Stream:	Internet Engineering Task Force (IETF)
RFC:	9654
Obsoletes:	8954
Updates:	6960
Category:	Standards Track
Published:	August 2024
ISSN:	2070-1721
Author:	H. Sharma, Ed. *Netskope Inc*

RFC 9654
Online Certificate Status Protocol (OCSP) Nonce Extension

Abstract

RFC 8954 imposed size constraints on the optional Nonce extension for the Online Certificate Status Protocol (OCSP). OCSP is used to check the status of a certificate, and the Nonce extension is used to cryptographically bind an OCSP response message to a particular OCSP request message.

Some environments use cryptographic algorithms that generate a Nonce value that is longer than 32 octets. This document also modifies the "Nonce" section of RFC 6960 to clearly define and differentiate the encoding format and values for easier implementation and understanding. This document obsoletes RFC 8954, which includes updated ASN.1 modules for OCSP, and updates RFC 6960.

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 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9654.

Copyright Notice

Table of Contents

1. Introduction

The Nonce extension was previously defined in Section 4.4.1 of [RFC6960]. The Nonce cryptographically binds an OCSP request and a response. It guarantees the freshness of an OCSP response and avoids replay attacks. This extension was updated in [RFC8954]. [RFC8954] limits the maximum Nonce length to 32 octets. To support cryptographic algorithms that generate a Nonce that is longer than 32 octets, this document updates the maximum allowed size of the Nonce to 128 octets. In addition, this document recommends that the OCSP requester and responder use a Nonce with a minimum length of 32 octets.

1.1. Requirements Language

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.

2. OCSP Extensions

The message formats for OCSP requests and responses are defined in [RFC6960] and the Nonce extension was updated in [RFC8954]. [RFC6960] also defines the standard extensions for OCSP messages based on the extension model employed in X.509 version 3 certificates (see [RFC5280]). [RFC8954] replaces Section 4.4.1 of [RFC6960] to limit the minimum and maximum length for the Nonce value. This document extends the maximum allowed nonce length to 128 octets and does not change the specifications of any of the other extensions defined in [RFC6960].

2.1. Nonce Extension

The Nonce cryptographically binds a request and a response to prevent replay attacks. The Nonce is included as one of the requestExtensions in requests; in responses, it is included as one of the responseExtensions. In both the request and the response, the Nonce is identified by the object identifier id-pkix-ocsp-nonce, while the extnValue is the encoded value of Nonce. If the Nonce extension is present, then the length of the Nonce **MUST** be at least 1 octet and can be up to 128 octets. Implementations compliant with [RFC8954] will not be able to process nonces generated per the new specification with sizes in excess of the limit (32 octets) specified in [RFC8954].

An OCSP requester that implements the extension in this document **MUST** use a minimum length of 32 octets for Nonce in the Nonce extension.

An OCSP responder that supports the Nonce extension **MUST** accept Nonce lengths of at least 16 octets and up to and including 32 octets. A responder **MAY** choose to respond without the Nonce extension for requests in which the length of the Nonce is in between 1 octet and 15 octets or 33 octets and 128 octets.

Responders that implement the extension in this document **MUST** reject any OCSP request that has a Nonce with a length of either 0 octets or greater than 128 octets, with the malformedRequest OCSPResponseStatus as described in Section 4.2.1 of [RFC6960].

