Correlation of multiple responses of forked INVITES in Back to Back User Agents
draft-jesske-dispatch-forking-answer-correlation-03

Abstract

This document describes the scenarios where multiple early dialogs can be created. The main use case is based on forking. But also forwarding of SIP Invites and other applications may create early dialogs. The scenarios shown describe how a correlation/multiplexing of multiple early dialogs caused by forked or forwarded INVITEs in Back to Back User Agents can apply. Existing RFC's are analyzed how forking is described and points to facts which may be taken in to consideration.

Requirements Language

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].

Status of This Memo

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

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

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

This Internet-Draft will expire on September 4, 2015.
Table of Contents

1.  Problem Statement ........................................... 3
2.  Consideration of RFC’s on SIP signalling procedures under consideration for forking use cases ................. 5
   2.1.  Overview .................................................. 6
   2.2.  RFC3261 Session Initiation Protocol .................... 6
   2.3.  RFC3960 Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP) ..................... 8
   2.4.  RFC3262 Reliability of provisional responses ............ 8
   2.5.  RFC3312 Integration of Resource Management and Session Initiation Protocol (SIP) ......................... 8
   2.6.  RFC3841 Caller Preferences for the Session Initiation Protocol (SIP) .................................. 9
   2.7.  RFC5393 Addressing an Amplification Vulnerability in Session Initiation Protocol (SIP) Forking Proxies .. 9
   2.8.  RFC6228 Session Initiation Protocol (SIP) Response Code for Indication of Terminated Dialog ............... 10
   2.9.  RFC3326 The Reason Header Field for the Session Initiation Protocol (SIP) .................................. 10
   2.10. Conclusions ................................................ 10
3.  Requirements on forking in SIP networks interconnecting with other SIP networks ..................................... 11
4.  Forking use cases .............................................. 11
   4.1.  Normal Forking use case .................................. 11
   4.2.  Multiples provisional responses without SDP .......... 13
   4.3.  Forking use case with provisional responses with SDP using 100rel ........................................... 15
   4.4.  Forking use case with provisional responses with SDP using 100rel and preconditions ....................... 18
   4.5.  Forking use case with early media played ............... 22
   4.6.  Multiples early dialogs and announcements due to call forwarding ........................................... 25
1. Problem Statement

Within RFC3261 [RFC3261] the handling of multiples early responses received from different UAS for one sent INVITE is described as basis feature. Such behavior results usually in case of forking or also when an INVITE is forwarded to another target.

The forking feature described in RFC3261 [RFC3261] allows to spread the INVITE request towards multiples registered UA’s for one identity but is present at different locations.

A forwarding as described in RFC3960 [RFC3960] of a SIP Dialog may result in multiples early dialogs (181 and 180 with different to tags). A 181 response may also initiate an early dialog.

There are scenarios within the scope of this document where multiple responses received should be multiplexed to a single dialog to avoid complexity or miss behavior for the originating network/UAC:

1. When a SIP entity (B2BUA) provides a service within the SIP path where multiples early responses received must be multiplexed into a single dialog.

2. Providing interconnecting between different networks with different behavior where a the originating network implementations do not wish, have restrictions or may not able handle multiples responses while the destination still deliver multiples responses or early dialogs.

3. Providing interconnection where the terminating service provider will have or must have control what kind of early dialog information should be provided to the originating network based on bilateral agreement, service provider policy or regulatory policy.

4. Service provider providing the forking feature or interconnecting to networks providing multiples early dialogs to INVITES would like to increase the reliability of the network in increasing successful calls even if UAC do not really support the correct handling of
multiples early dialogs. Please Note that this scenario is assuming that old or faulty implementations are existing. This is real life and service provider are already using healing functions within B2BUA but it is clear that the correct way is to replace or correct faulty implementations of SIP.

Scenario 1:

A service example may be a directory service providing a early dialog/session with a Interactive Voice Response (IVR) system for requesting a destination number where the user will be automatically forwarded to a identity where a couple of UAS are registered within the SIP network. Thus the INVITE will be forked and result in multiples responses to the INVITE sent by the SIP entity providing the service. It could be also a normal call forwarding scenario providing a short announcement to the UAC indicating the forwarding. Due to the fact that a early session was already provided by the service it needs an mapping of the multiples responses towards the UAC to provide the ringing state, tone or provide further announcements.

