Network Working Group
Request for Comments: 5616
Category: Informational

N. Cook
Cloudmark

August 2009

Streaming Internet Messaging Attachments

Abstract

This document describes a method for streaming multimedia attachments received by a resource- and/or network-constrained device from an IMAP server. It allows such clients, which often have limits in storage space and bandwidth, to play video and audio email content.

The document describes a profile for making use of the URLAUTH-authorized IMAP URLs (RFC 5092), the Network Announcement SIP Media Service (RFC 4240), and the Media Server Control Markup Language (RFC 5022).

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

Email clients on resource- and/or network-constrained devices, such as mobile phones, may have difficulties in retrieving and/or storing large attachments received in a message. For example, on a poor network link, the latency required to download the entire attachment before displaying any of it may not be acceptable to the user. Conversely, even on a high-speed network, the device may not have enough storage space to secure the attachment once retrieved.

For certain media, such as audio and video, there is a solution: the media can be streamed to the device, using protocols such as RTP [RTP]. Streaming can be initiated and controlled using protocols such as SIP [SIP] and particularly the media server profiles as specified in RFC 4240 [NETANN] or MSCML [MSCML]. Streaming the media to the device addresses both the latency issue, since the client can start playing the media relatively quickly, and the storage issue, since the client does not need to store the media locally. A tradeoff is that the media cannot be viewed/played when the device is offline.

Examples of the types of media that would benefit from the ability to stream to the device include:

- Voice or video mail messages received as an attachment
- Audio clips such as ring tones received as an attachment
- Video clips, such as movie trailers, received as an attachment

The client may wish to present the user with the ability to use simple "VCR-style" controls such as pause, fast-forward, and rewind. In consideration of this, the document presents two alternatives for streaming media -- a simple mechanism that makes use of the announcement service of RFC 4240, and a more complex mechanism which allows VCR controls, based on MSCML (RFC 5022) [MSCML]. The choice of which mechanism to use is up to the client; for example, it may be based on limitations of the client or the configured media server. This document presents suggestions for determining which of these streaming services are available.

2. Conventions Used in This Document

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 RFC 2119 [KEYWORDS].

In examples, "C:" and "S:" indicate lines sent by the client and server, respectively. If a single "C:" or "S:" label applies to multiple lines, then some of the line breaks between those lines are for editorial clarity only and may not be part of the actual protocol exchange.

3. Mechanism

3.1. Overview of Mechanism

The proposed mechanism for streaming media to messaging clients is a profile for making use of several existing mechanisms, namely:

- IMAP URLAUTH Extension [URLAUTH] - Providing the ability to generate an IMAP URL that allows access by external entities to specific message parts, e.g., an audio clip.
- URLFETCH Binary Extension [URLFETCH_BINARY] - Providing the ability to specify BINARY and BODYPARTSTRUCTURE arguments to the URLFETCH command.

- Media Server Announcement Service (RFC 4240) [NETANN] - Providing the ability for a media server to stream media using a reference provided by the media server client in a URL.

- Media Server Interactive Voice Response (IVR) Service (RFC 5022) [MSCML] - Providing the ability to stream media as above, but with VCR-style controls.

The approach is shown in the following figure:

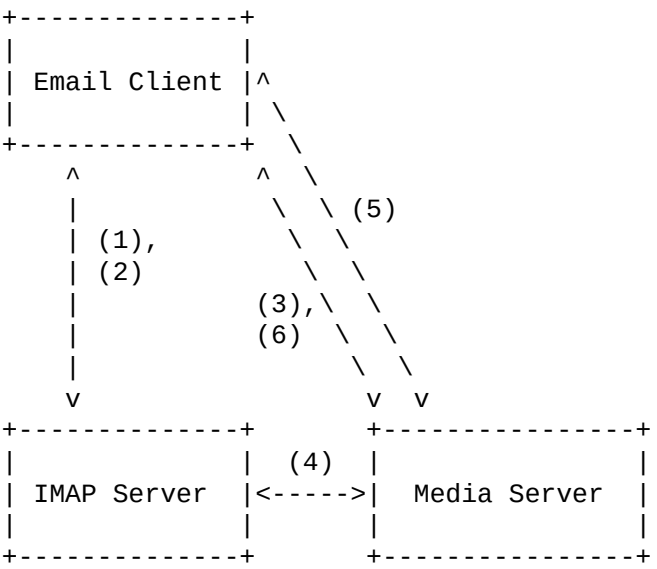


Figure 1: Proposed Mechanism

The proposed mechanism has the following steps:

(1) The client determines from MIME headers of a particular message that a particular message part (attachment) should be streamed to the user. Note that no assumptions are made about how/when/if the client contacts the user of the client about this decision. User input may be required in order to initiate the proposed mechanism.

(2) The client constructs an IMAP URL referencing the message part, and uses the GENURLAUTH [URLAUTH] command to generate a URLAUTH-authorized IMAP URL.

(3) The client connects to a SIP Media Server using the announcement service as specified in RFC 4240 [NETANN], or the IVR service as specified in RFC 5022 [MSCML], and passes the URLAUTH-authorized URL to the media server.

(4) The media server connects to the IMAP server specified in the referenced URL, and uses the IMAP URLFETCH [URLAUTH] command to retrieve the message part.

(5) The media server streams the retrieved message part to the client using RTP [RTP].

(6) The media server or the client terminates the media streaming, or the streaming ends naturally. The SIP session is terminated by either client or server.

It should be noted that the proposed mechanism makes several assumptions about the mobile device, as well as available network services, namely:

- The mobile device is provisioned with, or obtains via some dynamic mechanism (see Section 3.2), the location of a media server which supports either RFC 4240 [NETANN] and/or RFC 5022 [MSCML].

- The media server(s) used by the mobile device support the IMAP URL [IMAPURL] scheme for the announcement and/or IVR services.

- The IMAP server used by the mobile device supports generating anonymous IMAP URLs using the URLAUTH mechanism as well as the IMAP URLFETCH BINARY [URLFETCH_BINARY] extension.

3.2. Media Server Discovery

This section discusses possibilities for the automatic discovery of suitable media servers to perform streaming operations, and provides for such a mechanism using the IMAP METADATA [METADATA] extension.

