SIP INFO Event Framework
draft-kaplan-sip-info-events-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of 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/1id-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 May 8, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

This document defines a proposed solution for defining, negotiating and exchanging info-event notifications in INFO messages, within SIP invite-created dialogs, for applications which need to exchange session-related information inside the invite-created dialog. This draft addresses issues and open items from RFC 2976, and updates it.
1. Introduction

The SIP INFO method was defined in [RFC2976] to convey session related control information inside an Invite-created dialog. While it has been widely adopted for specific application use cases, such as ISUP and DTMF exchange, the original RFC did not define a negotiation mechanism nor a means by which to explicitly indicate the type of application information contained in the INFO. This led to problems associated with static configuration, and potential interoperability problems due to undefined content syntax and semantics. This draft aims at addressing those deficiencies, to provide a framework for explicit negotiation of capabilities and content context using info-event packages. This draft is meant to update [RFC2976] SIP INFO.

The INFO method as defined in [RFC2976] does not provide any context for the information which it carries. While it may sometimes be clear what the content is, based on the Content-Type, such is only true while only one contextual usage of the content-type is ever employed. For example, if the Content-Type were "image/jpeg", it would be clear that the MIME-attached content was a JPEG image, but not what its purpose was. It could be being sent as a caller-id picture, or as a contact-icon, or whatever. The sender is not sure which JPEG to give the receiver if it supports a JPEG content-type,
and the receiver does not know which JPEG is being sent if it supports receiving more than one. Thus there needs to be a well-defined and documented statement of what the information sent is for. Event packages, as defined in [RFC3265] perform that role for Subscription-based events, whereas this document provides a similar framework for Invite-based events.

Unlike [RFC3265], the mechanism defined in this draft is not based on the usage of the SUBSCRIBE and NOTIFY methods, and does not create a separate subscription dialog or a subscription usage within an existing dialog. Instead, it uses the INVITE method and its responses to indicate and negotiate supported Event packages, and the INFO method to convey the events. This mechanism is not appropriate for all IANA-registered Subscribe Event package types, and support for this mechanism should be explicitly indicated in future Info-Event package definitions and registrations.

When the invite-created dialog is terminated the lifetime of the negotiated invite-events is also terminated.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. The terminology in this document conforms to RFC 2828, "Internet Security Glossary".

3. Applicability

This draft proposes updating the [RFC2976] SIP INFO method to include explicit negotiation of supported info-event packages in the INVITE transaction, and indication of the indicated event using a new header in the INFO request.

4. Overview of Operation

The general concept is that the UAC generating an INVITE request includes the Event packages it supports sending and receiving, in two new headers: Send-Event and Recv-Event. If the UAS supports this mechanism, it responds with the info-event packages it wishes to actually receive and send in the same named headers (in reverse), in the provisional and final responses. When either side has indication of what the far-end supports, it may send the information defined by the info-event package in an INFO request, following the same INVITE-created dialog and usage, as per [RFC2976]. The INFO request indicates the specific info-event package it is associated
with, using an Event header with the package name, and any associated MIME-attached body defined for that Event package.

5. Event Negotiation

The UAC and UAS MUST negotiate which event packages are supported and will be used in which direction, before sending an INFO message for any specific event package. Negotiation always starts with the UAC indicating which event packages it supports and wishes to send or receive, using the two new headers "Send-Event" and "Recv-Event", in the INVITE request. The UAS always indicates which events, of those offered by the UAC, it wishes to accept for sending and receiving to/from the UAC in its responses.

5.1. UAC Behavior

A UAC supporting this draft MAY indicate one or more event packages it wishes to support sending during the Invite dialog, by including their package names in the "Send-Event" header in the INVITE request. Not including the Send-Event header in the INVITE means the UAC does not have an intention, and MUST NOT, send any invite-events during the dialog.

The UAC MAY indicate one or more event packages it wishes to support receiving during the Invite-created dialog, by including their package names in the "Recv-Event" header in the INVITE request. Not including the Recv-Event header in the INVITE means the UAC will not accept info-events during the dialog.

When the UAC receives a Recv-Event header in any provisional 1xx or 2xx response, it is an indication from the far-end UAS that it (the UAS) supports receiving the indicated event indications. Any indicated Recv-Events from the UAS which were not offered by the UAC in a Send-Event MUST be ignored, and MUST NOT constitute a protocol failure.
When the UAC receives a Send-Event header in any provisional 1xx or 2xx final response, it is an indication from the far-end UAS that it (the UAS)_supports sending the indicated event indications. Any indicated Send-Events from the UAS which were not offered by the UAC in a Recv-Event MUST be ignored, and MUST NOT constitute a protocol failure.

