INTERNET-DRAFT
<draft-aboba-rtp-http-01.txt>

Bernard Aboba
Microsoft Corporation

Payload Format for HTTP Encoding in RTP

1. Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress.''

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt'' listing contained in the Internet-Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

The distribution of this memo is unlimited. It is filed as <draft-aboba-rtp-http-01.txt>, and expires May 10, 1997. Please send comments to the authors.

2. Abstract

This document specifies a payload format for use in encoding HTTP within RTP. This payload format can be used for unreliable multicasting of resources such as Web pages, stock tickers, etc. As with other RTP applications, receiver feedback and group membership information is provided via RTCP.

3. Introduction

3.1. Purpose

Considerable interest has recently arisen in the multicasting of resources residing on HTTP servers. Many of these uses can be satisfied by ureliable transmission.

This document specifies a payload format suitable for encoding of HTTP within RTP. It is expected that this payload format will prove useful for unreliable multicasting of resources, either on a one-shot basis, or in a data carousel-style format, where resources are continually re-multicast. This specification is not expected to be used with uni-

cast, since unicast applications will instead use HTTP over TCP.

3.2. Overview

Multicasting of HTTP payloads is useful in a variety of applications, and as a result, several approaches have been taken. One of these is to put a single resource within a payload; another is to pack multiple resources in a payload. If multiple resources are placed within a single payload, this can be accomplished either via encapsulation or via a packing method such as MHTML, defined in [5].

This document specifies encapsulation of a single resource per payload. As defined in this specification, the HTTP payload consists of a preamble header, a MIME-like header, and entity-body content. Information on the resource being transmitted (such as the URI and the offset) is placed in the preamble header so as to avoid requiring receivers to parse MIME-headers in the HTTP payload in order to determine what portions of a resource have been received. As a result, this specification does not propose any new MIME headers, and any MIME headers allowable in an HTTP GET response may be enclosed in the payload.

This simplifies construction of unicast-multicast proxies, since the MIME-like header in the payload can be identical to that returned in the response to an HTTP-GET request. Proxies may therefore make a request for a URL, stuff the response into a payload, and multicast it. This is an efficient process since the proxy does not need to parse the response or construct an MHTML package.

4. RTP header

Rather than defining a new profile, this specification assumes that HTTP payloads will be transmitted using the RTP profile defined in [3], that is the RTP profile for audio and video conferencing. Additional required parameters are accommodated via definition of an HTTP payload format. As a result, there is no need for header extensions, SDES private extensions, new sender or receiver report fields, new RTCP packet types, or changes in the reporting interval constants.

Nevertheless, some clarifications are useful in describing how HTTP payload encoding is to be used with the profile defined in [3].

4.1. Extension bit

Since this specification does not define header extensions for encoding of HTTP payloads, the extension bit will typically be cleared.

4.2. CSRC count

Since HTTP payloads do not require mixing, there is no need for a contributing source field. As a result, the CSRC count field is set to zero.

4.3. Marker Bit

For use with HTTP payloads, a zero marker bit is used to indicate the last packet for a resource. The first packet is distinguished by inclusion of a MIME -like header after the preamble header.

4.4. Payload types

A dynamic payload type is used. As a result, there is no need to assign a static payload type.

4.5. SSRC

Unlike with A/V payloads, a sender may use a single synchronization source for transmission of multiple HTTP payload streams. Thus a JPEG file may be transmitted with the same SSRC as an HTML file. This does not present a problem since the timestamp is used to uniquely identify resource streams.

4.6. Timestamp

Since a single synchronization source is used for transmission of multiple resources, an additional parameter is needed to identify packets within the same resource. The URI was not felt to be sufficient for this purpose, since a given resource may be multicast in data-carousel style, changing with each transmission.

As a result, the RTP timestamp field is used for this purpose. For use with HTTP payloads, the timestamp is constant for all packets that form a single transmission of a resource, and represents the time at which the sender began to transmit the resource.

Due to out of order delivery it is possible that packets from one resource will be intermingled with packets from another resource, sent to the same group and port. In this case the combination of the SSRC and timestamp can be used to demultiplex the resource stream.

