Network Working Group G. Camarillo (Editor), W. Marshall (Editor), Jonathan Rosenberg
Request for Comments: 3312 Ericsson, AT&T, dynamicsoft
Category: Standards Track October 2002

Integration of Resource Management and SIP

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This document defines a generic framework for preconditions, which are extensible through IANA registration. This document also discusses how network quality of service can be made a precondition for establishment of sessions initiated by the Session Initiation Protocol (SIP). These preconditions require that the participant reserve network resources before continuing with the session. We do not define new quality of service reservation mechanisms; these preconditions simply require a participant to use existing resource reservation mechanisms before beginning the session.

Contents

1 Introduction

Some architectures require that at session establishment time, once the callee has been alerted, the chances of a session establishment failure are minimum. One source of failure is the inability to reserve network resources for a session. In order to minimize "ghost rings", it is necessary to reserve network resources for the session before the callee is alerted. However, the reservation of network resources frequently requires learning the IP address, port, and session parameters from the callee. This information is obtained as a result of the initial offer/answer exchange carried in SIP. This exchange normally causes the "phone to ring", thus introducing a chicken-and-egg problem: resources cannot be reserved without performing an initial offer/answer exchange, and the initial offer/answer exchange can't be done without performing resource reservation.

The solution is to introduce the concept of a precondition. A precondition is a set of constraints about the session which are introduced in the offer. The recipient of the offer generates an answer, but does not alert the user or otherwise proceed with session establishment. That only occurs when the preconditions are met. This can be known through a local event (such as a confirmation of a resource reservation), or through a new offer sent by the caller.

This document deals with sessions that use SIP [1] as a signalling protocol and SDP [2] to describe the parameters of the session.

We have chosen to include the quality of service preconditions in the SDP description rather than in the SIP header because preconditions are stream specific.

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 BCP 14, RFC 2119 [3].

3 Overview

In order to ensure that session establishment does not take place until certain preconditions are met, we distinguish between two different state variables that affect a particular media stream: current status and desired status. This document defines the quality of service status.

The desired status consists of a threshold for the current status. Session establishment stops until the current status reaches or surpasses this threshold. Once this threshold is reached or surpassed, session establishment resumes.

For example, the following values for current and desired status would not allow session establishment to resume:

```
current status = resources reserved in the send direction
desired status = resources reserved in both (sendrecv) directions
```

On the other hand, the values of the example below would make session establishment resume:

```
current status = resources reserved in both (sendrecv) directions
desired status = resources reserved in the send direction
```

These two state variables define a certain piece of state of a media stream the same way the direction attribute or the codecs in use define other pieces of state. Consequently, we treat these two new variables in the same way as other SDP media attributes are treated in the offer/answer model used by SIP [4]: they are exchanged between two user agents using an offer and an answer in order to have a shared view of the status of the session.

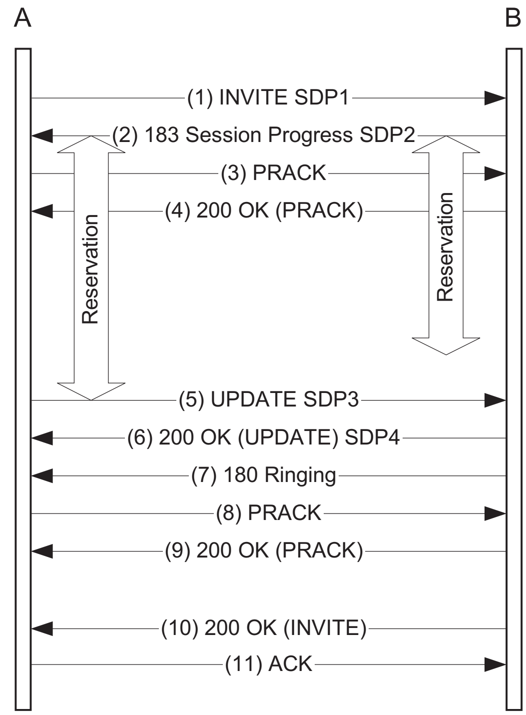

Figure 1: Basic session establishment using preconditions

Figure 1 shows a typical message exchange between two SIP user agents using preconditions. A includes quality of service preconditions in the SDP of the initial INVITE. A does not want B to be alerted until there are network resources reserved in both directions (sendrecv) end-to-end. B agrees to reserve network resources for this session before alerting the callee. B will handle resource reservation in the B->A direction, but needs A to handle the A->B direction. To indicate so, B returns a 183 (Session Progress) response to A asking A to start resource reservation and to confirm to B as soon as the A->B direction is ready for the session. A and B both start resource reservation. B finishes reserving resources in the B->A direction, but does not alert the user yet, because network resources in both directions are needed. When A finishes reserving resources in the A->B direction, it sends an UPDATE [5] to B. B returns a 200 (OK) response for the UPDATE, indicating that all the preconditions for the session have been met. At this point in time, B starts alerting the user, and session establishment completes normally.

