Lemonade
Internet Draft: Lemonade Requirements Server to Client Notifications
Document: draft-smaes-lemonade-s2c-notification-reqs-01
Expires: August 2005

S. H. Maes
C. Wilson

February 2005

Lemonade Requirements for Server to Client Notifications

Status of this Memo

Abstract

This document describes Lemonade requirements for server to client notifications. These server to client notifications provide information on crucial changes to a client.

This document does not assume how notifications are provided to the clients: the client to server notifications may be actively pushed to a client through different mechanisms rather than requiring the client to initiate contact to ask for state changes; or they may be polled by the client, still avoiding that the client performs full state comparisons.

Conventions used in this document

In examples, "C:" and "S:" indicate lines sent by the client and server respectively.

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 [RFC2119].

An implementation is not compliant if it fails to satisfy one or more of the MUST or REQUIRED level requirements for the protocol(s) it implements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements for a protocol is said to be "unconditionally compliant" to that protocol; one that satisfies all the MUST level requirements but not all the SHOULD level requirements is said to be "conditionally compliant." When describing the general syntax, some definitions are omitted as they are defined in [RFC3501].

Table of Contents

1. Introduction

In this document, we assume clients with limited computing resources and battery life time able to connect to a Lemonade server through a network with constrained or costly bandwidth (e.g. mobile network).

The lemonade profile [LEMONADEPROFILE] extends IMAPv4 Rev1 [RFC3501] with optimizations that are especially useful when accessing, manipulating or sending messages from resource and bandwidth constrained clients.

Users of wired systems (desktop, laptop) have come to expect a quasi-instantaneous reflection on the client of changes that take place on the mail server. These same users are expecting the same behavior on mobile devices while, at the same time, minimizing bandwidth and power requirements to allow normal client response over a nominal time frame.

As a result, server to client notifications are required to support some or all of the following:
- avoid unnecessary polling when possible
- enable quasi instantaneous client wake up and update when possible
- avoid full fledged state comparison by the client as typically performed by IMAP clients.

The Lemonade server to client notifications inform the client of changes in an end user's mailbox.

Solutions like IMAP IDLE work well over well connected, high

bandwidth reliable network. IDLE is however not designed for use over a network with restricted bandwidth where connectivity can often be lost.

This document provides requirements for server to client notifications that would be outside the IMAP sessions.

Within the scope of IETF Lemonade, such server to client notifications are expected to be exchanged over IP. However, as discussed in [MEMAIL], it is important that mobile e-mail may also rely on notifications outside the IP band.

2. Use cases for Lemonade Server to Client Notifications

Use cases are at a very high level. As the work progress more details may be added. These use cases may also expand into additional use cases and requirements for Lemonade Profile.

2.1. Outband notifications

Permanent connections from the client to the server have an associated cost in battery power which may limit the lifetime of the device beyond the usefulness of the email client. Permanent connections may also be prohibitively expensive as well depending on the network operatorÆs billing structure.

In order to preserve battery power and/or limit the cost of connection, it is desirable for a mobile client not to maintain a permanent data connection (IP address, etc...). However, the user expects that his client will reflect changes that take place on the mail server quasi-instantaneously (e.g. new e-mail, deleted e-mail, change of status (read/unread), e-mail move ...).

By allowing the mobile device to receive outband notifications over a wireless network (GSM, CDMA, WLAN, à), the client is made aware of the event (i.e. it is awakened). The client can react as determined by its settings (user preferences, client settings ...) by updating its state if it has enough information, or connecting to the server to access the information required to update its state.

2.2. Inband notifications

Even when a client is always data connected, in order to preserve battery power or to limit the cost of connection, a client (mobile device) limits its requests for changes to the server. Still the user expects that his client will quasi-instantaneously reflect changes that take place on the mail server (e.g. new e-mail, deleted e-mail, change of status (read/unread), e-mail move, ...).

With a mechanism for inband notifications, the client can await for server events to request information on changes on the server. The client can react as determined by its settings (user preferences, client settings, ...) by updating its state (if it has enough information) or connect to the server to access the information required to update its state.

Such a mechanism should minimize bandwidth requirements and processing requirements on the client.

These notifications may be inband to the the IMAP band or within the same data channel but outside the IMAP band (e.g. SIP event notification).

2.3. Event-based synchronization

The client spends a significant time during the lifetime of the connection making sure that the server and client are properly synchronized. In order to minimize this cost (bandwidth and processing) a mobile client can avoid full state comparisons by simply collecting all the changes that took place on the server and applying them on the client. These events can be actively sent to the client by the server (notification) or made available to a client that request synchronization with the server.

2.4. Notification filtering

A user may decide to send the client (mobile device) only

notifications related to special events (e.g. e-mail marked urgent or e-mail from a particular sender).

Other changes can be kept on the server (delayed notification) and made available during the event-based synchronization that takes place when the client connects to the server.

2.5. Notification buffering or polling

On a network where notifications may not be reliable, a client may connect to server without having been notified or may not have received all the notifications the server has sent. The server then provides the notifications that have not yet been acted on for the client to perform event-based synchronization.

