A Session Identifier for the Session Initiation Protocol (SIP)

Abstract

   There is a need for having a globally unique session identifier for
   the same SIP session that can be consistently maintained across SIP
   Proxies, Back-to-Back User Agents (B2BUAs), and other SIP
   middleboxes, for the purpose of troubleshooting.  This document
   proposes a new SIP header to carry such a value: Session-ID.

   The mechanism defined in this document has been widely deployed, and
   is being followed in a backward-compatible fashion for a new
   Standards Track document produced by the INSIPID Working Group.

Status of This Memo

   This document is not an Internet Standards Track specification; it is
   published for informational purposes.

   This document is a product of the Internet Engineering Task Force
   (IETF).  It represents the consensus of the IETF community.  It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG).  Not all documents
   approved by the IESG are a candidate for any level of Internet
   Standard; see Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc7329.

Table of Contents

1.  Introduction

   This RFC, which contains the text of an individual Internet-Draft
   that was submitted originally to the DISPATCH Working Group, is being
   published now as an Informational document to provide a reference for
   later RFCs.  The mechanism defined in this document has been widely
   deployed and is being followed in a backward-compatible fashion for a
   new Standards Track document produced by the INSIPID Working Group.

   The SIP [RFC3261] Call-ID header value is a globally unique
   identifier, which is mandatory in all requests/responses and
   identifies SIP messages belonging to the same dialog or registration.
   It provides a portion of the SIP message dialog-matching criteria and
   is used in such things as "Replaces" headers [RFC3891] and dialog-
   event packages [RFC4235] for matching to dialogs, and in SIP Identity
   [RFC4474] and Connected Identity [RFC4916] as one of the inputs for
   signing.

   In practice, the Call-ID is often changed by SIP Back-to-Back User
   Agents (B2BUAs) and other such middleboxes in the logical end-to-end
   message path.  A B2BUA logically represents a SIP User Agent Server
   (UAS) and User Agent Client (UAC), and as such generates a new
   Call-ID value for the dialog it creates on its UAC side; in fact, for
   some B2BUA scenarios the Call-ID *must* be changed for SIP to
   function properly.

   At the same time, there is a need for a unique, common, consistent
   end-to-end identifier to help troubleshoot SIP sessions and message
   flows as they cross SIP nodes.  Troubleshooting is more complicated
   if multiple legs of the session are on different sides of B2BUAs, due
   to the lack of a common identifier such as a Call-ID to tie the legs
   together.  Proprietary mechanisms are currently used to achieve this
   goal.

   Therefore, in order to provide an identifier that will not be
   modified/replaced by B2BUAs, this document proposes a new SIP Header
   "Session-ID" and mandatory rules for the value of such a header.  The
   rules are designed to be such that the value in the Session-ID header
   is not considered unsafe or private and does not have any property
   that would cause B2BUAs to change it.  The goal of this document is
   to enable troubleshooting by providing a unique identifier for a
   given session that can successfully cross B2BUAs, such as Application
   Servers, softswitches, Private Branch Exchanges (PBXs), Session
   Border Controllers (SBCs), feature servers, etc.

1.1.  Requirements

   The following requirements drive the need for Session-ID:

   REQ1: It must be possible for an administrator to use the identifier
         to identify a set of dialogs that have a direct correlation
         with each other such that they represent the same SIP session,
         with as high a probability as possible.

   REQ2: It must be possible to pass the identifier through SIP B2BUAs,
         with as high a probability as possible.  This requirement
         drives the following requirements:

         REQ2a: The identifier must not reveal any information related
                to any SIP device or domain identity, including IP
                address, port, hostname, domain name, username, Address-
                of-Record (AoR), MAC address, IP address family,
                transport type, etc.

         REQ2b: The identifier must not reveal to the receiver of it
                that the Call-ID, tags, or any other SIP header or body
                portion have been changed by middleboxes, with as high a
                probability as possible.

         REQ2c: The identifier must not be used for anything at a SIP
                layer to change the behavior of the SIP protocol.

