SIP WG
Internet-Draft
Expires: August 1, 2004

R. Mahy
Cisco Systems, Inc.
Feb 2004

Connection Reuse in the Session Initiation Protocol (SIP)
draft-ietf-sip-connect-reuse-01.txt

Status of this Memo

Copyright Notice

Abstract

When SIP entities use a connection oriented protocol to send a request, they typically originate their connections from an ephemeral port. The SIP protocol includes mechanisms which insure that responses to a request, and new requests sent in the original direction reuse an existing connection. However, new requests sent in the opposite direction are unlikely to reuse the existing connection. This frequently causes a pair of SIP entities to use one connection for requests sent in each direction, and can result in potential scaling and performance problems. This document proposes requirements and a mechanism which address this deficiency.

Table of Contents

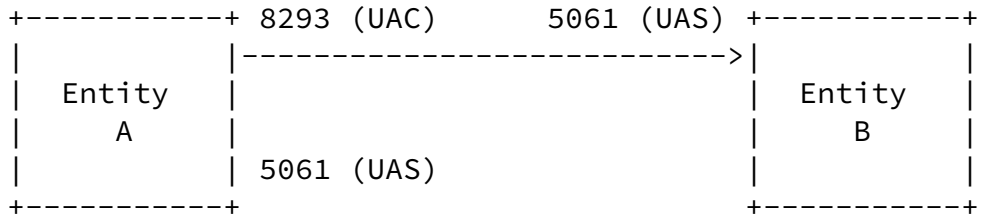
1. Conventions

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 [2].

2. Introduction and Problem Statement

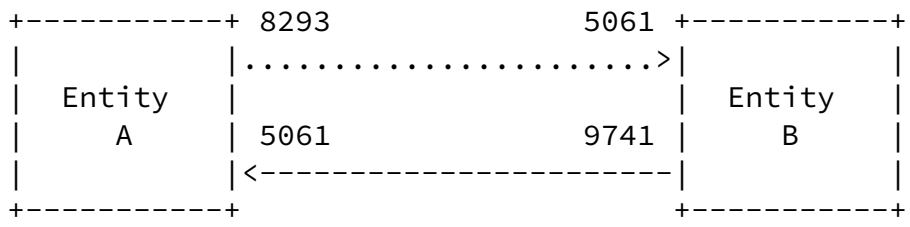
SIP [1] entities can communicate using either unreliable/ connectionless (ex: UDP) or reliable/connection-oriented (ex: TCP, SCTP [11]) transport protocols. When SIP entities use a connection-oriented protocol (such as TCP or SCTP) to send a request, they typically originate their connections from an ephemeral port.

In the following example, Entity A listens for SIP requests over TLS [4] on TCP port 5061 (the default port for SIP over TLS over TCP), but uses an ephemeral port (port 8293) for a new connection to Entity B. These entities could be SIP User Agents or SIP Proxy Servers.



The SIP protocol includes mechanisms which insure that responses to a request reuse the existing connection which is typically still available, and also includes provisions for reusing existing connections for other requests sent by the originator of the connection. However, new requests sent in the opposite direction (routed from the target of the original connection toward the originator of the original connection) are unlikely to reuse the existing connection. This frequently causes a pair of SIP entities to use one connection for requests sent in each direction, as shown

below.



This extra pair of connections can result in potential scaling and performance problems. For example, each new connection using TLS requires a TCP 3-way handshake, a handful of round-trips to establish TLS, and (typically) expensive asymetric authentication and key

generation algorithms, and certificate verification. This effectively doubles the load on each entity. Setting up a second connection can also cause excessive delay (especially in networks with long round-trip times) for subsequent requests, even requests in the context of an existing dialog (for example a reINVITE or BYE after an initial INVITE, or a NOTIFY after a SUBSCRIBE [8] or a REFER [9]).

Consider the call flow shown below where Proxy A and Proxy B use the Record-Route mechanism to stay involved in a dialog. Proxy B will establish a new TLS connection just to send a BYE request.

