Handing Large User Datagram Protocol (UDP) Responses in the Session
Initiation Protocol (SIP)
draft-gurbani-sip-large-udp-response-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any
applicable patent or other IPR claims of which he or she is aware
have been or will be disclosed, and any of which he or she becomes
aware will be disclosed, in accordance with Section 6 of 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

This Internet-Draft will expire on April 6, 2007.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

The Session Initiation Protocol (SIP) mandates a maximum size for any
request transmitted over UDP. This maximum is set to the lesser of
1300 bytes or the path maximum transmission unit (MTU) size minus 200
bytes. If the size of the request exceeds this maximum, SIP requires
implementations to switch the downstream transport to be a congestion
controlled transport. However, when sending a response, a SIP implementation cannot choose the transport; it must use the transport specified by the Via. This document discusses the problems large responses can cause on UDP, and proposes an update to SIP to help diagnose and avoid those problems.

Table of Contents

1. Terminology ................................................. 3
2. Problem Statement ........................................... 3
3. Solution ..................................................... 4
   3.1 UAS Considerations ........................................ 4
   3.2 UAC Considerations ........................................ 4
   3.3 Proxy Considerations ...................................... 4
4. Alternatives Evaluated ....................................... 5
   4.1 Error response to force transport change ................. 5
      4.1.1 UAS Considerations ................................... 5
      4.1.2 UAC Considerations ................................... 5
      4.1.3 Proxy Considerations ................................ 5
   4.2 Use message/sipfrag ....................................... 6
   4.3 Request in the backwards direction ....................... 6
5. Security Considerations .................................... 6
6. IANA Considerations ........................................ 6
7. Acknowledgments ............................................. 6
8. References .................................................. 7
   8.1 Normative References ..................................... 7
   8.2 Informative References ................................... 7
Authors’ Addresses ............................................. 7
Intellectual Property and Copyright Statements ............ 9
1. Terminology

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 [1].

2. Problem Statement

SIP has been defined to work over many transports, congestion controlled and otherwise. RFC3261 [2] contains normative text on how requests that are larger than a defined maximum cause a switch to using a congestion controlled transport. This maximum is the path MTU minus 200 bytes. If the path MTU is not known, SIP requires that it be assumed to be no larger than 1500 bytes.

When a transport lacking congestion control (like UDP) is used in SIP, a sending host will fragment the UDP packet into its respective IP datagrams if the size of the UDP packet exceeds the path MTU. Since reliability in the form of retransmitting lost IP fragments is not built into UDP as it is in TCP, a missing (or late arrival) IP fragment results in the entire UDP packet being discarded. Some SIP implementations and network address translation (NAT) devices do not handle fragmentation gracefully, dropped UDP datagrams can be quite likely. Reliability over UDP is built into SIP by application-level retransmissions, but under some conditions retransmissions do more harm than good to the network as a whole [5].

RFC 3261 [2] instructs implementations to use a congestion controlled transport whenever the size of a request to be sent downstream approaches a maximum allowed datagram size. While this works for requests, there is nothing equivalent in RFC3261 for responses. There are legitimate cases in SIP where the size of the response far exceeds a single datagram. Since SIP routes responses over the same transport as its corresponding request, an intermediary that received a request over UDP and sent it downstream over TCP is forced to send a large response over UDP to its upstream neighbor. If the UDP datagram is fragmented, the upstream system may not be able to reassemble the UDP datagram to successfully receive the SIP message, resulting in a request timeout.

Note that UDP fragmentation and reassembly works unless there is a lossy link or there is a NAT between the communicating endpoints. However, since in any deployment there is no deterministic way to minimize the chance of traversal over a lossy link or through a NAT, a general purpose solution to the large UDP response problem is an attractive option. This document proposes some normative changes to SIP to improve the diagnosis of these problems.
3. Solution

There may be multiple solutions to the problem identified above. Some alternatives the authors considered are discussed in Section 4 below. This section describes the one the authors consider most promising.

There is nothing that can be done to ensure that a UDP-fragmented response will be reliably received, so the objective of these proposed changes is to diagnose the problem when it does occur. To accomplish this, it is necessary to provide a response that is small enough to traverse the UDP hop(s) that alerts the requestor (and any other systems in the path) that a large response is being sent. The proposal defines two new provisional response codes:

140 Large Request Transport Change: This response is sent by a proxy whenever it is changing to a congestion-controlled transport because the request size is over the maximum allowed as mandated by RFC 3261 [2] section 18.1.1.