Once an early dialog is established, through a 1xx provisional response with To-tags, and the UAC has received a Recv-Event header from the UAS in the response, the UAC MAY send any event indications supported by the UAS. The 2xx final response updates the state of the event packages, such that the 2xx contains the full and final list of Send-Event and Recv-Event packages. If the 2xx does not contain a Send-Event header, the UAS is indicating it will not send any, and if it does not contain a Recv-Event header, the UAS will not accept any, regardless of what a previous 1xx response might have indicated. At this point negotiation is considered complete.

5.2. UAS Behavior

When a UAS receives an INVITE request, it checks for a Send-Event and Recv-Event headers. One or both may exist. For each event package name in the received Send-Event header, the UAS decides if it wishes to accept receiving such events from the UAC. If so, it MUST copy the name(s) of those acceptable events into a Recv-Event header in any, and all subsequent, provisional 1xx and final 2xx responses.

For each event package name listed in the received Recv-Event header, the UAS decides if it wishes to send such events to the UAC. If so, it MUST copy the name(s) of those events it wishes to generate into a Send-Event header in any, and all subsequent, provisional 1xx and final 2xx responses.

Note that "any, and all subsequent," means that the UAS may decide not to indicate any events in early 1xx responses; but once it indicates such in a 1xx, it must continue to do so in subsequent responses including the final 2xx response in order to keep the event indication support state the same.

The UAS MUST NOT send any INFO messages with info-events until it has received an acknowledgement (e.g. PRACK for a provisional response, or an ACK for a final response it sent) that the UAC has received the SIP response sent by the UAS, indicating that the UAC has received the Send-Event header from the UAS. This assures the UAC can perform any necessary logic it needs to in order to receive the event, and can associate the INFO with the proper dialog.
NOTE: Unlike the NOTIFY method, the INFO method CAN NOT be used to establish a dialog.

6. INFO Method

The INFO method as defined in [RFC2976] is updated by this draft with a new "Event" header, which is identical to the header by the same name for SUBSCRIBE and NOTIFY methods. As in those methods, the "Event" header in an INFO request will contain a single info-event package name for which a notification is being signaled by the INFO request. The package name in the "Event" header MUST match one of the event packages received in the "Recv-Event" header in the received INVITE for the UAS, or response for the UAC. In other words one can only send what the other end is willing to receive.

Event packages may define semantics associated with the body of their INFO requests; if they do so, those semantics apply. INFO bodies are expected to provide additional details about the nature of the event which has occurred and the resultant resource state.

[NOTE: do we need to support the "id" concept as well? Is there a use case?]

As per [RFC2976], the INFO method can only be used within an INVITE-generated dialog (early or full) and usage. It has no lifetime or usage of its own, as it is inexorably linked to that of the INVITE. The dialog identifiers defined in [RFC3261] must match those of the provisional or final responses to the INVITE, and as a result INFO requests do not fork. INFO requests may be sent in either direction, once the UAC or UAS has received the Send-Event/Recv-Event indications of what the far-end supports. For the UAS, it must receive the ACK for the INVITE’s 200-ok, or the PRACK for a provisional response, before sending an INFO request. In other words it must know the far-end has received its indication of what events it is willing to send before it sends them.
7. Re-Invites and usages and forks, oh my!

A Re-INVITE or UPDATE request MUST NOT contain any Send-Event or Recv-Event headers, and such headers MUST be ignored on reception. The info-event package negotiation is performed during the initial INVITE transaction, and there is no re-negotiation.

[Is that ok? Is there a need to have re-negotiation? (is there a use case?) It sure makes it simple not to have things change in the middle of a session]

There is no separate "usage" for the INFO request, as defined by [RFC5057]. The INFO is related to but not integral to the Invite usage, such that some failure responses to an INFO request do not affect the Invite usage whatsoever, as described in [RFC5057]. Other failures may terminate the usage or dialog completely. The lifetime of info-events and thus is exactly the same as that of the Invite. If an Invite usage fails or is terminated, the Info-based event state no longer exists, and an INFO request MUST NOT be sent.

The original INVITE request may be forked along the path, resulting in multiple 1xx provisional responses, each with a separate set of send/receive info-event packages supported. It is typically up to local-policy to determine how to handle such situations, however it should be clear that an INFO request does not itself fork, since it uses the dialog identifiers and thus follows the path to a specific fork.

