Network Working Group
Request for Comments: 3891
Category: Standards Track

R. Mahy
Cisco Systems, Inc.
B. Biggs
R. Dean

September 2004

The Session Initiation Protocol (SIP) "Replaces" Header

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.

Copyright Notice

Abstract

This document defines a new header for use with Session Initiation Protocol (SIP) multi-party applications and call control. The Replaces header is used to logically replace an existing SIP dialog with a new SIP dialog. This primitive can be used to enable a variety of features, for example: "Attended Transfer" and "Call Pickup". Note that the definition of these example features is non-normative.

Table of Contents

1. Overview

This document describes a SIP [1] extension header field as part of the SIP multiparty applications architecture framework [10]. The Replaces header is used to logically replace an existing SIP dialog with a new SIP dialog. This is especially useful in peer-to-peer call control environments.

One use of the "Replaces" header is to replace one participant with another in a multimedia conversation. While this functionality is already available using 3rd party call control [11] style call control, the 3pcc model requires a central point of control which may not be desirable in many environments. As such, a method of performing these same call control primitives in a distributed, peer-to-peer fashion is very desirable.

Use of a new INVITE with a new header for dialog matching was chosen over making implicit associations in an incoming INVITE based on call-id or other fields for the following reasons:

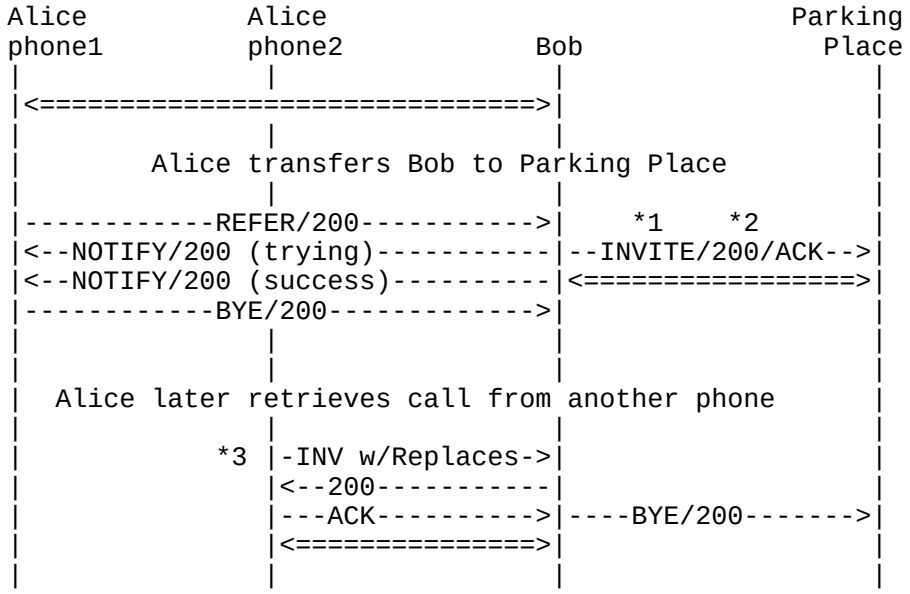
- An INVITE already has the correct semantics for a new call
- Using an explicit Replaces header in a new request makes the intent of the request obvious.

- A unique call-id may be given to the replacement call. This avoids dialog matching problems in any of the related User Agents.

- There are no adverse effects if the header is unsupported.

The Replaces header enables services such as attended call transfer, retrieve from park, and transition from locally mixed conferences to two party calls in a distributed peer-to-peer way. This list of services is not exhaustive. Although the Replaces header is frequently used in combination with the REFER [8] method as used in a Transfer [12], they may be used independently.

For example, Alice is talking to Bob from phone1. She transfers Bob to a Parking Place while she goes to the lab. When she gets there she retrieves the "parked" call from phone2 by sending an INVITE with a Replaces header field to Bob with the dialog information Bob shared with the Parking Place. Alice got this information using some out of band mechanism. Perhaps she subscribed to this information from the Parking Place (using the session dialog package [13]), or went to a website and clicked on a URI. A short call flow for this example follows. (Via and Max-Forwards headers are omitted for clarity.)



```
Message *1: Bob-> Parking Place

INVITE sip:parkingplace@example.org SIP/2.0
To: <sip:parkingplace@example.org>
From: <sip:bob@example.org>;tag=7743
Call-ID: 425928@bobster.example.org
CSeq: 1 INVITE
Contact: <sip:bob@bobster.example.org>
Referred-By: <sip:alice@phone1.example.org>

Message *2: Parking Place -> Bob

SIP/2.0 200 OK
To: <sip:parkingplace@example.org>;tag=6472
From: <sip:bob@example.org>;tag=7743
Call-ID: 425928@bobster.example.org
CSeq: 1 INVITE
Contact: <sip:parkplace@monopoly.example.org>

Message *3: Alice@phone2 -> Bob

INVITE sip:bob@bobster.example.org
To: <sip:bob@example.org>
From: <sip:alice@phone2.example.org>;tag=8983
Call-ID: 09870@phone2.example.org
CSeq: 1 INVITE
Contact: <sip:alice@phone2.example.org>
Require: replaces
Replaces: 425928@bobster.example.org;to-tag=7743;from-tag=6472
```

