

RSA-PSS in XMLDSig

Konrad Lanz, IAIK/SIC A-SIT Konrad.Lanz@iaik.tugraz.at

Dieter Bratko, IAIK Dieter.Bratko@iaik.tugraz.at

Peter Lipp, IAIK Peter.Lipp@iaik.tugraz.at

Position Paper for the W3C Workshop on Next Steps for XML Signature and XML Encryption

25/26 September 2007 -- Mountain View, California
hosted by VeriSign

Abstract

This document specifies syntax and semantics for using the RSA Probabilistic Signature Scheme (RSA-PSS) within the [XMLDSig] standard. It proposes an example namespace for referencing the PSS signature method and suggests an XML schema notation for the algorithm parameters used by PSS.

Introduction

XMLDSig [XMLDSig] references one RSA based signature method, <http://www.w3.org/2000/09/xmlsig#rsa-sha1> (RSAwithSHA1). RSAwithSHA1 uses the RSASSA-PKCS1-v1_5 signature algorithm ([PKCS#1v1.5], [PKCS#1v2.1]). Currently there are no known attacks against RSASSA-PKCS1-v1_5 itself; however, Daniel Bleichenbacher recently showed an implementation vulnerability ([FIN-BLEICH]).

The security of the RSA-PSS signature scheme (RSASSA-PSS, specified in [PKCS#1v2.1]) is more closely related to the security of the underlying RSA problem. In contrast to the deterministic encoding method of RSASSA-PKCS1-v1_5, RSA-PSS uses a randomized method allowing a tighter security proof ([PKCS1v2.1], [KAL-PSS]).

[PKCS#1v2.1], [KAL-PSS] recommend preferring the RSA-PSS signature method and encourage a gradual transition.

The Cryptographic Message Syntax standard (CMS, [RFC 3852]) already has adopted the RSA-PSS signature method ([RFC 4056]). The draft for the third revision to the official DSS specification [FIPS 186-3 Draft] in section 5.5 references [PKCS#1 v2.1] and considers RSA-PSS as approved.

Hence this document proposes to adopt RSA-PSS for the [XMLDSig] signature standard. It introduces a namespace for referencing the PSS signature method and specifies an XML schema notation for the algorithm parameters used by RSA-PSS.

RSA-PSS Parameters

The relevant parameters for RSA-PSS specified in [PKCS#1v2.1] are:

- the digest method (dm)
- the mask generation function (MGF)
 - the digest method if used in the MGF (mgf-dm)
- the salt length (sl)
- the usually constant trailer field (tf)

These parameters are required for verification of an RSA-PSS signature and potentially chosen on a per signature basis. The average signer/application developer should not have to take the burden of making a secure choice for those parameters. Thus we propose a set of comprehensive default values for this signature scheme. A common agreement for these default values has to be assumed so that the algorithm identifier can be the only required parameter.

Algorithm Namespace and Parameter Schema Notation

Namespace

As the signature method is extensible by its schema definition (Figure 1) in any namespace other than its own (`xmlns="http://www.w3.org/2000/09/xmlsig#"`) the adoption of RSA-PSS can only be achieved by defining algorithm identifiers and parameters as attributes and elements in a different namespace.

In this document we use the namespace <http://www.example.org/xmlsig-pss/> as we do not have the authority to assign namespaces in the domain “[w3.org](http://www.w3.org/)”. This namespace is eventually to be replaced by the final namespace if this proposal is chosen to be adopted and could look something like <http://www.w3.org/2007/09/xmlsig-pss/>.

Algorithm Identifiers

The Algorithm Identifier for RSA-PSS used in this document is <http://www.example.org/xmlsig-pss/#rsa-pss>.

As RSA-PSS uses a mask generation function an additional algorithm identifier is needed as parameter, currently [PKCS#1v2.1] specifies mask generation function 1 (MGF1).

The Algorithm Identifier for MGF1 used in this document is <http://www.example.org/xmlsig-pss/#mgf1>

Algorithm identifiers for hash functions specified in XML encryption [XMLEnc] (SHA-256, SHA-512), those for SHA-224 and SHA-384 of [RFC4051] and also the one for SHA-1 specified in [XMLDSig] are considered to be valid algorithm identifiers for hash functions in the sense of this document.

Choosing the default Hash Function and default Salt Length

The parameter defaults in [PKCS#1v2.1] are to use SHA-1 as the digest method and in the mask generation function (MGF1). The corresponding salt length default value is 20 bytes. The trailer field is fixed to the constant octet 0xBC.

