Network Working Group
Request for Comments: 5658
Category: Standards Track

T. Froment
Tech-invite
C. Lebel
B. Bonnaerens
Alcatel-Lucent

October 2009

Addressing Record-Route Issues in the Session Initiation Protocol (SIP)

Abstract

A typical function of a Session Initiation Protocol (SIP) Proxy is to insert a Record-Route header into initial, dialog-creating requests in order to make subsequent, in-dialog requests pass through it. This header contains a SIP Uniform Resource Identifier (URI) or SIPS (secure SIP) URI indicating where and how the subsequent requests should be sent to reach the proxy. These SIP or SIPS URIs can contain IPv4 or IPv6 addresses and URI parameters that could influence the routing such as the transport parameter (for example, transport=tcp), or a compression indication like "comp=sigcomp". When a proxy has to change some of those parameters between its incoming and outgoing interfaces (multi-homed proxies, transport protocol switching, or IPv4 to IPv6 scenarios, etc.), the question arises on what should be put in Record-Route header(s). It is not possible to make one header have the characteristics of both interfaces at the same time. This document aims to clarify these scenarios and fix bugs already identified on this topic; it formally recommends the use of the double Record-Route technique as an alternative to the current RFC 3261 text, which describes only a Record-Route rewriting solution.

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

Over the years, it has been noticed in interoperability events like SIPit, that many implementations had interoperability problems due to various Record-Routing issues or misinterpretations of [RFC3261]; in particular, when a change occurs between the incoming and outgoing sides of a proxy: transport protocol switching, "multi-homed" proxies (including IPv4 to IPv6 interface changes), etc. Multiple documents have addressed the question, each of them generally providing an adequate recommendation for its specific use case, but none of them gives a general solution or provides a coherent set of clarifications:

- [RFC3486], Section 6, describes the double Record-Routing as an alternative to the Record-Route rewriting in responses. This document is limited in scope to the "comp=sigcomp" parameter when doing compression with Signalling Compression (SIGCOMP).

- [RFC3608], Section 6.2, recommends the usage of double Record-Routing instead of the rewriting solution described in [RFC3261] for "Dual-homed" proxies. Those are defined as "proxies connected to two (or more) different networks such that requests are received on one interface and proxied out through another network interface".

- Section 3.1.1 of [V6Tran] mandates double Record-Routing for multi-homed proxies doing IPv4/IPv6 transitions, when the proxy inserts IP addresses in the Record-Route header URI.

The observed interoperability problems can be explained by the fact that, despite these multiple documents, the RFC 3261 description has not been changed, and many implementations don't support extensions like Service-Route ([RFC3608]) or SIGCOMP ([RFC3486]).

This document also aims to clarify an identified bug referenced in [BUG664]. In particular, it takes into account the [BUG664] recommendation, which says that "the language that describes this, needs to clearly capture that this applies to all types of different interface on each side issues, including IPv4 on one side and IPv6 on the other".

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 [RFC2119].

3. Problem Statement

3.1. Background: Multi-Homed Proxies

A multi-homed proxy is a proxy connected, like a router, to two or more different networks, with an interface into each network, such that traffic comes "in" one network and goes "out" a different one. A simple example is shown here:

