Construction of the Route Header Field in the Session Initiation Protocol (SIP)
draft-rosenberg-sip-route-construct-02

Status of this Memo

Copyright Notice

Abstract

The Route header field in the Session Initiation Protocol (SIP) is used to cause a request to visit a set of hops on its way towards the final destination. Several specifications have defined rules for how a user agent obtains and then uses a set of Route header fields in the transmission of a request. These include the SIP specification itself, the Service-Route header field specification, the SIP server option in the Dynamic Host Configuration Protocol (DHCP), and others. Unfortunately, these specifications are not consistent and the

resulting behavior at clients and servers is not clear or complete. This document resolves this problem by defining a consistent set of logic, and in the process, serves as an update to the Service-Route specification.

Table of Contents

1. Introduction

The Route header field in the Session Initiation Protocol (SIP) protocol is used to cause a request to visit a set of hops on its way towards the final destination. RFC 3261 [2] discusses how a client constructs the Route header field for requests. However, this logic is restricted to mid-dialog requests, where the route set was learned as a result of record-routing.

However, additional sources of routes can exist for a UA. These include default outbound proxies, a service route learned from the Service-Route header field [3], and a redirection utilizing loose routing [7]. In total, there are four sources of potential route headers. The way in which these various sources are reconciled is unclear. Furthermore, the various specifications are unclear about which requests these Route headers are applicable to. Do they apply to REGISTER? Do they apply to mid-dialog requests? Finally, RFC 3608 is underspecified, which can result in interoperability problems.

Section 2 reviews the existing sources of route sources. Section 3 discusses problems with the existing specifications. Section 5 overviews the proposed changes in behavior. Section 6 provides a detailed description of element behavior, and Section 7 defines the grammar for the new parameters specified here.

2. Existing Sources

This section examines the current set of route header field sources.

2.1. Default Outbound Proxies

RFC 3261 discusses default outbound proxies. In Section 8.1.1.1, it makes reference to its interaction with Route header fields:

In some special circumstances, the presence of a pre-existing route set can affect the Request-URI of the message. A pre-existing route set is an ordered set of URIs that identify a chain of servers, to which a UAC will send outgoing requests that are outside of a dialog. Commonly, they are configured on the UA by a user or service provider manually, or through some other non-SIP mechanism. When a provider wishes to configure a UA with an outbound proxy, it is RECOMMENDED that this be done by providing it with a pre-existing route set with a single URI, that of the outbound proxy.

When a pre-existing route set is present, the procedures for populating the Request-URI and Route header field detailed in Section 12.2.1.1 MUST be followed (even though there is no dialog), using the desired Request-URI as the remote target URI.

The default outbound proxy can be learned either through DHCP [8], through configuration (such as the SIP configuration framework [9]), or through other means. In the IP Multimedia Subsystem (IMS), the default outbound proxy is the P-CSCF and is learned through GPRS specific techniques.

RFC 3261 does not explicitly say the set of messages to which this route set applies. However, the text above implies that it is for all requests outside of a dialog.

2.2. Service Route

RFC 3608 specifies the Service-Route header field. This header field is provided to the UA in a 2xx response to a REGISTER request. The client uses this to populate its Route header fields for outgoing requests. However, RFC 3608 explicitly says that the decision a UA makes about how it combines the service route with other outbound routes is a matter of local policy. Furthermore, RFC 3608 does not clearly define to which requests the service route applies, and in particular, whether or not it applies to a REGISTER request or a mid-dialog request.

Furthermore, RFC 3608 specifies that a service-route is associated with an Address-of-Record (AOR), and is shared by all contacts

associated with the same AOR. It also specifies that the Service-Route URI can only be ones known to the registrar apriori, as opposed to learned through the registration itself.

2.3. Record-Routes

RFC 3261 provides a detailed description of the record-routing mechanism, and how the user agents in a dialog construct route sets from the Record-Route header field values. RFC 3261 is also clear that the resulting route set applies to mid-dialog requests. It implies (though does not explicitly say) that the resulting route set overrides any default outbound proxies (which represent a pre-loaded route set).

2.4. Loose Routes

Loose routing, introduced in [7], defines mechanisms for using Route header fields to address and invoke services in a user agent. It also specifies a redirection mechanism whereby a server can redirect

a client, and ask it to either modify the topmost Route header field of its request, or add a new Route header field to the topmost request. The specification indicates that it is applicable to both mid-dialog and out-of-dialog requests. Since the client can be a user agent, this forms another potential source of a Route header field for user agents.

