INTERNET-DRAFT
Expires: April 2, 2000

Mark Day
Lotus

Sonu Aggarwal
Microsoft

Gordon Mohr
CMGI Solutions

Greg Hudson
MIT

Proposed Design Decisions for IMPP
draft-day-impp-basis-00.txt

1. Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This document and related documents are discussed on the impp mailing list. To join the list, send mail to impp-request@iastate.edu. To contribute to the discussion, send mail to impp@iastate.edu. The archives are at http://lists.fsck.com/cgi-bin/wilma/pip. The IMPP working group charter, including the current list of group documents, can be found at http://www.ietf.org/html.charters/impp-charter.html.

2. Abstract

As of this writing, the IMPP working group has largely reached consensus on the requirements for IMPP [Reqts]. In addition, Hudson [Issues] has catalogued a set of design issues discussed on the mailing list.

This document proposes a starting point for further discussions about

the design of the Instant Messaging and Presence Protocol (IMPP). It is not a fully-specified protocol, but a summary of design decisions on which the authors were able to reach agreement during two full days of discussion. Our discussions drew on knowledge of operational and architectural aspects of Activerse Ding!, Lotus Sametime, Microsoft Exchange instant messaging, MSN Messenger, and Zephyr, among other diverse, widely-deployed systems supporting presence and/or instant messaging. We believe that the issues on which we reached agreement are issues on which the working group as a whole should be able to quickly reach rough consensus. We hope that the working group will be able to adopt these decisions as the basis for further work on designing IMPP; of course, we recognize the possibility that strong technical arguments not hitherto considered may cause the group to arrive at different decisions in some instances.

We also identified some areas, primarily related to security, in which we were lacking enough information or experience to make a solid recommendation. We draw the working group's attention to these issues in the hope that others can contribute.

Although we have treated presence and instant messaging in the same document, we believe that these design decisions allow implementors to offer either service independently of the other.

3. Contents

4. Inter-Domain Architecture

We agreed that there are multiple independent domains. Each domain has its own users and handles authentication of users within that domain.

A domain has 3 possibly-separable functions for its users: authenticator, traffic-forwarder, and locator/directory. The authenticator is responsible for authenticating a user to the domain. A traffic-forwarder is responsible for getting requests and responses to/from other domains. A locator/directory is responsible for mapping user-friendly names to service addresses.

We discussed whether SRV lookups are required: that is, whether a client must perform an SRV lookup to find an appropriate server within its domain, and whether a server must perform an SRV lookup to find an appropriate server in another domain. Although requiring such a lookup substantially increases the complexity of the minimal client, we eventually concluded that the improved manageability is worth the cost.

5. Protocol substrate

We decided that the protocol should be carried on TCP. We discussed many other possibilities, but found all of them had more drawbacks than TCP.

To briefly summarize:

UDP with TCP fallback -- DNS works this way, and operational experience suggests that the occasional TCP-mode connections alarm administrators. The fallback mechanism means additional complexity in protocol.

Straight UDP -- would need fragmentation/reassembly story (for IMs larger than a packet -- can't guarantee short msgs). The protocol would also need to keep authentication information and similar "connection-related" state. UDP with all these elaborations seems like TCP but slower, and adds unnecessary implementation complexity.

UDP with URL reference for large messages -- this was suggested but was not popular. One concern: when can the web site stop holding the message at that URL?

Operational experience with both Lotus Sametime and Microsoft Exchange deployments suggest that TCP's limitations aren't significant in practice.

6. Naming

We decided that there should be an email-like form of IMPP address that can be presented to users and solicited from users: user@host. This facilitates using existing email addresses as IMPP addresses, where possible, to make it easier for end-users. Operational experience suggests that end users would prefer to avoid remembering and communicating a new IMPP-specific address.

However, the canonical form is a URL because it's unambiguous: impp://host/user.

The "email form" transforms to/from URL by simple textual rearrangement. Other than possible quoting of disallowed characters, there is no rewriting or rearrangement internal to the host or user part. After quoting translation is done, the string on the left side of the "@" in the email form is identical to the string on the right side of the single "/" in the URL form.

7. Reliability and Responses

We decided that all requests, including change notifications, potentially need to be acknowledged.