There are two possibilities for clients with regard to determining the hostname and port number information of a suitable media server:

1. No discovery of media servers is required: clients are configured with suitable media server information in an out-of-band manner.

2. Discovery of media servers is required: clients use a discovery mechanism to determine a suitable media server that will be used for streaming multimedia message parts.

There are several scenarios where media server discovery would be a requirement for streaming to be successful:

- Client is not configured with the address of any media servers.

- Client is configured with the address of one or more media servers, but the IMAP server is configured to only accept URLFETCH requests from specific media servers (for security or site policy reasons), and thus streaming would fail due to the media server not being able to retrieve the media from the IMAP server.

There is also a scenario where media server discovery would improve the security of the streaming mechanism, by avoiding the use of completely anonymous URLs. For example, the client could discover a media server address that was an authorized user of the IMAP server for streaming purposes, which would allow the client to generate a URL, which was secure in that it could *only* be accessed by an entity that is trusted by the IMAP server to retrieve content. The issue of trust in media servers is discussed more fully in Section 4.

This document describes using the IMAP METADATA [METADATA] extension, via the use of a server entry that provides the contact information for suitable media servers for use with the IMAP server. Media Server discovery is optional: clients are free to use pre-configured information about media servers, or to fall back to pre-configured information if they encounter IMAP servers that do not support either the METADATA extension or the proposed entry, or that do not provide a value for the entry.

A METADATA entry with the name of "/shared/mediaServers" is used to store the locations of suitable media servers known to the IMAP server. The entry is formatted according to the formalSyntax specified in Section 8. This consists of a tuple of a URI and optional "stream" string, where the URI is surrounded by <> symbols, the URI and "stream" are separated using a colon ":", and tuples are separated using a ";".

The "stream" string (c.f. the "stream" access identifier from [ACCESSID]) is used to identify media servers capable of connecting to the IMAP server as users authorized to retrieve URLs constructed using the "stream" access identifier. It indicates that the client MUST create the content URI using the "stream" access identifier. See Section 3.3 for a description of how the client should make use of the access identifier when generating IMAP URLs.)

Example values of the /shared/mediaServers METADATA entry (N.B. Any line-wrapping below is for the purpose of clarity):

"<sip:ivr@ms.example.net:5060>:stream;<sip:annc@
ms1.example.net:5060>;<sips:ivr@ms2.example.net:5061>"

"<sip:ivr@192.0.2.40:5060>;<sip:192.0.2.41:5060>;<sips:annc@
192.0.2.42:5060>:stream"

It should be noted that the URI specified in the ABNF (in Section 8) is generic, i.e., not restricted to SIP URIs; however, this document only specifies how to make use of SIP URIs. Additionally, the "userinfo" (known as the "service indicator" in RFC 4240 and RFC 4722) component of the URI is optional; if specified, it gives the client additional information about the media server capabilities. For example, a "userinfo" component of "annc" indicates that the media server supports RFC 4240, and "ivr" indicates support for RFC 4722. Section 3.4 further describes how clients should behave if the "userinfo" component is not present.

Clients SHOULD parse the value of the /shared/mediaServers entry, and contact a media server using one of the returned URIs. The servers are returned in order of preference as suggested by the server; however, it is left to the client to decide if a different order is more appropriate when selecting the media server(s) to contact, as well as the selection of alternates under failure conditions.

Administrators configuring the values of the /shared/mediaServers entry, who do not know the capabilities of the media servers being configured, SHOULD NOT include a "userinfo" component as part of the URI. In that case, the client will determine which service to use as specified in Section 3.4. Note that if a media server supports multiple services, a URI with the appropriate userinfo component SHOULD be configured for each service.

Note that even though the media server address can be discovered dynamically, it is assumed that the necessary security arrangements between the client and the media server already exist. For example, the media server could use SIP digest authentication to provide access only to authenticated clients; in this case, it is assumed the username and password have already been set up. Likewise, if the client wants to authenticate the media server using, e.g., TLS and certificates, it is assumed the necessary arrangements (trust anchors and so on) already exist. In some deployments, the clients and media servers may even be willing to rely on the security of the underlying network, and omit authentication between the client and the media server entirely. See Section 4 for more details.

3.3. Client Use of GENURLAUTH Command

The decision to make use of streaming services for a message part will usually be predicated on the content type of the message part. Using the capabilities of the IMAP FETCH command, clients determine the MIME [MIME] Content-Type of particular message parts, and based on local policies or heuristics, they decide whether streaming for that message part will be attempted.

Once the client has determined that a particular message part requires streaming, the client generates an IMAP URL that refers to the message part according to the method described in RFC 5092 [IMAPURL]. The client then begins the process of generating an URLAUTH URL by appending ";EXPIRE=<datetime>" and ";URLAUTH=<access>" to the initial URL.

The ";EXPIRE=<datetime>" parameter is optional; however, it SHOULD be used, since the use of anonymous URLAUTH-authorized URLs is a security risk (see Section 4), and it ensures that at some point in the future, permission to access that URL will cease. IMAP server implementors may choose to reject anonymous URLs that are considered insecure (for example, with an EXPIRE date too far in the future), as a matter of local security policy. To prevent this from causing interoperability problems, IMAP servers that implement this profile MUST NOT reject GENURLAUTH commands for anonymous URLs on the basis of the EXPIRE time, if that time is equal to, or less than, 1 hour in the future.

The <access> portion of the URLAUTH URL MUST be 'stream' (see [ACCESSID]) if an out-of-band mechanism or the media server discovery mechanism discussed in Section 3.2 specifies that the media server is an authorized user of the IMAP server for the purposes of retrieving content via URLFETCH. Without specific prior knowledge of such a configuration (either through the discovery mechanism described in this document, or by an out-of-band mechanism), the client SHOULD use the 'stream' access identifier, which will cause streaming to fail if the media server is not an authorized user of the IMAP server for the purposes of streaming.

However, if the client wishes to take the risk associated with generating a URL that can be used by any media server (see Section 4), it MAY use 'anonymous' as the <access> portion of the URLAUTH URL passed to the GENURLAUTH command. For example, the client may have been pre-configured with the address of media servers in the local administrative domain (thus implying a level of trust in those media servers), without knowing whether those media servers have a pre-existing trust relationship with the IMAP server to be used (which may well be in a different administrative domain). See Section 4 for a full discussion of the security issues.

