XEP-0274: Design Considerations for Digital Signatures in XMPP

Kurt Zeilenga
mailto:Kurt.Zeilenga@Isode.COM
xmpp:Kurt.Zeilenga@Isode.COM

2018-11-03
Version 0.4.1

Status	Type	Short Name
Deferred	Informational	N/A

This document discusses considerations for the design of Digital Signatures in XMPP, including use cases and requirements. The document also discusses various ways XML Digital Signatures could be used in XMPP.

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2018 by the XMPP Standards Foundation (XSF).

Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the ”Specification”), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

Warranty

NOTE WELL: This Specification is provided on an ”AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF’s Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

Contents

1 Introduction

Digital signatures may be used to provide a number of security services, including message authentication, message integrity and non-repudiation. There are many use cases for Digital Signatures in the Extensible Messaging and Presence Protocol (XMPP [1]).
XMPP can be described as a mean for exchanging structured information or stanzas between two or more entities. To accomplish this exchange, a number of other entities may be involved. For instance, communication of a stanza between two client entities will typically involve one or more server entities. Entities may exchange stanzas through service entities, such as a chat room service, to effect one-to-many communications.
Any entity involved in the exchange of a stanza may have wish to include one or more digital signatures for the benefit of any entity involved in the exchange:

- A client might wish to sign information it exchanges with another client for the benefit of this client (e.g, to provide message origin authentication service and content integrity service)
- A client might wish to sign a message in order to bind a Security Label to that message.
- A client may wish to sign information it sends to a chat room for the benefit of the chat room service and/or for the benefit of room occupants.
- The chat room service may wish to sign information it forwards to room occupants for the benefit of room occupants, such as to bind the client's JID to the client's room JID.
- A server involved in the exchange of a stanza between two clients may wish to sign information for the benefit of another server involved in the exchange (e.g., to provide delivery path validation).
- A server may wish to add additional data to a message, for example a Security Label, and bind that data to the message with a digital signature.

Digital signatures are provided to serve specific purposes. These purposes might include authentication of a particular entity involved in the exchange and integrity of information that entity provided.
This document discusses considerations for the design of general-purpose digital signature extension for XMPP. The document discusses use cases and requirements, as well as explores the solution space. The document also discusses existing solutions in this area.
This document contains a numerous examples intended to aide in the discussion of design issues. The examples are examples generally abbreviated and often use informal syntaxes.

[1] Extensible Messaging and Presence Protocol (XMPP) <https://xmpp.org/>.

2 Use Cases

2.1 Use in directed one-to-one stanzas

Directed one-to-one stanzas are stanzas which are exchanged between two entities, the originator of the stanza and intended recipient of that stanza, without exchanging through services which provide re-direction of stanzas (such as a groupchat service). The stanza may be handled by one or more other entities.
Examples of directed one-to-one stanzas include chat <message/> used in one-to-one chat sessions and <iq/> stanzas (excepting those exchanged through services providing re-direction).
The originator may wish to provide a signature for the benefit of the intended recipient. The intended recipient could use this signature to authenticate the originator and to ensure integrity of originator provided information.
Entities handling the stanza may wish to provide a signature for the benefit of the intended recipient. For instance, where a originator is a client and does not provide a signature, the client's server may wish to provide a signature for the benefit of the intended recipient. The intended recipient could use this signature to authenticate this server and to ensure integrity of the information as forwarded by this server.

2.2 Use in redirected one-to-one stanzas

Redirected one-to-one stanzas which are exchanged between two entities, the originator of the stanza and intended recipient of that stanza, through a service which provides re-direction of stanzas. The stanza may be handled by one or more other entities.
A multi-user chat (MUC) 'private message' is an example of redirected one-to-one stanza.
The originator's server may wish to provide a signature for the benefit of the re-direction service. The service could use this signature to authenticate the originator and to ensure integrity of originator provided information.
The originator may wish to provide a signature for the benefit of the intended recipient. The intended recipient could use this signature to authenticate the originator and to ensure integrity of originator provided information. However, this signature would by itself not establish any relationship between the signer and 'from' address in the stanza as received, nor does it establish this signature establish that the stanza was processed by the re-direction service. As in the directed one-to-one stanza, a originating client's server may wish to provide a signature for the benefit of the intended recipient.
The re-direction service may wish to provide a signature for the benefit of the intended recipient. The intended recipient could use this signature to authenticate the service and hence establish the service processed the stanza. The intended recipient could also use the signature to ensure the integrity of the information as redirected by the service.