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [2].

This document refers frequently to the terms "confirmed dialog" and "early dialog". These are defined in Section 12 of SIP [1].

3. User Agent Server Behavior: Receiving a Replaces Header

The Replaces header contains information used to match an existing SIP dialog (call-id, to-tag, and from-tag). Upon receiving an INVITE with a Replaces header, the User Agent (UA) attempts to match this information with a confirmed or early dialog. The User Agent Server (UAS) matches the to-tag and from-tag parameters as if they were tags

present in an incoming request. In other words, the to-tag parameter is compared to the local tag, and the from-tag parameter is compared to the remote tag.

If more than one Replaces header field is present in an INVITE, or if a Replaces header field is present in a request other than INVITE, the UAS MUST reject the request with a 400 Bad Request response.

The Replaces header has specific call control semantics. If both a Replaces header field and another header field with contradictory semantics are present in a request, the request MUST be rejected with a 400 "Bad Request" response.

If the Replaces header field matches more than one dialog, the UA MUST act as if no match is found.

If no match is found, the UAS rejects the INVITE and returns a 481 Call/Transaction Does Not Exist response. Likewise, if the Replaces header field matches a dialog which was not created with an INVITE, the UAS MUST reject the request with a 481 response.

If the Replaces header field matches a dialog which has already terminated, the UA SHOULD decline the request with a 603 Declined response. (If the matched invitation was just terminated, the replacement request should fail as well. Declining the request with a 600-class response prevents an irritating race-condition where the UA rings or alerts for a replacement call which is not wanted.)

If the Replaces header field matches an active dialog, the UA MUST verify that the initiator of the new INVITE is authorized to replace the matched dialog. If the initiator of the new INVITE has been successfully authenticated as equivalent to the user who is being replaced, then the replacement is authorized. For example, if the user being replaced and the initiator of the replacement dialog share the same credentials for Digest authentication [6], or they sign the replacement request with S/MIME [7] with the same private key and present the (same) corresponding certificate used in the original dialog, then the replacement is authorized.

Alternatively, the Referred-By mechanism [4] defines a mechanism that the UAS can use to verify that a replacement request was sent on behalf of the other participant in the matched dialog (in this case, triggered by a REFER request). If the replacement request contains a Referred-By header that corresponds to the user being replaced, the UA SHOULD treat the replacement as if the replacement was authorized by the replaced party. The Referred-By header SHOULD reference a corresponding, valid Refererred-By Authenticated Identity Body [5].

The UA MAY apply other local policy to authorize the remainder of the request. In other words, the UAS may apply a different policy to the replacement dialog than was applied to the replaced dialog.

In addition, the UA MAY use other authorization mechanisms defined for this purpose in standards track extensions. Extensions could define other mechanisms for transitively asserting authorization of a replacement.

If authorization is successful, the UA attempts to accept the new INVITE, reassign the user interface and other resources of the matched dialog to the new INVITE, and shut down the replaced dialog. If the UA cannot accept the new INVITE (for example: it cannot establish required QoS or keying, or it has incompatible media), the UA MUST return an appropriate error response and MUST leave the matched dialog unchanged.

If the Replaces header field matches a confirmed dialog, it checks for the presence of the "early-only" flag in the Replaces header field. (This flag allows the UAC to prevent a potentially undesirable race condition described in Section 7.1.) If the flag is present, the UA rejects the request with a 486 Busy response. Otherwise, it accepts the new INVITE by sending a 200-class response, and shuts down the replaced dialog by sending a BYE. If the Replaces header field matches an early dialog that was initiated by the UA, it accepts the new INVITE by sending a 200-class response, and shuts down the replaced dialog by sending a CANCEL.

If the Replaces header field matches an early dialog that was not initiated by this UA, it returns a 481 (Call/Transaction Does Not Exist) response to the new INVITE, and leaves the matched dialog unchanged. Note that since Replaces matches only a single dialog, the replacement dialog will not be retargeted according to the same forking logic as the original request which created the early dialog.