Scenario 2:

Due to the fact that the world of SIP networks is steadily growing and SIP networks based on IETF RFC 3261 are existing with different flavors like based on pure SIP, 3GPP IMS or ETSI TISPAN NGN and many others. Now connecting these networks which may be operated by different service providers may result in complexity due to signaling, charging or even worse in faulty cases. Providing voice services via SIP over restricted access capabilities may need restrictions with regard to numbers of early sessions provided in backward direction or signaling load.

Other service providers do only allow single dialog over interconnection to avoid charging complexity with too many early sessions or early dialogs.

Also equipment and UAC not understanding or misbehaving when receiving multiples responses may be in focus when restricting the interconnection use case to a single early dialog.

Scenario 3:

Within this scenario the terminating service provider apply specific policy. This may be either in guaranteeing his customers to provide forking to for all entities registered for a specific identity, or allow only specific announcements (e.g in a specific language) passed through or to ensure the bilateral agreements with other operators.
where only single dialog is mandated. Also in forwarding scenarios some originating networks do not wish to have a forwarding indication (i.e. 181) or early session in backward direction.

Scenario 4:

As already stated this is a real life scenario but seen from implementation point of view the wrong approach. The real solution is to replace or correct faulty implementations of SIP.

But nevertheless the problem existing is that feature forking is not really supported by all UAC in that way that a connection between UAC and one of the UAS will succeed in a successful communication even one of the UAS sends a 200 OK for the INVITE. This apply also to networks having B2BUA’s (e.g. PSTN interworking Gateways or Session Border Controller) in the path which should understand multiples responses and handle it correctly.

Providing solutions for scenarios 1-3 will also provide a possibility for service provider to support scenario 4.

This document evaluates the existing forking mechanism described in RFC3261 [RFC3261] and further RFC’s extending SIP for early dialog handling, resource handling and reliability of provisional responses. Also directives and SIP extensions will be investigated where forking or fraud based on forking may be avoided. It will result in different use cases and solutions for the above mentioned scenarios where multiples responses are received by B2BUA and may be multiplexed to a single dialog.

This document describes how a correlation/multiplexing for multiples early dialogs and other received Responses can done within a B2BUA. The possible roles of a B2BUA is described in the taxonomy document RFC7029 [RFC7029] The role of the B2BUA is a Signaling/Media Plane B2BUA Role. Thus many possible use cases which will be possible looking on the used features are considered in this document.

It is NOT within the scope of this document to deploy new SIP procedures but it is within the scope to use existing SIP procedures to describe the proper multiplexing.

2. Consideration of RFC’s on SIP signalling procedures under consideration for forking use cases
2.1. Overview

The following sections show the documentation for forking in different RFCs and what is missing to apply a correlation of multiples early dialogs. Also what procedures should apply for a correct handling of multiples early dialogs.

2.2. RFC3261 Session Initiation Protocol

SIP defined in RFC3261 [RFC3261] describes how forking should apply. In Section 10 of RFC3261 [RFC3261] it is defined that registration creates bindings in a location service for a particular domain that associates an address-of-record URI with one or more contact addresses. Thus when receiving an INVITE with a Request-URI that matches the address-of-record, the proxy will forward the request to the contact addresses registered to that address-of-record. Proxy behavior in RFC3261 [RFC3261] section 16.6 describes that a stateful proxy MAY choose to "fork" a request, routing it to multiple destinations. Any request that is forwarded to more than one location MUST be handled statefully. Forking apply serial or parallel. Thus the forking proxy is either waiting for a final response or a timeout before sending the INVITE to the next contact or sent the INVITE parallel to the contacts registered for one address of record. Considering the forking proxy when the INVITE was sent one or multiple provisional responses may arrive before one or more final responses are received. As RFC3261 [RFC3261] describes these provisional responses may create "early dialogs" and will be forwarded to the UAC.

This concludes that receiving provisional responses may either create early dialogs or will be ignored by the UAC.

The UAC procedures allow receiving one or more provisional responses. The description how to be have exactly is not given by RFC3261 [RFC3261]. Under Section 12.1 "Creation of a Dialog" describes that early dialogs will be established by non final 101-199 responses. No more hints given within this section (e.g. 12.2.1.2 Processing the Responses) how to behave when receiving multiples provisional responses. The termination of early dialogs is clear described that if the dialog is terminated (with BYE) all early dialogs are also terminated. The termination of early dialogs will also be processed when receiving a final non 200 OK response.

