XEP-0138: Stream Compression

Joe Hildebrand
mailto:jhildebr@cisco.com
xmpp:hildjj@jabber.org

Peter Saint-Andre
mailto:peter@andyet.net
xmpp:stpeter@stpeter.im
https://stpeter.im/

2009-05-27
Version 2.0

Status	Type	Short Name
Final	Standards Track	compress

This document defines an XMPP protocol extension for negotiating compression of XML streams, especially in situations where standard TLS compression cannot be negotiated. The protocol provides a modular framework that can accommodate a wide range of compression algorithms; the ZLIB compression algorithm is mandatory-to-implement, but implementations may support other algorithms in addition.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

XMPP Core [1] specifies the use of Transport Layer Security (TLS; see RFC 5246 [2]) for encryption of XML streams, and TLS includes the ability to compress encrypted traffic (see RFC 3749 [3]). However, not all computing platforms are able to implement TLS, and traffic compression may be desirable for communication by applications on such computing platforms. This document defines a mechanism for negotiating the compression of XML streams outside the context of TLS.

2 Use Case

The protocol flow is as follows:

Listing 1: Receiving Entity Offers Stream Compression Feature

```
<stream:features>
  <starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
  <compression xmlns='http://jabber.org/features/compress'>
    <method>zlib</method>
    <method>lzw</method>
  </compression>
</stream:features>
```

Note: The <compression/> element MUST contain at least one <method/> child element. Each <method/> element MUST contain XML character data that specifies the name of a compression method, and such method names SHOULD be registered as described in the Compression Methods Registry section of this document. The methods SHOULD be provided in order of preference.
The initiating entity then MAY request compression by specifying one of the methods advertised by the receiving entity:

Listing 2: Initiating Entity Requests Stream Compression

```
<compress xmlns='http://jabber.org/protocol/compress'>
  <method>zlib</method>
</compress>
```

Note: If the initiating entity did not understand any of the advertised compression methods, it SHOULD ignore the compression option and proceed as if no compression methods were advertised.
If the initiating entity requests a stream compression method that is not supported by the

[1] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.
[2] RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2 <http://tools.ietf.org/html/rfc5246>.
[3] RFC 3749: Transport Layer Security Protocol Compression Methods <http://tools.ietf.org/html/rfc3749>.

receiving entity, the receiving entity MUST return an <unsupported-method/> error:

Listing 3: Receiving Entity Reports That Method is Unsupported

```
<failure xmlns='http://jabber.org/protocol/compress'>
  <unsupported-method/>
</failure>
```

If the receiving entity finds the requested method unacceptable or unworkable for any other reason, it MUST return a <setup-failed/> error:

Listing 4: Receiving Entity Reports That Negotiation of Stream Compression Failed

```
<failure xmlns='http://jabber.org/protocol/compress'>
  <setup-failed/>
</failure>
```

Note: Failure of the negotiation SHOULD NOT be treated as an unrecoverable error and therefore SHOULD NOT result in a stream error. In particular, the initiating entity is free to retry the compression negotiation if it fails.
If no error occurs, the receiving entity MUST inform the initiating entity that compression has been successfully negotiated:

Listing 5: Receiving Entity Acknowledges Negotiation of Stream Compression

```
<compressed xmlns='http://jabber.org/protocol/compress'/>
```

Both entities MUST now consider the previous (uncompressed) stream to be null and void, just as with TLS negotiation and SASL negotiation (as specified in RFC 3920) and MUST begin compressed communications with a new (compressed) stream. Therefore the initiating entity MUST initiate a new stream to the receiving entity:

Listing 6: Initiating Entity Initiates New (Compressed) Stream

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='shakespeare.lit'>
```

If compression processing fails after the new (compressed) stream has been established, the entity that detects the error SHOULD generate a stream error and close the stream:

Listing 7: Entity Closes Stream Because of a Processing Error

```
<stream:error>
  <undefined-condition xmlns='urn:ietf:params:xml:ns:xmpp-streams'/>
  <failure xmlns='http://jabber.org/protocol/compress'>
```

