Network Working Group
Internet-Draft
Expires: May 2, 2001

D. Crocker
Brandenburg Consulting
A. Diacakis
F. Mazzoldi
Network Projects Inc.
C. Huitema
Microsoft Corporation
G. Klyne
Content Technologies
M. Rose
Invisible Worlds
J. Rosenberg
R. Sparks
dynamicsoft
H. Sugano
Fujitsu Laboratories Ltd.

November 2000

A Common Profile for Instant Messaging (CPIM)
draft-ietf-impp-cpim-01

Status of this Memo

Copyright Notice

Abstract

Semantics and data formats for common services of Instant Messaging and online Presence, independent of underlying transport infrastructure, are described. The CPIM profile meets the requirements specified in RFC 2779 using a minimalist approach allowing interoperation of a wide range of IM and Presence systems.

Table of Contents

1. Introduction

To achieve interoperation of IM systems that are compliant with RFC 2779[8], there must be a common agreement on both Instant Messaging and Presence services. This memo defines such an agreement according to the philosophy that there must be no loss of information between IM systems that are minimally conformant to RFC2779.

This memo focuses on interoperation. Accordingly only those aspects of IM that require interoperation are discussed. For example, the "open instant inbox" operation is not applicable as this operation occurs within a single IM system and not across systems.

Service behavior is described abstractly in terms of operations invoked between the consumer and provider of a service. Accordingly, each IM service must specify how this behavior is mapped onto its own protocol interactions. The choice of strategy is a local matter, providing that there is a clear relation between the abstract behavior of the service (as specified in this memo) and how it is faithfully realized by a particular IM service.

The parameters for each operation are defined using an abstract syntax. Although the syntax specifies the range of possible data values, each IM service must specify how well-formed instances of the abstract representation are encoded as a concrete series of bits.

For example, one strategy might transmit presence information as key/value pairs, another might use a compact binary representation, and a third might use nested containers. The choice of strategy is a local matter, providing that there is a clear relation between the abstract syntax (as specified in this memo) and how it is faithfully encoded by an particular IM service.

1.1 Terminology

This memos makes use of the vocabulary defined in RFC 2778[7]. Terms such as as CLOSED, INSTANT INBOX, INSTANT MESSAGE, OPEN, PRESENCE SERVICE, PRESENTITY, SUBSCRIPTION, and WATCHER are used in the same meaning as defined therein.

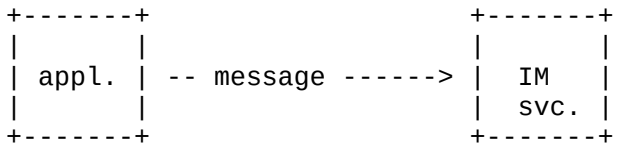
1.2 A Note on The Examples

In the examples which follow, this memo uses time-sequence diagrams annotated with XML fragments to illustrate operations and their parameters. The use of XML is an artifact of this memo’s presentation style and does not imply any requirement for the use of XML in an IM system.

2. Abstract Messaging Service

2.1 Overview of the Messaging Service

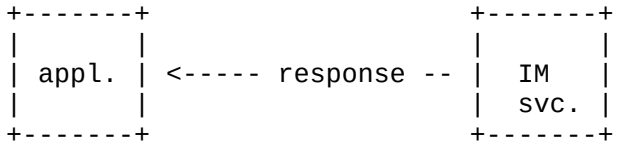
When an application wants to send a message to an INSTANT INBOX, it invokes the message operation, e.g.,




```
<message source='im:fred@example.com'
         destination='im:barney@example.com'
         transID='1' />
    ...
Content-Type: text/plain; charset="us-ascii"

Yabba, dabba, doo!
```

The service immediately responds by invoking the response operation containing the same transaction-identifier, e.g.,




```
<response status='success' transID='1' />
```

2.2 Identification of INSTANT INBOXes

An INSTANT INBOX is specified using the IM URI (Section 5.1)f RFC 822[1] (i.e., "local@domain") is used, where the local-part MUST be interpreted and assigned semantics only by the system specified in the domain part of the identifier. Representation of non-ASCII character sets in local-part strings is limited to the standard methods provided as extensions to RFC 822[1]

2.2.1 Address Resolution

A client determines the address of an appropriate system running a server by resolving the destination domain name that is part of the identifier to either an intermediate relay system or a final target system.