The latest NIST drafts [FIPS 186-3 Draft] in connection with [FIPS 180-3 Draft], [NIST SP 800-107 Draft] and [NIST SP 800-57 Draft] however make a case for moving away from SHA-1 to hash functions with longer output lengths of the SHA family.

[NIST SP 800-57 Draft] states that, SHA-1 has recently been demonstrated to provide less than 80 bits of security for digital signatures and currently assesses the security strength against collisions at 69 bits. NIST further recommends in their draft for the generation of digital signatures in new systems to use the larger hash functions. NIST however still includes SHA-1: "... to reflect it's widespread use in existing systems, for which the reduced security strength may not be of great concern when only 80-bits of security are required."

We would like to add here that recently successful collision attacks on reduced (56 steps [WaYiYu], 64 steps [CaRe]) and full versions ([WaYiYu]) of the SHA-1 hash algorithm have been published. In 2005, Wang showed that it is possible to find a collision on the full SHA-1 algorithm in less (2^{69}) than 2^{80} operations ([WaYiYu]) and announced a further reduction to 2^{63} operations ([WaYaYa]).

Since cryptanalysis research proceeds, practical attacks against SHA-1 may be only a matter of time. Thus it might be advisable not to use SHA-1 in future applications where a strong collision resistance is required.

Although attacks on existing digital signatures require finding a second preimage to a given fixed hash value, collisions are believed to be helpful for the preparation of meaningful messages resulting in the same hash value.

Hence we consider SHA-1 as seriously tarnished and propose SHA-256 as default for RSA-PSS in [XMLDSig].

[PKCS#1v2.1] recommends using the same hash function for the mask generation function as for digesting the message to be signed and to choose the salt length equal to the output length of the digest method output length in bytes. For SHA-256 this is $256/8 = 32$ bytes. For other hash functions this parameter should be set accordingly.

Proposed Defaults for RSA-PSS in XMLDSig

The digest method (dm) defaults to the algorithm id of SHA-256, the mask generation function defaults to the algorithm id of MGF1, the digest method of the mask generation function if used defaults to (dm) and the salt length is computed by the output length of (dm) in bits divided by eight.

- the digest method (dm) SHA-256
- the mask generation function (MGF) MGF1
 - the digest method if used in the MGF (mgf-dm) = dm
- the salt length = output-bits(dm)/8
- the usually constant trailer field 1 (corresponds to 0xbc)

Proposed XML Schema for RSA-PSS

```
<element name="RSAPSSParams" type="pss:RSAPSSParamsType">
  <annotation>
    <documentation>We have to define a top level element so that it can be
used in ds:SignatureMethod</documentation>
  </annotation>
</element>

<complexType name="RSAPSSParamsType">
  <sequence>
    <element ref="ds:DigestMethod" minOccurs="0"/>
    <element name="MaskGenerationFunction"
      type="pss:MaskGenerationFunctionType" minOccurs="0"/>
    <element name="SaltLength" type="int" minOccurs="0"/>
    <element name="TrailerField" type="int" default="1" minOccurs="0"/>
  </sequence>
</complexType>
```

Figure 1: Schema for RSA-PSS Parameters

Figure 1 shows the schema to parameterize the signature method.

```
<complexType name="MaskGenerationFunctionType">
  <sequence>
    <element ref="ds:DigestMethod" minOccurs="0"/>
    <any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Algorithm" type="anyURI"
    default="http://www.example.org/xmlsig-pss/#mgf1"/>
</complexType>
```

Figure 2: SchemaType for Mask Generation Function Parameters

According to [PKCS#1v2.1] the mask generation function (MGF) – if based on a hash algorithm¹ -- is recommended to use the same hash function as the hash function fingerprinting the message. However the structures in [PKCS#1v2.1] allow for separate parameterization of the MGF and the message digest. The approach taken here will also enable to parameterize the (MGF) separately (see Figure 2), but will default by definition to use the same hash function as for digesting the message if not specified.

Eventually the trailer field parameter can be given and defaults to 1 representing an effective binary value of ‘0xbc’. Other values are not specified in [PKCS#1v2.1] and hence this parameter should not be used unless required by other specifications.

Validating against Parameters encoded in the Key or Certificate

[RFC 4055] makes an approach to encode the RSA-PSS parameters within the subjectPublicKeyInfo field of an X.509 certificate. Nevertheless [RFC 4055] asks for the parameters to be added to the signature unless the default values are used. Hence [RFC 4055] is not in con-

¹ Currently [PKCS#1v2.1] only specifies one mask generation function, MGF1 which is based on a hash algorithm and uses SHA-1 by default; this document however proposes SHA-256 as default for MGF1.

flict to this proposal. The binding of those parameters to the key as in [RFC 4055] however should not be circumvented and the following rules apply.