The value of the Nonce **MUST** be generated using a cryptographically strong pseudorandom number generator (see [RFC4086]). The minimum Nonce length of 1 octet is defined to provide backward compatibility with older OCSP requesters that follow [RFC6960].

```
id-pkix-ocsp           OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-nonce     OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
Nonce ::= OCTET STRING(SIZE(1..128))
```

The following is an example of an encoded OCSP Nonce extension with a 32-octet Nonce in hexadecimal format.

```
30 2f 06 09 2b 06 01 05 05 07 30 01 02 04 22 04
20 dd 49 d4 07 2c 44 9d a1 c3 17 bd 1c 1b df fe
db e1 50 31 2e c4 cd 0a dd 18 e5 bd 6f 84 bf 14
c8
```

Here is the decoded version of the above example. Offset, Length, and Object Identifier are in decimal.

```
Offset  Length
0       47    : SEQUENCE {
2       9     :  OBJECT IDENTIFIER ocspNonce
              :   (1 3 6 1 5 5 7 48 1 2)
13      34    :  OCTET STRING, encapsulates {
15      32    :   OCTET STRING
              :    DD 49 D4 07 2C 44 9D A1 C3 17 BD 1C 1B DF FE DB
              :    E1 50 31 2E C4 CD 0A DD 18 E5 BD 6F 84 BF 14 C8
              :  }
              : }
```

3. Security Considerations

The security considerations of OCSP, in general, are described in [RFC6960]. During the interval in which the previous OCSP response for a certificate is not expired but the responder has a changed status for that certificate, a copy of that OCSP response can be used to indicate that the status of the certificate is still valid. Including a requester's nonce value in the OCSP response ensures that the response is the most recent response from the server and not an old copy.

3.1. Replay Attack

The Nonce extension is used to avoid replay attacks. Since the OCSP responder may choose not to send the Nonce extension in the OCSP response even if the requester has sent the Nonce extension in the request [RFC5019], an on-path attacker can intercept the OCSP request and respond with an earlier response from the server without the Nonce extension. This can be mitigated by configuring the server to use a short time interval between the thisUpdate and nextUpdate fields in the OCSP response.

4. IANA Considerations

For the ASN.1 modules in Appendixes A.1 and A.2, IANA has assigned the following object identifiers (OIDs) in the "SMI Security for PKIX Module Identifier" registry (1.3.6.1.5.5.7.0):

Value	Description
111	id-mod-ocsp-2024-88
112	id-mod-ocsp-2024-08

Table 1

5. References

5.1. 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>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[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>.

[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>.

[RFC8954] Sahni, M., Ed., "Online Certificate Status Protocol (OCSP) Nonce Extension", RFC 8954, DOI 10.17487/RFC8954, November 2020, <https://www.rfc-editor.org/info/rfc8954>.

5.2. Informative References

[Err5891] RFC Errata, Erratum ID 5891, RFC 6960, <https://www.rfc-editor.org/errata/eid5891>.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <https://www.rfc-editor.org/info/rfc5912>.

Appendix A. ASN.1 Modules

This section includes the ASN.1 modules for OCSP and replaces the entirety of Section 5 of [RFC8954]. It addresses Errata ID 5891 [Err5891] as well.

Appendix A.1 includes an ASN.1 module that conforms to the 1998 version of ASN.1 for all syntax elements of OCSP. This module replaces the module in Appendix B.1 of [RFC6960].

Appendix A.2 includes an ASN.1 module, corresponding to the module present in Appendix A.1, that conforms to the 2008 version of ASN.1. This module replaces the modules in Section 4 of [RFC5912] and Appendix B.2 of [RFC6960]. Although a 2008 ASN.1 module is provided, the module in Appendix A.1 remains the normative module per the policy of the PKIX Working Group.