2.3 Use in redirected one-to-group stanzas

Redirected one-to-many stanzas which are exchanged between two or more entities, the originator of the stanza and a group of recipients, through a service which provides re-direction of stanzas of a single stanza to a set of recipients. The stanza may be handled by one or more other entities.
A multi-user chat (MUC) message to all occupants is an example of redirected one-to-group stanza.
The originator's server may wish to provide a signature for the benefit of the re-direction service. The service could use this signature to authenticate the originator and to ensure integrity of originator provided information.
The originator may wish to provide a signature for the benefit of each recipient in the group. Each recipient could use this signature to authenticate the originator and to ensure integrity of originator provided information. However, this signature would by itself not establish any relationship between the signer and the 'from' address in the stanza as received, nor does it establish this signature establish that the stanza was processed by the re-direction service. As in the directed one-to-one stanza, a originating client's server may wish to provide a signature for the benefit of the each recipient.
The re-direction service may wish to provide a signature for the benefit of each recipient in the group. Each recipient could use this signature to authenticate the service and hence establish the service processed the stanza. Each could also use the signature to ensure the integrity of the information as redirected by the service.

2.4 Use in presence stanzas

The presence can be viewed as a specialized "publish-subscribe" mechanism. Commonly the publishing entity sends a <presence/> stanza to a presence service and the presence service than forwards the stanza to each subscriber. In basic user presence, the publishing entity is the user's client and the presence service is presence service is the provided by this client's server. In this case, the 'to' address is empty.
The publisher may wish to sign the signature for the benefit of each subscriber. Each subscriber could use this signature to authenticate the publisher and to ensure integrity of publisher provided information.
The presence service may wish to provide a signature for the benefit of each subscriber. Each subscriber could use this signature to authenticate the service and hence establish the service processed the stanza. Each could also use the signature to ensure the integrity of the information as redirected by the service.
A presence stanza may also directed to another entity, possibly through a re-direction service. This use is similar to the directed one-to-one and redirected one-to-one cases detailed above.

3 General Requirements

For the purposes of this memo, the following requirements are stipulated for a general solution:

1. The extension shall support client signing of stanzas.
2. The extension shall support service (e.g., multi-user chat service) signing of stanzas.
3. The extension shall support server signing stanzas.
4. The extension shall support multiple signatures in a stanza. That is, multiple entities can sign a stanza.
5. The extension shall support signing of <iq/> stanzas.
6. The extension shall support signing of <message/> stanzas, including chat and groupchat.
7. The extension shall support signing of <presence/> stanzas.
8. The extension shall support selective signing of stanzas. That is, a signer can sign select portions of a stanza.
9. The extension shall support signing of externally referenced object. That is, the signature may include a message digest of an external object, such as an HTTP accessible content.
10. The extension shall allow selective verification of signed elements.
11. The extension shall allow independent handling of verification errors in signed content.
12. The extension shall allow signers to provide signed copies of data likely to be modified by intermediate entities, such as stanza 'to' and 'from' attributes.
13. The extension should avoid duplication of content.
14. The extension must provide a means for relating signed content with unsigned content.
15. The extension should support querying for key information in XMPP (e.g., <iq/>).
16. The extension should support communicating key information through their XMPP-published vCard.
17. The extension should be designed such that the successful verification of a signature is independent of the extension support in entities involved in the exchange.
18. The extension should be compatible with object encryption, supporting encryption of signed content, signing of encrypted content, and signing of encrypted signed content (e.g., triple wrap content).

