

Network Working Group
Internet Draft
Intended status: Informational
Expires: April 9, 2013

Paul E. Jones
Gonzalo Salgueiro
James Polk
Cisco Systems
Laura Liess
Deutsche Telekom
Hadriel Kaplan
Acme Packet
October 9, 2012

Requirements for an End-to-End Session Identification in
IP-Based Multimedia Communication Networks
draft-ietf-insipid-session-id-reqts-02.txt

Abstract

This document specifies the requirements for an end-to-end session identifier in IP-based multimedia communication networks. This identifier would enable endpoints, intermediate devices, and management and monitoring systems to identify a session end-to-end across multiple SIP devices, hops, and administrative domains.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 9, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

| | |
|--|---|
| 1. Introduction..... | 2 |
| 2. Terminology..... | 3 |
| 3. Session Identifier Use Cases..... | 3 |
| 3.1. End-to-end identification of a communication session..... | 3 |
| 3.2. DELETED..... | 4 |
| 3.3. Protocol Interworking..... | 4 |
| 3.4. Traffic Monitoring..... | 4 |
| 3.5. DELETED..... | 5 |
| 3.6. Tracking transferred sessions..... | 5 |
| 3.7. Session Signal Logging..... | 5 |
| 3.8. Identifier Porting to Other Protocols - RTCP..... | 5 |
| 3.9. 3PCC Use Case..... | 5 |
| 4. Requirements for the End-to-End Session Identifier..... | 6 |
| 5. Related Work in other Standards Organizations..... | 7 |
| 5.1. Coordination with the ITU-T..... | 7 |
| 5.2. Requirements within 3GPP..... | 7 |
| 6. Security Considerations..... | 8 |
| 7. IANA Considerations..... | 8 |
| 8. Acknowledgments..... | 8 |
| 9. References..... | 8 |
| 9.1. Normative References..... | 8 |
| 9.2. Informative References..... | 8 |
| Author's Addresses..... | 9 |

1. Introduction

IP-based multimedia communication systems like SIP [1] and H.323 [2] have the concept of a "call identifier" that is globally unique. The identifier is intended to represent an end-to-end communication session from the originating device to the terminating device. Such an identifier is useful for troubleshooting, billing, session tracking, and so forth.

Unfortunately, there are a number of factors that contribute to the fact that the current call identifiers defined in SIP and H.323 are not suitable for end-to-end session identification. Perhaps most significant is the fact that the syntax for the call identifier in SIP and H.323 is different between the two protocols. This important fact makes it impossible for call identifiers to be exchanged end-to-end when a network utilizes one or more session protocols.

Another reason why the current call identifiers are not suitable to identify the session end-to-end is that in real-world deployments devices like Back-to-Back User Agents often change the values as the session signaling passes through. This is true even when a single session protocol is employed and not a byproduct of protocol interworking.

Lastly, identifiers that might have been used to identify a session end-to-end fail to meet that need when sessions are manipulated through supplementary service interactions. For example, when a session is transferred or if a PBX joins or merges two communication sessions together locally, the end-to-end properties of currently-defined identifiers are lost.

2. Terminology

SIP defines additional terms used in this document that are specific to the SIP domain such as "proxy"; "registrar"; "redirect server"; "user agent server" or "UAS"; "user agent client" or "UAC"; "user agent" (UA); "back-to-back user agent" or "B2BUA"; "dialog"; "transaction"; "server transaction".

In this document, the word "session" refers to a "communication session" that may exist between two SIP user agents or that might pass through one or more intermediary devices, including B2BUAs or SIP Proxies.

The term "end-to-end" in this document means the communication session from the point of origin, passing through any number of intermediaries, to the ultimate point of termination. It is recognized that legacy devices may not support the "end-to-end" session identifier, though an identifier might be created by an intermediary when it is absent from the session signaling.

3. Session Identifier Use Cases

3.1. End-to-end identification of a communication session

SIP messaging that either does not involve SIP servers or only involves SIP proxies, the Call-ID: header value sufficiently identifies each SIP message within a transaction or dialog. This is not the case when either B2BUAs or SBCs are in the signaling path between UAs. Therefore, we need the ability to identify each communication session per transaction through a single SIP header-value regardless of which type of SIP servers are in the signaling path between UAs. For transactions that create a dialog, have each message within the same dialog MUST use the same identifier.

Derived Requirements: All Requirements in Section 4

3.2. DELETED

3.3. Protocol Interworking

A communication session might originate in an H.323 endpoint and pass through a Session Border Controller before ultimately reaching a terminating SIP user agent. Likewise, a call might originate on a SIP user agent and terminate on an H.323 endpoint. It MUST be possible to identify such sessions end-to-end across the plurality of devices, networks, or administrative domains.

It is expected that the ITU-T will define protocol elements for H.323 to make the end-to-end signaling possible.

Derived Requirements: REQ7, REQ9a

3.4. Traffic Monitoring

UA A and UA B communicate using SIP messaging with a SIP B2BUA acting as a middlebox which belongs to a SIP service provider. For privacy reasons, the B2BUA changes the SIP headers that reveal information related to the SIP users, device or domain identity. The service provider uses an external device to monitor and log all SIP traffic coming to and from the B2BUA. In the case of failures reported by the customer or when security issue arise (e.g. theft of service), the service provider has to analyze the logs from the past several days or weeks and correlates those messages which were messages for a single end-to-end SIP session.

For this scenario, we must consider three particular use cases:

- a) The UAs A and B support the end-to-end Session Identifier.

Derived Requirements: REQ1, REQ4, REQ5, REQ8.

- b) Only the UA A supports the end-to-end Session Identifier, the UA B does not.

Derived Requirements: REQ1, REQ4, REQ5, REQ7, REQ8.

- c) UA A and UA B do not support the end-to-end Session Identifier.

Derived Requirements: REQ1, REQ4, REQ5, REQ7, REQ8

3.5. DELETED

3.6. Tracking transferred sessions

It is difficult to track which SIP messages were involved in the same call across transactions, especially when invoking supplementary services such as call transfer or call join. The ability to track communications sessions as they are transferred, one side at a time, through until completion of the session (i.e., until a BYE is sent).

Derived Requirements: REQ1, REQ2, REQ10

3.7. Session Signal Logging

An after the fact search of SIP messages to determine which were part of the same transaction or call is difficult when B2BUAs and SBCs are involved in the signaling between UAs. Mapping more than one Call-ID together can be challenging because all of the values in SIP headers on one side of the B2BUA or SBC will likely be different than those on the other side. If multiple B2BUAs and/or SBCs are in the signaling path, more than two sets of header values will exist, creating more of a challenge. Creating a common header value through all SIP entities will greatly reduce any challenge for the purposes of debugging, communication tracking (such as for security purposes in case of theft of service), etc.

Derived Requirements: REQ1, REQ4, REQ7, REQ8

3.8. Identifier Porting to Other Protocols - RTCP

There may be a desire to associate SIP session signaling with corresponding media flows. To facilitate this association, it should be possible to insert the Session-ID into a media-related message, such as an RTCP sender report message. This association would allow, as an example, for network monitoring equipment to associate troubled network flows with the end-to-end SIP session signaling.

Derived Requirements: REQ9c

3.9. 3PCC Use Case

Third party call control refers to the ability of an entity to create a call in which communication is actually between two or more parties. For example, a B2BUA acting as a third party controller could establish a call between two SIP UA's using 3PCC procedures as described in section 4.1 of RFC 3725 the flow for which is reproduced below.

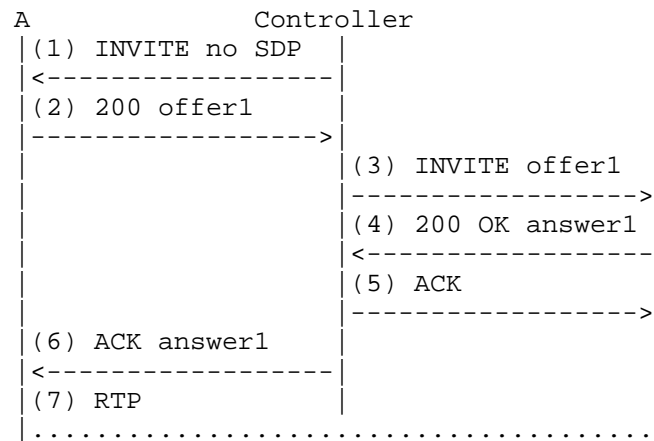


Figure 1 - Session-ID 3PCC Scenario

Such a flow must result in a single session identifier being used for the communication session between UA A and UA B. This use case does not extend to three SIP UAs.

Derived Requirements: REQ10

4. Requirements for the End-to-End Session Identifier

The following requirements are derived from the use cases and additional constraints regarding the construction of the identifier.

REQ1: It must be possible for an administrator or an external device which monitors the SIP-traffic to use the identifier to identify those dialogs, transactions and messages which were at some point in time components of a single end-to-end SIP session (e.g., parts of the same call).

REQ2: It must be possible to correlate two end-to-end sessions when a session is transferred or if two different sessions are joined together via an intermediary (e.g., a PBX). This might result in a change in the value of the end-to-end Session-Identifier.

REQ4: It must be possible to pass the identifier unchanged through SIP B2BUAs or other intermediaries.

REQ5: The identifier must not reveal any information related to any SIP user, device or domain identity. This includes any IP Address, port, hostname, domain name, username, Address-of-Record, MAC address, IP address family, transport type, subscriber ID, Call-ID, tags, or other SIP header or body parts.

REQ7: It must be possible to identity SIP traffic with an end-to-end session identifier from and to end devices that do not support this new identifier, such as by allowing an intermediary to inject an identifier into the session signaling.

REQ8: The identifier should be unique in time and space, similar to the Call-ID.

REQ9a: The identifier should be constructed in such a way as to make it suitable for transmission in SIP and H.323.

REQ9c: The identifier should be constructed in such a way as to make it suitable for transmission in SIP and RTCP [3].

REQ10: It must be possible to correlate two end-to-end sessions when the sessions are created by a third party controller using 3PCC procedures shown in Figure 1 of RFC 3725 [6].

5. Related Work in other Standards Organizations

5.1. Coordination with the ITU-T

IP multimedia networks are often comprised of a mix of session protocols like SIP and H.323. A benefit of the Session Identifier is that it uniquely identifies a communication session end-to-end across session protocol boundaries. Therefore, the need for coordinated standardization activities across Standards Development Organizations (SDOs) is imperative.

To facilitate this, a parallel effort is underway in the ITU-T to introduce the Session Identifier for the H.323 protocol. The ITU-T SG16 has approved contribution C.552 [4] as a work item with the intent that it be a coordinated and synchronized effort between the ITU-T and the IETF.

5.2. Requirements within 3GPP

3GPP identified in their Release 9 the need for a Session Identifier for O&M purposes to correlate flows in an end-to-end communication session. TS24.229 (IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP)) [5] points to the fact that the Session Identifier can be used to correlate SIP messages belonging to the same session. In the case where signaling passes through SIP entities like B2BUAs, the end-to-end session identifier indicates that these dialogs belong to the same end-to-end SIP communication session.

6. Security Considerations

An end-to-end identifier, if not properly constructed, could provide information that would allow one to identify the individual, device, or domain initiating or terminating a communication session. In adherence with REQ5, the solution produced in accordance with these requirements MUST NOT provide any information that allow one to identify a person, device, or domain. This means that information elements such as the MAC address or IP address MUST NOT be used when constructing the end-to-end session identifier.

7. IANA Considerations

There are no IANA considerations associated with this document.

8. Acknowledgments

The authors would like to acknowledge Chris Pearce for his contribution and collaboration in developing this document.

This document was prepared using 2-Word-v2.0.template.dot.

9. References

9.1. Normative References

- [1] Rosenberg, J., et al., "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] Recommendation ITU-T H.323, "Packet-based multimedia communications systems", December 2009.

9.2. Informative References

- [3] Schulzrinne, H., et al., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.
- [4] International Telecommunications Union, "End-to-End Session Identifier for IP-based Multimedia Communication Systems", March 2011, ITU-T Contribution C.552, http://ftp3.itu.int/av-arch/avc-site/2009-2012/1103_Gen/SessionID.zip.
- [5] 3GPP, "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3", 3GPP TS 24.229 10.3.0, April 2011.
- [6] Rosenberg, J., Peterson, J., Schulzrinne, H., Camarillo, G., "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", RFC 3725, April 2004.

