Lemonade Intermediate meeting
Washington, DC
November 2004

Lemonade Requirements for Server to Client Notifications

draft-ietf-lemonade-server-to-client-notifications-00.txt

S. H. Maes

C. Wilson

Motivation

- Introduction
- Use Cases
- Requirements
- Security considerations

Use Cases (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 operator’s 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 ...).

Use Cases (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.

Use Cases (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.

Use Cases (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.

Use Cases (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.

Use Cases (6/7)

- 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 o cost of bandwidth).
  - From pushed notifications (inband or outband) to periodic poll.

- 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).

Use Cases (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.

Use Cases (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.

Use Cases (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.

Use Cases (11/12)

- 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.

- Notification from multiple server:

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

Use Cases (13)

- 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.

Requirements (1-5)

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

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 mechanism

Requirements (6-9)

R-6: Notifications MUST support outband notification mechanisms for clients and networks that support such mechanisms. These include:

- SMS
- Push (e.g. WAP Push)
- MMS
- IP/UDP (WLAN, BT)

R-7: Notifications MUST support inband notification mechanisms for clients that are data connected to the network and support pushed notification to the client.

R-8: 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-9: 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.

Requirements (10-14)

R-10: A client MUST be able to query for Notifications that have not yet been acted upon by the client.

R-11: The overall notification mechanism MUST be end-to-end reliable even if the notification transport / channel may be unreliable (e.g. SMS).

R-12: 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-13: 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-14: It MUST be possible to change the notification filtering rules from the client.

Requirements (15-19)

R-15: It MUST be possible to change the notification mechanisms from the client (e.g. new device, inband, outband, polling, ...)

R-16: 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-17: 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-18: 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-19: Outband notification MUST support the associated addressing schema of the mobile network that may differ from the IP addresses (that may not exist).

Requirements (20-26)

R-20: Notification SHOULD be designed to allow quasi-instantaneous transmission to the client when supported by network and device.

R-21: Notifications MUST be designed to minimize bandwidth requirements to convey their intended information.

R-22: A client MUST always be able to associate a notification with the correct originating server in order to update its state properly.

R-23: Notifications MUST be compatible with firewalls when appropriate.

R-24: Notification MUST be confidential when needed.

R-25: Proxy deployments MUST be compatible with end-to-end confidential notifications.

R-26: All notification mechanisms MUST be designed to minimize processing requirements on the client.

Security considerations

- 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.