atompub	F. Chen
Internet-Draft	M. Hondo
Expires: February 12, 2010	R. Salz
	IBM
	August 11, 2009

Advanced Security Processing for Atom and APP Documents
draft-rsalz-atom-adv-sec-00.txt

Status of this Memo

Copyright Notice

Abstract

The Atom and APP specifications specify simple uses of cryptography to sign and encrypt their documents. Each document is processed completely, and in isolation. This document specifies additional uses that enable selective protection or encryption of content, and allow a "trust path" to be created across "atom:link" elements.

Table of Contents

1. Introduction

TOC

The Atom and APP specifications define how to use XML Digital Signature (Eastlake, D. and J. Reagle, “XML Encryption Syntax and Processing,” December 2002.) [W3C.REC-xmlenc-core-20021210], and XML Encryption (Eastlake, D. and J. Reagle, “XML Encryption Syntax and Processing,” December 2002.) [W3C.REC-xmlenc-core-20021210] to prevent the contents of a document from being modified or inadvertently disclosed. This specification profiles how to use WS-Security (Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, “Web Services Security: SOAP Message Security 1.1,” February 2006.) [oasis-wssec-v1.1] to provide selective protection and encryption of a document and allow a single document to be encrypted for multiple recipients.

Linking is a key point of Atom and APP, and this document defines a new element that can be used to establish a chain of trust, from the document with the "atom:link" to the content being referred-to.

1.1. Notational Conventions and Terminology

TOC

The conventions and terminology of both The Atom Syndication Format (Nottingham, M., Ed. and R. Sayre, Ed., “The Atom Syndication Format,” December 2005.) [RFC4287] and The Atom Publishing Protocol (Gregorio, J. and B. de hOra, “The Atom Publishing Protocol,” October 2007.) [RFC5023], known as Atom and APP respectively, are incorporated here by reference. Within this document, the phrase "Atom document" is used to refer to any document defined by either RFC.

The following namespace prefixes are used in this document:

Prefix	Namespace URI

atom	http://www.w3.org/2005/Atom
app	http://www.w3.org/2007/app
atomsec	http://www.ibm.com/xmlns/stdwip/atomext/security
dsig	http://www.w3.org/2000/09/xmldsig#
xenc	http://www.w3.org/2001/04/xmlenc#
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsu	"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

2. Using WS-Security

TOC

The "wsse:Security" element is part of a specification informally known as WS-Security. The element is defined as a container for security information, including security tokens, XML encryption keys, and/or XML digital signatures for SOAP message security. This specification uses the semantics of WS-Security processing to achieve the goals .
An Atom or APP dcoument MUST NOT have more than one "wsse:Security" child element. While compatible with [oasis-wssec-v1.1] (Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, “Web Services Security: SOAP Message Security 1.1,” February 2006.) processing, the schema for the "wsse:Security" element is taken from [oasis-wssec-v1.0] (Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, “Web Services Security: SOAP Message Security 1.0,” March 2004.).
One feature of the WS-Security header is that it should be possible to process in a forward-streaming manner. That is, security tokens appear before they are used (such as in signatures), and encipherment keys appear before the encrypted data. Therefore, if present, the "wsse:Security" element SHOULD be the first child of the Atom document.
The "wsse:SecurityTokenReference" element can be used to help enable streaming processing, and to reduce multiple occurrences of the same credentials. To do this, each credential is made the child of a "wsse:BinarySecurityToken" element, which in turn is a child of the "wsse:Security" element. For example:

```
<wsse:Security>
  <wsse:BinarySecurityToken
      wsu:Id="ssl-cert-1"
      ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-prof
    MII...
  </wsse:BinarySecurityToken>.
  ...
</wsse:Security>
```

Note that [W3C.REC-xmlenc-core-20021210] (Eastlake, D. and J. Reagle, “XML Encryption Syntax and Processing,” December 2002.) specifies many ways to identify key material, including Distinguished Name, Issuer/SerialNumber, etc. Any method other than embedding the entire certificate SHOULD NOT be used.
In this document, anywhere a certificate is referenced, the content may instead be a "wsse:SecurityTokenReference" to a token in the "wsse:Security" element. For example, the following two document fragments are semantically equivalent:

```
<dsig:KeyInfo>
  <dsig:X509Data>
    <wsse:SecurityTokenReference>
      <wsse:Reference URI="#ssl-cert-1"/>
    </wsse:SecurityTokenReference>
  </dsig:X509Data>
<dsig:KeyInfo>

<dsig:KeyInfo>
  <dsig:X509Data>
    <dsig:X509Certificate>
      MII...
    </dsig:X509Certificate>
  </dsig:X509Data>
<dsig:KeyInfo>
```

A reference other than a direct reference using the "wsse:Reference" element MUST NOT be used.

