XEP-0241: Encryption of Archived Messages

Ian Paterson
mailto:ian.paterson@clientside.co.uk
xmpp:ian@zoofy.com

2008-04-30
Version 0.1

Status	Type	Short Name
Deferred	Standards Track	NOT_YET_ASSIGNED

This specification defines methods for encrypting messages that are archived at an XMPP server.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

Message Archiving (XEP-0136) [1] defines a technology for archiving messages at an XMPP server instead of locally on a client device. This specification defines methods for encrypting such messages.

2 Encryption of Manually-Archived Collections

Clients SHOULD encrypt manually-archived collections (although early implementations of this protocol MAY prefer to defer encryption and decryption to later releases). Servers MUST support the manual-archiving of encrypted collections.
Before uploading a sequence of messages to a collection, the client SHOULD select a symmetric data encryption algorithm, generate a suitable random encryption key, give the key a unique (for the user) name, encrypt the symmetric key with one of the user's public keys, and wrap the result inside one or more <EncryptedKey/> elements, as specified in XML Encryption [2].
To ensure that all its user's clients will be able to decrypt the collection, the client SHOULD create one <EncryptedKey/> element for each of its user's public keys that are being published using Public Key Publishing (XEP-0189) [3]. However, the client MUST NOT create an <EncryptedKey/> element for any public key until it has confirmed that it belongs to the user. Note: The fact that a public key is being published using Public Key Publishing (XEP-0189) [4] is *not* sufficient proof of ownership, since the user's server may have been compromised at some stage. The method of confirmation is beyond the scope of this document.
The client SHOULD use the symmetric key to encrypt the joined sequence of <to/>, <from/> and <note/> elements, base64 encode the resulting sequence of bytes, and wrap it inside an <EncryptedData/> element, as described in XML Encryption.
Clients may add one or more <EncryptedData/> or <EncryptedKey/> elements to a collection using exactly the same method as for <to/>, <from/> and <note/> elements (see Uploading Messages to a Collection). One collection may contain <EncryptedData/> elements encrypted with different symmetric keys.
When appending <EncryptedData/> elements to a collection, the client MAY reuse a symmetric Key that has already been uploaded to the collection. In this case the client SHOULD NOT resend <EncryptedKey/> elements.
Note: A collection that contains <EncryptedData/> or <EncryptedKey/> elements MUST NOT contain <to/> or <from/> or <note/> elements.

Listing 1: Storing encrypted messages and keys in a collection

```
<iq type='set' id='crypt1'>
  <save xmlns='urn:xmpp:tmp:archive'>
```

[1] XEP-0136: Message Archiving <https://xmpp.org/extensions/xep-0136.html>.
[2] XML Encryption Syntax and Processing <http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/>.
[3] XEP-0189: Public Key Publishing <https://xmpp.org/extensions/xep-0189.html>.
[4] XEP-0189: Public Key Publishing <https://xmpp.org/extensions/xep-0189.html>.

```
    <chat with='juliet@capulet.com/chamber'
          start='1469-07-23T19:22:31Z'
          subject='She␣speaks!'>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
                     Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <EncryptionMethod Algorithm='http://www.w3.org/2001/04/xmlenc#
            aes128-cbc'/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>dataKey1</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>+OGQ0SR+
            ysraP6LnD43m77VkIVni5c7yPeIbkFdicZ</CipherValue></
            CipherData>
      </EncryptedData>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey1</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey1fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey1</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey2fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
    </chat>
  </save>
</iq>
```

The <CipherData/> child of each <EncryptedData/> element contains the base64-encoded symmetric-encrypted messages. The <EncryptionMethod/> and <KeyInfo/> children specify the symmetric encryption algorithm and the name of the symmetric key used to encrypt the messages.