Author's Addresses

Roland Jesske
Deutsche Telekom NP
64295 Darmstadt
Heinrich-Hertz-Str. 3-7
Germany

Phone: +49 6151 628 2766
Email: R.Jesske@telekom.de

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com
IM: xmpp:paulej@packetizer.com

Hadriel Kaplan
Acme Packet
71 Third Ave.
Burlington, MA 01803, USA

Email: hkaplan@acmepacket.com

Laura Liess
Deutsche Telekom NP
64295 Darmstadt
Heinrich-Hertz-Str. 3-7
Germany

Phone: +49 6151 268 2761
Email: laura.liess.dt@gmail.com

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: salvatore.loreto@ericsson.com

James Polk
Cisco Systems, Inc.
3913 Treemont Circle
Colleyville, Texas,
USA

Phone: +1 817 271 3552
Email: jmpolk@cisco.com
IM: xmpp:jmpolk@cisco.com

Parthasarathi Ravindran
Sonus Networks, Inc.
Prestige Shantiniketan - Business Precinct
Whitefield Road
Bangalore, Karnataka 560066
India

Email: pravindran@sonusnet.com

Gonzalo Salgueiro
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 392 3266
Email: gsalguei@cisco.com
IM: xmpp:gsalguei@cisco.com

Network Working Group
Internet Draft
Intended status: Informational
Expires: August 8, 2014

Paul E. Jones
Gonzalo Salgueiro
James Polk
Cisco Systems
Laura Liess
Deutsche Telekom
Hadriel Kaplan
Oracle
February 8, 2014

Requirements for an End-to-End Session Identification in
IP-Based Multimedia Communication Networks
draft-ietf-insipid-session-id-reqts-11.txt

Abstract

This document specifies the requirements for an end-to-end session identifier in IP-based multimedia communication networks. This identifier would enable endpoints, intermediate devices, and management and monitoring systems to identify a session end-to-end across multiple SIP devices, hops, and administrative domains.

Status of this Memo

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

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

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

This Internet-Draft will expire on August 8, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

| | |
|--|----|
| 1. Introduction..... | 2 |
| 2. Conventions used in this document..... | 3 |
| 3. Terminology..... | 3 |
| 3.1. What does the Session Identifier Identify?..... | 3 |
| 3.2. Communication Session..... | 4 |
| 3.3. End-to-End..... | 5 |
| 4. Session Identifier Use Cases..... | 5 |
| 4.1. End-to-end identification of a communication session..... | 5 |
| 4.2. Protocol Interworking..... | 6 |
| 4.3. Traffic Monitoring..... | 6 |
| 4.4. Tracking transferred sessions..... | 6 |
| 4.5. Session Signal Logging..... | 7 |
| 4.6. Identifier Syntax..... | 7 |
| 4.7. 3PCC Use Case..... | 7 |
| 5. Requirements for the End-to-End Session Identifier..... | 8 |
| 6. Related Work in other Standards Organizations..... | 9 |
| 6.1. Coordination with the ITU-T..... | 9 |
| 6.2. Requirements within 3GPP..... | 9 |
| 7. Security Considerations..... | 9 |
| 8. IANA Considerations..... | 10 |
| 9. Acknowledgments..... | 10 |
| 10. Contributors..... | 10 |
| 11. References..... | 10 |
| 11.1. Normative References..... | 10 |
| 11.2. Informative References..... | 10 |
| Author's Addresses..... | 12 |

1. Introduction

IP-based multimedia communication systems like SIP [1] and H.323 [2] have the concept of a "call identifier" that is globally unique. The identifier is intended to represent an end-to-end communication session from the originating device to the terminating device. Such an identifier is useful for troubleshooting, session tracking, and so forth.

Unfortunately, there are a number of factors that mean that the current call identifiers defined in SIP and H.323 are not suitable for end-to-end session identification. Perhaps most significant is the fact that the syntax for the call identifier in SIP and H.323 is different between the two protocols. This important fact makes it impossible for call identifiers to be exchanged end-to-end when a network uses both of these session protocols.

Another reason why the current call identifiers are not suitable to identify the session end-to-end is that in real-world deployments

devices like Back-to-Back User Agents (B2BUAs) often change the values as the session signaling passes through. This is true even when a single session protocol is employed and not a byproduct of protocol interworking.

Lastly, identifiers that might have been used to identify a session end-to-end fail to meet that need when sessions are manipulated through supplementary service interactions. For example, when a session is transferred or if a private branch exchange (PBX) joins or merges two communication sessions together locally, the end-to-end properties of currently-defined identifiers are lost.

This document specifies the requirements for an end-to-end session identifier in IP-based multimedia communication networks. This identifier would enable endpoints, intermediate devices, and management and monitoring systems to identify a session end-to-end across multiple SIP devices, hops, and administrative domains.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Terminology

3.1. What does the Session Identifier Identify?

The identifier this document places requirements on, the session identifier, identifies a set of signaling messages associated with exactly two endpoints which, from each endpoint's perspective, are related to a single invocation of a communication application.

How the endpoints determine which signaling messages share a given identifier (that is, what constitutes a single invocation of a communication application) is intentionally left loosely defined.

The term "call" is often used as an example of such an invocation for voice and video communication, but different protocols and deployments define the scope of a "call" in different ways. For instance, some systems would associate all of the activity between all three parties involved in a transfer a single "call".

Similarly, the term "session" is often used as an example of such an invocation, but this term is overloaded to describe both signaling and media level interaction. A single invocation of the communication application, as described above, may involve multiple RTP "sessions" as described by RFC 3550 [4], possibly even multiple concurrent sessions.

In this document, unless otherwise qualified, the term "communication session", or simply "session", will refer only to the set of signaling messages identified by the common session identifier. That is, a "session" is a set of signaling messages associated with exactly two endpoints that, from each endpoint's perspective, are related to a single invocation of a communication application.

The requirements in this document put some constraints on what an endpoint will consider the same, or a different, invocation of a communication session. They also ensure that related sessions (as this document is using the term) can be correlated using only the session identifiers for each session. Again, what constitutes a "related" session is intentionally left loosely defined.

The definition considers messages associated with exactly two endpoints instead of messages sent between two endpoints to allow for intermediaries that create messages on an endpoint's behalf. It is possible that an endpoint may not see all of the messages in a session (as this document is using the term) associated with it.

This definition, and the requirements in this document that put some constraint on what an endpoint should consider the same, or a different, invocation of a communication session facilitates specifying an identifier that allows the two endpoints to use two entirely different protocols (hence potentially have different ideas of what a single invocation means) or use two applications that have a different idea of what a single invocation means.

3.2. Communication Session

A communication session may exist between two SIP user agents and that may pass through one or more intermediary devices, including B2BUAs or SIP proxies. For example:

```

UA-A                Middlebox(es)                UA-B
SIP message(s) -----[ ]---[ ]-----> SIP message(s)
SIP message(s) <-----[ ]---[ ]----- SIP message(s)

```

Figure 1 - Communication Session through Middlebox(es)

The following are examples of acceptable communication sessions as described in Section 3.1 and are not exhaustive:

- o A call directly between two user agents
- o A call between two user agents with one or more SIP middleboxes in the signaling path
- o A call between two user agents that was initiated using third-party call control (3PCC) [6]

- o A call between two user agents (e.g., between Alice and Carol) that results from a different communication session (e.g., Alice and Bob) wherein one of those user agents (Alice) is transferred to another user agent (Carol) using a REFER request or a re-INVITE request

The following are not considered communication sessions:

- o A call between any two user agents wherein two or more user agents are engaged in a conference call via a conference focus:
 - o each call between the user agent and the conference focus would be a communication session, and
 - o each of these is a distinct communication session.
- o A call between three user agents (e.g., Alice, Bob, and Carol) wherein the first user agent (Alice) ad hoc conferences the other two user agents (Bob and Carol)
 - o The call between Alice and Bob would be one communication session.
 - o The call between Alice and Carol would be a different communication session.

3.3. End-to-End

The term "end-to-end" in this document means the communication session from the point of origin, passing through any number of intermediaries, to the ultimate point of termination. It is recognized that legacy devices may not support the end-to-end session identifier. Since such an endpoint will not create a session identifier, an intermediary device that supports this identifier can inject an identifier into the session signaling.

4. Session Identifier Use Cases

4.1. End-to-end identification of a communication session

For SIP messaging that either does not involve SIP servers or only involves SIP proxies, the Call-ID header field value sufficiently identifies each SIP message within a transaction (see Section 17 of [1]) or dialog (see Section 12 of [1]). This is not the case when either B2BUAs or Session Border Controllers (SBCs) [7] are in the signaling path between User Agents (UAs). Therefore, we need the ability to identify each communication session through a single SIP header field regardless of which type of SIP servers are in the signaling path between UAs. For messages that create a dialog, each message within the same dialog MUST use the same session identifier.

Derived Requirements: All Requirements in Section 5

4.2. Protocol Interworking

A communication session might originate in an H.323 [2] endpoint and pass through an SBC before ultimately reaching a terminating SIP user agent. Likewise, a call might originate on a SIP user agent and terminate on an H.323 endpoint. It MUST be possible to identify such sessions end-to-end across the plurality of devices, networks, or administrative domains.

It is anticipated that the ITU-T will define protocol elements for H.323 to make the end-to-end signaling possible.

Derived Requirements: REQ5, REQ7

4.3. Traffic Monitoring

UA A and UA B communicate using SIP messaging with a SIP B2BUA acting as a middlebox which belongs to a SIP service provider. For privacy reasons, the B2BUA changes the SIP header fields that reveal information related to the SIP users, device or domain identities. The service provider uses an external device to monitor and log all SIP traffic coming to and from the B2BUA. In the case of failures reported by the customer or when security issues arise (e.g. theft of service), the service provider has to analyze the logs from the past several days or weeks and correlates those messages which were messages for a single end-to-end SIP session.

For this scenario, we must consider three particular use cases:

- a) The UAs A and B support the end-to-end session identifier.

Derived Requirements: REQ1, REQ3, REQ4, REQ6.

- b) Only the UA A supports the end-to-end session identifier, the UA B does not.

Derived Requirements: REQ1, REQ3, REQ4, REQ5, REQ6.

- c) UA A and UA B do not support the end-to-end session identifier.

Derived Requirements: REQ1, REQ3, REQ4, REQ5, REQ6

4.4. Tracking transferred sessions

It is difficult to track which SIP messages were involved in the same call across transactions, especially when invoking supplementary services such as call transfer or call join. There exists a need for the ability to track communication sessions as they are transferred, one side at a time, until completion of the session (i.e., until a BYE is sent).

Derived Requirements: REQ1, REQ2, REQ9

4.5. Session Signal Logging

An after-the-fact search of SIP messages to determine which messages were part of the same transaction or call is difficult when B2BUAs and SBCs are involved in the signaling between UAs. Mapping more than one Call-ID together can be challenging because all of the values in SIP header fields on one side of the B2BUA or SBC will likely be different than those on the other side. If multiple B2BUAs and/or SBCs are in the signaling path, more than two sets of header field values will exist, creating more of a challenge. Creating a common header field value through all SIP entities will greatly reduce any challenge for the purposes of debugging, communication tracking (such as for security purposes in case of theft of service), etc.

Derived Requirements: REQ1, REQ3, REQ5, REQ6

4.6. Identifier Syntax

A syntax that is too lax (e.g., one that allows special characters or a very long identifier) would make it difficult to encode the identifier in other protocols. Therefore, the syntax of the identifier should be reasonably constrained.

Derived Requirements: REQ8

4.7. 3PCC Use Case

Third party call control refers to the ability of an entity to create a call in which communication is actually between two or more parties other than the one setting up the call. For example, a B2BUA acting as a third party controller could establish a call between two SIP UA's using 3PCC procedures as described in section 4.1 of RFC 3725 [6], the flow for which is reproduced below.

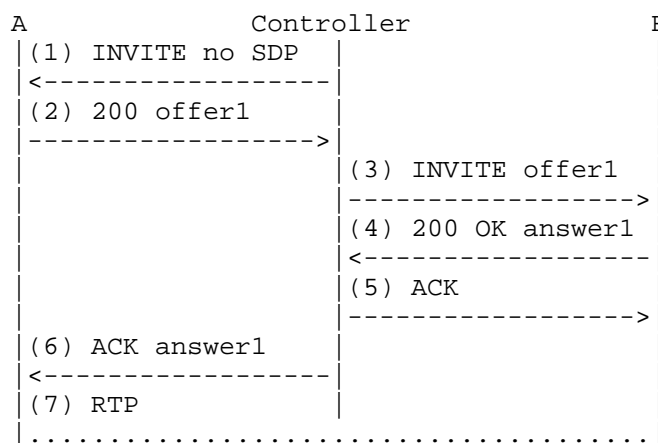


Figure 2 - Session Identifier 3PCC Scenario

Such a flow must result in a single session identifier being used for the communication session between UA A and UA B. This use case does not extend to three SIP UAs.

Derived Requirements: REQ9

5. Requirements for the End-to-End Session Identifier

The following requirements are derived from the use cases and additional constraints regarding the construction of the identifier.

REQ1: It MUST be possible for an administrator or an external device which monitors the SIP-traffic to use the identifier to identify those dialogs, transactions and messages which were at some point in time components of a single end-to-end SIP session (e.g., parts of the same call).

REQ2: It MUST be possible to correlate two end-to-end sessions when a session is transferred or if two different sessions are joined together via an intermediary (e.g., a PBX).

REQ3: The solution MUST require that the identifier, if present, pass unchanged through SIP B2BUAs or other intermediaries.