A.1. OCSP in ASN.1 - 1998 Syntax

```
<CODE BEGINS>
OCSP-2024-88
      { iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0)
        id-mod-ocsp-2024-88(111) }

DEFINITIONS EXPLICIT TAGS ::=
BEGIN

IMPORTS
    AuthorityInfoAccessSyntax, CRLReason, GeneralName
    FROM PKIX1Implicit88 -- From [RFC5280]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-pkix1-implicit(19) }

    Name, CertificateSerialNumber, Extensions,
    id-kp, id-ad-ocsp, Certificate, AlgorithmIdentifier
    FROM PKIX1Explicit88 -- From [RFC5280]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-pkix1-explicit(18) } ;

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 }

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

Version ::= INTEGER { v1(0) }

Nonce ::= OCTET STRING (SIZE(1..128))

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 }

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

OCSPResponseStatus ::= ENUMERATED {
   successful       (0),  -- Response has valid confirmations
   malformedRequest (1),  -- Illegal confirmation request
   internalError    (2),  -- Internal error in issuer
   tryLater         (3),  -- Try again later
                          -- (4) is not used
   sigRequired      (5),  -- Must sign the request
   unauthorized     (6)   -- Request unauthorized
}

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

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,
```

```
                           -- The format for GeneralizedTime is
                           -- as specified in Section 4.1.2.5.2
                           -- [RFC5280]
   responses               SEQUENCE OF SingleResponse,
   responseExtensions  [1] EXPLICIT Extensions OPTIONAL }

ResponderID ::= CHOICE {
   byName              [1] Name,
   byKey               [2] KeyHash }

KeyHash ::= OCTET STRING
            -- SHA-1 hash of responder's public key (i.e., the
            -- SHA-1 hash of the value of the BIT STRING
            -- subjectPublicKey [excluding the tag, length, and
            -- number of unused bits] in the responder's
            -- certificate)

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

CertStatus ::= CHOICE {
   good                [0] IMPLICIT NULL,
   revoked             [1] IMPLICIT RevokedInfo,
   unknown             [2] IMPLICIT UnknownInfo }

RevokedInfo ::= SEQUENCE {
   revocationTime        GeneralizedTime,
   revocationReason  [0] EXPLICIT CRLReason OPTIONAL }

UnknownInfo ::= NULL

ArchiveCutoff ::= GeneralizedTime

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER

ServiceLocator ::= SEQUENCE {
   issuer       Name,
   locator      AuthorityInfoAccessSyntax }

CrlID ::= SEQUENCE {
    crlUrl     [0] EXPLICIT IA5String OPTIONAL,
    crlNum     [1] EXPLICIT INTEGER OPTIONAL,
    crlTime    [2] EXPLICIT GeneralizedTime OPTIONAL }

PreferredSignatureAlgorithms ::= SEQUENCE OF
                                 PreferredSignatureAlgorithm

PreferredSignatureAlgorithm ::= SEQUENCE {
   sigIdentifier   AlgorithmIdentifier,
   certIdentifier  AlgorithmIdentifier OPTIONAL }


-- Object Identifiers
```

```
id-kp-OCSPSigning            OBJECT IDENTIFIER ::= { id-kp 9 }
id-pkix-ocsp                 OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic           OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
id-pkix-ocsp-nonce           OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
id-pkix-ocsp-crl             OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }
id-pkix-ocsp-response        OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }
id-pkix-ocsp-nocheck         OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
id-pkix-ocsp-archive-cutoff  OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }
id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }
id-pkix-ocsp-pref-sig-algs   OBJECT IDENTIFIER ::= { id-pkix-ocsp 8 }
id-pkix-ocsp-extended-revoke OBJECT IDENTIFIER ::= { id-pkix-ocsp 9 }

END


<CODE ENDS>
```