The <CarriedKeyName/> child of each <EncryptedKey/> element contains the name of the symmetric key it contains. The name is referenced by the <KeyName/> child of the <KeyInfo/> child of an <EncryptedData/> element. The <CipherData/> child of each <EncryptedKey/> element contains the base64-encoded public-key-encrypted symmetric key. The <EncryptionMethod/> and <KeyInfo/> children specify the public key encryption algorithm and the name of the public key used to encrypt the symmetric key. The name of the public key MAY

refer to the name in the <KeyName/> child of one of the <KeyInfo/> elements that are being published using Public Key Publishing (XEP-0189) [5].

Listing 2: Private chat with encrypted attributes form

```
<iq type='set' id='form2'>
  <save xmlns='urn:xmpp:tmp:archive'>
    <chat with='benvolio@montague.net'
          start='1469-07-21T03:01:54Z'>
      <x xmlns='jabber:x:data' type='submit'>
        <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
                       Type='http://www.w3.org/2001/04/xmlenc#Content'
                          >
          <EncryptionMethod Algorithm='http://www.w3.org/2001/04/
              xmlenc#aes128-cbc'/>
          <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
            <KeyName>dataKey1</KeyName>
          </KeyInfo>
          <CipherData><CipherValue>+OGQ0SR+
              ysraP6LnD43m77VkIVni5c7yPeIbkFdicZ</CipherValue></
              CipherData>
        </EncryptedData>
      </x>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey1</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey1fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
    </chat>
  </save>
</iq>
```

The x:data form MAY be removed from a collection simply by uploading an empty form. Note: The server SHOULD NOT return an error if it finds that the form to be deleted does not exist.

Listing 3: Deleting the attributes form

```
<iq type='set' id='form3'>
  <save xmlns='urn:xmpp:tmp:archive'>
    <chat with='benvolio@montague.net'
          start='1469-07-21T03:01:54Z'>
      <x xmlns='jabber:x:data' type='submit'/>
```

[5] XEP-0189: Public Key Publishing <https://xmpp.org/extensions/xep-0189.html>.

```
    </chat>
  </save>
</iq>
```

3 Enabling Auto-Archiving with Encryption

The client can enable auto-archiving with server-side encryption by setting the 'save' attribute to "true" or "1" and setting the 'encrypt' attribute to "true" or "1".

Listing 4: Client enables auto archiving with encryption

```
<iq type='set' id='auto2'>
  <auto encrypt='true' save='true' xmlns='urn:xmpp:tmp:archive'/>
</iq>
```

If the server does not support encryption but the client attempts to enable encryption, the server MUST return a <feature-not-implemented/> error.

Listing 5: Server Does Not Support Encrypted Messages

```
<iq type='error' id='auto2'>
  <error type='cancel'>
    <feature-not-implemented xmlns='urn:ietf:params:xml:ns:xmpp-
        stanzas'/>
  </error>
</iq>
```

If the server supports encryption but there is no public key available for the user (e.g., as published via Public Key Publishing (XEP-0189) [6], the server MUST return a <not-acceptable/> error.

Listing 6: No Public Key Available

```
<iq type='error' id='auto2'>
  <error type='modify'>
    <not-acceptable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the server supports encryption (see Determining Server Support), it MUST encrypt all the messages that it archives automatically (including any message collections that are currently being recorded) by following exactly the same procedure as clients use when manually archiving collections (see Encryption).
The client MAY also specify one or more public keys (in addition to any public keys that the user may be publishing using Public Key Publishing (XEP-0189) [7]). The server MUST use them

[6] XEP-0189: Public Key Publishing <https://xmpp.org/extensions/xep-0189.html>.
[7] XEP-0189: Public Key Publishing <https://xmpp.org/extensions/xep-0189.html>.

all to encrypt all the symmetric keys it generates and add these to the collection wrapped in <EncryptedKey/> elements.

Listing 7: Client enables auto archiving with encryption

```
<iq type='set' id='auto2'>
  <auto save='true'
        encrypt='true'
        xmlns='urn:xmpp:tmp:archive'>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyValue>
        <KeyName>romeoPublicKey3fingerprint</KeyName>
        <RSAKeyValue>
          <Modulus>xA7SEU+
              e0yQH5rm9kbCDN9o3aPIo7HbP7tX6WOocLZAtNfyxSZDU16ksL6W
            jubafOqNEpcwR3RdFsT7bCqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVRgBUwUlV

            5foK5hhmbktQhyNdy/6LpQRhDUDsTvK+g9Ucj47es9AQJ3U=
          </Modulus>
          <Exponent>AQAB</Exponent>
        </RSAKeyValue>
      </KeyValue>
    </KeyInfo>
  </auto>
</iq>
```

As soon as the server has finished archiving a collection, it MUST securely destroy all copies of the symmetric key it used to encrypt the messages. Note: If the security of the server is compromised, then only the collections being recorded during the attack will be revealed (i.e. only those messages that would have been compromised even if they had not been archived).

4 Retrieving a List of Encrypted Collections

If a collection contains <EncryptedData/> or <EncryptedKey/> elements then the 'crypt' attribute of the <chat/> element MUST be set to 'true':

Listing 8: Receiving the first page of a list

```
<iq type='result' to='romeo@montague.net/orchard' id='list1'>
  <list xmlns='urn:xmpp:tmp:archive'>
    <chat with='juliet@capulet.com/chamber'
          start='1469-07-21T02:56:15Z'
          subject='She␣speaks!'
          crypt='true'
          version='0'/>
    .
```

```
    [28 more collections]
    .
    <chat with='balcony@house.capulet.com'
          start='1469-07-21T03:16:37Z'
          version='4'/>
    <set xmlns='http://jabber.org/protocol/rsm'>
      <first index='0'>1469-07-21T02:56:15Zjuliet@capulet.com</first>
      <last>1469-07-21T03:16:37Zbalcony@house.capulet.com</last>
      <count>1372</count>
    </set>
  </list>
</iq>
```

5 Retrieving an Encrypted Collection

The items in encrypted collections are typically larger than the items in an unencrypted collection, since each <EncryptedData/> element typically contains many messages. So the client SHOULD take even more care not to request a page of <EncryptedData/> elements that is so big it might exceed rate limiting restrictions.

Listing 9: Requesting the first page of an encrypted collection with all versions of keys

```
<iq type='get' id='page1'>
  <retrieve xmlns='urn:xmpp:tmp:archive'
            with='juliet@capulet.com/chamber'
            start='1469-07-23T19:22:31Z'
    <set xmlns='http://jabber.org/protocol/rsm'>
      <max>5</max>
    </set>
  </retrieve>
</iq>
```

In addition to the requested <EncryptedData/> elements, the server MUST return all the <EncryptedKey/> elements that it possesses for the user whose symmetric key name (wrapped in its <CarriedKeyName/> child) is referenced by the <KeyName/> child of the <KeyInfo/> child of any of the <EncryptedData/> elements in the returned page.

Listing 10: Receiving the first page of an encrypted collection

```
<iq type='result' to='romeo@montague.net/orchard' id='page1'>
  <chat xmlns='urn:xmpp:tmp:archive'
        with='juliet@capulet.com/chamber'
        start='1469-07-23T19:22:31Z'
        subject='She␣speaks!'
        version='5'>
    <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
```

```
                Type='http://www.w3.org/2001/04/xmlenc#Content'>
    <EncryptionMethod Algorithm='http://www.w3.org/2001/04/xmlenc#
        aes128-cbc'/>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyName>dataKey1</KeyName>
    </KeyInfo>
    <CipherData><CipherValue>+OGQ0SR+
        ysraP6LnD43m77VkIVni5c7yPeIbkFdicZ</CipherValue></CipherData
        >
  </EncryptedData>
  .
  [3 more <EncryptedData/> elements]
  .
  <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
                Type='http://www.w3.org/2001/04/xmlenc#Content'>
    <EncryptionMethod Algorithm='http://www.w3.org/2001/04/xmlenc#
        aes128-cbc'/>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyName>dataKey2</KeyName>
    </KeyInfo>
    <CipherData><CipherValue>+OGQ0SR+
        ysraP6LnD43m77VkIVni5c7yPeIbkFdicZ</CipherValue></CipherData
        >
  </EncryptedData>
  <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CarriedKeyName>dataKey1</CarriedKeyName>
    <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
        rsa-1_5"/>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyName>romeoPublicKey1fingerprint</KeyName>
    </KeyInfo>
    <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue></
        CipherData>
  </EncryptedKey>
  <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CarriedKeyName>dataKey1</CarriedKeyName>
    <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
        rsa-1_5"/>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
      <KeyName>romeoPublicKey2fingerprint</KeyName>
    </KeyInfo>
    <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue></
        CipherData>
  </EncryptedKey>
  <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CarriedKeyName>dataKey2</CarriedKeyName>
    <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
        rsa-1_5"/>
    <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
```

