

Introduction

There is a clear requirement and growing demand for mobile messaging. With publication of RFC 5550 (The Internet Email to Support Diverse Service Environments (Lemonade) Profile) in August 2009 now is a good time to examine progress made in Mobile Email, and what this means for users and system administrators. RFC 5550 is a product of the Internet Engineering Task Force (IETF) LEMONADE Working Group [LEMONADE]. The core of the LEMONADE work is based on Simple Mail Transfer Protocol (SMTP) [RFC5321], Internet Message Access Protocol (IMAP) [RFC3501] and Sieve mail filtering language [RFC5228], which provide a good basis for supporting mobile devices, such as PDAs, cell phones or just plain old laptops. There are a number of LEMONADE extensions, to optimize performance and functionality of mobile devices.

Capabilities provided by LEMONADE servers

Basic IMAP provides such capabilities as storing messages on the server and filing them into one or more mailboxes. In addition, IMAP is specifically designed to enable remote searching and message annotation. This allows the client to selectively download messages, which is of utmost importance for devices with limited memory and storage.

The SMTP and IMAP protocols provide a viable solution for mobile email. However, a key constraint in most mobile environments is relatively low connection bandwidth and relatively high connection latency. The underlying SMTP and IMAP protocols need to be used sensibly to optimize for this environment.

Let's examine features provided by LEMONADE servers in more detail. RFC 5550 mandates support for the following functionality in compliant servers (note that the list is not complete and omits some details):

- Extensions for avoiding overhead of base64 encoding of binary data in email (8BITMIME [SMTP-8BITMIME] and BINARYMIME [SMTP-BINARYMIME] SMTP extensions; BINARY [IMAP-BINARY] IMAP extension)
- Compression of email traffic (TLS compression in SMTP/IMAP [TLS-COMP], COMPRESS=DEFLATE [IMAP-COMPRESS] IMAP extension)
- Quick mailbox resynchronization [IMAP-CONDSTORE] [IMAP-QRESYNC]
- Improved server side searching and sorting facilities [IMAP-CONTEXT][IMAP-ESEARCH] [IMAP-SORT]
- "Forward without download" [IMAP-CATENATE][IMAP-URLAUTH][IMAP-UIDPLUS] [SMTP-BURL] that describes exchanges between Lemonade clients and servers to allow submitting new email messages incorporating content from other email messages stored on the server
- Attachment conversion [IMAP-CONVERT]
- Extensions to support notifications from mailstores [IDLE][IMAP-NOTIFY]
- Delivery-time email processing [RFC5228][SIEVE-IMAP4FLAGS][SIEVE-VACATION]
- Better error reporting, message delivery status notifications, etc.

Despite RFC 5550 being a new document, there are already several implementations of many of the features listed above. Due to the size limitations of this article it is not possible to describe in detail all of them. The major ones are described below.

8bit clean transport

With traditional (unextended) SMTP/IMAP, clients need to encode/decode binary attachments, because neither base SMTP, nor base IMAP allow 8bit data by default. For example this means that a JPEG picture attached to an email message needs to be encoded using base64 encoding. This not only requires extra CPU to process (which might be an issue for small devices such as cell phones), but also requires more bandwidth for which the user might have to pay. In particular, base64 encoding increases size of attachments by 1/3rd.

BINARY IMAP extension allows the client to download and uploaded attachments in “raw” (binary) form. 8BITMIME and BINARYMIME SMTP extensions allow clients to send messages in binary form. When either IMAP BINARY or 8BITMIME/BINARYMIME is not available, the client is either forced to send (via SMTP) and upload a copy of the message (via IMAP) in base64 encoded form, or it has to use 2 versions of the message (one with binary attachments and one with base64 encoded attachments) for sending and uploading.

Most of the existing SMTP servers (including Postfix, Sendmail and Microsoft Exchange) support 8BITMIME. Many clients support it as well, some just rely that it is always present.

Compression

SMTP/IMAP extensions for 8bit transport can be considered a form of compression, when compared to the unextended SMTP/IMAP. RFC 5550 goes one step further and requires that compliant IMAP/SMTP servers support Zlib compression. In both SMTP and IMAP this is available as a part of TLS support. In IMAP, compression can also be supported using COMPRESS=DEFLATE extension. COMPRESS=DEFLATE was added because in multiple OS's, TLS is implemented as a system library, so it might be quite difficult to extend such library to support compression.

Compression can provide good bandwidth saving. For example some tests with Polymer email client using COMPRESS=DEFLATE showed that IMAP protocol sent by the server can be compressed down to 35-40% of the original size.