For session creation the description of response handling to an sent INVITE is more precise. Within section 13.1 it is stated that a UA will open multiples dialogs when receiving multiples 200OK to an forked INVITE which are then within the same call. Within section 13.2.2.1 lxx Responses it is stated that Provisional responses for an
INVITE request can create "early dialogs". Early dialogs are only needed for sending requests to its peer within the dialog before the initial INVITE transaction completes. This imply that if the UAC does not support the creation of "early dialogs" that provisional responses will be ignored by the UAC. Provisional responses may contain the same exact SDP answer sent prior to the answer within the final 2xx response. Editors Note: It is not stated in RFC3261 what happen in cases of early dialogs arriving with different SDP and also receiving RTP for that dialogs. i.e. more information will deliver RFC3960 "Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP)"

Note that it is not clearly stated what happen if the SDP is different between provisional response and final answer. It is assumed that such information will be ignored by the UA.

The conclusion is that rules for UAC on responses in general and merged requests are defined. For the normal RFC3261 [RFC3261] forking use case only final Responses (200 OK) will be considered to create a dialog/session. This also apply when provisional responses due to other services (e.g. 181 in case of forwarding) will arrive at the UAC.

A couple of rules with regard to responses when forwarding it to the UAC apply to the forking proxy. 1xx and 2xx response will be forwarded immediately. For 6xx Responses the proxy SHOULD cancel all client pending transactions. The Response-Context as defined in RFC3261 [RFC3261] Section 16.7 Response Proceeding" does describe how the forking proxy should behave when receiving responses from different branches. According to the rules in this section the proxy will hold the received non 2xx final responses until for all INVITE transactions a final response is received. The Forking proxy has to choose a final response. One free chosen out of of the lowest response class stored. Preference to that response giving additional information affecting resubmission (such as 401, 407, 415, 420, and 484 if the 4xx class is chosen) A 503 should be changed to 500.

Thus with regard to non 2xx and 1xx final responses the forking proxy has to aggregate and act as central element

Conclusion is that RFC3261 [RFC3261] describes the possibility of forking a INVITE, the forking proxy behavior and the UAC behavior to either ignore all provisional responses or open multiples early dialogs and accept one or more final responses.
2.3. **RFC3960 Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP)**

As for early media and Forking RFC3960 [RFC3960] describes two models for providing early media. There are the gateway model and the Application Server model described. Both have their constrains with forking. The gateway model will present all early media streams to the user in cases where the UAC will not choose one media stream it will confuse the human hearing this mix of media streams. Also bandwidth restrictions will end in bad quality.

The application Server model is based on an early-session disposition type which will not be supported by every UAC.

This RFC does not describe the possibility of multiplexing multiples early media streams to one. Or a choosing mechanism for UAC which of the early dialog should be used to play the announcement.

Since this RFC does not consider B2BUA with media awareness as defined in RFC7029 [RFC7029] some possible functionality is missing where media can be correlated.

2.4. **RFC3262 Reliability of provisional responses**

The RFC describing the reliability of provisional Responses RFC3262 [RFC3262] does not describe directly interactions with forking. It defines the PRACK method and describes the procedures to make a provisional response reliable. Thus the SDP of a 200OK is not longer needed.

The important statement is that user agents that support this 100rel MUST support all offer/answer exchanges that are possible based on the rules in Section 13.2 of RFC3261 [RFC3261], based on the existence of INVITE and PRACK as requests, and 2xx and reliable 1xx as non-failure reliable responses. Thus with forking the UAC has to make each received provisional Response reliable. Specific procedures for a forking proxy does not exist since it has only to pass the responses.

2.5. **RFC3312 Integration of Resource Management and Session Initiation Protocol (SIP)**

RFC3313 [RFC3313] describing the precondition mechanism is not mentioning any interactions with forking relevant issues.

Since RFC3262 [RFC3262] is a precondition to apply to the procedures of RFC3313 [RFC3313]. Following that the UAC has to make the resource reservation with each forking branch where the provisional
response is stating 100rel required. The UAC will have to implement complex procedures to make the resource reservation to all received SDP. And it could lead to the same problems as stated for the early media use case. Also network procedures may be influences. I.e B2BUA with media awareness as defined in RFC7029 [RFC7029] have to reserve resources and have to proceed correctly like in releasing branches where capacity restrictions will apply within the media reservation functionality.