8. New Headers

[Should the new headers be optional for OPTIONS and REGISTER?]

<table>
<thead>
<tr>
<th>Header field</th>
<th>where proxy</th>
<th>ACK</th>
<th>BYE</th>
<th>CAN</th>
<th>INV</th>
<th>OPT</th>
<th>REG</th>
<th>PRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send-Event</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>o</td>
<td>-</td>
<td>o</td>
<td>-</td>
</tr>
<tr>
<td>Send-Event</td>
<td>2xx</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>o</td>
<td>o</td>
<td>-</td>
</tr>
<tr>
<td>Recv-Event</td>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>o</td>
<td>-</td>
</tr>
<tr>
<td>Recv-Event</td>
<td>2xx</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>-</td>
</tr>
<tr>
<td>Recv-Event</td>
<td>18x</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>o</td>
<td>o</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

8.1 "Send-Event" header

[Todo: Insert text]
8.2 "Recv-Event" header

[Todo: Insert text]

8.3 Augmented BNF Definitions

Send-Event = "Send-Event" HCOLON 1*(invite-event-package *( SEMI invite-event-param ))
Recv-Event = "Send-Event" HCOLON 1*(invite-event-package *( SEMI invite-event-param ))
invite-event-package = token-nodot
token-nodot = 1*( alphanum / "-" / "!" / "@" / "#" / "$" / "%" / "^" / "&" / "*" / "('" / ")" )
Invite-event-param = generic-param / ( "id" EQUAL token )

9. Example Exchange

In the following example, Alice initiates a call to Bob. Alice can support sending or receiving "foo" events, and sending "bar" events.

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

INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/UDP 192.0.2.1:5060;branch=z9hG4bKnashds10
From: Alice <sip:alice@example.net>;tag=1234567
To: Bob <sip:bob@example.com>
Call-Id: 123456mcmxcix
CSeq: 1 INVITE
Contact: <sip:alice@192.0.2.1>
Send-Event: foo, bar
Recv-Event: foo

Bob checks the headers, can support sending but not receiving "foo", and sending but not receiving "bar". Note that since Bob does not support receiving anything Alice can send, the response does not list any Recv-Event events.

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP 192.0.2.1:5060;branch=z9hG4bKnashds10
From: Alice <sip:alice@example.net>;tag=1234567
To: Bob <sip:bob@example.com>;tag=abcdefg
Call-Id: 123456mcmxcix
CSeq: 1 INVITE
Send-Event: foo
Since he sent the Send-Event header in the 180, Bob also sends it in the 200.

SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.0.2.1:5060;branch=z9hG4bKnashds10
From: Alice <sip:alice@example.net>;tag=1234567
To: Bob <sip:bob@example.com>;tag=abcdefg
Call-Id: 123456mcmxcix
CSeq: 1 INVITE
Contact: <sip:bob@192.0.2.2>
Send-Event: foo

Alice can send an info-event as soon as she received the 180, but in this example she would not have been able to since Bob didn’t say he could receive any Info-based events in his 180 response. Bob must wait to receive the ACK before sending any "foo" events (ACK not shown), at which point he sends the following:

INFO sip:alice@192.0.2.1 SIP/2.0
Via: SIP/2.0/UDP 192.0.2.2:5060;branch=z9hG4bKnabcdef
To: Alice <sip:alice@example.net>;tag=1234567
From: Bob <sip:bob@example.com>;tag=abcdefg
Call-Id: 123456mcmxcix
CSeq: 2 INFO
Contact: <sip:bob@192.0.2.2>
Event: foo

10. Security Considerations

There are no specific security issues for this mechanism, beyond those already applicable to SIP-based session signaling.

One interesting property of Info-based event indication is that the same digest-challenge mechanism used for Invite-based authentication can be reused for the INFO request. However this assumes the device which knows the credentials in order to perform the Invite challenge is still in the path for the INFO, or that the far-end UAS knows such credentials.
11. IANA Considerations

This document will define an info-event-type namespace, the specifics of which are TBD.

12. References

12.1. Normative References


12.2. Informative References


Author’s Address

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

Email: hkaplan@acmepacket.com

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: christer.holmberg@ericsson.com
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).
The authors wish to thank Brian Stucker, Francois Audet, Paul Kyzivat, Adam Roach, Eric Burger, Robert Sparks for their suggestions and comments; and for Dean Willis, who was ahead of his time, having submitted a similar draft 5 years ago.