SIPCORE
Internet-Draft
Obsoletes: RFC 2976
(if approved)
Intended status: Standards Track
Expires: January 6, 2010

E. Burger
NeuStar, Inc.
H. Kaplan
Acme Packet
C. Holmberg
Ericsson

July 5, 2009

Session Initiation Protocol (SIP) INFO Method and Package Framework draft-ietf-sipcore-info-events-00

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/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 January 6, 2010.

Copyright Notice

Abstract

This document provides new semantics for the SIP INFO method of RFC 2976. These new semantics defined here are fully backwards compatible with the old semantics. Core to the new semantics is a mechanism for defining, negotiating and exchanging Info Packages that use the INFO method. Applications that need to exchange session-related information within a SIP INVITE-created session, also known as application level information, use these INFO requests. This draft addresses issues and open items from RFC 2976 and replaces it.

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** [RFC2119]. The terminology in this document conforms to the **Internet Security Glossary** [RFC4949].

Be aware this document strictly follows **RFC 3261** [RFC3261] for the definition of the terms User Agent Server (UAS) and User Agent Client (UAC). Specifically, the UAC issues a SIP request and the UAS responds. This terminology may be confusing when one combines the INFO case with the INVITE case. For an INVITE, the initiator of the session is the UAC and the target of the session is the UAS. However, it is possible for the target UA of the session, the UAS of the INVITE transaction, to send an INFO to the initiating UA of the session, the UAC of the INVITE transaction. From the perspective of the INFO, the target UA of the session (INVITE UAS) is, in fact, the UAC (sender) of the INFO request. Likewise, from the perspective of the INFO, the initiating UA of the session (INVITE UAC) is the UAS (recipient) of the INFO request. Since this document strictly follows RFC 3261, we refer to the UA that issues the INVITE as the "initiating UA" and the UA that responds to the INVITE as the "target UA" to remove any confusion.

Table of Contents

TOC

1. Introduction

The **SIP protocol** [RFC3261] defines session control messages used to setup and tear down a SIP controlled session. In addition, a SIP User Agent (UA) can use the re-INVITE and UPDATE methods during a session to change the characteristics of the session. Most often, this is to change the properties of media flows related to the session or to update the **SIP session timer** [RFC4028]. The purpose of the **INFO message** [RFC2976] is to carry application level information along the SIP signaling path. Note the INFO method does not change the SIP session state. It may, however, change application state for applications using the SIP session.

While INFO has been widely adopted for specific application use cases, such as ISUP and DTMF exchange, **RFC 2976** [RFC2976] neither defined a negotiation mechanism nor a means by which to explicitly indicate the type of application information contained in the INFO message. This led to problems associated with static configuration. In addition, the industry realized there was a potential for interoperability problems due to undefined

content syntax and semantics. This draft addresses these deficiencies and provides a framework for explicit negotiation of capabilities and content context using "Info Packages".

The INFO method as defined by RFC 2976 did not provide any context for the information the request carried. While it may sometimes be clear what the content is based on the Content-Type, this is only true where there is only one contextual usage of the content-type. For example, if the Content-Type is "image/jpeg", the MIME-attached content is a JPEG image. However, there are many useful ways a UAS can render an image. Said differently, there are different contexts for an image in SIP. The image could be a caller-id picture, a contact icon, a photo for sharing, and so on. The sender does not know which image to send to the receiver if the receiver supports an image content type. Likewise, the receiver does not know the context of an image the client is sending if the receiver supports receiving more than one image content type. Thus, we need a well defined and documented statement of what the information sent is for. This situation is identical to the **context issue in Internet Mail** [RFC3458]. RFC 3458 goes into this and other issues in detail.

Event Packages [RFC3265] perform the role of disambiguating the context of a message for subscription-based events. This document provides a similar framework for INVITE-based application level information exchange. However, while the mechanism described here is similar to subscription-based events, there is no formal relationship between this mechanism and the subscription mechanism. In particular, when a UAC issues a SUBSCRIBE, it creates a dialog usage.

The mechanism defined here creates neither a separate subscription dialog nor a subscription usage within an existing session. Instead, it uses the INVITE method and its responses to indicate supported Info Packages and the INFO method to convey the Info Packages.

Each UA enumerates which Info Packages it can receive. If a first UA indicates it can receive a package and a second UA can send the package, the second UA can send INFO methods containing the payload for that package. The Recv-Info header indicates which packages a UA is willing to receive. The Info-Package header indicates which package a particular INFO method request belongs to. There is a reserved Info Package, "nil", which indicates the UA conforms to this document, but does not wish to receive Info Packages. This enables other UAs that conform to this document to detect legacy UAs. A legacy UA will not include a Recv-Info header in their SIP session establishment or modification requests. Conversely, a UA that supports Info Packages will have a Recv-Info header. **Section 3** describes Info Package advertisement in detail.

This document does not describe any specific Info Package type extensions. One must extend this protocol by other documents, herein referred to as "Info Packages". **Section 7** describes guidelines for creating these extensions.

The INFO method does not change the state of SIP calls or the parameters of the sessions SIP initiates. It merely sends optional application layer information, generally related to the session.

Applications need to be aware that application level information transported by the INFO method constitutes mid-session signaling. These messages traverse the post-session-setup SIP signaling path. This is the path taken by SIP re-INVITEs, BYEs, and other SIP requests within an individual session. SIP proxy servers will receive, and potentially act on, mid-session signaling information. Application designers need to understand this can be a feature, as when the User Agents are exchanging information that elements in the SIP signaling path need to be aware of. Conversely, this can be a problem, as messages these network elements have no interest in can also put a significant burden on those element's ability to process other traffic. Moreover, such network elements may not be able to read end-to-end encrypted INFO bodies.

2. Applicability

TOC

This document replaces the **SIP INFO method document** [RFC2976] to include explicit negotiation of supported Info Packages in the INVITE transaction and indication of the Info Package to use by using a new header field in the INFO request. As described in **Section 4.1**, the mechanism described here is backwards compatible with legacy, RFC 2976 INFO mechanisms.

3. Info Package Behavior

TOC

As stated in the Conventions section, the term UAC refers to the UAC (sender) of the INFO method and UAS refers to the recipient of the INFO method. "Initiating UA" refers to the sender of an initial INVITE to establish a session and "target UA" refers to the recipient of that INVITE request.

3.1. UAS Behavior

TOC

A UAS supporting this document MUST advertise the set of Info Packages it is willing to receive in Recv-Info header(s) in dialog usage requests and responses for session establishment or target refresh. This includes INVITE, UPDATE, PRACK, ACK, and their non-failure responses (101-199 and 2xx only).

Once a UAS indicates support for an Info Package by sending a Recv-Info header with one or more package names, the UAS MUST be prepared to receive an INFO containing that package. Note this may occur before dialog negotiation completes.

Recall the UAC of an INVITE may choose to receive (be a UAS for) INFO methods. This UA may chose not to offer any packages in the initial INVITE and subsequently advertise packages from the target UA's subsequent responses, in order to support **third-party call control** [RFC3725].

A UAS lists multiple packages by enumerating the package name(s), separated by commas, as values for the Recv-Info header in the session establishment exchange. A UAS may also list multiple packages by including multiple Recv-Info headers. The UAS may also combine multiple Recv-Info headers with one or more packages in each header value. If the UAS prefers to receive one package over another, the UAS MUST list the preferred Info Package lexically earlier in the message. That is, by listing it earlier in a list within a given Recv-Info header or listing it in a previous Recv-Info header in a given message. The UAS MUST NOT list a package more than once. This order is only a hint to the UAC, as there is no meaningful way of enforcing the use of a preferred package at the UAC.