Some of above requirements may well be, if not outright mutually exclusive, in opposition to each other. It is suspected that set of reasonable solutions meeting all of the above requirements may be empty. To produce a reasonable solution, it is expected that some of the above requirements be eliminated and hence limiting the solution to some subset of the applications of digital signatures in XMPP.

4 Existing Solutions

4.1 XMPP E2E

The Internet Engineering Task Force (IETF) [2] standardized a signing and encryption facility for XMPP known as XMPP E2E [3]. XMPP E2E is based upon Secure/Multipurpose Internet Message Extensions (S/MIME [4]) and the Cryptographic Message Syntax (CMS [5]). As its name implies, XMPP E2E is intended to be an end-to-end solution. That is, it enables a sender to sign content sent to a specific recipient.
An advantage of the XMPP E2E approach is that it uses an encapsulating signature which protects the signed content from alteration as it is exchanged over an XMPP network. A disadvantage is that implementations which do not support XMPP E2E cannot make use of the signed content.
At the time of this writing, XMPP E2E has not been widely implemented. XMPP E2E appears to have limited applicability.

4.2 PGP signatures in XMPP

The Current Jabber OpenPGP Usage (XEP-0027) [6] (XMPP PGP), like the XMPP E2E, uses an encapsulating signature to protects the signed content from alteration as it is exchanged over an XMPP network. Like XMPP E2E, it is intended to be an end-to-end solution.
At the time of this writing, XMPP PGP has not been widely implemented (though some implementations do exist). XMPP PGP appears to have limited applicability.

[2] The Internet Engineering Task Force is the principal body engaged in the development of new Internet standard specifications, best known for its work on standards such as HTTP and SMTP. For further information, see <http://www.ietf.org/>.
[3] RFC 3923: End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP) <http://tools.ietf.org/html/rfc3923>.
[4] RFC 3851: Secure/Multipurpose Internet Mail Extensions (S/MIME) <http://tools.ietf.org/html/rfc3851>.
[5] RFC 3852: Cryptographic Message Syntax (CMS) <http://tools.ietf.org/html/rfc3852>.
[6] XEP-0027: Current Jabber OpenPGP Usage <https://xmpp.org/extensions/xep-0027.html>.

4.3 Encapsulating Digital Signatures in XMPP

The Encapsulating Digital Signatures in XMPP (XEP-0285) [7] (XMPP EgDSIG), like the XMPP E2E and XMPP PGP, uses an encapsulating signature to protects the signed content from alteration as it is exchanged over an XMPP network.
Unlike XMPP E2E and XMPP PGP, the solution is not intended to be strictly end-to-end in that multiple signers and verifiers where contemplated, now of which is necessarily an end entity. However, like XMPP E2E, it does not supported *optimistic signing*.

4.4 CDCIE-CCP

Alternative approaches have been developed. For instance, the Cross Domain Collaborative Information Environment (CDCIE [8]) Client Chat Protocol (CDCIE-CCP [9]), an XMPP-based protocol, supports signing of XMPP stanzas utilizes XML digital signatures (XMLDSIG [10]) ”enveloped” signatures over the whole stanza.
An advantage of the CDCIE-CCP approach is that, because it uses an encapsulated signature, implementations need not support CDCIE-CPP to make use of the stanza. The disadvantage is that the signature always over the entire stanza. Alteration of the stanza, as is common (often required) when exchanging stanzas over an XMPP network, will invalidate the signature.
While this approach has been implemented and deployed to some extent, the approach appears to have applicability limited to the CDCIE.

4.5 Encapsulated Digitial Signatures in XMPP

The Encapsulated Digital Signatures in XMPP (XEP-0290) [11] (XMPP DSIG) is an encapsulated signature proposal similar to that signed manifest approach suggested below. This approach is intended to support a wide range of uses and applications including *optimistic signing* in general communciations.
Unlike CDCIE-CCP approach, XMPP DSIG signatures are not ”enveloped” signatures over the whole stanza but signatures over an object which details the stanza contents.