The client uses the URL generated as a parameter to the GENURLAUTH command, using the INTERNAL authorization mechanism. The URL returned by a successful response to this command will then be passed to the media server. If no successful response to the GENURLAUTH command is received, then no further action will be possible with respect to streaming media to the client.

Examples:

```
C: a122 UID FETCH 24356 (BODYSTRUCTURE)
S: * 26 FETCH (BODYSTRUCTURE (("TEXT" "PLAIN"
S: ("CHARSET" "US-ASCII") NIL
S: NIL "7BIT" 1152 23)("VIDEO" "MPEG"
NIL NIL "BASE64" 655350)) UID 24356)
S: a122 OK FETCH completed.
C: a123 GENURLAUTH "imap://joe@example.com/INBOX/;uid=24356/;
section=1.2;expire=2006-12-19T16:39:57-08:00;
urlauth=anonymous" INTERNAL
S: * GENURLAUTH "imap://joe@example.com/INBOX/;uid=24356/;
section=1.2;expire=2006-12-19T16:39:57-08:00;
urlauth=anonymous:
internal:238234982398239898a9898998798b987s87920"
S: a123 OK GENURLAUTH completed


C: a122 UID FETCH 24359 (BODYSTRUCTURE)
S: * 27 FETCH (BODYSTRUCTURE (("TEXT" "PLAIN"
S: ("CHARSET" "US-ASCII") NIL
S: NIL "7BIT" 1152 23)("AUDIO" "G729"
NIL NIL "BASE64" 87256)) UID 24359)
S: a122 OK FETCH completed.
C: a123 GENURLAUTH "imap://joe@example.com/INBOX/;uid=24359/;
section=1.3;expire=2006-12-19T16:39:57-08:00;
urlauth=stream" INTERNAL
S: * GENURLAUTH "imap://joe@example.com/INBOX/;uid=24359/;
section=1.3;expire=2006-12-20T18:31:45-08:00;
urlauth=stream:
internal:0982309234092840923840928402934802394 82"
S: a123 OK GENURLAUTH completed
```

3.4. Client Determination of Media Server Capabilities

Once an authorized IMAP URL has been generated, it is up to the client to pass that URL to a suitable media server that is capable of retrieving the URL via IMAP, and streaming the content to the client using the RTP [RTP] protocol.

This section specifies the behavior of clients that have not determined (either statically through configuration, or dynamically through a discovery process as discussed in Section 3.2), the capabilities of the media server with respect to the services (i.e., RFC 4240 or 5022) supported by that media server. Clients that have determined those capabilities should use the mechanisms described in Sections 3.5 or 3.7, as appropriate.

If the client supports the MSCML IVR service, then it SHOULD attempt to contact the media server using the MSCML protocol by sending a SIP INVITE that has the service indicator "ivr".

Assuming the media server responds to the INVITE without error, the client can carry on using the MSCML IVR service as specified in Section 3.7. If the media server responds with an error indicating that the "ivr" service is not supported, then if the client supports it, the client SHOULD attempt to contact the media server using the announcement service, as described in Section 3.5.

The following example shows an example SIP INVITE using the "ivr" service indicator:

```
C: INVITE sip:ivr@ms2.example.com SIP/2.0
< SIP Header fields omitted for reasons of brevity >
```

3.5. Client Use of the Media Server Announcement Service

Assuming the client or media server does not support use of the MSCML protocol, the media server announcement service is used, as described in RFC 4240 [NETANN]. This service allows the client to send a SIP INVITE to a special username ('annc') at the media server (the "announcement" user), supplying the URL obtained as per Section 3.3.

The SIP INVITE is constructed as shown in the examples below; note that as per RFC 4240, the play parameter is mandatory and specifies the authorized IMAP URL to be played.

Examples of valid SIP INVITE URIs sent to the media server announcement service:

```
C: sip:annc@ms2.example.net;
play=imap:%2F%2Fjoe@example.com%2FINBOX%2F%3Buid%3D24356%2F%3Bsection
%3D1.2%3Bexpire%3D2006-12-19T16:39:57-08:00%3Burlauth%3Danonymous:
internal:238234982398239898a9898998798b987s87920
```

```
C: sip:annc@ms1.example.net;
play=imap:%2F%2Ffred@
example.com%2FINBOX%2F%3Buid%3D24359%2F%3Bsection
%3D1.3%3Bexpire%3D2006-12-20T18:31:45-08:00%3Burlauth%3Dstream:
internal:0982309234092840923840928402934802394 82
```

Notice that many of the characters that are used as parameters of the IMAP URI are escaped, as otherwise they would change the meaning of the enclosing SIP URI, by being regarded as SIP URI parameters instead of IMAP URL parameters.

If the client receives a 200 (OK) response, the media server has successfully retrieved the content from the IMAP server and the negotiated RTP stream will shortly begin.

There are many possible response codes; however, a response code of 404 received from the media server indicates that the content could not be found or could not be retrieved for some reason. For example, the media server may not support the use of IMAP URLs. At this point, there are several options to the client, such as using alternate media servers, or giving up in attempting to stream the required message part.

3.6. Media Negotiation and Transcoding

This document uses standards and protocols from two traditionally separate application areas: Mobile Email (primarily IMAP) and Internet Telephony/Streaming (e.g., SIP/RTP). Since the document primarily addresses enhancing the capabilities of mobile email, it is felt worthwhile to give some examples of simple SIP/SDP exchanges and to discuss capabilities such as media negotiation (using SDP) and media transcoding.

In the below example, the client contacts the media server using the SIP INVITE command to contact the announcement service (see Section 3.5), advertising support for a range of audio and video codecs (using SDP [SDP]), and in response the media server advertises only a set of audio codecs. This process is identical for the IVR service, except that the IVR service does not use the SIP Request-URI to indicate the content to be played; instead, this is carried in a subsequent SIP INFO request.

The client and server now know from the SDP session description advertised by both client and server that communication must be using the subset of audio codecs supported by both client and server (in the example SDP session description below, it is clear that the server does not support any video codecs). The media server may perform transcoding (i.e., converting between codecs) on the media received from the IMAP server in order to satisfy the codecs supported by the client. For example, the media server may downgrade the video retrieved from the IMAP server to the audio component only.