2.  Terminology

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in [RFC2119].

   This document uses the terms "header field" and "header field value"
   following the definition of those terms in [RFC3261]; they are not
   interchangeable.  The "header field" is the entire SIP header's
   contents, including any parameters.  The "header field value" is only
   the field-value portion, which does not include header parameters.

3.  Overview of Operation

   The general concept is that the UAC generating an out-of-dialog
   request generates a new, pseudorandom, unique value that remains
   constant for the duration of the transaction, any dialog created from
   that request, or for a registration.  The value is inserted in a new
   Session-ID header field defined in this document.  The UAC and UAS
   then reflect this header field value in all messages for the duration
   of the dialog.  In other words, the Session-ID provides a value

   similar in nature to Call-ID, except one that crosses B2BUAs and that
   has no sensitive information in it.

   To aid in migration of deployments, a B2BUA or Proxy MAY also
   generate and/or insert the value on behalf of a UAC or UAS, if one or
   the other does not support this document's mechanism.

   Although the Session-ID concept is similar to that of Call-ID, it is
   not used for message dialog-matching rules in RFC 3261, nor does it
   change the Call-ID usage, nor does it replace the Call-ID value.
   Instead, this new header field provides an identifier for
   troubleshooting uses only.

   The format of the Session-ID value is restricted, both to avoid
   detection of the system type that generated it and to keep it a
   hexadecimal representation such that it can be stored as a 128-bit
   binary value in log records.

4.  Session-ID Behavior

4.1.  Generating a Session-ID Value

   This document proposes the Session-ID header value be generated based
   on a defined hash mechanism for creating a 128-bit pseudorandom value
   and be encoded as its lowercase hex representation.  The reason for
   specifying the mechanism is twofold: to make it impossible to
   determine the manufacturer of the device that generated it by looking
   at its format or value, and to allow devices to generate the same
   value if they have the same private key.

   The Session-ID value is generated by taking the Call-ID header value
   and SHA-1 hashing it based on HMAC (as defined in [RFC2104]) using a
   locally generated pseudorandom 128-bit system secret key to create a
   128-bit resultant HMAC value.  The secret key makes the resultant
   HMAC value not re-creatable by other parties; this is necessary to
   prevent detection of Call-IDs being changed, as required by REQ2b.
   Otherwise, middleboxes may have motivation to remove the Session-ID
   in order to hide the fact that they changed the Call-ID.

   Per [RFC2104], the algorithm is thus HMAC-SHA-1-128(Call-ID_value,
   secret_key), and the 128-bit result is encoded using lowercase
   alphanumeric hex representation, as defined in Section 7.1
   ("Augmented BNF Definitions").

   In order to enable troubleshooting of in-dialog messages, a generator
   needs to remember or re-create the same Session-ID value for the
   duration of a given dialog(s).  This is described in more detail in
   the subsequent sections of this document.

4.2.  UAC Behavior

   The rules for when a UAC generates a new Session-ID value are similar
   as those for Call-ID value: a UAC supporting this document's
   mechanism MUST generate a new unique Session-ID value when it
   generates an out-of-dialog request or when there is a new
   Registration.  The exception to this rule is for out-of-dialog REFER
   requests or for an INVITE with a Replaces header field (see
   [RFC3891]), as described in Section 5.

   The UAC MUST reuse the same Session-ID value for in-dialog messages
   as that of the original dialog-creating request, and for any out-of-
   dialog request that it retransmits or re-generates in response to a
   3xx that it reformulates due to failure responses.  This follows the
   rules in [RFC3261] for Call-ID generation.

   Session-ID values in Registration "refreshes" -- REGISTER requests
   that are used to update the expiry time but not to register a new
   contact -- MUST use the same Session-ID value as previous REGISTER
   requests.  New Registrations, which add or change the Contact URI for
   the AoR, but do not simply delete them, MUST use a new Session-ID
   value.  This follows the behavior of Call-ID per RFC 3261; it is
   reiterated here because some devices incorrectly change their Call-ID
   value for every re-Registration, and they MUST NOT do the same to the
   Session-ID.