8. Presence format

We decided that XML has advantages as a basis for the presence format markup. However, we were aware of the deficiencies of XML as discussed on the mailing list. We agreed that the best solution would be to use a simplified XML as the basis for the markup. The goal is to have a markup that can be parsed easily by those implementations that have an existing XML parser, but also can be parsed easily by those who have to build a parser from scratch. Our initial agreement is that the markup will use tags only, no attributes, and no processing instructions.

We discussed the use of vCard as a basis for the presence format. We decided that it would be useful to copy many of its fields and mechanisms, but that there is no need to either copy vCard syntax nor attempt to treat the presence format as an extension of vCard.

Based on the desirability of using MIME-based security mechanisms (S/MIME, PGP/MIME)for end-to-end security, we also agreed that a presence information object should be a MIME object. So a presence information object is a MIME object containing XML-based information. We also agreed that text/xml is not the correct MIME content-type for the contained markup; it should probably be application/impp.

9. Instant message format

We decided that the format of instant messages is essentially identical to email, using RFC-822 headers [RFC822] and MIME [MIME].

10. Authentication

Domains authenticate themselves to each other independent of how users authenticate themselves to a domain. Domain 1 can authenticate itself to Domain 2 and thereby allow user in Domain 1 to subscribe to presence information of a user in Domain 2.

Domain 1 can make policy choices as to whether users should contact Domain 2 via Domain 1 servers or directly.

Domain 2 can choose to refuse traffic from users in Domain 1 except when it comes via Domain 1 servers.

Domains trust or distrust each other to do the right thing. Domains trust or distrust their users. These trust concerns are independent: Domain 1 can either trust or not that Domain 2 has done a good job of authenticating user@Domain2. Domain 1 cannot condition its trust of user@Domain2 on the authentication scheme used between that user and Domain2.

11. Operations

11.1 Presence Operations

The following set of operations is sufficient: Fetch, Subscribe, Unsubscribe, Change, Change-Notify, Terminate-Subscription, Set-ACL, Get-ACL.

However, we did not all agree on the need for separate Set-ACL and Get-ACL operations, noting that it would be possible to encode access control lists (ACL) in the presence information itself. With such an encoding, the Fetch and Change operations would be used to get and set that information.

We agreed that the full set of operations is required for

client/server interoperability (i.e. for company X client to talk with company Y server). A smaller set is required for inter-domain interoperability, where the missing operations can be assumed to only take place within a domain: Fetch, Subscribe, Unsubscribe, Change-Notify, Terminate-Subscription.

11.2 Instant Message Operations

A single operation is sufficient: Send. This operation is required for all forms of interoperability.

12. Connection management

We considered the problem of managing the transfer of a large amount of data: for example, the delivery of small instant messages might be delayed while waiting for a multi-megabyte transfer to complete on a single TCP connection.

Although there are potential advantages to having the protocol explicitly "chunk" such large messages (multiplexing the underlying TCP connection), we decided against the added complexity.

Instead, we decided there should be no explicit chunking in the protocol. In the case where a short message is delayed by a long transfer, we recommend the opening of another TCP connection and letting the transport layer do the chunking.

We also agreed that as a management issue, both the initiator and the target of a connection open should be able to indicate capacity limits and/or maximum acceptable message size. Exceeding these advertised limits is sufficient grounds for the victim to close the connection.

13. Inter-domain connection flow

A TCP connection between two domains may be used on behalf of multiple users in both domains, and accordingly may be kept open for long times. However, correct operation of the protocol cannot depend on keeping such an interdomain connection open. In particular, a domain may need to deal with a situation in which it must open a connection to another domain in order to send a response or a change notification.

14. Leases

Every subscription has an associated lease time, set by the server where the subscription state is stored. To maintain a subscription, the subscribing party must renew that lease before it expires.

The renewal process is initiated by the subscribing client.

The renewal time may be set to different intervals for local (client-to-domain-server) and remote (interdomain) connections by a sophisticated local domain server, although a server can also simply forward information between a remote domain and a local user without altering any renewal times. In general, the lease time(s) are set by the server(s) that must manage the subscription resources.

15. Security

15.1. Connection-Level Security

15.1.1. User-to-Domain

We agreed that a plausible starting proposal for user-to-domain connection-level security is to use SASL [SASL] as the negotiation framework. We agreed that negotiating to TLS [TLS] should be possible but is not reasonable to mandate (because of performance concerns, among others).

15.1.2. Inter-domain

We believe that a plausible starting proposal is to use SASL as the negotiation framework, but we do not know what are reasonable solutions to negotiate to within that framework.