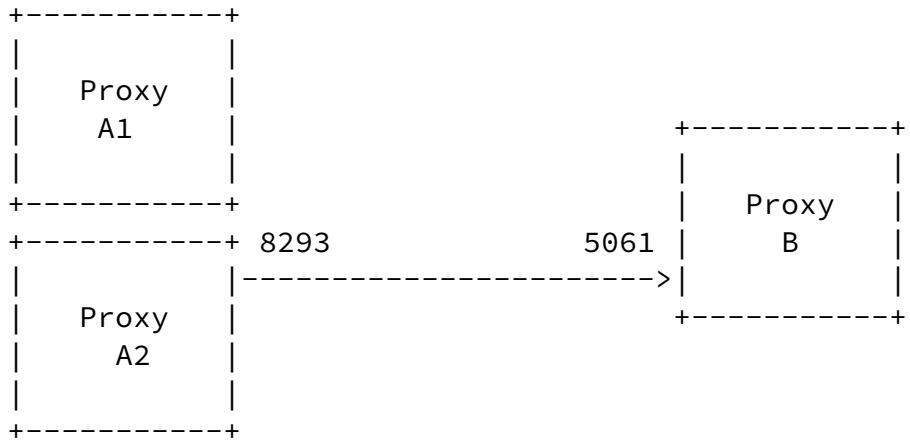
```
INVITE ->   create connection 1
<- 200      response over connection 1
ACK ->      reuse connection 1

<- BYE      create connection 2
-> 200      response over connection 2
```

ReINVITEs are expected to be handled automatically and rapidly in order to avoid media and session state from being out of step. If a reINVITE requires a new TLS connection, the reINVITE could be delayed by several extra round-trip times. Depending on the round-trip time, this combined delay could be perceptible or even annoying to a human user. This is especially problematic for some common SIP call flows (for example, the recommended example flow in figure number 4 in 3pcc [7]) use many reINVITEs.

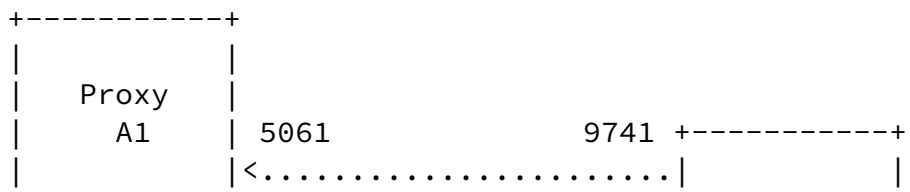
Consider also a call flow where a handheld organizer sends a REFER request which establishes a dialog to a SIP phone. Typically this would require a second connection back to the handheld to be established.

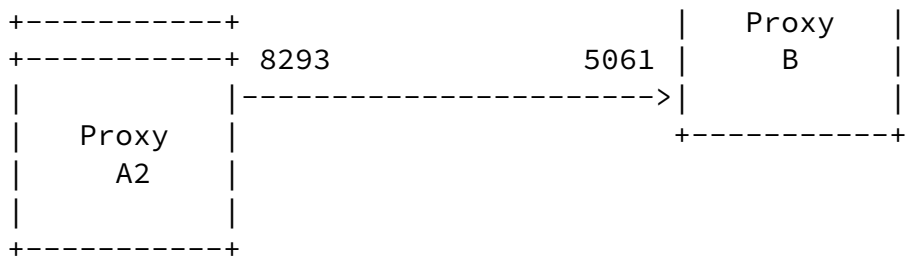
```
REFER ->                connection 1
<- 202                  connection 1
<- NOTIFY               connection 2
200 ->                  connection 2
           INVITE ->
           <- 200
<- NOTIFY               connection 2
200 ->                  connection 2
```

Likewise when clusters or farms of cooperating SIP servers (for example proxy servers) are configured together, SIP entities have no way to prefer a server with an existing connection. For example, Proxy server B has no mechanism to choose an existing connection with Proxy cluster A.