There is an important issue to consider when the UAS advertises support for multiple packages that one might interpret to be similar or equivalent. The UAC has no method of knowing whether the UAS would like the UAC to send a single INFO request with the preferred package or for the UAC to send multiple INFO requests with the same or similar information. The behavior is entirely up to the UAC and the rules specified by the package definitions.

If a UAS does not wish to receive any Info Packages, the UAS MUST indicate this by including one and only one Recv-Info header with the value 'nil'. This enables the UAC to

discern the difference between a UAS that understands Info Packages but does not wish to receive any from a legacy UAS that does not understand Info Packages. A UAC conforming to this document can always send or receive legacy INFO usages without packages.

Info Package capability advertisement occurs within the context of a session negotiation exchange. The Info Package capability set received by the UAC within the last exchange is the one the UAC will use to chose Info Packages from. Also note that due to glare, an INFO request may be in flight prior to the UAC receiving an updated capability set removing a given Info Package. Thus, the UAS MUST be prepared to handle an INFO request with an Info Package payload with a newly delisted Info Package. Proper handling does include rejecting the request with a 469. See **Section 4.3** for more on this topic.

TOC

3.2. UAC Behavior

A UAC MUST NOT send INFO requests for a given INFO package until the UAC receives an "INVITE dialog usage" request or response (for session establishment or target refresh) with a Recv-Info header listing the given Info Package.

At any time during an "INVITE dialog usage" request or response, if a UAS sends one or more Recv-Info headers, the UAC MUST replace the old set of supported Info Packages with the collection of Info Packages enumerated by the current message.

If the UAS does not send any Recv-Info headers in a message, then the list of supported Info Packages does not change.

A UAC MUST cease sending INFO requests for a given INFO package when the UAC receives an "INVITE dialog usage" request or response (for session establishment or target refresh) that does not contain a Recv-Info header listing the given Info Package. Note the UAC MUST be prepared to receive a **469 response** at any time, even if the UAS advertised it could receive the Info Package. This situation can occur if the UAC sends the INFO request at the same time the UAS advertises it no longer supports the Info Package in question.

If the UAC receives a Recv-Info header with the value 'nil', the UAC MUST NOT send any INFO methods that contain Info Packages.

The UAS may advertise support for multiple Info Packages. If some of these packages have similar or equivalent functionality and the UAC supports multiple such packages, the UAC SHOULD chose to send Info Package payload(s) from the Info Package listed lexically earlier in the last Recv-Info advertisement the UAC received from the UAS. This document cannot make this protocol action a must strength, as the concept of "similar or equivalent" is highly Info Package specific.

INFO itself does not necessitate the use of Require or Proxy-Require headers. There is no token defined for "Supported" headers. If necessary, clients may probe for the support of this version of INFO using the OPTIONS request defined in **SIP** [RFC3261]. One could envision a particular Info Package implementation that relied on either of these headers. See **Section 7** for more on this issue.

The presence of the Recv-Info header in a message is sufficient to indicate support for this version of INFO. The "methods" parameter for **Contact** [RFC3841] is not sufficient to determine if the endpoints support Info Packages, as the INFO method supported might be the obsolete **RFC 2976** [RFC2976] version.

For Info Packages, this draft does not provide a means of requiring support for a specific Info Package. If the UAS does not indicate support for an Info Package that the UAC

requires, and the UAC requires the use of that package, the UAC can use any supported **RFC3261** [RFC3261] method to terminate the session.

A UAC MAY send a legacy **INFO** [RFC2976] method at any time.

TOC

3.3. Package Versions

The protocol mechanism described herein does not provide for a package versioning mechanism. This is for two reasons. The first is that if an Info Package has a capability for forward and backward compatibility in the Info Package payload, then that compatibility comes from the application level semantics of the information. This means it is the responsibility of the application to handle such compatibility and not the INFO framework. For example, one could use XML versioning techniques in the payload to indicate versions of the Info Package.

The second reason we do not have a package versioning system is not all payloads have sufficient capability to carry payload versions. In this situation, it is highly unlikely payloads will be backwards compatible. That is, what one really is defining is a new Info Package. This is more especially so when one considers User Agents can advertise package support but cannot advertise package version support. Even if we did allow for package versioning, as a parameter to the Recv-Info header value, for example, it is lexically equivalent to having a new Info Package.

UACs MUST NOT depend on any lexical parsing of the Info Package name for versioning, such as "fooV1" and "fooV2" or "foo.1" and "foo.2".

TOC

3.4. Advertisement Example

Here is an INVITE. The initiating UA advertises the following.

```
INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/TCP pc33.example.com;branch=z9hG4bK776
Max-Forwards: 70
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.example.com
CSeq: 314159 INVITE
Recv-Info: P, R
Contact: <sip:alice@pc33.example.com>
Content-Type: application/sdp
Content-Length: ...

...
```

This means the initiating UA is willing to receive from Info Packages P and R.

In this next message, the target UA responds with a 200 OK:

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP pc33.example.com;branch=z9hG4bK776;received=192.0.2.1
To: Bob <sip:bob@example.com>;tag=a6c85cf
From: Alice <sip:alice@example.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.example.com
```

```
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.example.com>
Recv-Info: R, T
Content-Type: application/sdp
Content-Length: ...

...
```

This indicates the target UA is willing to receive from Info Packages R and T.

The initiating UA then confirms in an ACK, as shown.