REQ4: The identifier MUST NOT reveal any information related to any SIP user, device or domain identity. Additionally, it MUST NOT be possible to correlate a set of session identifiers produced over a period of time with one another, or with a particular user or device. This includes any IP Address, port, hostname, domain name, username, Address-of-Record, MAC address, IP address family, transport type, subscriber ID, Call-ID, tags, or other SIP header field or body parts.

REQ5: It MUST be possible to identify SIP traffic with an end-to-end session identifier from and to end devices that do not support this new identifier, such as by allowing an intermediary to inject an identifier into the session signaling.

REQ6: The identifier SHOULD be unique in time and space, similar to the Call-ID.

REQ7: The identifier SHOULD be constructed in such a way as to make it suitable for transmission in SIP [1] and H.323 [2].

REQ8: The identifier SHOULD use a restricted syntax and length so as to allow the identifier to be used in other protocols.

REQ9: It MUST be possible to correlate two end-to-end sessions when the sessions are created by a third party controller using 3PCC procedures shown in Figure 1 of RFC 3725 [6].

6. Related Work in other Standards Organizations

6.1. Coordination with the ITU-T

IP multimedia networks are often comprised of a mix of session protocols like SIP [1] and H.323 [2]. A benefit of the session identifier is that it uniquely identifies a communication session end-to-end across session protocol boundaries. Therefore, the need for coordinated standardization activities across Standards Development Organizations (SDOs) is imperative.

To facilitate this, a parallel effort is underway in the ITU-T to introduce the session identifier for H.323 in such a way as to be interoperable with the procedures defined by the IETF.

6.2. Requirements within 3GPP

3GPP identified in their Release 9 the need for a session identifier for operation and maintenance purposes to correlate flows in an end-to-end communication session. 3GPP TS24.229 [5] points to the fact that the session identifier can be used to correlate SIP messages belonging to the same session. In the case where signaling passes through SIP entities like B2BUAs, the end-to-end session identifier indicates that these dialogs belong to the same end-to-end SIP communication session.

7. Security Considerations

The security vulnerabilities, attacks, and threat models affecting other similar SIP identifiers are well documented in RFC 3261 and are equally applicable to the end-to-end session identifier and subject to the same mitigating security best practices. Further, storage of the Session Identifier in a log file is also subject to the security considerations specified in RFC 6872 [8].

An end-to-end identifier, if not properly constructed, could provide confidential information that would allow one to identify the individual, device, or domain initiating or terminating a communication session. In adhering to REQ4, the solution produced in accordance with these requirements MUST take appropriate measures to properly secure and obfuscate sensitive or private information that might allow one to identify a person, device, or domain. This means that the end-to-end session identifier MUST NOT reveal information elements such as the MAC address or IP address. It is outside the scope of this document to specify the implementation details of such security and privacy measures. Those details may vary with the specific construction mechanism selected for the end-to-end session identifier and, therefore, will be discussed in suitable detail in the solution document specifying the actual end-to-end identifier.

A key security consideration is to ensure that an attacker cannot surreptitiously spoof the identifier and effectively render it

useless to diagnostic equipment that cannot properly correlate signaling messages due to the duplicate session identifiers that exist in the same space and time. In accordance with REQ6, this end-to-end identifier MUST be sufficiently long and random to prevent it from being guessable as well as avoid collision with another identifier. The secure transport of the identifier, need for authentication, encryption, etc. should be appropriately evaluated based on the network infrastructure, transport domain and usage scenarios for the end-to-end session identifier.

8. IANA Considerations

There are no IANA considerations associated with this document.

9. Acknowledgments

The authors would like to acknowledge Paul Kyzivat, Christer Holmberg, Charles Eckel, Andy Hutton, Salvatore Loreto, Keith Drage, Chris Pearce for their contribution and collaboration in developing this document.

This document was prepared using 2-Word-v2.0.template.dot.

10. Contributors

Two other people originally participated as co-authors and provided substantial contributions to this document, namely Roland Jesske, Parthasarathi Ravindran.

11. References

11.1. Normative References

- [1] Rosenberg, J., et al., "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] Recommendation ITU-T H.323, "Packet-based multimedia communications systems", December 2009.
- [3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

- [4] Schulzrinne, H., et al., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.
- [5] 3GPP TS 24.229, "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3".

- [6] Rosenberg, J., Peterson, J., Schulzrinne, H., Camarillo, G.,
"Best Current Practices for Third Party Call Control (3pcc) in
the Session Initiation Protocol (SIP)", RFC 3725, April 2004.
- [7] Hautakorpi, J., Camarillo, G., Penfield, R., Hawrylyshen, A.,
and M. Bhatia, "Requirements from Session Initiation Protocol
(SIP) Session Border Control (SBC) Deployments", RFC 5853,
April 2010.
- [8] Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., Festor, O.,
"The Common Log Format (CLF) for the Session Initiation
Protocol (SIP): Framework and Information Mode", RFC 6872,
February 2013.

Author's Addresses

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com
IM: xmpp:paulej@packetizer.com

Hadriel Kaplan
Oracle
71 Third Ave.
Burlington, MA 01803, USA

Email: hadriel.kaplan@oracle.com

Laura Liess
Deutsche Telekom NP
64295 Darmstadt
Heinrich-Hertz-Str. 3-7
Germany

Phone: +49 6151 268 2761
Email: laura.liess.dt@gmail.com

James Polk
Cisco Systems, Inc.
3913 Treemont Circle
Colleyville, Texas,
USA

Phone: +1 817 271 3552
Email: jmpolk@cisco.com
IM: xmpp:jmpolk@cisco.com

Gonzalo Salgueiro
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 392 3266
Email: gsalguei@cisco.com
IM: xmpp:gsalguei@cisco.com

Network Working Group
Internet Draft
Intended status: Standards Track
Expires: April 22, 2013

P. Jones
C. Pearce
J. Polk
G. Salgueiro
Cisco Systems
October 22, 2012

End-to-End Session Identification in IP-Based Multimedia
Communication Networks
draft-jones-insipid-session-id-01.txt

Abstract

This document describes an end-to-end Session Identifier for use in IP-based Multimedia Communication systems that enables endpoints, intermediate devices, and management systems to identify a session end-to-end, associate multiple endpoints with a given multipoint conference, track communication sessions when they are redirected, and associate one or more media flows with a given communication session.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on April 22, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

| | |
|--|----|
| 1. Introduction..... | 2 |
| 2. Conventions used in this document..... | 3 |
| 3. Session Identifier Requirements and Use Cases..... | 3 |
| 4. Constructing the Session Identifier..... | 3 |
| 5. Transmitting the Session Identifier in SIP..... | 4 |
| 6. Endpoint Behavior..... | 5 |
| 7. Processing by Intermediaries..... | 6 |
| 8. Associating Endpoints in a Multipoint Conference..... | 6 |
| 9. Correlating Media Flows with Sessions..... | 7 |
| 10. Various Call Flow Operations Utilizing the Session ID..... | 7 |
| 10.1. Basic Session-ID Construction with 2 UUIDs..... | 7 |
| 10.2. Basic Call Transfer using REFER..... | 8 |
| 10.3. Basic Call Transfer using reINVITE..... | 10 |
| 10.4. Single Focus Conferencing..... | 11 |
| 10.5. Single Focus Conferencing using WebEx..... | 12 |
| 10.6. Basic 3PCC for two UAs..... | 13 |
| 11. Compatibility with a Previous Implementation..... | 14 |
| 12. Security Considerations..... | 15 |
| 13. IANA Considerations..... | 15 |
| 14. Acknowledgments..... | 16 |
| 15. References..... | 16 |
| 15.1. Normative References..... | 16 |
| 15.2. Informative References..... | 16 |
| Author's Addresses..... | 17 |

1. Introduction

IP-based multimedia communication systems like SIP [1] and H.323 [2] have the concept of a "call identifier" that is globally unique. The identifier is intended to represent an end-to-end communication session from the originating device to the terminating device. Such an identifier is useful for troubleshooting, billing, session tracking, and so forth.

Unfortunately, there are a number of factors that contribute to the fact that the current call identifiers defined in SIP and H.323 are not suitable for end-to-end session identification. A fundamental issue in protocol interworking is the fact that the syntax for the call identifier in SIP and H.323 is different between the two protocols. This important fact makes it impossible for call identifiers to be exchanged end-to-end when a network utilizes one or more session protocols.

Another reason why the current call identifiers are not suitable to identify the session end-to-end is that in real-world deployments devices like session border controllers often change the session signaling as it passes through the device, including the value of the call identifier. While this is deliberate and useful, it makes it very difficult to track sessions end-to-end.

This draft presents a new identifier, referred to as the Session Identifier, or "Session ID", and associated syntax intended to overcome the issues that exist with the currently defined call identifiers. The proposal in this document attempts to comply with the requirements specified in [5]. This proposal also has capabilities not mentioned in [5], shown in call flows in section 10. Additionally, this proposal attempts to account for a previous, proprietary version of a SIP Session ID header, proposing a backwards compatibility of sorts, described in section 11.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Session Identifier Requirements and Use Cases

Requirements and Use Cases for the end-to-end Session Identifier can be found in a separate memo titled "Requirements for an End-to-End Session Identification in IP-Based Multimedia Communication Networks" [5].

4. Constructing the Session Identifier

The Session Identifier is comprised of two RFC 4122 defined UUIDs [4], with each UUID created by the endpoints participating in the session. The SIP user agent (UA) initially transmitting the SIP request will create a UUID and transmit that to the ultimate destination UA. Likewise, the responding UA will create a UUID and transmit that to the first UA. These two distinct UUIDs form what is

referred to as the Session Identifier and is represented in this document in set notation of the form {A,B}, where A is UUID value from the UA transmitting a message and B is the UUID value from the intended recipient of the message, i.e., not an intermediary server along the signaling path. The set {A,B} is equal to the set {B,A}, and thus both represent the same Session Identifier.

In the case where only one UUID is known, such as when a UA first initiates a SIP request, the Session ID would be {A}, where "A" represents the single UUID value transmitted.

Devices that act upon the Session Identifier may wish to represent these UUID values in some manner other than a pair of distinct values for, as examples, logging or internal value comparison. A device MAY take the two UUID values and produce a single 32-octet binary value that can be efficiently compared with other values. When constructing a single binary value out of the component UUIDs, it is RECOMMENDED that devices perform a binary comparison of the two UUIDs, starting with the most significant byte of each UUID. The UUID with the higher binary value is placed after the UUID with the lower binary value. As an example, if the Session Identifier {A,B} is stored and treated as a single binary value and "A" is numerically greater than "B", then the two values would be concatenated as B||A. When only one UUID value is known, entities MAY assume the absent UUID has a value of zero (i.e., 16 octets with a zero value).

Consider the following example.

Endpoint 1 produces this UUID: 0xaeffa652b22911dfa81f12313a006823

Endpoint 2 produces this UUID: 0xbellaafc8b22911df86c412313a006823

The resulting concatenated Session Identifier would be:
0xaeffa652b22911dfa81f12313a006823bellaafc8b22911df86c412313a006823

In the above example, the UUIDs are presented as a string of hexadecimal characters that correspond to the binary values comprising the UUID as shown in the table at the end of Section 4.1.2 of RFC 4122 [4].

How a device acting on Session Identifiers stores, processes, or utilizes the Session Identifier is outside the scope of this document.

5. Transmitting the Session Identifier in SIP

Each session initiated or accepted MUST have a local UA-generated UUID associated with the session. This value MUST remain unchanged throughout the duration of that session.

A SIP UA MUST convey its Session Identifier UUID in all transmitted messages within the same session. To do this, each transmitted message MUST include the "Session-ID" header. The Session-ID header has the following syntax:

```
Session-ID = "Session-ID" HCOLON sess-id
              ( SEMI rcvr-uuid )
              *( SEMI generic-param )

sess-id      = 32(DIGIT / %x61-66) ;32 chars of [0-9a-f]

rcvr-uuid    = "rcvr" EQUAL 32(DIGIT / %x61-66)
```

The "sess-id" value represents the UUID value of the UA transmitting the message. If the UA transmitting the message previously received a UUID value from its peer endpoint, it MUST include that UUID as the "rcvr" parameter. For example, using the UUID values from the previous section, a Session-ID header might appear like this:

```
Session-ID: aeffa652b22911dfa81f12313a006823;
           rcvr=bellaf8b22911df86c412313a006823
```

The UUID values are presented as strings of hexadecimal characters, with the most significant byte of the UUID appearing first.

6. Endpoint Behavior

To comply with this specification, SIP UAs MUST include a Session-ID header-value in all messages transmitted as a part of a communication session. Session-ID header-values MUST NOT be present in any other SIP header than the Session-ID header.

A non-intermediary UAS that receives a Session-ID header MUST take note of the first UUID value that it receives in the Session-ID header and assume that that is the UUID of the peer endpoint within that communications session. UAs MUST include this received UUID value as the "rcvr" parameter when transmitting subsequent messages.

