Abstract

The Session Initiation Protocol allows the use of UDP for transport of SIP messages. The use of UDP inherently risks network congestion problems, as UDP itself does not define congestion prevention, avoidance, detection, or correction mechanisms. This problem is aggravated by large SIP messages which fragment at the UDP level. Transport protocols in SIP are also negotiated on a per-hop basis, at the SIP level, so SIP proxies may convert from TCP to UDP and so forth. This document defines what it means for SIP nodes to be congestion safe and specifies an extension by which a SIP User Agent may require that its requests are treated in a congestion safe manner.
Table of Contents

1. Terminology ................................................. 3
2. Background .................................................... 3
3. Definition of Congestion Safety for SIP ..................... 3
4. Assuring Transitive Congestion Safety with Proxy-Require ... 4
5. Responsible use of SIP over UDP ............................. 4
  5.1 Requirements For Use of SIP Over UDP .................... 6
  5.2 Pacing SIP Requests Over UDP .............................. 6
  5.3 Proxy Rejects Requests That Would Require UDP Fragmentation . 7
6. Syntax of Extensions and Changes to SIP Specifications .... 8
7. IANA Considerations ........................................... 9
8. Acknowledgements ............................................. 10
   Normative References ........................................... 10
   Authors’ Addresses .............................................. 10
   Full Copyright Statement ....................................... 11
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 [RFC2119].

2. Background

The Session Initiation Protocol RFC 3261 [4] provides application support over multiple transport protocols, including UDP and TCP. Transport negotiation is not "end to end" with SIP. Instead, each SIP hop individually determines which transport to use. For example, a User Agent (UA) may use TCP to talk to a proxy, that proxy my use UDP to talk to another proxy, and that second proxy may use SCTP to talk to a destination UA.

UDP has inherent issues with congestion management. The protocol has not explicit mechanisms for avoiding, detecting, or adapting to network congestion. SIP attempts to deal with this in two ways:
1. Retransmission timers with exponential back offs.
2. Attempting to limit the size of transmissions over UDP to reduce the effects of fragmentation.

This would appear to be an incomplete solution. One solution might be to deprecate UDP entirely for SIP. However, there is a large installed base using UDP, and there are legitimately places where UDP appears to be quite useful such as tiny mobile phones and in extremely high-volume proxies connecting over dedicated networks.

As an alternative, this draft:
1. Defines what it means for a SIP node to be "congestion-safe".
2. Defines a mechanism whereby a congestion-safe UA may require that any proxy processing its requests be congestion safe.
3. Defines a mechanism whereby a proxy may reject a request that it would be forced to fragment, and in so doing inform the originating UA of relevant sizing parameters.

3. Definition of Congestion Safety for SIP

A SIP node can be considered "congestion safe" if it never emits a request in a manner not known to be congestion safe. Requests may be considered congestion-safe if any one of the following criteria is met:
1. The transport toward the next SIP hop is TCP, SCTP, or other transport providing congestion control and the next hop is known to be either a UA or a congestion-safe proxy.
2. The transport toward the next hop is UDP, the next hop is known to be a UA or congestion-safe proxy, and the network between the
two is known to support congestion management at a lower layer. Note that this is an uncommon case in typical Internet applications.

3. If the only available transport toward the next hop is UDP and the next hop is known to be a UA or congestion-safe proxy, the request MAY be transmitted over UDP or rejected by local policy. If the request is transmitted over UDP, the procedures described under the heading "Responsible use of SIP over UDP" in this document MUST be followed.

The preceding uses the phrase "the next hop is known to be either a UA or a congestion-safe proxy." Such knowledge may be derived either through administrative configuration or through use of the Proxy-Require mechanism defined herein under the heading "Assuring Transitive Congestion Safety with Proxy-Require".

4. Assuring Transitive Congestion Safety with Proxy-Require

SIP provides a mechanism whereby a user agent making a request can be assured that any proxy servicing that request support a specific extension or set of behavior. To do so, the user agent includes a "Proxy-Require" header field with a value indicating a tag for the specific extension or behavior required. There is an IANA registration process for these tags. Proxies not recognizing a specific tag or unwilling to support the associated behavior MUST reject a request referencing that tag with a 420 response, which has the semantic "Unsupported".

We herein define a tag value of "congestion-safe". A proxy forwarding a request containing a Proxy-Require with this tag value MUST manifest the property of congestion-safety as defined by this document.