For clients using the announcement service, the media server MUST return an error to the INVITE if it cannot find a common codec between the client, server and media, or it cannot transcode to a suitable codec. Similarly, for clients using the MSCML IVR service, the media server MUST return a suitable error response to the <playcollect> request.

Example SIP INVITE and SDP Media Negotiation

```
C: INVITE sip:annc@ms2.example.com;
play=imap:%2F%2Fjoe@example.com%2FINBOX%2F%3Buid%3D24356%2F%3B
section%3D1.2%3Bexpire%3D2006-12-19T16:39:57-08:00%3Burlauth%3D
anonymous:internal:238234982398239898a9898998798b987s87920 SIP/2.0
C: From: UserA <sip:UAA@example.com>
C: To: NetAnn <sip:annc@ms2.example.com>
C: Call-ID: 8204589102@example.com
C: CSeq: 1 INVITE
C: Contact: <sip:UAA@192.0.2.40>
C: Content-Type: application/sdp
C: Content-Length: 481
C:
C: v=0
C: o=UserA 2890844526 2890844526 IN IP4 192.0.2.40
C: s=Session SDP
C: c=IN IP4 192.0.2.40
C: t=3034423619 0
C: m=audio 9224 RTP/AVP 0 8 3 98 101
C: a=alt:1 1 : 01BB7F04 6CBC7A28 192.0.2.40 9224
C: a=fmtp:101 0-15
C: a=rtpmap:98 ilbc/8000
C: a=rtpmap:101 telephone-event/8000
C: a=recvonly
C: m=video 9226 RTP/AVP 105 34 120
C: a=alt:1 1 : 01BCADB3 95DFFD80 192.0.2.40 9226
C: a=fmtp:105 profile=3;level=20
C: a=fmtp:34 CIF=2 QCIF=2 MAXBR=5120
C: a=rtpmap:105 h263-2000/90000
C: a=rtpmap:120 h263/90000
C: a=recvonly

S: SIP/2.0 200 OK
S: From: UserA <sip:UAA@example.com>
S: To: NetAnn <sip:annc@ms2.example.com>
S: Call-ID: 8204589102@example.com
S: CSeq: 1 INVITE
S: Contact: <sip:netann@192.0.2.41>
S: Content-Type: application/sdp
S: Content-Length: 317
S:
S: v=0
S: o=NetAnn 2890844527 2890844527 IN IP4 192.0.2.41
S: s=Session SDP
S: c=IN IP4 192.0.2.41
S: t=3034423619 0
S: m=audio 17684 RTP/AVP 0 8 3 18 98 101
```

```
S: a=rtpmap:0 PCMU/8000
S: a=rtpmap:8 PCMA/8000
S: a=rtpmap:3 GSM/8000
S: a=rtpmap:18 G729/8000
S: a=fmtp:18 annexb=no
S: a=rtpmap:98 iLBC/8000
S: a=rtpmap:101 telephone-event/8000
S: a=fmtp:101 0-16

C: ACK sip:netann@192.0.2.41 SIP/2.0
C: From: UserA <sip:UAA@example.com>
C: To: NetAnn <sip:annc@ms2.example.com>
C: Call-ID: 8204589102@example.com
C: CSeq: 1 ACK
C: Content-Length: 0
```

3.7. Client Use of the Media Server MSCML IVR Service

Once the client has determined that the media server supports the IVR service, it is up to the client to generate a suitable MSCML request to initiate streaming of the required media.

When using the IVR service, the initial SIP invite is used only to establish that the media server supports the MSCML IVR service, and to negotiate suitable media codecs. Once the initial SIP INVITE and response to that INVITE have been completed successfully, the client must generate a SIP INFO request with MSCML in the body of the request to initiate streaming.

The <playcollect> request is used, as this allows the use of dual tone multi-frequency (DTMF) digits to control playback of the media, such as fast-forward or rewind.

Since the <playcollect> request is used purely for its VCR-like capabilities, there is no need for the media server to perform DTMF collection. Therefore, the playcollect attributes "firstdigittimer", "interdigittimer", and "extradigittimer" SHOULD all be set to "0ms", which will have the effect of causing digit collection to cease immediately after the media has finished playing.

The "ffkey" and "rwkey" attributes of <playcollect> are used to control fast-forward and rewind behavior, with the "skipinterval" attribute being used to control the ’speed’ of these actions.

The <prompt> tag is used to specify the media to be played, and SHOULD have a single <audio> tag that gives the URL of the media, as per the Section 3.3. The audio-specific name of the tag is historical, as the tag can be used for video as well as audio

content. The "stoponerror" attribute SHOULD be set to "yes", so that meaningful error messages will be returned by the media server in the event of problems such as retrieving the media from the IMAP server.

An example SIP INFO request using the <playcollect> request is shown at the end of this section.

It should be noted that under normal (i.e., non-error) conditions, the response to the <playcollect> request is a SIP 200 (OK) response. The media server then streams the media, and only when the media has finished playing (naturally or due to a user request) does the media server send a <playcollect> response, which includes details of the media played, such as length and any digits collected.

The client may suspend playback of the media at any time by either sending the DTMF escape key (specified as an attribute to the <playcollect> request) or by sending a <stop> request to the media server in a SIP INFO request. Upon receipt of the request, the media server will acknowledge it, and then cease streaming of the media, followed by a SIP INFO request containing the <playcollect> response.

If the media server cannot play the media for any reason (for example, if it cannot retrieve the media from the IMAP server), streaming will not take place, and the <playcollect> response will be sent, usually with meaningful values in the <error_info> element.