UAs MUST ignore the value in the "rcvr" parameter in any message it receives, as this value may be incorrect due to service interactions as shown in examples later in this document.

For any purpose the UA has for the Session-ID, it MUST assume that the Session-ID is {A,B} where "A" is the UUID value of this endpoint and "B" is the UUID value of the peer endpoint, taken from the most recently received message within this session.

An endpoint MUST assume that the UUID value of the peer UA MAY change at any time due to service interactions. However, once an UA

allocates a UUID value for a communication session, the UA MUST NOT change that UUID value for the duration of the session, including when communication attempts are retried due to receipt of 4xx messages, when the session is redirected in response to a 3xx message, or when a session is transferred via a REFER message [6].

It is also important to note that if a session is forked by an intermediary in the network, the initiating UA may receive multiple responses back from different endpoints, each of which will contain a different UUID value. UAs MUST take care to ensure that the correct UUID value is returned in the "rcvr" parameter when responding to those endpoints.

7. Processing by Intermediaries

Intermediaries that wish to utilize the Session-ID MAY extract the UUID header-values from any SIP message. Alternatively, intermediaries MAY observe the first UUID value in the Session-ID header for messages sent in each direction and use those values to locally construct the Session Identifier.

Intermediaries MUST NOT alter the UUID values found in the Session-ID header, except as described in this section.

If performing interworking between SIP and another session protocol, an intermediary MUST convert the Session-ID header as necessary so that it properly places the correct UUID value of the message transmitter and assumed recipient. This is a protocol gateway or interworking function.

Intermediary devices that transfer a call, such as by joining together two different "call legs", MUST properly construct a Session-ID header that contains the correct UUID values and correct placement of those values. As described above, the recipient of any message initiated by the intermediary will assume that the first UUID value belongs to the peer endpoint.

Devices that initiate communication sessions following the procedures for third party call control MUST fabricate a UUID value that will be utilized only temporarily. Once the responding endpoint provides a UUID value in a response message, the temporary value MUST be discarded and replaced with the endpoint-provided UUID value. Refer to the third-party call control example for an illustration.

8. Associating Endpoints in a Multipoint Conference

Multipoint Control Units (MCUs) group two or more sessions into a single multipoint conference. The MCU should utilize the same UUID value for each session that is grouped into the same conference. In

so doing, each individual session in the conference will have a unique Session Identifier (since each endpoint will create a unique UUID of its own), but will also have one UUID in common with all other participants in the conference.

Intermediary devices, such as proxies or session border controllers, or network diagnostics equipment might assume that when they see two or more sessions with different Session Identifiers, but with one UUID in common, that the sessions are part of the same conference.

Note, however, that this assumption of being part of the same conference is not always true. For example, in a SIP forking scenario, there might also be what appears to be multiple sessions with a shared UUID value. This is actually desirable. What is desired is to allow for the association of related sessions. Whether sessions are related because of forking or because endpoints are communicating as a part of a conference does not matter. They are nonetheless related.

9. Correlating Media Flows with Sessions

It may be desirable to insert the Session Identifier header-value into media-related packets, such as RSVP messages or RTCP packets. In so doing, it is possible for network elements to

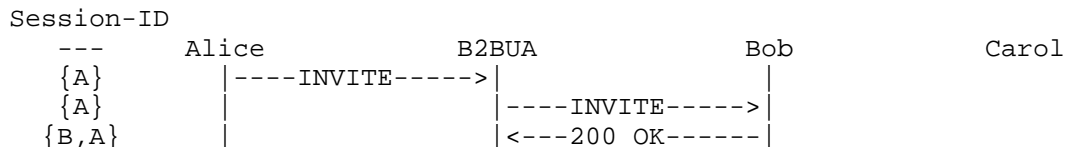
1. correlate session signaling with media flows;
2. associate multiple media flows with a single session; and
3. associate multiple media flows from multiple devices that are part of a single conference

Notwithstanding the foregoing, the use of the Session Identifier for purposes other than end-to-end session identification is outside the scope of this document.

10. Various Call Flow Operations Utilizing the Session ID

Seeing something frequently makes understanding easier. With that in mind, we include several call flows with the initial UUID and the complete Session-ID indicated per message, as well as when the Session-ID changes according to the rules within this document during certain operations/functions.

10.1. Basic Session-ID Construction with 2 UUIDs



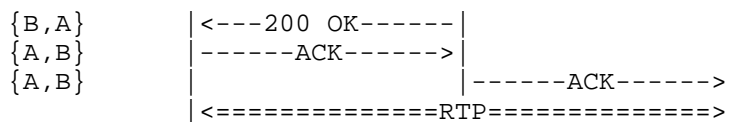
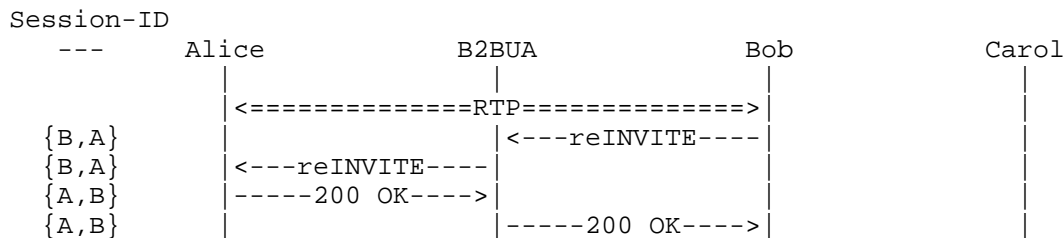


Figure 1 - Session-ID Creation when Alice calls Bob

Operation/Rules:

- o Transmitter of SIP message places its Session-ID UUID first in order;
- o UA-Alice sends its UUID in INVITE;
- o B2BUA receives an INVITE with a Session-ID header-value from UA-Alice, and transmits INVITE towards UA-Bob with an unchanged Session-ID header-value;
- o UA-Bob receives Session-ID and adds its UUID to construct the whole/complete Session-ID header-value in the 200 OK;
- o UA-Bob orders the UUIDs such that its UUID is first when UA-Bob is transmitting the SIP message;
- o B2BUA receives the 200 OK response with a complete Session-ID header-value from UA-Bob, and transmits 200 OK towards UA-Alice with an unchanged Session-ID header-value; while maintaining the order of UUIDs in the Session-ID header-value;
- o UA-Alice, upon reception of the 200 OK from the B2BUA, transmits the ACK towards the B2BUA with its UUID positioned first, and the UUID from UA-Bob positioned second in the Session-ID header-value.
- o B2BUA receives the ACK with a complete Session-ID header-value from UA-Alice, and transmits ACK towards UA-Bob with an unchanged Session-ID header-value; while maintaining the order of UUIDs in the Session-ID header-value;

10.2. Basic Call Transfer using REFER



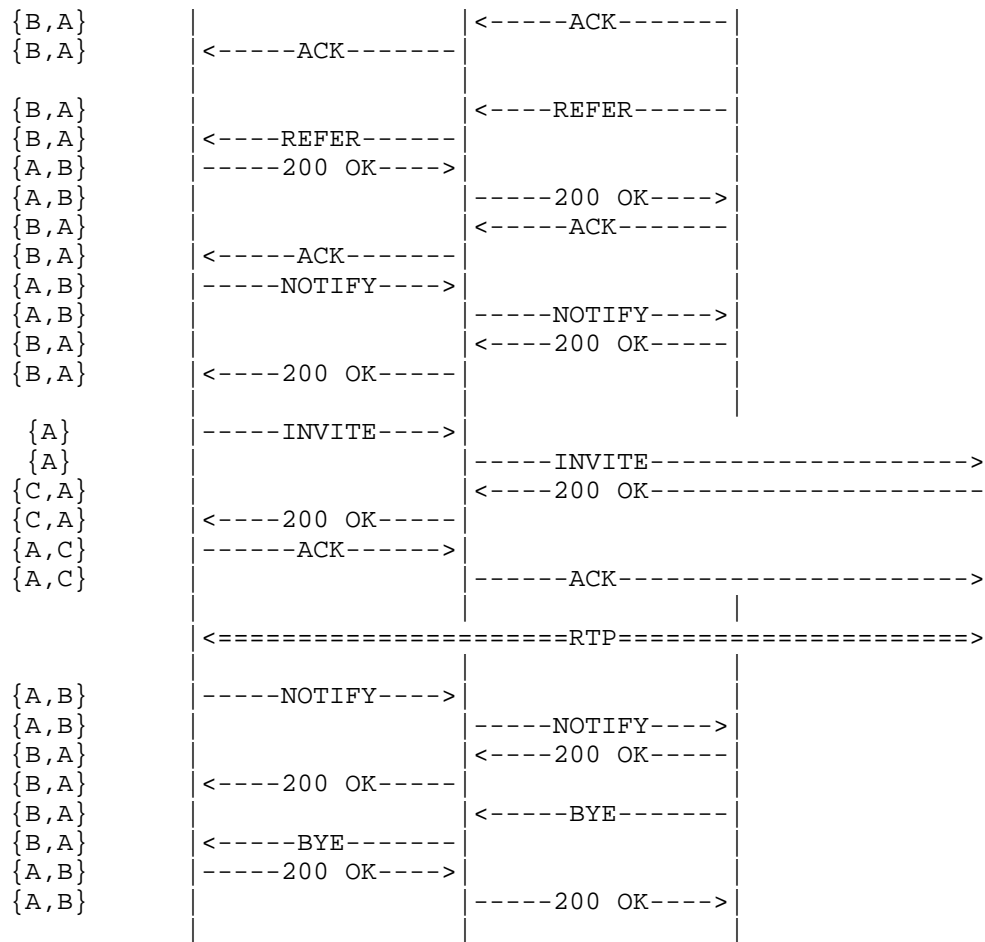


Figure 2 - Call Transfer using REFER

Operation/Rules:

Starting from the existing Alice/Bob call described in Figure 1, which established an existing Session-ID header-value...

- o UA-Bob reINVITES Alice to call Carol, using a REFER transaction, as described in [RFC3515]. UA-Alice is initially put on hold, then told in the REFER who to contact with a new INVITE, in this case UA-Carol.
- o UA-Alice retains her UUID from the Alice-to-Bob call {A} when requesting a call with UA-Carol. This same UUID traverses the B2BUA unchanged.

- o UA-Carol receives the INVITE with a Session-ID UUID {A}, creates its own UUID {C}, and combines them to form a full Session-ID {C,A} in the 200 OK to the INVITE. This Session-ID header-value traverses the B2BUA unchanged towards UA-Alice.
- o UA-Alice receives the 200 OK with the Session-ID {C,A} and both responses to UA-Carol with an ACK, generates a NOTIFY to Bob with a Session-ID {A,B} indicating the call transfer was successful.
- o It does not matter which UA terminates the Alice-to-Bob call; Figure 2 shows UA-Bob doing this transaction.

10.3. Basic Call Transfer using reINVITE

Alice is talking to Bob. Bob pushes a button on his phone to transfer Alice to Carol via the B2BUA (using reINVITE).

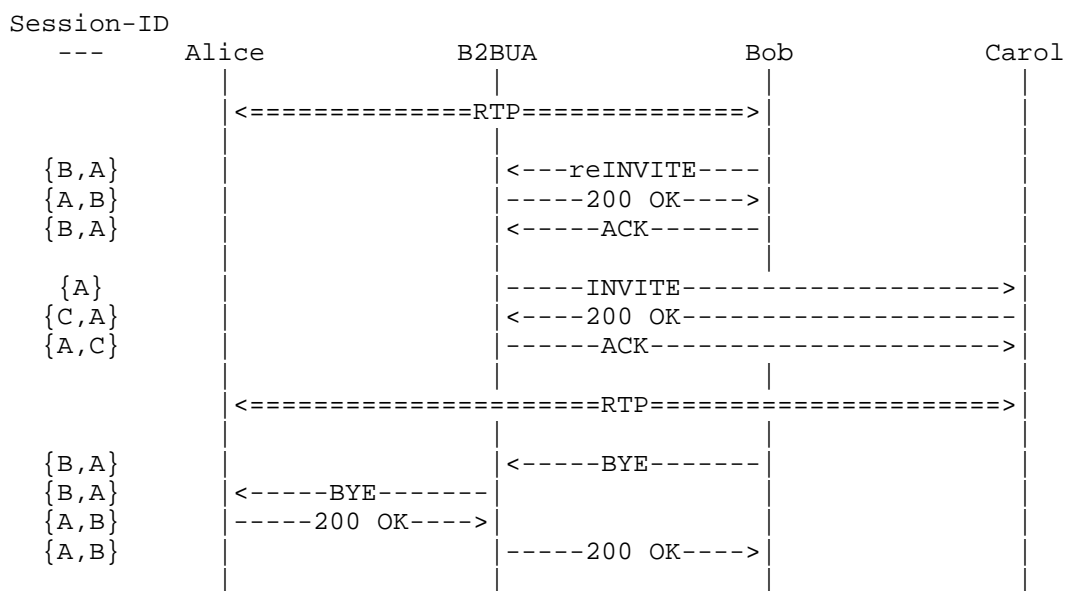


Figure 3 - Call transfer using reINVITE

Operation/Rules:

- o We assume the call between Alice and Bob from Section 10.1 is operational with Session-ID {A,B}.
- o Bob sends a reINVITE to Alice to transfer her to Carol.