[7] XEP-0285: Encapsulating Digital Signatures in XMPP <https://xmpp.org/extensions/xep-0285.html>.
[8] USFCOM fact sheet: Multinational Information Sharing and the Cross Domain Collaborative Information Environment <http://www.jfcom.mil/about/facts_prt/MNIS.pdf>.
[9] Cross Domain Collaborative Information Environment (CDCIE) Chat Client Protocol Specification, Version 2.0, Trident Systems, Inc., 12 March 2008
[10] XML Signature Syntax and Processing, W3C Recommendation, 10 June 2008 <http://www.w3.org/TR/xmldsig-core/>.
[11] XEP-0290: Encapsulated Digital Signatures in XMPP <https://xmpp.org/extensions/xep-0290.html>.

5 Protocol Design Discussion

5.1 Encapsulated v. Encapsulating Signatures

An encapsulating signature is a signature approach that encapsulates the signed content within the signature syntax. An encapsulated signature is a signature approach where the signature syntax in encapsulated within the structure of the signed content. XMPP E2E and XMPP PGP are examples of the former. CDCIE-CCP and XMPP DSIG are examples of the latter. The following example illustrates, using pseudo language, an encapsulating signature over a <message/> stanza.

Listing 1: Encapsulating Signature

```
<encapslating-signature>
          <signedInfo>
            <message to='romeo@example.net' type='chat'>
              <body>Art thou not Romeo, and a Montague?</body>
            </message>
          </signedInfo>
          <signature-over-signedInfo/>
        </encapslating-signature>
```

To transfer a signed <message/> using an encapsulating signature, one needs to send it within <message/> stanza.

Listing 2: Transfer of an Encapsulating Signature

```
<message to='romeo@example.net' type='chat'>
          <encapslating-signature>
            <signedInfo>
              <message to='romeo@example.net' type='chat'>
                <body>Art thou not Romeo, and a Montague?</body>
              </message>
            </signedInfo>
            <signature-over-signedInfo/>
          </encapslating-signature>
        </message>
```

The following example illustrates, using pseudo language, an encapsulated signature over a <message/> stanza.

Listing 3: Encapsulated Signature

```
<message to='romeo@example.net' type='chat'>
          <body>Art thou not Romeo, and a Montague?</body>
          <encapsulated-signature>
            <signature-over-message/>
          </encapsulated-signature>
```

```
        </message>
```

Applicability of a simple (non-nesting) encapsulating signatures, such as in XMPP E2E and XMPP PGP, are generally limited to end-to-end use cases. That is, cases where the originator of a stanza signs the stanza and send it through the XMPP network to its intended recipient, and only the intended recipient is expected to make use of the signed content. Entities between the signer and the intended recipient are expected to forward of the stanza without regard to the encapsulating signature, and without themselves signing the stanza. The approach does not require forwarding entities to support the signing extension.
Simple encapsulating signatures have limited applicability in MUC and PubSub use cases. For instance, an occupant can sign its submissions to the service for the benefit of the service and the service can sign reflected stanzas to occupants. In providing non-anonymous chat rooms, in addition to signing the reflected content, the service should include and sign the stanza it received when it was signed. This allows the occupants verify the content the service purports to have received, and to determine whether the reflected content is consistent given this. The following example illustrates an encapsulating signature over a groupchat <message/> stanza.

Listing 4: MUC submission with Encapsulating Signature

```
<message from='hag66@shakespeare.lit/pda' to='darkcave@chat.
   shakespeare.lit'
         type='groupchat' xml:lang='en'>
         <encapslating-signature>
           <signed-info>
             <message
               to='darkcave@chat.shakespeare.lit'
               type='groupchat'
               xml:lang='en'>
               <body>Harpier cries: 'tis␣time,␣'tis time.</body>
             </message>
           </signed-info>
           <signature-value>...</signature-value>
         </encapslating-signature>
       </message>
```

The following examples illustrates the signed reflection of the above stanza.