4 SDP parameters

We define the following media level SDP attributes:

```
current-status     =  "a=curr:" precondition-type
                      SP status-type SP direction-tag
desired-status     =  "a=des:" precondition-type
                      SP strength-tag SP status-type
                      SP direction-tag
confirm-status     =  "a=conf:" precondition-type
                      SP status-type SP direction-tag
precondition-type  =  "qos" | token
strength-tag       =  ("mandatory" | "optional" | "none"
                   =  | "failure" | "unknown")
status-type        =  ("e2e" | "local" | "remote")
direction-tag      =  ("none" | "send" | "recv" | "sendrecv")
```

Current status: The current status attribute carries the current status of network resources for a particular media stream.

Desired status: The desired status attribute carries the preconditions for a particular media stream. When the direction-tag of the current status attribute with a given precondition-type/status-type for a particular stream is equal to (or better than) the direction-tag of the desired status attribute with the same precondition-type/status-type for that stream, then the preconditions are considered to be met for that stream.

Confirmation status: The confirmation status attribute carries threshold conditions for a media stream. When the status of network resources reach these conditions, the peer user agent will send an update of the session description containing an updated current status attribute for this particular media stream.

Precondition type: This document defines quality of service preconditions. Extensions may define other types of preconditions.

Strength tag: The strength-tag indicates whether or not the callee can be alerted in case the network fails to meet the preconditions.

Status type: We define two types of status: end-to-end and segmented. The end-to-end status reflects the status of the end-to-end reservation of resources. The segmented status reflects the status of the access network reservations of both user agents. The end-to-end status corresponds to the tag "e2e" defined above and the segmented status to the tags "local" and "remote". End-to-end status is useful when end-to-end resource reservation mechanisms are available. The segmented status is useful when one or both UAs perform resource reservations on their respective access networks.

Direction tag: The direction-tag indicates the direction in which a particular attribute (current, desired or confirmation status) is applicable to.

The values of the tags "send", "recv", "local" and "remote" represent the point of view of the entity generating the SDP description. In an offer, "send" is the direction offerer->answerer and "local" is the offerer's access network. In an answer, "send" is the direction answerer->offerer and "local" is the answerer's access network.

The following example shows these new SDP attributes in two media lines of a session description:

```
m=audio 20000 RTP/AVP 0
a=curr:qos e2e send
a=des:qos optional e2e send
a=des:qos mandatory e2e recv
m=audio 20002 RTP/AVP 0
a=curr:qos local sendrecv
a=curr:qos remote none
a=des:qos optional local sendrecv
a=des:qos mandatory remote sendrecv
```

5 Usage of preconditions with offer/answer

Parameter negotiation in SIP is carried out using the offer/answer model described in [4]. The idea behind this model is to provide a shared view of the session parameters for both user agents once the answer has been received by the offerer. This section describes which values our new SDP attributes can take in an answer, depending on their value in the offer.

To achieve a shared view of the status of a media stream, we define a model that consists of three tables: both user agents implement a local status table, and each offer/answer exchange has a transaction status table associated to it. The offerer generates a transaction status table identical to its local status table and sends it to the answerer in the offer. The anwerer uses the information of this transaction status table to update its local status table. The answerer also updates the transaction status table fields that were out of date and returns this table to the offerer in the answer. The offerer can then update its local status table with the information received in the answer. After this offer/answer exchange, the local status tables of both user agents are synchronised. They now have a common view of the status of the media stream. Sessions that involve several media streams implement these tables per media stream. Note, however, that this is a model of user agent behavior, not of software. An implementation is free to take any approach that replicates the external behavior this model defines.

5.1 Generating an offer

Both user agents MUST maintain a local precondition status, which is referred to as a "local status table". Tables 1 and 2 show the format of these tables for both the end-to-end and the segmented status types. For the end-to-end status type, the table contains two rows; one for each direction (i.e., send and recv). A value of "yes" in the "Current" field indicates the successful reservation of that resource in the corresponding direction. "No" indicates that resources have not been reserved yet. The "Desired Strength" field indicates the strength of the preconditions in the corresponding direction. The table for the segmented status type contains four rows: both directions in the local access network and in the peer's access network. The meaning of the fields is the same as in the end-to-end case.

Before generating an offer, the offerer MUST build a transaction status table with the current and the desired status for each media stream. The different values of the strength-tag for the desired status attribute have the following semantics:

- None: no resource reservation is needed.
- Optional: the user agents SHOULD try to provide resource reservation, but the session can continue regardless of whether or not this provision is possible.
- Mandatory: the user agents MUST provide resource reservation. Otherwise, session establishment MUST NOT continue.