- o The B2BUA intercepts this reINVITE and sends a new INVITE with Alice's UUID {A} to Carol.
- o Carol receives the INVITE and accepts the request and adds her UUID {C} to the Session-ID for this session {C,A}.
- o Bob terminates the call (which Alice could too) with a BYE using their Session-ID {B,A}.

10.4. Single Focus Conferencing

Multiple users call into a conference server (say an MCU) to attend one of many conferences hosted on or managed by that server. Each user has to identify which conference they want to join, but this information is not in the SIP messaging. Rather, it is done via an IVR. Thus, each user goes through a two-step process of signaling to gain entry onto their conference call.

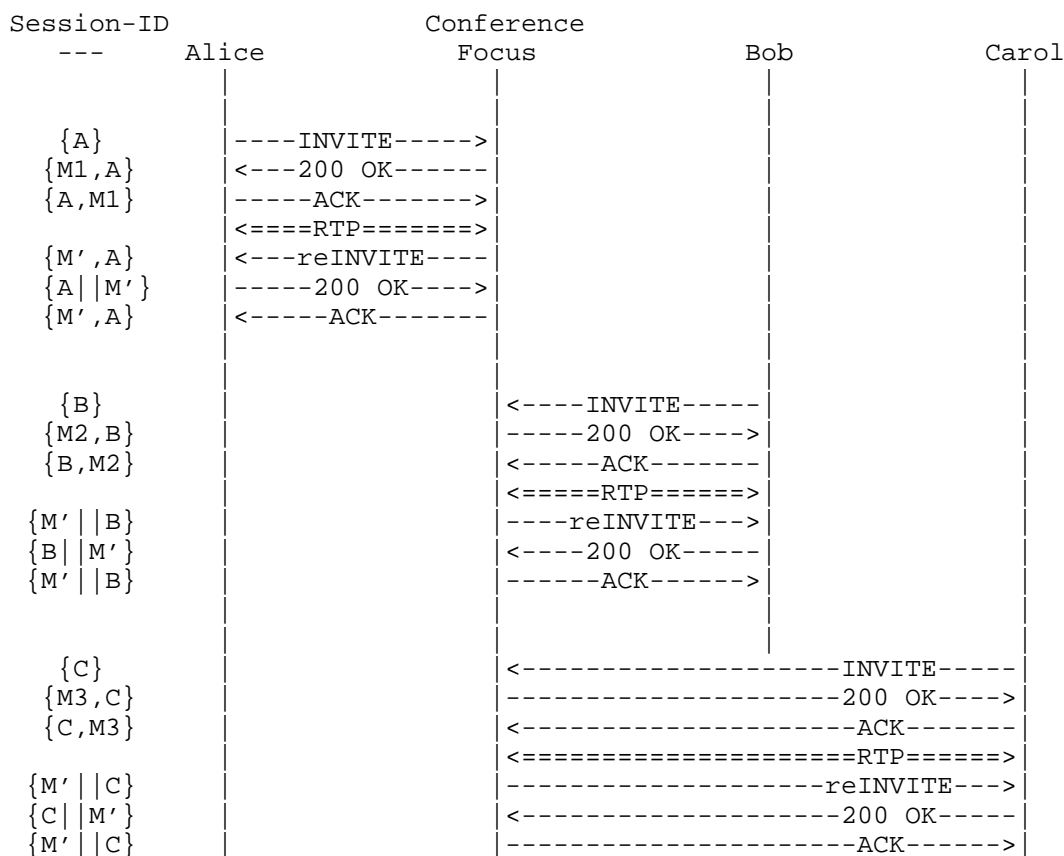


Figure 4 - Single Focus Conference Bridge

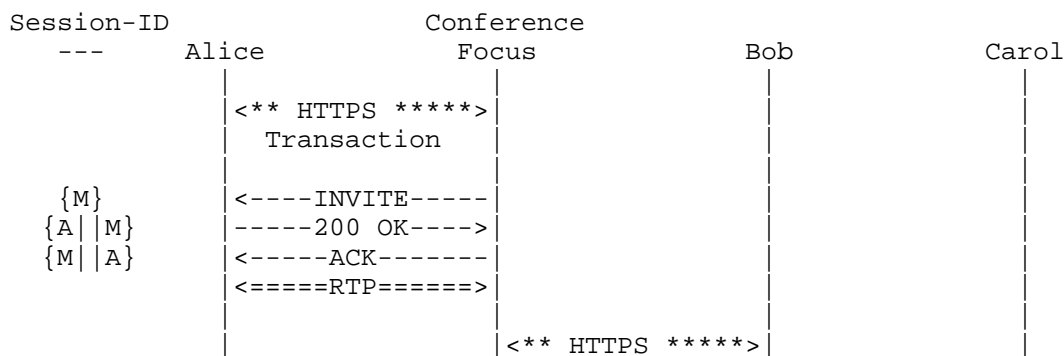
Operation/Rules:

Alice calls into a conference server to attend a certain conference. This is a two-step operation since Alice cannot include the conference ID and any passcode in the INVITE.

- o Alice sends an INVITE to the conference server with her UUID {A}.
- o The conference server accepts using a generic, temporary UUID {M1}.
- o Once Alice, the user, gains access to the IVR for this conference server, she enters a specific conference ID and whatever passcode (if needed) to enter a specific conference call.
- o Once the conference server is satisfied Alice has identified which conference she wants to attend (including any passcode verification), the conference server reINVITES Alice to the specific conference and includes the UUID {M'} for that conference. All valid participants in the same conference will receive this same UUID for identification purposes and to better enable monitoring, tracking and billing functions.
- o Bob goes through this two-step process of an INVITE transaction, followed by a reINVITE transaction to get this same UUID for that conference.
- o In this example, Carol (and each additional user) goes through the same procedures and steps as Alice to get on this same conference.

10.5. Single Focus Conferencing using WebEx

Alice, Bob and Carol call into same Webex conference.



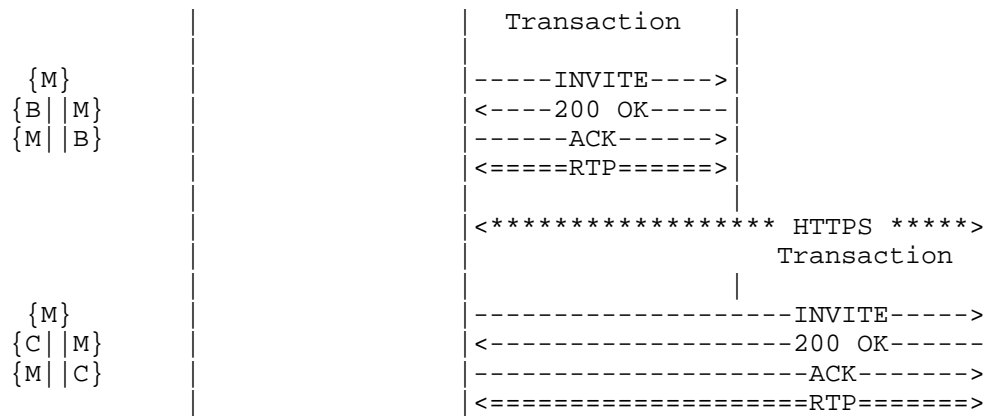


Figure 5 - Single Focus Webex Conference

Operation/Rules:

- o Alice communicates with Webex server with desire to join a certain meeting, by meeting number; also includes UA-Alice's contact information (phone number or URI).
- o Conference Focus server sends INVITE to UA-Alice to start session with the Session-ID of that server for this A/V conference call.
- o Bob and Carol perform same function to join this same A/V conference call as Alice.

10.6. Basic 3PCC for two UAs

External entity sets up call to both Alice and Bob for them to talk to each other.

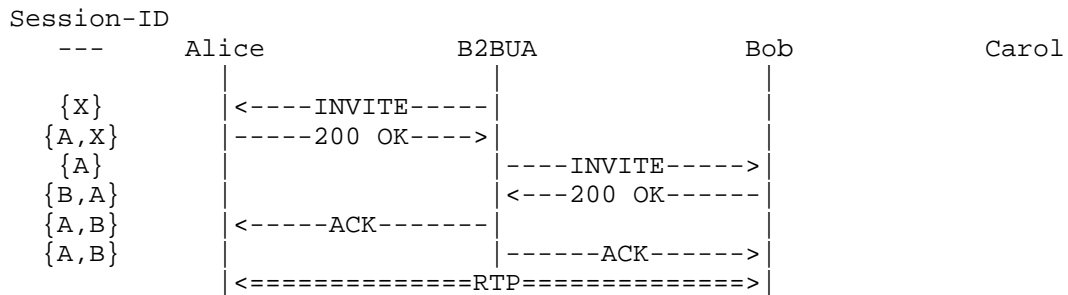


Figure 6 - 3PCC initiated call between Alice and Bob

Operation/Rules:

- o Some out of band procedure directs a B2BUA (or other SIP server) to have Alice and Bob talk to each other.
- o The SIP server INVITEs Alice to a session and uses a temporary UUID {X}.
- o Alice receives and accepts this call set-up and includes her UUID {A} in the Session-ID, now {A,X}.
- o The SIP server uses Alice's UUID {A}, and discards its own {X} to INVITE Bob to the session as if this came from Alice originally.
- o Bob receives and accepts this INVITE and adds his own UUID {B} to the Session-ID, now {B,A} for the response.
- o And the session is established.

11. Compatibility with a Previous Implementation

There is a much earlier and proprietary document that specifies the use of a Session-ID header that we will herewith attempt to achieve backwards compatibility. Neither Session-ID has any versioning information, so merely adding that this document describes "version 2" is insufficient. Here are the set of rules for compatibility between the two specifications. For the purposes of this discussion, we will label the proprietary specification of the Session-ID as version 0 and this specification as version 1 of the Session-ID.

The previous version only has a single value as a Session-ID, but has a generic-parameter value that can be of use.

In order to have a Version 0 talk to a Version 0 implementation, nothing needs to be done as far as the IETF is concerned.

In order to have a Version 1 talk to a Version 1 implementation, both implementations need to following this document (to the letter) and everything should be just fine.

In order to have a Version 0 talk to a Version 1 implementation, several aspects need to be looked at. They are:

- o The Version 0 UA will include a single UUID as its Session-ID.
- o The Version 1 UA will respond by including a complete Session-ID with two UUIDs, with the Version 1's UUID listed first (because it cannot know it is talking with a Version 0 implementation at this point).

- o The Version 0 UA will have to ignore the first UUID and react to the second UUID, which would be the recipient's Session-ID.
- o During subsequent transactions within this session, the Version 1 may receive SIP requests without its UUID, but with the Version 0's UUID. The Version 1 UA MUST add its UUID to the received Session-ID. The Version 0 implementation will merely disregard it each time it receives this Version 1 UUID (if it was not the first UUID).

In order to have a Version 1 talk to a Version 0 implementation, several aspects need to be looked at. They are:

- o The Version 1 UA will include a single UUID as its initial Session-ID header always, not knowing which version of UA it is communicating with.
- o The Version 0 UA will respond by seeing the UUID as a valid and complete Session-ID and not include another UUID or generic-param. Thus, the 200 OK will not include any Session-ID part of its own from the Version 0 implementation.
- o Open question - How do we want the Version 1 implementation interpret this?
- o Another open question - how do we want all intermediaries and/or monitoring/billing systems to interpret this single UUID complete Session-ID?

12. Security Considerations

When creating a UUID value, endpoints SHOULD ensure that there is no user or device-identifying information contained within the UUID. In some environments, though, use of a MAC address, which is one option when constructing a UUID, may be desirable, especially in some enterprise environments. When communicating over the Internet, though, the UUID value MUST utilize random values.

Other considerations???

13. IANA Considerations

The following is the registration for the 'Session-ID' header field to the "Header Name" registry at <http://www.iana.org/assignments/sip-parameters>:

RFC number: [this document]

Header name: 'Session-ID'

Compact form: none

14. Acknowledgments

The authors would like to thank Hadriel Kaplan and Christer Holmberg for their useful comments during the development of this document.

This document was prepared using 2-Word-v2.0.template.dot.

15. References

15.1. Normative References

- [1] Rosenberg, J., et al., "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] Recommendation ITU-T H.323, "Packet-based multimedia communications systems", December 2009.
- [3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [4] Leach, P., Mealling, M., Salz, R., "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005.
- [5] Jones, et al., "Requirements for an End-to-End Session Identification in IP-Based Multimedia Communication Networks", draft-jones-insipid-session-id-reqts-02.txt, October 2012.