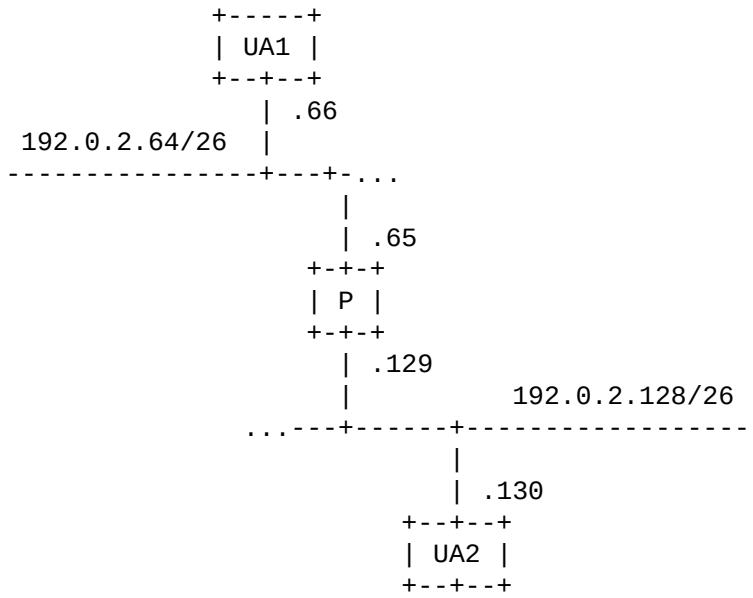

Figure 1: Multi-Homed Proxy Illustration

UA1 has one interface with IP address 192.0.2.66.

The Proxy P has two interfaces and two addresses:

--192.0.2.65

--192.0.2.129

UA2 has one interface with address, 192.0.2.130. There is potentially no IP-level route between UA1 and UA2 (pinging or traceroute does not work between these two hosts). They live in entirely different subnetworks. But they can still exchange SIP messages, because P is a SIP Proxy. This works in SIP because P can apply Record-Routing.

In most cases, there is still some IP connectivity between UA1 and UA2, but SIP proxy has to manage the SIP traffic between the two different "sides", e.g., with two different IP addresses, or one side using SIGCOMP and the other side not, etc.

3.2. Identified Problems

Handling of the Record-Route header in SIP Proxies is specified by following sections of [RFC3261]:

On the request processing side, [RFC3261], item 4 of Section 16.6 states that:

> The URI placed in the Record-Route header field value MUST be a SIP or SIPS URI. [...] The URI SHOULD NOT contain the transport parameter unless the proxy has knowledge (such as in a private network) that the next downstream element that will be in the path of subsequent requests supports that transport.

Following this statement, it is not clear how to decide when the proxy should insert the transport protocol parameter in the Record-Route URI.

On the response processing side, [RFC3261] recommends in step 8 of Section 16.7 that:

> If the selected response contains a Record-Route header field value originally provided by this proxy, the proxy MAY choose to rewrite the value before forwarding the response. This allows the proxy to provide different URIs for itself to the next upstream and downstream elements. A proxy may choose to use this mechanism for any reason. For instance, it is useful for multi-homed hosts.
>
> If the proxy received the request over Transport Layer Security (TLS), and sent it out over a non-TLS connection, the proxy MUST rewrite the URI in the Record-Route header field to be a SIPS URI.

Note that [RFC5630] has weakened the SIP/SIPS URI rewriting requirement in the Record-Route header by removing this second paragraph.

Indeed, [RFC3261] suggests rewriting the Record-Route header in responses.

This list highlights the utility of rewriting and double Record-Routing techniques that apply for any multi-homed proxy use case: whenever the proxy changes its IP address, the transport protocol, or the URI scheme between incoming and outgoing interfaces. Rewriting

and double Record-Routing are described, compared, and discussed in Sections 4 and 5; the specific question of whether or not to insert the transport parameter in the Record-Route URI is then discussed in Section 6.

4. Record-Route Rewriting

As frequently outlined in IETF mailing list discussions, Record-Route rewriting in responses is not the optimal way of handling multi-homed and transport protocol switching situations. Additionally, the consequence of doing rewriting is that the route set seen by the caller is different from the route set seen by the callee, and this has at least two negative implications:

1) The callee cannot sign the route set, because it gets edited by the proxy in the response. Consequently, end-to-end protection of the route set cannot be supported by the protocol. This means the Internet's principles of openness and end-to-end connectivity are broken.

2) A proxy must implement special "multi-homed" logic. During the request forwarding phase, it performs an output interface calculation and writes information resolving to the output interface into the URI of the Record-Route header. When handling responses, the proxy must inspect the Record-Route header(s), look for an input interface, and selectively edit them to reference the correct output interface. Since this lookup has to be done for all responses forwarded by the proxy, this technique implies a CPU drag.

