Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 24, 2015

R. Sparks
Oracle

October 21, 2014

Explicit Subscriptions for the REFER Method
draft-sparks-sipcore-refer-explicit-subscription-02

Abstract

The Session Initiation Protocol (SIP) REFER request, as defined by RFC3515, triggers an implicit SIP-Specific Event Notification framework subscription. Conflating the start of the subscription with handling the REFER request makes negotiating SUBSCRIBE extensions impossible, and complicates avoiding SIP dialog sharing. This document defines extensions to REFER to remove the implicit subscription and, if desired, replace it with an explicit one.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 24, 2015.

Table of Contents

1. Conventions and Definitions

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 [RFC2119].

2. Introduction

REFER as defined by [RFC3515] triggers an implicit SIP-Specific Event Framework subscription. Sending a REFER within a dialog established by an INVITE results in dialog reuse and the associated problems described in [RFC5057]. The SIP-Specific Event Notification framework definition [RFC6665] disallows such dialog reuse. Call transfer, as defined in [RFC5589], thus requires sending a REFER request on a new dialog, associating it with an existing dialog using the 'Target-Dialog' mechanism defined in [RFC4538].

Because there is no explicit SUBSCRIBE request, the tools for negotiating subscription details are unavailable for REFER subscriptions. This includes negotiating subscription duration and providing information through Event header field parameters. The use of the SIP 'Supported' and 'Require' extension mechanisms [RFC3261] is complicated by the implicit subscription. It is unclear whether the extension applies to handling the REFER request itself, or to the messages in the subscription created by the REFER, or both. Avoiding this confusion requires careful specification in each extension. Existing extensions do not provide this clarity.

This document defines two mechanisms that remove the implicit subscription, one of which replaces it with an explicit one. The benefits of doing so include:

- Allowing REFER to be used within INVITE-created dialogs without creating dialog reuse.

- Allowing standard subscription parameter negotiation.

- Allowing standard negotiation of SIP extensions.

There are limitations on when it is appropriate to use the extension that allows an explicit subscription, related directly to definition of non-INVITE transaction handling SIP. These limitations are discussed in Section 4.1.

3. Overview

This section provides a non-normative overview of the behaviors defined in subsequent sections.

3.1. Explicit Subscriptions

A SIP User-Agent (UA) that wishes to issue a REFER request that will not create an implicit subscription, but will allow an explicit one, will include a new option tag, 'explicitsub', in the Require header

field of the REFER request. This REFER could be sent either within an existing dialog, or as an out-of-dialog request.

If the recipient of the REFER accepts the request, it will begin managing the 'refer' event state described in RFC 3515, and will provide a URI that will reach an event server that will service subscriptions to that state. (In many cases, the recipient of the REFER will perform the role of event server itself.) That URI is returned in a new header field in the REFER response named 'Refer-Events-At'.

The UA that issued the REFER can now subscribe to the 'refer' event at the provided URI, using a SUBSCRIBE request with a new dialog identifier. The full range of negotiation mechanisms is available for its use in that request. As detailed in RFC 6665 and RFC 3515, the event server accepting the subscription will send an immediate NOTIFY with the current refer event state, additional NOTIFY messages as the refer state changes, and a terminal NOTIFY message when the referred action is complete. It is, of course, possible that the initial NOTIFY is also the terminal NOTIFY.

It is possible that the referred action is completed before the SUBSCRIBE arrives at the event server. The server needs to retain the final refer event state for some period of time to include in the terminal NOTIFY that will be sent for such subscriptions. It is also possible that a SUBSCRIBE will never arrive.

This extension makes it possible to separate the event server that will handle subscriptions from the UA that accepted the REFER. Such a UA could use mechanisms such as PUBLISH [RFC3903] to convey the refer event state to the event server. This extension also makes it possible to allow more than one subscription to the refer event state.

3.2. No Subscriptions

A UA that wishes to issue a REFER request that will not create an implicit subscription, and tell the recipient that it is not interested in creating an explicit subscription, will include a new option tag, 'nosub', in the Require header field of the REFER request. This REFER could be sent either within an existing dialog or as an out-of-dialog request.

If the recipient of the REFER accepts the request, it knows not to create an implicit subscription, and that no explicit subscription will be forthcoming. The recipient will continue to process the request indicated in the Refer-To header field as specified in RFC

3515, but it can avoid the cost of preparing to handle any subscriptions to the state of handling that request.

4. The Explicit Subscription Extension

4.1. Sending a REFER

To suppress the creation of any implicit subscription, and allow for an explicit one, a UA forming a REFER request will include the option tag 'explicitsub' in the "Require" header field of the request. The REFER request is otherwise formed following the requirements of [RFC3515]. Since this REFER has no chance of creating an implicit subscription, the UA MAY send the REFER request within an existing dialog or out-of-dialog.