```
ACK sip:ngw1@a.example.com SIP/2.0
Via: SIP/2.0/TCP pc33.example.com;branch=z9hG4bK776
To: Bob <sip:bob@example.com>;tag=a6c85cf
From: Alice <sip:alice@example.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.example.com
CSeq: 314163 ACK
Recv-Info: R
Content-Length: 0
```

The target UA can now send from package R to the initiating UA. Moreover, in this example, the target UA may not send from package P, as P no longer is in the initiating UA's Info Package set.

---

4. The INFO Method Request

TOC

4.1. INFO Requests

TOC

The INFO method provides additional, application level information that can further enhance a SIP application. It is important to note there are some types of application information for which using INFO messages are inappropriate. See **Appendix A** for a discussion of this.

The UAC MUST include the Info-Package header field when it sends an INFO request carrying an Info Package. The Info-Package header field value in an INFO request MUST contain a single Info Package token. That Info Package token MUST match one of the Info Packages the UAS indicated support for during the negotiation described in **Section 3**.

The UAC MAY send an INFO in a legacy usage context. See **Appendix B** for examples of legacy usages. In general, a legacy usage is where there is no Info-Package header. In this case, if the UAS has never offered a Recv-Info header or never offered a Recv-Info header with a package of a similar function to the legacy INFO usage, the UAC MAY send an INFO without an Info-Package header field and a body appropriate to the said legacy usage.

A UAC MUST NOT use the INFO method outside an INVITE dialog usage. The INFO method has no lifetime or usage of its own, as it is inexorably linked to that of the INVITE. When the INVITE-created session terminates, that signals the termination of the negotiated Info Packages. A UAS that receives an INFO message after the INVITE dialog usage terminates MUST respond with a 481 Call Does Not Exist.

The session identifiers defined in **RFC 3261** [RFC3261] must match those of the provisional or final responses to the INVITE. As a result, INFO requests cannot fork. The UAC may

send INFO requests once the UAS has sent the Recv-Info header field value, indicating what the UAS supports.

The converse is not true during initial session establishment. The initiating UA of the first INVITE MUST be prepared to receive multiple INFO requests, as the first INVITE may fork. Since session negotiation has not completed, and we allow early INFO requests, multiple target UAs may respond. This initial session establishment phase is the only time the UAS need be prepared to receive multiple INFO requests, as one would expect there may be messages from non-authoritative forked dialogs prior to their termination.

The construction of the INFO request is the same as any other request within an existing INVITE-initiated session. A UAC MAY send an INFO request on both an early and confirmed session.

The INFO request MUST NOT carry a Recv-Info header. The UAC can only negotiate Info Packages using the procedures of **Section 3**.

The signaling path for the INFO method is the signaling path established as a result of the session setup. This can be direct signaling between the calling and called user agents or a signaling path involving SIP proxy servers that were involved in the call setup and added themselves to the Record-Route header on the initial INVITE message.

TOC

4.2. INFO Request Body

The purpose of the INFO request is to carry application level information between SIP user agents. The INFO message body SHOULD carry this information, unless the message headers carry the information of interest. Note this is not an invitation to invent SIP headers for the purposes of application level information exchange. Rather, one could envision circumstances where existing SIP headers already convey the information the application has interest in.

If the Info Package defines a payload, and the package specification indicates it is appropriate to include a payload with the request, the UAC MUST include the payload with the MIME type specified by the Info Package.

If the Info Package definition directs the UAC to send a request without a payload, the UAC MUST send the INFO request without a body.

Some SIP extensions, which are orthogonal to INFO, may insert body parts unrelated to the INFO payload. User Agents MUST conform to RFC 3261 as updated by **body-handling** [I-D.ietf-sip-body-handling] to support multipart MIME handling. If there are bodies unrelated to the Info Package, and the Info Package also has a payload, the UAC MUST bundle these elements into a multipart MIME body. In this case, the UAS needs a means to unambiguously identify the body part belonging to the Info Package. To do this, the UAC MUST identify the Info Package payload MIME body part with a Content-Disposition of 'Info-Package'.

If the payload of an Info Package is already a multipart MIME body, the UAC MUST identify the payload with a Content-Disposition of 'Info-Package' in the headers for the appropriate MIME body part.

If there is no payload in the INFO request unrelated to the Info Package and the payload of the Info Package is not a multipart MIME, the UAC MUST identify the message, at the SIP header level, with a Content-Disposition of 'Info-Package'.

If there is no payload for the Info Package, they UAC MAY omit the Content-Disposition

header.

> NOTE: We could be lazy and even save 33 octets by allowing the UAC to construct a non-multipart MIME payload without a Content-Disposition header. However, mandating the presence makes parsing considerably easier, and it is easier to have it required now than run into a problem later.
>
> NOTE: One could offer that the Info-Package header is redundant, as we could have the Info Package name be a parameter for Content-Disposition. However, there could be corner cases with legacy INFO usage that makes this a poor choice.

TOC

4.3. Responses to the INFO Request Method

If a UAS receives an INFO request, it MUST send a final response. A UAS MUST send a 200 OK response for an INFO request with no message body and no Info-Package header if the UAS received the INFO request on an existing session. This protocol action supports legacy use of INFO as a keep-alive mechanism.

If the UAS receives an INFO request with an Info-Package the UAS advertised with a Recv-Info in the last session state update and the body of the INFO request is an appropriate MIME type for the Info Package, the UAS MUST send a 200 OK response.

If the INFO request contains a body the server does not understand then, in the absence of Info Package associated processing rules for the body, including the absence of an Info-Package header, the server MUST respond with a 415 Unsupported Media Type message.

If the INFO request indicates an Info Package type the server does not understand, then the server MUST respond with a 469 Bad INFO Package. In the terminology of **Multiple Dialog Usages** [RFC5057], this represents a Transaction Only failure.

If a server receives an INFO request with a body it understands, but the request has no Info-Package header, the UAS MAY use the body as it sees fit. If the UAS accepts the INFO request, the UAS MUST respond to the INFO request with a 200 OK. This enables legacy use of INFO. If the UAS needs to enforce strict compliance with the current INFO framework described here, the UAS MUST reject the request with a 469.

The UAS MUST send a 481 Call Leg/Transaction Does Not Exist message if the INFO request does not match any existing INVITE-initiated session.

The UAS MAY send other responses, such as Request Failure (4xx), Server Failure (5xx) and Global Failure (6xx) as appropriate for the request.

TOC

4.4. Routing Behavior

Unless stated otherwise, the protocol rules for the INFO request governing the usage of tags, Route and Record-Route, retransmission and reliability, CSeq incrementing and message formatting follow those in **RFC 3261** [RFC3261] as defined for the BYE request.

The INFO message MUST NOT change the state of the SIP session. Of course, outside the INFO machinery specific failure responses as documented in **the SIP dialog usages document** [RFC5057], may cause the INVITE session to terminate.

TOC

4.5. Behavior of Registrars

Registrars receiving a REGISTER request that includes Recv-Info headers MAY store such information and use it for routing purposes. How the registrar uses this information is beyond the scope of this document.

TOC

4.6. OPTIONS Processing

A UAC, the sender of the OPTIONS request, SHOULD include Recv-Info headers, populated appropriately for the packages the UAC supports. The UAS SHOULD include its set of Recv-Info packages. These strictures are of "should" strength because local policy might restrict the advertisement of full capabilities, the UA may know the best choice of equivalent packages to list from local configuration, and so on.

The UAS and UAC MUST NOT consider the OPTIONS request to be part of a capabilities negotiation. The OPTIONS request is purely a probe. For the UAC or UAS to renegotiate package support, they must use the procedures described in **Section 3**.

TOC

4.7. Order of Delivery

The INFO method does not define mechanisms for ensuring in-order delivery for overlapping INFO requests. That is, the UAC can send another INFO request before receiving a transaction response from the UAS to a prior INFO request. While the UAC will increment the CSeq header upon the transmission of new INFO messages, the UAS cannot use the CSeq to determine the sequence of INFO information. All a UAS can determine is the UAC sent one INFO message after another. This is due to the fact that there could be gaps in the INFO message CSeq count caused by a user agent sending re-INVITES or other SIP messages.

It is up to the individual Info Package definition to specify what happens when there are overlapping INFO requests. However, since it is legal SIP to have overlapping requests, the application must be able to handle the reception of overlapping requests. Overlapping requests can occur even if the particular instance of an application (Info Package) does not allow it, as the mechanism described here is package-agnostic. Thus, the Info Package needs to define the appropriate response. This is more especially so given the UAC could send from multiple Info Packages. Some of those packages may allow overlapping INFO requests, while others do not. In this situation, it would be hard to tell if the non-overlapping packages were being violated or not.

TOC

5. Formal INFO Method Definition

TOC

5.1. INFO Method

This document describes one new SIP method: INFO. This document replaces the definition and registrations found in **[RFC2976]**.

This table expands on Tables 2 and 3 in **[RFC3261]**.

Header	Where	INFO	
Accept	R	o	
Accept-Encoding	R	o	
Accept-Encoding	2xx	o	