3. Problems with Current Specifications

Numerous problems have arisen as a consequence of the combination of these specifications. These problems fit into two categories. The first are interoperability problems, and the second are missing capabilities.

An interoperability problem that has arisen is keeping an outbound proxy on the path for outbound requests. Consider a proxy in a hotel which a client discovers via DHCP and uses as its outbound proxy. This proxy wishes to be used for incoming and outgoing requests, both in and out of a dialog. If the home proxy provides a service route, the hotel proxy will not be able to determine what it needs to do in order to stay on the path. If the client implementation is such that

it appends the service route to its default outbound proxy, then the hotel proxy need not do anything to stay on the path. If, however, the client abandons its default proxy in favor of the service route, the hotel proxy would fall off the path of subsequent requests unless it inserted a Service-Route into the response of a REGISTER request. Interestingly, the latter is illegal behavior according to RFC 3608, but is the only mechanism available for ensuring that a proxy stay on the request path. Since RFC 3608 does not provide any normative behavior for combining service routes and outbound proxies, the hotel proxy cannot know what to do, thus causing the interoperability problem.

This points to the first major functional gap with RFC 3608. There is no standards-based way for keeping an outbound proxy on the path for outbound requests, when it is not the default outbound proxy. Consider a proxy in a hotel, PH-1 which a client discovers via DHCP and uses as its outbound proxy. When the client sends a REGISTER to this proxy, it forwards it to a second proxy in the hotel, PH-2, which then forwards it to the home proxy of the user, PA. PH-2 needs to be on the outbound path for all requests leaving the hotel. PA includes itself in a Service-Route header field in the response. The client receives this Service-Route. For an initial INVITE request, the client constructs a route set which includes its outbound proxy PH-1 followed by the URI from the Service-Route, PA. This request will traverse PH-1, which now follows the next Route header, sending it to PA. This is not the desired behavior. The problem is that the

Service-Route URI has provided a route that overrides the default outbound route behavior at PH-1.

Similarly, there is no way in RFC 3608 to change the outbound proxy, outside of an update in the client configuration. Such changes are extremely useful for many operational reasons. One example is movement of subscribers between geographically distributed sites in cases where a site must be gracefully taken out of service, and the subscribers using it need to be moved gracefully, over a period of an hour or two, from one site to the other. Since, at best, the impact of Service-Route on the outbound proxy is ambiguous, there is generally no way to affect it excepting configuration change. Using configuration updates as the only way to alter the outbound proxy is problematic. In practice, systems providing automated updates to client configuration (when they exist, as they often do not) are

decoupled from the operational systems that manage subscriber capacity and software upgrades of sites, making the change hard to affect through configuration. Furthermore, configuration updates are typically passed to clients once they are made. Here, however, the objective is to gracefully move subscribers. Using the randomized nature of REGISTER timings helps provide that; such a function is difficult to accomplish through configuration updates. Finally, many deployments use mechanisms other than [9] for updating client configurations. As a consequence, there is no common way across deployments to provide this very basic operational feature.

Another problem that has come up with RFC 3608 is that it will not work well with mid-dialog failover techniques identified for SIP Outbound [10]. These techniques require that the outbound proxy construct a URI for the Service-Route that is used by the UA for new requests outside of a dialog.

Finally, RFC 3608 is defined such that the service route is identical for all contacts registered to a specific AOR. This makes it applicable only for selecting a set of configured, well-known servers to use, and only ones within the domain of the owner of that AOR. This is a fairly narrow scope of applicability, and introduces a configuration burden on the registrar.

Architecturally, there is an inconsistency between record-routing and service route. With record-route, each proxy on the path of the request inserts a Record-Route header field, and this dictates the path of subsequent messages within a dialog both to and from the UA. With REGISTER, each proxy on the path of the request inserts a Path [4] header field to receive requests directed towards the client. However, the Service-Route is not the inverse of the Path, and is instead created through proprietary means by the registrar.

4. 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].

