XMPP
get your shopping cart ready!

Winfried Tilanus
xmpp:winfried@tilanus.com
mailto:winfried@tilanus.com
Fosdem’20 2020-02-02
© Winfried Tilanus 2020
CC BY-SA 4.0
XMPP: a modular protocol
Core and extensions

- Core
  - Basic functionalities
  - Maintained by IETF & IANA
  - Published in RFC’s

- Extensions (XEPs)
  - Whatever you want to add to the core
  - Anybody can write them
  - Can be private, can be published
XMPP Stack

- RFC-6121 XMPP Messaging & Presence
- RFC-7622 XMPP Addresses
- RFC-6120 XMPP Core
- RFC-7590 XMPP Over TLS
- RFC-7395 XMPP Over Websockets

XEP
XSF: XMPP Standards Foundation

- Elected members and elected teams
- Cooperates with IETF on core protocol
- Publishes and mandates a set of extensions
  - Free to implement by everyone

Others publish extensions too, like the IEEE on IOT
Note: advancement to ‘Draft’ and ‘Final’ is slow
Special XEPs

- Historical
  - Obsolete but still in use
- Informational
  - Best practices
- Procedural
  - Procedures of the XSF
- Informational & procedural don’t have ‘Draft’ and ‘Final’ but ‘Proposed’ and ‘Active’
<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>XEP-0001</td>
<td>XMPP Extension Protocols</td>
<td>Procedural</td>
<td>Active</td>
<td>2019-10-19</td>
</tr>
<tr>
<td>XEP-0002</td>
<td>Special Interest Groups (SIGs)</td>
<td>Procedural</td>
<td>Active</td>
<td>2002-01-11</td>
</tr>
<tr>
<td>XEP-0004</td>
<td>Data Forms</td>
<td>Standards Track</td>
<td>Final</td>
<td>2007-08-13</td>
</tr>
<tr>
<td>XEP-0009</td>
<td>Jabber-RPC</td>
<td>Standards Track</td>
<td>Final</td>
<td>2011-11-10</td>
</tr>
<tr>
<td>XEP-0012</td>
<td>Last Activity</td>
<td>Standards Track</td>
<td>Final</td>
<td>2008-11-26</td>
</tr>
<tr>
<td>XEP-0013</td>
<td>Flexible Offline Message Retrieval</td>
<td>Standards Track</td>
<td>Draft</td>
<td>2005-07-14</td>
</tr>
<tr>
<td>XEP-0019</td>
<td>Streamlining the SIGs</td>
<td>Procedural</td>
<td>Active</td>
<td>2002-03-20</td>
</tr>
<tr>
<td>XEP-0030</td>
<td>Service Discovery</td>
<td>Standards Track</td>
<td>Final</td>
<td>2017-10-03</td>
</tr>
<tr>
<td>XEP-0033</td>
<td>Extended Stanza Addressing</td>
<td>Standards Track</td>
<td>Draft</td>
<td>2017-01-11</td>
</tr>
<tr>
<td>XEP-0045</td>
<td>Multi-User Chat</td>
<td>Standards Track</td>
<td>Draft</td>
<td>2019-05-15</td>
</tr>
<tr>
<td>XEP-0047</td>
<td>In-Band Bytestreams</td>
<td>Standards Track</td>
<td>Final</td>
<td>2012-06-22</td>
</tr>
<tr>
<td>XEP-0048</td>
<td>Bookmarks</td>
<td>Standards Track</td>
<td>Draft</td>
<td>2007-11-07</td>
</tr>
<tr>
<td>XEP-0049</td>
<td>Private XML Storage</td>
<td>Historical</td>
<td>Active</td>
<td>2004-03-01</td>
</tr>
<tr>
<td>XEP-0050</td>
<td>Ad-Hoc Commands</td>
<td>Standards Track</td>
<td>Draft</td>
<td>2019-03-26</td>
</tr>
</tbody>
</table>
# XEP-0410: MUC Self-Ping (Schrödinger's Chat)

**Abstract**
This protocol extension for XEP-0045 Multi User Chat allows clients to check whether they are still joined to a chatroom.

**Author**
Georg Lukas

**Copyright**
© 1999 – 2020 XMPP Standards Foundation. SEE LEGAL NOTICES.

**Status**
Draft

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.

**Type**
Standards Track

**Version**
1.1.0 (2019-09-25)

![Document Lifecycle](chart)

## 1. Introduction

The Multi-User Chat (XEP-0045) [1] protocol was not designed to handle s2s interruptions or message loss well. Rather often, the restart of a server or a component causes a client to believe that it is still joined to a given chatroom, while the chatroom service does not know of this occupant.
Getting lost...

- 430 XEPs
- All kind of stuff
- Some deferred XEPs are widely used
- So how to find a XEP that is
  - needed for interop
  - and future proof?
Compliance suite to the rescue!

- Yearly updated
- List of XEPs to use for:
  - Core functions
  - Web
  - Instant Messaging
  - Mobile connections
- All split up to core server, advanced server, core client and advanced client
# 2.1 Core Compliance Suite

<table>
<thead>
<tr>
<th>Feature</th>
<th>Core Server</th>
<th>Core Client</th>
<th>Advanced Server</th>
<th>Advanced Client</th>
<th>Providers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core features</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>RFC 6120 [16] [17]</td>
</tr>
<tr>
<td>TLS</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>RFC 7590 [19]</td>
</tr>
<tr>
<td>Direct TLS</td>
<td>✗</td>
<td>✗</td>
<td>✔ [20]</td>
<td>✔</td>
<td>SRV records for XMPP over TLS (XEP-0368) [21]</td>
</tr>
<tr>
<td>Feature discovery</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>Service Discovery (XEP-0030) [22]</td>
</tr>
<tr>
<td>Feature broadcasts</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>Entity Capabilities (XEP-0115) [23]</td>
</tr>
<tr>
<td>Server Extensibility</td>
<td>✔</td>
<td>N/A</td>
<td>✔</td>
<td>N/A</td>
<td>Jabber Component Protocol (XEP-0114) [24]</td>
</tr>
<tr>
<td>Event publishing</td>
<td>✗</td>
<td>✗</td>
<td>✔ [25]</td>
<td>✔</td>
<td>Personal Eventing Protocol (XEP-0163) [26]</td>
</tr>
</tbody>
</table>
Software support

- Clients, servers and libraries document what XEPs they support
- Many of them have plug-ins to support extra XEPs
- Any payload that is send from client to client is always relayed by the XMPP servers, so you can easily create your own extension.
XEP-0114: Jabber Component Protocol

1. Introduction

The Jabber network has long included a wire protocol that enables trusted components to connect to Jabber servers. While this component protocol is minimal and will probably be superseded by a more comprehensive component protocol at some point, informational documentation of the existing protocol would be helpful for component and server developers. This specification provides such documentation.
XEP-0060: Publish-Subscribe

Abstract
This specification defines an XMPP protocol extension for generic publish-subscribe functionality. The protocol enables XMPP entities to create nodes (topics) at a pubsub service and publish information at those nodes; an event notification (with or without payload) is then broadcasted to all entities that have subscribed to the node. Pubsub therefore adheres to the classic Observer design pattern and can serve as the foundation for a wide variety of applications, including news feeds, content syndication, rich presence, geolocation, workflow systems, network management systems, and any other application that requires event notifications.

Authors
Peter Millard, Peter Saint-Andre, Ralph Meijer

Copyright
© 1999 – 2020 XMPP Standards Foundation. SEE LEGAL NOTICES.

Status
Draft

Type
Standards Track

Version
1.17.0 (2019-10-06)
XEP-0365: Server to Server communication over STANAG 5066 ARQ

Abstract
This specification defines operation over XMPP over the NATO STANAG 5066 data link service for point to point links (ARQ). This enables optimized XMPP performance over HF Radio (which STANAG 5066 was designed for) and over other data links using STANAG 5066.

Author
Steve Kille

Copyright
© 1999 – 2020 XMPP Standards Foundation. SEE LEGAL NOTICES.

Status
Deferred

WARNING: This document has been automatically Deferred after 12 months of inactivity in its previous Experimental state. Implementation of the protocol described herein is not recommended for production systems. However, exploratory implementations are encouraged to resume the standards process.

Type
Standards Track

Version
0.2.1 (2018-07-21)

1. Introduction

This specification arose from requirements to operate over HF Radio, which has exceedingly high latency (sometimes minutes) low bandwidth link characteristics, and a generally poor signal-to-noise ratio.
XEP-0301: In-Band Real Time Text

1. Introduction

This document defines a specification for real-time text transmitted in-band over an XMPP network.

Real-time text is transmitted instantly while it is being typed or created. The recipient can immediately read the sender's text as it is written, without waiting. It allows text to be used as conversationally as a telephone conversation, including in situations where speech is not practical (e.g., environments that must be quiet, environments too noisy to hear, restrictions on phone use, situations where speaking is a privacy or security concern, and/or when participant(s) are deaf or hard of hearing). It is also used for transmission of live transcription.
1. Introduction

Applications written on top of XMPP often need to exchange data that has no existing standard. Such applications are often written by

Abstract

This specification proposes a simple mechanism by which applications can transfer data safely, without needing additional protocol design work. It is intended to provide a protocol that is trivial to implement and can be driven with a simple API.

Author

Dave Cridland

Copyright

© 1999 – 2020 XMPP Standards Foundation. SEE LEGAL NOTICES.

Status

ProtoXEP

Type

Standards Track

Version

0.0.1 (2019-12-30)

WARNING: This document has not yet been accepted for consideration or approved in any official manner by the XMPP Standards Foundation, and this document is not yet an XMPP Extension Protocol (XEP). If this document is accepted as a XEP by the XMPP Council, it will be published at <http://xmpp.org/extensions/> and announced on the <standards@xmpp.org> mailing list.
XEP-0419 Improving Security

Table of Contents

1. Introduction
2. Requirements
3. Particulars
3.1. Legacy Namespaces
3.2. New-style Namespaces
3.3. Stanza Encryption
4. Security Considerations
5. IANA Considerations
6. XMPP Registrar Considerations
7. Acknowledgements

Appendices

A. Document Information
B. Author Information
C. Legal Notices
D. Relation to XMPP

We therefore recommend that all stanzas on the wire are fully encrypted with Double ROT-13. Given the following stanza:

Example 1. Original unencrypted stanza

```xml
<message from='chris.davidland@crap-security.example' to='lucas.george@shiteam.example' type='chat' id='12
   <some-metadata xmlns='urn:xmpps:example:metadata'/>
   <body>Hey, Lucas!</body>
</message>
```

The following shows a correctly encrypted stanza:

Example 2. Stanza with encrypted meta-data and payload

```xml
<message from='chris.davidland@crap-security.example' to='lucas.george@shiteam.example' type='chat' id='12
   <some-metadata xmlns='urn:xmpps:example:metadata'/>
   <body>Hey, Lucas!</body>
</message>
```
XEP-0419 Improving Security

Table of Contents

1. Introduction
2. Requirements
3. Particulars
3.1. Legacy Namespaces
3.2. New-style Namespaces
3.3. Stanza Encryption
4. Security Considerations
5. IANA Considerations
6. XMPP Registrar Considerations
7. Acknowledgements

Appendices

A. Document Information
B. Author Information
C. Legal Notices
D. Relation to XMPP

We therefore recommend that all stanzas on the wire are **only encrypted with Double ROT-13**, given the following stanza:

**Example 1.** Original unencrypted stanza

```xml
<message from='chris.davidland@crap-security.example' to='lucas.george@shitem.example' type='chat' id='12'
  <some-metadata xmlns='urn:xmpps:example:metadata'/>
  <body>Hey, Lucas!</body>
</message>
```

The following shows a correctly encrypted stanza:

**Example 2.** Stanza with encrypted meta-data and payload

```xml
<message from='chris.davidland@crap-security.example' to='lucas.george@shitem.example' type='chat' id='12'
  <some-metadata xmlns='urn:xmpps:example:metadata'/>
  <body>Hey, Lucas!</body>
</message>
```
Example 1. Traditional XMPP stanza

```xml
<presence/>
```

That string can be represented in binary as follows:

Example 2. Binary representation

```plaintext
00111100111000011100100110111001110010110111110
```

The bit sequence is therefore represented in Binary XMPP as follows (line breaks are provided only for the purpose of readability):

Example 3. Binary XMPP representation

```plaintext
<zero/><zero/><one/><one/><one/><zero/><zero/>
<zero/><one/><zero/><zero/><zero/><zero/><zero/>
<zero/><one/><one/><zero/><zero/><zero/><zero/>
<zero/><one/><zero/><zero/><one/><zero/><zero/>
<zero/><one/><zero/><zero/><zero/><zero/><zero/>
<zero/><one/><zero/><zero/><one/><zero/><zero/>
<zero/><one/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
<zero/><zero/><zero/><zero/><zero/><zero/><zero/>
```
Need to know more?

- Don’t forget to look at this years compliance suite
- Feel free to ask hints, tips and assistance!
- Developers channels:
  - xmpp:jdev@muc.xmpp.org
  - mailto:jdev@jabber.org