Accept-Encoding	415	c	
Accept-Language	R	o	
Accept-Language	2xx	o	
Accept-Language	415	c	
Alert-Info		-	
Allow	R	o	
Allow	200	-	
Allow	405	o	
Authentication-Info	2xx	o	
Authorization	R	o	
Call-ID	c	m	
Call-Info		o	
Contact		-	
Content-Disposition		o	
Content-Encoding		o	
Content-Language		o	
Content-Length		o	
Content-Type		*	
CSeq	c	m	
Date		o	
Error-Info	3xx-6xx	o	
Expires		-	
From	c	m	
Geolocation	R	o	
Max-Breadth	R	-	
Max-Forwards	R	o	
MIME-Version		o	
Min-Expires		-	
Organization		o	
Priority	R	-	
Privacy	R	o	
Proxy-Authenticate	407	o	
Proxy-Authorization	R	o	
Proxy-Require	R	o	
Reason	r	o	
Record-Route	R	o	
Record-Route	2xx,18x	o	
Require		o	
Retry-After	R	-	
Retry-After	404,480,486	o	
Retry-After	503	o	
Retry-After	600,603	o	
Route	R	o	
Security-Client	R	o	
Security-Server	421,494	o	
Security-Verify	R	o	
Server	r	o	
Subject	R	o	
Supported	R	o	
Supported	2xx	o	
Timestamp		o	
To	c	m	(w/ Tag)
Unsupported	420	o	
User-Agent		o	
Via		m	
Warning	r	o	

WWW-Authenticate	401	m
WWW-Authenticate	407	o

Figure 1: Table 1: Summary of Header Fields

TOC

5.2. INFO Headers

This table expands on tables 2 and 3 in **[RFC3261]**.

Header field	where	ACK	BYE	CAN	INV	OPT	REG	PRA	INF	MSG	UPD	SUB	NOT	RFR
Info-Package	R	-	-	-	-	-	-	-	o*	-	-	-	-	-
Recv-Info	R	o	-	-	o	o	o	o	-	-	o	-	-	-
Recv-Info	2xx	o	-	-	o	o	-	o	-	-	o	-	-	-
Recv-Info	1xx	o	-	-	o	o	-	o	-	-	o	-	-	-
Recv-Info	r	o	-	-	-	o	-	o	-	-	o	-	-	-

* Info-Package is MANDATORY for INFO messages sent using Info Packages as described in this document. Info-Package is OPTIONAL for legacy (RFC2976) INFO messages.

Table 2: INFO-related Header Fields

TOC

5.2.1. Info-Package header

This document adds Info-Package to the definition of the element "message-header" in the SIP message grammar.

For the purposes of matching Info Package types indicated in Recv-Info with those in the Info-Package header field value, one compares the Info-package-name portion of the Info-package-type portion of the Info-Package header octet-by-octet with that of the Recv-Info header value. That is, the Info Package name is case sensitive. Info-package-param is not part of the comparison-checking algorithm.

This document does not define values for Info-Package types. Individual Info Packages define these values. Such documents MUST register such values with IANA. These values are **Specification Required** [RFC5226].

TOC

5.2.2. Recv-Info header

This document adds Recv-Info to the definition of the element "general-header" in the **SIP** [RFC3261] message grammar. **Section 3** describes the Recv-Info header usage.

TOC

6. Legacy Uses of INFO

Several RFC-defined and other standards-defined uses of **RFC 2976 INFO** [RFC2976] exist and are in use, as well as numerous proprietary uses. **Appendix B** describes some of these usages. By definition, identifying such uses has relied on either static local configuration or implicit context determination based on the body Content-Type or Content-Disposition value or some proprietary mechanism. This draft cannot forbid nor avoid such uses, since local configuration can always override standardized mechanisms.

To maintain backward compatibility with the extant standardized uses of INFO, a server MAY interpret an INFO request with no "Info-Package" header as being of such legacy use.

It should be noted that such legacy use will not "break" the mechanism in this draft. For example, if a UA supports **SIP-T** [RFC3372], it does so based on static local configuration or based on acceptance of the application/isup body. If it adds support for this draft's Info Package negotiation mechanism, the local configuration still applies, and the UA will send/receive INFO messages based on SIP-T regardless of the Info Package negotiation. It will also be able to send/receive INFO messages based on the Info Packages it negotiated. If, at some future time, an Info Package is defined for SIP-T, the UA can indicate such in the negotiation, and again local configuration would supersede if need be. The UA would not lose the ability to use SIP-T with legacy devices. Rather, it would gain the ability to use it with devices which support this draft and with which it did not have such local configuration set, and could avoid failures related to unsupported bodies.

It is the hope of this draft's authors that vendors that implement proprietary INFO uses submit their mechanisms as Info Package extension documents, and follow the Info Package negotiation mechanism defined in this draft.

TOC

7. Info Package Requirements

Info Packages SHOULD NOT reiterate any of the behavior described in this document, unless required for clarity or emphasis. However, such packages MUST describe the behavior that extends or modifies the behavior described in this document.

Info Packages MUST NOT weaken any behavior designated with "SHOULD" or "MUST" in this document. However, Info Packages MAY strengthen "SHOULD", "MAY", or "RECOMMENDED" requirements to "MUST" strength if the application requires it.

In addition to the normal sections expected in standards-track RFCs and SIP extension documents, authors of Info Packages need to address each of the issues detailed in the following subsections, whenever applicable.

TOC

7.1. Applicability

This section, which MUST be present, describes why any of the other established user-to-user data transfer protocols are not appropriate for the given Info Package. Common reasons can be a requirement for SIP Proxies or back-to-back User Agents (B2BUAs) to see the application level information. Consideration in this section MUST describe what happens if one or both endpoints encrypt the payload.

TOC

7.2. Info Package Name

This section, which MUST be present, defines the token name that designates the Info

Package. The name MUST conform to the token ABNF production described in **Section 8**. It MUST include the information that appears in the IANA registration of the token. For information on registering such types, see **Section 9**.

TOC

7.3. Info Package Parameters

If the "Info-Package" header allows parameters to modify the behavior of the Info Package, this section MUST clearly define the syntax and semantics of such parameters.

TOC

7.4. Info Package Tags

If useful for the Info Package to have SIP option tags, this is the place to define the tag. Note that if the Info Package defines a SIP option tag, the Info Package must conform to the **SIP Change Process** [RFC3427].

TOC

7.5. INFO Bodies

Each Info Package MUST define what type or types of bodies are expected in INFO requests. Such packages MUST specify or cite detailed specifications for the syntax and semantics associated with such a body.

The UAS MUST enumerate every MIME type associated with the Info Packages advertised in the UAS' Recv-Info header the UAS is willing to receive. If a UAC sends a body that includes something not enumerated by the UAS, this is a protocol error and the UAS MUST respond appropriately.

TOC

7.6. UAC generation of INFO requests

Each Info Package MUST describe the process by which a UA generates and sends an INFO request. This includes detailed information about what events cause the UA to send an INFO request.

If the Info Package does not allow overlapping (outstanding) INFO requests, the Info Package MUST disclose this in the section describing UA generation of INFO requests. Note the generic protocol machinery of the INFO method has no way of enforcing such a requirement. **Section 7.7** describes this situation.

TOC

7.7. UAS processing of INFO requests

The Info Package MAY describe the process followed by the UA upon receipt of an INFO request. Since INFO does not change SIP state, and may not even change application state, there may be no useful guidance required in the Info Package specification for UA processing.

If the info Package does not permit overlapping INFO requests, it is important to note the issuance of overlapping INFO requests is an application-layer protocol failure and not an

INFO method failure. Therefore, in the event a UAC issues overlapping INFO requests for an Info Package, the UAS MUST NOT return an error response as a result of the overlapping INFO request. Of course, if there are other problems with the request that results in a failure, the UAC issues the appropriate response code. This section of the Info Package specification MUST describe the application level response to overlapping INFO requests. Examples include a new INFO request back to the offending UAC indicating an application error, ignoring the overlapping request and processing it to the UAS' best effort, or terminating the entire SIP session.

TOC

7.8. Rate of INFO Requests

Each Info Package MUST define a requirement of MUST strength which defines an absolute maximum on the rate at which an Info Package of a given type can generate INFO messages by a UA in a session.

If possible, a package MUST define a throttle mechanism that allows UAs to further limit the rate of INFO messages.

TOC

7.9. IANA Registrations

The Info Package MUST have an IANA Considerations section that includes definitions for the Info Package Name and, if needed, supported MIME types.

TOC

7.10. Security Considerations

The INFO mechanism transports application level information. One implication of this is INFO messages may require a higher level of protection than the underlying SIP-based session signaling. If the application transports sensitive information, such as credit card numbers, health history, personal identifiers, and so on, the Info Package MUST document security procedures that exceed the default procedures presented in this document. In most circumstances, it is not sufficient for a package to attempt to mandate TLS for the signaling channel to secure the data carried by the INFO. Intermediaries will have access to the payload and past the first hop, there is no way to assure subsequent hops will not transmit the payload in clear text. The only way to ensure secure transport at the application level is to have the security at the application level. The most common method of achieving this is to use end-to-end security techniques such as **S/MIME** [RFC3851]. If the application demands this level of security, the Info Package definition MUST indicate such.

TOC

7.11. Examples

We RECOMMEND Info Packages include several demonstrative message flow diagrams paired with several typical, syntactically correct, and complete messages.

Documents describing Info Packages MUST clearly indicate the examples are informative and not normative, with instructions that implementers refer to the main text of the document for exact protocol details.

8. Syntax

TOC

This section describes the syntax extensions required for the INFO method. The previous sections describe the semantics. Note the formal syntax definitions described in this document use the ABNF format used in **SIP** [RFC3261] and contain references to elements defined therein.

