Internet Engineering Task Force (IETF)
Request for Comments: 6277
Updates: 2560
Category: Standards Track
ISSN: 2070-1721
June 2011

S. Santesson
3xA Security
P. Hallam-Baker
Default Deny Security

# Online Certificate Status Protocol Algorithm Agility

## Abstract

The Online Certificate Status Protocol (OCSP) requires server responses to be signed but does not specify a mechanism for selecting the signature algorithm to be used. This may lead to avoidable interoperability failures in contexts where multiple signature algorithms are in use. This document specifies rules for server signature algorithm selection and an extension that allows a client to advise a server that specific signature algorithms are supported.

## Status of This Memo



## Copyright Notice

Table of Contents



## 1. Introduction

The Online Certificate Status Protocol (OCSP) [RFC2560] defines a protocol for obtaining certificate status information from an online service. An OCSP responder may or may not be issued an OCSP responder certificate by the certification authority (CA) that issued the certificate whose status is being queried. An OCSP responder may provide pre-signed OCSP responses or may sign responses when queried.

RFC 2560 [RFC2560] specifies a means for an OCSP responder to indicate the signature and digest algorithms used in a response but not how those algorithms are specified. The only algorithm requirements established by that protocol specification are that the OCSP client SHALL support the Digital Signature Algorithm (DSA) sig-alg-oid specified in Section 7.2.2 of [RFC2459] and SHOULD be capable of processing RSA signatures as specified in Section 7.2.1 of [RFC2459]. The only requirement placed on responders by RFC 2560 is that they SHALL support the SHA1 hashing algorithm.

This document specifies rules for server signature algorithm selection and an extension that allows a client to advise a server that specific signature algorithms are supported.

### 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].

## 2. OCSP Algorithm Agility Requirements

Since algorithms other than those that are mandatory to implement are allowed and since a client currently has no mechanism to indicate its algorithm preferences, there is always a risk that a server choosing a non-mandatory algorithm will generate a response that the client may not support.

While an OCSP responder may apply rules for algorithm selection, e.g., using the signature algorithm employed by the CA for signing certificate revocation lists (CRLs) and certificates, such rules may fail in common situations:

- o The algorithm used to sign the CRLs and certificates may not be consistent with the key pair being used by the OCSP responder to sign responses.
- o A request for an unknown certificate provides no basis for a responder to select from among multiple algorithm options.

Without modifying the protocol, the last criterion cannot be resolved through the information available from in-band signaling using the protocol described in RFC 2560 [RFC2560].

In addition, an OCSP responder may wish to employ different signature algorithms than the one used by the CA to sign certificates and CRLs for several reasons:

- o The responder may employ an algorithm for certificate status response that is less computationally demanding than for signing the certificate itself.
- o An implementation may wish to guard against the possibility of a compromise resulting from a signature algorithm compromise by employing two separate signature algorithms.

This document describes:

- o A mechanism that allows a client to indicate the set of preferred signature algorithms.

o Rules for signature algorithm selection that maximize the probability of successful operation in the case that no supported preferred algorithm(s) are specified.

## 3. Updates to Mandatory and Optional Cryptographic Algorithms

Section 4.3 ("Mandatory and Optional Cryptographic Algorithms") of RFC 2560 [RFC2560] is updated as follows:

OLD: Clients that request OCSP services SHALL be capable of processing responses signed used DSA keys identified by the DSA sig-alg-oid specified in section 7.2.2 of [RFC2459]. Clients SHOULD also be capable of processing RSA signatures as specified in section 7.2.1 of [RFC2459]. OCSP responders SHALL support the SHA1 hashing algorithm.

NEW: Clients that request OCSP services SHALL be capable of processing responses signed using RSA with SHA-1 (identified by sha1WithRSAEncryption OID specified in [RFC3279]) and RSA with SHA-256 (identified by sha256WithRSAEncryption OID specified in [RFC4055]). Clients SHOULD also be capable of processing responses signed using DSA keys (identified by the id-dsa-with-sha1 OID specified in [RFC3279]). Clients MAY support other algorithms.

