                The Session Initiation Protocol (SIP) Refer Method

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.

Abstract

   This document defines the REFER method. This Session Initiation
   Protocol (SIP) extension requests that the recipient REFER to a
   resource provided in the request. It provides a mechanism allowing
   the party sending the REFER to be notified of the outcome of the
   referenced request. This can be used to enable many applications,
   including call transfer.

   In addition to the REFER method, this document defines the the refer
   event package and the Refer-To request header.

Table of Contents

1. Overview

   This document defines the REFER method. This SIP [1] extension
   requests that the recipient REFER to a resource provided in the
   request.

   This can be used to enable many applications, including Call
   Transfer. For instance, if Alice is in a call with Bob, and decides
   Bob needs to talk to Carol, Alice can instruct her SIP user agent
   (UA) to send a SIP REFER request to Bob's UA providing Carol's SIP
   Contact information. Assuming Bob has given it permission, Bob's UA
   will attempt to call Carol using that contact. Bob's UA will then
   report whether it succeeded in reaching the contact to Alice's UA.

2. The REFER Method

   REFER is a SIP method as defined by RFC 3261 [1]. The REFER method
   indicates that the recipient (identified by the Request-URI) should
   contact a third party using the contact information provided in the
   request.

   Unless stated otherwise, the protocol for emitting and responding to
   a REFER request are identical to those for a BYE request in [1]. The
   behavior of SIP entities not implementing the REFER (or any other
   unknown) method is explicitly defined in [1].

   A REFER request implicitly establishes a subscription to the refer
   event. Event subscriptions are defined in [2].

   A REFER request MAY be placed outside the scope of a dialog created
   with an INVITE. REFER creates a dialog, and MAY be Record-Routed,
   hence MUST contain a single Contact header field value. REFERs
   occurring inside an existing dialog MUST follow the Route/Record-
   Route logic of that dialog.

2.1 The Refer-To Header Field

   Refer-To is a request header field (request-header) as defined by
   [1]. It only appears in a REFER request. It provides a URL to
   reference.

      Refer-To = ("Refer-To" / "r") HCOLON (name-addr / addr-spec) *
      (SEMI generic-param)

   The following should be interpreted as if it appeared in Table 3 of
   RFC 3261.

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG
Refer-To	R		-	-	-	-	-	-

   The Refer-To header field MAY be encrypted as part of end-to-end
   encryption.

   The Contact header field is an important part of the Route/Record-
   Route mechanism and is not available to be used to indicate the
   target of the reference.

Examples

Refer-To: sip:alice@atlanta.example.com

Refer-To: <sip:bob@biloxi.example.net?Accept-Contact=sip:bobsdesk.
      biloxi.example.net&Call-ID%3D55432%40alicepc.atlanta.example.com>

Refer-To: <sip:dave@denver.example.org?Replaces=12345%40192.168.118.3%3B
         to-tag%3D12345%3Bfrom-tag%3D5FFE-3994>

Refer-To: <sip:carol@cleveland.example.org;method=SUBSCRIBE>

Refer-To: http://www.ietf.org

   Long headers field values are line-wrapped here for clarity only.

2.2 Header Field Support for the REFER Method

   This table adds a column to tables 2 and 3 in [1], describing header
   field presence in a REFER method. See [1] for a key for the symbols
   used. A row for the Refer-To request-header should be inferred,
   mandatory for REFER. Refer-To is not applicable for any other
   methods. The proxy column in [1] applies to the REFER method
   unmodified.