Note that if the REFER forks (see [RFC3261]), only one final response will be returned to the issuing UA. If it is important that the UA be able to subscribe to any refer state generated by accepting this request, the request needs to be formed to limit the number of places that it will be accepted to one. This can be achieved by sending the REFER request within an existing dialog, or by using the Target-Dialog mechanism defined in [RFC4538]. If it is possible for the request to be accepted in more than one location, and things would go wrong if the UA did not learn about each location that the request was accepted, using this extension is not appropriate.

4.2. Processing a REFER Response

The UA will process responses to the REFER request as specified in [RFC3515] (and, consequently, [RFC3261]). In particular, if the REFER was sent to an element that does not support or is unwilling to use this extension, the response will contain a 420 Bad Extension response code (see section 8.1.3.5 of [RFC3261]). As that document states, the UA can retry the request without using this extension.

If the UA receives a 2xx-class response, it will contain a Refer-Events-At header field (Section 4.8) with a single URI as its value. If the UA is interested in the state of the referenced action, it will subscribe to the 'refer' event at that URI.

4.3. Processing a Received REFER

An element receiving a REFER request requiring the 'explicitsub' extension will use the same admissions policies that would be used without the extension, with the addition that it is acceptable to admit an in-dialog REFER request requiring this extension since it can not create another usage inside that dialog. In particular, see section 5.2 of [RFC3515].

Accepting a REFER request that requires 'explicitsub' does not create a dialog, or a new usage within an existing dialog. The element MUST NOT create an implicit subscription when accepting the REFER request.

An element that accepts a REFER request with 'explicitsub' in its Require header field MUST return a 200 response containing a sip: or sips: URI that can be used to subscribe to the refer event state associated with this REFER request. This URI MUST uniquely identify this refer event state. The URI needs to reach the event server when used in a SUBSCRIBE request from the element that sent the REFER. One good way to ensure the URI provided has that property is to use a GRUU [RFC5627] for the event server. As discussed in Section 8, possession of this URI is often the only requirement for authorizing a subscription to it. Implementations may wish to provide a URI constructed in a way that is hard to guess. Again, using a GRUU is one good way to achieve this property.

The accepting element will otherwise proceed with the processing defined in [RFC3515].

The event server identified by the Refer-Events-At URI could receive SUBSCRIBE requests at any point after the response containing the Refer-Events-At header is sent. Implementations should take care to ensure the event server is ready to receive those SUBSCRIBE requests before sending the REFER response, but as with all non-INVITE responses, the response should be sent as soon as possible (see [RFC4321]). It is also possible that the referred action may complete before any SUBSCRIBE request arrives. The event server will need to maintain the final refer event state for a period of time after the action completes in order to serve such subscriptions (see Section 4.6).

4.4. Subscribing to the 'refer' Event

A UA that possesses a URI obtained from a Refer-Events-At header field, MAY subscribe to the refer event state at that URI. It does so following the requirements of [RFC6665], placing the token 'refer' in the Event: header field and the URI in the Request-URI of the SUBSCRIBE request. The SUBSCRIBE request MUST NOT reuse any existing dialog identifiers.

Subsequent handling of the subscription MUST follow the requirements of [RFC6665] and [RFC3515]. In particular, as discussed in section 2.4.6, the NOTIFY messages in the subscription might include an id parameter in their Event header fields. Subsequent SUBSCRIBE requests used to refresh or terminate this subscription MUST contain this id parameter. Note that the rationale for the id parameter provided in that section is not relevant when this extension is used.

The URI returned in the Refer-Events-At header field uniquely identifies appropriate state, making the id parameter redundant. However, this behavioral requirement is preserved to reduce the number of changes to existing implementations in order to support this extension, and to make it more likely that existing diagnostic tools will work with little or no modification.

4.5. Processing a Received SUBSCRIBE

An event server receiving a SUBSCRIBE request will process it according to the requirements of [RFC6665]. The event server MAY choose to authorize the SUBSCRIBE request based on the Request-URI corresponding to existing refer event state. It MAY also require further authorization as discussed in Section 8.

When accepting a subscription, the event server will establish the initial subscription duration using the guidance in section 3.4 of [RFC3515].

4.6. Sending a NOTIFY

NOTIFY messages within a subscription are formed and sent following the requirements in [RFC3515]. See, in particular, section 2.4.5 of that document.

4.7. Managing 'refer' Event State

As described in [RFC3515], an element creates the state for event 'refer' when it accepts a REFER request. It updates that state as the referred request proceeds, ultimately reaching a state where the request has completed, and the final state is known.