Therefore, this document recommends using the double Record-Route approach to avoid rewriting the Record-Route. This recommendation applies to all uses of Record-Route rewriting by proxies, including transport protocol switching and multi-homed proxies.

5. Double Record-Routing

The serious drawbacks of the rewriting technique explain why the double Record-Routing solution has consequently been recommended in SIP extensions like [RFC3486] or [RFC3608].

This technique consists of inserting before any existing Record-Route header, a Record-Route header with the URI reflecting to the input interface, including schemes and/or URI parameters, and secondly, a Record-Route header with the URI reflecting to the output interface. When processing the response, no modification of the recorded route is required. This is completely backward compatible with [RFC3261]. Generally speaking, the time complexity will be less in double

Record-Routing since, on processing the response, the proxy does not have to do any rewrites (and thus, no searching). Moreover, the handling of in-dialog requests and responses requires no special handling anymore.

When double Record-Routing, the proxy will have to handle the subsequent in-dialog request(s) as a spiral, and consequently devote resources to maintain transactions required to handle the spiral. What is considered to be a spiraling request is explained in Section 6 of [RFC3261]. In order to avoid a spiral, the proxy can be smart and scan an extra Route header ahead to determine whether the request will spiral through it. If it does, it can optimize the second spiral through itself. Even though this is an implementation decision, it is much more efficient to avoid spiraling. So, this means, in Section 16.4, "Route Information Preprocessing" [RFC3261], implementors can choose that a proxy MAY remove two Route headers instead of one when using the double Record-Routing.

The following example is an extension of the example given in [V6Tran]. It illustrates a basic call flow using double Record-Routing in a multi-homed IPv4 to IPv6 proxy, and annotates the dialog state on each User Agent (UA). In this example, proxy P1, responsible for the domain biloxy.example.com, receives a request from an IPv4-only upstream client. It proxies this request to an IPv6-only downstream server. Proxy P1 is running on a dual-stack host; on the IPv4 interface, it has an address of 192.0.2.254. On the IPv6 interface, it is configured with an address of 2001:db8::1. Some mandatory SIP headers have been omitted to ease readability.