```
    <processing-failed/>
  </failure>
</stream:error>
</stream:stream>
```

3 Business Rules

The following business rules apply:

- If stream compression is negotiated, it MUST be used in both directions.
- TLS compression and stream compression SHOULD NOT be used simultaneously.
- Stream compression MAY be offered after TLS negotiation if TLS compression is not used.

For detailed recommendations regarding the order of stream feature negotiation, refer to Recommended Order of Stream Feature Negotiation (XEP-0170) [4].

4 Mandatory-to-Implement Technologies

Support for the ZLIB compression method as specified in RFC 1950 [5] is REQUIRED.
All other methods are OPTIONAL; such methods may be defined in future specifications or by registration as described in the Compression Methods Registry section of this document.

5 Optional Technologies

Implementations MAY support the following methods in addition to ZLIB:

- Stream Compression with LZW (XEP-0229) [6]

6 Implementation Notes

When using ZLIB for compression, the sending application SHOULD complete a partial flush of ZLIB when its current send is complete. Note that this statement is deliberately somewhat

[4] XEP-0170: Recommended Order of Stream Feature Negotiation <https://xmpp.org/extensions/xep-0170.html>.
[5] RFC 1950: ZLIB Compressed Data Format Specification version 3.3 <http://tools.ietf.org/html/rfc1950>.
[6] XEP-0229: Stream Compression with LZW <https://xmpp.org/extensions/xep-0229.html>.

vague: the sending application may end up performing this partial flush after sending every XML stanza, but on the other hand may perform the partial flush only after sending a group of stanzas that have been queued up for delivery. When to flush the state of the compression application is up to the sending application.

7 Security Considerations

Stream encryption via TLS (as defined in RFC 3920) and stream compression (as defined herein) are not mutually exclusive, but stream encryption via TLS MUST be negotiated before negotiation of stream compression in order to secure the stream.
Many of the security considerations related to TLS compression (see Section 6 of RFC 3749) also apply to stream compression.

8 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [7].

9 XMPP Registrar Considerations

9.1 Stream Features

The XMPP Registrar [8] includes 'http://jabber.org/features/compress' in its registry of stream features.

9.2 Protocol Namespaces

The XMPP Registrar includes 'http://jabber.org/protocol/compress' in its registry of protocol namespaces.

[7] 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/>.

[8] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

9.3 Compression Methods Registry

The XMPP Registrar maintains a registry of compression methods at <https://xmpp.org/registrar/compress.html>.

9.3.1 Process

In order to submit new values to this registry, the registrant shall define an XML fragment of the following form and either include it in the relevant XMPP Extension Protocol or send it to the email address <registrar@xmpp.org>:

```
<method>
  <name>the XML character data of the method element</name>
  <desc>a natural-language description of the compression method</desc
     >
  <doc>the document that specifies or registers the compression method
     </doc>
</method>
```

The registrant may register more than one compression method at a time, each contained in a separate <method/> element.

9.3.2 Registration

```
<method>
  <name>zlib</name>
  <desc>the ZLIB compression method</desc>
  <doc>RFC 1950</doc>
</method>
```

10 XML Schemas

10.1 Stream Feature

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://jabber.org/features/compress'
    xmlns='http://jabber.org/features/compress'
    elementFormDefault='qualified'>

  <xs:annotation>
```

```
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0138: http://www.xmpp.org/extensions/xep-0138.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='compression'>
    <xs:complexType>
      <xs:sequence>
        <xs:element name='method' type='xs:NCName' maxOccurs='
            unbounded'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

10.2 Protocol Namespace

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://jabber.org/protocol/compress'
    xmlns='http://jabber.org/protocol/compress'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0138: http://www.xmpp.org/extensions/xep-0138.html
    </xs:documentation>
  </xs:annotation>

  <xs:import namespace='urn:ietf:params:xml:ns:xmpp-stanzas'
             schemaLocation='http://xmpp.org/schemas/stanzaerror.xsd'/
                >

  <xs:element name='compress'>
    <xs:complexType>
      <xs:sequence>
        <xs:element name='method' type='xs:NCName' minOccurs='1'
            maxOccurs='unbounded'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='compressed' type='empty'/>
```

```
  <xs:element name='failure'>
    <xs:complexType>
      <xs:choice>
        <xs:element name='setup-failed' type='empty'/>
        <xs:element name='processing-failed' type='empty'/>
        <xs:element name='unsupported-method' type='empty'/>
        <xs:sequence xmlns:err='urn:ietf:params:xml:ns:xmpp-stanzas'>
          <xs:group ref='err:stanzaErrorGroup'/>
          <xs:element ref='err:text' minOccurs='0'/>
        </xs:sequence>
      </xs:choice>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```