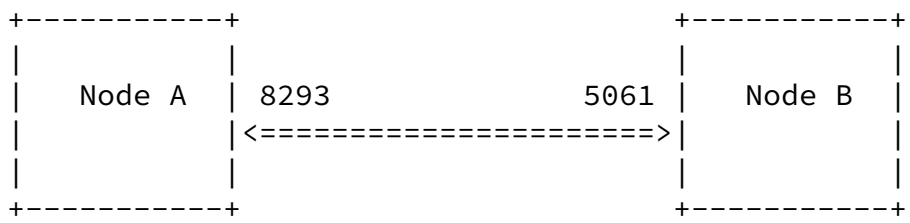


As a result, Proxy B might open a new connection to another proxy server for requests sent in the opposite direction.





The rules for handling the Transport layer described in Section 18 of SIP [1] do not associate incoming connections with the listening port which corresponds to the same SIP entity. If the Tranport layer had some way to associate these connections, then request and responses originated from either node could reuse existing connections as shown below.



3. Requirements

1. A connection sharing mechanism SHOULD allow SIP entities to reuse existing connections for requests and repsonses originated from

either peer in the connection.

2. A connection sharing mechanism SHOULD allow SIP entities to reuse existing connections with closely coupled nodes which act as a single SIP entity (for example a cluster of nodes acting as a proxy server).

3. A connection sharing mechanism MUST NOT require UACs (clients) to send all traffic from well-know SIP ports.

4. A connection sharing mechanism MUST NOT require configuring ephemeral port numbers in DNS.

5. A connection sharing mechanism MUST prevent unauthorized

hijacking of other connections.

6. Connection sharing SHOULD persist across SIP transactions and dialogs.

4. Behavior

The proposed mechanism uses a new Via header field parameter. The "alias" parameter is included in a Via header field value to indicate that the originator of the request wants to create a transport layer alias, so that the sent-by address becomes mapped to the current connection.

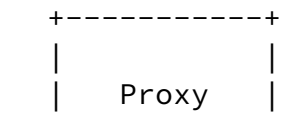
Assuming the Via header field value shown below from the most recent request arrived over a connection from 10.54.32.1 port 8241:

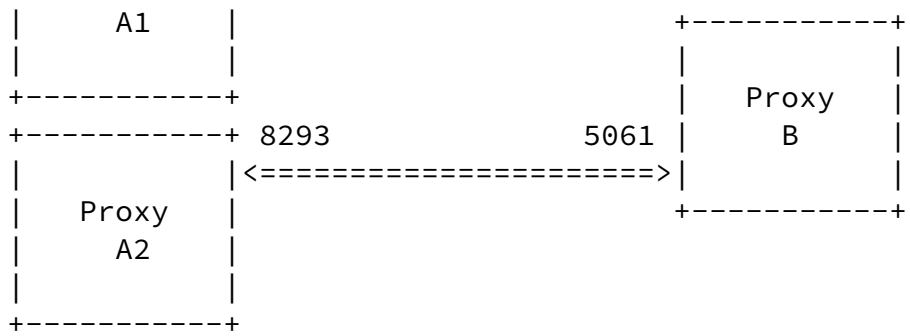
```
Via: SIP/2.0/TLS 10.54.32.1:5061;branch=z9hG4bKa7c8dze ;alias
```

The transport layer creates an alias, such that any requests going to the "advertised address" (10.54.32.1 port 5061) are instead sent over the existing connection (to the "target" of the alias) which is coming from port 8241. This sharing continues as long as the target connection stays up. The SIP community recommends that servers keep connections up unless they need to reclaim resources, and that clients keep connections up as long as they are needed. Connection reuse works best when the client and the server maintain their connections for long periods of time. SIP entities therefore SHOULD NOT drop connections on completion of a transaction or termination of a dialog.

Likewise when clusters or farms of cooperating SIP servers (for example proxy servers) are configured together, the proposed mechanism allows a SIP entity to select a server with an existing

connection. With the proposed mechanism, Proxy B sends requests for Proxy cluster A to node A2 with whom it shares an existing connection.