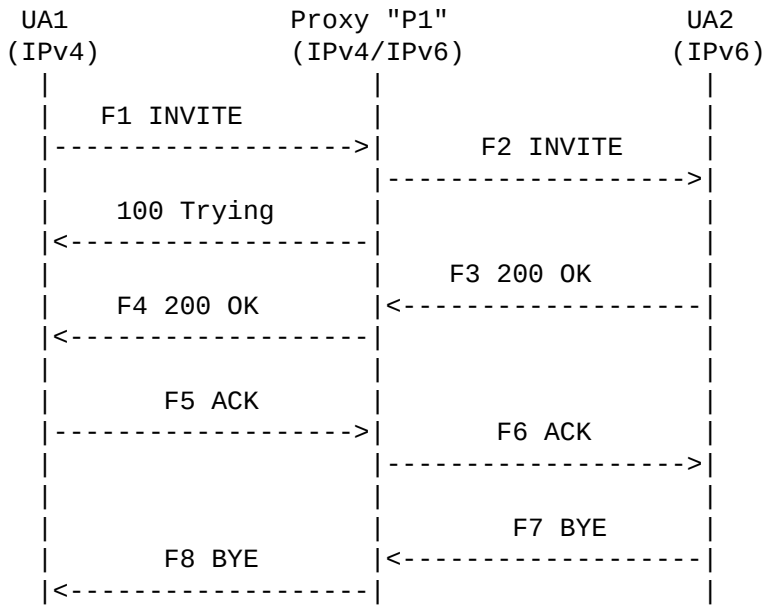

Figure 2: IPv4 to IPv6 Multi-Homed Proxy Illustration

```
F1 INVITE UA1 -> P1 (192.0.2.254:5060)

INVITE sip:bob@biloxi.example.com SIP/2.0
Route: <sip:192.0.2.254:5060;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=1234
To: Bob <sip:bob@biloxi.example.com>
Contact: <sip:alice@192.0.2.1>

        F2 INVITE P1 (2001:db8::1) -> UA2

        INVITE sip:bob@biloxi.example.com SIP/2.0
        Record-Route: <sip:[2001:db8::1];lr>
        Record-Route: <sip:192.0.2.254:5060;lr>
        From: Alice <sip:alice@atlanta.example.com>;tag=1234
        To: Bob <sip:bob@biloxi.example.com>
        Contact: <sip:alice@192.0.2.1>

                Dialog State at UA2:
                Local URI     = sip:bob@biloxi.example.com
                Remote URI    = sip:alice@atlanta.example.com
                Remote target = sip:alice@192.0.2.1
                Route Set     = sip:[2001:db8::1];lr
                                sip:192.0.2.254:5060:lr
```

```
                  F3 200 OK UA2 -> P1 (2001:db8::1)

                  SIP/2.0 200 OK
                  Record-Route: <sip:[2001:db8::1];lr>
                  Record-Route: <sip:192.0.2.254:5060;lr>
                  From: Alice <sip:alice@atlanta.example.com>;tag=1234
                  To: Bob <sip:bob@biloxi.example.com>;tag=4567
                  Contact: <sip:bob@[2001:db8::33]>

          F4 200 OK P1 -> UA1

          SIP/2.0 200 OK
          Record-Route: <sip:[2001:db8::1];lr>
          Record-Route: <sip:192.0.2.254:5060;lr>
          From: Alice <sip:alice@atlanta.example.com>;tag=1234
          To: Bob <sip:bob@biloxi.example.com>;tag=4567
          Contact: <sip:bob@[2001:db8::33]>

   Dialog State at UA1:
   Local URI     = sip:alice@atlanta.example.com
   Remote URI    = sip:bob@biloxi.example.com
   Remote target = sip:bob@[2001:db8::33]
   Route Set     = sip:192.0.2.254:5060:lr
                   sip:[2001:db8::1];lr

   F5 ACK UA1 -> P1 (192.0.2.254:5060)

   ACK sip:bob@[2001:db8::33] SIP/2.0
   Route: <sip:192.0.2.254:5060:lr>
   Route: <sip:[2001:db8::1];lr>
   From: Alice <sip:alice@atlanta.example.com>;tag=1234
   To: Bob <sip:bob@biloxi.example.com>;tag=4567

          F6 ACK P1 (2001:db8::1) -> UA2

          ACK sip:bob@[2001:db8::33] SIP/2.0
          From: Alice <sip:alice@atlanta.example.com>;tag=1234
          To: Bob <sip:bob@biloxi.example.com>;tag=4567
          (both Route headers have been removed by the proxy)

                  F7 BYE UA2 -> P1 (2001:db8::1)

                  BYE sip:alice@192.0.2.1 SIP/2.0
                  Route: <sip:[2001:db8::1];lr>
                  Route: <sip:192.0.2.254:5060:lr>
                  From: Bob <sip:bob@biloxi.example.com>;tag=4567
                  To: Alice <sip:alice@atlanta.example.com>;tag=1234
```

```
          F8 BYE P1 (192.0.2.254:5060) -> UA1

          BYE sip:alice@192.0.2.1 SIP/2.0
          From: Bob <sip:bob@biloxi.example.com>;tag=4567
          To: Alice <sip:alice@atlanta.example.com>;tag=1234
```

Figure 3: Multi-Homed IPv4 to IPv6 Double Record-Routing Illustration

6. Usage of Transport Protocol Parameter

This section describes a set of problems that is related to the usage of transport protocol URI parameters in the Record-Route header. In some circumstances, interoperability problems occur because it is not clear whether or not to include the transport parameter on the URI of the Record-Route header. This was identified as a frequent problem in past SIPit events.

