Transcoding Services Invocation in
the Session Initiation Protocol (SIP)
Using Third Party Call Control (3pcc)

Status of This Memo

Copyright Notice

Abstract

   This document describes how to invoke transcoding services using
   Session Initiation Protocol (SIP) and third party call control. This
   way of invocation meets the requirements for SIP regarding
   transcoding services invocation to support deaf, hard of hearing and
   speech-impaired individuals.

Table of Contents

1. Introduction

   The framework for transcoding with SIP [4] describes how two SIP [1]
   UAs (User Agents) can discover incompatibilities that prevent them
   from establishing a session (e.g., lack of support for a common codec
   or common media type). When such incompatibilities are found, the
   UAs need to invoke transcoding services to successfully establish the
   session. 3pcc (third party call control) [2] is one way to perform
   such invocation.

2. General Overview

   In the 3pcc model for transcoding invocation, a transcoding server
   that provides a particular transcoding service (e.g., speech-to-text)
   is identified by a URI. A UA that wishes to invoke that service
   sends an INVITE request to that URI establishing a number of media
   streams. The way the transcoder manipulates and manages the contents
   of those media streams (e.g., the text received over the text stream
   is transformed into speech and sent over the audio stream) is service
   specific.

   All the call flows in this document use SDP. The same call flows
   could be used with another session description protocol that provides
   similar session description capabilities.

3. Third Party Call Control Flows

   Given two UAs (A and B) and a transcoding server (T), the invocation
   of a transcoding service consists of establishing two sessions; A-T
   and T-B. How these sessions are established depends on which party,
   the caller (A) or the callee (B), invokes the transcoding services.
   Section 3.2 deals with callee invocation and Section 3.3 deals with
   caller invocation.

   In all our 3pcc flows we have followed the general principle that a
   200 (OK) response from the transcoding service has to be received
   before contacting the callee. This tries to ensure that the
   transcoding service will be available when the callee accepts the
   session.

   Still, the transcoding service does not know the exact type of
   transcoding it will be performing until the callee accepts the
   session. So, there is always the chance of failing to provide
   transcoding services after the callee has accepted the session. A
   system with more stringent requirements could use preconditions to
   avoid this situation. When preconditions are used, the callee is not
   alerted until everything is ready for the session.

3.1. Terminology

   All the flows in this document follow the naming convention below:

   SDP A: A session description generated by A. It contains, among
              other things, the transport address/es (IP address and
              port number) where A wants to receive media for each
              particular stream.

   SDP B: A session description generated by B. It contains, among
              other things, the transport address/es where B wants to
              receive media for each particular stream.

   SDP A+B: A session description that contains, among other things,
              the transport address/es where A wants to receive media
              and the transport address/es where B wants to receive
              media.

   SDP TA: A session description generated by T and intended for A.
              It contains, among other things, the transport address/es
              where T wants to receive media from A.

   SDP TB: A session description generated by T and intended for B.
              It contains, among other things, the transport address/es
              where T wants to receive media from B.

   SDP TA+TB: A session description generated by T that contains, among
              other things, the transport address/es where T wants to
              receive media from A and the transport address/es where T
              wants to receive media from B.

3.2. Callee's Invocation

   In this scenario, B receives an INVITE from A, and B decides to
   introduce T in the session. Figure 1 shows the call flow for this
   scenario.