For example, on receipt of a message with the topmost Via header shown below, the transport layer creates an alias such that requests going to the advertised address (proxy-farm-a.example.com) are sent over the target connection (from 10.54.32.1:8241).

```
  Via: SIP/2.0/TLS proxy-farm-a.example.com;branch=z9hG4bK7c8ze;alias
```

As a result, this has an important interaction with the DNS resolution mechanisms for SIP described in RFC3263 [6]. When new requests arrive for proxy-farm-a.example.com, proxy B still needs to perform a DNS NAPTR lookup to select the transport. Once the transport is selected, an SRV lookup would ordinarily occur to find the appropriate port number. In this case, the transport layer uses a connection reuse alias instead of performing the SRV query.

Below is a partial DNS zone file for atlanta.com.

```
; NAPTR queries for the current domain (example.com)
;
; order pref flags service regexp replacement
proxy-farm-a IN NAPTR 50 50 "s" "SIPS+D2T" "" _sips._tcp.

; SRV records for the proxy use 5060/5061
;
;; Priority Weight Port Target
_sips._tcp.proxy-farm-a  IN SRV 0 1 5061 host-a1
_sips._tcp.proxy-farm-a  IN SRV 0 1 5061 host-a2

host-a1  IN A 10.54.32.1
host-a2  IN A 10.54.32.2
```

The existence of an alias parameter is treated as a request which asks the transport layer to create an alias (named by the sent-by parameter, which could be a hostname) that points to the alias target (the current connection)

This mechanism is fully backwards compatible with existing implementations. If the proposed Via parameter is not understood by the recipient, it will be ignored and the two implementations will revert to current behavior (two connections).

4.1 Authorizing an alias request

Authorizing connection aliases is essential to prevent connection hijacking. For example a program run by a malicious user of a multiuser system could attempt to hijack SIP requests destined for the well-known SIP port from a large relay proxy.

To correctly authorize an alias, both the active connection and the alias need to authenticate using the same credentials. This could be accomplished using one of two mechanisms. The first (and preferred) mechanism is using TLS mutual authentication, such that the subjectAltName of the originator certificate corresponds to both the current connection and the target address of the alias. The Via sent-by address needs to be within the scope protected by the certificate presented by the originator during TLS mutual authentication and the received IP address needs be a valid IP address for the sent-by host or hosts. In other words, the sent-by address MUST be resolvable from the subjectAltName of the originator certificate, and the received IP address MUST be resolvable from the sent-by address. This is in addition to other requirements for TLS authentication and authorization discussed in SIP [1] and Locating SIP Servers [6].

Following this logic step-by-step:

1. Verify that the certificate presented is not expired and is rooted in a trusted certificate chain.

2. Verify that the subjectAltName in the certificate covers the "advertised address" (the address in the Via sent-by production). If the advertised address and the subjectAltName match exactly then the certificate covers the address. Also, use DNS to resolve if the advertised name is resolvable from the subjectAltName (start by resolving the subjectAltName with first NAPTR, next SRV, then finally address records). If any of the resolved addresses (port numbers can be ignored in this case) matches the advertised address, then the certificate covers the address.

3. Finally, Verify that the advertised address can resolve to the IP address over which the connection was received.

For example, take the example in the previous section of proxy B receiving an alias request from host-a2.example.com. Proxy B verifies that the presented certificate is valid and trusted. Proxy B checks that proxy-farm-a.example.com is both the advertised name and the subjectAltName in the certificate. Finally, proxy B verifies that this connection is coming from 10.54.32.2, which is one of the addresses in DNS for proxy-farm-a.example.com

The second mechanism is to accept an alias if the target address of the alias is equivalent (using SIP comparison rules) to a valid Contact already registered by the same user. This user could be authenticated through any SIP or TLS mechanism (ex: user certificate, or Kerberos [10]), but would typically use Digest authentication [5]. For example, if Alice registers a Contact of 198.168.67.89:5061, she could inform Proxy 1 of the existence of a connection to her from Proxy 2. This would allow her to preemptively originate TLS connections, as her user agent may not have access to a site certificate with which to authenticate incoming TLS connections.