141 Sending Large Response: This response is sent by a UAS immediately before sending any final response whose size exceeds the default maximum of 1500 bytes if the response path includes one or more UDP hops.

The changes described below are partially (depending on request type) in conflict with [4]. This conflict could be resolved in more than one way. The authors solicit discussion of this issue, including the tradeoffs between the problems described in [6] and the problem of loosing large responses on UDP paths.

3.1 UAS Considerations

A UAS that needs to send a response that is larger than the default maximum SIP message size (1500 bytes) MUST inspect the chain of Via headers in the request. If any of the Via header entries includes a UDP hop, then the server MUST send a 141 Sending Large Response provisional response. The 141 response MUST include a Warning header identifying the UAS, and MUST NOT have any body.

3.2 UAC Considerations

A client that receives a 140 or 141 response to any request that subsequently times out MAY automatically retry that request using a congestion-controlled transport.

3.3 Proxy Considerations

When a proxy has receives a request whose topmost Via specifies UDP,
and at least one forwarded copy of that request is sent on a congestion controlled transport because it exceeds the default maximum size of 1500 bytes, then the proxy MUST send exactly one 140 Large Response Transport Change provisional response to that request.

Note that timing of this 140 response is not required to have any particular relationship to that of forwarding the request; it may be sent before or after the request is forwarded.

4. Alternatives Evaluated

Three alternatives were considered during the evaluation of possible solutions. These were:

4.1 Error response to force transport change

Introduce a new final response ‘489 - Use Congestion-controlled Transport’. If a SIP element would otherwise send any final response, but the size of that response exceeds the maximum SIP datagram size and an examination of the Vias shows that the response will traverse at least one UDP hop, then send an error response to force a change of transports.

Since the 489 response would have no body, it is likely to be small enough when it reaches the UDP hop to be small enough to not cause fragmentation, and so is likely to reach the requestor.

4.1.1 UAS Considerations

A server that wants to send a response over UDP that is larger than the high watermark MUST inspect the chain of Via headers in the request. If any of the Via header entries includes a UDP hop, then the server MUST respond with a 489 response. The body of the 489 MUST be empty.

4.1.2 UAC Considerations

A client that receives a 489 response to a request MAY retry that request using a congestion-controlled transport.

4.1.3 Proxy Considerations

No different than Section 3.2 for the client transaction state machine and Section 3.1 for the server transaction state machine.

The problem with this approach is that it may create failures that would otherwise not have happened. It is possible for fragmented UDP datagrams to be reassembled successfully, so even a very large
success response might reach the requestor over a path including one or more UDP hops. In addition, it may not be possible for the requestor to force a transport change; the requestor or some proxy on the path may be limited to UDP. Using this error would eliminate any possibility of such a configuration working.

4.2 Use message/sipfrag

An appealing alternative is to use a 489 response with a body corresponding to the "message/sipfrag" MIME type [3]. The body can include more diagnostic information, much the same as is done for hop limit expiration [7].

This alternative was eschewed in favor of simplicity since it then opens up the possibility of the response getting bigger and what to prune.

4.3 Request in the backwards direction

A second alternative is to have the server that generates a response use a 489, but then open up a new request in the direction of the client. The client’s URI can be obtained from the Contact header of the original request. The new request can be sent over TCP and it can contain information to allow the client to match the new request with the one that generated the 489.

There are two issues with this approach. For one, it will not work for requests that omit the Contact header. Additionally, the UAC may not be directly accessible (i.e., it may be behind a NAT or a firewall) requiring a GRUU.

5. Security Considerations

To put in.

6. IANA Considerations

To put in.

7. Acknowledgments

The alternative in Section 4.3 was proposed by Jonathan Rosenberg on the SIP mailing list.

8. References
8.1 Normative References


8.2 Informative References


Authors’ Addresses

Vijay K. Gurbani  
Lucent Technologies, Inc./Bell Laboratories  
2701 Lucent Lane  
Room 9F-546  
Lisle, IL  60532  
USA  

Phone: +1 630 224-0216  
Email: vkg at bell hyphen labs dot com
Scott Lawrence
Pingtel Corp.
400 West Cummings Park
Suite 2200
Woburn, MA 01801
USA

Phone:  +1 781 938 5306
Email:  slawrence@pingtel.com