```
INFOm                = %x49.4E.46.4F ; INFO in caps
extension-method     = INFOm / token

Info-Package         =  "Info-Package" HCOLON Info-package-type
Recv-Info            =  "Recv-Info" HCOLON Info-package-list
Info-package-list    =  "nil"
                     / Info-package-type *( COMMA Info-package-type )
Info-package-type    =  Info-package-name *( ";" Info-package-param)
Info-package-name    =  token
Info-package-param   =  token
```

NOTE on the Recv-Info production: if the value is "nil", there can be one and only one Recv-Info header in the SIP message.

9. IANA Considerations

TOC

9.1. Update to Registration of SIP INFO Method

TOC

Please update the existing registration in the SIP Methods and Response Codes registry under the SIP Parameters registry that states:

```
Method:      INFO
Reference:   [RFC2976]
```

to:

```
Method:      INFO
Reference:   [RFCXXXX]
```

9.2. Registration of the Info-Package Header Field

TOC

Please add the following new SIP header field in the Header Fields subregistry under the SIP Parameters registry.

```
Header Name:   Info-Package
Compact Form:  (none)
```

```
Reference:      [RFCXXXX]
```

TOC

9.3. Registration of the Recv-Info Header Field

Please add the following new SIP header field in the Header Fields subregistry under the SIP Parameters registry.

```
Header Name:    Recv-Info
Compact Form:   (none)
Reference:      [RFCXXXX]
```

TOC

9.4. Creation of the Info Packages Registry

Please create a subregistry in the SIP Parameters registry for Info Packages. This subregistry has a modified **First Come First Served** [RFC5226] policy.

The following data elements populate the Info Package Registry.

- Info Package Name: The Info Package Name is a case-sensitive token. In addition, IANA shall not register multiple Info Package names that have identical case-insensitive values.
- Info Package Payload MIME Types: A list of zero or more registered MIME types from the MIME Type Registry.
- Standards Status: Values are "Standards Track" or empty. See below for a discussion and rules on this field.
- Reference: If there is a published specification describing the Info Package, place a reference to that specification in this column. See below for a discussion on this field.

If there is a published specification, the registration MUST include a reference to such specification. The Standards Status field is an indicator of the level of community review for the Info Package specification. If the specification meets the requirements for **Specification Required** [RFC5226], the value for the Standards Status field is "Standards Track". Otherwise, the field is empty.

This document uses the Info Package Name "nil" to represent "no Info Package present" and as such, IANA shall not honor a request to register the "nil" Info Package.

The initial population of this table shall be:

```
Name          MIME Type                 Standards Status      Reference
nil                                      Standards Track       [RFCXXXX]
```

TOC

9.5. Registration of the Info-Package Content-Disposition

Please add the following registration to the Content-Disposition registry. The description suitable for the IANA registry is as follows.

The payload of the message carrying this Content-Disposition header field value is the payload of an Info Package.

9.6. SIP Response Code 469 Registration

TOC

Please register the 469 response code in the Session Initiation Protocol Parameters - Response Codes registry as follows.

```
Response Code: 469
Default Reason Phrase: Bad INFO Package
Reference: RFCXXXX
```

10. Examples

TOC

10.1. Simple Info Package

TOC

Here Alice sends Bob a simple Info Package payload.

```
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>
Info-Package: foo
Content-type: application/foo
Content-length: 24

I am a foo message type
```

10.2. Multipart INFO Example

TOC

Other SIP extensions can put payloads into an INFO method, independent of the Info Package. In this case, the Info Package payload gets put into a Multipart MIME body, with the content disposition indicating which body belongs to the Info Package. Since there is one and only one Info Package payload in the message, we only need to tag which body part goes with the Info Package.

```
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: 7 INFO
```

```
Contact: <sip:bob@192.0.2.2>
Info-Package: foo
mumble-extension: <cid:abcd9999qq>
Content-Type: multipart/mixed;boundary="theboundary"
Content-Length: ...

--theboundary
Content-Type: application/mumble
Content-Id: abcd9999qq
...

<mumble stuff>

--theboundary
Content-Type: application/foo
Content-Disposition: Info-Package
Content-length: 24