We recommend that the working group organize to answer the following questions:

1. What SASL mechanisms are most applicable to interdomain traffic?

2. If IMPP allows both secure and insecure domains, how can IMPP prevent an attacker from pretending to be a secure domain operating insecurely?

3. What, if anything, needs to be included in an application protocol specification so that it can use TLS or other transport-level mechanisms?

15.2. End-to-end security

We decided that both the presence format and the instant message format should have their content expressed as MIME, so that end-to-end solutions using S/MIME and PGP/MIME are usable.

16. Access Control Lists (ACLs)

ACLs and how to manipulate them are properly the subject of a longer

discussion on their own, and we recommend the working group organize to carry out that discussion. IMPP can achieve interdomain interoperability without getting ACLs right, but client/server interoperability requires a consistent solution for setting and getting ACLs.

There was no consensus among the authors about whether we should plan to focus on interdomain interoperability first (possibly sacrificing client/server interoperability), or insist on client/server interoperability (thereby delaying interdomain interoperability).

17. Protocol Messages

This section outlines the messages of a protocol but does not represent a complete specification.

Every protocol message has a request-ID, version, method name, and method-dependent arguments.

Each protocol message is a request or a response. A party sending a request can (and usually should) generate a new request-ID for the request. A party sending a response must use a request-ID derived from the request-ID of the corresponding request.

The requests are:

SUBSCRIBE
UNSUBSCRIBE
FETCH
SEND
CHANGE
CHANGE-NOTIFY
TERMINATE-SUBSCRIPTION

The responses are:

OK
ERROR

18. Protocol Transitions

The following sections describe each of the requests in more detail. Again, this is not intended as a complete protocol specification. ACLs are not considered at all, and the methods that are described are not necessarily described completely.

We sketch the arguments and the meaning of an OK response. We also provide a list of likely ERROR responses, though of course no single request can produce all of these errors.

Note that this section does not specify any particular protocol encoding; the next section considers protocol encoding decisions.

18.1. SUBSCRIBE

SUBSCRIBE Target-URL

OK: returns Presence-Information, Lease-Time.
 -- The subscription has been created or extended for the length of time indicated by Lease-Time.

18.2. UNSUBSCRIBE

UNSUBSCRIBE Target-URL

OK:
 -- The subscription has been eliminated.

18.3. FETCH

FETCH Target-URL

OK: returns Presence-Information

18.4. SEND

SEND Target-URL Message-Body

OK: returns Presence-Information [usually empty but sometimes containing "on vacation"-style information]
 -- message has been delivered to target

18.5. CHANGE

CHANGE Target-URL Presence-Information Lease-Time
[Special value of "infinity" in Lease-Time means to change underlying "permanent" value.]

OK: returns Lease-Time (possibly different from supplied Lease-Time).
 -- The new Presence-Information will be supplied until the expiration of the returned Lease-Time.

18.6. CHANGE-NOTIFY

CHANGE-NOTIFY Target-URL Origin-URL Presence-Information

OK:

-- Notification received, Target-URL acknowledges that Origin-URL now has the value of Presence-Information described.

18.7. TERMINATE-SUBSCRIPTION

TERMINATE-SUBSCRIPTION Target-URL Origin-URL

OK:
-- Target-URL acknowledges that Origin-URL has terminated any subscription that Target-URL previously had to Origin-URL.

18.8. Kinds of ERROR

Target unknown;
Subscription unknown;
Permission denied;
User authentication required;
Domain authentication required [interdomain only];
Temporarily forwarded to <URL>;
Permanently forwarded to <URL>;
Operation not supported;
Redirect to <server>;
Connection failure;
Message too large;
Mailbox full;
Mailbox unavailable;
MIME type not understood.

19. Protocol Encoding

We agreed that the protocol should be encoded as text. The syntactic style of the encoding follows HTTP, but there is no attempt to either extend HTTP or to reuse HTTP.

The following is an example of the way the protocol could be encoded in compliance with this decision. We emphasize that this is only an example, and we are not arguing that this is the exact protocol encoding that must be used.

On interdomain requests, two additional headers are required. The From: header identifies the originator of a request, and is used by a remote domain to determine the domain to which a response should be sent. The Connection-ID: header allows the user's local domain to distinguish between different connections or logins of the same user. We propose that the originating client should set these headers in all cases. The local domain must check these headers to ensure their correctness but does not rewrite or envelope the original request.