The offerer then decides whether it is going to use the end-to-end status type or the segmented status type. If the status type of the media line will be end-to-end, the user agent generates records with the desired status and the current status for each direction (send and recv) independently, as shown in table 1:

Direction	Current	Desired Strength
send	no	mandatory
recv	no	mandatory

Table 1: Table for the end-to-end status type

If the status type of the media line will be segmented, the user agent generates records with the desired status and the current status for each direction (send and recv) and each segment (local and remote) independently, as shown in table 2:

Direction	Current	Desired Strength
local send	no	none
local recv	no	none
remote send	no	optional
remote recv	no	none

Table 2: Table for the segmented status type

At the time of sending the offer, the offerer's local status table and the transaction status table contain the same values.

With the transaction status table, the user agent MUST generate the current-status and the desired status lines following the syntax of Section 4 and the rules described below in Section 5.1.1.

5.1.1 SDP encoding

For the end-to-end status type, the user agent MUST generate one current status line with the tag "e2e" for the media stream. If the strength-tags for both directions are equal (e.g., both "mandatory") in the transaction status table, the user agent MUST add one desired status line with the tag "sendrecv". If both tags are different, the user agent MUST include two desired status lines, one with the tag "send" and the other with the tag "recv".

> The semantics of two lines with the same strength-tag, one with a "send" tag and the other with a "recv" tag, is the same as one "sendrecv" line. However, in order to achieve a more compact encoding, we have chosen to make the latter format mandatory.

For the segmented status type, the user agent MUST generate two current status lines: one with the tag "local" and the other with the tag "remote". The user agent MUST add one or two desired status lines per segment (i.e., local and remote). If, for a particular segment (local or remote), the tags for both directions in the transaction status table are equal (e.g., both "mandatory"), the user agent MUST add one desired status line with the tag "sendrecv". If both tags are different, the user agent MUST include two desired status lines, one with the tag "send" and the other with the tag "recv".

Note that the rules above apply to the desired strength-tag "none" as well. This way, a user agent that supports quality of service but does not intend to use them, adds desired status lines with the strength-tag "none". Since this tag can be upgraded in the answer, as described in Section 5.2, the answerer can request quality of service reservation without a need of another offer/answer exchange.

The example below shows the SDP corresponding to tables 1 and 2.

```
m=audio 20000 RTP/AVP 0
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
m=audio 20002 RTP/AVP 0
a=curr:qos local none
a=curr:qos remote none
a=des:qos optional remote send
a=des:qos none remote recv
a=des:qos none local sendrecv
```

5.2 Generating an Answer

When the answerer receives the offer, it recreates the transaction status table using the SDP attributes contained in the offer. The answerer updates both its local status and the transaction status table following the rules below:

Desired Strength: We define an absolute ordering for the strength-tags: "none", "optional" and "mandatory". "Mandatory" is the tag with the highest grade and "none" the tag with the lowest grade. An answerer MAY upgrade the desired strength in any entry of the transaction status table, but it MUST NOT downgrade it. Therefore, it is OK to upgrade a row from "none" to "optional", from "none" to "mandatory" or from "optional" to "mandatory", but not the other way around.

Current Status: For every row, the value of the "Current" field in the transaction status table, and in the local status table of the answerer, have to be compared. Table 3 shows the four possible combinations. If both fields have the same value (two first rows of table 3, nothing needs to be updated. If the "Current" field of the transaction status table is "Yes" and the field of the local status table is "No" (third row of table 3), the latter MUST be set to "Yes". If the "Current" field of the transaction status table is "No" and the field of the local status table is "Yes" (forth row of table 3), the answerer needs to check if it has local information (e.g., a confirmation of a resource reservation has been received) about that particular current status. If it does, the "Current" field of the transaction status table is set to "Yes". If the answerer does not have local information about that current status, the "Current" field of the local status table MUST be set to "No".

Transac. status table	Local status table	New values transac./local
no	no	no/no
yes	yes	yes/yes
yes	no	yes/yes
no	yes	depends on local info

Table 3: Possible values for the "Current" fields

Once both tables have been updated, an answer MUST be generated following the rules described in Section 5.1.1, taking into account that "send", "recv", "local" and "remote" tags have to be inverted in the answer, as shown in table 4.

Offer	Answer
send	recv
recv	send
local	remote
remote	local

Table 4: Values of tags in offers and answers

At the time the answer is sent, the transaction status table and the answerer's local status table contain the same values. Therefore, this answer contains the shared view of the status of the media line in the current-status attribute and the negotiated strength and direction-tags in the desired-status attribute.

If the resource reservation mechanism used requires participation of both user agents, the answerer SHOULD start resource reservation after having sent the answer and the offerer SHOULD start resource reservation as soon as the answer is received. If participation of the peer user agent is not needed (e.g., segmented status type), the offerer MAY start resource reservation before sending the offer and the answerer MAY start it before sending the answer.