Thunderbird 3 supports COMPRESS=DEFLATE.

Quick mailbox resynchronization

Resynchronization is a costly part of an IMAP session, and mobile networks are generally more prone to unintended disconnection, which in turn makes this problem more acute. Therefore RFC 5550 describes a suite of extensions to reduce the synchronization cost.

IMAP CONDSTORE [IMAP-CONDSTORE] and QRESYNC [IMAP-QRESYNC] extensions provide a way for a client to quickly resynchronize any mailbox by asking the server to return all flag changes and message expunges that have occurred since a previously recorded state. This works both to synchronize changes done by multiple clients (for example to mark messages read on a cell phone as read on a laptop mail client), and can also speed up client reconnect in case of TCP session disconnect.

In general, CONDSTORE/QRESYNC extensions become more useful with growth of mailbox size.

Without these extensions the client is forced to download IMAP flags for all old messages, irrespectively of whether or not any IMAP flag was changed. For a mailbox with 3000 messages this operation would cause the server to return about 140Kb of extra data. With CONDSTORE extension present this traffic can typically be eliminated (*).

(*) Note that CONDSTORE would require some extra traffic to convey message flag state. In the worst case scenario (when each messages had at least one flag changed since the last synchronization), CONDSTORE synchronization would result in about 190Kb of data per 3000 messages. However such situation is quite unlikely to occur in real life.

Support for CONDSTORE/QRESYNC is available in multiple servers, both open-source and commercial (e.g. Cyrus, Dovecot, Zimbra, Isode M-Box, Sun). Several email clients support CONDSTORE: Thunderbird 3 and Nokia Modest are among them.

Improved server side searching and sorting facilities

Server side sorting is one of the operations provided by the base IMAP specification. RFC 5550 extends this functionality in several ways:

- IMAP ESEARCH extension [IMAP-ESEARCH] allows the client to request results of a search in various forms – the first/last matching message, number of matching messages and a compact representation of all matching messages. Number of matching messages is of help to clients that are only interested in showing message counts (e.g. the number of unread and non-deleted messages). The compact representation of search results is a form of compression that can help to minimize traffic. Several email clients like those for the iPhone and Windows Mobile devices are using the unextended SEARCH command to find recent messages. They can benefit from the ESEARCH extension. For example in my INBOX with 28536 messages, there are currently 8771 messages marked as spam. A standard search for all such messages (“SEARCH KEYWORD Junk”) returns them as 52Kb long string. The same result returned from “SEARCH RETURN (COUNT ALL) KEYWORD Junk” results in a 21Kb long string.
- IMAP CONTEXT=SEARCH extension [IMAP-CONTEXT] provides a way to create virtual “views” of mailboxes, for example to only operate on non-spam messages received after a certain date. This technique, common in many desktop clients as a client-side capability, is useful for constrained clients to minimize the quantity of messages and notification data. This can also be used on sorted views of mailboxes.

Push Notifications

The concept of "push email" has been widely marketed as a desirable feature of mobile email services, to enable users to get immediate notification of and access to new messages.

Polling is a Poor Solution

The basic approach used by many email devices is to connect to the server to access new messages. This is a good model for many uses of mobile email, where access to email is under user control – the user checks email when the user wants to.

In order achieve automatic notification of new messages, a simple approach is to use 'polling' where the mobile client automatically connects to the server at intervals to check for new messages. However, there are two main problems with this approach:

1. Frequent polling is an inefficient use of network, mobile device and server resources. Polling generates more traffic which increases the cost to the user. Polling also eats more battery in devices like cell phones, as it prevents going into dormant mode to conserve battery.
2. New mail notification is only as frequent as the polling, and not 'immediate'. Many users are expecting to be notified about new mail within seconds of its arrival.

Polling is a poor solution for a user needing immediate notification.

In-band push notifications

Most of IMAP servers can notify clients about new mail arrival when such client issue any IMAP command. This is quite efficient. However most of the users don't receive email continuously, so sometimes IMAP clients don't have any tasks to do. The IMAP IDLE [IDLE] extension was designed in order to avoid polling the IMAP server for changes. IDLE is probably one of the most frequently implemented extensions in both clients (such as Outlook and Thunderbird) and servers, as indicated in [IMAP-SERVER-CAPA].

The work of the IDLE extension is simple: it introduces a new IMAP command called IDLE, which tells the server that the client is awaiting notification about any changes to the currently selected mailbox. Once such notification happens, the client cancels the IDLE command and can act upon the notification, for example to retrieve a newly arrived message.