3. The "atomsec:credentials" element

TOC

The "atomsec:credentials" element is a child element of an "atom:link" element. It specifies one or more security tokens that may be used to validate the provenance of the data being linked-to. The "use" attribute specifies how the credentials are to be used. Two values are specified, "content" and "transport". If the attribute is not present, it is equivalent to providing it with the value "content".

```
atomSecCredentials =
    element atomsec:credentials {
        atomCommonAttributes*,
        attribute use { "content" | "transport" } ?,
        (extensionElement*)
    }
```

The "transport" value specifies that the credentials can be used to validate the SSL certificate of the server containing the linked-to data. The "content" value specifies that the credentials can be used to validate any digital signatures on the linked-to data.

An "atom:link" element can have no more than two "atom:credentials" elements, one for each of content and transport credentials.

The child elements of the "atomsec:credentials" element SHOULD be "dsig:KeyInfo" elmenents used to locate X.509 certificate data. These certificates can be used by the relying party to verify the specified trust chain.

A party that has out of band knowledge MAY ignore this element.

Here is an example of an Atom document pointing to signed content:

```
<atom:link href="http://www.example.org/2003/12/13/foo">
  <atomsec:credentials>
    <!-- This is the certificate that should be signing the content
    at the href.  Or the signer's CA... -->
    <dsig:KeyInfo>
      <dsig:X509Data>
        <dsig:X509Certificate>
          MII...
        </dsig:X509Certificate>
      </dsig:X509Data>
    </dsig:KeyInfo>
  </atomsec:credentials>
</atom:link>
```

Here is another example, one that specifies the CA that issued the SSL/TLS certificate used by the server for the host www.example.com:

```
<atom:link href="https://www.example.org/2003/12/13/foo">
  <atomsec:credentials use="transport">
    <!-- SSL CA, presumably not a typical "browser trusted" one. -->
    <dsig:KeyInfo>
      <dsig:X509Data>
        <dsig:X509Certificate>
          MII...
        </dsig:X509Certificate>
      </dsig:X509Data>
    </dsig:KeyInfo>
  </atomsec:credentials>
</atom:link>
```

4. Selectively Signing parts of an Atom document

TOC

A "dsig:Signature" element in the "wsse:Security" element may be used to sign one or more elements of an Atom document. The signature should be detached, with one "dsig:Reference" for each item being signed.

The processing rules are defined by sections 8.2 and 8.4 of [oasis-wssec-v1.1] (Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, “Web Services Security: SOAP Message Security 1.1,” February 2006.) with the following additions:

- *The SOAP message envelope is not used and the "wsse:Security" element MUST be the first element of the root document.

- *Each element being signed MUST have a unique "xml:id" attribute if not already present.

- *The enveloped-signature transform as defined by [W3C.REC-xmlenc-core-20021210] (Eastlake, D. and J. Reagle, “XML Encryption Syntax and Processing,” December 2002.) SHOULD be used to sign the root document. If there are additional signed elements, they MUST be signed first so that the root signature will include its descendant's signatures.