2.6. RFC3841 Caller Preferences for the Session Initiation Protocol (SIP)

RFC3841 [RFC3841] describes extensions to the Session Initiation Protocol (SIP) which allow a caller (UAC) to express preferences about request handling in servers.

One of the caller preferences defined in RFC3841 [RFC3841]. is a method to signal the "fork-directive" to indicate that the UAC does not wish that SIP proxy will fork the INVITE request. This directive is an optional SIP feature the forking proxy is not forced to apply to the directive sent by the UAC. Also not each network element providing forking has implemented this directive.

The parallel-directive does indicate how a SIP proxy may fork the request. Either "parallel" or "sequential".

A B2BUA with media awareness as defined in RFC7029 [RFC7029] may use this directive to avoid multiples provisional responses sent back due to forking. But as said it will not give security for avoiding forking. Thia may a tool for interconnection between service providers.

2.7. RFC5393 Addressing an Amplification Vulnerability in Session Initiation Protocol (SIP) Forking Proxies

To avoid too many forking (possible early Dialogs) RFC5393 [RFC5393] defines the Max-Breadth header to avoid to many forked Requests caused by forking proxies. But there is no effect on correlating the responses. This helps to reduce a cascading of multiples forking in the forward path. The number in the header gives the maximum branches (parallel possible early dialogs) of a forked request. Exceeding the maximum will result in error responses 440.

A Max-Breadth of 1 restricts a request to pure serial forking rather than restricting it from being forked at all.

RFC5393 [RFC5393] specifies normative changes to the SIP protocol. And it mandates if a SIP proxy receives a request with no Max-Breadth
header field value, it MUST add one, with a value that is RECOMMENDED to be 60. Seen from theory a parallel forking could be avoided but nevertheless not all SIP networks will have implemented this RFC and will understand these extensions.

2.8. RFC6228 Session Initiation Protocol (SIP) Response Code for Indication of Terminated Dialog

RFC6228 [RFC6228] defines a new response code to close early dialogs proper. In case where a forking proxy realizes that a 200 OK has been processed the proxy can sent 199 responses to the other open dialogs. This helps in case of correlation when early dialogs has been sent till the end user.

In consideration with the rules defined in RFC3261 for forking proxy a received non 2xx final to an initial dialog initiation request that it recognizes as terminating one or more early dialogs associated with the request. The forking proxy generates normally (e.g. non 100rel, no final response sent) 199 response upstream for each of the terminated early dialogs.

2.9. RFC3326 The Reason Header Field for the Session Initiation Protocol (SIP)

RFC3326 [RFC3326] defines the Reason header and describes one use where the INVITE is forked and results in a rejection, the error response may never be forwarded to the client unless all the other branches also reject the request. This problem is known as the "Heterogeneous Error Response Forking Problem", or HERFP. It is foreseen that a solution to this problem may involve encapsulating the final error response in a provisional response. The Reason header field is a candidate to be used for such encapsulation. In this case the forking proxy will release the dialogs.

This will help to analyze problems with forking but will

2.10. Conclusions

All above mentioned RFCs and procedures describes a piece of the whole picture how forking apply and what procedures are useable for such cases. It is also fact that UA’s and B2BUA’s (e.g. PSTN GW) are existing that will not support multiples early dialogs. Also the support of caller preferences is not secured or implemented by UAC and also SIP forking proxies. Service providers will provide forking independent what the source will support or not. Thus the only solution seen is to describe procedures which apply in B2BUA to support an correlation of multiples early dialogs and other received 18x Responses.
3. Requirements on forking in SIP networks interconnecting with other SIP networks

To improve interoperability with devices which do not support forking, a service provider shall have the possibility to use a B2BUA to multiplex multiple downstream dialogs into a single dialog toward the caller.

The B2BUA providing such possibility must have to understand where the SIP dialog request is coming from and if this originating network or entity or UA can or cannot support multiple responses. This could be also assumed by Service Level Agreements (SLA) between the interconnection partners.

The main requirement is to have an entity that can receive multiple responses based on forked INVITES which now can be correlated to one single dialog towards the originating entity.

To act proactive the B2BUA should also be possible to include and handle preferences (e.g. non forking wished) based on the originating and terminating network.

