encapsulates {
686  22:                           SEQUENCE {
688  20:                             [0]
       :               8E C2 14 09 60 76 EA 90 38 E9 39 AE 1B 6D 52 C4
       :               17 7D 9F BE
       :                             }
       :                           }
       :                         }
710  12:                       SEQUENCE {
712   3:                         OBJECT IDENTIFIER
       :                           basicConstraints (2 5 29 19)
717   1:                         BOOLEAN TRUE
720   2:                         OCTET STRING, encapsulates {
722   0:                           SEQUENCE {}
       :                           }
       :                         }
724  14:                       SEQUENCE {
726   3:                         OBJECT IDENTIFIER keyUsage (2 5 29 15)
731   1:                         BOOLEAN TRUE
734   4:                         OCTET STRING, encapsulates {
736   2:                           BIT STRING 7 unused bits
       :                             '1'B (bit 0)
       :                           }
       :                         }
740  19:                       SEQUENCE {
742   3:                         OBJECT IDENTIFIER
       :                           extKeyUsage (2 5 29 37)
747  12:                         OCTET STRING, encapsulates {
749  10:                           SEQUENCE {
751   8:                             OBJECT IDENTIFIER
       :                               ocspSigning (1 3 6 1 5 5 7 3 9)
       :                             }
       :                           }
       :                         }
761  15:                       SEQUENCE {
763   9:                         OBJECT IDENTIFIER
       :                           ocspNoCheck (1 3 6 1 5 5 7 48 1 5)
774   2:                         OCTET STRING, encapsulates {
776   0:                           NULL
       :                           }
       :                         }
       :                       }
       :                     }
       :                   }
778  10:                 SEQUENCE {
780   8:                   OBJECT IDENTIFIER
       :                     ecdsaWithSHA512 (1 2 840 10045 4 3 4)
       :                   }
790 138:                 BIT STRING, encapsulates {
794 134:                   SEQUENCE {
797  65:                     INTEGER
       :               14 2A 8C D6 0A 6C 65 C7 74 65 DF 11 5B C1 FF F8
       :               BE 0E 21 B4 DA 1A DA 53 D9 06 34 A5 DE 89 07 0F
       :               75 94 5A 8D 0B 18 FE 17 59 3D 5C 9A CA 49 00 15
       :               54 06 BF 6F 72 5A 64 EB 11 AC 7E AF 8A 19 4E DC
       :               C6
864  65:                     INTEGER
       :               49 0B 0B 49 A6 2E E6 D3 44 31 F6 BF EE 80 D5 AC
       :               9C 21 52 88 A5 1D C6 EB E3 EE 68 3D 94 9B 73 D2
       :               17 B1 44 96 4A 14 E0 D6 24 6E 5D 52 1F FF 05 CD
       :               B0 F2 FC B0 81 86 28 76 E5 EE E1 02 DC A0 FD 7B
       :               08
       :                     }
       :                   }
       :                 }
       :               }
       :             }
       :           }
       :         }
       :       }
       :     }
       :   }
```

Acknowledgments

The authors of this version of the document wish to thank Alex Deacon and Ryan Hurst for their work to produce the original version of the lightweight profile for the OCSP protocol.

The authors of this version of the document wish to thank Paul Kyzivat, Russ Housley, Rob Stradling, Roman Danyliw, and Wendy Brown for their reviews, feedback, and suggestions.

The authors wish to thank Magnus Nystrom of RSA Security, Inc., Jagjeet Sondh of Vodafone Group R&D, and David Engberg of CoreStreet, Ltd. for their contributions to the original [RFC5019] specification. Listed organizational affiliations reflect the authors' affiliations at the time RFC 5019 was published.

Authors' Addresses

Tadahiko Ito
SECOM CO., LTD.
Email: tadahiko.ito.public@gmail.com

Additional contact information:

> **伊藤 忠彦**
> SECOM CO., LTD.

Clint Wilson
Apple, Inc.
Email: clintw@apple.com

Corey Bonnell
DigiCert, Inc.
Email: corey.bonnell@digicert.com

Sean Turner
sn3rd
Email: sean@sn3rd.com