   The UAC MUST include the Session-ID header field in every SIP message
   it transmits.

4.3.  UAS Behavior

   A UAS compliant with this document MUST copy a Session-ID header
   field (received in a request) into responses and subsequent upstream
   requests sent within the dialog.

   If an out-of-dialog request is received without a Session-ID header
   field, the UAS SHOULD generate a new one for subsequent use in the
   transaction and dialog, as defined for a UAC, and use the same value
   in all responses and upstream in-dialog requests for the same dialog.

4.4.  Proxy Behavior

   A Proxy MUST NOT remove or modify the Session-ID header field it
   receives, if one is in the message.  By definition, a Proxy that is
   compliant with RFC 3261 would not modify or remove such a header.

   If the Proxy forks a request, it MUST copy the same Session-ID header
   field into all the forked request copies.  If the Proxy recurses
   requests due to 3xx redirection, or regenerates requests due to
   failures, it MUST use the same Session-ID header field as the
   original request, just as the UAC does.

   If the Proxy locally generates any response or request based on a
   received request, including 100 Trying, it MUST insert any received
   Session-ID header field from the original request into the response
   message it locally creates.  This is necessary for troubleshooting
   purposes.

   A Proxy compliant with this document MAY generate a new Session-ID or
   insert a previously saved one if and only if none existed in a
   received message, following the rules for doing so as a B2BUA as
   defined in Section 4.5.

4.5.  B2BUA Behavior

   A B2BUA compliant with this document MUST copy:

   -  the Session-ID header field it receives in requests as a UAS into
      the related requests it generates as a UAC, and

   -  any Session-ID header field it receives in responses as a UAC into
      the correlated responses it generates as a UAS.

   If the B2BUA forks or creates multiple requests as a UAC, from a
   request it received as a UAS, the B2BUA MUST copy the same Session-ID
   header field it received into all the forks/requests.  If the B2BUA
   recurses on 3xx responses, or regenerates requests due to failures,
   it MUST use the same Session-ID field, just as the UAC does.

   If the B2BUA locally generates any response or request based on a
   received request, including 100 Trying, it MUST insert any received
   Session-ID field from the original request into the response message
   it locally creates.

   A B2BUA MAY remember the received Session-ID value for the duration
   of the transaction and dialog, for the purpose of reinsertion, in
   case the far end does not support this document.

   In all cases, if the SIP message received by a B2BUA contains a
   Session-ID header field, a B2BUA compliant with this document MUST
   NOT remove, modify, or replace it as it "forwards" the message on the
   other logical UA "side" of itself.

4.5.1.  B2BUA Generation of New Session-ID

   If an out-of-dialog request is received by a B2BUA compliant with
   this document, and the request does *not* contain a Session-ID header
   field, the B2BUA MAY generate a new one.  The new Session-ID value
   MUST be calculated based on the received Call-ID of the received
   request, even if the B2BUA uses a different Call-ID value for
   requests generated on its other "side(s)".  It MUST then insert the
   new Session-ID in any requests or responses it generates, as if it
   had actually received the new Session-ID from the UAC, following the
   rules previously defined for a B2BUA.  This allows for a B2BUA to
   provide a migration to Session-ID deployment, on behalf of upstream
   nodes that do not yet support it.

   As defined previously, if any received message already had a
   Session-ID, a B2BUA compliant with this document would not replace
   it.

4.5.2.  B2BUA Insertion of Saved Session-ID

   If a Session-ID was received in an out-of-dialog request, or the
   B2BUA locally generated one because none existed, the B2BUA SHOULD
   insert the same Session-ID field into all responses and upstream
   in-dialog requests if and only if a Session-ID is not already in
   them.  This allows for a B2BUA to provide a migration to Session-ID
   deployment on behalf of downstream nodes that do not yet support it.