I am a foo message type
--theboundary--
```

TOC

11. Modifications to SIP Change Process

This document updates **RFC 3427** [RFC3427] to add a process for registering new Info Packages. The process for registering new Info Packages follows the process outlined in Section 4.3 of RFC 3427 for the registration of SIP Event Packages. Namely, the registration of a new SIP Info Package requires the DISPATCH chairs to assign an individual to perform expert review of the proposal if the work is not a RAI work item in itself.

TOC

12. Security Considerations

By eliminating multiple uses of INFO messages without adequate community review and by eliminating the possibility for rogue SIP User Agents from confusing another User Agent by purposely sending unrelated INFO messages, we expect this document's clarification of the use of INFO to improve the security of the Internet. Whilst rogue UACs can still send unrelated INFO messages, this framework provides mechanisms for which the UAS and other security devices can filter for approved Info Packages.

If the content of the Info Package payload is private, User Agents will need to use end-to-end encryption, such as S/MIME, to prevent access to the content. This is particularly important as transport of INFO is likely not to be end-to-end, but through SIP proxies and back-to-back user agents (B2BUA's), which the user may not trust.

The INFO mechanism transports application level information. One implication of this is INFO messages may require a higher level of protection than the underlying SIP-based session signaling. In particular, if one does not protect the SIP signaling from eavesdropping or authentication and repudiation attacks, for example by using TLS transport, then the INFO request and its contents will be vulnerable, as well. Even with SIP/TLS, any SIP hop along the path from UAC to UAS can view, modify, or intercept INFO requests, as they can with any SIP request. This means some applications may require end-to-end encryption of the INFO payload, beyond, for example, hop-by-hop protection of the SIP signaling itself. Since the application dictates the level of security required, individual Info Packages have to enumerate these requirements. In any event, the INFO

Framework described by this document provides the tools for such secure, end-to-end transport of application data.

One interesting property of Info Package use is one can reuse the same digest-challenge mechanism used for INVITE-based authentication for the INFO request. For example, one could use a quality-of-protection (qop) value of authentication with integrity (auth-int), to challenge the request and its body, and prevent intermediate devices from modifying the body. 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.

13. References

TOC

13.1. Normative References

TOC

[I-D.ietf-sip-body-handling]	Camarillo, G., "**Message Body Handling in the Session Initiation Protocol (SIP)**," draft-ietf-sip-body-handling-06 (work in progress), March 2009 (**TXT**).
[RFC2119]	Bradner, S., "**Key words for use in RFCs to Indicate Requirement Levels**," RFC 2119, BCP 14, March 1997.
[RFC3261]	Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "**SIP: Session Initiation Protocol**," RFC 3261, June 2002 (**TXT**).
[RFC5226]	Narten, T. and H. Alvestrand, "**Guidelines for Writing an IANA Considerations Section in RFCs**," BCP 26, RFC 5226, May 2008 (**TXT**).

13.2. Informative References

TOC

[I-D.ietf-speechsc-mrcpv2]	Shanmugham, S. and D. Burnett, "**Media Resource Control Protocol Version 2 (MRCPv2)**," draft-ietf-speechsc-mrcpv2-19 (work in progress), June 2009 (**TXT**).
[I-D.saleem-msml]	Sharratt, G. and A. Saleem, "**Media Server Markup Language (MSML)**," draft-saleem-msml-08 (work in progress), February 2009 (**TXT**).
[RFC0768]	Postel, J., "**User Datagram Protocol**," STD 6, RFC 768, August 1980 (**TXT**).
[RFC0793]	Postel, J., "**Transmission Control Protocol**," STD 7, RFC 793, September 1981 (**TXT**).
[RFC2616]	**Fielding, R.**, **Gettys, J.**, **Mogul, J.**, **Frystyk, H.**, **Masinter, L.**, **Leach, P.**, and **T. Berners-Lee**, "**Hypertext Transfer Protocol -- HTTP/1.1**," RFC 2616, June 1999 (**TXT**, **PS**, **PDF**, **HTML**, **XML**).
[RFC2976]	Donovan, S., "**The SIP INFO Method**," RFC 2976, October 2000 (**TXT**).
[RFC3080]	**Rose, M.**, "**The Blocks Extensible Exchange Protocol Core**," RFC 3080, March 2001 (**TXT**, **HTML**, **XML**).
[RFC3265]	Roach, A., "**Session Initiation Protocol (SIP)-Specific Event Notification**," RFC 3265, June 2002 (**TXT**).
[RFC3372]	Vemuri, A. and J. Peterson, "**Session Initiation Protocol for Telephones (SIP-T): Context and Architectures**," BCP 63, RFC 3372, September 2002 (**TXT**).
[RFC3427]	Mankin, A., Bradner, S., Mahy, R., Willis, D., Ott, J., and B. Rosen, "**Change Process for the Session Initiation Protocol (SIP)**," BCP 67, RFC 3427, December 2002 (**TXT**).
[RFC3428]	Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C., and D. Gurle, "**Session Initiation Protocol (SIP) Extension for Instant Messaging**," RFC 3428, December 2002 (**TXT**).
[RFC3458]	Burger, E., Candell, E., Eliot, C., and G. Klyne, "**Message Context for Internet Mail**," RFC 3458, January 2003 (**TXT**).
[RFC3725]	Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "**Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)**," BCP 85, RFC 3725, April 2004 (**TXT**).
[RFC3841]	Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "**Caller Preferences for the Session Initiation Protocol (SIP)**," RFC 3841, August 2004 (**TXT**).
[RFC3851]	Ramsdell, B., "**Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification**," RFC 3851, July 2004 (**TXT**).
[RFC4028]	Donovan, S. and J. Rosenberg, "**Session Timers in the Session Initiation Protocol (SIP)**," RFC 4028, April 2005 (**TXT**).

[RFC4145] Yon, D. and G. Camarillo, "**TCP-Based Media Transport in the Session Description Protocol (SDP)**," RFC 4145, September 2005 (**TXT**).

[RFC4240] Burger, E., Van Dyke, J., and A. Spitzer, "**Basic Network Media Services with SIP**," RFC 4240, December 2005 (**TXT**).

[RFC4497] Elwell, J., Derks, F., Mourot, P., and O. Rousseau, "**Interworking between the Session Initiation Protocol (SIP) and QSIG**," BCP 117, RFC 4497, May 2006 (**TXT**).

[RFC4730] Burger, E. and M. Dolly, "**A Session Initiation Protocol (SIP) Event Package for Key Press Stimulus (KPML)**," RFC 4730, November 2006 (**TXT**).

[RFC4949] Shirey, R., "**Internet Security Glossary, Version 2**," RFC 4949, August 2007 (**TXT**).

[RFC4960] Stewart, R., "**Stream Control Transmission Protocol**," RFC 4960, September 2007 (**TXT**).

[RFC4975] Campbell, B., Mahy, R., and C. Jennings, "**The Message Session Relay Protocol (MSRP)**," RFC 4975, September 2007 (**TXT**).

[RFC5022] Van Dyke, J., Burger, E., and A. Spitzer, "**Media Server Control Markup Language (MSCML) and Protocol**," RFC 5022, September 2007 (**TXT**).

[RFC5057] Sparks, R., "**Multiple Dialog Usages in the Session Initiation Protocol**," RFC 5057, November 2007 (**TXT**).

[RFC5168] Levin, O., Even, R., and P. Hagendorf, "**XML Schema for Media Control**," RFC 5168, March 2008 (**TXT**).

[W3C.REC-voicexml21-20070619] Porter, B., McGlashan, S., Lee, A., Burnett, D., Carter, J., Oshry, M., Bodell, M., Baggia, P., Rehor, K., Burke, D., Candell, E., and R. Auburn, "**Voice Extensible Markup Language (VoiceXML) 2.1**," World Wide Web Consortium Recommendation REC-voicexml21-20070619, June 2007 (**HTML**).

TOC

Appendix A. Info Package Considerations

This section covers several issues that one should take into consideration when proposing new Info Packages.

TOC

A.1. Appropriateness of Usage

When designing an Info Package using the method described in this document for application level information exchange, it is important to consider: is INFO and, more importantly, is signaling within a SIP session, an appropriate mechanism for the problem set? Is it because it is the most reasonable and appropriate choice, or merely because "it's easy"?

These are difficult issues to consider, especially when presented with real-world deadlines and implementation cost issues. However, choosing to use INFO for inappropriate uses *will* lead to issues in the real world, not the least of which are certain types of middleboxes which will remove the device from the network if it is found to cause damage to other SIP nodes.

Therefore, the following sections provide consideration guidelines and alternatives to INFO use.

TOC

A.2. Dialog Fate-Sharing

INFO, by design, is a method within an INVITE dialog usage. **RFC 5057** [RFC5057] enumerates the problems with using dialogs for multiple usages, and we strongly urge the reader to review RFC 5057. The most relevant issue is a failure of transmission or processing of an INFO request may render the INVITE session terminated, depending on the type of failure. Prior to RFC 5057, it was not clear if the INFO usage was a separate usage or not. RFC 5057 clarifies the INFO method is always part of the INVITE usage.