5. Overview of Operation

This specification updates the behaviors in RFC 3608. In particular, a registrar, upon receipt of a REGISTER, uses the Path header field values to construct the Service-Route in the response. In addition, the registrar retains an instance of the Path (and resulting Service Route) for each registered contact. The Path and Service-Route remain valid for the duration of the registration, and are updated for each registration refresh.

In order to retain backwards compatibility with systems based on RFC 3608, proxies compliant to this specification include a flag, "p2sr", in their Path header field values. When the registrar receives the REGISTER request, it examines the sequence of Path header field URI. If it sees that one or more contiguous proxies at the end of the Path sequence do not support this mechanism, the registrar omits those URI from the Service-Route, and omits the Require header field parameter indicating support for this specification in the response. This causes the UA to revert to existing behavior, augmenting the route set with the outbound proxy [[OPEN ISSUE: well, thats true for IMS at least. UA behavior isn't defined at all in this area in RFC 3608. Alternative is to have two option tags - one that says to augment, and one that says don't.]] If, however, all of the Path URI include the "p2sr" flag, an option tag is placed into the Require header field is placed in the response, indicating that the Service-Route overrides the outbound proxy.

The rules for constructing the Route for a request at the UA follow in a straightforward manner from this. Mid-dialog requests always use the set of URI learned from the Record-Route. A request outside of the scope of a dialog, including a REGISTER refresh, uses the Service-Route, and based on the Require flag, may or may not override the outbound proxy. Finally, in all cases, if a request generates a redirect response that contains a loose route, the Route set is further modified or augmented based on the redirection.

6. Detailed Processing Rules

6.1. Registrar Behavior

This specification updates the procedures from RFC 3608.

The procedures in this specification MUST NOT be followed unless the REGISTER request contains a Supported header field with the "sr" option tag.

Assuming the REGISTER request contains this option tag, the registrar examines the set of Path header field values, starting from the top (the proxy closes to, but not including the registrar itself) towards the bottom (the proxy farthest away from the registrar). If the registrar is planning on adding itself to the Service-Route, it adds itself to the top of the list. Its own URI MUST include the "p2sr" Path header field parameter.

If the resulting list is such that there are 0 or more contiguous values starting at the top which contain the "p2sr" Path header field parameter, followed by 0 or more contiguous values which do not contain this parameter, the registrar SHOULD follow the remaining procedures of this specification in the construction of the Service-Route header field in the response. If not, the procedures defined here MUST NOT be used. In addition, if none of the Path header field values contain the "p2sr" Path header field parameters, the procedures defined here MUST NOT be used.

Consider the example network of Figure 1. The UAC is separated from the registrar by three proxies, P1, P2 and P3. The UAC supports the mechanism in this specification and indicates this through the "sr" option tag in the Supported header field of its REGISTER request.

```
+-------+      +-------+       +-------+       +-------+       +-------+
|       |      |       |       |       |       |       |       |       |
| UAC   |----->| P1    |------>| P2    |------>| P3    |------>| Reg   |
|       |      |       |       |       |       |       |       |       |
+-------+      +-------+       +-------+       +-------+       +-------+
```

Figure 1

When the UAC registers, each of the proxies inserts itself onto the Path header field of the REGISTER. Each of the proxies either supports this extension (and thus inserts a "ps2r" parameter into its Path header field value) or it does not (in which case no parameter is inserted). The following table shows, for various Path sequences, whether or not the modified Service-Route procedures of this specification would be used.

Path Header Field	Use New SR Procedures?	Notes
P3;p2sr, P2;p2sr, P1;p2sr	Y	All proxies support it
P3;p2sr, P2;p2sr, P1	Y	Proxies closest to registrar support, followed by ones that don't
P3;p2sr, P2 P1	Y	Proxies closest to registrar support, followed by ones that don't
P3, P2, P1	N	No proxies support it
reg;p2sr P3, P2, P1	Y	Registrar planning on inserting itself onto the Service-Route
P3, P2;p2sr, P1	N	Set of proxies that support it must be contiguous and closest to registrar
P3;p2sr, P2, P1;p2sr	N	Set of proxies that support it must be contiguous and closest to registrar

Figure 2

> This constraint basically says that the Path has to be built either by a proxy chain which all support this spec, or by a chain whereby a bunch didn't support it, followed by a bunch that did. This works well in IMS deployments where the visited network doesn't support the mechanism, but the home network does.