Only resolvable, fully-qualified, domain names (FQDNs) are permitted when domain names are used in the messaging service (i.e., domain names that can be resolved to SRV[9] or A RRs).

2.2.1.1 Domain Name Lookup

A client lexically identifies a domain to which instant messages will be delivered for processing, a DNS lookup MUST be performed to resolve the domain[2]. The names MUST be fully-qualified domain names (FQDNs) Ãù mechanisms for inferring FQDNs from partial names or local aliases are a local matter.

The lookup first attempts to locate SRV RRs associated with the domain. If a CNAME RR is found instead, the resulting domain is processed as if it were the initial domain.

If one or more SRV RRs are found for a given domain, a sender MUST NOT utilize any A RRs associated with that domain unless they are located using the SRV RRs; otherwise, if no SRV RRs are found, but an A RR is found, then the A RR is treated as if it was associated with an implicit SRV RR, with a preference of 0, pointing to that host.

2.2.1.2 Processing SRV RRs

To process an IM URI, a lookup is performed for SRVs for the target domain and a desired IM transport protocol.

For example, if the destination INSTANT INBOX is "im:fred@example.com", and the sender wishes to use an IM transport protocol called "SIP", then a SRV lookup is performed for:

    _im._sip.example.com.

The returned RRs, if any, specify the next-hop server.

The choice of IM transport protocol is a local configuration option for each system.

Using this mechanism, seamless routing of IM traffic is possible, regardless of whether a gateway is necessary for interoperation. To achieve this transparency, a seperate RR for a gateway must be present for each transport protocol and domain pair that it serves.

2.2.1.3 Processing Multiple Addresses

When the lookup succeeds, the mapping can result in a list of alternative delivery addresses rather than a single address, because of multiple SRV records, multihoming, or both. For reliable operations, the client MUST be able to try each of the relevant addresses in this list in order, until a delivery attempt succeeds. However, there MAY also be a configurable limit on the number of alternate addresses that can be tried. In any case, the client SHOULD try at least two addresses. Two types of information are used to rank the host addresses: multiple SRV records, and multihomed hosts.

Multiple SRV records contain a preference indication that MUST be used in sorting. Lower numbers are preferrable to higher ones. If there are multiple destinations with the same preference, and there is no clear reason to favor one (e.g., by recognition of an easily-reached address), then the sender MUST randomize them to spread the load across multiple servers for a specific destination.

The destination host (perhaps taken from the preferred SRV record) may be multihomed, in which case the resolver will return a list of alternative IP addresses. It is the responsibility of the resolver to have ordered this list by decreasing preference if necessary, and the sender MUST try them in the order presented.

2.3 Format of Instant Messages

An INSTANT MESSAGE comprises a MIME Multipart/Related, Type=message/RFC822+XML object, as defined in XML/MIME[5]. Representation of non-ASCII character sets in MIME is a standard feature of MIME.

Note that the IETF provides numerous technologies that allow end-users to exchange authenticated and private messages formatted as MIME objects, c.f., PGP-MIME[4] and S/MIME[6].

2.4 The Messaging Service

Section 6 and Section 7 define the abstract syntax of the operations invoked with the service.

Note that the transaction-identifier parameters used with the service are potentially long-lived. Accordingly, the values of transaction-identifiers should appear to be unpredictable.

2.4.1 The Message Operation

When an application wants to send an INSTANT MESSAGE, it invokes the message operation.

The message operation has these parameters:

- the source parameter specifies the INSTANT INBOX on whose behalf this message is sent (using an IM URI);
- the destination parameter specifies the INSTANT INBOX that the message should be delivered to (using an IM URI);
- the transID parameter specifies the transaction-identifier associated with this operation; and,
- the message to be sent.

When the service is informed of the message operation, it performs these steps:

1. If the source or destination does not refer to a valid INSTANT INBOX, a response operation having status "failure" is invoked.

2. If access control does not permit the application to request this operation, a response operation having status "failure" is invoked.

3. Otherwise:

    1. If the service is able to successfully deliver the message, a response operation having status "success" is invoked.

    2. If the service is unable to successfully deliver the message, a response operation having status "failure" is invoked.

    3. If the service must delegate responsibility for delivery, and if the delegation will not result in a future authoritative indication to the service, a response operation having status "indeterminant" is invoked.