In RFC 3515 implementations, it was a reasonable design choice to destroy the refer event state immediately after sending the NOTIFY that terminated the implicit subscription. This is not the case when using this extension. It is possible for the referenced request to complete very quickly, perhaps sooner than the time it takes the response to the REFER to traverse the network to the UA that sent the request, and the time it takes that agent to send the SUBSCRIBE request for the event state to the URI the response provides. Thus the event server MUST retain the final refer event state for a reasonable period of time, which SHOULD be at least 2*64*T1 (that is, 64 seconds), representing an upper-bound estimate of the time it would take to complete two non-INVITE transactions: the REFER, and an immediate SUBSCRIBE.

If an otherwise acceptable SUBSCRIBE arrives during this retention period, the subscription would be accepted, and immediately

terminated with a NOTIFY containing the final event state with a Subscription-State of terminated with a reason value of "noresource".

4.8. The Refer-Events-At Header Field

The 'Refer-Events-At' header field is an extension-header as defined by [RFC3261]. Its ABNF is as follows:

```
Refer-Events-At: "Refer-Events-At" HCOLON
                 LAQUOT ( SIP-URI / SIPS-URI ) RAQUOT
                 * ( SEMI generic-param )
```

See [RFC3261] for the definition of the elements used in that production.

Note that this rule does not allow a full addr-spec as defined in RFC 3261, and it mandates the use of the angle brackets. That is:

```
Refer-Events-At: <sips:vPT3izGmo8NTxaPADRZvEAY22BKx@example.com;gr>
```

is well formed, but

```
Refer-Events-At: sip:wsXa9mkHtPcGu8@example.com
```

is invalid.

The 'Refer-Events-At' header field is only meaningful in a 200 response to a REFER request. If it appears in the header of any other SIP message, its meaning is undefined and it MUST be ignored.

5. The No Subscription Extension

5.1. Sending a REFER

To suppress the creation of any implicit subscription, and signal that no explicit subscription will be forthcoming, a UA forming a REFER request will include the option tag 'nosub' in the "Require" header field of the request. The REFER request is otherwise formed following the requirements of [RFC3515]. Since this REFER has no chance of creating an implicit subscription, the UA MAY send the REFER request within an existing dialog or out-of-dialog.

5.2. Processing a REFER Response

The UA will process responses to the REFER request as specified in [RFC3515] (and, consequently, [RFC3261]). In particular, if the REFER was sent to an element that does not support or is unwilling to use this extension, the response will contain a 420 Bad Extension

response code (see section 8.1.3.5 of [RFC3261]). As that document states, the UA can retry the request without using this extension.

5.3. Processing a Received REFER

An element receiving a REFER request requiring the 'nosub' extension will use the same admissions policies that would be used without the extension, with the addition that it is acceptable to admit an in-dialog REFER request requiring this extension since it can not create another usage inside that dialog. In particular, see section 5.2 of [RFC3515].

Accepting a REFER request that requires 'nosub' does not create a dialog, or a new usage within an existing dialog. The element MUST NOT create an implicit subscription when accepting the REFER request. Futhermore, the element accepting the REFER request is not required to maintain any state for serving refer event subscriptions.

The accepting element will otherwise proceed with the processing defined in [RFC3515].

6. The 'explicitsub' and 'nosub' Option Tags

This document defines the 'explicitsub' option tag, used to signal the use of the extension defined in Section 4, and the 'nosub' option tag, used to signal the use of the extension defined in Section 5.

The use of either option tag in a Require header field is only defined when it appears in a REFER request. A UA MUST NOT include the 'explicitsub' or 'nosub' option tag in the Require header field of any request other than REFER. A UA MUST NOT include the 'explicitsub' or 'nosub' option tag in the Require header field of any SIP response other than a 421 response to a REFER request.

The 'explicitsub' and 'nosub' option tags MAY appear in the Supported header field of SIP messages, and in sip.extensions feature tag defined in [RFC3840]. This signals only that the UA including the value is aware of the extensions. In particular, a UA can only invoke the use of one of the extensions in a request. A User-Agent Server (UAS) that is processing a REFER request that lists 'explicitsub' or 'nosub' in its Supported header field and wishes to use one of those extensions will return a 421 response indicating which extension is required.

OPEN ISSUE: This description of the use of 421 is not yet perfectly aligned with RFC3261's definition.

7. Updates to RFC 3515

The requirement in section 2.4.4 of [RFC3515] to reject out-of-dialog SUBSCRIBE requests to event 'refer' is removed. An element MAY accept a SUBSCRIBE request to event 'refer', following the requirements and guidance in this document. REFER is no longer the only mechanism that can create a subscription to event 'refer'.

[RFC6665] section 8.3.1 deprecates the 202 Accepted response code. New implementations of REFER, whether using the 'explicitsub' extension or not, will never emit a 202 response code. Where RFC 3515 specifies using 202, new implementations MUST use 200 instead.

8. Security Considerations

The considerations of [RFC3515] all still apply to a REFER request using this extension. The considerations there for the implicit subscription apply to any explicit subscription for the 'refer' event.