In general, if additional information is needed for inter-domain

requests beyond From: and Connection-ID:, this example encoding might require a server to insert information in the middle of a request. We agreed that this would be a problem in terms of efficiency and ease of debugging. Accordingly, we agreed that the working group may need to develop an encoding with a clear notion of an envelope for inter-domain information that leaves the original client request untouched inside the envelope.

Here is an example of a user subscribing to presence information in a different domain:

```
SUBSCRIBE 57AQ impp/1.0
From: impp://mit.edu/ghudson
Connection-ID: 6
Target: impp://example.com/Mark_Day
```

57AX is the new request-ID, impp/1.0 is a protocol version number. The From: and Connection-ID: fields make explicit information that the local domain already knows. This information is expected to be determined during session setup.

On receiving the request from an mit.edu domain server, an example.com domain server takes the necessary steps to create a subscription and sends the following reply:

```
OK (SUBSCRIBE) 57AQ impp/1.0
To: impp://mit.edu/ghudson
Connection-ID: 6
Lease-Time: 3000
Content-Type: Application/Presence
Content-Length: n
....
```

OK indicates a succesful operation, the original request is indicated in parentheses to aid debugging, and the original request-ID (57AQ) is used for the matching response. The original interdomain From: header becomes an interdomain To: header, and the original Connection-ID: header is sent back unchanged. The "...." following the Content-Length: header is an n-byte MIME-encoded object containing an XML-based markup of the presence information for impp://example.com/Mark_Day.

20. Conclusions

We have reached consensus on the following design decisions, which we recommend to the IMPP working group as a whole:

1. A multi-domain architecture, in which user-to-domain authentication is separated from inter-domain authentication.
2. TCP as the protocol substrate.
3. URLs as canonical names, with an email-like form as user-presentable

equivalent.
4. Requiring SRV lookups by clients.
5. Encoding protocol operations as text in an HTTP-like style.
6. Using the requests SUBSCRIBE, UNSUBSCRIBE, FETCH, SEND, CHANGE, CHANGE-NOTIFY, and TERMINATE-SUBSCRIPTION.
7. Having OK or ERROR responses to any request.
8. Encoding presence information as a MIME object containing a presence format based on simple XML.
9. Encoding an instant message as a MIME object.
10. Using SASL to negotiate user-to-domain authentication.

We also recommend further study and discussion by the working group to investigate the security-related architecture and specific issues suggested.

We see this document as a plausible starting point of a design process for IMPP. We acknowledge that the working group may choose not to incorporate this document into its process, and indeed may organize the design process in an entirely different fashion. Nevertheless, we have joined together in producing this document with the hope that our collaboration might serve as the basis for a group-wide consensus on some relevant issues.

21. References

[Reqts] M. Day, S. Aggarwal, G. Mohr, J. Vincent. Instant Messaging / Presence Protocol Requirements. Work in progress, draft-ietf-impp-reqts-03.txt.

[Issues] G. Hudson. Instant Messaging / Presence Protocol Design Issues. Work in progress, draft-hudson-impp-issues-00.txt.

[SASL] J. Myers. Simple Authentication and Security Layer (SASL). RFC 2222, October 1997.

[TLS] T. Dierks, C. Allen. The TLS Protocol Version 1.0. RFC 2246, January 1999.

[RFC822] D. Crocker. Standard for the format of ARPA Internet text messages. RFC 822, August 1982. Also see updates: RFC-1123, RFC-1138, RFC-1148, RFC-1327, RFC-2156.

[MIME] N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One. RFC 2045, November 1996.

[S/MIME] B. Ramsdell, ed. S/MIME Version 3 Message Specification. RFC 2633, June 1999.

[PGP/MIME] M. Elkins. MIME Security with Pretty Good Privacy (PGP). RFC 2015, October 1996.

[HTTP] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, **P. Leach, T. Berners-Lee. Hypertext Transfer Protocol -- HTTP/1.1. RFC** 2616, June 1999.

22. Authors' Addresses

Mark Day
<Mark_Day@lotus.com>
Lotus Development Corporation
55 Cambridge Parkway
Cambridge, MA 02142
USA

Sonu Aggarwal
<sonuag@microsoft.com>
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

Gordon Mohr
<gojomo@activerse.com>
CMGI Solutions, Inc.
1301 W. 25th St Suite 500
Austin, TX 78705
USA

Greg Hudson
<ghudson@mit.edu>
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139
USA