15.2. Informative References

- [6] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.
- [7] Braden, R., et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.
- [8] Schulzrinne, H., et al., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

Author's Addresses

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com
IM: xmpp:paulej@packetizer.com

Chris Pearce
Cisco Systems, Inc.
2300 East President George Bush Highway
Richardson, TX 75082
USA

Phone: +1 972 813 5123
Email: chrep@cisco.com
IM: xmpp:chrep@cisco.com

James Polk
Cisco Systems, Inc.
3913 Treemont Circle
Colleyville, Texas, USAUSA

Phone: +1 817 271 3552
Email: jmpolk@cisco.com
IM: xmpp:jmpolk@cisco.com

Gonzalo Salgueiro
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 392 3266
Email: gsalguei@cisco.com
IM: xmpp:gsalguei@cisco.com

Network Working Group
Internet Draft
Intended status: Standards Track
Expires: August 20, 2013

P. Jones
C. Pearce
J. Polk
G. Salgueiro
Cisco Systems
February 20, 2013

End-to-End Session Identification in IP-Based Multimedia
Communication Networks
draft-jones-insipid-session-id-02.txt

Abstract

This document describes an end-to-end Session Identifier for use in IP-based Multimedia Communication systems that enables endpoints, intermediate devices, and management systems to identify a session end-to-end, associate multiple endpoints with a given multipoint conference, track communication sessions when they are redirected, and associate one or more media flows with a given communication session.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on August 20, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

| | |
|--|----|
| 1. Introduction..... | 2 |
| 2. Conventions used in this document..... | 3 |
| 3. Session Identifier Requirements and Use Cases..... | 3 |
| 4. Constructing the Session Identifier..... | 3 |
| 5. Transmitting the Session Identifier in SIP..... | 4 |
| 6. Endpoint Behavior..... | 5 |
| 7. Processing by Intermediaries..... | 6 |
| 8. Associating Endpoints in a Multipoint Conference..... | 7 |
| 9. Various Call Flow Operations Utilizing the Session ID..... | 7 |
| 9.1. Basic Session-ID Construction with 2 UUIDs..... | 8 |
| 9.2. Basic Call Transfer using REFER..... | 9 |
| 9.3. Basic Call Transfer using reINVITE..... | 10 |
| 9.4. Single Focus Conferencing..... | 11 |
| 9.5. Single Focus Conferencing using WebEx..... | 13 |
| 9.6. Cascading Conference Bridge Support for the Session-ID... | 14 |
| 9.7. Basic 3PCC for two UAs..... | 15 |
| 10. Compatibility with a Previous Implementation..... | 16 |
| 11. Security Considerations..... | 17 |
| 12. IANA Considerations..... | 17 |
| 13. Acknowledgments..... | 18 |
| 14. References..... | 18 |
| 14.1. Normative References..... | 18 |
| Author's Addresses..... | 19 |

1. Introduction

IP-based multimedia communication systems like SIP [1] and H.323 [2] have the concept of a "call identifier" that is globally unique. The identifier is intended to represent an end-to-end communication session from the originating device to the terminating device. Such an identifier is useful for troubleshooting, session tracking, and so forth.

Unfortunately, there are a number of factors that contribute to the fact that the current call identifiers defined in SIP and H.323 are not suitable for end-to-end session identification. A fundamental issue in protocol interworking is the fact that the syntax for the call identifier in SIP and H.323 is different between the two protocols. This important fact makes it impossible for call identifiers to be exchanged end-to-end when a network utilizes one or more session protocols.

Another reason why the current call identifiers are not suitable to identify the session end-to-end is that in real-world deployments devices like session border controllers often change the session signaling as it passes through the device, including the value of the call identifier. While this is deliberate and useful, it makes it very difficult to track sessions end-to-end.

This draft presents a new identifier, referred to as the Session Identifier, or "Session ID", and associated syntax intended to overcome the issues that exist with the currently defined call identifiers. The proposal in this document attempts to comply with the requirements specified in Error! Reference source not found.. This proposal also has capabilities not mentioned in [5], shown in call flows in section 10. Additionally, this proposal attempts to account for a previous, proprietary version of a SIP Session ID header, proposing a backwards compatibility of sorts, described in section 11.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Session Identifier Requirements and Use Cases

Requirements and Use Cases for the end-to-end Session Identifier can be found in a separate memo titled "Requirements for an End-to-End Session Identification in IP-Based Multimedia Communication Networks" Error! Reference source not found..

4. Constructing the Session Identifier

The Session Identifier is comprised of two RFC 4122 defined UUIDs [4], with each UUID representing one of the endpoints participating in the session. The SIP user agent (UA) initially transmitting the SIP request will create a UUID and transmit that to the ultimate destination UA. Likewise, the responding UA will create a UUID and

transmit that to the first UA. These two distinct UUIDs form what is referred to as the Session Identifier and is represented in this document in set notation of the form {A,B}, where A is UUID value from the UA transmitting a message and B is the UUID value from the intended recipient of the message, i.e., not an intermediary server along the signaling path. The set {A,B} is equal to the set {B,A}, and thus both represent the same Session Identifier.

In the case where only one UUID is known, such as when a UA first initiates a SIP request, the Session ID would be {A}, where "A" represents the single UUID value transmitted.

Since SIP sessions are subject to any number of service interactions, SIP INVITE messages might be forked as sessions are established, and since conferences might be established or expanded with endpoints calling in or the conference focus calling out, the construction of the Session Identifier from a set of UUIDs is important.

To understand this better, consider that a UA participating in a communication session might be replaced with another, such as the case where two "legs" of a call are joined together by a PBX. Suppose that UA A and UA B both call UA C. Further suppose that UA C uses a local PBX function to join the call between itself and UA A with the call between itself and UA B. This merged call needs to be identified and identification of such sessions is natural and easily traceable when utilizing UUID values assigned by each entity in the communication session.

In the case of forking, UA A might send an INVITE that gets forked to five different UAs, as an example. Until one UA returns a 200 OK to the initial INVITE, a means of identifying each of these separate communication sessions is needed and allowing the set of {A, B1}, {A, B2}, {A, B3}, {A, B4}, and {A, B5} makes this possible.

For conferencing scenarios, it is also useful to have a two-part Session-ID where the conference focus specifies one UUID. This might allow for correlation among the participants in a single conference, for example.

How a device acting on Session Identifiers stores, processes, or utilizes the Session Identifier is outside the scope of this document.

5. Transmitting the Session Identifier in SIP

Each session initiated or accepted MUST have a local UA-generated UUID associated with the session. This value MUST remain unchanged throughout the duration of that session.

A SIP UA MUST convey its Session Identifier UUID in all transmitted messages within the same session. To do this, each transmitted message MUST include the "Session-ID" header. The Session-ID header has the following ABNF [5] syntax:

```
session-id      = "Session-ID" HCOLON local-uuid
                  *(SEMI sess-id-param)

local-uuid      = sess-uuid

remote-uuid     = sess-uuid

sess-uuid       = 32(DIGIT / %x61-66) ;32 chars of [0-9a-f]

sess-id-param   = remote-param / generic-param

remote-param    = "remote" EQUAL remote-uuid
```

The productions "SEMI", "EQUAL", and "generic-param" production is defined in RFC 3261. The production DIGIT is defined in RFC 5234.

The Session-ID header MUST NOT have more than one "remote" parameter.

The "local-uuid" in the Session-ID header represents the UUID value of the UA transmitting the message. If the UA transmitting the message previously received a UUID value from its peer endpoint, it MUST include that UUID as the "remote" parameter. For example, using the UUID values from the previous section, a Session-ID header might appear like this:

```
Session-ID: aeffa652b22911dfa81f12313a006823;
           remote=bellafc8b22911df86c412313a006823
```

The UUID values are presented as strings of lower-case hexadecimal characters, with the most significant byte of the UUID appearing first.

6. Endpoint Behavior

To comply with this specification, SIP UAs MUST include a Session-ID header-value in all messages transmitted as a part of a communication session.

A non-intermediary UAS that receives a Session-ID header MUST take note of the first UUID value that it receives in the Session-ID header and assume that that is the UUID of the peer endpoint within that communications session. UAs MUST include this received UUID

value as the "remote" parameter when transmitting subsequent messages.

It should be noted that messages received by a UA might contain a "remote" parameter that does match the UAs UUID. This might happen as a result of service interactions by intermediaries and MUST NOT be regarded as an error.

For any purpose the UA has for the Session-ID, it MUST assume that the Session-ID is {A,B} where "A" is the UUID value of this endpoint and "B" is the UUID value of the peer endpoint, taken from the most recently received message within this session.

An endpoint MUST assume that the UUID value of the peer UA MAY change at any time due to service interactions. However, once an UA allocates a UUID value for a communication session, the UA MUST NOT change that UUID value for the duration of the session, including when communication attempts are retried due to receipt of 4xx messages, when the session is redirected in response to a 3xx message, or when a session is transferred via a REFER message [6].

It is also important to note that if a session is forked by an intermediary in the network, the initiating UA may receive multiple responses back from different endpoints, each of which will contain a different UUID value. UAs MUST take care to ensure that the correct UUID value is returned in the "remote" parameter when responding to those endpoints.

7. Processing by Intermediaries

Intermediaries that wish to utilize the Session-ID MAY extract the UUID header-values from any SIP message. Alternatively, intermediaries MAY observe the first UUID value in the Session-ID header for messages sent in each direction and use those values to locally construct the Session Identifier.

Intermediaries MUST NOT alter the UUID values found in the Session-ID header, except as described in this section.

Intermediary devices that transfer a call, such as by joining together two different "call legs", MUST properly construct a Session-ID header that contains the correct UUID values and correct placement of those values. As described above, the recipient of any message initiated by the intermediary will assume that the first UUID value belongs to the peer endpoint.

If a SIP message having no Session-ID header is received by an intermediary, the intermediary MAY assign a "local-uuid" value to represent the sending endpoint and insert that value into all

signaling messages on behalf of the sending endpoint. If the intermediary is aware of a "remote" value that identifies the receiving UA, it MUST insert that value if also inserting the "local-uuid" value.

Devices that initiate communication sessions following the procedures for third party call control MUST fabricate a UUID value that will be utilized only temporarily. Once the responding endpoint provides a UUID value in a response message, the temporary value MUST be discarded and replaced with the endpoint-provided UUID value. Refer to the third-party call control example for an illustration.

Whenever there is a UA that does not implement this specification communicating through a B2BUA, the B2BUA MAY become dialog stateful and insert a UUID value into the Session-ID header on behalf of the UA according to the rules stated in Section 6.

8. Associating Endpoints in a Multipoint Conference

Multipoint Control Units (MCUs) group two or more sessions into a single multipoint conference. The MCU should utilize the same UUID value for each session that is grouped into the same conference. In so doing, each individual session in the conference will have a unique Session Identifier (since each endpoint will create a unique UUID of its own), but will also have one UUID in common with all other participants in the conference.

Intermediary devices, such as proxies or session border controllers, or network diagnostics equipment might assume that when they see two or more sessions with different Session Identifiers, but with one UUID in common, that the sessions are part of the same conference.

Note, however, that this assumption of being part of the same conference is not always true. For example, in a SIP forking scenario, there might also be what appears to be multiple sessions with a shared UUID value. This is actually desirable. What is desired is to allow for the association of related sessions. Whether sessions are related because of forking or because endpoints are communicating as a part of a conference does not matter. They are nonetheless related.

9. Various Call Flow Operations Utilizing the Session ID

Seeing something frequently makes understanding easier. With that in mind, we include several call flows with the initial UUID and the complete Session-ID indicated per message, as well as when the Session-ID changes according to the rules within this document during certain operations/functions.

9.1. Basic Session-ID Construction with 2 UUIDs

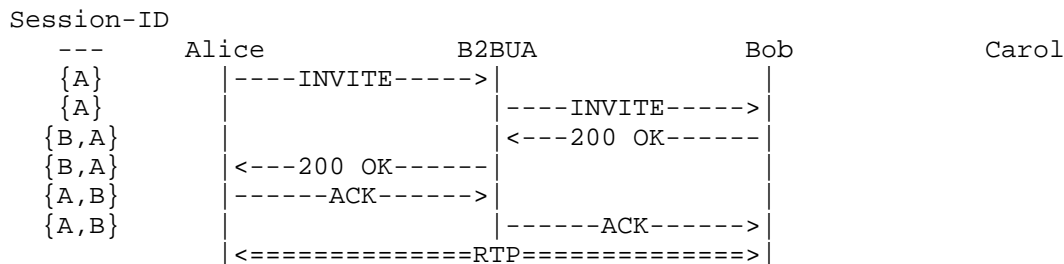


Figure 1 - Session-ID Creation when Alice calls Bob

Operation/Rules:

- o Transmitter of SIP message places its Session-ID UUID first in order;
- o UA-Alice sends its UUID in INVITE;
- o B2BUA receives an INVITE with a Session-ID header-value from UA-Alice, and transmits INVITE towards UA-Bob with an unchanged Session-ID header-value;
- o UA-Bob receives Session-ID and adds its UUID to construct the whole/complete Session-ID header-value in the 200 OK;
- o UA-Bob orders the UUIDs such that its UUID is first when UA-Bob is transmitting the SIP message;
- o B2BUA receives the 200 OK response with a complete Session-ID header-value from UA-Bob, and transmits 200 OK towards UA-Alice with an unchanged Session-ID header-value; while maintaining the order of UUIDs in the Session-ID header-value;
- o UA-Alice, upon reception of the 200 OK from the B2BUA, transmits the ACK towards the B2BUA with its UUID positioned first, and the UUID from UA-Bob positioned second in the Session-ID header-value.
- o B2BUA receives the ACK with a complete Session-ID header-value from UA-Alice, and transmits ACK towards UA-Bob with an unchanged Session-ID header-value; while maintaining the order of UUIDs in the Session-ID header-value;

9.2. Basic Call Transfer using REFER

From the example built within Section 9.1 (the basic session-ID establishment), we proceed to this 'Basic Call Transfer using REFER' example.