5.  Handling SIP Transfer Scenarios

   The transfer or movement of SIP sessions represents a complication
   for a mechanism like Session-ID.  On the one hand, the replacement
   SIP session represents a new one and could reasonably be expected to
   use a new Session-ID value; on the other hand, from a troubleshooting
   and human-user perspective, it is clearly related to, if not just a
   continuation of, the previous session.  Since the purpose of this
   document's mechanism is to aid monitoring and troubleshooting, and
   it's not used for actual SIP protocol mechanics, the behavior defined
   in this section is to reuse the same Session-ID value for the
   replacement SIP session.

   In order to do so, the Session-ID of the "original" session is
   transferred as well, in the Refer-To URI of a REFER request as
   described in [RFC3515].  Furthermore, out-of-dialog REFER and INVITE
   with Replaces requests as described in [RFC3891] use the appropriate
   Session-ID values.  This assumes, of course, that the UAs involved
   support the Session-ID mechanism.  If they do not, then it is
   possible for the Session-ID to not be "carried forward" to the new

   SIP dialog.  Unfortunately, this means troubleshooting such dialogs
   is not improved or aided by this document's mechanism; but, it would
   not "break" anything at a SIP layer.

   It should also be noted that using the same Session-ID for the
   transferred-to dialog means the same Session-ID now exists in two
   independent dialogs, because the original one may well continue due
   to the implicit Subscription usage created by a REFER.  That implicit
   Subscription-based usage will continue to use the same Session-ID as
   the new dialog created to the transferred-to party.

   In the following subsections, the term "UA" is used for User Agent.
   The language applies to the SIP device that creates the request,
   whether it be a UA or B2BUA.

## 5.1.  Out-of-Dialog REFER

   A UA compliant with this document MUST use the same Session-ID header
   field value for an out-of-dialog REFER request it generates, as the
   original dialog the REFER is targeted to (i.e., as if the REFER had
   been in-dialog).  For example, if UA Bob has a SIP dialog X to Alice,
   and Bob sends an out-of-dialog REFER to Alice to refer her to
   Charlie, the Session-ID header field value of the REFER request would
   be the same as that used in dialog X.

## 5.2.  Refer-To URI

   A UA compliant with this document MUST add the Session-ID header
   field as an embedded header in the Refer-To header field URI of any
   REFER request it generates, using the value of the session it is
   referring to.  For example, if UA Bob has a SIP dialog X to Alice and
   dialog Y to Charlie, and Bob sends a REFER request to Alice to refer
   her to Charlie, the Session-ID header field value embedded in the
   Refer-To URI of the REFER request would be the same as that used in
   dialog Y.

## 5.3.  Out-of-Dialog INVITE with Replaces

   When generating an out-of-dialog INVITE with a Replaces header field
   as described in [RFC3891], a UA compliant with this document MUST use
   the same Session-ID header field value for the INVITE request as that
   used for the dialog it is replacing, if it knows the value.
   Typically, the UA would know the value by having received it in the
   Refer-To header field of a REFER, as described previously.  For
   example, if UA Bob has a SIP dialog X to Alice and dialog Y to
   Charlie, and Bob sends a REFER request to Alice to refer her to
   Charlie, the Session-ID header field value embedded in the Refer-To

URI of the REFER request would be the one used in dialog Y, which
Alice would use as the Session-ID header field value for her INVITE
to Charlie.

If the UA does not know the Session-ID of the dialog it is replacing,
for example, because it is not embedded in the Refer-To URI of a
received REFER, then it MUST use a new Session-ID value, calculated
using the mechanism as defined in Section 4.1 with the Call-ID of the
INVITE.

6.  Session-ID Migration and Failure Scenarios

SIP is already widely deployed on the Internet, and it is impractical
to expect all UAs to be upgraded to support this document's mechanism
in the near future.  A solution for gradual migration is necessary
and is provided by this document by allowing B2BUAs or Proxies to
perform the Session-ID generator and inserter role.  Even within
those device types, it is impractical to expect all B2BUAs to support
this mechanism all at once or any time in the near future.
Therefore, it is expected that some B2BUAs and/or UAs will support
generating and inserting Session-ID, while others will not support
Session-ID at all.