Header	Where	REFER
Accept	R	o
Accept	2xx	-
Accept	415	c
Accept-Encoding	R	o
Accept-Encoding	2xx	-
Accept-Encoding	415	c
Accept-Language	R	o
Accept-Language	2xx	-
Accept-Language	415	c
Alert-Info		-
Allow	Rr	o
Allow	405	m
Authentication-Info	2xx	o
Authorization	R	o
Call-ID	c	m
Call-Info		-
Contact	R	m
Contact	1xx	-
Contact	2xx	m
Contact	3-6xx	o
Content-Disposition		o
Content-Encoding		o

```
      Content-Language                     o
      Content-Length                       o
      Content-Type                         *
      CSeq                        c        m
      Date                                 o
      Error-Info               3-6xx       o
      Expires                     R        o
      From                        c        m
      In-Reply-To                          -
      Max-Forwards                R        m
      Min-Expires                          -
      MIME-Version                         o
      Organization                         o
      Priority                    R        -
      Proxy-Authenticate         401       o
      Proxy-Authenticate         407       m
      Proxy-Authorization         R        o
      Proxy-Require               R        o
      Record-Route                R        o
      Record-Route             2xx,18x     o
      Reply-To                             -
      Require                              c
      Retry-After         404,413,480,486  o
      Retry-After              500,503     o
      Retry-After              600,603     o
      Route                       R        c
      Server                      r        o
      Subject                     R        -
      Supported                 R,2xx      o
      Timestamp                            o
      To                        c(1)       m
      Unsupported                420       o
      User-Agent                           o
      Via                       c(2)       m
      Warning                     r        o
      WWW-Authenticate           401       m
      WWW-Authenticate           407       o
```

      Table 1: Header Field Support

2.3 Message Body Inclusion

   A REFER method MAY contain a body. This specification assigns no
   meaning to such a body. A receiving agent may choose to process the
   body according to its Content-Type.

2.4 Behavior of SIP User Agents

2.4.1 Forming a REFER request

   REFER is a SIP request and is constructed as defined in [1]. A REFER
   request MUST contain exactly one Refer-To header field value.

2.4.2 Processing a REFER request

   A UA accepting a well-formed REFER request SHOULD request approval
   from the user to proceed (this request could be satisfied with an
   interactive query or through accessing configured policy). If
   approval is granted, the UA MUST contact the resource identified by
   the URI in the Refer-To header field value as discussed in Section
   2.4.3.

   If the approval sought above for a well-formed REFER request is
   immediately denied, the UA MAY decline the request.

   An agent responding to a REFER method MUST return a 400 (Bad Request)
   if the request contained zero or more than one Refer-To header field
   values.

   An agent (including proxies generating local responses) MAY return a
   100 (Trying) or any appropriate 4xx-6xx class response as prescribed
   by [1].

   Care should be taken when implementing the logic that determines
   whether or not to accept the REFER request. A UA not capable of
   accessing non-SIP URIs SHOULD NOT accept REFER requests to them.

   If no final response has been generated according to the rules above,
   the UA MUST return a 202 Accepted response before the REFER
   transaction expires.

   If a REFER request is accepted (that is, a 2xx class response is
   returned), the recipient MUST create a subscription and send
   notifications of the status of the refer as described in Section
   2.4.4.

2.4.3 Accessing the Referred-to Resource

   The resource identified by the Refer-To URI is contacted using the
   normal mechanisms for that URI type. For example, if the URI is a
   SIP URI indicating INVITE (using a method=INVITE URI parameter for
   example), the UA would issue a new INVITE using all of the normal
   rules for sending an INVITE defined in [1].

2.4.4 Using SIP Events to Report the Results of the Reference

   The NOTIFY mechanism defined in [2] MUST be used to inform the agent
   sending the REFER of the status of the reference. The dialog
   identifiers (To, From, and Call-ID) of each NOTIFY must match those
   of the REFER as they would if the REFER had been a SUBSCRIBE request.

   Each NOTIFY MUST contain an Event header field with a value of refer
   and possibly an id parameter (see Section 2.4.6).

   Each NOTIFY MUST contain a body of type "message/sipfrag" [3].

   The creation of a subscription as defined by [2] always results in an
   immediate NOTIFY. Analogous to the case for SUBSCRIBE described in
   that document, the agent that issued the REFER MUST be prepared to
   receive a NOTIFY before the REFER transaction completes.