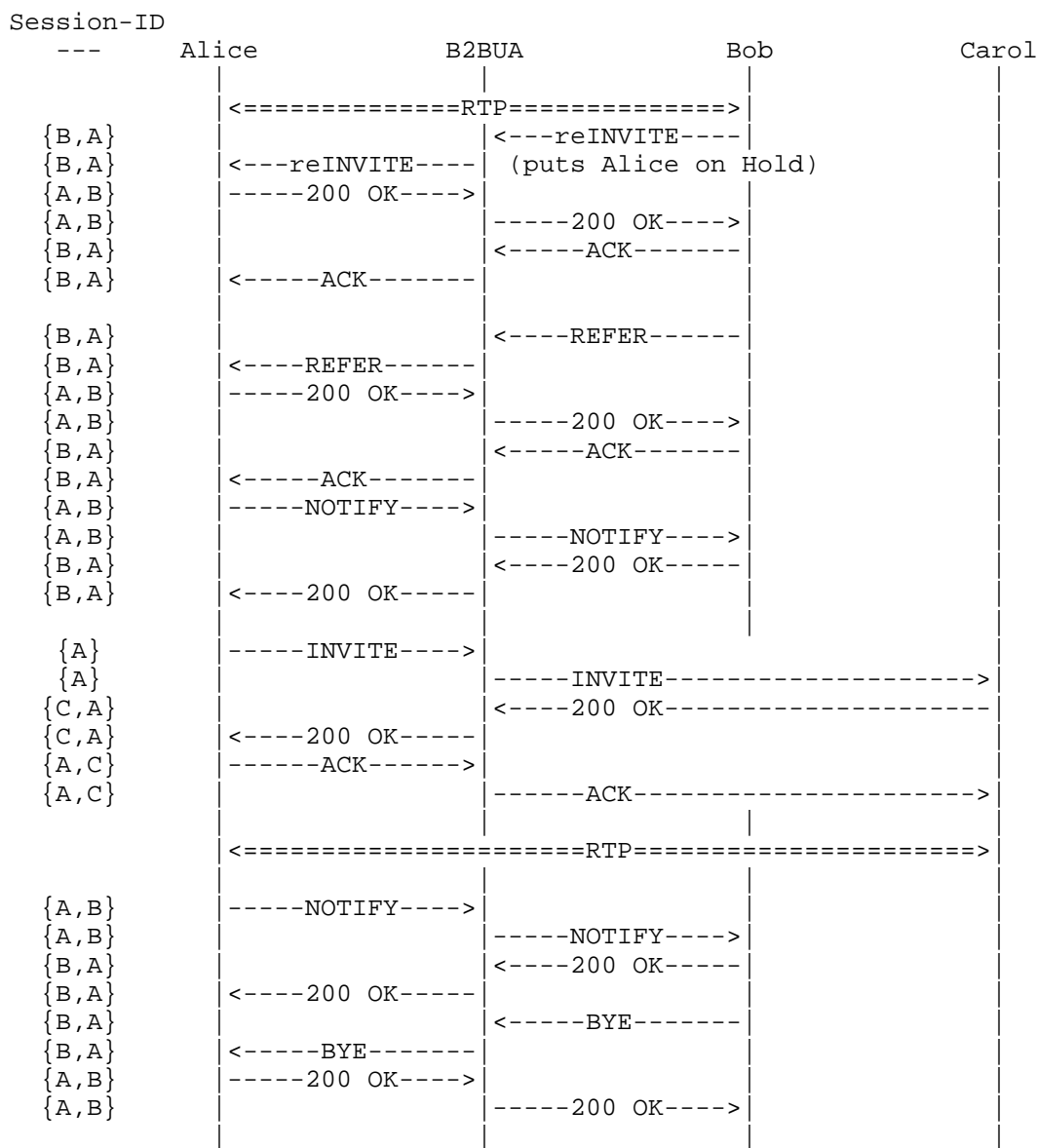


Figure 2 - Call Transfer using REFER

Operation/Rules:

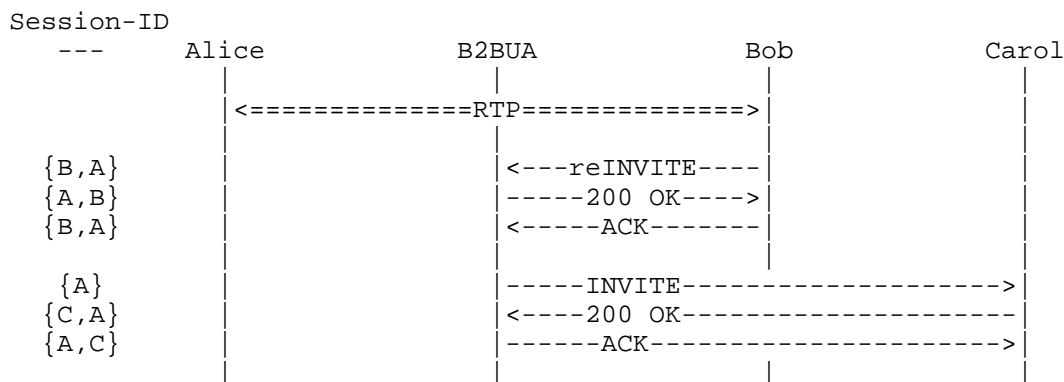
Starting from the existing Alice/Bob call described in Figure 1, which established an existing Session-ID header-value...

- o UA-Bob reINVITES Alice to call Carol, using a REFER transaction, as described in [RFC3515]. UA-Alice is initially put on hold, then told in the REFER who to contact with a new INVITE, in this case UA-Carol.
- o UA-Alice retains her UUID from the Alice-to-Bob call {A} when requesting a call with UA-Carol. This same UUID traverses the B2BUA unchanged.
- o UA-Carol receives the INVITE with a Session-ID UUID {A}, creates its own UUID {C}, and combines them to form a full Session-ID {C,A} in the 200 OK to the INVITE. This Session-ID header-value traverses the B2BUA unchanged towards UA-Alice.
- o UA-Alice receives the 200 OK with the Session-ID {C,A} and both responses to UA-Carol with an ACK, generates a NOTIFY to Bob with a Session-ID {A,B} indicating the call transfer was successful.
- o It does not matter which UA terminates the Alice-to-Bob call; Figure 2 shows UA-Bob doing this transaction.

9.3. Basic Call Transfer using reINVITE

From the example built within Section 9.1 (the basic session-ID establishment), we proceed to this 'Basic Call Transfer using reINVITE' example.

Alice is talking to Bob. Bob pushes a button on his phone to transfer Alice to Carol via the B2BUA (using reINVITE).



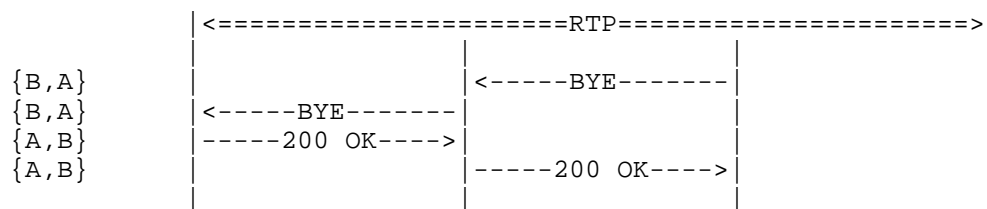


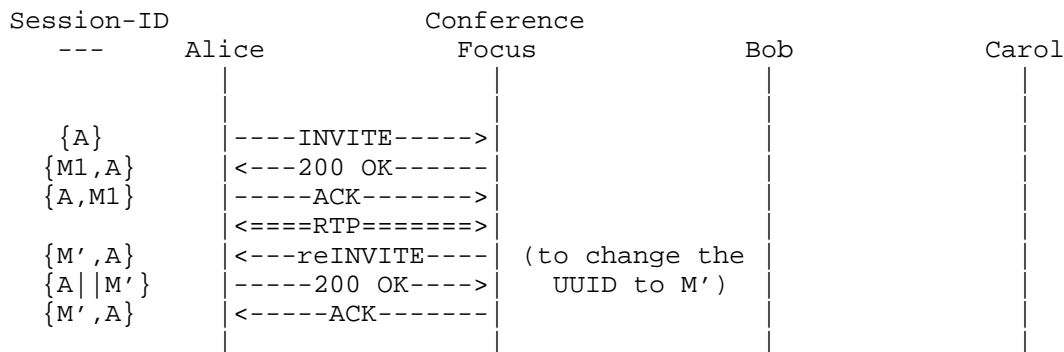
Figure 3 - Call transfer using reINVITE

Operation/Rules:

- o We assume the call between Alice and Bob from Section 9.1 is operational with Session-ID {A,B}.
- o Bob sends a reINVITE to Alice to transfer her to Carol.
- o The B2BUA intercepts this reINVITE and sends a new INVITE with Alice's UUID {A} to Carol.
- o Carol receives the INVITE and accepts the request and adds her UUID {C} to the Session-ID for this session {C,A}.
- o Bob terminates the call (which Alice could too) with a BYE using their Session-ID {B,A}.

9.4. Single Focus Conferencing

Multiple users call into a conference server (say, an MCU) to attend one of many conferences hosted on or managed by that server. Each user has to identify which conference they want to join, but this information is not necessarily in the SIP messaging. It might be done by having a dedicated address for the conference or via an IVR, as assumed in this example. Each user in this example goes through a two-step process of signaling to gain entry onto their conference call.



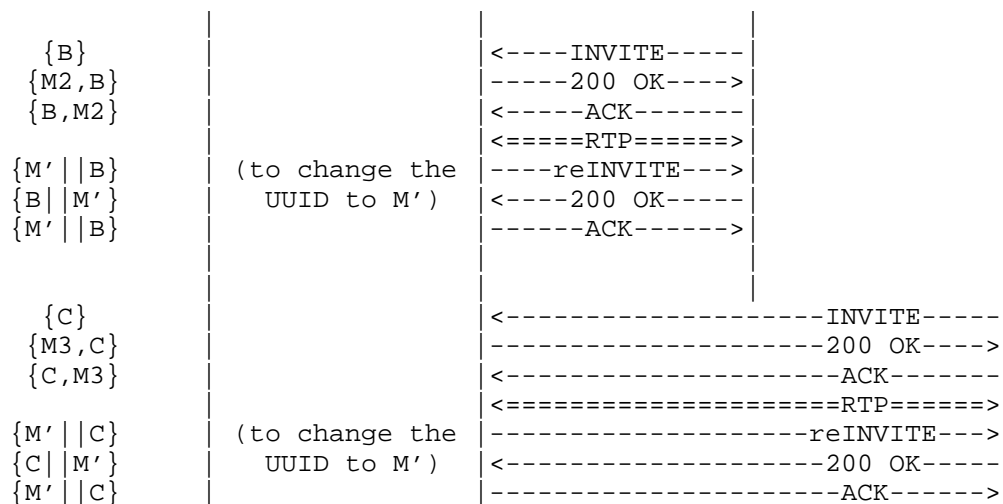


Figure 4 - Single Focus Conference Bridge

Operation/Rules:

Alice calls into a conference server to attend a certain conference. This is a two-step operation since Alice cannot include the conference ID and any passcode in the INVITE.

- o Alice sends an INVITE to the conference server with her UUID {A}.
- o The conference server accepts using a generic, temporary UUID {M1}.
- o Once Alice, the user, gains access to the IVR for this conference server, she enters a specific conference ID and whatever passcode (if needed) to enter a specific conference call.
- o Once the conference server is satisfied Alice has identified which conference she wants to attend (including any passcode verification), the conference server reINVITEs Alice to the specific conference and includes the UUID {M'} for that conference. All valid participants in the same conference will receive this same UUID for identification purposes and to better enable monitoring, and tracking functions.
- o Bob goes through this two-step process of an INVITE transaction, followed by a reINVITE transaction to get this same UUID for that conference.

- o In this example, Carol (and each additional user) goes through the same procedures and steps as Alice to get on this same conference.

9.5. Single Focus Conferencing using WebEx

Alice, Bob and Carol call into same Webex conference.