Due to the varying types of B2BUAs (such as PBXs, SBCs, Application
Servers, feature servers, and softswitches of various flavors) and
the numerous SIP deployment models in use, there are going to be
cases in which Session-ID will fail to be a consistent value for all
related dialogs or fail to successfully match.  The goal of this
document is to improve troubleshooting of current deployments as much
as possible -- and, in this author's opinion, that is the best that
can be done given the constraints.

One example is for forked requests: if a UAC that does not support
this mechanism sends a request to a Proxy or B2BUA that also does not
support this mechanism, each fork could reach B2BUAs or UASs that
*do* support this mechanism.  In such a case, each of those forked-to
B2BUA/UAS will generate unique Session-IDs and put them in their
responses, temporarily leading to multiple, different Session-ID
values for the same related early dialogs.  Typically, the UAC would
eventually only accept one of the dialogs, and only one Session-ID
would remain.

7.  New 'Session-ID' Header

   This document adds the "Session-ID" token to the definition of the
   element "message-header" in the SIP message grammar.  The Session-ID
   header is a single-instance header.

7.1.  Augmented BNF Definitions

    Session-ID            =   "Session-ID" HCOLON sess-id
                                *( SEMI generic-param )
    sess-id               =   32(DIGIT / %x61-66)  ; 32 chars of [0-9a-f]

   NOTE: The sess-id value is technically case-INSENSITIVE, but only
   lowercase characters are allowed.

   See the Security Considerations section for discussion about using
   header parameters in Session-ID header fields.

8.  Example Exchange

   In the following example, Alice initiates a call to Bob.  Alice
   generates a Session-ID header in the out-of-dialog INVITE.

   Alice generates the following.  (Note: much has been left out for
   simplicity.)

      INVITE sip:bob@example.com SIP/2.0
      Via: SIP/2.0/UDP 192.0.2.1:5060;branch=z9hG4bKnashds10
      From: Alice <sip:alice@example.net>;tag=1234567
      To: Bob <sip:bob@example.com>
      Call-Id: 123456mcmxcix@1.2.3.4
      Session-ID: f81d4fae7dec11d0a76500a0c91e6bf6
      CSeq: 1 INVITE
      Contact: <sip:alice@192.168.1.1>

9.  Security Considerations

   There are several security considerations surrounding this document's
   mechanism.

   The Session-ID generation algorithm should provide a reasonably
   random 32-character Session-ID value to avoid collisions and should
   not let one re-create the original Call-ID.

   There is no known security issue with viewing or modifying the
   Session-ID, other than to hamper troubleshooting efforts.

9.1.  Security Considerations for Administrators

   The requirement for the Session-ID is to be an identifier which
   cannot be used by a recipient to identify if the Call-ID has been
   changed by middleboxes.  As such, a UAS/UAC cannot detect the
   original Call-ID, nor whether it has been changed; thus,
   administrators should not be concerned if the Session-ID header field
   is "passed through".

9.2.  Security Considerations for Session-ID Extensions

   The Session-ID's value is created from the Call-ID using a hashing
   mechanism based on [RFC2104], using SHA-1 and a secret key known only
   to the system generating the Session-ID.  Because the algorithm is
   defined in this document, it should be fairly secure from detecting
   the generator of the Session-ID, in terms of manufacturer or code
   base.

   The Session-ID generation algorithm should provide a reasonably
   random 128-bit Session-ID value, to avoid collisions, and should not
   let one re-create the original Call-ID.  The secret key MUST only be
   used for the Session-ID mechanism, in case a weakness is found that
   reveals the key.  One such weakness may be that a UAC generates one
   or more Call-IDs that have a property that makes determining the key
   more likely.

