



# XMPP

## XEP-0286: XMPP on Mobile Devices

Dave Cridland

<mailto:dave.cridland@isode.com>

<xmpp:dave.cridland@isode.com>

2010-09-15

Version 0.1

Status	Type	Short Name
Deferred	Informational	NOT_YET_ASSIGNED

This document provides background information for XMPP implementors concerned with mobile devices operating in a cellular network such as 3G.

# Legal

## Copyright

This XMPP Extension Protocol is copyright © 1999 - 2011 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 <http://xmpp.org/about-xmpp/xsf/xsf-ipr-policy/> or obtained by writing to XMPP Standards Foundation, 1899 Wynkoop Street, Suite 600, Denver, CO 80202 USA).

# Contents

1	Introduction	1
2	Overview	1
3	Compression	1
4	Radio Power	2
4.1	Idle . . . . .	2
4.2	FACH . . . . .	2
4.3	DCH . . . . .	3
5	Conclusions	3
6	Notable Extensions	4
7	Acknowledgements	5
8	Security Considerations	5
9	IANA Considerations	5
10	XMPP Registrar Considerations	5

## 1 Introduction

The use of XMPP on mobile devices is little understood, since few XMPP implementors have good mobile knowledge, and few mobile engineers have good XMPP knowledge. In addition, as the mobile landscape has changed, optimal protocol designs and usage patterns have also changed. This has led to the sub-optimal combination of a large amount of mostly undocumented lore, as well as several outdated concepts being discussed as fact.

This XEP aims to provide useful background knowledge of mobile handset behaviours, and essentially distills a number of conversations with experienced mobile engineers and XMPP implementors, providing useful background as general suggestions.

## 2 Overview

Mobile handsets typically have two constraints - power and bandwidth. The advent of 3G technology and beyond has tended to mean that bandwidth is radically higher, and comparable to broadband speeds - however many operators still charge based on transferred data, hence bandwidth remains an important issue for cost purposes.

The major cost of power in the handset for our purposes is the radio - here, too, bandwidth plays a part, but as this document will show, the time the radio is forced to be available to receive also costs substantially.

Whilst this document refers to [RFC 3920](#)<sup>1</sup>, implementors are advised to take note of [RFC 6120](#)<sup>2</sup>.

## 3 Compression

XMPP is known to compress well. Both TLS, part of [XMPP Core](#)<sup>3</sup>, and [Stream Compression](#)<sup>4</sup> can provide access to the DEFLATE codec ([RFC 1951](#)<sup>5</sup>), which provides access to simple stream compression.

Compression ratios vary with usage, however, typical usage by a general client appears to show a 20% ratio (an 80% reduction in bandwidth) in longer sessions<sup>6</sup>. Server implementors should note that there is a substantial memory cost per codec of 300KB assuming maximum settings - this may be dramatically reduced by reducing the memory level and window bits of the implementation - lowering memory level primarily causes increased CPU usage, whereas lowering the window bits directly degrade compression.

At an exemplary point in one experiment, the author found the following figures<sup>7</sup>:

---

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

<sup>2</sup>RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <<http://tools.ietf.org/html/rfc6120>>.

<sup>3</sup>RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <<http://tools.ietf.org/html/rfc6120>>.

<sup>4</sup>XEP-0138: Stream Compression <<http://xmpp.org/extensions/xep-0138.html>>.

<sup>5</sup>RFC 1951 DEFLATE Compressed Data Format Specification version 1.3 <<http://tools.ietf.org/html/rfc1951>>.

<sup>6</sup>Fixed-purpose clients, such as the Buddycloud client, do see even lower ratios, approaching 10%.

<sup>7</sup>The compression ratio is here given as Original/Compressed, hence a 100% compression ratio is no compression at all, and 0% would represent infinite compression.

Window Bits	Compression Ratio (approx)
15	20%
14	22%
13	25%
12	30%
11	38%
10	43%
9	60%

Although there is an equal cost for the mobile device to compress, it is considered that the compression codec memory and CPU costs - while certainly translating into power cost - are outweighed by two factors. Firstly, they're likely to reduce the transmission cost by a greater amount, and secondly they will also reduce the encryption cost in TLS.

Care, however, should be taken not to use XEP-0138 compression when TLS compression is in effect.

## 4 Radio Power

Mobile handsets have a number of levels for radio activity. 3G radios can be either Idle, or else in an increasingly capable - and increasingly power-hungry - series of levels, through FACH to DCH.

For the purposes of investigating this, power consumption (or rather battery depletion rate, as current) and timeouts were measured on the 3UK network, with a Nokia E71, using the Energy Profiler. A "typical handset" mentioned here has a 1000mAh battery - some smartphones have up to 1500mAh. Note that all timeouts are under the control of the network operator, not the handset or application.

### 4.1 Idle

Idle state is when the radio is neither receiving nor transmitting. It may have live (although silent) TCP connections. The cost is low. There is also a PCH level, which is similarly low-power, and again is only used when the radio is silent.

The current here was measured as 8mA - likely affected more by the energy profiler than much else.

### 4.2 FACH

FACH uses a shared channel for low-bandwidth communications. Packet sizes must be small - around 128 octets maximum, although this is operator controlled. Raising to this state takes around 2.5 seconds before the data can flow - although the power cost rises instantly - and

after the session has returned to silence, it will remain at FACH level for some time. Note that this threshold includes the overheads from TCP and TLS, which are 52 and 5 octets respectively, leaving around 70 octets for the payload data. Here, the current was measured as 140mA, and a timeout of 8 seconds - this timeout is at the lower end of the expected range, which could be up to around 2 minutes. On a typical handset, this will exhaust the battery in around 7 hours.

### 4.3 DCH

DCH uses a dedicated channel for high bandwidth communications. Again, raising to this state takes 2.5 seconds at the DCH power level, and there is a timeout before dropping back. Some operators will drop back to FACH for the duration of the FACH timeout - others will drop back to Idle/PCH.

Sending more than the FACH threshold will raise the radio all the way to DCH - taking, again, 2.5 seconds.

The measurements here were 380mA and 8 seconds - this is sufficient to flatten a typical handset battery in less than 3 hours, and the figures are considered normal.

Transmission of data can use up to 2W<sup>8</sup>, and raising the level itself takes between 2 and 3 seconds to take effect - during which time the handset cannot receive or transmit, but still incurs the power cost. There are packets sent from and to the handset during this time.

Experimentation suggests that uncompressed XMPP will never trigger the FACH state, leaping directly into the more costly DCH state. However, compression does make FACH possible, if rare.

## 5 Conclusions

As with anything, there are no hard and fast rules. If there were, they might look like these. First, for devices:

Transmit no data. Transmitting costs significant power, and moreover raises the radio state. Not transmitting will allow it to maximize the time spent in the low-cost Idle state.

If you must transmit, then transmit only a small volume. If there is only a small amount of data transmitted - less than 128 octets typically - the radio will only raise to FACH, which is significantly cheaper than DCH.

If you must transmit, then compress as hard as possible. Since individual octets have an associate power - and often financial - cost, it's worth maximizing the compression algorithm, even if the volume of traffic will raise to DCH.

---

<sup>8</sup>The author's hazy recollection of  $P=IV$  suggests around a 570mA current

If you have transmit a lot, then do a lot. If the radio is raised to DCH anyway, then you may as well go fetch that avatar you were missing, since you're chewing through power anyway.

If you receive, then transmit. If your peer raises the radio state, you may as well use it.

And for servers, similar rules apply:

Send no data. Sending data will cause the handset to be raised out of Idle. This immediately costs massively higher power.

If you must send, send tiny bits. Sending small enough data maximizes the likelihood that the devices radio will only be raised to FACH levels.

If you receive, then send anything you have. Receiving data indicates that the radio is active - it'll stay active for some time, so sending data doesn't incur the overhead of raising the radio state, and won't increase power drain on the handset.

If you must send when not receiving, send plenty. Sending data will raise the radio's state - unless you can tell this will only raise it to FACH, it's worth sending as much as possible.

Finally, protocol designers should aim to minimize any responses required from the handset, and ensure keepalive traffic, if any, fits inside FACH wherever possible.

## 6 Notable Extensions

This section provides pointers to other documents which may be of interest to those developing mobile clients, or considering support for them in servers.

[Stream Compression](#)<sup>9</sup> provides application stream level compression, useful if the device TLS stack does not support TLS-based compression.

[Entity Capabilities](#)<sup>10</sup> provides a mechanism for caching, and hence eliding, the disco#info requests needed to negotiate optional features.

[Roster Versioning](#)<sup>11</sup> provides a relatively widely deployed extension for reducing the roster fetch bandwidth, in most cases reducing it to a simple affirmation that the client has the current roster. This saves not only bandwidth, but also reduces local storage writes.

[Stream Management](#)<sup>12</sup> provides session resumption over TCP, enabling a client to handle the case where the coverage is patchy. The <r/> and <a/> elements also provide a keepalive facility in a small number of octets.

[Stanza Interception and Filtering Technology](#)<sup>13</sup> provides a mechanism which, amongst other things, would allow a presence "hush", buffering presence during certain states.

---

<sup>9</sup>XEP-0138: Stream Compression <<http://xmpp.org/extensions/xep-0138.html>>.

<sup>10</sup>XEP-0115: Entity Capabilities <<http://xmpp.org/extensions/xep-0115.html>>.

<sup>11</sup>XEP-0237: Roster Versioning <<http://xmpp.org/extensions/xep-0237.html>>.

<sup>12</sup>XEP-0198: Stream Management <<http://xmpp.org/extensions/xep-0198.html>>.

<sup>13</sup>XEP-0273: Stanza Interception and Filtering Technology <<http://xmpp.org/extensions/xep-0273.html>>.

## 7 Acknowledgements

The author is not a mobile expert, and relied on the knowledge and patient help of several others. In particular, thanks are due to Jussi Laako, Markku Vampari, and Markus Isomaki of Nokia, and Simon Tennant of Buddycloud.

The attribution of any mistakes herein is zealously guarded by the author, however.

## 8 Security Considerations

This document does not discuss a protocol, thus introduces no new security considerations.

## 9 IANA Considerations

None.

## 10 XMPP Registrar Considerations

None.