The following gives an example dialog between a client and media server, including a rewind request, and termination of the playback by use of the escape key. Some elements of the SIP dialog such as full SIP header fields and SDP are omitted for reasons of brevity. (The protocol diagram in Section 3.9.2 shows the high-level message flow between all the components, including the IMAP server.)

```
C: INVITE sip:ivr@ms.example.com SIP/2.0
C: From: UserA <sip:UAA@example.com>
C: To: IVR <sip:ivr@ms.example.com>
C: Call-ID: 3298420296@example.com
C: CSeq: 1 INVITE
C: Contact: <sip:UAA@192.0.2.40>
C: Content-Type: application/sdp
C: Content-Length: XXX
C:
C: <SDP Here>

S: SIP/2.0 200 OK
S: From: UserA <sip:UAA@example.com>
S: To: IVR <sip:ivr@ms.example.com>
S: Call-ID: 3298420296@example.com
```

```
S: CSeq: 1 INVITE
S: Contact: <sip:ivr@192.0.2.41>
S: Content-Type: application/sdp
S: Content-Length: XXX
S:
S: <SDP Here>

C: ACK sip:ivr@ms.example.com SIP/2.0
C: From: UserA <sip:UAA@example.com>
C: To: IVR <sip:ivr@ms2.example.com>
C: Call-ID: 3298420296@example.com
C: CSeq: 1 ACK
C: Content-Length: 0

C: INFO sip:ivr@192.0.2.41 SIP/2.0
C: From: UserA <sip:UAA@example.com>
C: To: IVR <sip:ivr@ms.example.com>
C: Call-ID: 3298420296@example.com
C: CSeq: 2 INFO
C: Content-Type: application/mediaservercontrol+xml
C: Content-Length: 423
C:
C: <?xml version="1.0"?>
C: <MediaServerControl version="1.0">
C: <request>
C: <playcollect id="332985001"
C: firstdigittimer="0ms" interdigittimer="0ms" extradigittimer="0ms"
C: skipinterval="6s" ffkey="6" rwkey="4" escape="*">
C: <prompt stoponerror="yes"
C: locale="en_US" offset="0" gain="0" rate="0"
C: delay="0" duration="infinite" repeat="0">
C: <audio url="imap://joe@example.com/INBOX/;uid=24356/;section=1.2;
expire=2006-12-19T16:39:57-08:00;urlauth=anonymous:
internal:238234982398239898a9898998798b987s87920"/>
C: </prompt>
C: </playcollect>
C: </request>
C: </MediaServerControl>

S: SIP/2.0 200 OK
S: From: UserA <sip:UAA@example.com>
S: To: IVR <sip:ivr@ms.example.com>
S: Call-ID: 3298420296@example.com
S: CSeq: 2 INFO
S: Contact: <sip:ivr@192.0.2.41>
S: Content-Length: 0
```

```
   S: <Media server retrieves media from IMAP server and streams to
   client>

   C: <Client streams 6 key>

   S: <Media Server fast forwards media by 6 seconds>

   C: <Client streams * key>

   S: <Media Server stops streaming>

   S: INFO sip:UAA@192.0.2.40 SIP/2.0
   S: From: IVR <sip:ivr@ms.example.com>
   S: To: UserA <sip:UAA@example.com>
   S: Call-ID: 3298420296@example.com
   S: CSeq: 5 INFO
   S: Contact: <sip:ivr@192.0.2.41>
   S: Content-Type: application/mediaservercontrol+xml
   S: Content-Length: XXX
   S:
   S: <?xml version="1.0"?>
   S: <MediaServerControl version="1.0">
   S: <response id="332985001" request="playcollect" code="200"
   S: reason="escapekey" playduration="34s"
   S: playoffset="34s" digits="" />
   S: </MediaServerControl>

   C: SIP/2.0 200 OK
   C: From: IVR <sip:ivr@ms.example.com>
   C: To: UserA <sip:UAA@example.com>
   C: Call-ID: 3298420296@example.com
   C: CSeq: 5 INFO
   C: Content-Length: 0

   C: BYE sip:ivr@192.0.2.41 SIP/2.0
   C: From: UserA <sip:UAA@example.com>
   C: To: IVR <sip:ivr@ms.example.com>
   C: Call-ID: 3298420296@example.com
   C: CSeq: 6 BYE
   C: Content-Length: 0

   S: SIP/2.0 200 OK
   S: From: UserA <sip:UAA@example.com>
   S: To: IVR <sip:ivr@ms.example.com>
   S: Call-ID: 3298420296@example.com
   S: CSeq: 6 BYE
   S: Contact: <sip:ivr@192.0.2.41>
   S: Content-Length: 0
```

3.8. Media Server Use of IMAP Server

This section describes how the media server converts the IMAP URL received via the announcement or IVR service into suitable IMAP commands for retrieving the content.

The media server first connects to the IMAP server specified in the URL. Once connected, the media server SHOULD use TLS [TLS] to encrypt the communication path.

If the media server has a user identity on the IMAP server, the media server SHOULD authenticate itself to the IMAP server using the media server's user identity.

If the media server is not configured as an authorized user of the IMAP server, then the behavior specified in IMAP URL [IMAPURL] MUST be followed. That is, if the server advertises AUTH=ANONYMOUS IMAP capability, the media server MUST use the AUTHENTICATE command with the ANONYMOUS [ANONYMOUS] SASL mechanism. If SASL ANONYMOUS is not available, the username "anonymous" is used with the "LOGIN" command and the password is supplied as the Internet email address of the administrative contact for the media server.

Once authenticated, the media server issues the URLFETCH command, using the URL supplied in the 'play' parameter of the SIP INVITE (or audio tag of the MSCML). If the IMAP server does not advertise URLAUTH=BINARY in its post-authentication capability string, then the media server returns a suitable error code to the client.

The additional parameters to the URLFETCH command specified in (URLFETCH BINARY) [URLFETCH_BINARY] are used by the media server to tell the IMAP server to remove any transfer encoding and return the content type of the media (as content-type information is not contained in the IMAP URL).

A successful URLFETCH command will return the message part containing the media to be streamed. If the URLFETCH was unsuccessful, then the media server MUST return an appropriate error response to the client.

Assuming the content is retrieved successfully, the media server returns a 200 (OK) response code to the client. After an ACK is received, an RTP stream is delivered to the client using the parameters negotiated in the SDP.

If appropriate, the media server MAY choose to implement connection caching, in which case, connection and disconnection from the IMAP server are handled according to whatever algorithm the media server chooses. For example, the media server may know, a priori, that it

will always access the same IMAP server using the same login credentials with an access pattern that would benefit from connection caching, without unduly impacting server resources.