   The implicit subscription created by a REFER is the same as a
   subscription created with a SUBSCRIBE request. The agent issuing the
   REFER can terminate this subscription prematurely by unsubscribing
   using the mechanisms described in [2]. Terminating a subscription,
   either by explicitly unsubscribing or rejecting NOTIFY, is not an
   indication that the referenced request should be withdrawn or
   abandoned. In particular, an agent acting on a REFER request SHOULD
   NOT issue a CANCEL to any referenced SIP requests because the agent
   sending the REFER terminated its subscription to the refer event
   before the referenced request completes.

   The agent issuing the REFER may extend its subscription using the
   subscription refresh mechanisms described in [2].

   REFER is the only mechanism that can create a subscription to event
   refer. If a SUBSCRIBE request for event refer is received for a
   subscription that does not already exist, it MUST be rejected with a
   403.

   Notice that unlike SUBSCRIBE, the REFER transaction does not contain
   a duration for the subscription in either the request or the
   response. The lifetime of the state being subscribed to is
   determined by the progress of the referenced request. The duration
   of the subscription is chosen by the agent accepting the REFER and is
   communicated to the agent sending the REFER in the subscription's
   initial NOTIFY (using the Subscription-State expires header
   parameter). Note that agents accepting REFER and not wishing to hold
   subscription state can terminate the subscription with this initial
   NOTIFY.

2.4.5 The Body of the NOTIFY

   Each NOTIFY MUST contain a body of type "message/sipfrag" [3]. The
   body of a NOTIFY MUST begin with a SIP Response Status-Line as
   defined in [1]. The response class in this status line indicates the
   status of the referred action. The body MAY contain other SIP header
   fields to provide information about the outcome of the referenced
   action. This body provides a complete statement of the status of the
   referred action. The refer event package does not support state
   deltas.

   If a NOTIFY is generated when the subscription state is pending, its
   body should consist only of a status line containing a response code
   of 100.

   A minimal, but complete, implementation can respond with a single
   NOTIFY containing either the body:

      SIP/2.0 100 Trying

   if the subscription is pending, the body:

      SIP/2.0 200 OK

   if the reference was successful, the body:

      SIP/2.0 503 Service Unavailable

   if the reference failed, or the body:

      SIP/2.0 603 Declined

   if the REFER request was accepted before approval to follow the
   reference could be obtained and that approval was subsequently denied
   (see Section 2.4.7).

   An implementation MAY include more of a SIP message in that body to
   convey more information. Warning header field values received in
   responses to the referred action are good candidates. In fact, if
   the reference was to a SIP URI, the entire response to the referenced
   action could be returned (perhaps to assist with debugging).
   However, doing so could have grave security repercussions (see
   Section 5). Implementers must carefully consider what they choose to
   include.

   Note that if the reference was to a non-SIP URI, status in any
   NOTIFYs to the referrer must still be in the form of SIP Response
   Status-Lines. The minimal implementation discussed above is

sufficient to provide a basic indication of success or failure. For
example, if a client receives a REFER to a HTTP URL, and is
successful in accessing the resource, its NOTIFY to the referrer can
contain the message/sipfrag body of "SIP/2.0 200 OK". If the
notifier wishes to return additional non-SIP protocol specific
information about the status of the request, it may place it in the
body of the sipfrag message.

2.4.6 Multiple REFER Requests in a Dialog

A REFER creates an implicit subscription sharing the dialog
identifiers in the REFER request. If more than one REFER is issued
in the same dialog (a second attempt at transferring a call for
example), the dialog identifiers do not provide enough information to
associate the resulting NOTIFYs with the proper REFER.

Thus, for the second and subsequent REFER requests a UA receives in a
given dialog, it MUST include an id parameter[2] in the Event header
field of each NOTIFY containing the sequence number (the number from
the CSeq header field value) of the REFER this NOTIFY is associated
with. This id parameter MAY be included in NOTIFYs to the first
REFER a UA receives in a given dialog. A SUBSCRIBE sent to refresh
or terminate this subscription MUST contain this id parameter.

2.4.7 Using the Subscription-State Header Field with Event Refer

Each NOTIFY must contain a Subscription-State header field as defined
in [2]. The final NOTIFY sent in response to a REFER MUST indicate
the subscription has been "terminated" with a reason of "noresource".
(The resource being subscribed to is the state of the referenced
request).