The status of the resource reservation of a media line can change between two consecutive offer/answer exchanges. Therefore, both user agents MUST keep their local status tables up to date, using local information throughout the duration of the session.

6 Suspending and Resuming Session Establishment

A user agent server that receives an offer with preconditions SHOULD NOT alert the user until all the mandatory preconditions are met; session establishment is suspended until that moment (e.g., a PSTN gateway reserves resources without sending signalling to the PSTN.)

A user agent server may receive an INVITE request with no offer in it. In this case, following normal procedures defined in [1] and [5], the user agent server will provide an offer in a reliable 1xx response. The user agent client will send the answer in another SIP request (i.e., the PRACK for the 1xx). If the offer and the answer contain preconditions, the user agent server SHOULD NOT alert the user until all the mandatory preconditions in the answer are met.

> Note that in this case, a user agent server providing an initial offer with preconditions, a 180 (Ringing) response with preconditions will never be sent, since the user agent server cannot alert the user until all the preconditions are met.

A UAS that is not capable of unilaterally meeting all of the mandatory preconditions MUST include a confirm-status attribute in the SDP (offer or answer) that it sends (see Section 7). Further, the SDP (offer or answer) that contains this confirm-status attribute MUST be sent as soon as allowed by the SIP offer/answer rules.

While session establishment is suspended, user agents SHOULD not send any data over any media stream. In the case of RTP [6], neither RTP nor RTCP packets are sent.

A user agent server knows that all the preconditions are met for a media line when its local status table has a value of "yes" in all the rows whose strength-tag is "mandatory". When the preconditions of all the media lines of the session are met, session establishment SHOULD resume.

For an initial INVITE suspending and resuming session establishment is very intuitive. The callee will not be alerted until all the mandatory preconditions are met. However, offers containing preconditions sent in the middle of an ongoing session need further explanation. Both user agents SHOULD continue using the old session parameters until all the mandatory preconditions are met. At that moment, the user agents can begin using the new session parameters. Section 13 contains an example of this situation.

7 Status Confirmation

The confirm-status attribute MAY be used in both offers and answers. This attribute represents a threshold for the resource reservation. When this threshold is reached or surpassed, the user agent MUST send an offer to the peer user agent, reflecting the new current status of the media line as soon as allowed by the SIP offer/answer rules. If this threshold is crossed again (e.g., the network stops providing resources for the media stream), the user agent MUST send a new offer as well, as soon as allowed by the SIP offer/answer rules.

If a peer has requested confirmation on a particular stream, an agent MUST mark that stream with a flag in its local status table. When all the rows with this flag have a "Current" value of "yes", the user agent MUST send a new offer to the peer. This offer will contain the current status of resource reservation in the current-status attributes. Later, if any of the rows with this flag transition to "No", a new offer MUST be sent as well.

Confirmation attributes are not negotiated. The answerer uses the value of the confirm-status attribute in the offer and the offerer uses the value of this attribute in the answer.

For example, if a user agent receives an SDP description with the following attributes:

```
m=audio 20002 RTP/AVP 0
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
a=conf:qos remote sendrecv
```

It will send an offer as soon as it reserves resources in its access network ("remote" tag in the received message) for both directions (sendrecv).

8 Refusing an offer

We define a new SIP status code:

```
Server-Error =  "580"  ;Precondition Failure
```

When a UAS acting as an answerer, cannot or is not willing to meet the preconditions in the offer, it SHOULD reject the offer by returning a 580 (Precondition-Failure) response.

Using the 580 (Precondition Failure) status code to refuse an offer is useful when the offer comes in an INVITE or in an UPDATE request. However, SIP does not provide a means to refuse offers that arrive in a response (1xx or 2xx) to an INVITE. If a UAC generates an initial INVITE without an offer and receives an offer in a 1xx or 2xx response which is not acceptable, it SHOULD respond to this offer with a correctly formed answer and immediately send a CANCEL or a BYE.

If the offer comes in a 1xx or 2xx response to a re-INVITE, A would not have a way to reject it without terminating the session at the same time. The same recommendation given in Section 14.2 of [1] applies here:

> "The UAS MUST ensure that the session description overlaps with its previous session description in media formats, transports, other parameters that require support from the peer. This is to avoid the need for the peer to reject the session description. If, however, it is unacceptable to A, A SHOULD generate an answer with a valid session description, and then send a BYE to terminate the session."

580 (Precondition Failure) responses and BYE and CANCEL requests indicating failure to meet certain preconditions SHOULD contain an SDP description indicating which desired status triggered the failure. Note that this SDP description is not an offer or an answer, since it does not lead to the establishment of a session. The format of such a description is based on the last SDP (an offer or an answer) received from the remote UA.