If the mechanisms in this specification are to be used, the registrar MUST construct the Service-Route in the registration response by taking each URI from the list which contained the "p2sr" header field parameter, and inverting the order. The registrar MUST add an option tag to the Require header field in the response (adding the header

field if necessary) with the value "sr". The URI in the Service-Route header field values SHOULD NOT contain the "p2sr" parameter;

that parameter is a Path header field value and is not needed in the Service-Route.

The resulting Service-Route MUST be recomputed for each registration refresh, and for each new registration. The server MAY store the values associated with it, though this is not necessary for proper operation of this specification.

In addition, the registrar MUST only return in a 200 OK response to the REGISTER request, the Contact header field associated with the registration which was just performed. [[OPEN ISSUE: This is really orthogonal, and it is probably controversial. Basically it proposes to use this new service route mechanism as a vehicle for eliminating query registers and third party registrations.]]. A UA compliant to this specification will never generate a registration with anything except for a single Contact.

If the mechanisms in this specification are not used, the registrar MUST follow the procedures of RFC 3608 and construct the Service-Route as it would otherwise. It MUST omit the "sr" option tag from a Require header field in the response.

6.2. Proxy Behavior

This specification updates the proxy processing rules in RFC 3608.

A proxy compliant to this specification which inserts a Path header field value into a REGISTER request MUST include a "p2sr" Path header field parameter with its value. If the response to the REGISTER request includes the Require header field that includes the "sr" option tag, it means that the UA will be using that URI (which will be echoed in the Service-Route) as a Route header field value for requests outside of a dialog. In this case, the proxy MAY remove its value from the Service-Route in the response, or MAY modify it.

When the UA initiates a request outside of a dialog, that request will contain a route set which includes the URIs learned from the Service-Route. Consequently, a proxy MUST be prepared to receive such a request, in which case the topmost Route header will be the

URI the proxy passed to the UA in the 200 OK response to REGISTER.

6.3. UAC Behavior

6.3.1. REGISTER Processing

A UA compliant to this specification MUST include the "sr" option tag in the Supported header field of its REGISTER request. Such a UA MUST include only a single Contact in each REGISTER request, which

points to the UA performing the registration. It MUST NOT generate a "query REGISTER" which contains no contacts, MUST NOT include multiple Contact header field values in its registration, and MUST NOT register a Contact which does not directly point to the UA itself.

When the REGISTER response arrives, and it is a 2xx response, the UA looks for the presence of a Supported header field in the response with the "sr" option tag. If present, the UA is operating in "override" mode, as described below. If not present, the UA is operating in "augment" mode, as described below. In either case, the UA MUST cache the contents of the Service-Route header field for the duration of its registration.

A single UA may still be performing multiple registrations, for purposes of high availability, as a consequence of the procedures defined in SIP outbound [6]. In this case, the UA will end up with multiple sets of Service-Route, each of which is bound to the particular flow that was registered (and its associated Contact).

6.3.2. Request Origination

It is RECOMMENDED that a UA compliant to this specification also be compliant to UA loose routing [7]. This allows proxies to utilize a redirection to further augment the way in which the route set for a request is constructed.

The primary question addressed by this specification is how the UA constructs the route set for a request.

Determination of the route set for a request depends on whether the request is generated as a consequence of a redirection. If the UA

indicated support for loose routing in its request (as described in [7], the Route set for the recursed request is generated from the request which generated the recursion, as described there.

This specification assumes that a UA may have one or more configured outbound proxies, each in the form of a SIP URI. Each of these will either be a loose route (in which case the request would contain that URI in the Route header field) or not (in which case the UA will just send the request to that target without including its URI in the topmost Route request).

For a request sent by a UAC that is not the result loose route recursion, the following logic MUST be used to compute the route set:

- If the request is a mid-dialog request, the route set is computed per the procedures in Section 12.2.1.1 of RFC 3261. The route set will not contain any routes learned from configuration, DHCP, Service-Route or any other mechanism.

- If the request is not a mid-dialog request, the client checks to see if it has learned a service route as a result of registration. The UAC may have learned numerous service routes, one for each unique AOR/Contact that it registered. In the case of registrations using the mechanisms of [6], the Contact includes the flow ID and instance ID, so that the client may have a distinct service route for each unique AOR/Flow ID/Instance ID combination. As such, when sending a request, the client selects the service route corresponding to the contact which is sending the request. [[OPEN ISSUE: Need to say more about this selection.]]. If the UA is operating in "override" mode for this route set, the URIs from this service route become the route set. If the UA is operating in the "augment" mode for this route set, the UA takes the outbound proxy URI it used for the REGISTER request which created the route set, and appends that URI to the top of the service route.