4. Forking use cases

4.1. Normal Forking use case

SIP defined in RFC3261 [RFC3261] describe how forking should apply. Also rules for UA for responses and merged requests are defined. Since the provisional response is not final there should be no influence on the call stated due to media. An SDP sent in a 18x must be the same as for the final response. The numbers in brackets shows the INVITES/early dialogs created.
As you can see, this figure shows the normal forking case where each UA sends a 18x either with or without SDP. UAS_4 sends a final response and the forking proxy has to cancel all other open provisional responses. The 487 is sent back to the forking proxy on each early dialog (2) and (3) created in the UAS.

Further possible scenarios are that two or three UAS will answer the call with 200 in time so that the UAC has now three open sessions which is not really the goal when a user would like to communicate with only one person.
Figure 1 shows an normal example of forking. With receiving 18x the UAC has to open transaction. So only UAA_4 answers the dialog correctly. It is task of the forking proxy to cancel the remaining open early dialogs

4.2. Multiples provisional responses without SDP

This section describes the use case where multiples 18x responses are sent back which doesn’t contain any SDP. This use case appear when the UAS instances where the INVITE is forked without any specific requirements and support of specific extensions like 100rel

In this specific case the B2BUA has not to anchor the media and acts only as signalling B2BUA and has to maintain the signalling legs
UAC  B2BUA  Forking Proxy  UAS_2  UAS_3  UAS_4

- INVITE(1) F1→  - INVITE F2 -->  - INVITE F3(2)→
- INVITE F4(3)------->
- INVITE F5(4)-------->
<-- 18x (2) F6--

<-- 18x (1) F8-- <-- 18x (2) F7 --
<-- 18x (3) F9------>
<-- 18x (4) F11-------->

<-- 18x (3) F10--
<-- 18x (4) F12--

<-- 200 SDP (4)F13-------->

<-- 200 (4) F14--
<-- 200 (1) F15--

- CANCEL F16 -->
<-- 200 (2) F17 -
<-- 487 (2) F18 -
-- ACK (2) F19--

- CANCEL (3) F20 ---->
<----200 (3) F21 ----
<---- 487 (3) F22 ----
----- ACK (3) F23 ---->

- ACK(1) F24-->

- ACK (4) F26->
--- ACK (4) F27--------->

Figure 2: Example Call Flow

The B2BUA will maintain a Dialog between UAC and the incoming part of
the B2BUA (UAS) And acts as UAC towards the terminating network
(UAS_2, UAS_3 and UAS_4). The INVITE F1 will have another Call-Id as
INVITE F2, also the to-tags are different.

The first 18x (F7) will be passed towards the UAC. The tags (to-tag, from-tag), call-id etc will be stored by the B2BUA. The 18x sent
towards the UAC will contain the to-tag, Call-Id of INVITE F1 and the
from-tag is generated by the B2BUA. With arriving further 18x (F10,
F12) the B2BUA has to store the status of the to-tags etc and will
not forward the 180 to the UA.
With arriving the 200OK (F14) at the B2BUA with the SDP sent back form UAS_4 the B2BUA will sent a 200 OK towards the UAC. In this case the B2BUA re-written the to-tag, from-tan and call-id as already done or response 18x F7.

As for normal forking the forking proxy is responsible for canceling the open dialogs with the CANCEL F16 and F20. The CANCEL will be initiated with receiving/sending the final 200 OK response And ACK F24 finalizes the call initiation.

The B2BUA has normally not to anchor signalling plane i.e as signalling B2BUA. Such option may be applied when sending the INVITE F2 towards the UAS. In cases where the received INVITE has no tags included which indicate the possibility of 100 rel, preconditions or early media the B2BUA may handle this specific call as a stateless proxy.

4.3. Forking use case with provisional responses with SDP using 100rel

This section describes use case where 100rel will be used. This apply in networks where announcements will be played reliability. The mechanism for making the provisional responses reliable is described in RFC3262 [RFC3262]. A B2BUA doing correlation in between allows only one early dialog sent back to the UAC. The B2BUA has to anchor the SIP Dialogs as well as the media reservation streams.

This use case apply where multiples 18x responses are sent back with different SDP content. This use case appear when the UAS instances where the INVITE is forked to will use different codecs. E.G one UAS is a video phone answering with a video codec the other one a mobile phone and a further one using a DECT entity.

