HTTP Multipart Batched Request Format
draft-snell-http-batch-01

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on December 14, 2009.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

This document specifies a format for packaging multiple, independent HTTP requests into a single multipart payload.
Table of Contents

1. Introduction ..................................................... 3
2. The Multipart/Http Content Type ................................. 3
3. IANA Considerations .............................................. 6
   3.1. The ‘Multipart/Batch’ Content-Type ................. 6
4. Security Considerations ......................................... 6
5. Normative References ........................................... 7
Author’s Address .................................................. 7
1. Introduction

This specification defines a format for packaging multiple, independent HTTP requests into a single multipart MIME payload. The intent is to provide applications with a method of grouping sets of individual HTTP requests for processing as a unit. This approach is designed to work around a number of limitations that currently exist in HTTP pipelining, such as the inability to pipeline a mix of idempotent and non-idempotent requests. In addition, it addresses the inability of a server to determine the completeness and optimum order of processing for a logical collection of requests before attempting to process any of the individual requests. Note that batching HTTP requests using the mechanism defined here, is not intended to replace pipelining.

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in [RFC2119].

2. The Multipart/Http Content Type

The multipart/http media type is a new subtype of MIME multipart message in which the body parts are either HTTP request or response messages. A multipart/http message MUST NOT contain a mixture of both request and response messages.

In addition to the boundary parameter required for all Multipart message types, the multipart/http type has two additional optional parameters: "msgtype" and "version".

The "msgtype" parameter value specifies either "request" or "response" and indicates the type of the enclosed messages.

The "version" parameter value specifies the HTTP Version (e.g. "1.1") of the enclosed messages. If not present, the version can be determined by examining each individual request part. All of the enclosed messages MUST be of the same HTTP Version.

The following illustrates an example multipart/http message containing two HTTP Request messages.
A multipart/http message containing HTTP request messages is called a Batch Request. A multipart/http message containing HTTP response messages is called a Batch Response.

A client application creates an Batch Request message with one or more individual HTTP requests and transmits that to the server as the payload of another HTTP request message as illustrated in Listing 1. Each body part MUST specify a Content-ID header specifying a reference identifier for the individual request message.

The server receives the Batch Request and processes each body part in any order the server determines to be appropriate.
Upon processing the Batch Request, the server will generate a Batch Response message containing one HTTP response message for every message in the request. Each part of Batch Response MUST contain a Content-ID header specifying the value of the Content-ID of the matching request. Client applications MUST NOT assume that the ordering of responses in the Batch Response matches the ordering of requests in the Batch Request and MUST use the Content-ID header values to correlate request and response messages.

A server need not wait until all batched requests have been processed to create and begin sending the Batch Response back to the client.

The example below contains a Batch Response for the Batch Request in Listing 1.

```
HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 20:00:00 GMT
Content-Length: 291
Content-Type: multipart/http;version=1.1
  ;msgtype=request;boundary=batch
Mime-Version: 1.0
--batch
Content-ID: <226e35d0-34fa-11de-b418-0800200c9a66@example.org>

HTTP/1.1 415 Unsupported Media Type
Date: Wed, 29 Apr 2009 20:00:00 GMT
--batch
Content-ID: <22fe12d0-ef12-e1de-c456-0834212c9a88@example.org>

HTTP/1.1 401 Unauthorized
Date: Wed, 29 Apr 2009 20:00:00 GMT
--batch
Content-ID: <df536860-34f9-11de-b418-0800200c9a66@example.org>

HTTP/1.1 204 OK
Date: Wed, 29 Apr 2009 20:00:00 GMT
--batch--
```

A server MUST determine the status of an HTTP request containing a Batch Request based solely on its ability to successfully process the Batch Request as a whole and not on the status of each individual contained message. For instance, if the server is capable of understanding and processing the Batched Request, even if some or all of the contained requests fail individually, the server SHOULD respond with a status of either 200 OK or 202 Accepted. The 201 Created, 204 No Content, 205 Reset Content, and 206 Partial Content responses are not appropriate responses for a successfully processed
batch request.

While it is possible for Batch Request and Batch Response messages to be transferred using Chunked Transfer Coding, the individual HTTP request and response messages contained within MUST NOT use Chunked Transfer Coding.

Batch Responses are not cacheable. Individual HTTP response messages contained within the Batch Response might be cacheable.

3. IANA Considerations

3.1. The ‘Multipart/Batch’ Content-Type

| Media Type Name: multipart |
| Media Subtype Name: http |
| Required Parameters: |
| boundary: The standard MIME boundary parameter |
| Optional Parameters: |
| version: The HTTP-Version number of the enclosed messages (e.g., "1.1") |
| msgtype: The message type -- "request" or "response" |

Encoding considerations:

HTTP messages enclosed by this type are in "binary" format

Security considerations: Depends on the batched content-type

Published specification: This document

Contact: James M Snell
jasnell@us.ibm.com
877-511-5082

4. Security Considerations

It is possible that a malicious client could attempt to bypass security policies by batching requests that otherwise would have been blocked by protective firewalls, proxies and server configurations. Applications supporting batched requests must be prepared to deal with such requests and should be implemented so that security policies are properly enforced on batched requests.

TODO: Discussion about how each batched request must be processed separately when it comes to applying security rules. Each has it’s own authentication and authorization. A batched request must not inherit the authentication/authorization of the containing Multipart Batch Request or the other requests in the batch. Just because a user is authorized to submit a batch of requests, that doesn’t mean
they are authorized to submit any of the individually batched requests.

5. Normative References


Author’s Address

James M Snell
IBM
Hanford, CA  93230

Phone:
Email: jasnell@us.ibm.com
URI:  http://www.ibm.com