    4. If the service must delegate responsibility for delivery, and if the delegation will result in a future authoritative indication to the service, then a response operation is invoked immediately after the indication is received.

When the service invokes the response operation, the transID parameter is identical to the value found in the message operation invoked by the application.

2.4.2 Looping

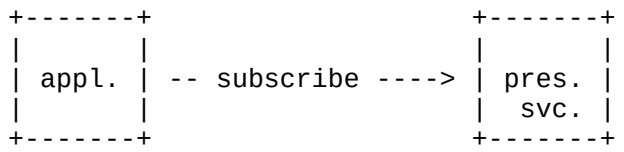
The dynamic routing of instant messages can result in looping of a message through a relay. Detection of loops is not always obvious, since aliasing and group list expansions can legitimately cause a message to pass through a relay more than one time.

[[[In Internet Mail, counting the number of Received headers is the accepted technique for guessing that looping is occurring. Use of this technique will require Instant Messaging to support Received headers. /editor]]]

3. Abstract Presence Service

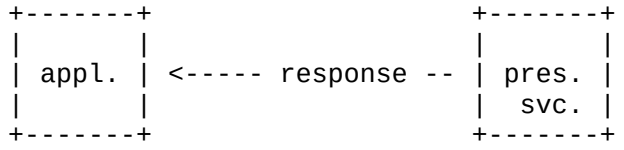
3.1 Overview of the Presence Service

When an application wants to (periodically) receive the presence information associated with a PRESENTITY, it invokes the subscribe operation, e.g.,




```
<subscribe watcher='pres:wilma@example.com'
           target='pres:fred@example.com'
           duration='86400' transID='2' />
```

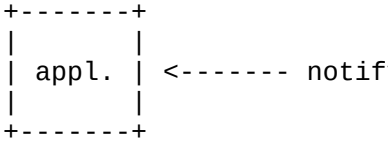
The service immediately responds by invoking the response operation containing the same transaction-identifier, e.g.,




```
<response status='success' transID='2' duration='3600' />
```

A WATCHER may have at most one subscription for a PRESENTITY.

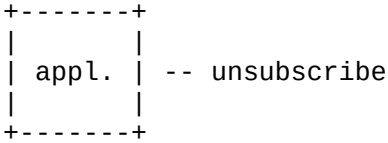
If the response operation indicates success, the service immediate invokes the notify operation to communicate the presence information to the WATCHER, e.g.,




```
<notify watcher='pres:wilma@example.com'
        target='pres:fred@example.com'
        transID='1234'>
    <presence entityInfo='http://www.example.com/fred/'>
        <tuple destination='im:fred@example.com'
               status='open' />
    </presence>
</notify>
```

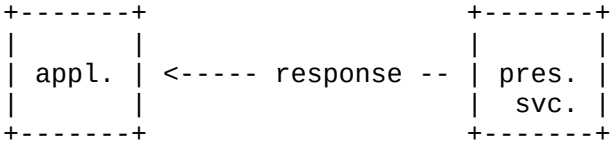
If the duration parameter is non-zero, then for up to the specified duration, the service invokes the notify operation whenever there are any changes to the PRESENTITY's presence information. Otherwise, exactly one notify operation is invoked, achieving a one time poll of the presence information. Regardless, there is no application response to the notify operation (i.e., the application does not invoke a response operation when a notify operation occurs).

The application may prematurely cancel a subscription by invoking the unsubscribe operation, e.g.,




```
<unsubscribe watcher='pres:wilma@example.com'
             target='pres:fred@example.com'
             transID='3' />
```

The service immediately responds by invoking the response operation containing the same transaction-identifier, e.g.,




```
<response status='success' transID='3' />
```

3.2 Identification of PRESENTITIES

A PRESENTITY is specified using the PRES URI (Section 5.2) scheme. Briefly, the "addr-spec" syntax of RFC 822[1] (i.e., "local@domain") is used, where the local-part MUST be interpreted and assigned semantics only by the host specified in the domain part of the identifier. Representation of non-ASCII character sets in local-part strings is limited to the standard methods provided as extensions to RFC 822[1]

To resolve identifiers associated with the Presence service, the mechanism defined in Section 2.2.1 is used, except that the processing of a PRES URI is performed by looking up SRV RRs for a desired presence transport protocol.