If a NOTIFY indicates a reason that indicates a re-subscribe is
appropriate according to [2], the agent sending the REFER is NOT
obligated to re-subscribe.

In the case where a REFER was accepted with a 202, but approval to
follow the reference was subsequently denied, the reason and retry-
after elements of the Subscription-State header field can be used to
indicate if and when the REFER can be re-attempted (as described for
SUBSCRIBE in [2]).

2.5 Behavior of SIP Registrars/Redirect Servers

A registrar that is unaware of the definition of the REFER method
will return a 501 response as defined in [1]. A registrar aware of
the definition of REFER SHOULD return a 405 response.

This specification places no requirements on redirect server behavior
beyond those specified in [1]. Thus, it is possible for REFER
requests to be redirected.

2.6 Behavior of SIP Proxies

SIP proxies do not require modification to support the REFER method.
Specifically, as required by [1], a proxy should process a REFER
request the same way it processes an OPTIONS request.

3. Package Details: Event refer

This document defines an event package as defined in [2].

3.1 Event Package Name

The name of this event package is "refer".

3.2 Event Package Parameters

This package uses the "id" parameter defined in [2]. Its use in
package is described in Section 2.4.6.

3.3 SUBSCRIBE Bodies

SUBSCRIBE bodies have no special meaning for this event package.

3.4 Subscription Duration

The duration of an implicit subscription created by a REFER request
is initially determined by the agent accepting the REFER and
communicated to the subscribing agent in the Subscription-State
header field's expire parameter in the first NOTIFY sent in the
subscription. Reasonable choices for this initial duration depend on
the type of request indicated in the Refer-To URI. The duration
SHOULD be chosen to be longer than the time the referenced request
will be given to complete. For example, if the Refer-To URI is a SIP
INVITE URI, the subscription interval should be longer than the
Expire value in the INVITE. Additional time MAY be included to
account for time needed to authorize the subscription. The
subscribing agent MAY extend the subscription by refreshing it, or
terminate it by unsubscribing. As described in Section 2.4.7, the
agent accepting the REFER will terminate the subscription when it
reports the final result of the reference, indicating that
termination in the Subscription-State header field.

3.5 NOTIFY Bodies

   The bodies of NOTIFY requests for event refer are discussed in
   Section 2.4.5.

3.6 Notifier processing of SUBSCRIBE requests

   Notifier processing of SUBSCRIBE requests is discussed in Section
   2.4.4.

3.7 Notifier Generation of NOTIFY Requests

   Notifier generation of NOTIFY requests is discussed in Section 2.4.4.

3.8 Subscriber Processing of NOTIFY Requests

   Subscriber processing of NOTIFY requests is discussed in Section
   2.4.4.

3.9 Handling of Forked Requests

   A REFER sent within the scope of an existing dialog will not fork. A
   REFER sent outside the context of a dialog MAY fork, and if it is
   accepted by multiple agents, MAY create multiple subscriptions.
   These subscriptions are created and managed as per "Handling of
   Forked Requests" in [2] as if the REFER had been a SUBSCRIBE. The
   agent sending the REFER manages the state associated with each
   subscription separately. It does NOT merge the state from the
   separate subscriptions. The state is the status of the referenced
   request at each of the accepting agents.

3.10 Rate of Notifications

   An event refer NOTIFY might be generated each time new knowledge of
   the status of a referenced requests becomes available. For instance,
   if the REFER was to a SIP INVITE, NOTIFYs might be generated with
   each provisional response and the final response to the INVITE.
   Alternatively, the subscription might only result in two NOTIFY
   requests, the immediate NOTIFY and the NOTIFY carrying the final
   result of the reference. NOTIFYs to event refer SHOULD NOT be sent
   more frequently than once per second.

3.11 State Agents

   Separate state agents are not defined for event refer.