(Currently, no use cases have been identified for replacing just a single dialog in this circumstance.)

4. User Agent Client Behavior: Sending a Replaces Header

A User Agent that wishes to replace a single existing early or confirmed dialog with a new dialog of its own, MAY send the target User Agent an INVITE request containing a Replaces header field. The User Agent Client (UAC) places the Call-ID, to-tag, and from-tag information for the target dialog in a single Replaces header field and sends the new INVITE to the target. If the user agent only wishes to replace an early dialog (as in the Call Pickup example in Section 7.1), the UAC MAY also include the "early-only" parameter in

the Replaces header field. A UAC MUST NOT send an INVITE with a Replaces header field that attempts to replace an early dialog which was not originated by the target of the INVITE with a Replaces header field.

Note that use of this mechanism does not provide a way to match multiple dialogs, nor does it provide a way to match an entire call, an entire transaction, or to follow a chain of proxy forking logic. For example, if Alice replaces Cathy in an early dialog with Bob, but Bob does not answer, Alice's replacement request will not match other dialogs to which Bob's UA redirects, nor other branches to which his proxy forwards. Although this specification takes reasonable precautions to prevent unexpected behavior in the face of forking, implementations SHOULD only address replacement requests (i.e., set the Request-URI of the replacement request) to the SIP Contact URI of the target.

5. Proxy behavior

Proxy Servers do not require any new behavior to support this extension. They simply pass the Replaces header field transparently as described in the SIP specification.

Note that it is possible for a proxy (especially when forking based on some application layer logic, such as caller screening or time-of-day routing) to forward an INVITE request containing a Replaces header field to a completely orthogonal set of Contacts other than the original request it was intended to replace. In this case, the INVITE request with the Replaces header field will fail.

6. Syntax

6.1. The Replaces Header

The Replaces header field indicates that a single dialog identified by the header field is to be shut down and logically replaced by the incoming INVITE in which it is contained. It is a request header only, and defined only for INVITE requests. The Replaces header field MAY be encrypted as part of end-to-end encryption. Only a single Replaces header field value may be present in a SIP request.

This document adds the following entry to Table 2 of [1]. Additions to this table are also provided for extension methods defined at the time of publication of this document. This is provided as a courtesy to the reader and is not normative in any way. MESSAGE, SUBSCRIBE and NOTIFY, REFER, INFO, UPDATE, PRACK, and PUBLISH are defined respectively in [15], [16], [8], [17], [18], [19], and [20].

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG	MSG
Replaces	R		-	-	-	o	-	-	-

Header field	where	SUB	NOT	REF	INF	UPD	PRA	PUB
Replaces	R	-	-	-	-	-	-	-

The following syntax specification uses the augmented Backus-Naur Form (BNF) as described in RFC 2234 [3]. The syntax below relies on a number of productions from SIP [1].

```
Replaces        = "Replaces" HCOLON callid *(SEMI replaces-param)
replaces-param  = to-tag / from-tag / early-flag / generic-param
to-tag          = "to-tag" EQUAL token
from-tag        = "from-tag" EQUAL token
early-flag      = "early-only"
```

A Replaces header field MUST contain exactly one to-tag and exactly one from-tag, as they are required for unique dialog matching. For compatibility with dialogs initiated by RFC 2543 [9] compliant UAs, a tag of zero matches both tags of zero and null. A Replaces header field MAY contain the early-flag.

Examples:

```
Replaces: 98732@sip.example.com
          ;from-tag=r33th4x0r
          ;to-tag=ff87ff

Replaces: 12adf2f34456gs5;to-tag=12345;from-tag=54321;early-only

Replaces: 87134@171.161.34.23;to-tag=24796;from-tag=0
```

6.2. New Option Tag for Require and Supported Headers

This specification defines a new Require/Supported header option tag "replaces". UAs which support the Replaces header MUST include the "replaces" option tag in a Supported header field. UAs that want explicit failure notification if Replaces is not supported MAY include the "replaces" option in a Require header field.