For example, if the destination PRESENTITY is "pres:fred@example.com", and the sender wishes to use a presence transport protocol called "PEPP", then a SRV lookup is performed for:

    _pres._pepp.example.com.

3.3 Format of Presence Information

Section 9 defines the syntax for presence information using an XML DTD.

Each PRESENTITY's presence information contains an "entityInfo" attribute, and contains one or more "tuple" elements:

- the "entityInfo" attribute specifies arbitrary information about the PRESENTITY (using a URI); and,
- each "tuple" element specifies information associated with the PRESENTITY.

Each "tuple" element has a "destination" attribute, a "status" attribute, and contains arbitrary content:

- the "destination" attribute specifies a URI;
- the "status" attribute is either OPEN or CLOSED; and,
- the content of the "tuple" element contains arbitrary information about the tuple.

3.4 The Presence Service

Section 6 and Section 8 define the abstract syntax of the operations invoked with the service.

An implementation of the service must maintain information about both presence information and in-progress operations in persistent storage.

Note that the transaction-identifier parameter used with the service is potentially long-lived. Accordingly, the values generated for this parameter should appear to be unpredictable.

3.4.1 The Subscribe Operation

When an application wants to (periodically) receive the presence information associated with an PRESENTITY, it invokes the subscribe operation.

The subscribe operation has these parameters:

- the watcher parameter specifies the WATCHER associated with the subscription;
- the target parameter specifies the PRESENTITY associated with the presence information;
- the duration parameter specifies the maximum number of seconds that the SUBSCRIPTION should be active; and,
- the transID parameter specifies the transaction-identifier associated with this operation.

When the service is informed of the subscribe operation, it performs these steps:

1. If the watcher or target parameter does not refer to a valid PRESENTITY, a response operation having status "failure" is invoked.

2. If access control does not permit the application to request this operation, a response operation having status "failure" is invoked.

3. If the duration parameter is non-zero, and if the watcher and target parameters refer to an in-progress subscribe operation for the application, a response operation having status "failure" is invoked.

4. Otherwise:

    1. A response operation having status "success" is immediately invoked. (If the service chooses a different duration for the subscription then it conveys this information in the response operation.)

    2. A notify operation, corresponding to the target’s presence information, is immediately invoked for the watcher.

    3. For up to the amount of time indicated by the duration parameter, if the target’s presence information changes, and if access control allows, a notify operation is invoked for the watcher.

    Note that if the duration parameter is zero-valued, then the subscribe operation is making a one-time poll of the presence information. Accordingly, Step 4.3 above does not occur.

When the service invokes a response operation as a result of this processing, the transID parameter is identical to the value found in the subscribe operation invoked by the application.

3.4.2 The Notify Operation

The service invokes the notify operation whenever the presence information associated with a PRESENTITY changes and there are subscribers to that information.

The notify operation has these parameters:

- the watcher parameter specifies the WATCHER associated with the subscription;
- the target parameter specifies the PRESENTITY associated with the presence information;
- the transID parameter specifies the transaction-identifier associated with this operation; and,
- the presence information for the PRESENTITY.

There is no application response to the notify operation.

3.4.3 The Unsubscribe Operation

When an application wants to terminate a subscription, it invokes the unsubscribe operation.

The unsubscribe operations has these parameters:

- the watcher parameter specifies the WATCHER associated with the subscription;
- the target parameter specifies the PRESENTITY associated with the presence information; and,
- the transID parameter specifies the transaction-identifier associated with this operation.

When the service is informed of the unsubscribe operation, it performs these steps:

1. If the watcher and target parameters do not refer to an in-progress subscribe operation for the application, a response operation having status "failure" is invoked.
2. Otherwise, the in-progress subscribe operation for the application is terminated, and a response operation having status "success" is invoked by the service.

Note that following a successful unsubscribe operation, the WATCHER may receive further notifications. Although the service will no longer invoke the notify operation after successfully processing a unsubscribe operation, earlier notify operations may still be in progress.

4. Security Considerations

This memo makes no specific requirements on security procedures for interoperation between IM systems. Accordingly, trust between interconnected IM systems is determined in a bilateral matter.

However this memo does require that each IM system control access to its Instant Messaging and Presence services. Consult both RFC 2778 and RFC2779 for a discussion of security considerations for for IM systems.

4.1 Threats

Attacks, of concern for instant messaging, include access, deletion, insertion, reordering and modification of messages by unauthorized principals. Replay is a combination of a subset of these attacks.