A.2. OCSP in ASN.1 - 2008 Syntax

```
<CODE BEGINS>
OCSP-2024-08
     { iso(1) identified-organization(3) dod(6) internet(1)
       security(5) mechanisms(5) pkix(7) id-mod(0)
       id-mod-ocsp-2024-08(112) }

DEFINITIONS EXPLICIT TAGS ::=
BEGIN

IMPORTS
    Extensions{}, EXTENSION
    FROM PKIX-CommonTypes-2009 -- From [RFC5912]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-mod-pkixCommon-02(57) }

    AlgorithmIdentifier{}, DIGEST-ALGORITHM,
    SIGNATURE-ALGORITHM, PUBLIC-KEY
    FROM AlgorithmInformation-2009 -- From [RFC5912]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-mod-algorithmInformation-02(58) }

    AuthorityInfoAccessSyntax, GeneralName,
    CrlEntryExtensions, CRLReason
    FROM PKIX1Implicit-2009 -- From [RFC5912]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-mod-pkix1-implicit-02(59) }

    Name, Certificate, CertificateSerialNumber,
    id-kp, id-ad-ocsp
    FROM PKIX1Explicit-2009 -- From [RFC5912]
    { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-mod-pkix1-explicit-02(51) }
```

```
    sa-dsaWithSHA1, sa-rsaWithMD2,
    sa-rsaWithMD5, sa-rsaWithSHA1
    FROM PKIXAlgs-2009 -- From [RFC5912]
    { iso(1) identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7)
    id-mod(0) id-mod-pkix1-algorithms2008-02(56) } ;

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
                           {{ re-ocsp-nonce | re-ocsp-response |
                           re-ocsp-preferred-signature-algorithms,
                           ... }} OPTIONAL }

Signature ::= SEQUENCE {
   signatureAlgorithm   AlgorithmIdentifier
                           { SIGNATURE-ALGORITHM, {...}},
   signature            BIT STRING,
   certs             [0] EXPLICIT SEQUENCE OF
                           Certificate OPTIONAL }

Version ::= INTEGER  {  v1(0) }

Nonce ::= OCTET STRING (SIZE(1..128))

Request ::= SEQUENCE {
   reqCert                     CertID,
   singleRequestExtensions  [0] EXPLICIT Extensions
                                {{ re-ocsp-service-locator,
                                ... }} OPTIONAL }

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

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

OCSPResponseStatus ::= ENUMERATED {
   successful        (0), -- Response has valid confirmations
   malformedRequest  (1), -- Illegal confirmation request
   internalError     (2), -- Internal error in issuer
   tryLater          (3), -- Try again later
                          -- (4) is not used
   sigRequired       (5), -- Must sign the request
   unauthorized      (6)  -- Request unauthorized
}
```

```
RESPONSE ::= TYPE-IDENTIFIER

ResponseSet RESPONSE ::= { basicResponse, ... }

ResponseBytes ::= SEQUENCE {
   responseType  RESPONSE.&id ({ResponseSet}),
   response      OCTET STRING (CONTAINING RESPONSE.
                     &Type({ResponseSet}{@responseType}))}

basicResponse RESPONSE ::=
   { BasicOCSPResponse IDENTIFIED BY id-pkix-ocsp-basic }

BasicOCSPResponse ::= SEQUENCE {
  tbsResponseData      ResponseData,
  signatureAlgorithm   AlgorithmIdentifier
                           { SIGNATURE-ALGORITHM,
                               { sa-dsaWithSHA1 |
                                 sa-rsaWithSHA1 |
                                 sa-rsaWithMD5  |
                                 sa-rsaWithMD2,
                                 ... }},
  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
                              {{ re-ocsp-nonce |
                                 re-ocsp-extended-revoke,
                                 ... }} OPTIONAL }

ResponderID ::= CHOICE {
  byName   [1] Name,
  byKey    [2] KeyHash }

KeyHash ::= OCTET STRING
            -- SHA-1 hash of responder's public key