However the IDLE extension is known to have some deficiencies, which prompted development of RFC 5465 [IMAP-NOTIFY]. In particular, IDLE doesn't allow to control which events should be generated by the server and how the client should be notified about them.

Out-of-band notifications

An alternative to the IMAP IDLE approach is a mechanism whereby the server pushes something to the client when a new message arrives, without there being an open TCP connection from the client to the server. Such out-of-band mechanism is useful when messages of interest to the user are infrequent. This allows the client to shutdown the IMAP connection to mailstore to conserve battery. It also helps to workaroud another problem with IMAP IDLE: due to IMAP inactivity autologout feature, IDLE connections need to be refreshed no later than every 30 minutes. (In practice NAT boxes make this interval even lower.) Such refreshes generate a bit of traffic, even though it is insignificant in comparison to the rest of IMAP exchange.

We will use Sieve email scripting language [RFC5228] [ENOTIFY] to demonstrate how out-of-band notifications can be generated. Support for Sieve is mandated for Lemonade-compliant Mail Delivery Agents.

```
require ["enotify"];

#Notify with out-of-band notification about messages from my boss
if header :contains "from" "boss@example.org" {
    # Notify about the new message using XMPP
    if notify_method_capability
        "xmpp:tim@example.com?message;subject=SIEVE"
        "Online"
        "yes" {
            notify :importance "1" :message "You got an email from boss"
                "xmpp:tim@example.com?message;subject=SIEVE";
        }
    }
}
```

In the above script notification is sent to a XMPP/jabber instant messaging (IM) account, if the corresponding IM user is online.

The following Sieve script shows how to generate out-of-band SMS notifications, but only when the client is not connected to mailstore using IMAP:

```
require ["enotify"];

#Notify with out-of-band notification about messages from my boss
if header :contains "from" "boss@example.org" {
    if not notify_method_capability
        "imap://myaccount@example.com"
        "Online"
        "yes" {
        notify :importance "1" :message "You got mail"
            "sms:+15105550101";
    }
}
```

In this example "+15105550101" is the destination phone number, "myaccount" is the IMAP username of the Sieve script owner on the server running IMAP ("example.com"). The `notify_method_capability` test is used to check if the specified user ("imap://myaccount@example.com") is "online".

If "sms:" notification method is not supported by the Sieve engine, in many cases "mailto:" notifications can be used instead. This assumes that your phone service provider has an email-to-SMS gateway.

Keep in mind that different Sieve engines support different notification methods. For example, Dovecot Sieve and Isode M-Box support "mailto:" notifications. In order to improve Sieve script portability between implementations, they can be written to test for supported notification methods using the "valid_notify_method" test. See [ENOTIFY] for more details.

Also note that in most cases notifications from Sieve scripts would require some extra Sieve engine configuration. For example SMS notifications would require configuring SMS gateway, and mailto: is likely to require configuration of the submission SMTP server.

Users who are keen to receive SMS notifications via a SMS-to-email gateway, but can't use the "notify" action because it is not available in the Sieve engine of their choice, can use the "redirect" action instead. But note that it would leave exact details of how conversion to SMS is handled up to a particular gateway. For example users can't rely on how the gateway is going to handle attachments, i.e. whether it is going just to strip them, convert them, or reject the whole message.

Delivery-time filtering

Sieve scripting language allows for quite flexible email processing. In addition to more traditional email processing such as checking and filing of email messages to different IMAP mailboxes, it also allows for automatic email notifications to the sender. This can be done using the Sieve "vacation" action [SIEVE-VACATION]. Despite the name this action is not limited to pure vacation auto-responder.

```
require ["enotify", "vacation"];

# Check if the sender is team-lead
if allof (envelope "from" "team-lead@example.net",
    not notify_method_capability
```

```

        "xmpp:myjid@example.com"
        "Online"
        "yes") {

vacation :mime text:
Content-Type: multipart/mixed; boundary=foo

--foo
Content-Type: text/plain; charset=us-ascii

I am sorry I am not available at the moment.
Here is my availability for this week.

--foo
Content-Type: text/calendar; charset=UTF-8; method=PUBLISH
Content-Transfer-Encoding: 7bit

BEGIN:VCALENDAR
VERSION:2.0
CALSCALE:GREGORIAN
METHOD:PUBLISH
BEGIN:VFREEBUSY
UID:901T095957Z-76A912@example.com
ORGANIZER:mailto:john_doe@example.com
DTSTAMP:20091012T083000Z
DTSTART:20091012T090000Z
DTEND:20091016T200000Z
FREEBUSY:20091012T090000Z/20091012T190000Z
FREEBUSY:20091013T090000Z/20091013T190000Z
FREEBUSY:20091016T090000Z/20091016T190000Z
COMMENT:This is my busy time information for this week
END:VFREEBUSY
END:VCALENDAR
--foo--
.
}