And one or more UAS will require reliability of provisional responses. Please Note that such a behavior could be also caused by an application server playing specific announcements which acts on behalf of the UAS.

There is the possibility based on the request if the 100rel mechanism will only be used between B2BUA and UAS or really end to end. In case where the 100rel is stated as supported it is not mandatory to use it. When the UAS want to have the 18x reliable it will set the 100rel into the require header field. In that case where a 18x is sent back with a 100rel required then the B2BUA may play the role of anchoring the media and apply the 100rel between B2BUA and UAS and let the UAC to B2BUA connection as unreliable.
Please note that this is only needed in cases where the media anchoring has to do some manipulation of the media e.g. transcoding of codecs.

Where the B2BUA decides to pass the required 100rel header field the UAC will send then the PRACK and waits for the 200 OK. In case there are further 18x with equal other type of SDP arriving at the B2BUA. B2BUA has to keep handle the 100rel and sent the PRACK to the UAS.

Preconditions: INVITE 100rel supported is set and in 18x a SDP with 100rel required is sent back.

<table>
<thead>
<tr>
<th>UAC</th>
<th>B2BUA</th>
<th>Forking Proxy</th>
<th>UAS_2</th>
<th>UAS_3</th>
<th>UAS_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>- INVITE F1--&gt;</td>
<td>- INVITE F2 --&gt;</td>
<td>- INVITE F3(2)--&gt;</td>
<td>- INVITE F4(3)--&gt;</td>
<td>- INVITE F5(4)--&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;-18x SDP(2)F7-</td>
<td>- PRACK(2)F10--&gt;</td>
<td>- PRACK(2) F10--&gt;</td>
<td>- 200 (2)F12--</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 200(1) F13-</td>
<td>&lt;- 200 (2)F12--</td>
<td>&lt;- 18x (3) F14--------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;- 18x (3)F15--</td>
<td>- PRACK (2)F16--&gt;</td>
<td>- PRACK (3) F17 -------&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;- 200 (3)F19--</td>
<td>&lt;=200 (3) F18 ------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;- 18x (4)F20----------</td>
<td>- PRACK(4)F22--&gt;</td>
<td>- PRACK(4) F23 ------------&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;- 200 (4)F25-</td>
<td>&lt;=200(4) F24 ------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;=200SDP(4)F27-</td>
<td>- 200 SDP (4)F26--------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;-UPDATE(1) F28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 3: Example Call Flow with 100rel

- 200 (1) F29->
<- 200(1) F30-
- CANCEL (2) F31->
---200 (2) F32--
---487 (2) F33--
---ACK (2) F34->
- CANCEL (3) F35 ----> 
------200 (3) F46 ------
------487 (3) F37 ------
------ACK (3) F38 ------
- ACK(2) F39->
- ACK (4) F40->
--- ACK (4) F41---------->

Call will be forked to 3 end devices UAS_2, UAS_3 and UAS_4. UAS_2 answers with 18x (F6-F7) containing a SDP and 100rel required. 18x (F8) is passed to the UAC and made reliable with PRACK (F9-F13)
Further 18x (F15/F21) arrive at the B2BUA which are the answers from UAS_3 and UAS_4. The B2BUA stores the to tag for the call context. The B2BUA will answer the 18x (F15/F21) and made it reliable with PRACK (F16-F19/F22-F24). The B2BUA does not need any media awareness for this procedures as long there is no need for transcoding or other manipulation of media.

With arriving of a 200 OK (F27) at the B2BUA from UAS_4 the B2BUA has to construct first an UPDATE (F28) in the case that the SDP differs from the last 18x (F8) sent to the UAC. The 200 OK (F26) received by the B2BUA and triggers the final response to the UAC (F27). With this method the problem could be that media clipping may appear.

Also all other open Call legs are canceled (F31-44) by normal forking proxy procedures as described in RFC3261 [RFC3261].

A possibility to avoid media clipping is that the first 18x (F8) sent back has to contain all SDP possibilities requested within the INVITE. This has the advantage that the UAS_4 can already start with sending back RTP with any of the codec indicated. But it needs also that the B2BUA has to sent immediate the UPDATE to restrict the codecs used by UAC. Because in case the UAC starts to send RTP with a codec not understood by the UAS the B2BUA has to be either media-aware to transcode or discard the RTP sent by the UAC which also will result in a media clipping.