In the case of parameters being specified in the signature method and also in the key² or certificate the parameters in the signature method must be validated against those given in the key/certificate as follows.

- the digest method refers to the same digest method as in the key/certificate
- the mask generation function refers to the same mask generation function as in the key/certificate
 - the digest method if used in the MGF refers to the same digest method as in the key/certificate
- the salt length is at least as long as the one in the key/certificate
- the trailer field is the same as specified by the key/certificate (the effective value for the RSA-PSS scheme is relevant)

The signature method is considered to carry the default values unless explicitly specified by values given in RSAPSSParams or its children. Parameters within a key or certificate may also be provided by default values not specified in this specification. In such a case the foreign default values are treated just as normal values and will have to be added to the signature method during signing and are expected to be present during verification unless they are equal to the default values specified in this specification.

We additionally propose to perform these checks when creating the signature as well as on verification to exclude the possibility of signatures with inconsistent parameters being created by mistake.

Conclusion

We have made a point to adopt RSA-PSS as a signature method that should be supported by the current XML Digital Signatures specification and by future specifications. We have pointed out that recent attacks on SHA-1 imply that SHA-1 should not be the default for future signature generation any more and hence this proposal employs SHA-256 as its default hash algorithm. Default values for parameters follow the recommendations in [PKCS#1 v2.1] beyond their ASN.1 descriptions. The relation between this specification and approaches encoding the RSA-PSS parameters with the key or certificate has been discussed.

² At the time of writing there is an ongoing discussion in the IETF PKIX working group to obsolete RFC 4055 and move the parameters from the subjectPublicKeyInfo field of a certificate to a newly to be defined certificate extension. For that reason we do not propose a schema for a RFC 4055 based KeyInfo containing PSS parameters.

References

[FIN-BLEICH] Hal Finney: Bleichenbacher's RSA signature forgery based on implementation error, 17 Aug. 2006, <http://www.imc.org/ietf-openpgp/mail-archive/msg14307.html>

[PKCS#1v1.5] PKCS#1 v1.5: RSA Encryption Standard RSA Laboratories; 1 Nov. 1993, <ftp://ftp.rsasecurity.com/pub/pkcs/ascii/pkcs-1.asc>

[PKCS#1v2.1] PKCS#1 v2.1: RSA Cryptography Standard RSA Laboratories; 14 June 2002, <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf>

[RFC 3852] Russ Housley: Cryptographic Message Syntax (CMS); RFC 3852; July 2004, <http://tools.ietf.org/html/rfc3852>

[RFC4051] D. Eastlake 3rd: Additional XML Security Uniform Resource Identifiers (URIs) ; RFC 4051; Apr. 2005 <http://tools.ietf.org/html/rfc4051>

[RFC 4055] Jim Schaad: 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 <http://tools.ietf.org/html/rfc4055>

[RFC 4056] Jim Schaad: Use of the RSASSA-PSS Signature Algorithm in Cryptographic Message Syntax (CMS); RFC 4056; June 2005 <http://tools.ietf.org/html/rfc4056>

[XMLDSig] XML-Signature Syntax and Processing W3C Recommendation 12 Feb. 2002, <http://www.w3.org/TR/xmlsig-core/>

[XMLEnc] XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002 <http://www.w3.org/TR/xmlenc-core/>

[KAL-PSS] Burt Kaliski: Raising the Standard for RSA Signatures: RSA-PSS, RSA Laboratories 26 Feb. 2003, <http://www.rsa.com/rsalabs/node.asp?id=2005>

[FIPS 186-3 Draft] Digital Signature Standard (DSS) FIPS 186-3, March 2006 http://csrc.nist.gov/publications/drafts/fips-186-3/Draft-FIPS-186-3%20_March2006.pdf

[FIPS 180-3 Draft] Secure Hash Standard (SHS), June 2007, http://csrc.nist.gov/publications/drafts/fips-180-3/draft_fips-180-3_June-08-2007.pdf

[NIST SP 800-107 Draft] Recommendation for Using Approved Hash Algorithms, NIST July 2007, <http://csrc.nist.gov/publications/drafts/Draft-SP-800-107/Draft-SP800-107.pdf>

[NIST SP 800-57 Draft] Recommendation for Key Management, NIST March 2007, http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf

[CaRe] Christophe De Canniere, Christian Rechberger: Finding SHA-1 Characteristics; Presented at the Second NIST Cryptographic Hash Workshop (Santa Barbara, California, USA, August 2006), to appear at ASIACRYPT 2006

[WaYaYa] Xiaoyun Wang, Andrew Yao, Frances Yao: Cryptanalysis of SHA-1. Presented at the First NIST Cryptographic Hash Workshop, Oktober 2005

