XEP-0138: Stream Compression

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.

NOTICE: The protocol defined herein is a Draft Standard of the XMPP Standards Foundation. Implementations are encouraged and the protocol is appropriate for deployment in production systems, but some changes to the protocol are possible before it becomes a Final Standard.

Document Information

Series: XEP
Number: 0138
Publisher: XMPP Standards Foundation
Status: Draft
Type: Standards Track
Version: 1.3
Last Updated: 2007-09-26
Approving Body: XMPP Council
Dependencies: XMPP Core
Supersedes: None
Superseded By: None
Short Name: compress
XML Schema for compress namespace:
<http://www.xmpp.org/schemas/compress.xsd>
XML Schema for feature namespace:
<http://www.xmpp.org/schemas/compress-feature.xsd>
Registry: <http://www.xmpp.org/registrar/compress.html>
Wiki Page: <http://wiki.jabber.org/index.php/Stream Compression (XEP-0138)>

Author Information

Joe Hildebrand

Email: jhildebrand@jabber.com
JabberID: hildjj@jabber.org

Peter Saint-Andre

JabberID: stpeter@jabber.org
URI: https://stpeter.im/

Legal Notices

Copyright

This XMPP Extension Protocol is copyright (c) 1999 - 2008 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.

Disclaimer of 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. In no event shall the XMPP Standards Foundation or the authors of this Specification be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification.

Limitation of 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 out of the use or inability to use 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.

IPR Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF's Intellectual Property Rights Policy (a copy of which may be found at <http://www.xmpp.org/extensions/ipr-policy.shtml> or obtained by writing to XSF, P.O. Box 1641, Denver, CO 80201 USA).

Discussion Venue

The preferred venue for discussion of this document is the Standards discussion list: <http://mail.jabber.org/mailman/listinfo/standards>.

Errata may be sent to <editor@xmpp.org>.

Relation to XMPP

The Extensible Messaging and Presence Protocol (XMPP) is defined in the XMPP Core (RFC 3920) and XMPP IM (RFC 3921) specifications contributed by the XMPP Standards Foundation to the Internet Standards Process, which is managed by the Internet Engineering Task Force in accordance with RFC 2026. Any protocol defined in this document has been developed outside the Internet Standards Process and is to be understood as an extension to XMPP rather than as an evolution, development, or modification of XMPP itself.

Conformance Terms

The following keywords as used in this document are to be interpreted as described in RFC 2119: "MUST", "SHALL", "REQUIRED"; "MUST NOT", "SHALL NOT"; "SHOULD", "RECOMMENDED"; "SHOULD NOT", "NOT RECOMMENDED"; "MAY", "OPTIONAL".

Table of Contents

---

1. Introduction

XMPP Core [1] specifies the use of Transport Layer Security (TLS; see **RFC 4346** [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:

Example 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:

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

Example 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:

Example 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:

Example 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:

Example 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:

Example 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.
- If both TLS (whether including TLS compression or not) and stream compression are used, then TLS MUST be negotiated first, followed by negotiation of stream compression.
- Because negotiation of stream compression should not be completed after application of any encryption layers and because SASL negotiation (see **RFC 3920**) may involve application of an encryption layer, stream compression SHOULD be negotiated after SASL negotiation. For detailed recommendations regarding the order of stream feature negotiation, refer to **Recommended Order of Stream Feature Negotiation** [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** [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 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.

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.

9.3 Compression Methods Registry

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

9.3.1 Process

In order to submit new values to this registry, the registrant must 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:import namespace='urn:ietf:params:xml:ns:xmpp-stanzas'/>

  <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='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>
```

Notes

1. RFC 3920: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc3920>.

2. RFC 4346: The Transport Layer Security (TLS) Protocol Version 1.1 <http://tools.ietf.org/html/rfc4346>.

3. RFC 3749: Transport Layer Security Protocol Compression Methods <http://tools.ietf.org/html/rfc3749>.

4. XEP-0170: Recommended Order of Stream Feature Negotiation <http://www.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 <http://www.xmpp.org/extensions/xep-0229.html>.

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 <http://www.xmpp.org/registrar/>.

Revision History

Version 1.3 (2007-09-26)

Moved specification of LZW algorithm to XEP-0229.

(psa)

Version 1.2 (2007-08-22)

Clarified when compression shall be considered to start; per XEP-0170, specified that compression should be negotiated after SASL.

(psa)

Version 1.1 (2005-12-14)

More completely specified error handling; mentioned LZW (DCLZ) method.

(psa)

Version 1.0 (2005-06-16)

Per a vote of the Jabber Council, advanced status to Draft.

(psa)

Version 0.5 (2005-05-18)

Modifications to address Council feedback: used RFC 3920 terminology; specified error conditions; specified ZLIB as mandatory to implement.

(psa)

Version 0.4 (2005-05-11)

Corrected several errors in the schemas.

(psa)

Version 0.3 (2005-03-28)

Specified compression methods registry.

(psa)

Version 0.2 (2004-09-28)

Fixed TLS text per list discussion.

(psa)

Version 0.1 (2004-07-16)

Initial version.

(jjh/psa)

END