Example:

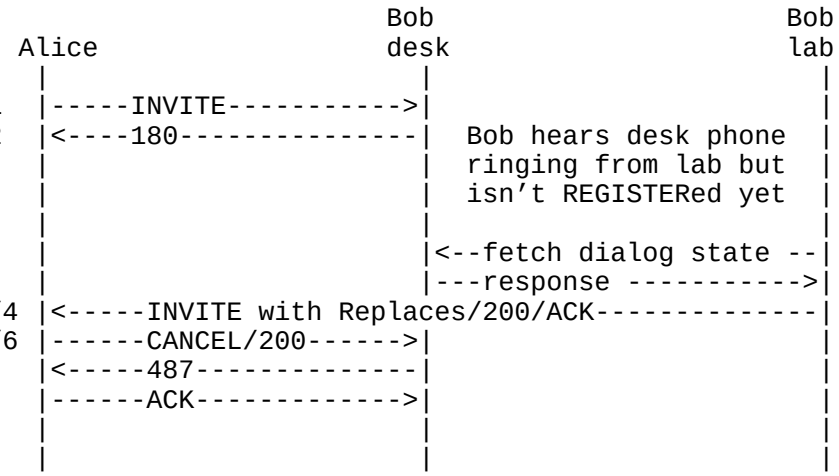
```
Require: replaces, 100rel
```

7. Usage Examples

The following non-normative examples are not intended to enumerate all the possibilities for the usage of this extension, but rather to provide examples or ideas only. For more examples, please see SIP Service Examples [14]. Via and Max-Forwards headers are omitted for clarity and brevity.

7.1. Replacing an Early Dialog at the Originator

In this example, Bob just arrived in the lab and hasn't registered there yet. He hears his desk phone ring. He quickly logs into a software UA on a nearby computer. Among other things, the software UA has access to the dialog state of his desk phone. When it notices that his phone is ringing, it offers him the choice of taking the call there. The software UA sends an INVITE with Replaces to Alice. When Alice's UA receives this new INVITE, it CANCELs her original INVITE and connects Alice to Bob.



Message *1: Alice -> Bob's desk phone

```
INVITE sip:bob@example.org SIP/2.0
To: <sip:bob@example.org>
From: <sip:alice@example.org>;tag=7743
Call-ID: 425928@phone.example.org
CSeq: 1 INVITE
Contact: <sip:alice@phone.example.org>
```

```
   Message *2: Bob's desk phone -> Alice

   SIP/2.0 180 Ringing
   To: <sip:bob@example.org>;tag=6472
   From: <sip:alice@example.org>;tag=7743
   Call-ID: 425928@phone.example.org
   CSeq: 1 INVITE
   Contact: <sip:bob@bobster.example.org>

   Message *3: Bob in lab -> Alice

   INVITE sip:alice@phone.example.org
   To: <sip:alice@example.org>
   From: <sip:bob@example.org>;tag=8983
   Call-ID: 09870@labpc.example.org
   CSeq: 1 INVITE
   Contact: <sip:bob@labpc.example.org>
   Replaces: 425928@phone.example.org
    ;to-tag=7743;from-tag=6472;early-only

   Message *4: Alice -> Bob in lab

   SIP/2.0 200 OK
   To: <sip:alice@example.org>;tag=9232
   From: <sip:bob@example.org>;tag=8983
   Call-ID: 09870@labpc.example.org
   CSeq: 1 INVITE
   Contact: <sip:alice@phone.example.org>

   Message *5: Alice -> Bob's desk

   CANCEL sip:bob@example.org SIP/2.0
   To: <sip:bob@example.org>
   From: <sip:alice@example.org>;tag=7743
   Call-ID: 425928@phone.example.org
   CSeq: 1 CANCEL
   Contact: <sip:alice@phone.example.org>

   Message *6: Bob's desk -> Alice

   SIP/2.0 200 OK
   To: <sip:bob@example.org>
   From: <sip:alice@example.org>;tag=7743
   Call-ID: 425928@phone.example.org
   CSeq: 1 CANCEL
   Contact: <sip:bob@bobster.example.org>
```

Message *7: Bob’s desk -> Alice

```
SIP/2.0 487 Request Terminated
To: <sip:bob@example.org>;tag=6472
From: <sip:alice@example.org>;tag=7743
Call-ID: 425928@phone.example.org
CSeq: 1 INVITE
```

8. Security Considerations

The extension specified in this document significantly changes the relative security of SIP devices. Currently in SIP, even if an eavesdropper learns the Call-ID, To, and From headers of a dialog, they cannot easily modify or destroy that dialog if Digest authentication or end-to-end message integrity are used.

This extension can be used to disconnect participants or replace participants in a multimedia conversation. As such, invitations with the Replaces header MUST only be accepted if the peer requesting replacement has been properly authenticated using a standard SIP mechanism (Digest or S/MIME), and authorized to request a replacement of the target dialog. All SIP implementations are already required to support Digest Authentication. In addition, implementations which support the Replaces header SHOULD also implement the Referred-By mechanism.