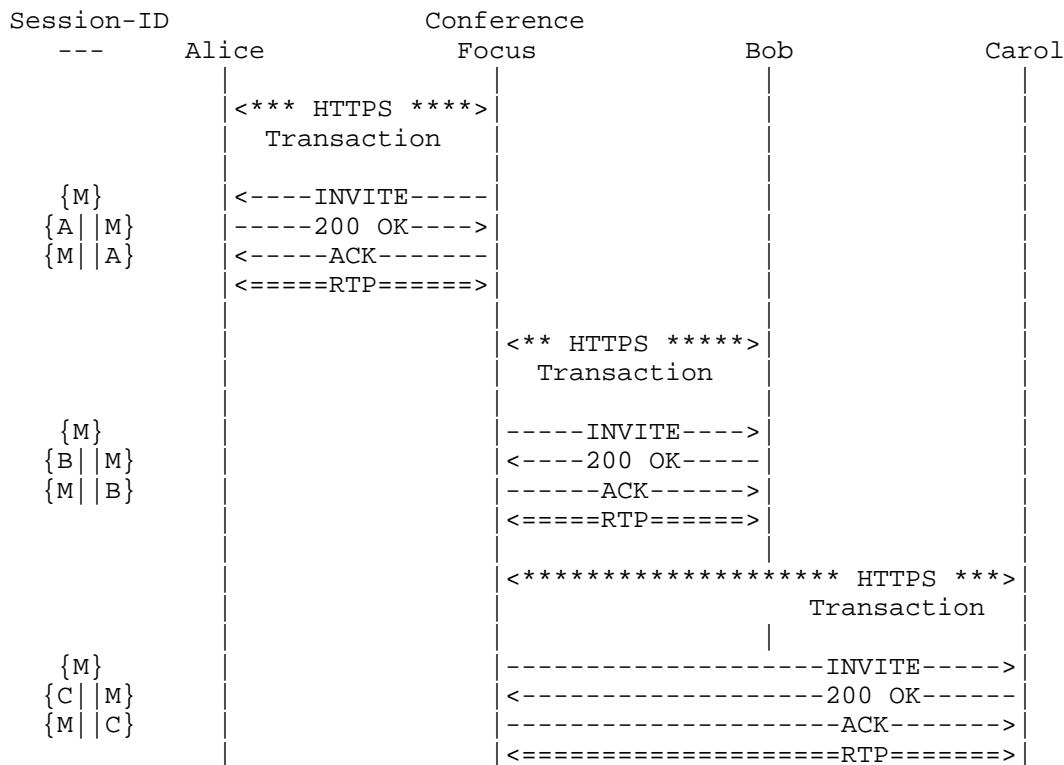


Figure 5 - Single Focus Webex Conference

Operation/Rules:

- o Alice communicates with Webex server with desire to join a certain meeting, by meeting number; also includes UA-Alice's contact information (phone number or URI).
- o Conference Focus server sends INVITE to UA-Alice to start session with the Session-ID of that server for this A/V conference call.
- o Bob and Carol perform same function to join this same A/V conference call as Alice.

9.6. Cascading Conference Bridge Support for the Session-ID

To expand conferencing capabilities requires cascading conference bridges. A conference bridge, or MCU, needs a way to identify itself when contacting another MCU. RFC 4579 [6] defines the 'isfocus' Contact: header parameter just for this purpose.

Cascading MCUs for the purpose of having each use the same UUID (aka half the Session-ID), in its simplest form, is one MCU informing another which UUID to use for joining UAs.

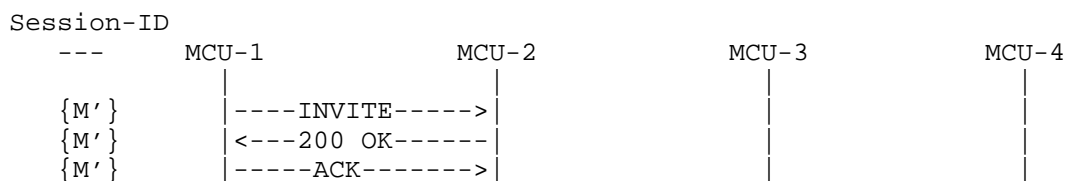


Figure 6 - MCUs Communicating Session-ID UUID for Bridge

Regardless of which MCU (1 or 2) a UA contacts for this conference, once the above exchange has been received and acknowledged, the UA will get the same M' UUID from the MCU for the complete Session-ID.

A more complex form would be a series of MCUs all being informed of the same UUID to use for a specific conference. This series of MCUs can either be informed

- o All by one MCU (that initially generates the UUID for the conference),
- o The one MCU that generates the UUID informs one or several MCUs of this common UUID, and they inform downstream MCUs of this common UUID each will be using for this one conference, or

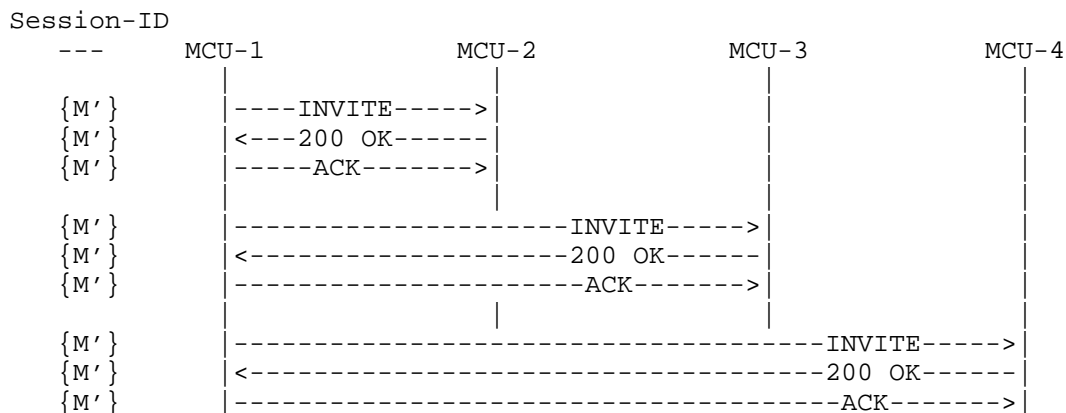


Figure 7 - MCU Communicating Session-ID UUID to More than One

Operation/Rules:

- o The MCU generating the Session-ID UUID communicates this in a separate INVITE, having a Contact header with the 'isfocus' header parameter. This will identify the MCU as what RFC 4579 conference-aware SIP entity.
- o The MCU that is contacted, i.e., the UAS MCU, does not populate or complete the Session-ID header value. The UAS MCU transmits a 200 OK response acknowledging it is to respond with this M' UUID to all requests for the designated conference.
- o An MCU that receives this M' UUID in an inter-MCU transaction, can communicate the M' UUID in a manner in which it was received (though this time this second MCU would be the UAC MCU), unless local policy dictates otherwise.

9.7. Basic 3PCC for two UAs

External entity sets up call to both Alice and Bob for them to talk to each other.

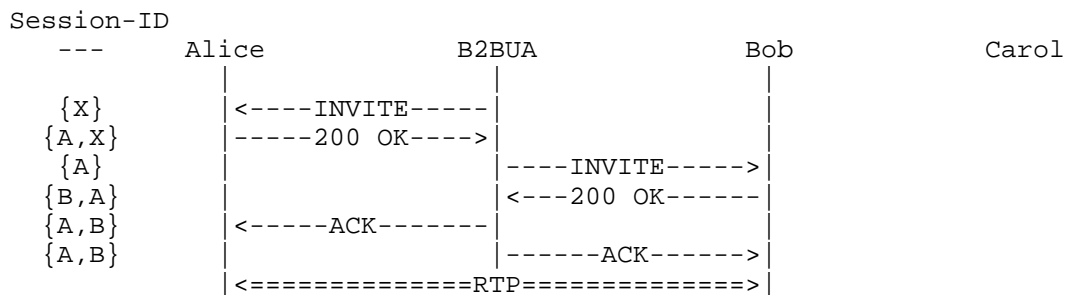


Figure 8 - 3PCC initiated call between Alice and Bob

Operation/Rules:

- o Some out of band procedure directs a B2BUA (or other SIP server) to have Alice and Bob talk to each other.
- o The SIP server INVITES Alice to a session and uses a temporary UUID {X}.
- o Alice receives and accepts this call set-up and includes her UUID {A} in the Session-ID, now {A,X}.

- o The SIP server uses Alice's UUID {A}, and discards its own {X} to INVITE Bob to the session as if this came from Alice originally.
- o Bob receives and accepts this INVITE and adds his own UUID {B} to the Session-ID, now {B,A} for the response.
- o And the session is established.

10. Compatibility with a Previous Implementation

There is a much earlier and proprietary document that specifies the use of a Session-ID header that we will herewith attempt to achieve backwards compatibility. Neither Session-ID has any versioning information, so merely adding that this document describes "version 2" is insufficient. Here are the set of rules for compatibility between the two specifications. For the purposes of this discussion, we will label the proprietary specification of the Session-ID as the "old" version and this specification as the "new" version of the Session-ID.

The previous (i.e., "old") version only has a single value as a Session-ID, but has a generic-parameter value that can be of use.

In order to have an "old" version talk to an "old" version implementation, nothing needs to be done as far as the IETF is concerned.

In order to have a "new" version talk to a "new" version implementation, both implementations need to follow this document (to the letter) and everything should be just fine.

In order to have an "old" version talk to a "new" version implementation, several aspects need to be looked at. They are:

- o The "old" version UA will include a single UUID as its Session-ID.
- o The "new" version UA will respond by including a complete Session-ID with two UUIDs, with the "new" version's UUID listed first (because it cannot know it is talking with an "old" version implementation at this point).
- o The "old" version UA will have to ignore the first UUID, and consider its singular "old" UUID as valid, as long as the value does not change..
- o During subsequent transactions within this session, the "new" version may receive SIP requests without its UUID, but with the

"old" version's UUID. The "new" version UA MUST add its UUID to the received Session-ID. The "old" version implementation will merely disregard it each time it receives this "new" version UUID (if it was not the first UUID).

In order to have a "new" version talk to an "old" Version implementation, several aspects need to be looked at. They are:

- o The "new" version UA will include a single UUID as its initial Session-ID header always, not knowing which version of UA it is communicating with.
- o The "old" version UA will respond by seeing the UUID as a valid and complete Session-ID and not include another UUID or generic-param. Thus, the 200 OK will not include any Session-ID part of its own from the "old" version implementation.

Rule: implementation supporting a "new" version of the Session-ID MUST NOT error or otherwise reject receiving only its own UUID back in any transaction. It MUST interpret this response to mean that it is communicating with an "old" Session-ID implementation.

- o Open question - how do we want all intermediaries and/or monitoring systems to interpret this single UUID complete Session-ID?

11. Security Considerations

When creating a UUID value, endpoints SHOULD ensure that there is no user or device-identifying information contained within the UUID. In some environments, though, use of a MAC address, which is one option when constructing a UUID, may be desirable, especially in some enterprise environments. When communicating over the Internet, though, the UUID value MUST utilize random values.

The Session-ID might be utilized for logging or troubleshooting, but MUST NOT be used for billing purposes. { Why does this matter? }

Other considerations???

12. IANA Considerations

The following is the registration for the 'Session-ID' header field to the "Header Name" registry at <http://www.iana.org/assignments/sip-parameters>:

RFC number: [this document]

Header name: 'Session-ID'

Compact form: none

13. Acknowledgments

The authors would like to thank Hadriel Kaplan, Christer Holmberg, and Paul Kyzivat for their invaluable comments during the development of this document.

This document was prepared using 2-Word-v2.0.template.dot.

14. References

14.1. Normative References

- [1] Rosenberg, J., et al., "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] Recommendation ITU-T H.323, "Packet-based multimedia communications systems", December 2009.
- [3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [4] Leach, P., Mealling, M., Salz, R., "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005.
- [5] Crocker, D., Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 5234, January 2008.
- [6] A. Johnston, O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", RFC 4579, August 2006

Informative References

- [6] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.
- [7] Schulzrinne, H., et al., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.
- [8] Jones, et al., "Requirements for an End-to-End Session Identification in IP-Based Multimedia Communication Networks", draft-jones-insipid-session-id-reqts-02.txt, October 2012.

Author's Addresses

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com
IM: xmpp:paulej@packetizer.com

Chris Pearce
Cisco Systems, Inc.
2300 East President George Bush Highway
Richardson, TX 75082
USA

Phone: +1 972 813 5123
Email: chrep@cisco.com
IM: xmpp:chrep@cisco.com

James Polk
Cisco Systems, Inc.
3913 Treemont Circle
Colleyville, Texas
USA

Phone: +1 817 271 3552
Email: jmpolk@cisco.com
IM: xmpp:jmpolk@cisco.com

Gonzalo Salgueiro
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 392 3266
Email: gsalguei@cisco.com
IM: xmpp:gsalguei@cisco.com