For each "m=" line in the last SDP description received, there MUST be a corresponding "m=" line in the SDP description indicating failure. This SDP description MUST contain exactly the same number of "m=" lines as the last SDP description received. The port number of every "m=" line MUST be set to zero, but the connection address is arbitrary.

The desired status line corresponding to the precondition that triggered the failure MUST use the "failure" strength-tag, as shown in the example below:

```
m=audio 20000 RTP/AVP 0
```

```
a=des:qos failure e2e send
```

8.1 Rejecting a Media Stream

In the offer/answer model, when an answerer wishes to reject a media stream, it sets its port to zero. The presence of preconditions does not change this behaviour; streams are still rejected by setting their port to zero.

Both the offerer and the answerer MUST ignore all the preconditions that affect a stream with its port set to zero. They are not taken into consideration to decide whether or not session establishment can resume.

9 Unknown Precondition Type

This document defines the "qos" tag for quality of service preconditions. New precondition-types defined in the future will have new associated tags. A UA that receives an unknown precondition-type, with a "mandatory" strength-tag in an offer, MUST refuse the offer unless the only unknown mandatory preconditions have the "local" tag. In this case, the UA does not need to be involved in order to meet the preconditions. The UA will ask for confirmation of the preconditions and, when the confirmation arrives, it will resume session establishment.

A UA refusing an offer follows the rules described in section 8, but instead of the tag "failure", it uses the tag "unknown", as shown in the example below:

```
m=audio 20000 RTP/AVP 0
a=des:foo unknown e2e send
```

10 Multiple Preconditions per Media Stream

A media stream MAY contain multiple preconditions. Different preconditions MAY have the same precondition-type and different status-types (e.g., end to end and segmented quality of service preconditions) or different precondition-types (this document only defines the "qos" precondition type, but extensions may define more precondition-types in the future).

All the preconditions for a media stream MUST be met in order to resume session establishment. The following example shows a session description that uses both end-to-end and segmented status-types for a media stream.

```
m=audio 20000 RTP/AVP 0
a=curr:qos local none
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
a=curr:qos e2e none
a=des:qos optional e2e sendrecv
```

11 Option Tag for Preconditions

We define the option tag "precondition" for use in the Require and Supported header fields. An offerer MUST include this tag in the Require header field if the offer contains one or more "mandatory" strength-tags. If all the strength-tags in the description are "optional" or "none", the offerer MUST include this tag in either a Supported header field or in a Require header field. It is, however, RECOMMENDED that the Supported header field be used in this case. The lack of preconditions in the answer would indicate that the answerer did not support this extension.

The mapping of offers and answers to SIP requests and responses is performed following the rules given in [5]. Therefore, a user agent including preconditions in the SDP MUST support the PRACK and UPDATE methods. Consequently, it MUST include the "100rel" [7] tag in the Supported header field and SHOULD include an Allow header field with the "UPDATE" tag [5].

12 Indicating Capabilities

The offer/answer model [4] describes the format of a session description to indicate capabilities. This format is used in responses to OPTIONS requests. A UA that supports preconditions SHOULD add desired status lines indicating the precondition-types supported for each media stream. These lines MUST have the "none" strength-tag, as shown in the example below:

```
m=audio 0 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=des:foo none e2e sendrecv
a=des:qos none local sendrecv
```

Note that when this document was published, the precondition-type "foo" has not been registered. It is used here in the session description above to provide an example with multiple precondition-types.

A UA that supports this framework SHOULD add a "precondition" tag to the Supported header field of its responses to OPTIONS requests.

13 Examples

The following examples cover both status types; end-to-end and segmented.

13.1 End-to-end Status Type

The call flow of Figure 2 shows a basic session establishment using the end-to-end status type. The SDP descriptions of this example are shown below:

SDP1: A includes end-to-end quality of service preconditions in the initial offer.

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.1
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
```

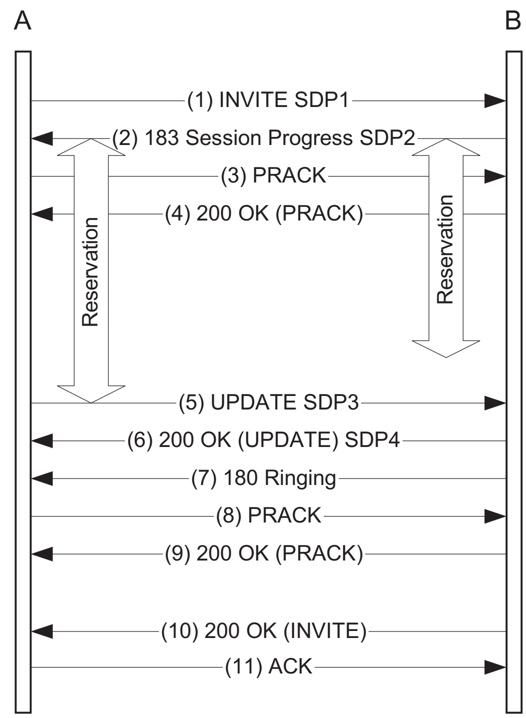

Figure 2: Example using the end-to-end status type

SDP2: Since B uses RSVP, it can know when resources in its "send" direction are available, because it will receive RESV messages from the network. However, it does not know the status of the reservations in the other direction. B requests confirmation for resource reservations in its "recv" direction to the peer user agent A in its answer.

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
```