Examples:

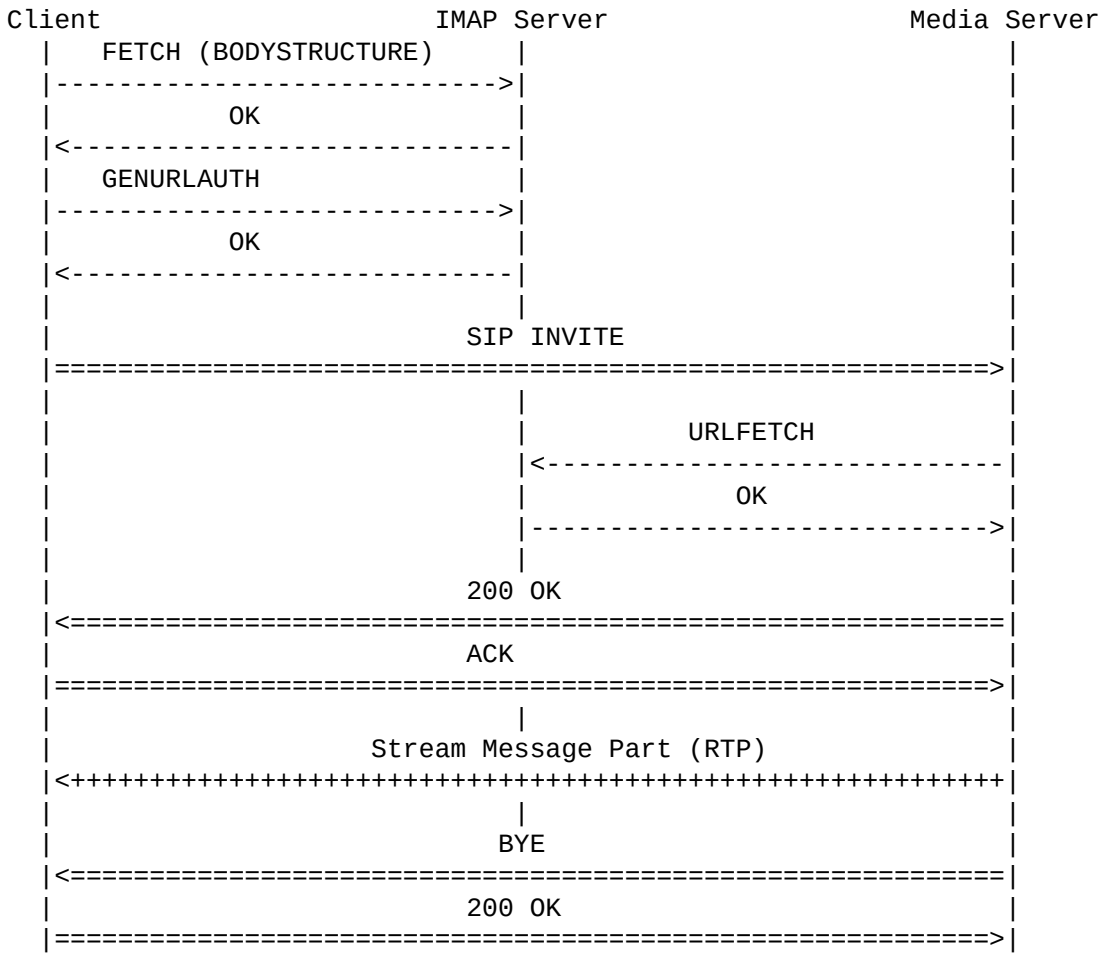
```
C: a001 LOGIN anonymous null
S: a001 OK LOGIN completed.
C: a002 URLFETCH
("imap://joe@example.com/INBOX/;uid=24356/;section=1.2;
expire=2006-12-19T16:39:57-08:00;urlauth=anonymous:
internal:238234982398239898a9898998798b987s87920" BODYPARTSTRUCTURE
BINARY)
S: * URLFETCH "imap://joe@example.com/INBOX/;uid=24356/;
section=1.2;expire=2006-12-19T16:39:57-08:00;urlauth=anonymous:
internal:238234982398239898a9898998798b987s87920"
(BODYPARTSTRUCTURE ("VIDEO" "MPEG" () NIL NIL "BINARY" 655350))
(BINARY ~{655350}
S: [ ~655350 octets of binary data, containing NUL octets ])
S: a002 OK URLFETCH completed.
C: a003 LOGOUT
S: a003 OK LOGOUT completed.
```

3.9. Protocol Diagrams

This section gives examples of using the mechanism described in the document to stream media from a media server to a client, fetching the content from an IMAP server. In all of the examples, the IMAP, SIP, and RTP protocols use the line styles "-", "=", and "+", respectively.