Here is an example:

```
<feed xmlns="http://www.w3.org/2005/Atom" xml:id="id3">

  <wsse:Security>

    <wsu:Timestamp xml:id="id1">
      <wsu:Created>XXXX-XX-XXT02:41:15Z</wsu:Created>
      <wsu:Expires>XXXX-XX-XXT02:46:15Z</wsu:Expires>
    </wsu:Timestamp>

    <wsse:BinarySecurityToken xml:id="SecurityToken-df8e8299-0d67-4eaf-bf5e-21c4dc6342ad
        ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-pr
      MII...
    </wsse:BinarySecurityToken>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
        <Reference URI="#id1">
          <Transforms>
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>cHt...</DigestValue>
        </Reference>
        <Reference URI="#id2">
          <Transforms>
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>Oj2...</DigestValue>
        </Reference>
      </SignedInfo>
      <SignatureValue>SWz...</SignatureValue>
      <KeyInfo>
        <wsse:SecurityTokenReference>
          <wsse:Reference URI="#SecurityToken-df8e8299-0d67-4eaf-bf5e-21c4dc6342ad"/>
        </wsse:SecurityTokenReference>
      </KeyInfo>
    </Signature>

    <!-- This is the enveloped signature, which covers all the other
         signatures and hence must be generated after all the other
         signatures are in place. -->
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
        <Reference URI="#id3">
```

```
          <Transforms>
            <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>cRY...</DigestValue>
        </Reference>
      </SignedInfo>
      <SignatureValue>QAw...</SignatureValue>
      <KeyInfo>
        <wsse:SecurityTokenReference>
          <wsse:Reference
              URI="#SecurityToken-df8e8299-0d67-4eaf-bf5e-21c4dc6342ad"/>
        </wsse:SecurityTokenReference>
      </KeyInfo>
    </Signature>
  </wsse:Security>

  <title>Example Feed</title>
  <link href="http://example.org/"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
     <name>John Doe</name>
  </author>
  <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>

  <entry xml:id="id1">
    <title>Atom-Powered Robots Run Amok</title>
    <link href="http://example.org/2003/12/13/atom03"/>
    <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
    <updated>2003-12-13T18:30:02Z</updated>
    <summary>Some text.</summary>
    <content type="xhtml" xml:lang="en" xml:base="http://diveintomark.org/">
      <div xmlns="http://www.w3.org/1999/xhtml">
        <p><i>[Update: The Atom draft is finished.]</i></p>
      </div>
    </content>

  </entry>
</feed>
```

5. Selectively Encrypting parts of an Atom document

TOC

The processing rules for encrypting parts of an Atom document are defined by sections 9.4.1 and 9.4.2 of [oasis-wssec-v1.1] (Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, “Web Services Security:

SOAP Message Security 1.1,” February 2006.) with the following additions:

- *The SOAP message envelope is not used and the "wsse:Security" element MUST be the first element of the root document.

- *Use WS-Security processing to encrypt the full document. If the encrypted result does not have to be valid according to the Atom schema's, then the [W3C.REC-xmlenc-core-20021210] (Eastlake, D. and J. Reagle, “XML Encryption Syntax and Processing,” December 2002.) standard SHOULD be used.

Here is an example:

```xml
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xml:id="id3">

  <wsse:Security>
    <xenc:EncryptedKey>
      <xenc:EncryptionMethod
          Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
      <dsig:KeyInfo>
        <wsse:SecurityTokenReference>
          <wsse:KeyIdentifier
              ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-to
            YPArt...
          </wsse:KeyIdentifier>
        </wsse:SecurityTokenReference>
      </dsig:KeyInfo>
      <xenc:CipherData xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
        <xenc:CipherValue>
        JG0...
        </xenc:CipherValue>
      </xenc:CipherData>
      <xenc:ReferenceList>
        <xenc:DataReference URI="#id1"/>
        <xenc:DataReference URI="#id2"/>
      </xenc:ReferenceList>
    </xenc:EncryptedKey>
  </wsse:Security>

  <title>Example Feed</title>
  <link href="http://example.org/"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
     <name>John Doe</name>
  </author>
  <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>

  <!-- this encrypted an atom:entry element. -->
  <xenc:EncryptedData Id="id1"
      Type="http://www.w3.org/2001/04/xmlenc#Element">
    <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
    <CipherData xmlns="http://www.w3.org/2001/04/xmlenc#">
      <CipherValue>
      Clm...
      </CipherValue>
    </CipherData>
  </xenc:EncryptedData>

  <!-- this encrypted another atom:entry element. -->
  <xenc:EncryptedData Id="id2" Type="http://www.w3.org/2001/04/xmlenc#Element">
```

```
         <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
         <CipherData xmlns="http://www.w3.org/2001/04/xmlenc#">
           <CipherValue>
           Clm...
           </CipherValue>
         </CipherData>
       </xenc:EncryptedData>

     </feed>
```

6. IANA Considerations

TOC

None.