The Proxy takes the following steps to authorize these requests:

1. The Proxy authenticates or authorizes the sender for an otherwise ordinary SIP request.

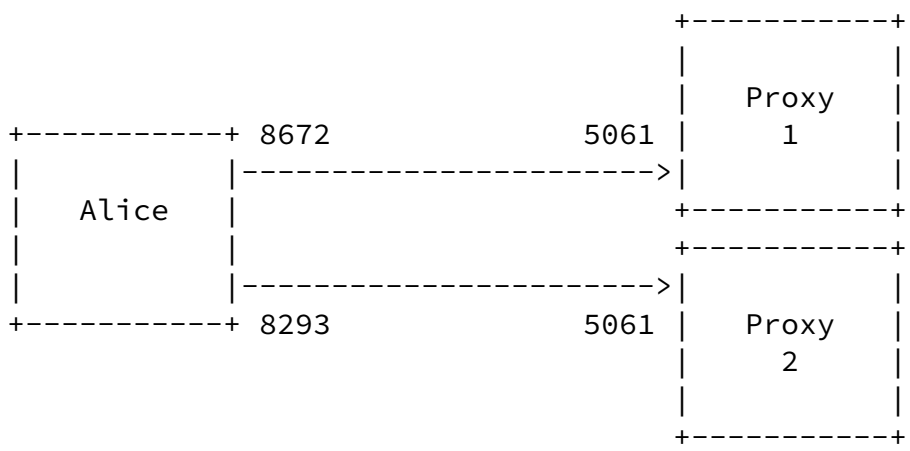
2. The Proxy looks for any Contacts in the location/registration service which have a hostport and transport that matches exactly the advertised address.

3. The Proxy checks if the user who sent the request would be authorized to change the Contact found when looking up the Contact URI in the location/registration service.

For example, Alice advertises the address "198.168.67.89:5061" in a request sent over a connection from "198.168.67.89:8293" to Proxy 2. The Proxy otherwise authenticates Alice's request (for example an INVITE request). The Proxy looks up 198.168.67.89:5061 and finds the following Contact: "Alice" <sips:reg2@198.168.67.89:5061>. Alice is authorized to modify Alice's contact, so Alice is authorized to alias an advertised address "reserved" by one of her Contacts. Alice then sends another request (this time an OPTIONS request for example) to

Proxy 1 from "198.168.67.89.8672" with the same Via header. Proxy 1 similarly authorizes Alice's request and stores the alias. Now if either proxy receives a request for 198.168.67.89:5061, it will forward the request over the appropriate existing connection with Alice.

Via: SIP/2.0/TLS 198.168.67.89:5061;branch=z9hG4bK7c8dze ;alias



4.2 Formal Syntax

The following syntax specification uses the augmented Backus-Naur Form (BNF) as described in RFC-2234 [3]. This document proposes to extend via-params to include a a new via-alias defined below.

```
via-params = via-ttl / via-maddr / via-received / via-branch /
             via-alias / via-extension

via-alias  = "alias"
```

5. Security Considerations

This document presents requirements and a mechanism for reusing existing connections easily. Connection reuse presents many opportunities for abuse and hijacking, but these attacks can be prevented if the guidelines in the authorization section are followed.

6. IANA Considerations

This document adds a parameter to the SIP header field parameters registry:

Header field in which parameter can appear: Via
Name of the parameter: alias
Reference: This document (RFC editor, please replace with the RFC number of this document)

7. Acknowledgments

Thanks to Jon Peterson for helpful answers about certificate behavior with SIP, Jonathan Rosenberg for his initial support of this concept, and Cullen Jennings for providing a sounding board for this idea.

Normative References

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[4] Dierks, T., Allen, C., Treese, W., Karlton, P., Freier, A. and P. Kocher, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[5] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[6] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

Informational References

[7] Rosenberg, J., Schulzrinne, H., Camarillo, G. and J. Peterson, "Best Current Practices for Third Party Call Control in the Session Initiation Protocol", draft-ietf-sipping-3pcc-03 (work in progress), March 2003.

[8] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[9] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.

[10] Kohl, J. and B. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993.

[11] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L. and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000.

Author's Address

Rohan Mahy
Cisco Systems, Inc.
5617 Scotts Valley Dr
Scotts Valley, CA 95066
USA

EMail: rohan@cisco.com

Intellectual Property Statement

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.