4.4. Forking use case with provisional responses with SDP using 100rel and preconditions

This section describes use case where preconditions will be used. This apply in mobile networks where resource reservation is needed. Also normal networks may use preconditions when access bandwidth is problematic or requested by end devices calling from mobile networks. The precondition mechanism in RFC3312 [RFC3312] shows the needed additional procedures. A B2BUA doing correlation in between allows only one early dialog sent back to the UAC. The B2BUA has to anchor the SIP Dialogs as well as the media reservation streams.

This use case apply where multiples 18x responses are sent back with different SDP content. This use case appear when the UAS instances where the INVITE is forked to will use different codecs. E.G one UAS is a video phone answering with a video codec the other one a mobile phone and a further one using a DECT entity.

With applying preconditions the resource reservation needs to be finalized before 200 OK is sent. In this scenario where the SIP call is forked and the first UAS answers with a 183 containing 100rel and preconditions requires an the correct SDP answer the UAC, and UAS
starts their resource reservation mechanism (F5-F18). The B2BUA acts as signalling and media-aware functionality between the Forking Server and the UAC.

When the UAS_3 will send back also an 183 containing SDP and their required header is set to 100rel and preconditions the B2BUA has to reserve the resources towards the UAS (F30-F37). This use case assumes that the leg between the UAC and B2BUA will not be updated to avoid to many codec renegotiation and resource reservation. Thus the B2BUA will renegotiate when the final 200 OK will arrive at the B2BUA.
Figure 4: Example Call Flow with preconditions
First Answer of UAS_2 is forwarded to UAC. This is one option. Another possibility would be to wait some time and decide based on content of the received provisional response (e.g. Codec or indication for early session) to forward the 18x of either UAS_2 or UAS_3. UAC reserves the resources and sent the regarding SDP in offer A2. Then UAS_2 answers when the resources are reserved too with answer A2.

UAS_3 send the 200OK so the UAC needs now the SDP from UAS_3. Since the last SDP was made reliable the B2BUA sends an UPDATE towards the UAC with the regarding codec.

Where the B2BUA decides to pass the required 100rel the UAC will send then the PRACK and waits for the 200 OK. In case there are further 18x with other type of SDP arriving at the B2BUA the UAC needs to be informed about change of codec. The UAC has again to sent the PRACK.
Editor’s Note: Question is if the above described mechanism would be a proper mechanism since the later renegotiation + resource reservation could cause media clipping.

4.5. Forking use case with early media played

This use case describes the case where early media is played due to application server actions. e.g. playing a ring tone or a specific announcement sent back.

<table>
<thead>
<tr>
<th>UAC</th>
<th>B2BUA</th>
<th>Forking Proxy</th>
<th>AS_2</th>
<th>AS_3</th>
<th>UAS_4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- INVITE F1---&gt;</td>
<td>- INVITE F2 --</td>
<td>-- INVITE F3(2)---&gt;</td>
<td>- INVITE F4(3)---&gt;</td>
<td>- INVITE F5(4)---&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;-18x SDP (2) F7--</td>
<td>&lt;-18x SDP (2) F6--</td>
<td>&lt;-18x SDP (2) F6--</td>
</tr>
<tr>
<td></td>
<td>&lt;-200 (1) F13-</td>
<td>&lt;-200 (1) F13-</td>
<td>&lt;-200 (2) F12--</td>
<td>&lt;-200 (2) F12--</td>
<td>&lt;-200 (2) F12--</td>
</tr>
<tr>
<td>&lt;========= &lt;= early media =========&gt;</td>
<td>&lt;========= &lt;= early media =========&gt;</td>
<td>&lt;========= &lt;= early media =========</td>
<td>&lt;========= &lt;= early media =========</td>
<td>&lt;========= &lt;= early media =========</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;-18x SDP (3) F14---</td>
<td>&lt;-18x SDP (3) F14---</td>
<td>&lt;-18x SDP (3) F14---</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;-200 (3) F18 ------</td>
<td>&lt;-200 (3) F18 ------</td>
<td>&lt;-200 (3) F18 ------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;-200 (4) F24 -------------</td>
<td>&lt;-200 (4) F24 -------------</td>
<td>&lt;-200 (4) F24 -------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;-200 SDP (4) F26----------</td>
<td>&lt;-200 SDP (4) F26----------</td>
<td>&lt;-200 SDP (4) F26----------</td>
</tr>
</tbody>
</table>
Figure 5: Example Call Flow with early media
AS_2 plays early media towards the UAC. There could be three possibilities.