[RFC3261], step 8 of Section 16.7 says:

> The URI SHOULD NOT contain the transport parameter unless the proxy has knowledge (such as in a private network) that the next downstream element that will be in the path of subsequent requests supports that transport.

The preceding seems to confuse implementors, resulting in proxies that insert a single Record-Route without a transport URI parameter, resulting in the problems described in this section.

6.1. UA Implementation Problems and Recommendations

Consider the following scenario: a SIP proxy, doing TCP to UDP transport protocol switching.

In this example, proxy P1, responsible for the domain biloxy.example.com, receives a request from Alice UA1, which uses TCP. It proxies this request to Bob UA2, which registered with a Contact specifying UDP as transport protocol. Thus, P1 receives an initial request from Alice over TCP and forwards it to Bob over UDP. For subsequent requests, it is expected that TCP could continue to be used between Alice and P1, and UDP between P1 and Bob, but this can not happen if a numeric IP address is used and no transport parameter is set on Record-Route URI. This happens because of procedures described in [RFC3263]. Some mandatory SIP headers have been omitted to ease readability.

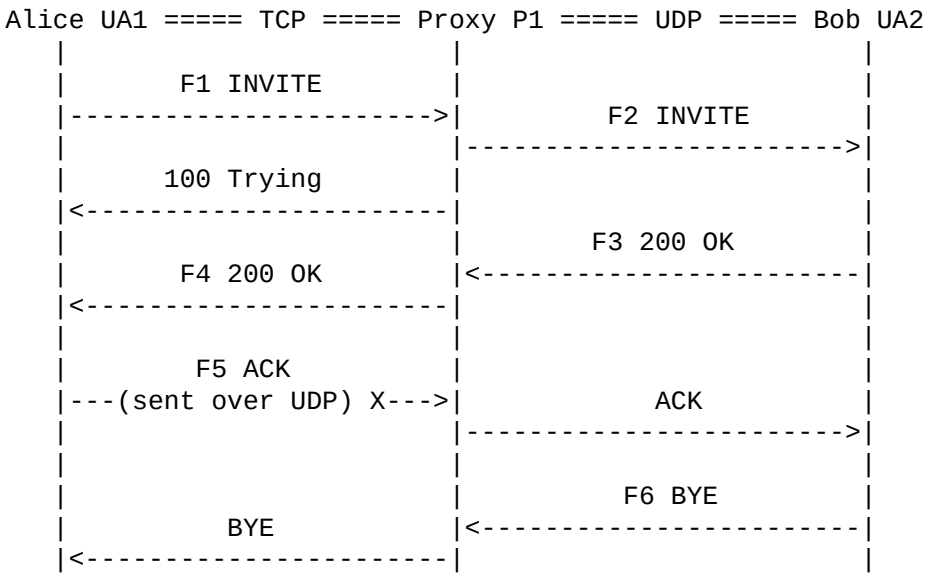

Figure 4: TCP to UDP Transport Protocol
Switching Issue Illustration

```
F1 INVITE UA1 -> P1 (192.0.2.1/tcp)

INVITE sip:bob@biloxi.example.com SIP/2.0
Route: <sip:192.0.2.1;lr;transport=tcp>
From: Alice <sip:alice@atlanta.example.com>;tag=1234
To: Bob <sip:bob@biloxi.example.com>
Contact: <sip:alice@ua1.atlanta.example.com;transport=tcp>

     F2 INVITE P1 -> UA2 (ua2.biloxi.example.com/udp)

     INVITE sip:bob@ua2.biloxi.example.com;transport=udp SIP/2.0
     Record-Route: <sip:192.0.2.1;lr> (NO transport param)
     From: Alice <sip:alice@atlanta.example.com>;tag=1234
     To: Bob <sip:bob@biloxi.example.com>
     Contact: <sip:alice@ua1.atlanta.example.com;transport=tcp>

     Dialog State at UA2:
     Local URI     = sip:bob@biloxi.example.com
     Remote URI    = sip:alice@atlanta.example.com
     Remote target = sip:alice@ua1.atlanta.example.com;transport=tcp
     Route Set     = sip:192.0.2.1;lr
```

```
            F3 200 OK UA2 -> P1 (192.0.2.1/udp)

            SIP/2.0 200 OK
            Record-Route: <sip:192.0.2.1;lr>
            From: Alice <sip:alice@atlanta.example.com>;tag=1234
            To: Bob <sip:bob@biloxi.example.com>;tag=4567
            Contact: <sip:bob@ua2.biloxi.example.com>

      F4 200 OK P1 -> UA1 (ua1.atlanta.example.com/tcp)

      SIP/2.0 200 OK
      Record-Route: <sip:192.0.2.1;lr>
      From: Alice <sip:alice@atlanta.example.com>;tag=1234
      To: Bob <sip:bob@biloxi.example.com>;tag=4567
      Contact: <sip:bob@ua2.biloxi.example.com>