After having sent the answer B starts reserving network resources for the media stream. When A receives this answer (2), it starts performing resource reservation as well. Both UAs use RSVP, so A sends PATH messages towards B and B sends PATH messages towards A.

As time passes, B receives RESV messages confirming the reservation. However, B waits until resources in the other direction are reserved as well since it did not receive any confirmation and the preconditions still have not been met.

SDP3: When A receives RESV messages it sends an updated offer (5) to B:

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.1
a=curr:qos e2e send
a=des:qos mandatory e2e sendrecv
```

SDP4: B responds with an answer (6) which contains the current status of the resource reservation (i.e., sendrecv):

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e sendrecv
a=des:qos mandatory e2e sendrecv
```

At this point in time, session establishment resumes and B returns a 180 (Ringing) response (7).

Let's assume, that in the middle of the session, A wishes to change the IP address where it is receiving media. Figure 3 shows this scenario.

SDP1: A includes an offer in a re-INVITE (1). A continues to receive media on the old IP address (192.0.2.1), but is ready to receive media on the new one as well (192.0.2.2):

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.2
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
```

SDP2: B includes a "conf" attribute in its answer. B continues sending media to the old remote IP address (192.0.2.1)

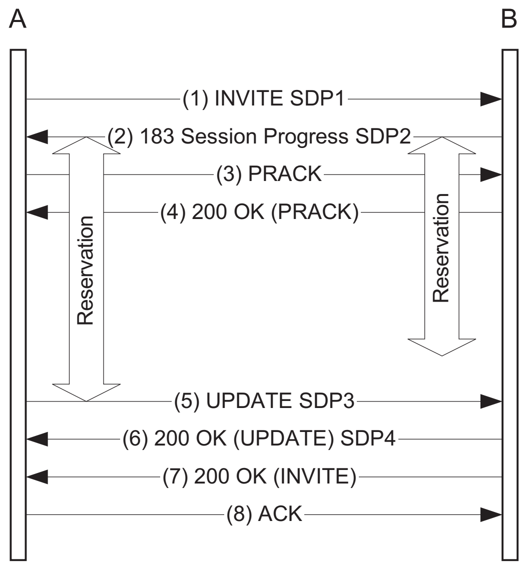

Figure 3: Session modification with preconditions

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
```

SDP3: When A receives RESV messages it sends an updated offer (5) to B:

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.2
a=curr:qos e2e send
a=des:qos mandatory e2e sendrecv
```

SDP4: B responds with an answer (6), indicating that the preconditions have been met (current status "sendrecv). It is now that B begins sending media to the new remote IP address (192.0.2.2).

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e sendrecv
a=des:qos mandatory e2e sendrecv
```

13.2 Segmented Status Type

The call flow of Figure 4 shows a basic session establishment using the segmented status type. The SDP descriptions of this example are shown below:

SDP1: A includes local and remote QoS preconditions in the initial offer. Before sending the

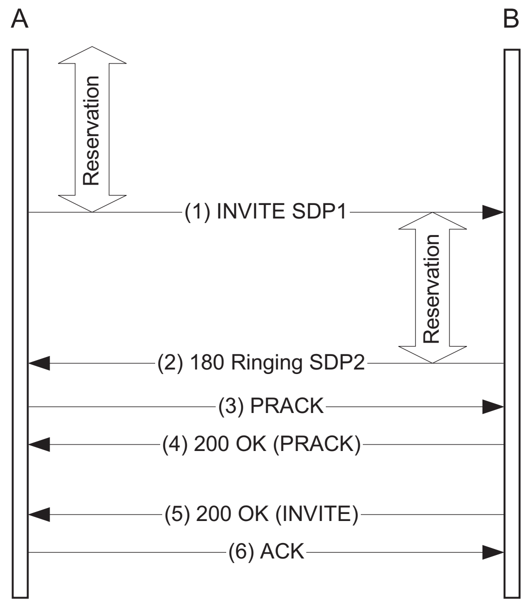

Figure 4: Example using the segmented status type

initial offer, A reserves resources in its access network. This is indicated in the local current status of the SDP below:

```
m=audio 20000 RTP/AVP 0 8
c=IN IP4 192.0.2.1
a=curr:qos local sendrecv
a=curr:qos remote none
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

SDP2: B reserves resources in its access network and, since all the preconditions are met, returns an answer in a 180 (Ringing) response (3).