These attacks can take place in the communication links between sending client and its server, between two servers, between the receiving client and its server, or by attacking any of the hosts involved. This document, not being concerned with client-server interchanges, only addresses threats aimed at server-server communication.

Countermeasures against unauthorized access are encrypted communication and encrypted messages.

Countermeasures against insertion of false messages are authentication and authorization of sending servers and strongly signed messages.

Countermeasures against reordered messages are date-stamped or serial-numbered messages, coupled with digital signatures that include the date or serial number, if modification is not otherwise guarded against.

Countermeasures against replayed messages are date stamps and unique message IDs, coupled with digital signatures that include the date or serial number, if modification is not otherwise guarded against.

Countermeasures against deletion of messages are integrity-protected connections between servers where the server’s identity is verified. Serial-numbered messages can also be useful in detecting deleted messages.

Attacks that target the server hosts rather than the communication channels can successfully defeat all countermeasures that depend on host security. Digital signatures and encrypted messages do not

depend on host security, for intermediate systems, but cannot by themselves guard against deletion or reordering of messages.

For presence, the attacks include giving presence information to unauthorized watchers, not reporting watcher information back to a presentity, and insertion, modification, deletion and replay of presence update messages. The same set of countermeasures are relevant.

Instant messaging and presence systems can provide security at two levels: hop-by-hop and/or end-to-end.

4.2 Hop-by-hop security

A useful but imperfect level of security can be provided on a hop-by-hop basis, with all aspects of the communication including message content and originator verification, using transport level security between servers. The main drawback of this approach is that it requires that each server that handles message or presence information must be trusted. But it is relatively easy to deploy, because it depends only on bilateral arrangements between directly communicating servers.

The underlying principles for using hop-by-hop security are:

(a) each server and/or domain must keep their own house in order, ensuring that operations and information accesses are allowed only to appropriately authorized parties, and

(b) each server and/or domain must make its own choices about the levels of trust to be established to any other server and/or domain with which they directly communicate. [[[Some debate about the degree of trust necessary between servers. /dc]]]

When passing IM and presence information between services using different protocols, a gateway system MUST be capable of using security mechanisms appropriate to each of the protocols concerned, and must have access to keys needed to authenticate any other system with which it needs to directly communicate in a secure fashion.

[[[SUGGESTION: to allow the use of common keys across different protocols, we might say that hop-by-hop security should be based on SASL, and specify specific profiles that should be used. This doesn't buy anything at the protocol level, but it might make it easier to leverage some common key-distribution infrastructure, and avoid having to distribute different keys for communicating with a gateway using different protocols.]]]

4.3 End-to-end security

End-to-end security is widely regarded as being more satisfactory than hop-by-hop security, as the need to trust intermediate parties is reduced. However, some aspects of end-to-end security are difficult to achieve because they need bilateral arrangement between any pair of communicating parties about acceptable security standards to use, and key exchange. Reliance on bilateral agreements does not scale well. A moderating alernative is a third-party certification service and this approach, so far, has not found large-scale use.

The two IETF standards for end-to-end MIME object security are OpenPGP[7] and S/MIME[8]. They require a public key operation for each message. For repeated, short transactions, this overhead can be onerous. A version of these specifications which permited re-use of the public key across multiple messages would greatly reduce instant messaging overhead.

4.3.1 Instant messages

End to end security for instant messages can be provided using any of the MIME-based security mechanisms (S/MIME [8], OpenPGP [7]), as instant message payload content is not interpreted or reformatted in transit.

[[[NOTE: may need to say something about allowable MIME C-T-Es?]]]

This specification allows any pair of communicating parties to use any MIME-based security framework for instant messages (c.f. section 2.3), but mechanisms for establishing the required bilateral arrangements and key exchange are not specified here.

4.3.2 Presence service

The situation regarding end-to-end security for presence services is unclear, as there is no common encapsulation framework specified for presence, and the presence data itself is not invariant across different IM services.

[[[NOTE: this raises a case for fixing the presence information to a specific format if end-to-end security capability is to be a requirement.]]]

5. IANA Considerations

The IANA assigns the "im" and "pres" URL schemes.

5.1 The IM URI Scheme

The Instant Messaging (IM) URI scheme designates an Internet resource, namely an INSTANT INBOX.