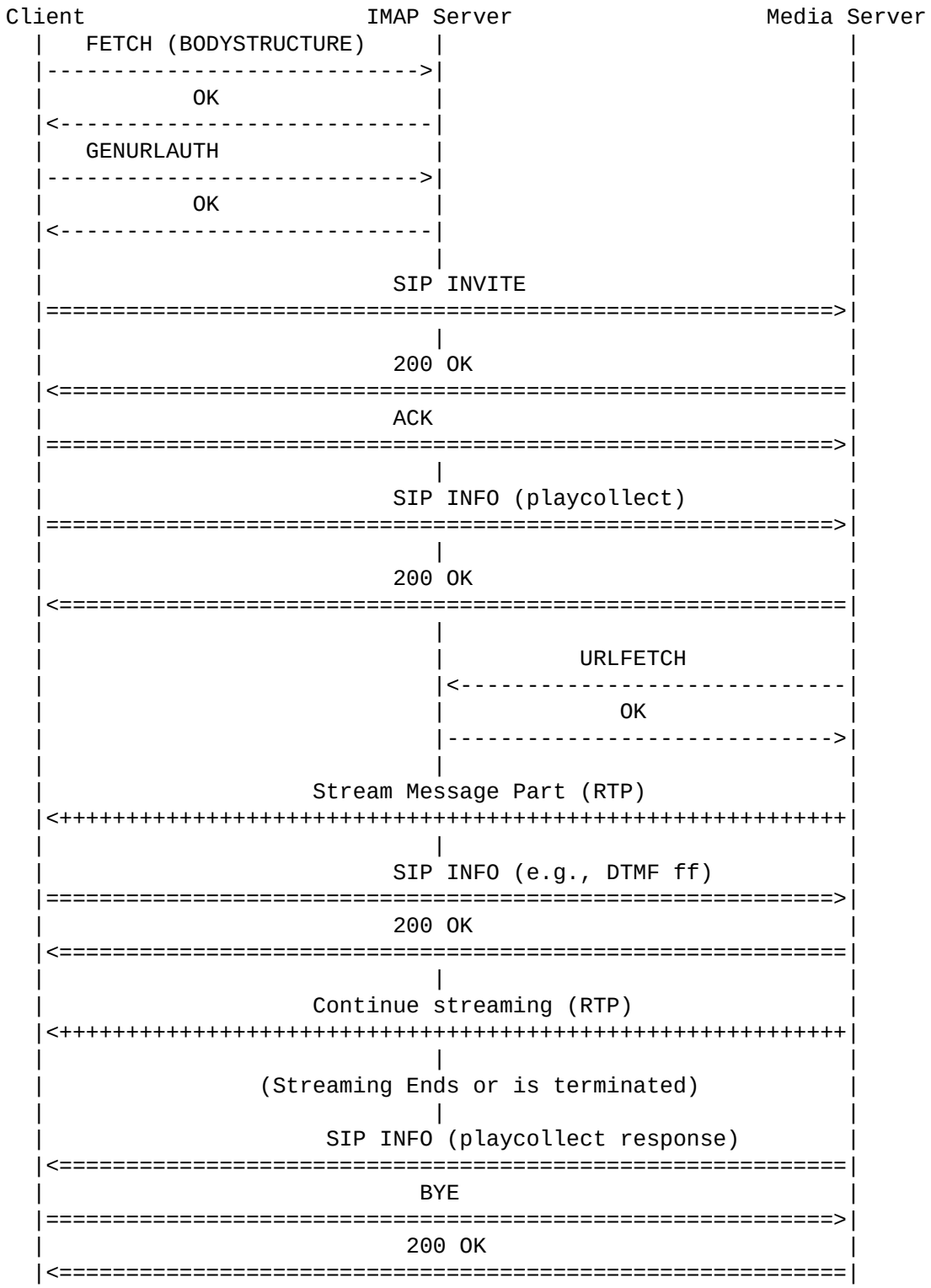
3.9.1. Announcement Service Protocol Diagram

The following diagram shows the protocol interactions between the email client, the IMAP server, and the media server when the client uses the announcement service.



3.9.2. IVR Service Protocol Diagram

The following diagram shows a simplified view of the protocol interactions between the email client, the IMAP server, and the media server when the client uses the MSCML IVR service.


Client
IMAP Server
Media Server
FETCH (BODYSTRUCTURE)
OK
GENURLAUTH
OK
SIP INVITE
200 OK
ACK
SIP INFO (playcollect)
200 OK
URLFETCH
OK
Stream Message Part (RTP)
SIP INFO (e.g., DTMF ff)
200 OK
Continue streaming (RTP)
(Streaming Ends or is terminated)
SIP INFO (playcollect response)
BYE
200 OK

4. Security Considerations

This document proposes the use of URLAUTH [URLAUTH] "pawn tickets", received over IMAP [IMAP], and transmitted over SIP [SIP], possibly within the MSCML payload of RFC 5022 [MSCML], in order to stream media received in messages. As such, the security considerations in all these documents apply to this specification.

In summary, as the authorized URLs may grant access to data, implementors of this specification need to consider the following with respect to the security implications of using IMAP URLs:

- Use of an anonymous pawn ticket grants access to any client of the IMAP server without requiring any authentication credentials. The security mechanisms referenced above (with the caveats specified below) SHOULD be used to prevent unauthorized access to the pawn ticket.

- Use of pawn tickets that contain the "stream" access identifier restricts access to the content to those entities that are authorized users of the IMAP server for the purposes of streaming retrieved content. Use of such pawn tickets is thus desirable and so implementors should consult Section 3.3, which describes when such pawn tickets should be used.

- If the announcement service is used to set up streaming, then RFC 4240 [NETANN] specifies that the pawn ticket is passed in the Request-URI, and so untrusted third parties may be able to intercept the pawn ticket. The SIP communication channel MAY be secured by using SIPS URIs [SIP], which would provide hop-by-hop TLS encryption.

- If the IVR service (RFC 5022 [MSCML]) is used to set up and control streaming, then MSCML is used to carry the pawn ticket in the body of the request, and so untrusted third parties may be able to intercept the pawn ticket. This MAY be secured by using SIPS URIs [SIP], which would provide hop-by-hop TLS encryption.

- Using SIPS URIs in the above situations protects the pawn ticket from third parties; however, it still allows proxies access to the pawn ticket, which could result in misuse by malicious proxies; see note below.

This document describes a mechanism that makes use of two separate servers to achieve the goal of streaming the content desired by the client. A major security implication of this is that the media server and IMAP server may well be located in separate administrative domains. This leads us to consider the security implications of a

three-way protocol exchange, and the potential trust model implicit in that tripartite relationship. The security implications of the individual protocols have already been referenced; therefore, this section describes the security considerations specific to the three-way data exchange, as follows:

- The client grants the media server full access to the potentially private media content specified by the IMAP URL. As a result, the client is responsible for verifying the authenticity of the media server to a degree it finds acceptable for the content (we can refer to this process as determining the "trust" that the client has in a particular media server). The security mechanisms provided by SIP [SIP] and RTP [RTP] may be used for this purpose, as well as out of band mechanisms such as pre-configuration.

- However, since the media server will retrieve content from an IMAP server on the user's behalf, the issue of security between the IMAP server and the media server also needs to be considered. A client has no way of determining (programatically at least) the security of the exchanges between the media server and the IMAP server. However, it can determine, using the "stream" token that is part of the media server discovery mechanism described in Section 3.2, that the media server has a pre-existing authentication relationship with the IMAP server for the purposes of retrieving content using IMAP URLs. The IMAP server administrator may put prerequisites on media server administrator before this relationship can be established, for example, to guarantee the security of the communication between the media server and the IMAP server.

- The above two security considerations will influence the decision the client makes with regards to generation of the pawn ticket that is subsequently passed to the media server. This document mandates the use of URLs protected with the "stream" access identifier where the client knows in advance that the "stream" authentication relationship between media server and IMAP server exists. However, it does allow the use of anonymous pawn tickets where the possibility exists that use of "stream" would cause streaming to fail.