2.6. Changes of notification mechanisms

A user may want to be able to change the notification mechanisms to be used:

- Outband to a new device address / new device
- From inband (when connected) to outband (to save batteries, because of change of network technologies or because of change of cost of bandwidth).
- From pushed notifications (inband or outband) to periodic poll.

2.7. Changes of filtering

While connected, a user may want to be able to change the notification filters to be applied (e.g. to add a user or event for push notification or delayed notification).

2.8. Notification content

A server may simply notify that an event took place and invite the client to access the notification details from the server.

It may also provide more information about the details of the event

in the notification to allow the client, as determined by its settings (user preferences, client settings, ...), to immediately update its state prior to connecting to the server.

2.9. End-to-end Notification confidentiality

A server that provides notifications with more information about the details of the events must encrypt the notification to preserve confidentiality.

2.10. End-to-end Notification reliability

Notifications may not be reliable under some conditions (e.g. outband notifications over mobile network or inband notifications lost when connectivity is lost).

The server maintains the notifications that have not yet been acted on for the client to perform event-based synchronization. After a certain period of inaction, based on settings or preferences, the server may send (e.g. periodically) an additional notification to prompt the client to access these remaining notifications; until the client acts upon them.

Similarly, if no notification was received for a while, and based on settings or preferences, a client may access the server to check for missing notifications stored by the server.

2.11. Notification buffering

After determining that a client does not react to notifications, the server may stop sending them and solely stored / buffer them on the server.

A mechanism as described in the End-to-end Notification reliability use case can be used for the client to eventually receive them.

2.12. Notification from multiple server

A client receives notifications associated to multiple servers / e-mail accounts.

2.13. Quick re-synchronization

A client receives inband or outband information about the events that have taken place on the server. Using this information, the client

can either update it state (based on the information provided by the notification), decide to wait for further changes or information or decide to contact the server to access more information.

In such case, the event payload may contain a significant amount of information of the IMAP state changes.

2.14. Notification service provider

A client is notified by a notification service provided by a service provider that reacts to events communicates by the e-mail server in an enterprise domain. Confidentiality and integrity of the notifications is maintained. A typical example would be a mobile e-mail service provided by a GSM or CDMA operator that provides client to server notification (and support for Lemonade profile) to enterprise e-mail server and employees.

2.15. Usage patterns

[MEMAIL] provides examples of deployment models including for notifications.

3. Server to server notifications

Lemonade is completing work on the requirements for server to server notifications: [LEMONADES2S].

Server to server notifications are supposed to inform other server of email server events and changes in the IMAP server state. They are typically taking place outside the IMAP band. It is natural to expect them to take place over IP.

As such it would be natural to target sharing specifications between S2C and S2S notifications. Indeed, S2S requirements seem to be essentially a subset of the S2C requirements described here, except may be for the nature of the information to exchange.

3.1. A mobile e-mail use case that releates S2C and S2S

[MEMAIL] introduces deployment models A, B, C.1 and C2. In all cases, there is value to offer a scalable notification mechanisms between the e-mail server and the mobile e-mail enabling server.

Deployment models A and B may not require such mechanisms if the two server are implemented as one component. But in general, this is not the case.

The notifications passing between the server and the mobile e-mail enabling server are expected to be very similar to the notifications

passed between the mobile enabling server and the client. The difference may be that the server to server notifications may cover multiple e-mail account and describe the details of the e-mail account in the notifications, while the server to client notifications are expected to be limited to a single or few accounts accessed by the clients.

Different usage model can be considered:

- The mobile e-mail enabling server processes the S2S event, adapts it to the affected client and sends it as a S2C notification to the client
- The mobile e-mail enabling server processes the S2S events and access the server to get more information. It then generates, if appropriate, a S2C notification for the affected clients and it sends it to that client.

3.2. Implementation considerations

Lessons learned from implementation and deployments of such systems indicate the scalability and reliability challenges with server to server notifications schemes that impose maintaining sessions between the server (e.g. IDLE session per user or device).

4. Requirements for Lemonade Client to Server Notifications

This section contains a list of requirements for the Lemonade Client to Server Notifications.

R-1: Notifications MUST support association to server mail events including:

- New incoming e-mail
- E-mail status change (read/unread, deleted, ...)
- E-mail moved to new folder
- New folder
- New sent e-mail

R-2: Notifications MUST support partial description that an event took place, as decided by server settings or preferences

R-3: Notification MAY provide details of the event that took place on the server (i.e. more details than just mentioning that the even took place)

R-4: When notifications provide additional details, they MUST support end-to-end confidentiality between server and client

R-5: Notification mechanisms MUST be independent of the transport mechanisms

R-6: The server to client notification MUST specify server to client notification payload in ways that are notification transport independent.

R-7: Notifications MUST support outband notification mechanisms for clients and networks that support such mechanisms. These include:
- SMS
- Push (e.g. WAP Push)
- MMS
- IP (e.g. SIP event notifications)
- UDP

However, specifications will be limited to IP transports.