How a User Agent determines which requests are legitimately authorized to make dialog replacements is non-trivial and depends on a considerable amount of local policy configuration. In general, there are four cases when an authorization for a replacement is reasonable or warranted.

1. Replacement made by a party considered equivalent to the replaced party

2. Replacement made on behalf of the replaced party (perhaps transitively)

3. Replacement made by a former participant

4. Replacement made by a specifically authorized party

Starting with #1 for example, if an executive and an assistant both receive requests for a shared address-of-record, if so configured, either should be able to replace dialogs of the other for the shared identity. Both could even share the same keying material (Digest or S/MIME), or one could hold an authorization document signed by the

other expressing this relationship. Likewise, in a call center environment, each call center agent could possess credentials to which supervisors also have access.

The most common use case of a replacement is on the request of the replaced participant (who no longer wants to be involved). This is the case in many features, such as completing an Attended Transfer and converting a 3-way call to a point-to-point call. Such replacements are typically triggered by a REFER [8] request from the replaced participant. The Referred-By [4] mechanism defines one way to identify the apparent original requester and can point to a SIP Authenticated Identity Body [5] (an S/MIME-based signed assertion) to secure this information.

In the example in section 1, Alice sends an INVITE with Replaces to Bob. Alice was a former participant in the conversation and had a previous dialog relationship with Bob. Alice can use the same Digest or S/MIME credentials she used to authenticate with Bob during the original call to prove that she was a former participant. Note that this justification for replacing calls is more dangerous than the others, and in most cases is another way to authorize that the replacing participant is available. Implementations SHOULD NOT rely on this method as an authorization mechanism.

The last scenario is the easiest to secure but the least likely to be useful in practice. It is unlikely that an arbitrary host in the Internet is aware of any special authorization relationship between the replaced and the replacing parties. However, this use case may be useful in some environments. Since this usage does not effectively degrade the security of the solution, it is still allowed.

Some mechanisms for obtaining the dialog information needed by the Replaces header (Call-ID, to-tag, and from-tag) include URIs on a web page, subscriptions to an appropriate event package, and notifications after a REFER request. Since manipulating this dialog information could cause User Agents to replace the wrong dialog, use of message integrity protection for this information is STRONGLY RECOMMENDED. Use of end-to-end security mechanisms to encrypt this information is also RECOMMENDED.

This extension was designed to take advantage of future signature or authorization schemes defined in standards track extensions. In general, call control features benefit considerably from such work.

9. IANA Considerations

9.1. Registration of "Replaces" SIP header

Name of Header:	Replaces
Short form:	none
Normative description:	section 6.1 of this document

9.2. Registration of "replaces" SIP Option-tag

Name of option:	replaces
Description:	Support for the SIP Replaces header
SIP headers defined:	Replaces
Normative description:	This document

10. Acknowledgments

Thanks to Robert Sparks, Alan Johnston, Dan Petrie, Ben Campbell, and many other members of the SIP WG for their continued support of the cause of distributed call control in SIP.

11. References

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

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

[4] Sparks, R., "The Session Initiation Protocol (SIP) Referred-By Mechanism", RFC 3892, September 2004.

[5] Peterson, J., "The Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, September 2004.

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

[7] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

11.2. Informative References

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

[9] Handley, M., Schulzrinne, H., Schooler, E., and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, March 1999.

[10] Mahy, R., "A Call Control and Multi-party usage framework for the Session Initiation Protocol (SIP)", Work in Progress, March 2003.

[11] Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", BCP 85, RFC 3725, April 2004.

[12] Sparks, R. and A. Johnston, "Session Initiation Protocol Call Control - Transfer", Work in Progress, February 2003.

[13] Rosenberg, J. and H. Schulzrinne, "An INVITE Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", Work in Progress, March 2003.

[14] Johnston, A. and S. Donovan, "Session Initiation Protocol Service Examples", Work in Progress, March 2003.

[15] Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C., and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, December 2002.

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

[17] Donovan, S., "The SIP INFO Method", RFC 2976, October 2000.

[18] Rosenberg, J., "The Session Initiation Protocol (SIP) UPDATE Method", RFC 3311, October 2002.

[19] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, June 2002.

[20] Campbell, B., "SIMPLE Presence Publication Mechanism", Work in Progress, February 2003.

12. Authors' Addresses

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

EMail: rohan@cisco.com

Billy Biggs

EMail: bbiggs@dumbterm.net

Rick Dean

EMail: rfc@fdd.com

13. Full Copyright Statement

Intellectual Property

Acknowledgement

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