```
m=audio 30000 RTP/AVP 0 8
c=IN IP4 192.0.2.4
a=curr:qos local sendrecv
a=curr:qos remote sendrecv
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

Let's assume that after receiving this response, A decides that it wants to use only PCM u-law (payload 0), as opposed to both PCM u-law and A-law (payload 8). It would send an UPDATE to B, possibly before receiving the 200 (OK) for the INVITE (5). The SDP would look like:

```
m=audio 20000 RTP/AVP 0
```

```
c=IN IP4 192.0.2.1
a=curr:qos local sendrecv
a=curr:qos remote sendrecv
a=des:qos mandatory local sendrecv
a=des:qos mandatory remote sendrecv
```

B would generate an answer for this offer and place it in the 200 (OK) for the UPDATE.

Note that this last offer/answer to reduce the number of supported codecs may arrive to the user agent server after the 200 (OK) response has been generated. This would mean that the session is established before A has reduced the number of supported codecs. To avoid this situation, the user agent client could wait for the first answer from the user agent before setting its local current status to "sendrecv".

13.3 Offer in a SIP response

The call flow of Figure 5 shows a basic session establishment where the initial offer appears in a reliable 1xx response. This example uses the end-to-end status type. The SDP descriptions of this example are shown below:

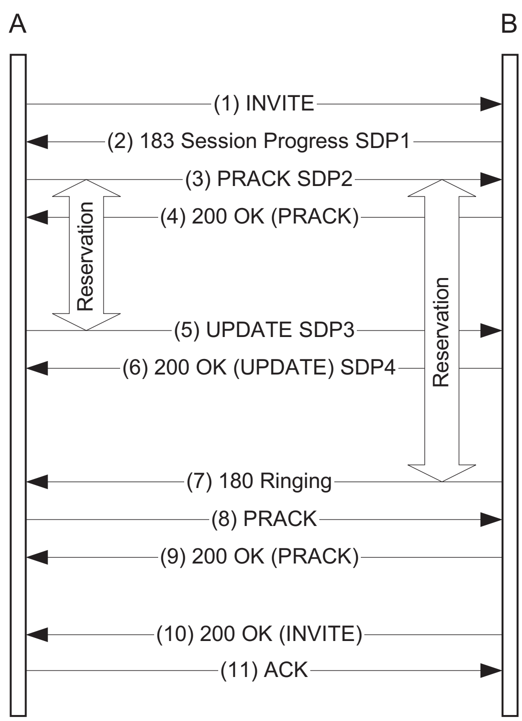

Figure 5: Example of an initial offer in a 1xx response

The first INVITE (1) does not contain a session description. Therefore, the initial offer is sent by B in a reliable 183 (Session Progress) response.

SDP1: B includes end-to-end quality of service preconditions in the initial offer. Since B uses RSVP, it can know when resources in its "send" direction are available, because it will receive RESV messages from the network. However, it does not know the status of the reservations in the other direction. B requests confirmation for resource reservations in its "recv" direction, to the peer user agent A, in its answer.

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
```

SDP2: A includes its answer in the PRACK for the 183 (Session Progress) response.

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.1
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
```

After having sent the answer, A starts reserving network resources for the media stream. When B receives this answer (3), it starts performing resource reservation as well. Both UAs use RSVP, so A sends PATH messages towards B and B sends PATH messages towards A.

SDP3: When A receives RESV messages it sends an updated offer (5) to B:

```
m=audio 20000 RTP/AVP 0
c=IN IP4 192.0.2.1
a=curr:qos e2e send
a=des:qos mandatory e2e sendrecv
```

SDP4: B responds with an answer (6) which contains the current status of the resource reservation (i.e., recv):

```
m=audio 30000 RTP/AVP 0
c=IN IP4 192.0.2.4
a=curr:qos e2e recv
a=des:qos mandatory e2e sendrecv
```

As time passes, B receives RESV messages confirming the reservation. At this point in time, session establishment resumes and B returns a 180 (Ringing) response (7).

14 Security Considerations

An entity in the middle of two user agents establishing a session may add desired-status attributes making session establishment impossible. It could also modify the content of the current-status parameters so that the session is established without meeting the preconditions. Integrity protection can be used to avoid these attacks.

An entity performing resource reservations upon reception of unauthenticated requests carrying preconditions can be an easy target for a denial of service attack. Requests with preconditions SHOULD be authenticated.

15 IANA considerations

This document defines three media level SDP attributes: desired-status, current-status and conf-status. Their format is defined in Section 4.

This document defines a framework for using preconditions with SIP. Precondition-types to be used with this framework are registered by the IANA when they are published in standards track RFCs. The IANA Considerations section of the RFC MUST include the following information, which appears in the IANA registry along with the RFC number of the publication.