4. Examples

4.1 Prototypical REFER callflow

```
    Agent A                    Agent B
       |                          |
       |     F1 REFER             |
       |------------------------->|
       |          F2 202 Accepted |
       |<-------------------------|
       |          F3 NOTIFY       |
       |<-------------------------|
       | F4 200 OK                |
       |------------------------->|
       |                          |
       |                          |
       |                          |-------->
       |                          |  (whatever)
       |                          |<------
       |                          |
       |                          |
       |          F5 NOTIFY       |
       |<-------------------------|
       |    F6 200 OK             |
       |------------------------->|
       |                          |
       |                          |
```
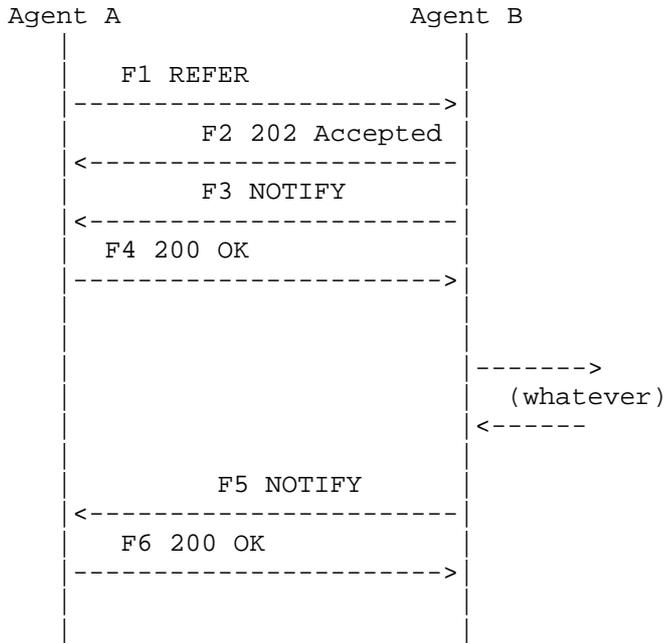
   Here are examples of what the four messages between Agent A and Agent
   B might look like if the reference to (whatever) that Agent B makes
   is successful. The details of this flow indicate this particular
   REFER occurs outside a session (there is no To tag in the REFER
   request). If the REFER occurs inside a session, there would be a
   non-empty To tag in the request.

Message One (F1)

REFER sip:b@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK2293940223
To: <sip:b@atlanta.example.com>
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809823 REFER
Max-Forwards: 70
Refer-To: (whatever URI)
Contact: sip:a@atlanta.example.com
Content-Length: 0

Message Two (F2)

SIP/2.0 202 Accepted
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK2293940223
To: <sip:b@atlanta.example.com>;tag=4992881234
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809823 REFER
Contact: sip:b@atlanta.example.com
Content-Length: 0

Message Three (F3)

NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9922ef992-25
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993402 NOTIFY
Max-Forwards: 70
Event: refer
Subscription-State: active;expires=(depends on Refer-To URI)
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 20

SIP/2.0 100 Trying

Message Four (F4)

SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9922ef992-25
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993402 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0

Message Five (F5)

NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9323394234
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993403 NOTIFY
Max-Forwards: 70

```
Event: refer
Subscription-State: terminated;reason=noresource
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 16

SIP/2.0 200 OK

Message Six (F6)

SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9323394234
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993403 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0
```