## 4. Client Indication of Preferred Signature Algorithms

A client MAY declare a preferred set of algorithms in a request by including a preferred signature algorithms extension in requestExtensions of the OCSPRequest [RFC2560].

```
id-pkix-ocsp-pref-sig-algs OBJECT IDENTIFIER ::= { id-pkix-ocsp 8 }

PreferredSignatureAlgorithms ::= SEQUENCE OF
                                 PreferredSignatureAlgorithm

PreferredSignatureAlgorithm ::= SEQUENCE {
   sigIdentifier        AlgorithmIdentifier,
   pubKeyAlgIdentifier  SMIMECapability OPTIONAL
   }
```

The syntax of AlgorithmIdentifier is defined in Section 4.1.1.2 of RFC 5280 [RFC5280]. The syntax of SMIMECapability is defined in RFC 5751 [RFC5751].

sigIdentifier specifies the signature algorithm the client prefers, e.g., algorithm=ecdsa-with-sha256. Parameters are absent for most common signature algorithms.

pubKeyAlgIdentifier specifies the subject public key algorithm identifier the client prefers in the server's certificate used to validate the OCSP response, e.g., algorithm=id-ecPublicKey and parameters= secp256r1.

pubKeyAlgIdentifier is OPTIONAL and provides means to specify parameters necessary to distinguish among different usages of a particular algorithm, e.g., it may be used by the client to specify what curve it supports for a given elliptic curve algorithm.

The client MUST support each of the specified preferred signature algorithms, and the client MUST specify the algorithms in the order of preference, from the most preferred to the least preferred.

Section 5 of this document describes how a server selects an algorithm for signing OCSP responses to the requesting client.

## 5. Responder Signature Algorithm Selection

RFC 2560 [RFC2560] does not specify a mechanism for deciding the signature algorithm to be used in an OCSP response. As previously noted, this does not provide a sufficient degree of certainty as to the algorithm selected to facilitate interoperability.

### 5.1. Dynamic Response

As long as the selected algorithm meets all security requirements of the OCSP responder, a responder MAY maximize the potential for ensuring interoperability by selecting a supported signature algorithm using the following order of precedence, where the first method has the highest precedence:

1. Select an algorithm specified as a preferred signing algorithm in the client request.

2. Select the signing algorithm used to sign a certificate revocation list (CRL) issued by the certificate issuer to provide status information for the certificate specified by CertID.

3. Select the signing algorithm used to sign the OCSPRequest.

4. Select a signature algorithm that has been advertised as being the default signature algorithm for the signing service using an out-of-band mechanism.

5. Select a mandatory or recommended signing algorithm specified for the version of the OCSP protocol in use.

A responder SHOULD always apply the lowest-numbered selection mechanism that results in the selection of a known and supported algorithm that meets the responder's criteria for cryptographic algorithm strength.

## 5.2. Static Response

For purposes of efficiency, an OCSP responder is permitted to generate static responses in advance of a request. The case may not permit the responder to make use of the client request data during the response generation; however, the responder SHOULD still use the client request data during the selection of the pre-generated response to be returned. Responders MAY use the historical client requests as part of the input to the decisions of what different algorithms should be used to sign the pre-generated responses.

## 6. Acknowledgements

The authors acknowledge Santosh Chokhani for the helpful comments made on earlier drafts, Sean Turner for proposing the syntax for algorithm identifiers, Jim Schaad for providing and testing the ASN.1 module in Appendix A, and Stephen Kent for valuable review and input.

## 7. Security Considerations

The mechanism used to choose the response signing algorithm MUST be considered to be sufficiently secure against cryptanalytic attack for the intended application.

In most applications, it is sufficient for the signing algorithm to be at least as secure as the signing algorithm used to sign the original certificate whose status is being queried. However, this criteria may not hold in long-term archival applications in which the status of a certificate is being queried for a date in the distant past, long after the signing algorithm has ceased to be considered trustworthy.

## 7.1. Use of Insecure Algorithms

It is not always possible for a responder to generate a response that the client is expected to understand and that meets contemporary standards for cryptographic security. In such cases, an OCSP responder operator MUST balance the risk of employing a compromised security solution and the cost of mandating an upgrade, including the risk that the alternative chosen by end users will offer even less security or no security.