- There exists the possibility of several types of attack by a malicious media server, SIP proxy, or other network elements even against pawn tickets protected with the "stream" access identifier. All of these attacks allow access to the RTP stream, if not the original content. These attacks include:

* The client contacts a malicious media server, MS1, that then proxies the streaming request to a second media server, MS2, that it has determined or guessed to have "stream" authorization credentials with the IMAP server specified in the pawn ticket. The media server can then redirect the streamed RTP traffic elsewhere.

* Any proxy on the path between the client and the media server has access to the client's message in cleartext. In this case, a malicious proxy could perform a man-in-the-middle attack and could change the message to redirect RTP traffic elsewhere.

* Any network element that is able to "see" the traffic between the client and the media server (or between any two proxies) can capture the pawn ticket, and then reissue a request using that pawn ticket to the same media server. Again the streamed traffic can be redirected to any desired location.

Media servers handling streaming requests will be making use of pawn-ticket URLs for the period of time required to process the streaming request, after which the URL will be forgotten. However, media servers may log the URLs received from clients, in which case, the private data contained in the IMAP server could be accessed by a malicious or curious media server administrator. Even URLs protected with EXPIRE may be accessed within the period of expiry. Therefore, media servers SHOULD remove or anonymize the internal portion of the IMAP URL when logging that URL.

Additionally, many of the security considerations in the Message Submission BURL Extension apply to this document, particularly around the use of pawn tickets and prearranged trust relationships such as those described above.

Message parts that are encrypted using mechanisms such as S/MIME [SMIME] are designed to prevent third parties from accessing the data, thus media servers will not be able to fulfill streaming requests for messages parts that are encrypted.

5. IANA Considerations

IANA has registered the following [METADATA] server entry to be used for media server discovery, using the [METADATA] registry.

To: iana@iana.org

Subject: IMAP METADATA Entry Registration

Type: Server

Name: /shared/mediaServers

Description: Defines a set of URIs containing the locations of
suitable media servers for streaming multimedia content

Content-type: text/plain; charset=utf-8

Contact: neil.cook@noware.co.uk

6. Digital Rights Management (DRM) Issues

This document does not specify any Digital Rights Management (DRM) mechanisms for controlling access to and copying of the media to be streamed. This is intentional. A reference to a piece of media content is created using the URLAUTH [URLAUTH] command; thus, any DRM required should be implemented within the media itself, as implementing checks within URLAUTH could affect any use of the URLAUTH command, such as the BURL [BURL] command for message submission.

The use of URLAUTH in this specification is believed to be pursuant with, and used only for, the execution of those rights to be expected when media is sent via traditional internet messaging, and causes no duplication of media content that is not essentially provided by the action of sending the message. In other words, the use of the content for downloading and viewing *is* implicitly granted by the sender of the message, in as much as the sender has the right to grant such rights.

The document author believes that if DRM is a requirement for Internet messaging, then a suitable DRM mechanism should be created. How such a mechanism would work is outside the scope of this document.

7. Deployment Considerations

This document assumes an Internet deployment where there are no network restrictions between the different components. Specifically, it does not address issues that can occur when network policies restrict the communication between different components, especially between the media server and the IMAP server, and between the client the media server. In particular, RFC 5022 states that "It is

unlikely, but not prohibited, for end-user SIP UACs to have a direct signaling relationship with a media server". This caveat makes it likely that firewalls and other network security mechanisms will be configured to block direct end-user access to media servers.

In order for either of the streaming mechanisms described in this document to work, local administrators MUST relax firewall policies such that appropriate SIP UACs (user agent clients) running on mobile devices are permitted to access the media servers directly using the SIP protocol. The detail of how the restrictions are relaxed (for example, only allowing clients connecting from the network space owned/maintained by the operator of the media server) is a matter of local policy, and so is outside the scope of this document.

8. Formal Syntax

The following syntax specification for the mediaServers METADATA entry value uses the Augmented Backus-Naur Form (ABNF) notation as specified in RFC 5234 [ABNF] and the "absolute-URI" definition from RFC 3986 [RFC3986].

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper or lower case characters to define token strings is for editorial clarity only. Implementations MUST accept these strings in a case-insensitive fashion.

```
media-servers = ms-tuple *(";" ms-tuple)
ms-tuple      = "<" absolute-URI ">" [":" "stream"]
```

9. Contributors

Eric Burger (eburger@standardstrack.com) provided the initial inspiration for this document, along with advice and support on aspects of the media server IVR and announcement services, as well as help with the IETF process.

Many people made helpful comments on the document, including Alexey Melnikov, Dave Cridland, Martijn Koster, and a variety of folks in the LEMONADE and SIPPING WGs.

10. References

10.1. Normative References

[ABNF] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[ACCESSID] Cook, N., "Internet Message Access Protocol (IMAP) - URL Access Identifier Extension", RFC 5593, June 2009.

[ANONYMOUS] Zeilenga, K., "Anonymous Simple Authentication and Security Layer (SASL) Mechanism", RFC 4505, June 2006.

[IMAP] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

[IMAPURL] Melnikov, A., Ed. and C. Newman, "IMAP URL Scheme", RFC 5092, October 2007.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[METADATA] Daboo, C., "The IMAP METADATA Extension", RFC 5464, February 2009.

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

Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

Freed, N. and J. Klensin, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", BCP 13, RFC 4289, December 2005.

Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996.

[MSCML] Van Dyke, J., Burger, E., Ed., and A. Spitzer, "Media Server Control Markup Language (MSCML) and Protocol", RFC 5022, September 2007.

[NETANN] Burger, E., Van Dyke, J., and A. Spitzer, "Basic Network Media Services with SIP", RFC 4240, December 2005.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RTP] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[SDP] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[SIP] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[TLS] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[URLAUTH] Crispin, M., "Internet Message Access Protocol (IMAP) - URLAUTH Extension", RFC 4467, May 2006.

[URLFETCH_BINARY] Cridland, D., "Extended URLFETCH for Binary and Converted Parts", RFC 5524, May 2009.

10.2. Informative References

[BURL] Newman, C., "Message Submission BURL Extension", RFC 4468, May 2006.

[SMIME] Ramsdell, B., Ed., ""Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification"", RFC 3851, July 2004.

Author's Address

Neil L Cook
Cloudmark

EMail: neil.cook@noware.co.uk