```
        <KeyName>romeoPublicKey1fingerprint</KeyName>
      </KeyInfo>
      <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue></
          CipherData>
    </EncryptedKey>
    <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
      <CarriedKeyName>dataKey2</CarriedKeyName>
      <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
          rsa-1_5"/>
      <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
        <KeyName>romeoPublicKey2fingerprint</KeyName>
      </KeyInfo>
      <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue></
          CipherData>
    </EncryptedKey>
    <set xmlns='http://jabber.org/protocol/rsm'>
      <first index='0'>0</first>
      <last>4</last>
      <count>7</count>
    </set>
  </chat>
</iq>
```

The client MAY limit the number of <EncryptedKey/> elements that it receives by specifying the name of one or more public keys for which it holds the associated private keys. The name of each public key MUST be wrapped in a <KeyName/> element.

Listing 11: Requesting the first page of an encrypted collection with specified version of keys

```
<iq type='get' id='page1'>
  <retrieve xmlns='urn:xmpp:tmp:archive'
            with='juliet@capulet.com/chamber'
            start='1469-07-23T19:22:31Z'>
    <KeyName xmlns='http://www.w3.org/2000/09/xmldsig#'>
        romeoPublicKey1fingerprint</KeyName>
    <set xmlns='http://jabber.org/protocol/rsm'>
      <max>1</max>
    </set>
  </retrieve>
</iq>
```

If the request includes one or more <KeyName/> elements then the server MUST only return those <EncryptedKey/> elements whose public key name (wrapped in the <KeyName/> child of the <KeyInfo/> child) is specified in the request.

6 Replacing EncryptedKey Elements

If a private key becomes obsolete or compromised then it may be necessary for a client to replace all <EncryptedKey/> elements that contain symmetric keys encrypted with the public key that is associated with the obsolete private key.
The client first requests a list of the affected <EncryptedKey/> elements from all collections by sending a <keys/> element to the server:

Listing 12: Requesting the first page of a list of keys

```
<iq type='get' id='pubkey1'>
  <keys xmlns='urn:xmpp:tmp:archive'>
    <KeyName xmlns='http://www.w3.org/2000/09/xmldsig#'>
       romeoPublicKey1fingerprint</KeyName>
    <set xmlns='http://jabber.org/protocol/rsm'>
      <max>50</max>
    </set>
  </keys>
</iq>
```

The server MUST return only <EncryptedKey/> elements whose symmetric encryption key is encrypted with the obsolete public key specified in the <KeyName/> child of the request:

Listing 13: Receiving the first page of a list of keys

```
<iq type='result' to='romeo@montague.net/orchard' id='pubkey1'>
  <keys xmlns='urn:xmpp:tmp:archive'>
    <chat with='juliet@capulet.com/chamber'
          start='1469-07-23T19:22:31Z'
          version='6'>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey1</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey1fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey2</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey1fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
```