- If the request is not a mid-dialog request, and there are are no service routes associated with the contact generating the request, the UAC uses the route set learned through configuration. [[OPEN ISSUE: Do we need to specify how to reconcile route sources learned across disparate configuration sources? For example DHCP

and a config file? These can come from different providers.]]

If the topmost URI in the resulting route set is not a loose route (which can happen when there is a configured outbound proxy that is not a loose route), the UA MUST remove that URI from the Route set, but still use it for purposes of sending the request.

7. Grammar

This specification defines an option tag and a Path header field parameter. Their syntax is as follows:

```
option-tag      \= "sr"
rr-param        \= "p2sr"
```

8. Security Considerations

The route set used by a user agent for generating initial requests into the network is very sensitive information. If this information is manipulated by an attacker, it can cause calls to be directed

towards intermediaries, which can then observe call patterns, intercept communications, and so on. Consequently, a UA using this specification SHOULD use sips when performing a registration. This makes sure that only entities along the request path can modify the route set used by the UA.

Even with sips, it is possible that a malicious home proxy could modify the route set used by the UA, eliminating the outbound proxy that would otherwise be used by it. This kind of attack is only meaningful in environments where the outbound proxy is in a different domain than the home proxy. However, presumably the outbound proxy is present to authorize access to services, and removing it will only result in denial of service to the user, which would appear to provide no benefit.

9. IANA Considerations

This specification registers a new option tag and a new Path header field parameter.

9.1. SIP Option Tag

This specification registers a new SIP option tag, as per the guidelines in Section 27.1 of RFC 3261.

Name: sr

Description: This option tag is used to identify the usage of Path reflected Service-Route, as defined by RFC XXXX [[NOTE TO IANA: Please replace XXXX with the RFC number of this specification]]

9.2. Header Field Parameter

This specification defines the "p2sr" header field parameter, as per the registry created by [5]. The required information is as follows:

Header field in which the parameter can appear: Path

Name of the Parameter: p2sr

RFC Reference: RFC XXXX [[NOTE TO IANA: Please replace XXXX with the RFC number of this specification.]]

10. Examples

TODO.

11. Acknowledgements

The author would like to thank Paul Kyzivat and Anders Kristensen for their comments.

12. References

12.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement

Levels", BCP 14, RFC 2119, March 1997.

[2] 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.

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

[4] Willis, D. and B. Hoeneisen, "Session Initiation Protocol (SIP) Extension Header Field for Registering Non-Adjacent Contacts", RFC 3327, December 2002.

[5] Camarillo, G., "The Internet Assigned Number Authority (IANA) Header Field Parameter Registry for the Session Initiation Protocol (SIP)", BCP 98, RFC 3968, December 2004.

[6] Jennings, C. and R. Mahy, "Managing Client Initiated Connections in the Session Initiation Protocol (SIP)", draft-ietf-sip-outbound-04 (work in progress), June 2006.

[7] Rosenberg, J., "User Agent Loose Routing in the Session Initiation Protocol (SIP)", draft-rosenberg-sip-ua-loose-route-00 (work in progress), October 2006.

12.2. Informative References

[8] Schulzrinne, H., "Dynamic Host Configuration Protocol (DHCP-for-IPv4) Option for Session Initiation Protocol (SIP) Servers", RFC 3361, August 2002.

[9] Petrie, D., "A Framework for Session Initiation Protocol User Agent Profile Delivery", draft-ietf-sipping-config-framework-09 (work in progress), October 2006.

[10] Rosenberg, J., "Discovering Outbound Proxies and Providing High Availability with Client Initiated Connections in the Session Initiation Protocol (SIP)", draft-rosenberg-sip-outbound-discovery-mid-dialog-00 (work in

progress), October 2006.

Author's Address

Jonathan Rosenberg
Cisco Systems
600 Lanidex Plaza
Parsippany, NJ 07054
US

Phone: +1 973 952-5000
Email: jdrosen@cisco.com
URI: http://www.jdrosen.net

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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