```

Another interesting way of using Sieve together with IMAP is by annotating email messages on delivery. This can be done using the `Imap4flags` Sieve extension [SIEVE-IMAP4FLAGS]. For example:

```

require ["fileinto", "imap4flags"];

if address :is "From" "myfriend@example.org" {
    addflag ["\\Flagged", "NonJunk"];
    keep;
    stop;
}

```

This script would mark messages with `From: myfriend@example.org` header as important (IMAP `\\Flagged` flag) and as non junk.

Conclusion

RFC 5550 provides plenty of mobile email features useful to end users. Implementations are beginning to appear.

References

- [LEMONADE-WG] <http://www.ietf.org/dyn/wg/charter/lemonade-charter.html>
- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.
- [RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.
- [RFC5228] Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language", RFC 5228, January 2008.
- [SMTP-8BITMIME] Klensin, J., Freed, N., Rose, M., Stefferud, E., and D. Crocker, "SMTP Service Extension for 8bit-MIMEtransport", RFC 1652, July 1994.
- [SMTP-BINARYMIME] Vaudreuil, G., "SMTP Service Extensions for Transmission of Large and Binary MIME Messages", RFC 3030, December 2000.
- [IMAP-BINARY] Nerenberg, L., "IMAP4 Binary Content Extension", RFC 3516, April 2003.
- [TLS-COMP] Hollenbeck, S., "Transport Layer Security Protocol Compression Methods", RFC 3749, May 2004.
- [IMAP-COMPRESS] Gulbrandsen, A., "The IMAP COMPRESS Extension", RFC 4978, August 2007.
- [IMAP-CONDSTORE] Melnikov, A. and S. Hole, "IMAP Extension for Conditional STORE Operation or Quick Flag Changes Resynchronization", RFC 4551, June 2006.
- [IMAP-QRESYNC] Melnikov, A., Cridland, D., and C. Wilson, "IMAP4 Extensions for Quick Mailbox Resynchronization", RFC 5162, March 2008.
- [IMAP-CONTEXT] Cridland, D. and C. King, "Contexts for IMAP4", RFC 5267, July 2008.
- [IMAP-ESEARCH] Melnikov, A. and D. Cridland, "IMAP4 Extension to SEARCH Command for Controlling What Kind of Information Is Returned", RFC 4731, November 2006.
- [IMAP-SORT] Crispin, M. and K. Murchison, "Internet Message Access Protocol - SORT and THREAD Extensions", RFC 5256, June 2008.
- [IMAP-CATENATE] Resnick, P., "Internet Message Access Protocol (IMAP) CATENATE Extension", RFC 4469, April 2006.
- [IMAP-UIDPLUS] Crispin, M., "Internet Message Access Protocol (IMAP) - UIDPLUS extension", RFC 4315, December 2005.
- [IMAP-URLAUTH] Crispin, M., "Internet Message Access Protocol (IMAP) - URLAUTH Extension", RFC 4467, May 2006.
- [SMTP-BURL] Newman, C., "Message Submission BURL Extension", RFC 4468, May 2006.
- [IMAP-CONVERT] Melnikov, A. and P. Coates, "Internet Message Access Protocol - CONVERT Extension", RFC 5259, July 2008.
- [IDLE] Leiba, B., "IMAP4 IDLE command", RFC 2177, June 1997.
- [IMAP-NOTIFY] Gulbrandsen, A., King, C., and A. Melnikov, "The IMAP NOTIFY Extension", RFC 5465, February 2009.
- [IMAP-SERVER-CAPA] <http://www.melnikov.ca/mel/devel/ServerReference.html>
- [ENOTIFY] Melnikov, A., Leiba, B., Segmuller, W., and T. Martin, "Sieve Email Filtering: Extension

for Notifications", RFC 5435, January 2009.

[SIEVE-IMAP4FLAGS] Melnikov, A., "Sieve Email Filtering: Imap4flags Extension", RFC 5232, January 2008.

[SIEVE-VACATION] Showalter, T. and N. Freed, "Sieve Email Filtering: Vacation Extension", RFC 5230, January 2008.