This update to RFC 3515 introduces a new authorization consideration. An element receiving an initial SUBSCRIBE request to the 'refer' event needs to decide whether the subscriber should be allowed to see the refer event state. In RFC 3515, this decision was conflated with accepting the REFER request, and the only possible subscriber was the element that sent the REFER. With this update, there may multiple subscribers to any given refer event state.

This document allows an element to accept an initial SUBSCRIBE request based on having a Request-URI that identifies existing refer event state. (Such a URI will have previously been sent in the Refer-Events-At header field in a successful REFER response). The element retrieving that URI from the response, and any elements that element shares the URI with are authorized to SUBSCRIBE to the event state. Consequently, the URI should be constructed so that it is not easy to guess, and should be protected against eavesdroppers when transmitted. For instance, SIP messages containing this URI SHOULD be sent using TLS or DTLS. An event server receiving a REFER request over an unprotected transport can redirect the requester to use a protected transport before accepting the request. A good way to ensure that subscriptions use a protected transport is to only construct sips: URIs. The event server can also require any of the additional authorization mechanisms allowed for any SIP request. For example, the event server could require a valid assertion of the subscriber's identity using [RFC4474].

The URI provided in a 'Refer-Events-At' header field will be used as the Request-URI of SUBSCRIBE requests. A malicious agent could take

advantage of being able to choose this URI in ways similar to the ways an agent sending a REFER request can take advantage of the Refer-To URI, as described in the security considerations section of RFC 3515. In particular, the malicious agent could cause a SIP SUBSCRIBE to be sent as raw traffic towards a victim. If the victim is not SIP aware, and the SUBSCRIBE is sent over UDP, there is (at most) a factor of 11 amplification due to retransmissions of the request. The potential for abuse in this situation is lower than that of the Refer-To URI, since the URI can only have a sip: or sips: scheme, and is only provided in a REFER response. A malicious agent would have to first receive a REFER request to take advantage of providing a Refer-Events-At URI.

9. IANA Considerations

9.1. Register the 'explicitsub' Option Tag

The option tag 'explicitsub' is registered in the 'Option Tag' subregistry of the 'Session Initiation Protocol (SIP) Parameters' registry by adding a row with these values:

Name: explicitsub

Description: This option tag identifies an extension to REFER to suppress the implicit subscription, and provide a URI for an explicit subscription.

Reference: (this document)

9.2. Register the 'nosub' Option Tag

The option tag 'nosub' is registered in the 'Option Tag' subregistry of the 'Session Initiation Protocol (SIP) Parameters' registry by adding a row with these values:

Name: nosub

Description: This option tag identifies an extension to REFER to suppress the implicit subscription, and indicate that no explicit subscription is forthcoming.

Reference: (this document)

9.3. Register the 'Refer-Events-At' Header Field

The header field described in Section 4.8 is registered in the 'Header Fields' subregistry of the 'Session Initiation Protocol (SIP) Parameters' registry by adding a row with these values:

Header Name: Refer-Events-At

compact: (none: the entry in this column should be blank)

Reference: (this document)

10. Changelog

RFC Editor - please remove this section when formatting this document as an RFC.

10.1. 01 to 02

1. Added the 'nosub' option tag

2. Added text calling out the limitations on explicitsub when the REFER might be accepted in more than one place.

10.2. 00 to 01

1. Replaced strawman proposal with a formal definition of the mechanism. Added an overview, and detailed security considerations.

11. References

11.1. 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.

[RFC3515] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, August 2004.

[RFC6665] Roach, A., "SIP-Specific Event Notification", RFC 6665, July 2012.

11.2. Informative References

[RFC3903] Niemi, A., "Session Initiation Protocol (SIP) Extension for Event State Publication", RFC 3903, October 2004.

[RFC4321] Sparks, R., "Problems Identified Associated with the Session Initiation Protocol's (SIP) Non-INVITE Transaction", RFC 4321, January 2006.

[RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.

[RFC4538] Rosenberg, J., "Request Authorization through Dialog Identification in the Session Initiation Protocol (SIP)", RFC 4538, June 2006.

[RFC5057] Sparks, R., "Multiple Dialog Usages in the Session Initiation Protocol", RFC 5057, November 2007.

[RFC5589] Sparks, R., Johnston, A., and D. Petrie, "Session Initiation Protocol (SIP) Call Control - Transfer", BCP 149, RFC 5589, June 2009.

[RFC5627] Rosenberg, J., "Obtaining and Using Globally Routable User Agent URIs (GRUUs) in the Session Initiation Protocol (SIP)", RFC 5627, October 2009.

Author's Address

Robert Sparks
Oracle
7460 Warren Parkway
Suite 300
Frisco, Texas 75034
US

Email: rjsparks@nostrum.com