Some uses of INFO can tolerate this fate sharing of the INFO message over the entire session. For example, in the SIP-T usage, it may be acceptable for a call to fail, or to tear

down the call, if one cannot deliver the associated SS7 information. The same is usually true for DTMF. However, it may not be acceptable for a call to fail if, for example, a DTMF buffer overflows. Then again, for some services, that may be the exact desired behavior.

A.3. Messaging Rates and Volume

TOC

There is no throttling mechanism for INFO. Consider that most call signaling occurs on the order of 7-10 messages per 3 minutes, although with a burst of 5-7 messages in one second during call setup. DTMF tones occur in bursts at a rate of up to 20 messages per second. This is a considerably higher rate than for call signaling. Sending constant GPS location updates, on the other hand, would incur an undue burden on SIP Proxies along the path.

Furthermore, SIP messages tend to be relatively small, on the order of 500 Bytes to 32K Bytes. SIP is a poor mechanism for direct exchange of bulk data beyond these limits, especially if the headers plus body exceed the **UDP MTU** [RFC0768]. Appropriate mechanisms for such traffic include **MSRP** [RFC4975], **COMEDIA** [RFC4145], or **HTTP** [RFC2616].

A.4. Is there a better alternative?

TOC

The first alternative for application level interaction is SIP Events, also known as **SUBSCRIBE/NOTIFY** [RFC3265]. In this model, a user agent requests state information, such as key pad presses from a device to an application server or key map images from an application server to a device. The SUBSCRIBE creates a new session that does not share the fate of the related INVITE-initiated session. Moreover, using the SUBSCRIBE model enables multiple applications to receive state updates. These applications can be outside the media path and potentially outside the INVITE-initiated session's proxy path. In fact, SIUBSCRIBE/NOTIFY is your only option if you need to exchange data outside a communications session.

SUBSCRIBE/NOTIFY messages pass through the SIP signaling infrastructure, such as SIP Proxies and B2BUAs. Application designers need to understand this can be a feature, as when the User Agents are exchanging information that elements in the SIP signaling path need to be aware of. Conversely, this can be a problem, as messages these network elements have no interest in can put a significant burden on those element's ability to process other traffic. Moreover, such network elements may not be able to read end-to-end encrypted SUBSCRIBE or NOTIFY bodies.

Implementers do need to be aware the price of having a protocol that works in all cases, can scale, can easily load balance, and will not mysteriously fail a session in the event of state synchronization failure does come at a cost. Session establishment is a minimum of two messages in addition to the INVITE session establishment. If the SUBSCRIBE application is co-resident with the INVITE application, the application will have to manage two SIP sessions instead of one. Tracking the application level state dominates memory and processing for some applications, and as such, the doubling of SIP sessions is not an issue. However, for other applications, this may be an issue.

The **MESSAGE method** [RFC3428] defines one-time instant message exchange, typically for sending MIME contents for rendering to the user.

Another model for application level information exchange is to establish a communication channel in the media plane. One model for this is **MRCPv2** [I-D.ietf-speechsc-mrcpv2].

Here, the INVITE-initiated session establishes a separate reliable, connection-oriented channel, such as a **TCP** [RFC0793] or **SCTP** [RFC4960] stream. One uses SIP to locate the remote endpoint, but uses a direct connection for the UUI. One then can create whatever protocol one wishes, whether from scratch (as in MRCPv2) or using a substrate such as **BEEP** [RFC3080].

A low latency requirement for the exchange of information is one strong indicator for using a media channel. Exchanging information through the SIP routing network can introduce hundreds of milliseconds of latency. In addition, if there will be a lot of information exchanged, and there is no need for the SIP routing network to examine the information, one should use a separate media channel.

Another model is to use a totally externally signaled channel, such as **HTTP** [RFC2616]. In this model, the user agent knows about a rendezvous point to direct HTTP requests to for the transfer of information. Examples include encoding of a prompt to retrieve in the SIP Request URI in **RFC 4240** [RFC4240] or the encoding of a SUBMIT target in a **VoiceXML** [W3C.REC-voicexml21-20070619] script.

MSRP [RFC4975] defines session-based instant messaging as well as bulk file transfer and other such large-volume uses. It is part of an INVITE-based session, similar to other media. Unlike INFO, MSRP follows a direct media path, rather than through the network elements composing the SIP signaling path.

A common reason people in the past used INFO for application level information exchange is the negotiation is very lightweight compared to SUBSCRIBE/NOTIFY. This is more especially so if it is not certain if there will be application level information exchange. The SUBSCRIBE/NOTIFY machinery requires the user agents to exchange rich capabilities and maintain state for additional SIP sessions. However, this is a weak argument if there is a high likelihood of application level information exchange. In this case, we recommend the use of a more robust application level information exchange protocol.

TOC

A.5. Alternatives for Common INFO Use

What alternatives to INFO are there for UA-to-UA application session signaling? As noted above, there are three broad classes of session signaling available. The choice depends on the circumstances. Following is a list of situations that have used INFO in the past.

- State updates
- User stimulus
- Direct signaling channel
- Proxy-aware signaling
- Dialog probe

TOC

A.5.1. State Updates

This is the broad class of one User Agent updating another with changes in state. The design goal of the **SUBSCRIBE/NOTIFY** [RFC3265] event framework is to meet just this need.

TOC

A.5.2. User Stimulus: Touch Tones and Others

This is the class of the user entering stimulus at one User Agent, and the User Agent transporting that stimulus to the other. A key thing to realize is key presses on the telephone keypad is user stimulus. Thus, the appropriate mechanism to use here is **KPML** [RFC4730].

TOC

A.5.3. Direct Signaling Channel

State updates and user stimulus tend to have relatively few messages per session. Sometimes, User Agents need to exchange a relatively high number of messages. In addition, User Agents may have a need for a relatively low-latency exchange of messages. In this latter case, the User Agent may not be able to tolerate the latency introduced by intermediate proxies. Likewise, the intermediate proxies may have no interest in processing all of that data.

In this case, establishing a separate, direct control channel, as in **MSRP** [RFC4975] or **MRCPv2** [I-D.ietf-speechsc-mrcpv2] is appropriate.

In addition, not every situation requires a SIP solution. Some signaling is really just one-shot to third-party endpoints. That situation may better be handled using an appropriate protocol, such as **HTTP** [RFC2616].

TOC

A.5.4. Proxy-Aware Signaling

Sometimes, one does want proxies to be in the signaling path for UA-to-UA application signaling. In this case, the use of a SIP request is appropriate. To date, there are no mechanisms for completely disambiguating INFO requests. For example, one could create a registry of INFO packages. The definition of the package would define the contexts for the various MIME Content-Types, as well as the context of the request itself. However, a package can have multiple content types. Moreover, having the context, or package identifier, at the SIP level precludes bundling multiple contexts responding in the same INFO request. For example, a User Agent might want to bundle two different responses in a multipart/mixed MIME body type.

Because there is no difference in either the protocol machinery or registration process due to these factors, we will not create an INFO framework. If one needs a SIP User Agent-to-SIP User Agent application session signaling transport protocol that touches all Record-Route proxies in a path, one MUST create a new SIP method as described in Section 27.4 of **RFC 3261** [RFC3261].