   In general, B2BUA behavior cannot be dictated by standards.  They do
   whatever their owners/operators wish them to do, or whatever is
   necessary to make their applications work.  This document attempts to
   normatively specify some B2BUA behavior, by creating a SIP header
   value for which the properties are such that B2BUAs should have no
   legitimate reason to interfere.  This effectively creates a "promise"
   that future uses of this Session-ID header field, including its value
   *and* any future defined parameters, maintain this benign property.
   Any future extensions to the Session-ID mechanism and header field
   MUST maintain this property, or else B2BUAs will begin to modify it
   again or remove it, and its value will be lost.

   Manufacturers of SIP devices should note that a B2BUA may inspect the
   Session-ID header field and may remove it if it does not comply with
   this document's value restrictions.  Any Session-ID header parameters
   MUST be registered with IANA and documented in RFCs from the IETF
   stream, pursuant to the requirements of [RFC3968].

10.  IANA Considerations

   IANA has registered a new SIP header field named 'Session-ID',
   pursuant to the registration policies for such in [RFC5727].  This is
   a single-instance header field and is appropriate for any SIP
   message, of any Method type, in any request or response.

   The ABNF rules [RFC5234] for this new header allow for header
   parameters; however, they must be registered following the rules of
   [RFC3968], as required by [RFC5727].

   This registration is intended to be temporary.  The author expects
   that a Standards Track definition of Session-ID will be published at
   a future date.  Assuming such a document is published, it will
   replace this registration with a reference to itself, at which point
   this document will no longer be referenced by IANA.

11.  Acknowledgments

   Thanks to Raphael Coeffic, Bob Penfield, Dale Worley, Paul Kyzivat,
   Ian Elz, Marco Stura, Martin Dolly, Martin Huelsemann, Laura Liess,
   and Adam Roach for their input.

12.  References

12.1.  Normative References

   [RFC2104]  Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-
              Hashing for Message Authentication", RFC 2104, February
              1997.

   [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.

   [RFC3891]  Mahy, R., Biggs, B., and R. Dean, "The Session Initiation
              Protocol (SIP) "Replaces" Header", RFC 3891, September
              2004.

   [RFC3968]  Camarillo, G., "The Internet Assigned Number Authority
              (IANA) Header Field Parameter Registry for the Session
              Initiation Protocol (SIP)", BCP 98, RFC 3968, December
              2004.

   [RFC5234]  Crocker, D., Ed., and P. Overell, "Augmented BNF for
              Syntax Specifications: ABNF", STD 68, RFC 5234, January
              2008.

   [RFC5727]  Peterson, J., Jennings, C., and R. Sparks, "Change Process
              for the Session Initiation Protocol (SIP) and the Real-
              time Applications and Infrastructure Area", BCP 67, RFC
              5727, March 2010.

12.2.  Informative References

   [RFC4235]  Rosenberg, J., Schulzrinne, H., and R. Mahy, Ed., "An
              INVITE-Initiated Dialog Event Package for the Session
              Initiation Protocol (SIP)", RFC 4235, November 2005.

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

   [RFC4916]  Elwell, J., "Connected Identity in the Session Initiation
              Protocol (SIP)", RFC 4916, June 2007.

Appendix A.  Use Cases Not in Scope for Session-ID

   It is very tempting to use a header field value such as that provided
   by Session-ID, for other purposes than troubleshooting.  In a
   previous document for this same Session-ID concept, the proposal
   included other uses; however, these were removed because any use case
   other than troubleshooting can easily lead to a B2BUA needing to
   change the value, in certain cases.  That would defeat the
   troubleshooting value of Session-ID.  This section discusses such use
   cases and explains why they are potentially harmful.

A.1.  Dialog Correlation for SIP

   Although Session-ID does provide a means to correlate separate SIP
   dialogs, messages, and transactions, it does so at a higher layer
   than SIP.  It does not replace the mechanics of SIP using the Call-ID
   and To/From tags of SIP messages to correlate SIP dialogs, nor in
   other uses such as Replaces headers or dialog-event packages.  It is
   tempting, however, to use it for exactly that purpose in certain
   cases.

