Preventing IP Fragmentation in Responses for the Session Initiation Protocol (SIP)
draft-petithuguenin-sip-fragmentation-responses-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 http://www.ietf.org/shadow.html.

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

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

There is limited support to prevent IP fragmentation when using the UDP transport with the Session Initiation Protocol (SIP). This document describes an extension to prevent fragmentation in responses.
Table of Contents

1. Introduction ............................................. 3
2. Terminology .............................................. 3
3. Overview of Operation ..................................... 3
4. Detailed Processing Rules ................................. 4
   4.1. UAC Behavior ........................................ 4
   4.2. Proxy Behavior ...................................... 4
   4.3. UAS Behavior ........................................ 5
5. Grammar ...................................................... 6
6. IANA Considerations ......................................... 6
7. Security Considerations ..................................... 6
8. Examples .................................................... 6
9. Acknowledgements .......................................... 6
10. References .................................................. 7
   10.1. Normative References ................................. 7
   10.2. Informative References .............................. 7
Author’s Address .............................................. 7
Intellectual Property and Copyright Statements ............... 8
1. Introduction

RFC 3261 [3] section 18.1.1 presents a simple algorithm to prevent fragmentation in the response, by sending the request over a congestion controlled transport protocol (e.g. TCP) if the size of the packet is too large.

The problem is that it is difficult to calculate the size of the packets that will be received from the size of the packet sent so the algorithm simply presumes that the response will be 200 bytes bigger than the request. The real size of the packet can be lower than this, and perhaps small enough to do not mandate the use of TCP, or can be higher than this and then resulting in unexpected fragmentation.

The fundamental assumption for this extension is that a proxy is capable to evaluate the number of bytes it will increase or decrease the size of the response received before forwarding it. If the proxy stores this value in the Via header, and all the other proxies on the path to the endpoint are also doing this, then the UAS will be able to evaluate if the response will be fragmented when sent back by this proxy.

2. 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].

3. Overview of Operation

The protocol works by adding one or two parameters on each Via. The first parameter, "inc", contains a negative or positive value that indicates the number of bytes that will be added or removed when traversing this network element. It also signals that this element is supporting this specification. The second parameter, "max", contains the MTU for the network element that added the Via. This parameter is added only when UDP is used. By walking through the Via list, an UAS supporting this specification can find which network element need to switch to a congestion controlled transport protocol to prevent IP fragmentation. If it is the case, the UAS sends a 607 Congestion Control response with a Redirect-Target header that is used to propagate the response to the chosen network element. The network element then resend the request by using a congestion controlled transport protocol.
4. Detailed Processing Rules

4.1. UAC Behavior

When an UAC sends over any protocol a request that is not the result of a 607 Congestion Control response, it MUST add an inc parameter equal to 0. When sending the request over UDP, an UAC MUST also add a max parameter with a value equal to the path MTU (or 1500 if the MTU is unknown) to the Via header field. For other protocols the UAC MUST NOT add the max parameter to the Via header field. The rules described in section 18.1.1 of RFC 3261 are used to choose if the request is sent over UDP or a RFC 2914 [2] congestion controlled transport protocol.

When an UAC receives a 607 Congestion Control response, it checks if the response was received from the UDP transport. If not, then an element did not follow this specification and the branch MUST be retried with the same protocol but without the inc and max Via parameters defined by this specification. If the response was received from the UDP transport, then the branch MUST be retried, but this time by using a RFC 2914 [2] congestion controlled transport protocol. The new request must have a Via header field with a different branch parameter, an inc parameter equal to 0 and no max parameter.

The UAC retries the branch by applying the RFC 3263 [4] algorithm to the same Request-URI, but this time by excluding UDP from the list of supported protocols.

4.2. Proxy Behavior

A proxy can choose to forward a response received. Before forwarding the response the proxy will remove and eventually add elements to the message and so will increase or decrease the size of the response by a number of bytes. In the majority of cases the size added or removed can be evaluated from the content of the request that will trigger the response and from the knowledge of the internal processing done by the proxy. The proxy MUST evaluate this value before proxying the request and add this value in an inc parameter to the Via header field that will be added as part of the proxy operation. If the request will be proxied using an UDP transport then the path MTU (or 1500 if the path MTU is not known) MUST also be added as a max parameter to the new Via header field. The max parameter MUST NOT be added for other protocols.

When a proxy receives a 607 Congestion Control response it MUST check for the presence of the Redirect-Target header field. If the field does not exist, or contains the value 0, then the response is for...
this proxy. If the response is for this proxy, then it checks that this response was received from the UDP transport. If not, then an element did not follow this specification and the branch must be retried with the same protocol, but without the Via parameters defined by this specification. If the response is for this proxy and was received from the UDP transport, then the branch must be retried, but this time by using a congestion controlled transport protocol. The new request must have a Via header field with a different branch parameter, an inc parameter and no max parameter. If the response is not for this proxy, the value of the Redirect-Target header field MUST be decremented by one. The proxy MUST cancel all the eventual pending branches and forwards the response downstream.

4.3. UAS Behavior

When an UAS receives a request it searches if an UDP transport was used somewhere in the path from the UAC and if the client that used this transport follows this specification. To do this the UAS searches the first Via header field that contains a max parameter, starting from the top Via header field. If none are found then there is no UDP transport in the path that can be changed to a congestion controlled transport protocol, and the remaining of this section can be skipped. If a Via header field with a max parameter is found, then the UAS must check that all the Via headers field between this Via header field and the top Via header field (including this two Via header fields) contain an inc parameter. If not then one of the proxies does not support this specification, and the remaining of this section can be skipped.

The UAS then calculates the size of the response that will be sent (if the UAS can send provisional responses the maximum size of all the possible responses is used). The UAS then inspects each Via header fields in the request, starting from the top Via. The processing stops when a Via header field does not contain an inc parameter or when there is no other Via header field to process. If the Via header field contains a max value, then the max value is compared to the size of the response. If the size of the response is higher than the value of the max parameter, then the UAS MUST send a 607 response. The UAS MUST add a Redirect-Target header field with a value equal to the index of the Via containing the max parameter, with the top Via having the index 0 and the subsequent Via incrementing the index value by one. If the response size is lower or equal to the max parameter value, then the inc parameter value must be added to the response size. The next Via header field will be then processed by using this modified value as the response size.
5. Grammar

This specification defines two new Via header field parameters, inc and max. The following ABNF [5] uses some definitions from RFC 3261 [3]:

```plaintext
via-params = via-ttl / via-maddr / via-received / via-branch / via-inc / via-max / via-extension
```

```plaintext
via-inc = "inc" EQUAL [-] 1*DIGIT
```

```plaintext
via-max = "max" EQUAL 1*DIGIT
```

This specification defines also a new header field parameters, Redirect-Target. The following ABNF [5] uses some definitions from RFC 3261 [3]:

```plaintext
Redirect-Target = "Redirect-Target" HCOLON target-val
```

```plaintext
target-val = 1*DIGIT
```

6. IANA Considerations

TBD

7. Security Considerations

TBD

8. Examples

TBD

9. Acknowledgements

Internal versions of this document were reviewed by Patrice Bruno, Lee Hong, Garth Judge, Suhas Joshi, Jim Kleck, Eric Lin, Jason Liu, Vadim Tsyganok and Qing Zhao.

The Redirect-Target header definition and behavior were borrowed from [6]

10. References
10.1. Normative References


10.2. Informative References


Author’s Address

Marc Petit-Huguenin  
8x8, Inc.  
3151 Jay Street  
Santa Clara, CA  95054  
US  
Phone: +1 408 654 0875  
Email: marc@8x8.com