In archival applications, it is quite possible that an OCSP responder might be asked to report the validity of a certificate on a date in the distant past. Such a certificate might employ a signing method that is no longer considered acceptably secure. In such circumstances, the responder MUST NOT generate a signature using a signing mechanism that is not considered acceptably secure.

A client MUST accept any signing algorithm in a response that it specified as a preferred signing algorithm in the request. Therefore, it follows that a client MUST NOT specify a preferred signing algorithm that is either not supported or not considered acceptably secure.

### 7.2. Man-in-the-Middle Downgrade Attack

The mechanism to support client indication of preferred signature algorithms is not protected against a man-in-the-middle downgrade attack. This constraint is not considered to be a significant security concern since the OCSP responder MUST NOT sign OCSP responses using weak algorithms even if requested by the client. In addition, the client can reject OCSP responses that do not meet its own criteria for acceptable cryptographic security no matter what mechanism is used to determine the signing algorithm of the response.

### 7.3. Denial-of-Service Attack

Algorithm agility mechanisms defined in this document introduce a slightly increased attack surface for denial-of-service attacks where the client request is altered to require algorithms that are not supported by the server. Denial-of-service considerations from RFC 4732 [RFC4732] are relevant for this document.

## 8. References

### 8.1. Normative References

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

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

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[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.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, June 2010.

## 8.2. Informative References

[RFC2459] Housley, R., Ford, W., Polk, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile", RFC 2459, January 1999.

[RFC4732] Handley, M., Ed., Rescorla, E., Ed., and IAB, "Internet Denial-of-Service Considerations", RFC 4732, December 2006.

## Appendix A. ASN.1 Modules

### A.1. ASN.1 Module

```
 OCSP-AGILITY-2009 { iso(1) identified-organization(3) dod(6)
     internet(1)  security(5) mechanisms(5) pkix(7) id-mod(0)
     id-mod-ocsp-agility-2009-93(66) }

 DEFINITIONS EXPLICIT TAGS ::=
 BEGIN

   EXPORTS ALL;   -- export all items from this module
   IMPORTS

   id-pkix-ocsp
     FROM OCSP-2009  -- From OCSP [RFC2560]
      { iso(1) identified-organization(3) dod(6) internet(1) security(5)
        mechanisms(5) pkix(7) id-mod(0) id-mod-ocsp-02(48) }

   AlgorithmIdentifier{}, SMIMECapability{}, 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) }

   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)} ;

    --  Add re-preferred-signature-algorithms to the set of extensions
    --  for TBSRequest.requestExtensions

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

   id-pkix-ocsp-pref-sig-algs OBJECT IDENTIFIER ::= { id-pkix-ocsp 8 }

   PreferredSignatureAlgorithms ::= SEQUENCE OF
                                    PreferredSignatureAlgorithm

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

 END
```

## A.2. 1988 ASN.1 Module

```
 OCSP-AGILITY-88 { iso(1) identified-organization(3) dod(6) internet(1)
     security(5) mechanisms(5) pkix(7) id-mod(0)
     id-mod-ocsp-agility-2009-88(67) }

 DEFINITIONS EXPLICIT TAGS ::=
 BEGIN

   -- EXPORTS ALL;
   IMPORTS

   id-pkix-ocsp  -- From [RFC2560]
     FROM OCSP
      { iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-ocsp(14)}

   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) };

   SMIMECapability
     FROM SecureMimeMessageV3dot1 -- From [RFC5751]
      { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
        smime(16) modules(0) msg-v3dot1(21) }

     id-pkix-ocsp-pref-sig-algs OBJECT IDENTIFIER ::= { id-pkix-ocsp 8 }

     PreferredSignatureAlgorithms ::= SEQUENCE OF
                                      PreferredSignatureAlgorithm

     PreferredSignatureAlgorithm ::= SEQUENCE {
        sigIdentifier        AlgorithmIdentifier,
        pubKeyAlgIdentifier  SMIMECapability OPTIONAL
        }

 END
```

## Authors' Addresses

Stefan Santesson
3xA Security AB
Sweden

Email: sts@aaa-sec.com

Phillip Hallam-Baker
Default Deny Security

EMail: hallam@gmail.com