            -- (excluding the tag and length and number
            -- of unused bits)

SingleResponse ::= SEQUENCE {
  certID                CertID,
  certStatus            CertStatus,
  thisUpdate            GeneralizedTime,
  nextUpdate        [0] EXPLICIT GeneralizedTime OPTIONAL,
  singleExtensions  [1] EXPLICIT Extensions
                        {{ re-ocsp-crl |
                           re-ocsp-archive-cutoff |
                           CrlEntryExtensions,
                           ... }} OPTIONAL }

CertStatus ::= CHOICE {
   good     [0] IMPLICIT NULL,
   revoked  [1] IMPLICIT RevokedInfo,
```

```
   unknown  [2] IMPLICIT UnknownInfo }

RevokedInfo ::= SEQUENCE {
   revocationTime        GeneralizedTime,
   revocationReason  [0] EXPLICIT CRLReason OPTIONAL }

UnknownInfo ::= NULL

ArchiveCutoff ::= GeneralizedTime

AcceptableResponses ::= SEQUENCE OF RESPONSE.&id({ResponseSet})

ServiceLocator ::= SEQUENCE {
   issuer    Name,
   locator   AuthorityInfoAccessSyntax }

CrlID ::= SEQUENCE {
   crlUrl   [0] EXPLICIT IA5String OPTIONAL,
   crlNum   [1] EXPLICIT INTEGER OPTIONAL,
   crlTime  [2] EXPLICIT GeneralizedTime OPTIONAL }

PreferredSignatureAlgorithms ::= SEQUENCE OF
                                 PreferredSignatureAlgorithm

PreferredSignatureAlgorithm ::= SEQUENCE {
   sigIdentifier  AlgorithmIdentifier
                     { SIGNATURE-ALGORITHM, {...} },
   certIdentifier AlgorithmIdentifier {PUBLIC-KEY, {...}}
                     OPTIONAL }


-- Certificate Extensions

ext-ocsp-nocheck EXTENSION ::= {
    SYNTAX NULL IDENTIFIED BY id-pkix-ocsp-nocheck }


-- Request Extensions

re-ocsp-nonce EXTENSION ::= {
    SYNTAX Nonce IDENTIFIED BY id-pkix-ocsp-nonce }

re-ocsp-response EXTENSION ::= {
    SYNTAX AcceptableResponses IDENTIFIED BY
    id-pkix-ocsp-response }

re-ocsp-service-locator EXTENSION ::= {
    SYNTAX ServiceLocator IDENTIFIED BY
    id-pkix-ocsp-service-locator }

re-ocsp-preferred-signature-algorithms EXTENSION ::= {
    SYNTAX PreferredSignatureAlgorithms IDENTIFIED BY
    id-pkix-ocsp-pref-sig-algs  }


-- Response Extensions

re-ocsp-crl EXTENSION ::= {
```

```
    SYNTAX CrlID IDENTIFIED BY id-pkix-ocsp-crl }

re-ocsp-archive-cutoff EXTENSION ::= {
    SYNTAX ArchiveCutoff IDENTIFIED BY
    id-pkix-ocsp-archive-cutoff }

re-ocsp-extended-revoke EXTENSION ::= {
    SYNTAX NULL IDENTIFIED BY id-pkix-ocsp-extended-revoke }


-- Object Identifiers

id-kp-OCSPSigning            OBJECT IDENTIFIER ::= { id-kp 9 }
id-pkix-ocsp                 OBJECT IDENTIFIER ::= id-ad-ocsp
id-pkix-ocsp-basic           OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
id-pkix-ocsp-nonce           OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
id-pkix-ocsp-crl             OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }
id-pkix-ocsp-response        OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }
id-pkix-ocsp-nocheck         OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
id-pkix-ocsp-archive-cutoff  OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }
id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }
id-pkix-ocsp-pref-sig-algs   OBJECT IDENTIFIER ::= { id-pkix-ocsp 8 }
id-pkix-ocsp-extended-revoke OBJECT IDENTIFIER ::= { id-pkix-ocsp 9 }

END


<CODE ENDS>
```

Acknowledgements

The authors of this document thank Mohit Sahni for his work to produce [RFC8954].

The authors also thank Russ Housley, Corey Bonnell, Michael StJohns, Tomas Gustavsson, and Carl Wallace for their feedback and suggestions.

Author's Address

Himanshu Sharma (EDITOR)
Netskope Inc
2445 Augustine Dr 3rd floor
Santa Clara, California 95054
United States of America
Email: himanshu@netskope.com
URI: www.netskope.com