7. Security Considerations

TOC

This entire document is about securing Atom documents. Authentication and authorization (e.g., to delete a update or delete a posting via APP) are not discussed here.

8. Normative References

TOC

[RFC4287]	Nottingham, M., Ed. and R. Sayre, Ed., "The Atom Syndication Format," RFC 4287, December 2005 (TXT, HTML, XML).
[RFC5023]	Gregorio, J. and B. de hOra, "The Atom Publishing Protocol," RFC 5023, October 2007 (TXT).
[W3C.REC-xmldsig-core-20080610]	Solo, D., Roessler, T., Reagle, J., Hirsch, F., and D. Eastlake, "XML Signature Syntax and Processing (Second Edition)," World Wide Web Consortium Recommendation REC-xmldsig-core-20080610, June 2008 (HTML).
[W3C.REC-xmlenc-core-20021210]	Eastlake, D. and J. Reagle, "XML Encryption Syntax and Processing," World Wide Web Consortium Recommendation REC-xmlenc-core-20021210, December 2002 (HTML).
[oasis-wssec-v1.0]	Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, "Web Services Security: SOAP Message Security 1.0," OASIS Standard 200401, March 2004.
[oasis-wssec-v1.1]	Nadalin, A., Kaler, C., Hallam-Baker, P., and R. Monzillo, "Web Services Security: SOAP Message Security 1.1," OASIS Standard (WS-Security 2004), February 2006.

Appendix A. Crypto and Security Primer

TOC

This section provides a very brief overview of common cryptographic techniques and how they are used. It is intended only to provide the smallest amount of information so that the rest of this document is understandable. It is not normative.

Public-key cryptography is done using two numbers that are related to each other. One of these numbers is public and can be known or used by anyone, while the other is private and should only be known by its "owner." The numbers have two very important properties. First, anything encrypted with the public key can only be decrypted with the private key. This means that anyone can generate text that is only legible to the key's owner. The second is that anything that is encrypted with the private key can be decrypted by anyone who has the public key. This allows anyone receiving such an encrypted item to know that it came from the key holder. The most common public-key algorithms are RSA and Elliptic-Curve.

A message digest, or hash takes an arbitrary sequence of bytes and returns a fixed-size identifier. The most important property is that it is statistically unlikely for two different streams to result in the same hash. The most useful hash algorithms are SHA-1 and SHA-256.

A message digest encrypted with a private key is a signature. Anyone with the original document can generate their own digest, decrypt the received one, and compare them. If the values do not match, then the document has been modified. The most common signature method is RSA/SHA-n.

In symmetric cryptography, the same key is used for encryption and decryption. This means that both sender and receiver have to have the same key. Common symmetric (or shared-secret) algorithms are AES and triple-DES.

A common technique is to "wrap" a symmetric key by encyrpting it with the recipient's public key. The bulk of the data is then encrypted using the symmetric key. This allows multiple recipients to see the data by only encrypting the symmetric key multiple times, rather than the entire data.

An X.509 certificate is a datum that contains a name and other identifying information, and a public key that can be associated with that name. The certificate also contains information such as its validity period, the ways in which the keys are to be used, and so on. For details see the IETF PKI Profile, RFC XXX. A certificate is signed by a CA, and common practice is for the entity receiving a certificate to be configured to trust everything signed by the CA, or perhaps the CA's CA, or further up the chain.

Authors' Addresses

TOC

Fred (Xiangfu) Chen
IBM
1 Rogers Street
Cambridge, MA 02142
USA
Phone: +1 617-693-6313
Email: chenfr@us.ibm.com
URI: http://www.ibm.com

Maryann Hondo
IBM
1 Rogers Street
Cambridge, MA 02142
USA
Phone: +1 617-693-1236
Email: mhondo@us.ibm.com
URI: http://www.ibm.com

Rich Salz
IBM
1 Rogers Street

	Cambridge, MA 02142
	USA
Phone:	+1 617-693-3808
Email:	rsalz@us.ibm.com
URI:	https://www.ibm.com/developerworks/mydeveloperworks/blogs/soma/