Listing 5: MUC reflection with Encapsulating Signature

```
<message from='darkcave@chat.shakespeare.lit/thirdwitch'
         to='crone1@shakespeare.lit/desktop' type='groupchat'
            xml:lang='en'>
         <encapslating-signature>
           <signed-info>
             <message
               from='darkcave@chat.shakespeare.lit/thirdwitch'
```

```
            to='crone1@shakespeare.lit/desktop' type='groupchat'
               xml:lang='en'>
              <body>Harpier cries: 'tis␣time,␣'tis time.</body>
            </message>
            <derived-from>
              <message from='hag66@shakespeare.lit/pda'
                to='darkcave@chat.shakespeare.lit' type='groupchat'
                   xml:lang='en'>
                <encapslating-signature>
                  <signedInfo>
                    <message to='darkcave@chat.shakespeare.lit'
                      type='groupchat' xml:lang='en'>
                      <body>Harpier cries: 'tis␣time,␣'tis time.</
                         body>
                    </message>
                  </signedInfo>
                  <signature/>
                </encapslating-signature>
              </message>
            </derived-from>
          </signed-info>
          <signature-value>...</signature-value>
        </encapslating-signature>
      </message>
```

In encapsulated signature solutions, as in CDCIE-CCP, any entities can make use of the signed content even if they do not support the signing extension. If the signature is over the entire stanza, as in CDCIE-CCP, the signature is likely not to be valid when the stanza is passed through multiple entities prior to verification. Hence, when the signature is over the entire stanza, the encapsulating signature approach applicability is generally limited to cases where there no entities between the signer and verifier. However, as discussed below, encapsulated selective signatures are generally more applicable.

5.2 Selective Signing

While an entity could provide a signature to be over the entire stanza, such signatures are likely be invalidated as the stanza exchanged over the XMPP. This is because XMPP allows and, in many cases, requires stanza to be modified as they are forwarded.
For instance, a client with the JID "juliet@example.com/Balcony" might send the following signed stanza:

Listing 6: Signature over entire stanza

```
<message to='romeo@example.net' type='chat' xml:lang='en'>
  <subject>Love</subject>
  <body>Art thou not Romeo, and a Montague?</body>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
```

```
    <SignedInfo>
      ...
      <Reference URI="">
        <Transforms>
          <Transform
            Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                signature"/>
        </Transforms>
        ...
      </Reference>
    </SignedInfo>
    <SignatureValue>...</SignatureValue>
    ...
  </Signature>
</message>
```

The example.com server is required, per XMPP Core [12], to add a 'from' attribute to the <message/> element before forwarding it to the example.net server. The example.net server is required to replace the 'to' attribute with the full JID of the romeo@example.net client it intends to forward the message to. These alternatations will "break" the signature.
XMLDSIG provides for a facility to selective sign XML content. For instance, the client could sign the <subject/> and <body/> element and their content. However, this by itself would not cover key aspects of the stanza, such that it was a chat <message/> addressed to a particular JID and sent from a particular JID. XMLDSIG allows for enveloping signatures, that is a signature that signs a data object contained within the <Signature/> element. The solution could define an element, such as <XMPPproperties/> used below, for including properties of the stanza in the signature.

5.3 Replay attack protection

The signature in Example 1 does not provide any protection against replay attack. To address replay attack, as well as other concerns, XMLDSIG defines the <SignatureProperties/> element for including information items about the generation of the Signature, such as the date/time the signature was generated.

5.4 Manifest Signing

While one could have <Signature/> which included a <Reference/> element for each of four elements discussed above within its <SignedInfo/> element, this would require reference validation for each <Reference/> (See 2.3 of XMLDSIG). To provide greater flexibility over handling of absent references and broken digest values, a <Manifest/> can be constructed and only it signed.

[12] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.