The syntax of an IM URL has the form:

"im:" addr-spec

where "addr-spec" is defined in RFC 822.

5.2 The PRES URI Scheme

The Presence (PRES) URI scheme designates an Internet resource, namely a PRESENTITY or WATCHER.

The syntax of a PRES URL has the form:

"pres:" addr-spec

where "addr-spec" is defined in RFC 822.

6. The Common Service DTD

```
   <!--
     DTD for the IM common profile, as of 2000-08-16


     Refer to this DTD as:

       <!ENTITY % IMCOMMON PUBLIC "-//Blocks//DTD IM COMMON//EN"
                  "http://xml.resource.org/syntaxes/IM/im-common.dtd">
       %IMCOMMON;
     -->
   <!--
     DTD data types:

          entity        syntax/reference     example
          ======        ================     =======
       a language tag
           LANG         c.f., [RFC-1766]      "en", "en-US", etc.

       seconds
           SECONDS      0..2147483647        600

       unique-identifier
           UNIQID       1..2147483647        42

       authoritative identity
           URI          c.f., [RFC-2396]      http://invisible.net/
     -->
   <!ENTITY % LANG "NMTOKEN">
   <!ENTITY % SECONDS "CDATA">
   <!ENTITY % UNIQID "CDATA">
   <!ENTITY % URI "CDATA">
   <!--
     Abstract syntax for the response operation
     -->
   <!ELEMENT response (#PCDATA)>
   <!ATTLIST response
      status (success | failure | indeterminant) #REQUIRED
      transID %UNIQID; #REQUIRED
      duration %SECONDS; #IMPLIED
      xml:lang %LANG; #IMPLIED
   >
```

7. The Messaging Service DTD

```
<!--
  DTD for the abstract IM messaging service, as of 2000-08-16


  Refer to this DTD as:

    <!ENTITY % IMMESSAGING PUBLIC "-//Blocks//DTD IM MESSAGING//EN"
             "http://xml.resource.org/syntaxes/IM/im-messaging.dtd">
    %IMMESSAGING;
  -->
<!ENTITY % IMCOMMON PUBLIC "-//Blocks//DTD IM COMMON//EN"
           "http://xml.resource.org/syntaxes/IM/im-common.dtd">
%IMCOMMON;
<!--
  DTD data types:

       entity        syntax/reference     example
       ======        ================     =======
       INBOX         c.f., Section 5.1    im:fred@example.com
  -->
<!ENTITY % INBOX "CDATA">
<!--
  Abstract syntax for the message operation
  -->
<!ELEMENT message (#PCDATA)>
<!ATTLIST message
   source %INBOX; #REQUIRED
   destination %INBOX; #REQUIRED
   transID %UNIQID; #REQUIRED
>
```

8. The Presence Service DTD

```
   <!--
     DTD for the abstract IM presence service, as of 2000-08-16

     Refer to this DTD as:

       <!ENTITY % IMPRESENCE PUBLIC "-//Blocks//DTD IM PRESENCE//EN"
              "http://xml.resource.org/syntaxes/IM/im-presence.dtd">
       %IMPRESENCE;
     -->
   <!ENTITY % IMCOMMON PUBLIC "-//Blocks//DTD IM COMMON//EN"
              "http://xml.resource.org/syntaxes/IM/im-common.dtd">
   %IMCOMMON;
   <!--
     DTD data types:

          entity        syntax/reference     example
          ======        ================     =======
          PRESENTITY    c.f., Section 5.2    pres:fred@example.com
     -->
   <!ENTITY % PRESENTITY "CDATA">
   <!--
     Abstract syntax for presence information
     -->
   <!ELEMENT presence (tuple+)>
   <!ATTLIST presence
      entityInfo %URI; ""
   >
   <!ELEMENT tuple (#PCDATA)>
   <!ATTLIST tuple
      destination %URI; #REQUIRED
      status (open | closed) #REQUIRED
   >
   <!--
     Abstract syntax for the subscribe operation
     -->
   <!ELEMENT subscribe EMPTY>
   <!ATTLIST subscribe
      watcher %PRESENTITY; #REQUIRED
      target %PRESENTITY; #REQUIRED
      duration %SECONDS; #REQUIRED
      transID %UNIQID; #REQUIRED
   >
   <!--
     Abstract syntax for the notify operation
     -->
   <!ELEMENT notify (presence)>
```

```
   <!ATTLIST notify
      watcher %PRESENTITY; #REQUIRED
      target %PRESENTITY; #REQUIRED
      transID %UNIQID; #REQUIRED
   >
   <!--
     Abstract syntax for the unsubscribe operation
     -->
   <!ELEMENT unsubscribe EMPTY>
   <!ATTLIST unsubscribe
      watcher %PRESENTITY; #REQUIRED
      target %PRESENTITY; #REQUIRED
      transID %UNIQID; #REQUIRED
   >
```

9. The Presence Information DTD

```
<!--
  DTD the IM presence information of 2000-11-6

  Refer to this DTD as:

     <!ENTITY % IMPRESENCEINFO PUBLIC "-//Blocks//DTD IM PRESENCE//EN"
       "http://xml.resource.org/syntaxes/IM/im-presence-info.dtd">
       %IMPRESENCEINFO;
  -->

<!ENTITY % IMCOMMON PUBLIC
           "-//Blocks//DTD IM COMMON//EN"
           "http://xml.resource.org/syntaxes/IM/im-common.dtd">
%IMCOMMON;

<!--
  DTD data types: entity syntax/reference example

       ======       ================    =======
       PRESENTITY   c.f., Section 5.2   pres:Fred@example.com
  -->

<!ENTITY % PRESENTITY "CDATA">
<!--
  Abstract syntax for presence information -->

<!ELEMENT presence (tuple+)>
<!ATTLIST presence
  entityInfo %URI; ""
>

<!ELEMENT tuple (#PCDATA)>
<!ATTLIST tuple
  destination %URI; #REQUIRED
  status (open | closed) #REQUIRED
>
```

References

[1] Crocker, D., "Standard for the format of ARPA Internet text messages", RFC 822, STD 11, Aug 1982.

[2] Mockapetris, P.V., "Domain names - concepts and facilities", RFC 1034, STD 13, Nov 1987.

[3] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[4] Callas, J., Donnerhacke, L., Finney, H. and R. Thayer, "OpenPGP Message Format", RFC 2440, November 1998.

[5] Klyne, G., "XML coding of RFC822 messages", I-D draft-klyne-message-rfc822-xml-00.txt, November 2000.

[6] Ramsdell, B., "S/MIME Version 3 Certificate Handling", RFC 2632, June 1999.

[7] Day, M., Rosenberg, J. and H. Sugano, "A Model for Presence and Instant Messaging", RFC 2778, February 2000.

[8] Day, M., Aggarwal, S. and J. Vincent, "Instant Messaging / Presence Protocol Requirements", RFC 2779, February 2000.

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

[10] Allocchio, C., "GSTN Address Element Extensions in E-mail Services", RFC 2846, June 2000.

Authors' Addresses

Dave Crocker
Brandenburg Consulting
675 Spruce Drive
Sunnyvale, CA 94086
US

Phone: +1 408 246 8253
EMail: dcrocker@dcrocker.net

Athanassios Diacakis
Network Projects Inc.
4516 Henry Street
Suite 113
Pittsburgh, PA 15213
US

Phone: +1 412 681 6950 x202
EMail: thanos@networkprojects.com

Florencio Mazzoldi
Network Projects Inc.
4516 Henry Street
Suite 113
Pittsburgh, PA 15213
US

Phone: +1 412 681 6950
EMail: flo@networkprojects.com

Christian Huitema
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
US

EMail: huitema@microsoft.com

Graham Klyne
Content Technologies
Henley Business Centre, Newtown Road
Oxfordshire RG9 1HG
UK

Phone: +44 118 930-1300
EMail: GK@Dial.pipex.com

Marshall Rose
Invisible Worlds
1179 North McDowell Boulevard
Petaluma 94954-655
US

Phone: +1 916-483-8878
EMail: mrose@dbc.mtview.ca.us

Jonathan Rosenberg
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
US

EMail: jdrosen@dynamicsoft.com

Robert Sparks
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
US

EMail: rsparks@dynamicsoft.com

Hiroyasu Sugano
Fujitsu Laboratories Ltd.
64 Nishiwaki, Ohkubo-cho
Akashi 674-8555
JP

EMail: suga@flab.fujitsu.co.jp

Appendix A. IM URL IANA Registration Template

This section provides the information to register the im: instant messaging URL.