TOC

A.5.5. Dialog Probe

Some implementations in the wild use INFO to probe if an INVITE-initiated session is alive. While this works, it is NOT RECOMMENDED. In particular, **RFC 4028** [RFC4028] describes how to ensure an INVITE-initiated session is alive.

TOC

A.5.6. Malicious Indicator

Take the case of Malicious Indicator. This is where a subscriber receives a call, realizes it is

a malicious call (threatening, SPIT, etc.). They then press the SPIT button (or press *xx), which tells their service provider to mark the UAC as a bad actor. One might be tempted to think that INFO would be a great option for this service. It follows the return path of the INVITE, and so the INFO will hit the caller's inbound proxy, which it can learn the caller is (statistically) a bad actor. That way the inbound proxy can do stuff like notify law enforcement, add a vote to "this is a SPIT source," or other useful action.

However, consider a few issues. First, since INFO lives exclusively within an established session, there is no way to assert this message after the call completes. Second, this mechanism relies on an active service provider topology. If there is no proxy in the chain that will eat the INFO, the caller will see the "this is a bad guy" message, which may have consequences in the real world. Third, there is no a'priori way for the UAS to know whether it can issue the INFO. The caller certainly will not advertise, "please tell me if I am bad, particularly I know in advance that I *am* a bad actor."

One approach is for the service provider's proxy to SUBSCRIBE for the SPIT event at the UAS. At this point, life is good, interoperable, and works across networks. This enables events after the session is torn down, as presumably the SPIT event will refer not to, "this session," which does not exist, but to "that session identifier," which exists (and is theoretically unique) forever.

Another approach that saves considerably on the overhead of subscriptions would be for the service provider to insert a HTTP URI in the initial INVITE, noting it is for reporting malicious behavior. When the subscriber presses the SPIT button, an HTTP POST gets executed, delivering the call information to the service provider. The service provider can encode basic call information in the HTTP URI and can instruct the device to send whatever arbitrary data is necessary in the POST. This method has the added benefit of being entirely outside the real-time SIP proxy network.

TOC

Appendix B. Legacy INFO Usages

We do not intend this section to be a comprehensive catalog of INFO usages. However, it should give the reader a flavor for current INFO usages.

TOC

B.1. ISUP

SIP-T uses Content-Type to identify ISUP protocol elements in an INFO message. See **RFC3372** [RFC3372].

TOC

B.2. QSIG

QSIG uses Content-Type to identify QSIG protocol elements in an INFO message. See **RFC4497** [RFC4497].

TOC

B.3. MSCML

MSCML uses a Require to ensure the UAS understands that INFO messages of the MSCML type are in fact MSCML messages. See **RFC5022** [RFC5022].

B.4. MSML

TOC

MSML endpoints just know the INFO messages carry MSML and from the Content-Type of the given INFO method request. See the **MSML** [I-D.saleem-msml] draft.

B.5. Video Fast Update

TOC

Microsoft, Polycom, and Radvision used INFO messages as an interim solution for requesting fast video update before the ability to request I-Frames in RTCP was available. See the **XML Schema for Media Control** [RFC5168] for more information.

Appendix C. Acknowledgements

TOC

We are standing on the shoulders of giants. Jonathan Rosenberg did the original "INFO Considered Harmful" Internet Draft on 26 December 2002, which influenced the work group and this document. Likewise, Dean Willis influenced the text from his Internet Draft, "Packaging and Negotiation of INFO Methods for the Session Initiation Protocol" of 15 January 2003. Four paragraphs come from Jonathan Rosenberg's INFO Litmus draft. My, we have been working on this for a long time!

This and other related drafts have elicited well over 450 messages on the SIP list. People who have argued with its thesis, supported its thesis, added to the examples, or argued with the examples, include the following individuals:

> Adam Roach, Bram Verburg, Brian Stucker, Chris Boulton, Cullen Jennings, Dale Worley, Dean Willis, Frank Miller, Gonzalo Camarillo, Gordon Beith, Henry Sinnreich, James Jackson, James Rafferty, Jeroen van Bemmel, Joel Halpern, John Elwell, Johnathan Rosenberg, Juha Heinanen, Keith Drage, Kevin Attard Compagno, Manpreet Singh, Martin Dolly, Mary Barnes, Michael Procter, Paul Kyzivat, Peili Xu, Peter Blatherwick, Raj Jain, Rayees Khan, Robert Sparks, Roland Jesske, Salvatore Loreto, Sam Ganesan, Sanjay Sinha, Spencer Dawkins, Steve Langstaff, Sumit Garg, and Xavier Marjou.

John Elwell and Francois Audet helped with QSIG references. In addition, Francois Audet provided actual text for the revised abstract. Keith Drage gave lots of excellent comments and helped immensely with **Figure 1**.

The work group version of this document benefited from the close readings and comments from

> John Elwell, Paul Kyzivat, Dean Willis, Francois Audet, Dale Worley, Andrew Allen, Adam Roach, Anders Kristensen, Gordon Beith, Ben Campbell, Bob Penfield, Keith Drage, Jeroen van Bemmel, Mary Barnes, and Salvatore Loreto.

Since publication of the first work group version of this document, we have had over 329 messages. New voices in addition to those included above include

> Arun Arunachalam, Christian Stredicke, Eric Rescorla, Inaki Baz Castillo, and Roni Evan.

However, any errors and issues we missed are still our own.

Appendix D. Change Log

[RFC EDITOR NOTE: Please remove this section when publishing]

Changes from draft-ietf-sip-info-events-03

- Clarified Abstract language
- All SIP dialogs are now refered to as sessions
- Clarified the image example in the Introduction
- Clarified the relationship (none) between SIP Event Packages and SIP Info Packages
- Really, really clarified the protocol is NOT a negotiation but an advertisement
- Split Section 3 into UAS and UAC behavior
- Moved the example in section 3 into its own sub-section, and used full SIP headers
- Clarified forking behavior
- Clarified language around when to send a body
- Added 469 error response, instead of reusing 489
- Clarified overlapping INFO method handling
- Fixed table 1 to follow 3261, not 2543
- Added REFER to the INFO Headers table
- replaced token-nodot with token for Info-Package values
- Clarified end-to-end security considerations
- Info Package parameters are semi-colon delimited, not dot delimited

Changes from -02

- Applicability statement explicitly says we're backwards compatible
- Explicitly state we work like UPDATE (both early and confirmed dialogs)
- Agreed text for IANA Considerations package registry

Changes from -01

- One and only one Info Package per INFO
- Removed Send-Info header, greatly simplifying negotiation
- Multiple body part identification through Content-Disposition: Info-Package
- Note that forking INVITEs may result in multiple INFO's coming back to INVITE originator
- Describe how a UAS can enforce strict adherence to this document
- Remove CANCEL INFO faux pas
- Better explained overlapping INFO issues and resolutions
- Token names are now really case sensitive
- Moved Info Package Considerations to an Appendix
- Introduced stronger, yet more open, IANA registration process
- Took a few more paragraphs from INFO Litmus to cover all bases.
- Added RFC 5168 to legacy usages

Changes from -00

- Corrected ABNF.
- Enabled sending of legacy INFO messages. Receiving legacy INFO messages was already here.
- Negotiation is not Offer/Answer, it is Offer/Offer.
- Created the explicit "nil" Info Package to indicate no info package.
- Fixed CANCEL impacting future transactions.
- Added Registrar behavior.

- Added OPTIONS processing.
- Clarified overlapping INFO method processing.
- Described multiple INFO bodies in a single INFO method.
- Took out Info-Package as a header for responses to the INFO method.
- Expanded on risks of using INFO and filled-in more on the alternatives
- Moved definitions of INFO into the body of the text and cleaned up IANA Considerations section
- Added legacy usages descriptions

Authors' Addresses

Eric W. Burger
NeuStar, Inc.
46000 Center Oak Plaza
Sterling, VA 20166-6579
USA
Email: eburger@standardstrack.com
URI: http://www.standardstrack.com

Hadriel Kaplan
Acme Packet
71 Third Ave.
Burlington, MA 01803
USA
Phone:
Fax:
Email: hkaplan@acmepacket.com
URI:

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas, 02420
Finland
Phone:
Fax:
Email: christer.holmberg@ericsson.com
URI: