Network Working Group
Request for Comments: 4622
Category: Standards Track

P. Saint-Andre
JSF

July 2006

# Internationalized Resource Identifiers (IRIs) and Uniform Resource Identifiers (URIs) for the Extensible Messaging and Presence Protocol (XMPP)

## Status of This Memo



## Copyright Notice



## Abstract

This document defines the use of Internationalized Resource Identifiers (IRIs) and Uniform Resource Identifiers (URIs) in identifying or interacting with entities that can communicate via the Extensible Messaging and Presence Protocol (XMPP).

Table of Contents

1. Introduction ....3
1.1. Terminology ....3
2. Use of XMPP IRIs and URIs ....4
2.1. Rationale ....4
2.2. Form ....4
2.3. Authority Component ....6
2.4. Path Component ....7
2.5. Query Component ....7
2.6. Fragment Identifier Component ....9
2.7. Generation of XMPP IRIs/URIs ....9
2.7.1. Generation Method ....9
2.7.2. Generation Notes ....10
2.7.3. Generation Example ....11
2.8. Processing of XMPP IRIs/URIs ....12
2.8.1. Processing Method ....12
2.8.2. Processing Notes ....13
2.8.3. Processing Example ....14
2.9. Internationalization ....14
3. IANA Registration of xmpp URI Scheme ....15
3.1. URI Scheme Name ....15
3.2. Status ....15
3.3. URI Scheme Syntax ....15
3.4. URI Scheme Semantics ....16
3.5. Encoding Considerations ....16
3.6. Applications/protocols That Use This URI Scheme Name ....16
3.7. Interoperability Considerations ....16
3.8. Security Considerations ....16
3.9. Contact ....17
3.10. Author/Change Controller ....17
3.11. References ....17
4. IANA Considerations ....17
5. Security Considerations ....17
5.1. Reliability and Consistency ....17
5.2. Malicious Construction ....18
5.3. Back-End Transcoding ....18
5.4. Sensitive Information ....18
5.5. Semantic Attacks ....19
5.6. Spoofing ....19
6. References ....20
6.1. Normative References ....20
6.2. Informative References ....20

## 1. Introduction

The Extensible Messaging and Presence Protocol (XMPP) is a streaming XML technology that enables any two entities on a network to exchange well-defined but extensible XML elements (called "XML stanzas") at a rate close to real time.

As specified in [XMPP-CORE], entity addresses as used in communications over an XMPP network must not be prepended with a Uniform Resource Identifier (URI) scheme (as specified in [URI]). However, applications external to an XMPP network may need to identify XMPP entities either as URIs or, in a more modern fashion, as Internationalized Resource Identifiers (IRIs; see [IRI]). Examples of such external applications include databases that need to store XMPP addresses and non-native user agents such as web browsers and calendaring applications that provide interfaces to XMPP services.

The format for an XMPP address is defined in [XMPP-CORE]. Such an address may contain nearly any [UNICODE] character and must adhere to various profiles of [STRINGPREP]. The result is that an XMPP address is fully internationalizable and is very close to being an IRI without a scheme. However, given that there is no freestanding registry of IRI schemes, it is necessary to define XMPP identifiers primarily as URIs rather than as IRIs, and to register an XMPP URI scheme instead of an IRI scheme. Therefore, this document does the following:

- Specifies how to identify XMPP entities as IRIs or URIs.
- Specifies how to interact with XMPP entities as IRIs or URIs.
- Formally defines the syntax for XMPP IRIs and URIs.
- Specifies how to transform XMPP IRIs into URIs and vice-versa.
- Registers the xmpp URI scheme.

### 1.1. Terminology

This document inherits terminology from [IRI], [URI], and [XMPP-CORE].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [TERMS].

## 2. Use of XMPP IRIs and URIs

### 2.1. Rationale

As described in [XMPP-IM], instant messaging and presence applications of XMPP must handle im: and pres: URIs (as specified by [CPIM] and [CPP]). However, there are many other applications of XMPP (including network management, workflow systems, generic publish-subscribe, remote procedure calls, content syndication, gaming, and middleware), and these applications do not implement instant messaging and presence semantics. Neither does a generic XMPP entity implement the semantics of any existing URI scheme, such as the http:, ftp:, or mailto: scheme. Therefore, it is appropriate to define a new URI scheme that makes it possible to identify or interact with any XMPP entity (not just instant messaging and presence entities) as an IRI or URI.

XMPP IRIs and URIs are defined for use by non-native interfaces and applications, and primarily for the purpose of identification rather than of interaction (on the latter distinction, see Section 1.2.2 of [URI]). In order to ensure interoperability on XMPP networks, when data is routed to an XMPP entity (e.g., when an XMPP address is contained in the 'to' or 'from' attribute of an XML stanza) or an XMPP entity is otherwise identified in standard XMPP protocol elements, the entity MUST be addressed as <[node@]domain[/resource]> (i.e., without a prepended scheme), where the "node identifier", "domain identifier", and "resource identifier" portions of an XMPP address conform to the definitions provided in Section 3 of [XMPP-CORE].

(Note: For historical reasons, the term "resource identifier" is used in XMPP to refer to the optional portion of an XMPP address that follows the domain identifier and the "/" separator character (for details, refer to Section 3.4 of [XMPP-CORE]; this use of the term "resource identifier" is not to be confused with the meanings of "resource" and "identifier" provided in Section 1.1 of [URI]).

### 2.2. Form

As described in [XMPP-CORE], an XMPP address used natively on an XMPP network is a string of Unicode characters that (1) conforms to a certain set of [STRINGPREP] profiles and [IDNA] restrictions, (2) follows a certain set of syntax rules, and (3) is encoded as [UTF-8]. The form of such an address can be represented using Augmented Backus-Naur Form ([ABNF]) as:

```
[ node "@" ] domain [ "/" resource ]
```

In this context, the "node" and "resource" rules rely on distinct profiles of [STRINGPREP], and the "domain" rule relies on the concept of an internationalized domain name as described in [IDNA]. (Note: There is no need to refer to punycode in the IRI syntax itself, since any punycode representation would occur only inside an XMPP application in order to represent internationalized domain names. However, it is the responsibility of the processing application to convert [IRI] syntax into [IDNA] syntax before addressing XML stanzas to the specified entity on an XMPP network.)

Naturally, in order to be converted into an IRI or URI, an XMPP address must be prepended with a scheme (specifically, the xmpp scheme) and may also need to undergo transformations that adhere to the rules defined in [IRI] and [URI]. Furthermore, in order to enable more advanced interaction with an XMPP entity rather than simple identification, it is desirable to take advantage of additional aspects of URI syntax and semantics, such as authority components, query components, and fragment identifier components.

Therefore, the ABNF syntax for an XMPP IRI is defined as shown below using Augmented Backus-Naur Form specified by [ABNF], where the "ifragment", "ihost", and "iunreserved" rules are defined in [IRI], the "pct-encoded" rule is defined in [URI], and DQUOTE is defined in [ABNF]:

```
  xmppiri    = "xmpp" ":" ihierxmpp
               [ "?" iquerycomp ]
               [ "#" ifragment ]
  ihierxmpp  = iauthpath / ipathxmpp
  iauthpath  = "//" iauthxmpp [ "/" ipathxmpp ]
  iauthxmpp  = inodeid "@" ihost
  ipathxmpp  = [ inodeid "@" ] ihost [ "/" iresid ]
  inodeid    = *( iunreserved / pct-encoded / nodeallow )
  nodeallow  = "!" / "$" / "(" / ")" / "*" / "+" / "," / ";" /
               "=" / "[" / "\" / "]" / "^" / "`" / "{" / "|" /
               "}"
  iresid     = *( iunreserved / pct-encoded / resallow )
  resallow   = "!" / DQUOTE / "$" / "&" / "'" / "(" / ")" /
               "*" / "+" / "," / ":" / ";" / "<" / "=" / ">" /
               "[" / "\" / "]" / "^" / "`" / "{" / "|" / "}"
  iquerycomp = iquerytype [ *ipair ]
  iquerytype = *iunreserved
  ipair      = ";" ikey "=" ivalue
  ikey       = *iunreserved
  ivalue     = *( iunreserved / pct-encoded )
```

However, the foregoing syntax is not appropriate for inclusion in the registration of the xmpp URI scheme, since the IANA recognizes only URI schemes and not IRI schemes. Therefore, the ABNF syntax for an XMPP URI rather than for IRI is defined as shown in Section 3.3 of this document (see below under "IANA Registration"). If it is necessary to convert the IRI syntax into URI syntax, an application MUST adhere to the mapping procedure specified in Section 3.1 of [IRI].

The following is an example of a basic XMPP IRI/URI used for purposes of identifying a node associated with an XMPP server:

```
xmpp:node@example.com
```

Descriptions of the various components of an XMPP IRI/URI are provided in the following sections.

## 2.3. Authority Component

As explained in Section 2.8 of this document, in the absence of an authority component, the processing application would authenticate as a configured user at a configured XMPP server. That is, the authority component section is unnecessary and should be ignored if the processing application has been configured with a set of default credentials.

In accordance with Section 3.2 of RFC 3986, the authority component is preceded by a double slash ("//") and is terminated by the next slash ("/"), question mark ("?"), or number sign ("#") character, or by the end of the IRI/URI. As explained more fully in Section 2.8.1 of this document, the presence of an authority component signals the processing application to authenticate as the node@domain specified in the authority component rather than as a configured node@domain (see the Security Considerations section of this document regarding authentication). (While it is unlikely that the authority component will be included in most XMPP IRIs or URIs, the scheme allows for its inclusion, if appropriate.) Thus, the following XMPP IRI/URI indicates to authenticate as "guest@example.com":

```
xmpp://guest@example.com
```

Note well that this is quite different from the following XMPP IRI/URI, which identifies a node "guest@example.com" but does not signal the processing application to authenticate as that node:

```
xmpp:guest@example.com
```

Similarly, using a possible query component of "?message" to trigger an interface for sending a message, the following XMPP IRI/URI signals the processing application to authenticate as "guest@example.com" and to send a message to "support@example.com":

xmpp://guest@example.com/support@example.com?message

By contrast, the following XMPP IRI/URI signals the processing application to authenticate as its configured default account and to send a message to "support@example.com":

xmpp:support@example.com?message

## 2.4. Path Component

The path component of an XMPP IRI/URI identifies an XMPP address or specifies the XMPP address to which an XML stanza shall be directed at the end of IRI/URI processing.

For example, the following XMPP IRI/URI identifies a node associated with an XMPP server:

xmpp:example-node@example.com

The following XMPP IRI/URI identifies a node associated with an XMPP server along with a particular XMPP resource identifier associated with that node:

xmpp:example-node@example.com/some-resource

Inclusion of a node is optional in XMPP addresses, so the following XMPP IRI/URI simply identifies an XMPP server:

xmpp:example.com

## 2.5. Query Component

There are many potential use cases for encapsulating information in the query component of an XMPP IRI/URI; examples include but are not limited to:

- sending an XMPP message stanza (see [XMPP-IM]),
- adding a roster item (see [XMPP-IM]),
- sending a presence subscription (see [XMPP-IM]),
- probing for current presence information (see [XMPP-IM]),

- triggering a remote procedure call (see [JEP-0009]),
- discovering the identity or capabilities of another entity (see [JEP-0030]),
- joining an XMPP-based text chat room (see [JEP-0045]),
- interacting with publish-subscribe channels (see [JEP-0060]),
- providing a SOAP interface (see [JEP-0072]), and
- registering with another entity (see [JEP-0077]).

Many of these potential use cases are application specific, and the full range of such applications cannot be foreseen in advance given the continued expansion in XMPP development; however, there is agreement within the Jabber/XMPP developer community that all the uses envisioned to date can be encapsulated via a "query type", optionally supplemented by one or more "key-value" pairs (this is similar to the "application/x-www-form-urlencoded" MIME type described in [HTML]).

As an example, an XMPP IRI/URI intended to launch an interface for sending a message to the XMPP entity "example-node@example.com" might be represented as follows:

```
xmpp:example-node@example.com?message
```

Similarly, an XMPP IRI/URI intended to launch an interface for sending a message to the XMPP entity "example-node@example.com" with a particular subject might be represented as follows:

```
xmpp:example-node@example.com?message;subject=Hello%20World
```

If the processing application does not understand query components or the specified query type, it MUST ignore the query component and treat the IRI/URI as consisting of, for example, <xmpp:example-node@example.com> rather than <xmpp:example-node@example.com?query>. If the processing application does not understand a particular key within the query component, it MUST ignore that key and its associated value.

As noted, there exist many kinds of XMPP applications (both actual and potential), and such applications may define query types and keys for use in the query component portion of XMPP URIs. The Jabber Registrar function (see [JEP-0053]) of the Jabber Software Foundation maintains a registry of such query types and keys at <http://www.jabber.org/registrar/querytypes.html>. To help ensure

interoperability, any application using the formats defined in this document SHOULD submit any associated query types and keys to that registry in accordance with the procedures specified in [JEP-0147].

## 2.6. Fragment Identifier Component

As stated in Section 3.5 of [URI], "The fragment identifier component of a URI allows indirect identification of a secondary resource by reference to a primary resource and additional identifying information." Because the resource identified by an XMPP IRI/URI does not make available any media type (see [MIME]) and therefore (in the terminology of [URI]) no representation exists at an XMPP resource, the semantics of the fragment identifier component in XMPP IRIs/URIs are to be "considered unknown and, effectively, unconstrained" (ibid.). Particular XMPP applications MAY make use of the fragment identifier component for their own purposes. However, if a processing application does not understand fragment identifier components or the syntax of a particular fragment identifier component included in an XMPP IRI/URI, it MUST ignore the fragment identifier component.

## 2.7. Generation of XMPP IRIs/URIs

### 2.7.1. Generation Method

In order to form an XMPP IRI from an XMPP node identifier, domain identifier, and resource identifier, the generating application MUST first ensure that the XMPP address conforms to the rules specified in [XMPP-CORE], including application of the relevant [STRINGPREP]; it MUST then concatenate the following:

1. The "xmpp" scheme and the ":" character

2. Optionally (if an authority component is to be included before the node identifier), the characters "//", an authority component of the form node@domain, and the character "/".

3. Optionally (if the XMPP address contained an XMPP "node identifier"), a string of Unicode characters that conforms to the "inodeid" rule, followed by the "@" character.

4. A string of Unicode characters that conforms to the "ihost" rule.

5. Optionally (if the XMPP address contained an XMPP "resource identifier"), the character "/" and a string of Unicode characters that conforms to the "iresid" rule.

6. Optionally (if a query component is to be included), the "?" character and query component.

7. Optionally (if a fragment identifier component is to be included), the "#" character and fragment identifier component.

In order to form an XMPP URI from the resulting IRI, an application MUST adhere to the mapping procedure specified in Section 3.1 of [IRI].

### 2.7.2. Generation Notes

Certain characters are allowed in the node identifier, domain identifier, and resource identifier portions of a native XMPP address but prohibited by the "inodeid", "ihost", and "iresid" rules of an XMPP IRI. Specifically, the "#" and "?" characters are allowed in node identifiers, and the "/", "?", "#", and "@" characters are allowed in resource identifiers, but these characters are used as delimiters in XMPP IRIs. In addition, the " " ([US-ASCII] space) character is allowed in resource identifiers but prohibited in IRIs. Therefore, all the foregoing characters MUST be percent-encoded when transforming an XMPP address into an XMPP IRI.

Consider the following nasty node in an XMPP address:

```
nasty!#$%()*+,-.;=?[\]^_‘{|}~node@example.com
```

That address would be transformed into the following XMPP IRI:

```
xmpp:nasty!%23$%25()*+,-.;=%3F[\]^_‘{|}~node@example.com
```

Consider the following repulsive resource in an XMPP address (split into two lines for layout purposes):

```
node@example.com
/repulsive !#"$%&’()*+,-./:;<=>?@[\]^_‘{|}~resource
```

That address would be transformed into the following XMPP IRI (split into two lines for layout purposes):

```
xmpp:node@example.com
/repulsive%20!%23"$%25&’()*+,-.%2F:;<=>%3F%40[\]^_‘{|}~resource
```

Furthermore, virtually any character outside the [US-ASCII] range is allowed in an XMPP address and therefore also in an XMPP IRI, but URI syntax forbids such characters directly and specifies that such characters MUST be percent-encoded. In order to determine the URI associated

with an XMPP IRI, an application MUST adhere to the mapping procedure specified in Section 3.1 of [IRI].

### 2.7.3. Generation Example

Consider the following XMPP address:

<jiři@čechy.example/v Praze>

Note: The string "ř" stands for the Unicode character LATIN SMALL LETTER R WITH CARON, and the string "č" stands for the Unicode character LATIN SMALL LETTER C WITH CARON, following the "XML Notation" used in [IRI] to represent characters that cannot be rendered in ASCII-only documents (note also that these characters are represented in their stringprep canonical form). The '<' and '>' characters are not part of the address itself but are provided to set off the address for legibility. For those who do not read Czech, this example could be Anglicized as "george@czech-lands.example/In Prague".

In accordance with the process specified above, the generating application would do the following to generate a valid XMPP IRI from this address:

1. Ensure that the XMPP address conforms to the rules specified in [XMPP-CORE], including application of the relevant [STRINGPREP] profiles and encoding as a [UTF-8] string.

2. Concatenate the following:

    1. The "xmpp" scheme and the ":" character.

    2. An "authority component" if included (not shown in this example).

    3. A string of Unicode characters that represents the XMPP address, transformed in accordance with the "inodeid", "ihost", and "iresid" rules.

    4. The "?" character followed by a "query component", if appropriate to the application (not shown in this example).

    5. The "#" character followed by a "fragment identifier component", if appropriate to the application (not shown in this example).

The result is this XMPP IRI:

<xmpp:jiři@čechy.example/v%20Praze>

In order to generate a valid XMPP URI from the foregoing IRI, the application MUST adhere to the procedure specified in Section 3.1 of [IRI], resulting in the following URI:

<xmpp:ji%C5%99i@%C4%8Dechy.example/v%20Praze>

## 2.8. Processing of XMPP IRIs/URIs

### 2.8.1. Processing Method

If a processing application is presented with an XMPP URI and not with an XMPP IRI, it MUST first convert the URI into an IRI by following the procedure specified in Section 3.2 of [IRI].

In order to decompose an XMPP IRI for interaction with the entity it identifies, a processing application MUST separate:

1. The "xmpp" scheme and the ":" character.

2. The authority component, if included (the string of Unicode characters between the "//" characters and the next "/" character, the "?" character, the "#" character, or the end of the IRI).

3. A string of Unicode characters that represents an XMPP address as transformed in accordance with the "inodeid", "ihost", and "iresid" rules.

4. Optionally the query component, if included, using the "?" character as a separator.

5. Optionally the fragment identifier component, if included, using the "#" character as a separator.

At this point, the processing application MUST ensure that the resulting XMPP address conforms to the rules specified in [XMPP-CORE], including application of the relevant [STRINGPREP]. The processing application then would either (1) complete further XMPP handling itself or (2) invoke a helper application to complete XMPP handling; such XMPP handling would most likely consist of the following steps:

1. If not already connected to an XMPP server, connect either as the user specified in the authority component or as the configured user at the configured XMPP server, normally by adhering to the XMPP connection procedures defined in [XMPP-CORE]. (Note: The processing application SHOULD ignore the authority component if it has been configured with a set of default credentials.)

2. Optionally, determine the nature of the intended recipient (e.g., via [JEP-0030]).

3. Optionally, present an appropriate interface to a user based on the nature of the intended recipient and/or the contents of the query component.

4. Generate an XMPP stanza that translates any user or application inputs into their corresponding XMPP equivalents.

5. Send the XMPP stanza via the authenticated server connection for delivery to the intended recipient.

### 2.8.2. Processing Notes

It may help implementors to note that the first two steps of "further XMPP handling", as described at the end of Section 2.8.1, are similar to HTTP authentication ([HTTP-AUTH]), while the next three steps are similar to the handling of mailto: URIs ([MAILTO]).

As noted in Section 2.7.2 of this document, certain characters are allowed in the node identifier, domain identifier, and resource identifier portions of a native XMPP address but prohibited by the "inodeid", "ihost", and "iresid" rules of an XMPP IRI. The percent-encoded octets corresponding to these characters in XMPP IRIs MUST be transformed into the characters allowed in XMPP addresses when processing an XMPP IRI for interaction with the represented XMPP entity.

Consider the following nasty node in an XMPP IRI:

```
xmpp:nasty!%23$%()*+,-.;=%3F[\]^_‘{|}~node@example.com
```

That IRI would be transformed into the following XMPP address:

```
nasty!#$%()*+,-.;=?[\]^_‘{|}~node@example.com
```

Consider the following repulsive resource in an XMPP IRI (split into two lines for layout purposes):

```
xmpp:node@example.com
/repulsive%20!%23"$%25&'()*+,-.%2F:;<=>%3F%40[\]^_‘{|}~resource
```

That IRI would be transformed into the following XMPP address (split into two lines for layout purposes):

```
node@example.com
/repulsive !#"$%&'()*+,-./:;<=>?@[\]^_‘{|}~resource
```

### 2.8.3. Processing Example

Consider the XMPP URI that resulted from the previous example:

```
<xmpp:ji%C5%99i@%C4%8Dechy.example/v%20Praze>
```

In order to generate a valid XMPP IRI from that URI, the application MUST adhere to the procedure specified in Section 3.2 of [IRI], resulting in the following IRI:

```
<xmpp:ji&#x159;i@&#x10D;echy.example/v%20Praze>
```

In accordance with the process specified above, the processing application would remove the "xmpp" scheme and ":" character to extract the XMPP address from this XMPP IRI, converting any percent-encoded octets from the "inodeid", "ihost", and "iresid" rules into their character equivalents (e.g., "%20" into the space character).

The result is this XMPP address:

```
<ji&#x159;i@&#x10D;echy.example/v Praze>
```

## 2.9. Internationalization

Because XMPP addresses are [UTF-8] strings and because octets outside the [US-ASCII] range within XMPP addresses can be easily converted to percent-encoded octets, XMPP addresses are designed to work well with Internationalized Resource Identifiers ([IRI]). In particular, with the exceptions of stringprep verification, the conversion of syntax-relevant [US-ASCII] characters (e.g., "?"), and the conversion of percent-encoded octets from the "inodeid", "ihost", and "iresid" rules into their character equivalents (e.g., "%20" into the [US-ASCII] space character), an XMPP IRI can be constructed directly by prepending the "xmpp" scheme and ":" character to an XMPP address. Furthermore, an XMPP IRI can be converted into URI syntax by adhering

to the procedure specified in Section 3.1 of [IRI], and an XMPP URI can be converted into IRI syntax by adhering to the procedure specified in Section 3.2 of [IRI], thus ensuring interoperability with applications that are able to process URIs but unable to process IRIs.

## 3. IANA Registration of xmpp URI Scheme

In accordance with [URI-SCHEMES], this section provides the information required to register the xmpp URI scheme.

### 3.1. URI Scheme Name

xmpp

### 3.2. Status

permanent

### 3.3. URI Scheme Syntax

The syntax for an xmpp URI is defined below using Augmented Backus-Naur Form as specified by [ABNF], where the "fragment", "host", "pct-encoded", and "unreserved" rules are defined in [URI] and DQUOTE is defined in [ABNF]:

```
  xmppuri   = "xmpp" ":" hierxmpp [ "?" querycomp ] [ "#" fragment ]
  hierxmpp  = authpath / pathxmpp
  authpath  = "//" authxmpp [ "/" pathxmpp ]
  authxmpp  = nodeid "@" host
  pathxmpp  = [ nodeid "@" ] host [ "/" resid ]
  nodeid    = *( unreserved / pct-encoded / nodeallow )
  nodeallow = "!" / "$" / "(" / ")" / "*" / "+" / "," / ";" /
              "=" / "[" / "\" / "]" / "^" / "`" / "{" / "|" /
              "}"
  resid     = *( unreserved / pct-encoded / resallow )
  resallow   = "!" / DQUOTE / "$" / "&" / "'" / "(" / ")" /
               "*" / "+" / "," / ":" / ";" / "<" / "=" / ">" /
               "[" / "\" / "]" / "^" / "`" / "{" / "|" / "}"
  querycomp = querytype [ *pair ]
  querytype = *( unreserved / pct-encoded )
  pair      = ";" key "=" value
  key       = *( unreserved / pct-encoded )
  value     = *( unreserved / pct-encoded )
```

### 3.4. URI Scheme Semantics

The xmpp URI scheme identifies entities that natively communicate using the Extensible Messaging and Presence Protocol (XMPP), and is mainly used for identification rather than for resource location. However, if an application that processes an xmpp URI enables interaction with the XMPP address identified by the URI, it MUST follow the methodology defined in Section 2 of RFC 4622, Use of XMPP IRIs and URIs, to reconstruct the encapsulated XMPP address, connect to an appropriate XMPP server, and send an appropriate XMPP "stanza" (XML fragment) to the XMPP address. (Note: There is no MIME type associated with the xmpp URI scheme.)

### 3.5. Encoding Considerations

In addition to XMPP URIs, there will also be XMPP Internationalized Resource Identifiers (IRIs). Prior to converting an Extensible Messaging and Presence Protocol (XMPP) address into an IRI (and in accordance with [XMPP-CORE]), the XMPP address must be represented as [UTF-8] by the generating application (e.g., by transforming an application's internal representation of the address as a UTF-16 string into a UTF-8 string), and the UTF-8 string must then be prepended with the "xmpp" scheme and ":" character. However, because an XMPP URI must contain only [US-ASCII] characters, the UTF-8 string of an XMPP IRI must be transformed into URI syntax by adhering to the procedure specified in RFC 3987.

### 3.6. Applications/protocols That Use This URI Scheme Name

The xmpp URI scheme is intended to be used by interfaces to an XMPP network from non-native user agents, such as web browsers, as well as by non-native applications that need to identify XMPP entities as full URIs or IRIs.

### 3.7. Interoperability Considerations

There are no known interoperability concerns related to use of the xmpp URI scheme. In order to help ensure interoperability, the Jabber Registrar function of the Jabber Software Foundation maintains a registry of query types and keys that can be used in the query components of XMPP URIs and IRIs, located at <http://www.jabber.org/registrar/querytypes.html>.

### 3.8. Security Considerations

See Section 5 of RFC 4622, Security Considerations.

### 3.9. Contact

Peter Saint-Andre [mailto:stpeter@jabber.org, xmpp:stpeter@jabber.org]

### 3.10. Author/Change Controller

This scheme is registered under the IETF tree. As such, the IETF maintains change control.

### 3.11. References

[XMPP-CORE]

## 4. IANA Considerations

This document registers a URI scheme. The registration template can be found in Section 3 of this document. In order to help ensure interoperability, the Jabber Registrar function of the Jabber Software Foundation maintains a registry of query types and keys that can be used in the query components of XMPP URIs and IRIs, located at <http://www.jabber.org/registrar/querytypes.html>.

## 5. Security Considerations

Providing an interface to XMPP services from non-native applications introduces new security concerns. The security considerations discussed in [IRI], [URI], and [XMPP-CORE] apply to XMPP IRIs, and the security considerations discussed in [URI] and [XMPP-CORE] apply to XMPP URIs. In accordance with Section 2.7 of [URI-SCHEMES] and Section 7 of [URI], particular security considerations are specified in the following sections.

### 5.1. Reliability and Consistency

Given that XMPP addresses of the form node@domain.tld are typically created via registration at an XMPP server or provisioned by an administrator of such a server, it is possible that such addresses may also be unregistered or deprovisioned. Therefore, the XMPP IRI/URI that identifies such an XMPP address may not be reliably and consistently associated with the same principal, account owner, application, or device.

XMPP addresses of the form node@domain.tld/resource are typically even more ephemeral (since a given XMPP resource identifier is typically associated with a particular, temporary session of an XMPP client at an XMPP server); therefore the XMPP IRI/URI that identifies such an XMPP address probably will not reliably and consistently be

associated with the same session. However, the procedures specified in Section 10 of [XMPP-CORE] effectively eliminate any potential confusion that might be introduced by the lack of reliability and consistency for the XMPP IRI/URI that identifies such an XMPP address.

XMPP addresses of the form domain.tld are typically long-lived XMPP servers or associated services; although naturally it is possible for server or service administrators to de-commission the server or service at any time, typically the IRIs/URIs that identify such servers or services are the most reliable and consistent of XMPP IRIs/URIs.

XMPP addresses of the form domain.tld/resource are not yet common on XMPP networks; however, the reliability and consistency of XMPP IRIs/URIs that identify such XMPP addresses would likely fall somewhere between those that identify XMPP addresses of the form domain.tld and those that identify XMPP addresses of the form node@domain.tld.

### 5.2. Malicious Construction

Malicious construction of XMPP IRIs/URIs is made less likely by the prohibition on port numbers in XMPP IRIs/URIs (since port numbers are to be discovered using [DNS-SRV] records, as specified in [XMPP-CORE]).

### 5.3. Back-End Transcoding

Because the base XMPP protocol is designed to implement the exchange of messages and presence information and not the retrieval of files or invocation of similar system functions, it is deemed unlikely that the use of XMPP IRIs/URIs would result in harmful dereferencing. However, if an XMPP protocol extension defines methods for information retrieval, it MUST define appropriate controls over access to that information. In addition, XMPP servers SHOULD NOT natively parse XMPP IRIs/URIs but instead SHOULD accept only the XML wire protocol specified in [XMPP-CORE] and any desired extensions thereto.

### 5.4. Sensitive Information

The ability to interact with XMPP entities via a web browser or other non-native application may expose sensitive information (such as support for particular XMPP application protocol extensions) and thereby make it possible to launch attacks that are not possible or that are unlikely on a native XMPP network. Due care must be taken

in deciding what information is appropriate for representation in XMPP IRIs or URIs.

In particular, advertising XMPP IRIs/URIs in publicly accessible locations (e.g., on websites) may make it easier for malicious users to harvest XMPP addresses from the authority and path components of XMPP IRIs/URIs and therefore to send unsolicited bulk communications to the users or applications represented by those addresses. Due care should be taken in balancing the benefits of open information exchange against the potential costs of unwanted communications.

To help prevent leaking of sensitive information, passwords and other user credentials are forbidden in the authority component of XMPP IRIs/URIs; in fact they are not needed, since the fact that authentication in XMPP occurs via [SASL] makes it possible to use the SASL ANONYMOUS mechanism, if desired.

### 5.5. Semantic Attacks

Despite the existence of non-hierarchical URI schemes such as [MAILTO], by association human users may expect all URIs to include the "//" characters after the scheme name and ":" character. However, in XMPP IRIs/URIs, the "//" characters precede the authority component rather than the path component. Thus, xmpp://guest@example.com indicates to authenticate as "guest@example.com", whereas xmpp:guest@example.com identifies the node "guest@example.com". Processing applications MUST clearly differentiate between these forms, and user agents SHOULD discourage human users from including the "//" characters in XMPP IRIs/URIs since use of the authority component is envisioned to be helpful only in specialized scenarios, not more generally.

### 5.6. Spoofing

The ability to include effectively the full range of Unicode characters in an XMPP IRI may make it easier to execute certain forms of address mimicking (also called "spoofing"). However, XMPP IRIs are no different from other IRIs in this regard, and applications that will present XMPP IRIs to human users must adhere to best practices regarding address mimicking in order to help prevent attacks that result from spoofed addresses (e.g., the phenomenon known as "phishing"). For details, refer to the Security Considerations of [IRI].

# 6. References

## 6.1. Normative References

[ABNF] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[IRI] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

[TERMS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[URI] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[XMPP-CORE] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, October 2004.

## 6.2. Informative References

[CPIM] Peterson, J., "Common Profile for Instant Messaging (CPIM)", RFC 3860, August 2004.

[CPP] Peterson, J., "Common Profile for Presence (CPP)", RFC 3859, August 2004.

[DNS-SRV] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[HTML] Raggett, D., "HTML 4.0 Specification", W3C REC REC-html40-19980424, April 1998.

[HTTP-AUTH] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[IDNA] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[JEP-0009] Adams, D., "Jabber-RPC", JSF JEP 0009, February 2006.

[JEP-0030] Hildebrand, J., Millard, P., Eatmon, R., and P. Saint-Andre, "Service Discovery", JSF JEP 0030, January 2006.

[JEP-0045] Saint-Andre, P., "Multi-User Chat", JSF JEP 0045, September 2005.

[JEP-0053] Saint-Andre, P., "Jabber Registrar", JSF JEP 0053, May 2004.

[JEP-0060] Millard, P., Saint-Andre, P., and R. Meijer, "Publish-Subscribe", JSF JEP 0060, June 2005.

[JEP-0072] Forno, F. and P. Saint-Andre, "SOAP Over XMPP", JSF JEP 0072, December 2005.

[JEP-0077] Saint-Andre, P., "In-Band Registration", JSF JEP 0077, January 2006.

[JEP-0147] Saint-Andre, P., "XMPP IRI/URI Query Components", JSF JEP 0147, March 2006.

[MAILTO] Hoffman, P., Masinter, L., and J. Zawinski, "The mailto URL scheme", RFC 2368, July 1998.

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[SASL] Melnikov, A. and K. Zeilenga, "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[STRINGPREP] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("STRINGPREP")", RFC 3454, December 2002.

[UNICODE] The Unicode Consortium, "The Unicode Standard, Version 3.2.0", 2000.

The Unicode Standard, Version 3.2.0 is defined by The Unicode Standard, Version 3.0 (Reading, MA, Addison-Wesley, 2000. ISBN 0-201-61633-5), as amended by the Unicode Standard Annex #27: Unicode 3.1 (http://www.unicode.org/reports/tr27/) and by the Unicode Standard Annex #28: Unicode 3.2 (http://www.unicode.org/reports/tr28/).

[URI-SCHEMES] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", RFC 4395, February 2006.

[US-ASCII] American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[UTF-8] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[XMPP-IM] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 3921, October 2004.

## Author's Address

Peter Saint-Andre
Jabber Software Foundation

EMail: stpeter@jabber.org
URI: xmpp:stpeter@jabber.org

## Full Copyright Statement



## Intellectual Property



## Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).