Dialog State at UA1:
Local URI     = sip:alice@atlanta.example.com
Remote URI    = sip:bob@biloxi.example.com
Remote target = sip:bob@ua2.biloxi.example.com
Route Set     = sip:192.0.2.1;lr


F5 ACK UA1 -> P1 (192.0.2.1/udp)

ACK sip:bob@ua2.biloxi.example.com SIP/2.0
Route: <sip:192.0.2.1;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=1234
To: Bob <sip:bob@biloxi.example.com>;tag=4567

            F6 BYE UA2 -> P1 (192.0.2.1/udp)

            BYE sip:alice@ua1.atlanta.example.com;transport=tcp SIP/2.0
            Route: <sip:192.0.2.1;lr>
            From: Bob <sip:bob@biloxi.example.com>;tag=4567
            To: Alice <sip:alice@atlanta.example.com>;tag=1234
```

Figure 5: TCP to UDP Transport Protocol Switching Issue Description

Since the proxy P1 does not insert any transport parameter in the Record-Route URI, subsequent in-dialog requests of UA1, like the ACK sent in F5, will be sent according to the behavior specified in Section 12.2 (requests within a Dialog) of [RFC3261]. That means that the routeset is used, and then, applying [RFC3263], the Route "sip:192.0.2.1" will resolve to a UDP transport by default (since no transport parameter is present here), and no Naming Authority Pointer (NAPTR) request will be performed since this is a numeric IP address.

In general, the interoperability problems arise when UA1 is trying to send the ACK: it is not ready to change its transport protocol for a mid-dialog request and just fails to do so, requiring the proxy implementor to insert the transport protocol in the Record-Route URI.

What happens if the proxy had Record-Routed its logical name (biloxi.example.com)? Since Bob is to be contacted over UDP, protocol switching will be avoided only if the resulting transport protocol of [RFC3263] procedures is UDP. For any other resulting transport protocol, the transport protocol switching issue described above will occur. Also, if one of the UAs sends an initial request using a different transport than the one retrieved from DNS, this scenario would be problematic.

In practice, there are multiple situations where UA implementations don't use logical names and NAPTR records when sending an initial request to a proxy. This happens, for instance, when:

1) UAs offer the ability to "choose" the transport to be used for initial requests, even if they support [RFC3263]. This is a frequent UA functionality that is justified by the following use cases:

   - when it is not possible to change the DNS server configuration and the implementation doesn't support all the transport protocols that could be configured by default in DNS (e.g., TLS).

   - when the end-user wants to choose his transport protocol for whatever reason, e.g., needing to force TCP, avoiding UDP/congestion, retransmitting, or fragmenting, etc.