5. HTTP Payload format

HTTP payloads consist of a preamble header, a MIME-like header, and entity-body content.

The purpose of the preamble header is identify the resource being sent, and to allow late joining receivers to calculate which portion of the resource they have missed. Depending on the application,

resources with missing portions will either be discarded or a repair request will be made. Since this specification is primarily intended for use in situations where a local repair may not be timely (such as with a frequently updated stock ticker), or where the resource will be re-multicast, it is not clear that an RTCP "NAK" packet is needed.

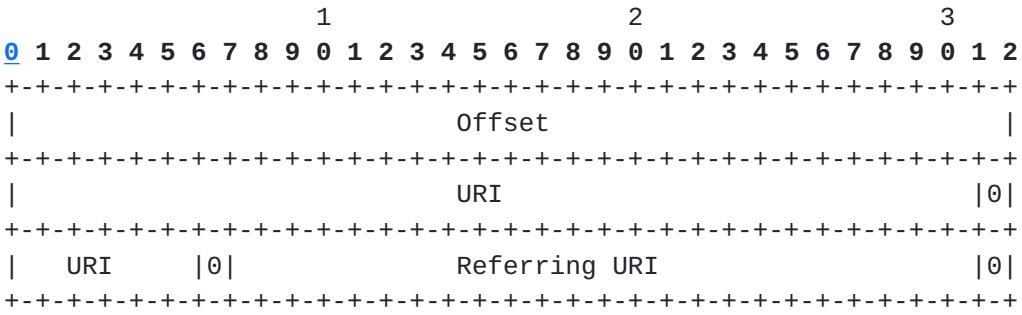
Therefore repairs are expected to be executed via a partial HTTP GET request.

While MIME-like headers could be used for resource identification (to provide information such as the URI, referring URI, content length and content range), prepending of a binary header reduces the amount of parsing required to get at this critical information. The use of a preamble header has the additional benefit of not requiring MIME-like headers to be present other than in the first packet. Thus, subsequent packets will typically contain only the preamble header and body content.

Thus, rather than adding MIME-like headers, it is intended that senders will simply retransmit the MIME-like header obtained in the HTTP GET response. Similarly, it is expected that the entity-body content will be retransmitted without modification.

5.1. Preamble header

The preamble header is comprised of a 4 octet fixed portion, and two variable-length strings.



5.1.1. Offset

The offset field is 32-bits long. For a data packet, it represents the octet offset within the resource identified by the RTP timestamp. Although the offset field will typically increase with increasing sequence numbers, this need not always be the case, since portions of a resource may be transmitted out of order. Receivers should therefore be prepared to handle packet ordering based on the offset rather than the sequence number.

While the offset information could have also been provided using the Content-range: HTTP response header, this would have required insertion of a MIME-like header within each packet, and parsing of this header by the receiver. It is felt that the offset mechanism is more

efficient.

5.1.2. URI

The URI field is a null-terminated string, representing the URI of the resource being transmitted. While it begins on a 32-bit boundary, it is not padded to such a boundary.

Given the expected size of the URI and referring URI fields, the variable-length strings are expected to comprise the vast majority of the preamble header. Given that each of the string fields may be 40 octets or larger in length, when the 28 octets of IP/UDP header and 12 octets of RTP header are added, the result may be 140 octets or more of overhead. This amount of overhead would be unacceptable if it were present in every packet. It is therefore appropriate to ask whether the URI and referring URI fields are required for inclusion within every packet.

The URI, offset, and packet length uniquely identify the portion of the resource being transmitted. Since receivers may join late, or miss portions of the transmission, receivers must be able to quickly bind a timestamp to a URI so that the incoming resource and missing portions can be identified.

However, it is believed that this goal can be accomplished by placing the URI within the first packet of a series, and then only within every subsequent nth packet. This results in a substantial reduction in overhead. For the purposes of this specification, it is suggested that n=4.

For packets in which the URI is not included, a single null octet is used to indicate the missing field.