   For example, suppose a call transfer case where Alice calls Bob
   through B2BUA-1.  Bob then calls Charlie and sends Charlie a REFER
   with embedded Replaces to make Charlie send an INVITE with a Replaces
   header to Alice, to replace the Alice-Bob session.  If Charlie uses a
   different B2BUA-2 to reach Alice, the INVITE with Replaces will fail
   because the Call-ID/tags won't match anything B2BUA-2 or Alice knows
   about.

```
  +-----+     +-------+     +-------+     +-----+     +-------+
  |Alice|     |B2BUA-1|     |B2BUA-2|     | Bob |     |Charlie|
  +-----+     +-------+     +-------+     +-----+     +-------+
     |            |             |            |            |
     |INVITE      |             |            |            |
     |callid:1a   |callid:1b    |            |            |
     |----------->|---------------------->|INVITE        |
     |sessid:1    |sessid:1     |            |callid:2a   |
     |            |             |            |----------->|
     |            |             |            |sessid:2    |
     |            |             |            |            |
     |            |             |            |REFER       |
     |            |             |            |referto:1b  |
     |            |             |            |----------->|
     |            |             |            |            |
     |            |             |            |      INVITE|
     |            |             |            | replaces:1b|
     |            |             |         X<---------------------|
     |            |      INVITE|             |      sessid:1|
     |            | replaces:1b|             |            |
     |        X<---------------------|         |            |
     |            |      sessid:1|             |            |
```
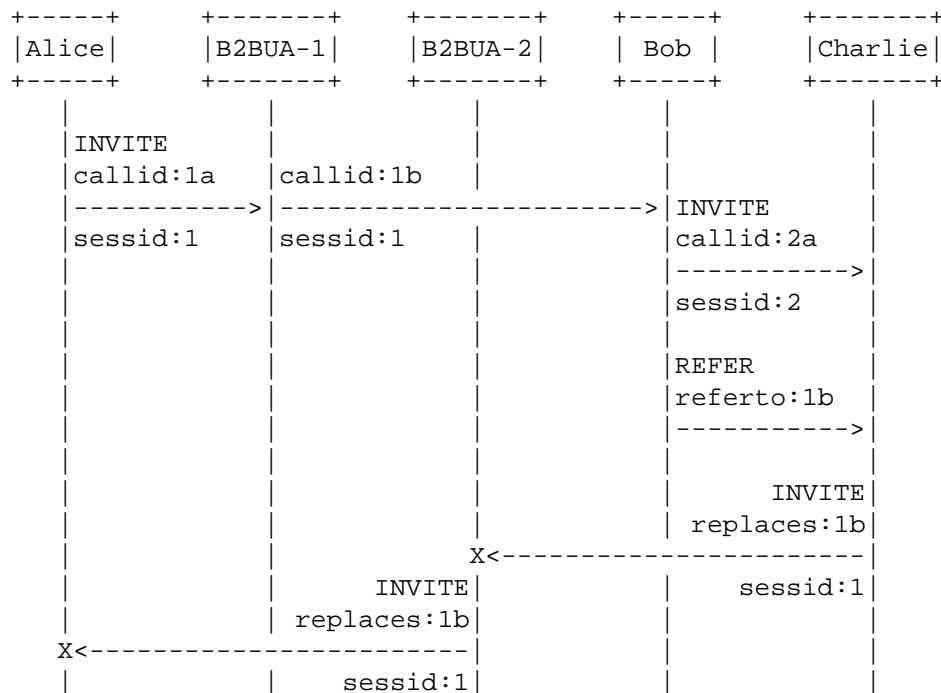
                   Example 1: Call Transfer Case