R-8: Notifications MUST support inband notification mechanisms for clients that are data connected to the network and support pushed notification to the client.

R-9: When available, it MUST be possible to use outband notification to wake up clients that are not permanently data connected to the network (e.g. no IP address).

R-10: Wake-up notification MUST allow passing additional information about the state of the IMAP server.

R-11: Lemonade servers MUST be able to store Notifications that have not yet been acted upon by the client and make them available when the client accesses the server.

R-12: A client MUST be able to query for Notifications that have not yet been acted upon by the client.

R-13: The overall notification mechanism MUST be end-to-end reliable even if the notification transport / channel may be unreliable (e.g. SMS).

R-14: The overall notification mechanism MUST provide event-based synchronization so that the client reflects all changes on the server based on settings / preferences / filtering.

R-15: The overall notification mechanisms MUST allow filtering of the notifications pushed to the client versus the notification kept on the server for when the client accesses it.

R-16: It MUST be possible to change the notification filtering rules from the client.

R-17: It MUST be possible to change the notification mechanisms from the client (e.g. new device, inband, outband, polling, ...)

R-18: The server MUST be able to limit the number of notifications sent to the client within a given time span if the client does not react to them and a certain number of notifications are pending on the server as determined by settings or preferences.

R-19: Clients MUST be able, based on settings or preferences, to handle situations where no notification has been received for a certain period of time by accessing the server and checking for notifications that have not been acted upon.

R-20: Servers MUST be able, based on settings or preferences, to handle situations where no notification has been acted upon for a certain period of time by periodically trying to notify the client server of pending notifications.

R-21: Outband notification MUST support the associated addressing

schema of the mobile network.

R-22: Notification SHOULD be designed to allow quasi-instantaneous transmission to the client when supported by network and device.

R-23: Notifications MUST be designed to minimize bandwidth requirements to convey their intended information.

R-24: A client MUST always be able to associate a notification with the correct originating server in order to update its state properly.

R-25: Notifications MUST be compatible with firewalls when appropriate.

R-26: Notification MUST be confidential when needed.

R-27: Proxy deployments MUST be compatible with end-to-end confidential notifications.

R-27: All notification mechanisms MUST be designed to minimize processing requirements on the client.

R-28: The notification payload MUST be extensible to accommodate S2S information requirements.

R-29: It MUST be possible to exchange the S2C notifications between servers.

R-30: The notifications MUST be compatible with the presence of intermediaries between the sender and recipient of the notifications.

R-31: The notification mechanisms MUST allow not constrain scalability of the server.

Security Considerations

We have the same security requirements for an in-response, polled and inband connectivity mode as IMAP.

For the outband connectivity mode, servers should use encryption methods for notifications if sensitive information is included in the payload of that notification.

When an implementation of Lemonade is proxy-based, this may create new security issues.

There may be SPAM issues. With the proliferation of SPAM opening notifications to a large user base could bring existing wireless networks to a halt. They may also lead to denial of service attack on client. Mechanisms may be needed to address these issues.

References

[LEMONADES2S] Decktor, G., ôServer To Server Notification Protocol Requirements", draft-ietf-lemonade-notify-s2s-00.txt, (Work in progress, proposed to WGLC as standard).

[MEMAIL] Maes, S.H., ôLemonade and Mobile e-mail", draft-maes-lemonade-mobile-email-xx.txt, (work in progress).

[RFC2119] Brader, S. "Keywords for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997. http://www.ietf.org/rfc/rfc2119

[RFC3501] Crispin, M. "IMAP4, Internet Message Access Protocol Version 4 rev1", RFC 3501, March 2003. http://www.ietf.org/rfc/rfc3501

[LEMONADEPROFILE] Maes, S.H. and Melnikov A., "Lemonade Profile", draft-ietf-lemonade-profile-XX.txt, (work in progress).

Version History

Revision 01:

[1] Update of status of document with boiler plate statement.
[2] Section 1: addition of scope qualifications and motivations.
[3] Section 2: Editorial re-ordering of text from section 1.
[4] Section 2.2: Additional details on inband notifications (i.e. within data connection but outside IMAP band).

[5] New section 2.13: Additional requirement on quick re-synchronization.

[6] New section 2.15: Additional considerations on mobile e-mail usage patterns.
[7] New section 3: Server to server notifications.
[8] Section 4: new requirements R-6, R-10 and R-28 to R-31.
[9] Updated references
[10] Version history
[11] Updated copyright section with boiler plate statements.

Authors Addresses

Stephane H. Maes
Oracle Corporation
500 Oracle Parkway
M/S 4op634
Redwood Shores, CA 94065
USA
Phone: +1-650-607-6296
Email: stephane.maes@oracle.com

Corby Wilson
Enterprise Mobility Systems
Nokia
503 Martindale Street
Suite 610
Pittsburgh, PA 15212
USA
Phone: +1-412-576-5402
Email: Corby.Wilson@nokia.com

Intellectual Property Statement

this standard. Please address the information to the IETF Executive Director.

Acknowledgement

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

Full Copyright Statement