1. directly switch the media through and discard any other media played.
2. wait for further answers and decide on local policy within the B2BUA.
3. cancel first early session and establish second early session.

The first one is assumed and shown.

RTP sent by the AS_2 will be not relayed by the B2BUA. In case this is local policy. Even in case the SDP itself allows it.

No early media is played. This reflects only an early dialog made reliable towards the B2BUA.

Early media can be indicated in using the the P-Early-Media as defined in RFC3312 [RFC3312].
4.6. Multiples early dialogs and announcements due to call forwarding

This use case describes the case where the SIP INVITE is forwarded and early media is played. This is a widely used service to play an announcement (e.g. "please wait your call is forwarded") when the call is forwarded due to the normal Service behavior.

Further this could also be happen in cases where you have waiting lines for a customer service center. INVITE F2 is forwarded by the forwarding AS to UAS_2. The forwarding AS sends back a 181 (Call Is Being Forwarded)F4. F5 181 which is received by the UAC is made reliable with PRACK F5/200 (OK) PRACK.
Figure 7: Example Call Flow Call Forwarding
F1: INVITE      SDP offer A1  (Codec A, Codec B)Dialog 1
F3: INVITE      SDP offer A2  (Codec A, Codec B)Dialog 2
F4: 181         SDP answer A2  (Codec A) Dialog 2
F5: 181         SDP answer A1  (Codec A) Dialog 1
F6: PRACK           no SDP Dialog 1
F7: PRACK           no SDP Dialog 2
F12: 200 OK(PRA) Dialog 2
F13: 200 OK(PRA) Dialog 1

Forwarding AS plays early media towards the UAC.

F10: 18x         SDP answer A3  (Codec B) Dialog 3
F12: PRACK           no SDP Dialog 3
F15: 200 OK(PRA) Dialog 3
F16: 200 OK(INV) same SDP answer as in F10 Dialog 3

After receiving the final answer the B2BUA updates the SDP between the UAC and B2BUA.

F18: UPDATE      SDP offer A4  (Codec B) Dialog 1
F19: 200 OK(UPD) SDP answer A4  (Codec B) Dialog 1
F20: 200 OK(INV) SDP Answer A4  (Codec B) Dialog 1

To indicate early media the P-Early-Media header could be used which is defined in RFC3312.

4.7. Methods to avoid forking

This section describes how the originating side could avoid that a SIP INVITE will be forked or at least avoid complex situations where forking could lead to complicated SIP call flows.

Using the methods of RFC3841 [RFC3841] which are the caller preferences may result that a forking proxy will not fork the INVITE. The caller preferences are not mandatory to executed by the forking proxy. But if supported the "no-fork" directive will avoid a forking

Using the Max-Breadth header defined in RFC5393 [RFC5393] avoids to many forked Requests caused by forking proxies. But there is no effect on correlating the responses. This helps to reduce a cascading of multiples forking in the forward path. The setting of the of Max-Breadth to 1 restricts a request to pure serial forking rather than restricting it from being forked at all. But this will help to avoid multiples provisional responses due to forking. But will not avoid provisional responses due to call forwarding. RFC5393 [RFC5393] specifies normative changes to the SIP protocol but this will only apply when this RFC is implemented properly by forking

Jesske                  Expires September 4, 2015              [Page 27]
proxies and the originating SIP entities within the path. Because each SIP element may include the Max-Breadth header.

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. Security Considerations

Currently no further security considerations are needed beyond considerations made in the referred RFC’s for SIP [RFC3261], reliability of provisional responses [RFC3262] and resource management [RFC3312].

7. Acknowledgments

The author like to thank Paul Kyzivat for his extensive review and comments on the first draft version.

8. References

8.1. Normative References


8.2. Informative References

[TS24.229] 3GPP, "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3", March 2014.

Appendix A. Appendix

Author’s Address
Roland Jesske
Deutsche Telekom
Heinrich-Hertz-Strasse 3-7
Darmstadt  64307
Germany

Phone: +4961515812766
Email: r.jesske@telekom.de