- Name of the precondition-type. The name MAY be of any length, but SHOULD be no more than ten characters long.
- Descriptive text that describes the extension.

The only entry in the registry for the time being is:

Pecondition-Type	Reference	Description
qos	RFC 3312	Quality of Service preconditions

This document also defines a new SIP status code (580). Its default reason phrase (Precondition Failure) is defined in section 8.

This document defines a SIP option tag (precondition) in section 11.

16 Notice Regarding Intellectual Property Rights

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

17 References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: session initiation protocol," RFC 3261, Internet Engineering Task Force, June 2002.

[2] M. Handley and V. Jacobson, "SDP: session description protocol," RFC 2327, Internet Engineering Task Force, Apr. 1998.

[3] S. Bradner, “Key words for use in RFCs to indicate requirement levels,” RFC 2119, Internet Engineering Task Force, Mar. 1997.

[4] J. Rosenberg and H. Schulzrinne, “An offer/answer model with session description protocol (SDP),” RFC 3264, Internet Engineering Task Force, June 2002.

[5] J. Rosenberg, “The session initiation protocol (SIP) UPDATE method,” RFC 3311, Internet Engineering Task Force, Oct. 2002.

[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: a transport protocol for real-time applications,” RFC 1889, Internet Engineering Task Force, Jan. 1996.

[7] J. Rosenberg and H. Schulzrinne, “Reliability of provisional responses in session initiation protocol (SIP),” RFC 3262, Internet Engineering Task Force, June 2002.

[8] C. Kalmanek, W. Marshall, P. Mishra, D. Nortz, and K. K. Ramakrishnan, “DOSA: an architecture for providing robust IP telephony service,” in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (Tel Aviv, Israel), Mar. 2000.

18 Contributors

The following persons contributed and were co-authors on earlier versions of this spec:

> K. K. Ramakrishnan (TeraOptic Networks), Ed Miller (Terayon), Glenn Russell (CableLabs), Burcak Beser (Pacific Broadband Communications), Mike Mannette (3Com), Kurt Steinbrenner (3Com), Dave Oran (Cisco), Flemming Andreasen (Cisco), Michael Ramalho (Cisco), John Pickens (Com21), Poornima Lalwaney (Nokia), Jon Fellows (Copper Mountain Networks), Doc Evans (D. R. Evans Consulting), Keith Kelly (NetSpeak), Adam Roach (dynamicsoft), Dean Willis (dynamicsoft), Steve Donovan (dynamicsoft), Henning Schulzrinne (Columbia University).

This ”manyfolks” document is the culmination of over two years of work by many individuals, most are listed here and in the following acknowledgements section. A special note is due to Flemming Andreasen, Burcak Beser, Dave Boardman, Bill Guckel, Chuck Kalmanek, Keith Kelly, Poornima Lalwaney, John Lawser, Bill Marshall, Mike Mannette, Dave Oran, K.K. Ramakrishnan, Michael Ramalho, Adam Roach, Jonathan Rosenberg, and Henning Schulzrinne for spearheading the initial ”single INVITE” quality of service preconditions work from previous, non-SIP compatible, ”two-stage Invite” proposals. These ”two-stage INVITE” proposals had their origins from Distributed Call Signaling work in PacketCable, which, in turn, had architectural elements from AT&T’s Distributed Open Systems Architecture (DOSA) work [8].

19 Acknowledgments

The Distributed Call Signaling work in the PacketCable project is the work of a large number of people, representing many different companies. The authors would like to recognize and thank the following for their assistance: John Wheeler, Motorola; David Boardman, Daniel Paul, Arris Interactive; Bill Blum, Jay Strater, Jeff Ollis, Clive Holborow, General Instruments; Doug Newlin, Guido Schuster, Ikhlaq Sidhu, 3Com; Jiri Matousek, Bay Networks; Farzi Khazai, Nortel; John Chapman, Bill Guckel, Cisco; Chuck Kalmanek, Doug Nortz, John Lawser, James Cheng, Tung-Hai Hsiao, Partho Mishra, AT&T; Telcordia Technologies; and Lucent Cable Communications.

Miguel Angel Garcia-Martin, Rohan Mahy and Mark Watson provided helpful comments and suggestions.

20 Authors' Addresses

Gonzalo Camarillo
Ericsson
Advanced Signalling Research Lab.
FIN-02420 Jorvas
Finland
Email: Gonzalo.Camarillo@ericsson.com

Bill Marshall
AT&T
Florham Park, NJ 07932
USA
Email: wtm@research.att.com

Jonathan Rosenberg
dynamicsoft
72 Eagle Rock Ave
East Hanover, NJ 07936
USA
Email: jdrosen@dynamicsoft.com

21 Full Copyright Statement

Acknowledgement

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