4.2 Multiple REFERs in a dialog

   Message One above brings an implicit subscription dialog into
   existence. Suppose Agent A issued a second REFER inside that dialog:

```
   Agent A                      Agent B
      |                            |
      |       F7 REFER             |
      |--------------------------->|
      |          F8 202 Accepted   |
      |<---------------------------|
      |          F9 NOTIFY         |
      |<---------------------------|
      |  F10 200 OK                |
      |--------------------------->|
      |                            |------->
      |                            |  (something different)
      |                            |<------
      |                            |
      |          F11 NOTIFY        |
      |<---------------------------|
      |  F12 200 OK                |
      |--------------------------->|
      |                            |
      |                            |
```
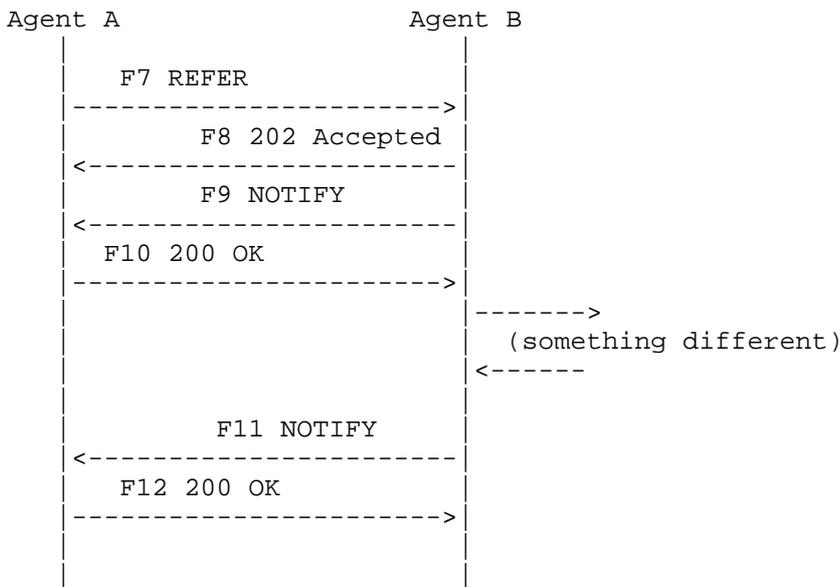
Message Seven (F7)

REFER sip:b@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK9390399231
To: <sip:b@atlanta.example.com>;tag=4992881234
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809824 REFER
Max-Forwards: 70
Refer-To: (some different URI)
Contact: sip:a@atlanta.example.com
Content-Length: 0

Message Eight (F8)

SIP/2.0 202 Accepted
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK9390399231
To: <sip:b@atlanta.example.com>;tag=4992881234
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809824 REFER
Contact: sip:b@atlanta.example.com
Content-Length: 0

Message Nine (F9)

NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9320394238995
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993404 NOTIFY
Max-Forwards: 70
Event: refer;id=93809824
Subscription-State: active;expires=(depends on Refer-To URI)
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 20

SIP/2.0 100 Trying

Message Ten (F10)

SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9320394238995
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com

```
CSeq: 1993404 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0
```

Message Eleven (F11)

```
NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK2994a93eb-fe
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993405 NOTIFY
Max-Forwards: 70
Event: refer;id=93809824
Subscription-State: terminated;reason=noresource
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 16
```

```
SIP/2.0 200 OK
```

Message Twelve (F12)