[WaYiYu] Xiaoyun Wang, Yiqun Lisa Yin, Hongbo Yu: Finding Collisions in the full SHA-1; CRYPTO 2005 (Santa Barbara, California, USA, August 2005) Proceedings, volume 3621 of LNCS, pages 17–36. Springer, 2005. (editor: Victor Shoup)

Appendix: Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<ds:Signature
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
  xmlns:pss="http://www.example.org/xmldsig-pss/#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#WithComments"/>
    <ds:SignatureMethod
      Algorithm="http://www.example.org/xmldsig-pss/#rsa-pss"/>
    <ds:Reference>
      <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmenc#sha256"/>
      <ds:DigestValue>abc=</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>abc=</ds:SignatureValue>
</ds:Signature>
```

Figure 3: The simplest example of an RSA-PSS SignatureMethod

In Figure 3 the simplest usage of RSA-PSS in [XMLDSig] is shown. The defaults specified before come into operation: SHA-256 for the message digest and the mask generation function MFG1 also employing SHA-256; the default salt length of $256/8=32$ bytes derived from digest output length; the trailer field having the value 1 corresponding to '0xbc'.

```
<?xml version="1.0" encoding="UTF-8"?>
<ds:Signature
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
  xmlns:pss="http://www.example.org/xmldsig-pss/#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#WithComments"/>
    <ds:SignatureMethod
      Algorithm="http://www.example.org/xmldsig-pss/#rsa-pss">
      <pss:RSAPSSParams>
        <ds:DigestMethod
          Algorithm="http://www.w3.org/2001/04/xmenc#sha512"/>
        </pss:RSAPSSParams>
      </ds:SignatureMethod>
    <ds:Reference>
      <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmenc#sha512"/>
      <ds:DigestValue>abc=</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>abc=</ds:SignatureValue>
</ds:Signature>
```

Figure 4: A simple example of an RSA-PSS SignatureMethod

In Figure 4 a simple usage of RSA-PSS in [XMLDSig] employing SHA-512 is shown. SHA-512 for the message digest and the mask generation function MFG1 also employing SHA-512; the default salt length of $512/8=64$ bytes derived from digest output length; the trailer field having the value 1 corresponding to '0xbc'.

```
<?xml version="1.0" encoding="UTF-8"?>
<ds:Signature
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
  xmlns:pss="http://www.example.org/xmldsig-pss/#">
```

```

xmlns:pss="http://www.example.org/xmlsig-pss/#">
<ds:SignedInfo>
  <ds:CanonicalizationMethod
    Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#WithComments"/>
  <ds:SignatureMethod
    Algorithm="http://www.example.org/xmlsig-pss/#rsa-pss">
    <pss:RSAPSSParams>
      <ds:DigestMethod
        Algorithm="http://www.w3.org/2000/09/xmlsig#sha1"/>
      </pss:RSAPSSParams>
    </ds:SignatureMethod>
    <ds:Reference>
      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmlsig#sha1"/>
      <ds:DigestValue>abc</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>abc</ds:SignatureValue>
</ds:Signature>

```

Figure 5: An example employing SHA-1, compatible with the defaults chosen in RFC 4055

In Figure 5 a simple usage of RSA-PSS in [XMLDSig] employing SHA-1 is shown. SHA-1 is to be used for the message digest and the mask generation function MFG1 also employing SHA-1 together with a default salt length of 20 bytes and the trailer field having the value 1 corresponding to '0xbc'.

```

<?xml version="1.0" encoding="UTF-8"?>
<ds:Signature
  xmlns:ds="http://www.w3.org/2000/09/xmlsig#"
  xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
  xmlns:pss="http://www.example.org/xmlsig-pss/#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#WithComments"/>
    <ds:SignatureMethod
      Algorithm="http://www.example.org/xmlsig-pss/#rsa-pss">
      <pss:RSAPSSParams>
        <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmlsig#sha1"/>
        <pss:SaltLength>32</pss:SaltLength>
      </pss:RSAPSSParams>
    </ds:SignatureMethod>
    <ds:Reference>
      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmlsig#sha1"/>
      <ds:DigestValue>abc</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>abc</ds:SignatureValue>
</ds:Signature>

```

Figure 6: An example employing SHA-1, compatible with the defaults chosen in RFC 4055

In Figure 6 a simple usage of RSA-PSS in [XMLDSig] employing SHA-1 is shown. SHA-1 is to be used for the message digest and the mask generation function MFG1 also employing SHA-1 together with a custom salt length of 32 bytes and the trailer field having the value 1 corresponding to '0xbc'.