A.1 URL scheme name

im

A.2 URL scheme syntax

The syntax replicates the existing mailto: URL syntax specified in RFC2368. The ABNF is:

```
IM-URL = "im:"     [ to ]  [ headers ]
to         =  #mailbox
headers    =  "?" header *( "&" header )
header     =  hname "=" hvalue
hname      =  *urlc
hvalue     =  *urlc
```

A.3 Character encoding considerations

Representation of non-ASCII character sets in local-part strings is limited to the standard methods provided as extensions to RFC 822[1]

A.4 Intended usage

Use of the im: URL follows closely usage of the mailto: URL. That is, invocation of an IM URL will cause the user’s instant messaging application to start, with destination address and message headers fill-in according to the information supplied in the URL.

A.5 Applications and/or protocols which use this URL scheme name

It is anticipated that protocols compliant with RFC2779, and meeting the interoperability requirements specified here, will make use of this URL scheme name.

A.6 Interoperability considerations

The underlying exchange protocol used to send an instant message may vary from service to service. Therefore complete, Internet-scale interoperability cannot be guaranteed. However, a service conforming to this specification permits gateways to achieve interoperability sufficient to the requirements of RFC2779.

A.7 Security considerations

When IM URLs are placed in instant messaging protocols, they convey the identity of the sender and/or the recipient. In some cases, anonymous messaging may be desired. Such a capability is beyond the scope of this specification.

A.8 Relevant publications

RFC2779, RFC2778

A.9 Person & email address to contact for further information

Dave Crocker<dcrocker@dcrocker.net>

A.10 Author/Change controller

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

A.11 Applications and/or protocols which use this URL scheme name

Instant messaging service; presence service

Appendix B. PRES URL IANA Registration Template

This section provides the information to register the pres: presence URL .

B.1 URL scheme name

pres

B.2 URL scheme syntax

The syntax replicates the existing mailto: URL syntax specified in RFC2368. The ABNF is:

```
PRES-URL = "pres:"     [ to ]  [ headers ]
to         =  #mailbox
headers    =  "?" header *( "&" header )
header     =  hname "=" hvalue
hname      =  *urlc
hvalue     =  *urlc
```

B.3 Character encoding considerations

Representation of non-ASCII character sets in local-part strings is limited to the standard methods provided as extensions to RFC 822[1]

B.4 Intended usage

Use of the pres: URL follows closely usage of the mailto: URL. That is, invocation of an PRES URL will cause the user's instant messaging application to start, with destination address and message headers fill-in according to the information supplied in the URL.

B.5 Applications and/or protocols which use this URL scheme name

It is anticipated that protocols compliant with RFC2779, and meeting the interoperability requirements specified here, will make use of this URL scheme name.

B.6 Interoperability considerations

The underlying exchange protocol used for presence may vary from service to service. Therefore complete, Internet-scale interoperability cannot be guaranteed. However, a service conforming to this specification permits gateways to achieve interoperability sufficient to the requirements of RFC2779.

B.7 Security considerations

When PRES URLs are placed in presence protocols, they convey the identity of the sender and/or the recipient. In some cases, anonymous messaging may be desired. Such a capability is beyond the scope of this specification.

B.8 Relevant publications

RFC2779, RFC2778

B.9 Person & email address to contact for further information

Dave Crocker<dcrocker@dcrocker.net>

B.10 Author/Change controller

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

B.11 Applications and/or protocols which use this URL scheme name

Instant messaging service; presence service

Appendix C. Issues of Interest

This appendix briefly discusses issues that may be of interest when designing an interoperation gateway.

C.1 Address Mapping

When mapping the service described in this memo, mappings which place special information into the im: address local-part MUST use the meta-syntax defined in RFC 2486[10].

C.1.1 Source-Route Mapping

The easiest mapping technique is a form of source-routing and usually is the least friendly to humans having to type the string. Source-routing also has a history of operational problems.

Use of source-routing for exchanges between different services is by a transformation that places the entire, original address string into the im: address local part and names the gateway in the domain part.

For example, if the destination INSTANT INBOX is "pepp://example.com/fred", then, after performing the necessary character conversions, the resulting mapping is:

    im:pepp=example.com/fred@relay-domain

where "relay-domain" is derived from local configuration information.

Experience shows that it is vastly preferrable to hide this mapping from end-users Ãù if possible, the mapping should be performed automatically by the underlying software.

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.