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK2994a93eb-fe
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993405 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0
```

5. Security Considerations

   The security considerations described in Section 26 of [1] apply to
   the REFER transaction. In particular, the implementation
   requirements and considerations in Section 26.3 address securing a
   generic SIP transaction. Special consideration is warranted for the
   authorization polices applied to REFER requests and for the use of
   message/sipfrag to convey the results of the referenced request.

5.1 Constructing a Refer-To URI

   This mechanism relies on providing contact information for the
   referred-to resource to the party being referred. Care should be
   taken to provide a suitably restricted URI if the referred-to
   resource should be protected.

5.2 Authorization Considerations for REFER

   As described in Section 2.4.2, an implementation can receive a REFER
   requests with a Refer-To URI containing an arbitrary scheme. For
   instance, a user could be referred to an online service such as a MUD
   using a telnet URI. Customer service could refer a customer to an
   order tracking web page using an HTTP URI. Section 2.4.2 allows a
   user agent to reject a REFER request when it can not process the
   referenced scheme. It also requires the user agent to obtain
   authorization from its user before attempting to use the URI.
   Generally, this could be achieved by prompting the user with the full
   URI and a question such as "Do you wish to access this resource
   (Y/N)". Of course, URIs can be arbitrarily long and are occasionally
   constructed with malicious intent, so care should be taken to avoid
   surprise even in the display of the URI itself (such as partial
   display or crashing). Further, care should be taken to expose as
   much information about the reference as possible to the user to
   mitigate the risk of being misled into a dangerous decision. For
   instance, the Refer-To header may contain a display name along with
   the URI. Nothing ensures that any property implied by that display
   name is shared by the URI. For instance, the display name may
   contain "secure" or "president" and when the URI indicates
   sip:agent59@telemarketing.example.com. Thus, prompting the user with
   the display name alone is insufficient.

   In some cases, the user can provide authorization for some REFER
   requests ahead of time by providing policy to the user agent. This
   is appropriate, for instance, for call transfer as discussed in [4].
   Here, a properly authenticated REFER request within an existing SIP
   dialog to a sip:, sips:, or tel: URI may be accepted through policy
   without interactively obtaining the user's authorization. Similarly,
   it may be appropriate to accept a properly authenticated REFER to an
   HTTP URI if the referror is on an explicit list of approved
   referrors. In the absence of such pre-provided authorization, the
   user must interactively provide authorization to reference the
   indicated resource.

   To see the danger of a policy that blindly accepts and acts on an
   HTTP URI, for example, consider a web server configured to accept
   requests only from clients behind a small organization's firewall.
   As it sits in this soft-creamy-middle environment where the small
   organization trusts all its members and has little internal security,
   the web server is frequently behind on maintenance, leaving it
   vulnerable to attack through maliciously constructed URIs (resulting
   perhaps in running arbitrary code provided in the URI). If a SIP UA
   inside this firewall blindly accepted a reference to an arbitrary
   HTTP URI, an attacker outside the firewall could compromise the web
   server. On the other hand, if the UA's user has to take positive

action (such as responding to a prompt) before acting on this URI,
the risk is reduced to the same level as the user clicking on the URI
in a web-browser or email message.

The conclusion in the above paragraph generalizes to URIs with an
arbitrary scheme. An agent that takes automated action to access a
URI with a given scheme risks being used to indirectly attack another
host that is vulnerable to some security flaw related to that scheme.
This risk and the potential for harm to that other host is heightened
when the host and agent reside behind a common policy-enforcement
point such as a firewall. Furthermore, this agent increases its
exposure to denial of service attacks through resource exhaustion,
especially if each automated action involves opening a new
connection.

User agents should take care when handing an arbitrary URI to a
third-party service such as that provided by some modern operating
systems, particularly if the user agent is not aware of the scheme
and the possible ramifications using the protocols it indicates. The
opportunity for violating the principal of least surprise is very
high.

5.3 Considerations for the use of message/sipfrag

Using message/sipfrag bodies to return the progress and results of a
REFER request is extremely powerful. Careless use of that capability
can compromise confidentiality and privacy. Here are a couple of
simple, somewhat contrived, examples to demonstrate the potential for
harm.

5.3.1 Circumventing Privacy

Suppose Alice has a user agent that accepts REFER requests to SIP
INVITE URIs, and NOTIFYs the referrer of the progress of the INVITE
by copying each response to the INVITE into the body of a NOTIFY.

Suppose further that Carol has a reason to avoid Mallory and has
configured her system at her proxy to only accept calls from a
certain set of people she trusts (including Alice), so that Mallory
doesn't learn when she's around, or what user agent she's actually
using.

Mallory can send a REFER to Alice, with a Refer-To URI indicating
Carol. If Alice can reach Carol, the 200 OK Carol sends gets
returned to Mallory in a NOTIFY, letting him know not only that Carol