This ability to force the transport protocol in UAs for initial requests SHOULD be avoided: selecting the transport protocol in the configuration of an outbound proxy means that [RFC3263] procedure is bypassed for initial requests. As a consequence, if the proxy Record-Routed with no transport parameter as is recommended in [RFC3261], the UA will be forced to use the [RFC3263]-preferred transport for subsequent requests anyway, which leads to the problematic scenario described in Figure 4.

2) UAs decide to always keep the same transport for a given dialog. This choice is erratic, since if the proxy is not Record-Routing, the callee MAY receive the subsequent request through a transport that is not the one put in its Contact. If a UA really wants to avoid transport protocol switching between the initial and subsequent request, it SHOULD rely on DNS records for that; thus,

it SHOULD avoid configuring statically the outbound proxy with a numeric IP address. A logical name, with no transport parameter, SHOULD be used instead.

3) UAs don't support [RFC3263] at all, or don't have any DNS server available. In that case, as illustrated previously, forcing UA1 to switch from TCP to UDP between initial request and subsequent request(s) is clearly not the desired default behavior, and it typically leads to interoperability problems. UA implementations SHOULD then be ready to change the transport protocol between initial and subsequent requests. In theory, any UA or proxy using UDP must also be prepared to use TCP for requests that exceed the size limit of path MTU, as described in Section 18.1.1 of [RFC3261].

6.2. Proxy Implementation Problems and Recommendations

In order to prevent UA implementation problems, and to maintain a reasonable level of interoperability, the situation can be improved on the proxy side. Thus, if the transport protocol changed between its incoming and outgoing sides, the proxy SHOULD use the double Record-Route technique and SHOULD add a transport parameter to each of the Record-Route URIs it inserts. When TLS is used on the transport on either side of the proxy, the URI placed in the Record-Route header field MUST encode a next-hop that will be reached using TLS. There are two ways for this to work. The first way is for the URI placed in the Record-Route to be a SIPS URI. The second is for the URI placed in the Record-Route to be constructed such that application of [RFC3263] resolution procedures to that URI results in TLS being selected. Proxies compliant with this specification MUST NOT use a "transport=tls" parameter on the URI placed in the Record-Route because the "transport=tls" usage was deprecated by [RFC3261]. Record-Route rewriting MAY also be used. However, the recommendation to put a transport protocol parameter on Record-Route URI does not apply when the proxy has changed the transport protocol due to the size of UDP requests as per Section 18.1.1 of [RFC3261]. As an illustration of the previous example, it means one of the following processing will be performed:

- Double Record-Routing: the proxy inserts two Record-Route headers into the SIP request. The first one is set, in this example, to Record-Route: <sip:192.0.2.1;lr;transport=tcp>, the second one is set to Record-Route: <sip:192.0.2.1;lr> with no transport, or with transport=udp, which basically means the same thing.

- Record-Route rewriting on responses: in the INVITE request sent in F2, the proxy puts the outgoing transport protocol in the transport parameter of Record-Route URI. Doing so, UA2 will correctly send

its BYE request in F6 using the same transport protocol as previous messages of the same dialog. The proxy rewrites the Record-Route when processing the 200 OK response, changing the transport parameter "on the fly" to "transport=tcp", so that the Route set will appear to be <sip:192.0.2.1;lr;transport=tcp> for UA1 and <sip:192.0.2.1;lr;transport=udp> for UA2.

It is a common practice in proxy implementations to support double Record-Route AND to insert the transport parameter in the Record-Route URI. This practice is acceptable as long as all SIP elements that may be in the path of subsequent requests support that transport. This restriction needs an explanation. Let’s imagine you have two proxies, P1 at "p1.biloxi.example.com" and P2 on the path of an initial request. P1 is Record-Routing and changes the transport from UDP to Stream Control Transmission Protocol (SCTP) because the P2 URI resolves to SCTP applying [RFC3263]. Consequently, the proxy P1 inserts two Record-Route headers:

Record-Route: <sip:p1.biloxi.example.com;transport=udp> and

Record-Route: <sip:p1.biloxi.example.com;transport=sctp>.