5. Responsible use of SIP over UDP

The fundamental problem with UDP is that it provides no feedback mechanism to allow a sender to pace its transmissions against the real performance of the network. While this tends to have no significant effect on extremely low-volume sender-receiver pairs, the impact of high-volume relationships on the network can be severe. Consider the following scenario, wherein the traffic between multiple UAs is funnelled through a single proxy-proxy relationship.
Example of large-fan out/fan-in likely to encounter congestion:

```
UA1----\            /----UA10
UA2-----\            /-----UA11
UA3------\          /------UA12
UA4-------\        /-------UA13
UA5--------P1------P2--------UA14
UA6--------/      \--------UA15
UA7-------/      \-------UA16
UA8-------/      \-----UA17
UA9----/              \----UA18
```

In this scenario, any requests from UA(1..9) to UA(10..18) traverse the proxy-proxy link P1&lt-->P2. Assuming current SIP practices, if this link is UDP and every UA emits a request simultaneously, each proxy will insert nine (one for each UA) requests, resulting in eighteen simultaneous requests on the P1&lt-->P2 link. Each request may require retransmissions, and large requests may require fragmentation to fit the link MTU -- at the worst case, producing more than one hundred packets per request, or approximately 2,000 simultaneously expressed packets in this scenario. If the capacity of link P1&lt-->P2 is inadequate to deliver these messages within the SIP retransmission window, the originating UAs (or the proxies, if acting in transaction-stateful mode) generate retransmissions, further compounding the problem into a "retransmission storm". Real-world scenarios may scale far more seriously. It is not unreasonable to assume that there may be tens of thousands of UAs on each side of the network.

Clearly the best thing to do is to use a more sophisticated transport protocol (TCP, SCTP, etc.) between P1 and P2, and between each UA and its associated proxy. If this is not feasible, it may be necessary to fall back to UDP. This is especially common in the case of low-capacity UAs such as those proposed for 3G wireless systems.

It should be noted that the fundamental problem not just between UAs and proxies, but whenever there is a high fan-out or fan-in ration. If in the above example, each UA were behind a "residential proxy", the problem would occur in similar fashion.

One might propose that SIP ALWAYS use a congestion-controlled transport to talk to proxies, and only fall back to UDP when the next hop is a UA. The primary problem with this approach is that in general, a SIP node does not and cannot know whether the next node is a UA or a proxy -- it is this ability to "insert" proxies into a...
sequence that provides much of the flexibility of SIP. A secondary
problem is that even if the next hop is a UA, some UAs are
sufficiently high volume, and some links sufficiently narrow, that
congestion might still result from the incautious use of UDP.

5.1 Requirements For Use of SIP Over UDP

The previously described problems with the general use of SIP over
UDP lead to the following two requirements for the use of UDP as a
transport protocol for SIP:

1. Large messages MUST NOT be transmitted over UDP. The SIP
   specification provides basic guidance for UAs. Congestion-safe
   proxies MUST follow the procedures described below under the
   heading "Proxy Rejects Request That Would Require UDP
   Fragmentation." UAs MAY also make use of the MTU feedback
   techniques in that section.

2. Nodes sending requests over UDP MUST pace those requests as
   described under the heading "Pacing SIP requests over UDP."

Response messages SHOULD be constrained to be smaller than the MTUs
established for requests by the preceding mechanisms, and systems
implementors should remain aware that SIP provides limited support
for managing response sizes. Further experience may indicate a need
for further control over response handling.

5.2 Pacing SIP Requests Over UDP

One simple way to describe the congestion problem is that UDP lets us
send packets without knowing whether those packets are arriving. The
simplest approach to dealing with this at the application level is to
send a request, then wait for some sort of response indicating that
the request was received before sending anything else. This produces
an effect described by some as "ping-ponging" -- traffic bounces back
and forth between two nodes like a ping-pong ball or tennis ball in a
match. Since there’s only one ball in play between any two players
at any given time, most of the potential for congestion cascades is
eliminated.

This pacing or serialization approach has the side-effect of
significantly reducing the maximum throughput, as transmission occurs
in only one direction at a time and there is at least a 2xRTT delay
between transmissions. More sophisticated algorithms such as those
in TCP and SCTP have been developed to address this, and it would be
inappropriate to duplicate that work here. Consequently, if greater
efficiency is required than that provided by this simple approach,
implementors should use TCP, SCTP, or another such protocol. But if
one absolutely must use UDP, this approach works, and is reasonably
efficient in the most likely application of "edge proxy" to UA and
other proxies with large fan-outs to individual low-volume nodes.

SIP has two sorts of request transactions: "invite" and "non-invite" transactions. Invite transaction use a three way sequence of "request, response, acknowledgement" and may include a "provisional response" between the request and response steps. Non-invite transactions use a two-way "request, response" sequence, and may also have a provisional response although that behavior has been deprecated.

Congestion-safe use of SIP over UDP requires waiting for some sort of response to a request (or a timeout, which has backoff properties) before sending another request to that same destination. A congestion-safe SIP node (UA or proxy) MUST NOT send a request to a given next-hop if there is an existing request to that destination which has not received some sort of response. The existing transaction MUST either receive a response (final or provisional) or time-out before a new request can be made to that next-hop.

This effectively requires congestion-safe proxies to act in a transaction-stateful manner on a per-next-hop destination basis, at least to the extent of tracking whether some sort of request is pending to each next-hop and correlating provisional and final responses to that request.