5.1.3. Referring URI

The referring URI field is a null-terminated string, representing the URI of the resource referring to the resource being transmitted. It begins immediately after the null signifying the end of the URI field, and is not padded to a 32-bit boundary.

The referring URI is used to identify the resource referring to the resource being transmitted. This is used by late joining receivers wishing to retrieve the context of the current transmission. Typically this is done via an HTTP GET request. However, in the case of data-carousel transmission, this is not necessary, since the referring resource will be re-transmitted at a later time.

As with the URI, the referring URI should not be transmitted within every packet. Instead, it should be placed within the first packet of a series, and then transmitted every 2n packets.

For packets in which the referring URI is not included, a single null octet is used to indicate the missing field.

6. Resource length

The resource length is not included in the preamble header. Typically, this information is included within the first packet via the Content-length: header. Although it is possible that the first and/or final packet will be lost, we do not believe that the resource length justifies inclusion within the preamble header. This is because the receiver need not know the total length of the resource to make a partial GET request for the remainder of the resource.

Note that it is possible that the final packets of a resource are lost. In this case, the receiver will note that it has not yet received a packet with the marker bit indicating completion of the resource transmission, after waiting a suitable interval after reception of the last packet. This interval is determined by the receiver's estimate of his RTT to the sender. After this interval has expired, the receiver will either wait for the re-multicasting of the resource, or will attempt to retrieve the missing portion via a partial HTTP GET.

7. Layered Data Carousels

Reference [4] discusses use of RTP in layered multicasting. Layered multicasting provides for receiver-driven rate adaptation. While this was originally proposed for use in transmission of audio and video, it can also be applied to data carousel style transmissions. In this application, each of the layers transmits the same resource, beginning at a different offset. Receivers with sufficient bandwidth may then subscribe to multiple groups, and receive the resource more quickly.

Reference [4] proposes guidelines for group address and port allocation, as well as modifications to RTP semantics suitable for allocation of SSRC identifiers across layered streams. SDES packets are then sent only on the lowest layer. It is expected that these modifications, once available, should be applicable to transmission of layered HTTP payloads.

8. Non-RTP means

In addition to information transmitted within the RTP encoding, it is expected that receivers will make use of session information transmitted by non-RTP means.

For example, the existence of a data-carousel style session is expected to be determined via SDP, transmitted in one of its encapsulations, such as SAP. The SDP announcement will provide information on the bandwidth allocated to the session, as well as the group address

(or in the case of a multi-group session, addresses) allocated to the session.

The SDP announcement will also be expected to indicate whether the data carousel transmission provides for re-multicast of the same

resource (in which case receivers can make partial GET requests for the missing portion), or whether it provides continually updated information (in which case receivers missing a portion of the resource should make a GET request for the entire resource).

9. Acknowledgements

Thanks to Peter Parnes of the University of Lulea, and Thomas Pfenning of Microsoft for useful discussions of this problem space.

10. References

[1] R. Braden. "Requirements for Internet hosts - application and support." STD 3, RFC 1123, IETF, October 1989.

[2] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. "RTP: A Transport Protocol for Real-Time Applications." RFC 1889, GMD Fokus, January 1996.

[3] H. Schulzrinne. "RTP Profile for Audio and Video Conferences with Minimal Control." RFC 1890, GMD Fokus, January 1996.

[4] M. F. Speer, S. McCanne. "RTP Usage with Layered Multimedia Streams." draft-speer-avt-layered-video-01.txt, Sun Microsystems, LBNL, June 1996.

[5] J. Palme, A. Hopmann. "MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML)." draft-ietf-mhtml-spec-03.txt, Stockholm University/KTH, ResNova Software, August, 1996.

[6] E. Levinson. "The MIME Multipart/Related Content-type." draft-ietf-mhtml-related-00.txt, Xison, May, 1996.

[7] J. Palme. "Sending HTML in E-mail, an informational supplement." draft-ietf-mhtml-info-03.txt, Stockholm University/KTH, August, 1996.

11. Authors' Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Phone: 206-936-6605
EMail: bernarda@microsoft.com