Putting all of the above together, the client might send the following signed stanza:

Listing 7: Client signed message

```
<message to='romeo@example.net' type='chat' xml:lang='en'>
  <subject id='X-subj'>Love</subject>
  <body id='X-body'>Art thou not Romeo, and a Montague?</body>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#" Id="sig">
    <SignedInfo>
      ...
      <Reference URI="#X-manifest">...</Reference>
    </SignedInfo>
    <SignatureValue>...</SignatureValue>
    <Object>
      <Manifest id='X-manifest'>
        <Reference URI="#X-subj">...</Reference>
        <Reference URI="#X-body">...</Reference>
        <Reference URI="#X-xmppprop">...</Reference>
        <Reference URI="#X-sigprop">...</Reference>
      </Manifest>
    </Object>
    <Object>
      <XMPPproperties id='X-xmppprop'>
        <stanza>message</stanza>
        <type>chat</type>
        <from>juliet@example.com</from>
        <to>romeo@example.net</to>
      </XMPPproperties>
    </Object>
    <Object>
      <SignatureProperties id="X-sigprop" Target="#X-sig">
          <SignatureProperty Target="#timestamp">
             <timestamp>2009-08-03T13:33:00Z</timestamp>
          </SignatureProperty>
        </SignatureProperties>
    </Object>
  </Signature>
</message>
```

5.5 Unambigious identification of content

The signature references needs to unambiguously identify content in stanza even in face of subsequent modification of that stanza. Failure to unambiguously identify signed content would also be problematic.
In the above example, signed child elements of the stanza were identified by 'id' attribute. As stanzas may be forwarded into any XMPP stream, such identifiers needs to remain unique.
Use of an extension attribute to identify elements may be problematic. In particular, the

XMPP specifications provide no assurance that this attribute would be forwarded with element. While one could identify signed content by other means, such as XPointer [13], these means would not unambiguously identify the signed content in the face of subsequent stanza modification.
The an 'id' attribute is could be used (or possibly 'xml:id'), it may be appropriate for the XMPP entity inserting a child element into a stanza to provide an 'xml:id' attribute regardless of what stanza content it might sign.

5.6 Multiple Signatures

Multiple entities can sign a stanza. A single entity may sign a stanza multiple times, typically on different occasions.
Each signer simply adds their <Signature/> element to the stanza, typically as the last element. A <Signature/> may sign other signatures, or portions thereof.
While a simple chat <message/> typically transits through only one or two XMPP servers and a groupchat <message/> may typically transits one to three XMPP servers, a stanza might include far more than four <Signature/> elements.

5.7 Optimistic Signing

Some users design the ability to *optimistic signing* of stanzas. That is, to sign all stanzas adhere to a configured criteria, such as all <message/> stanzas, they send. A key aspect of optimistic signing is that receiving entities not supporting the signing extension should be able to make use the message content (excluding the signature information) while those receiving entities supporting the extension can make use of the message content and the signature information.
Optimistic signing is available in E-mail through the use of S/MIME detached signatures. Use of S/MIME detached signatures can be problematic. Mail systems, especially restribution services such as mailing lists, are notorious for changing the signed content and hence breaking the signature.
In XMPP, as stanzas are generally altered in transit and hence optimistic signing will be fragile at best. Through use of selective signing and manifesting, issues may be mitigated to some degree. It is doubtful that a solution exists that provides optimistic signing and reliability verification.

5.7.1 Dual content

One possible optimistic signing solution is for stanzas to carry *alternative* sets of content, an unsigned content alternative and a signed content alternative. The premise of this approach is that an entity supporting the signing extension could make use of the signed content alternative while an entity not supporting the signing extension could make use of

[13] XML Pointer Language (XPointer), W3C Recommendation, 8 June 2001 <http://www.w3.org/TR/xptr>.

the unsigned content alternative. The approach has been suggested to as a mechanism for support extension-unaware entities downstream of extension-unware groupchat (or like) services use of the stanza content.
The following example not only illustrate this approach, but highlights some of the issues with this approach:

Listing 8: Dual content message

```
<message from='hag66@shakespeare.lit/pda' to='darkcave@chat.
    shakespeare.lit/laptop'
        type='groupchat' xml:lang='en'>
        <body>No.</body>
        <encapslating-signature>
          <signed-info>
            <message to='darkcave@chat.shakespeare.lit' type='
                groupchat' xml:lang='en'>
              <body>Yes.</body>
            </message>
          </signed-info>
          <signature-value>...</signature-value>
        </encapslating-signature>
        <delay xmlns='urn:xmpp:delay'
          from='shakespeare.lit'
          stamp='2002-09-10T23:08:25Z'/>
      </message>
```

But it should be obvious that the signed and unsigned contents are not proper alternatives. The signed content presumedly is what the signer sent. The unsigned content is presumedly a modified version of what the signer sent. The modifications are generally important to the entity making use of the stanza. In the above example, note that the to/from addresses of the signed content differ from the unsigned content. Note as well that the unsigned content contains a >delay/< element indicating that the stanza was delayed in transit. Such modifications are generally important to the proper processing of the content by not only this entity, but entities to which the content might be forwarded to. Dual content, even in absence of attacks, simply complicates such processing.
Note that the <body/> element values differ between the signed and unsigned content. While it reasonable straight forward (though significant work) to determine that the signed and unsigned content differ, it is extermely difficult to to determine whether the changes are due to normal processing or an attack.
Dual content adds significant blot. In simple cases, the approach effective doubles the content. However, in some use cases, the appraoch may lead to multiple doublings of the content.
It must be noted that verifying entities downstream of a redistribution will need some mechanism to determine who signed the stanza, determine what signer is an appropriate signer, and to obtain the public key of that signer. While certain information can be placed in key data, the question of whether the signer is an appropriate signer for purported sender (e.g., a room subscriber) generally would require information from the redistribution service,

and this would generally require the redistribution service to support an extension to make that information available to entities desiring to verify the signature(s). If one accepts the premise that downstream verification of redistributed stanzas, such as via a MUC service, cannot be performed without extension and cooperation of the redistribution service, then it follows that dual content can be avoided by having the MUC service also support the signing extension.
Dual content approaches should be avoided.

5.8 Key Info

While a signer may provide a <KeyInfo/> element within the <Signature/>, doing so will significantly increase the size of the <Signature/> element. As implementations may enforce a maximum stanza size as small as 10,000 bytes, use of <KeyInfo/> in stanza signatures should be limited.
It is also noted there are cases where the signer may not want to expose the key information to all entities involved in the exchange of stanza.
There are a number of ways key information may be published, such as in user's vCard. Key information can also be provided at request, such as by <iq/>.

6 Security Considerations

Care must be taken in the design of not only ensure it provides an effective digital signature solution for XMPP, but is designed itself with security in mind. This section discussions some security issues in providing a digital signature solution. The design should consider a general digital signature issues as well issues specific to the technologies used/involved, and particulars of the solution.
Due to the nature of XML and XMPP, an effective general digital signing solution for XMPP is likely to be quite complex. This document suggests nothing less. With complexity comes significant security risk. To minimize this risk, the solutions should avoid reinvention of needed technology, such as signature and key information syntaxes, by reusing well established and understood technologies such as XMLDSIG. Solutions should also favor simple and widely used features of such technologies over esoteric or rarely used features
Designers of the solution should be mind full of security considerations discussed in XMLDSIG (regardless of whether XMLDSIG is used in the solution)
If XMLDSIG is used, a number of security considerations would be introduced into the solution. Implementations need to take special care in processing XMLDSIG <Signature/> elements to avoid a wide range of attacks. For instance, an attacker could attempt to mount a Denial of Service attack by sending a <Signature/> purporting to sign arbitrary large and complex content. Or an attacker could attempt to mount a Distributed Denial of Service sending a message to a chatroom that containing <Signature/> with multiple references to large content hosted at the attack target in hopes that each room participant will repeated fetch it.

A <Signature/> element might also contain circler references.