Some may argue that this puts an excessive burden onto the SIP node, and that implementations that are "congestion-safe" per this specification will have reduced performance when used with UDP over a shared or public network. We counter that congestion-safe transport protocols are readily available, and that network users which insist on using unsafe transports (such as UDP) MUST be responsible for assuring that they do not impede the function of other users of the network, even at the expense of reducing their own efficiency. It is simply irresponsible to "blast away" at the network without regard for congestion or its impact on other users of the network.

5.3 Proxy Rejects Requests That Would Require UDP Fragmentation

A proxy may be faced with a request to deliver a large message using UDP as a transport. Fragmentation of such messages is problematic in several ways. Loss of any fragment requires time-out and retransmission of the message. The fragments are commonly transmitted out the interface at local interface (usually LAN) rates, without awareness of intervening network conditions. For these reason, we believe it in general a bad practice to send large requests over UDP.

While the actual MTU of a link may not be known, common practice
seems to indicate that the local interface MTU is likely to be a reasonable approximation. Where the actual path MTU is known, that value should be used instead.

When a congestion-safe SIP proxy processing a request determines that the next hop is reached via UDP, and that the request is larger than the effective MTU toward that hop and would consequently be fragmented, the proxy MUST reject that request with a 513 response.

The base SIP specification provides minimal guidance on dealing with oversized requests. There is an error response code, 513, with the semantic "request too large" that seems applicable. However, SIP provides no guidance on how to indicate what size might be allowed. We define here two extension header fields that may be used in a 513 response to indicate by the rejecting proxy the size of message allowed by that proxy. The extension header field "Proxy-Max-Size" may be used to indicate the largest allowable request to the originating UA. The extension header field "Proxy-Seen-Size" may be used to indicate the size of the rejected request as calculated by the rejecting proxy. In both cases, the size value used indicates the SIP message size, which does not include IP or transport protocol overhead.

A congestion-safe SIP proxy which rejects a request based on size SHOULD include a "Proxy-Max-Size" header field with a value indicating the largest size message allowed by this proxy on this link. If a Proxy-Max-Size header field is sent, the proxy MUST also include a "Proxy-Seen-Size" header indicating the size of the request as seen at this proxy.

A UA receiving a 513 response has the options of giving up, trying a smaller request, or trying a different set of proxies. Should it choose to try a smaller request, it may estimate the size of the largest message that can be sent by taking the original request size, subtracting it from the value of the Proxy-Seen-Size header field, and subtracting that result from the value of the Proxy-max-Size header field.

6. Syntax of Extensions and Changes to SIP Specifications

The syntax for the Proxy-Max-Size header field is:

Proxy-Max-Size = "Proxy-Max-Size" HCOLON 1*DIGIT

The syntax for the Proxy-Seen-Size header field is:

Proxy-Seen-Size = "Proxy-Seen-Size" HCOLON 1*DIGIT
Additions to SIP Table 3:

<table>
<thead>
<tr>
<th>Header field</th>
<th>where</th>
<th>proxy</th>
<th>ACK</th>
<th>BYE</th>
<th>CAN</th>
<th>INV</th>
<th>OPT</th>
<th>REG</th>
<th>PRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy-Max-Size</td>
<td>513</td>
<td>a</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Proxy-Seen-Size</td>
<td>513</td>
<td>a</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

7. IANA Considerations

This document defines the SIP extension header fields "Proxy-Max-Size" and "Proxy-Seen-Size", which IANA will add to the registry of SIP header fields defined in RFC 3261 [4].

This document also defines the SIP option tag "congestion-safe" which IANA will add to the registry of SIP option tags defined in RFC 3261 [4].

The following is the registration for the Proxy-Max-Size header field:

RFC Number: RFCXXXX [Note to IANA: Fill in with the RFC number of this specification.]

Header Field Name: Proxy-Max-Size

Compact Form: none

The following is the registration for the Proxy-Seen-Size header field:

RFC Number: RFCXXXX [Note to IANA: Fill in with the RFC number of this specification.]

Header Field Name: Proxy-Seen-Size

Compact Form: none

The following is the registration for the congestion-safe option tag:

RFC Number: RFCXXXX [Note to IANA: Fill in with the RFC number of this specification.]

Option Tag: congestion-safe
8. Acknowledgements

Robert Sparks and Jonathan Rosenberg argued with us vociferously over this topic and contributed substantial insight.

Normative References


Authors’ Addresses

Dean Willis
dynamicsoft Inc.
5100 Tennyson Parkway
Suite 1200
Plano, TX  75028
US

Phone: +1 972 473 5455
EMail: dean.willis@softarmor.com
URI:  http://www.dynamicsoft.com/

Ben Campbell
dynamicsoft Inc.
5100 Tennyson Parkway
Suite 1200
Plano, TX  75028
US

Phone: +1 972 473 5452
EMail: bcampbell@dynamicsoft.com
URI:  http://www.dynamicsoft.com/