is around, but also the IP address of the agent she's using.

5.3.2 Circumventing Confidentiality

   Suppose Alice, with the same user agent as above, is working at a
   company that is working on the greatest SIP device ever invented –
   the SIP FOO. The company has been working for months building the
   device and the marketing materials, carefully keeping the idea, even
   the name of the idea secret (since a FOO is one of those things that
   anybody could do if they'd just had the idea first). FOO is up and
   running, and anyone at the company can use it, but it's not available
   outside the company firewall.

   Mallory has heard rumor that Alice's company is onto something big,
   and has even managed to get his hands on a URI that he suspects might
   have something to do with it. He sends a REFER to ALICE with the
   mysterious URI and as Alice connects to the FOO, Mallory gets NOTIFYs
   with bodies containing

   Server: FOO/v0.9.7

5.3.3 Limiting the Breach

   For each of these cases, and in general, returning a carefully
   selected subset of the information available about the progress of
   the reference through the NOTIFYs mitigates risk. The minimal
   implementation described in Section 2.4.5 exposes the least
   information about what the agent operating on the REFER request has
   done, and is least likely to be a useful tool for malicious users.

5.3.4 Cut, Paste and Replay Considerations

   The mechanism defined in this specification is not directly
   susceptible to abuse through copying the message/sipfrag bodies from
   NOTIFY requests and inserting them, in whole or in part, in future
   NOTIFY requests associated with the same or a different REFER. Under
   this specification the agent replying to the REFER request is in
   complete control of the content of the bodies of the NOTIFY it sends.
   There is no mechanism defined here requiring this agent to faithfully
   forward any information from the referenced party. Thus, saving a
   body for later replay gives the agent no more ability to affect the
   mechanism defined in this document at its peer than it has without
   that body. Similarly, capture of a message/sipfrag body by
   eavesdroppers will give them no more ability to affect this mechanism
   than they would have without it.

   Future extensions may place additional constraints on the agent
   responding to REFER to allow using the message/sipfrag body part in a
   NOTIFY to make statements like "I contacted the party you referred me
   to, and here's cryptographic proof". These statements might be used

to affect the behavior of the receiving UA. This kind of extension
will need to define additional mechanism to protect itself from copy
based attacks.

6. Historic Material

This method was initially motivated by the call-transfer application.
Starting as TRANSFER, and later generalizing to REFER, this method
improved on the BYE/Also concept of the expired draft-ietf-sip-cc-01
by disassociating transfers from the processing of BYE. These
changes facilitate recovery of failed transfers and clarify state
management in the participating entities.

Early versions of this work required the agent responding to REFER to
wait until the referred action completed before sending a final
response to the REFER. That final response reflected the success or
failure of the referred action. This was infeasible due to the
transaction timeout rules defined for non-INVITE requests in [1]. A
REFER must always receive an immediate (within the lifetime of a
non-INVITE transaction) final response.

7. IANA Considerations

This document defines a new SIP method name (REFER), a new SIP header
field name with a compact form (Refer-To and r respectively), and an
event package (refer).

The following has been added to the method sub-registry under
http://www.iana.org/assignments/sip-parameters.

    REFER   [RFC3515]

The following information also has been be added to the header sub-
registry under http://www.iana.org/assignments/sip-parameters.

    Header Name: Refer-To

    Compact Form: r

    Reference: RFC 3515

This specification registers an event package, based on the
registration procedures defined in [2]. The following is the
information required for such a registration:

    Package Name: refer

    Package or Package-Template: This is a package.

   Published Specification: RFC 3515

   Person to Contact: Robert Sparks, rsparks@dynamicsoft.com

8. Acknowledgments

   This document is a collaborative product of the SIP working group.

9.   References

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

   [3] Sparks, R., "Internet Media Type message/sipfrag", RFC 3420,
        November 2002.

9.2 Informative References

   [4] Sparks, R. and A. Johnston, "Session Initiation Protocol Call
        Control - Transfer", Work in Progress.

10. Intellectual Property Statement

   The IETF takes no position regarding the validity or scope of any
   intellectual property or other rights that might be claimed to
   pertain to the implementation or use of the technology described in
   this document or the extent to which any license under such rights
   might or might not be available; neither does it represent that it
   has made any effort to identify any such rights. Information on the
   IETF's procedures with respect to rights in standards-track and
   standards-related documentation can be found in BCP-11. Copies of
   claims of rights made available for publication and any assurances of
   licenses to be made available, or the result of an attempt made to
   obtain a general license or permission for the use of such
   proprietary rights by implementors or users of this specification can
   be obtained from the IETF Secretariat.

   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.

11. Author's Address

     Robert J. Sparks
     dynamicsoft
     5100 Tennyson Parkway
     Suite 1200
     Plano, TX 75024

     EMail: rsparks@dynamicsoft.com

12. Full Copyright Statement

Acknowledgement