The problem arises if P2 is not Record-Routing, because the SIP element downstream of P2 will be asked to reach P1 using SCTP for any subsequent, in-dialog request from the callee, and this downstream SIP element may not support that transport.

In order to handle this situation, this document recommends that a proxy SHOULD apply the double Record-Routing technique as soon as it changes the transport protocol between its incoming and outgoing sides. If proxy P2 in the example above would follow this recommendation, it would perform double Record-Routing and the downstream element would not be forced to send requests over a transport it does not support.

By extension, a proxy SHOULD also insert a Record-Route header for any multi-homed situation (as the ones described in this document: scheme changes, sigcomp, IPv4/IPv6, transport changes, etc.) that may impact the processing of proxies being on the path of subsequent requests.

7. Conclusion

As a conclusion of this document, it is to notice that:

- Record-Route rewriting is presented as a technique that MAY be used, with the drawbacks outlined in Section 4.

- Double Record-Routing is presented as the technique that SHOULD be used, and is documented in Section 5.

- Record-Route header interoperability problems on transport protocol switching scenarios have been outlined and described in Section 6. This last section gives some recommendations to UA and proxy implementations to improve the situation. Proxies SHOULD use double Record-Routing for any multi-homed situation that MAY impact the further processing, and they SHOULD put transport protocol parameters on Record-Route URIs in some circumstances. UAs SHOULD NOT offer options to overwrite the transport for initial requests. Further, UAs SHOULD rely on DNS to express their desired transport and SHOULD avoid IP addresses with transport parameters in this case. Finally, UAs SHOULD be ready to switch transports between the initial request and further in-dialog messages.

8. Security Considerations

The recommendations in this document describe a way to use the existing protocol specified in RFC 3261 rather than introducing any new protocol mechanism. As such, they do not introduce any new security concerns, but additional consideration of already existing concerns is warranted. In particular, when a message is transiting two interfaces, the double Record-Route technique will carry information about both interfaces to each of the involved endpoints (and any intermediaries between this proxy and those endpoints), where the rewriting technique would only expose information about the interface closest to each given endpoint. If issues such as topology hiding or privacy (as described in [RFC3323]) are a concern, the URI values placed in the Record-Route for each interface should be carefully constructed to avoid exposing more information than was intended.

9. Acknowledgments

Thank you to Dean Willis, Vijay K. Gurbani, Joel Repiquet, Robert Sparks, Jonathan Rosenberg, Cullen Jennings, Juha Heinanen, Paul Kyzivat, Nils Ohlmeier, Tim Polk, Francois Audet, Adrian Farrel, Ralph Droms, Tom Batsele, Yannick Bourget, Keith Drage, and John Elwell for their reviews and comments.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3261] 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.

[RFC3263] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[RFC3323] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.

[RFC5630] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, October 2009.

10.2. Informative References

[BUG664] Sparks, RS., "Bug 664: Double record routing, http://bugs.sipit.net/show_bug.cgi?id=664", October 2002.

[RFC3486] Camarillo, G., "Compressing the Session Initiation Protocol (SIP)", RFC 3486, February 2003.

[RFC3608] Willis, D. and B. Hoeneisen, "Session Initiation Protocol (SIP) Extension Header Field for Service Route Discovery During Registration", RFC 3608, October 2003.

[V6Tran] Camarillo, G., El Malki, K., and V. Gurbani, "IPv6 Transition in the Session Initiation Protocol (SIP)", Work in Progress, August 2009.

Authors' Addresses

Thomas Froment
Tech-invite

EMail: thomas.froment@tech-invite.com

Christophe Lebel
Alcatel-Lucent
Lieu dit Le Mail
Orvault 44708
France

EMail: christophe.lebel@alcatel-lucent.com

Ben Bonnaerens
Alcatel-Lucent
Copernicuslaan 50
Antwerpen 2018
Belgium

EMail: ben.bonnaerens@alcatel-lucent.com