   If, on the other hand, Alice were to use the Session-ID value for
   correlation, she would see it matches her dialog with Bob (assuming
   the Session-ID were passed along in the Refer-To and Replaces info).
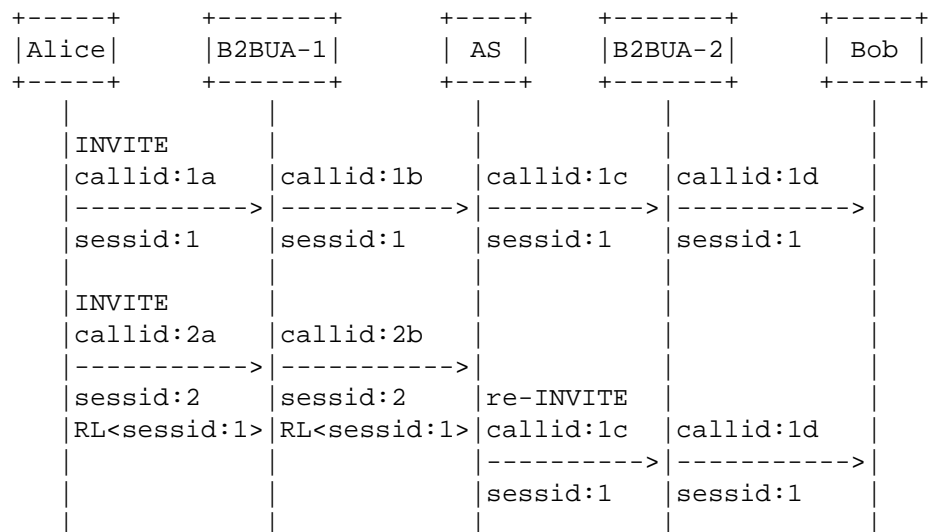
   There are problems with this approach, however.  The first problem
   is, by not sending the INVITE with Replaces to B2BUA-1, B2BUA-1 is in
   an incorrect state; the dialog is getting replaced, and the B2BUA
   doesn't know it.

   A second issue is the Session-ID doesn't identify enough information
   to replace a dialog.  Imagine there were a third B2BUA, such as a
   softswitch, between Alice and B2BUA-1 and B2BUA-2, and the INVITE
   with Replaces reached the softswitch before Alice.  The softswitch
   won't know which "side" the INVITE is replacing.  The To/From tags no
   longer match anything the softswitch knows about, so it can't figure
   out if the INVITE with Replaces is replacing the dialog from
   softswitch to Alice, or the one to Bob.  If we try to fix this by
   creating a tag-type value pair for Session-ID, we're back to devices
   changing those tag values and defeating the matching property.

   Another example is based on 3GPP 24.605 Annex A.2.2.  Alice has a
   call with Bob through multiple B2BUAs and an Application Server.  The
   dialogs of that call all have the same Session-ID, but unique
   Call-ID/tags.

   Alice wants to invoke a third-party conference facility in the AS and
   to reference the call she has with Bob for that.  In this particular
   3GPP scenario, to do that Alice sends a new INVITE to the AS with a
   resource-list body (a la RFC 5366) containing the call information
   for the original call.  This is the "RL<sessid:1>" piece in the
   diagram.  It has the Call-ID/tags as well, but they'll be wrong when
   received at the AS.

   The AS processes that list, can't match the Call-ID/tags in the
   resource-lit but does match the Session-ID, and sends a re-INVITE to
   party B within the original call's dialog.

```
    +-----+     +-------+      +----+     +-------+     +-----+
    |Alice|     |B2BUA-1|      | AS |     |B2BUA-2|     | Bob |
    +-----+     +-------+      +----+     +-------+     +-----+
      |             |            |            |            |
      |INVITE       |            |            |            |
      |callid:1a    |callid:1b   |callid:1c   |callid:1d   |
      |----------->|----------->|---------->|----------->|
      |sessid:1     |sessid:1    |sessid:1    |sessid:1    |
      |             |            |            |            |
      |INVITE       |            |            |            |
      |callid:2a    |callid:2b   |            |            |
      |----------->|----------->|            |            |
      |sessid:2     |sessid:2    |re-INVITE   |            |
      |RL<sessid:1>|RL<sessid:1>|callid:1c   |callid:1d   |
      |             |            |---------->|----------->|
      |             |            |sessid:1    |sessid:1    |
      |             |            |            |            |
```

                     Example 2: Resource List

Author's Address

   Hadriel Kaplan
   Oracle
   EMail: hadrielk@yahoo.com