```
      </EncryptedKey>
    </chat>
    .
    [49 more sets of collection keys]
    .
    <set xmlns='http://jabber.org/protocol/rsm'>
      <first index='0'>1469-07-23T19:22:31Zjuliet@capulet.com</first>
      <last>1469-08-03T13:24:06Zbalcony@house.capulet.com</last>
      <count>3810</count>
    </set>
  </keys>
</iq>
```

The client decrypts each symmetric key with the obsolete private key and encrypts it again with the new public key. The client then wraps each symmetric key in an <EncryptedKey/> element and asks the server to archive it in its associated collection on the server (see Encryption):

Listing 14: Storing encrypted keys in a collection

```
<iq type='set' id='crypt1'>
  <save xmlns='urn:xmpp:tmp:archive'>
    <chat with='juliet@capulet.com/chamber'
          start='1469-07-23T19:22:31Z'>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey1</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey2fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
      <EncryptedKey xmlns='http://www.w3.org/2001/04/xmlenc#'>
        <CarriedKeyName>dataKey2</CarriedKeyName>
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#
            rsa-1_5"/>
        <KeyInfo xmlns='http://www.w3.org/2000/09/xmldsig#'>
          <KeyName>romeoPublicKey2fingerprint</KeyName>
        </KeyInfo>
        <CipherData><CipherValue>E5Qbvfa2gI5lBZMAHryv4g</CipherValue><
            /CipherData>
      </EncryptedKey>
    </chat>
  </save>
</iq>
.
[49 more sets of collection keys]
```

```
.
```

Finally, the client asks the server to delete from each collection all <EncryptedKey/> elements whose symmetric encryption key is encrypted with the obsolete public key:

Listing 15: Deleting key(s) from a collection

```
<iq type='get' id='delete1'>
  <delete xmlns='urn:xmpp:tmp:archive'
          with='juliet@capulet.com/chamber'
          start='1469-07-23T19:22:31Z'>
    <KeyName xmlns='http://www.w3.org/2000/09/xmldsig#'>
       romeoPublicKey1fingerprint</KeyName>
  </delete>
</iq>
.
[49 more delete requests]
.
```

7 Determining Server Support

A client discovers whether its server supports this protocol using Service Discovery (XEP-0030) [8].

Listing 16: Client Service Discovery request

```
<iq from='romeo@montague.net/orchard'
    id='disco1'
    to='montague.net'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

If the server supports the *service-side* encryption feature, it MUST return a <feature/> element with the 'var' attribute set to 'urn:xmpp:tmp:archive:encrypt' (see Protocol Namespaces regarding issuance of one or more permanent namespaces).

Listing 17: Server Service Discovery response

```
<iq from='montague.net'
    id='disco1'
    to='romeo@montague.net/orchard'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
    ...
```

[8] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
    <feature var='urn:xmpp:tmp:archive:encrypt'/>
    ...
  </query>
</iq>
```

8 Security Considerations

Because the subject of each collection will not be encrypted even if its messages are encrypted, the client MUST warn its human user (if any) before including 'subject' attributes on encrypted collections.

9 IANA Considerations

No interaction with the Internet Assigned Numbers Authority (IANA) [9] is required as a result of this document.

10 XMPP Registrar Considerations

10.1 Service Discovery Features

The XMPP Registrar shall include the following features in its registry of service discovery features (see <https://xmpp.org/registrar/disco-features.html>), where the string "urn:xmpp:tmp:archive" shall be replaced with the URN issued by the XMPP Registrar:

- urn:xmpp:tmp:archive:encrypt

11 XML Schema

If this specification is advanced to a status of Draft, the schema for the 'urn:xmpp:tmp:archive' namespace (see Protocol Namespaces regarding issuance of one or more permanent namespaces) shall be updated to add a boolean 'encrypt' attribute to the <auto/> element, as follows.

```
<xs:element name='auto'>
    <xs:complexType>
      <xs:sequence>
        <xs:any processContents='lax' namespace='##other' minOccurs='0
            ' maxOccurs='unbounded'/>
```

[9] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

```
      </xs:sequence>
      <xs:attribute name='encrypt' type='xs:boolean' use='optional'/>
      <xs:attribute name='save' type='xs:boolean' use='required'/>
    </xs:complexType>
  </xs:element>
```