   In Figure 1, A can both hear and speak, and B is a deaf user with a
   speech impairment. A proposes to establish a session that consists
   of an audio stream (1). B wants to send and receive only text, so it
   invokes a transcoding service T that will perform both speech-to-text
   and text-to-speech conversions (2). The session descriptions of
   Figure 1 are partially shown below.

```
          A                      T                      B

          |                      |                      |
          |-------------------(1) INVITE SDP A-------------------->|
          |                      |                      |
          |                      |<---(2) INVITE SDP A+B------|
          |                      |                      |
          |                      |---(3) 200 OK SDP TA+TB---->|
          |                      |                      |
          |                      |<---------(4) ACK-----------|
          |                      |                      |
          |<------------------(5) 200 OK SDP TA--------------------|
          |                      |                      |
          |----------------------(6) ACK------------------------->|
          |                      |                      |
          | ************************ | ************************ |
          |*          MEDIA        *|*          MEDIA        *|
          | ************************ | ************************ |
          |                      |                      |
```
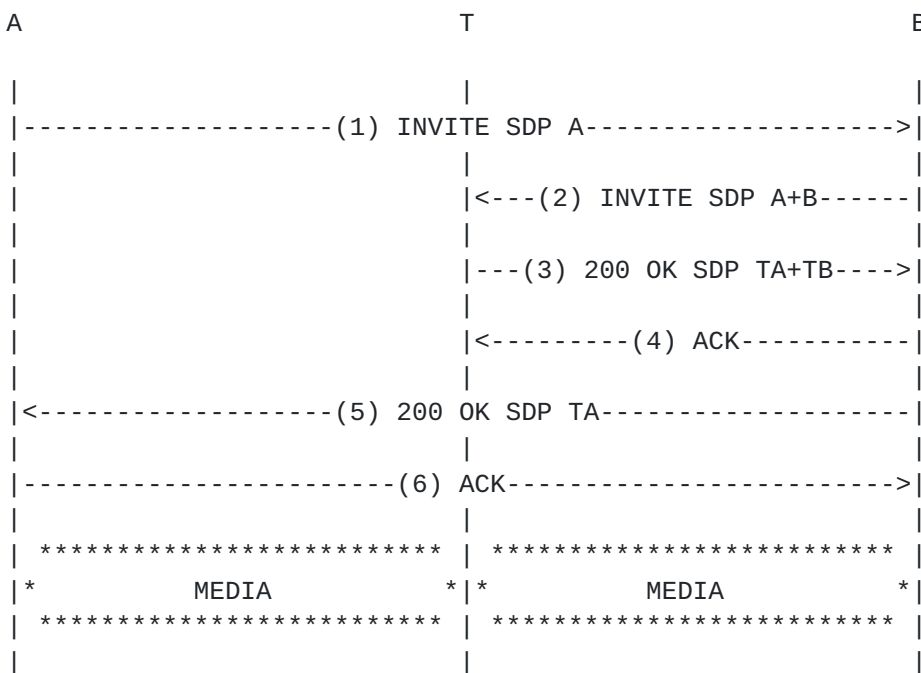
          Figure 1: Callee's Invocation of a Transcoding Service

   (1) INVITE SDP A

           m=audio 20000 RTP/AVP 0
           c=IN IP4 A.example.com

   (2) INVITE SDP A+B

           m=audio 20000 RTP/AVP 0
           c=IN IP4 A.example.com
           m=text 40000 RTP/AVP 96
           c=IN IP4 B.example.com
           a=rtpmap:96 t140/1000

   (3) 200 OK SDP TA+TB

           m=audio 30000 RTP/AVP 0
           c=IN IP4 T.example.com
           m=text 30002 RTP/AVP 96
           c=IN IP4 T.example.com
           a=rtpmap:96 t140/1000

   (5) 200 OK SDP TA

           m=audio 30000 RTP/AVP 0
           c=IN IP4 T.example.com

Four media streams (i.e., two bi-directional streams) have been
established at this point:

> 1. Audio from A to T.example.com:30000
>
> 2. Text from T to B.example.com:40000
>
> 3. Text from B to T.example.com:30002
>
> 4. Audio from T to A.example.com:20000

When either A or B decides to terminate the session, it sends a BYE
indicating that the session is over.

If the first INVITE (1) received by B is empty (no session
description), the call flow is slightly different. Figure 2 shows
the messages involved.

B may have different reasons for invoking T before knowing A's
session description. B may want to hide its lack of native
capabilities, and therefore wants to return a session description
with all the codecs that B supports, plus all the codecs that T
supports. Or T may provide recording services (besides transcoding),
and B wants T to record the conversation, regardless of whether
transcoding is needed.

This scenario (Figure 2) is a bit more complex than the previous one.
In INVITE (2), B still does not have SDP A, so it cannot provide T
with that information. When B finally receives SDP A in (6), it has
to send it to T. B sends an empty INVITE to T (7) and gets a 200 OK
with SDP TA+TB (8). In general, this SDP TA+TB can be different than
the one sent in (3). That is why B needs to send the updated SDP TA
to A in (9). A then sends a possibly updated SDP A (10) and B sends
it to T in (12). On the other hand, if T happens to return the same
SDP TA+TB in (8) as in (3), B can skip messages (9), (10), and (11).
So, implementors of transcoding services are encouraged to return the
same session description in (8) as in (3) in this type of scenario.
The session descriptions of this flow are shown below:

```
    A                          T                          B

    |                          |                          |
    |----------------------(1) INVITE----------------------->|
    |                          |                          |
    |                          |<-----(2) INVITE SDP B------|
    |                          |                          |
    |                          |---(3) 200 OK SDP TA+TB---->|
    |                          |                          |
    |                          |<---------(4) ACK-----------|
    |                          |                          |
    |<-------------------(5) 200 OK SDP TA-------------------|
    |                          |                          |
    |----------------------(6) ACK SDP A-------------------->|
    |                          |                          |
    |                          |<-------(7) INVITE----------|
    |                          |                          |
    |                          |---(8) 200 OK SDP TA+TB---->|
    |                          |                          |
    |<----------------(9) INVITE SDP TA---------------------|
    |                          |                          |
    |-----------------(10) 200 OK SDP A--------------------->|
    |                          |                          |
    |<----------------------(11) ACK------------------------|
    |                          |                          |
    |                          |<-----(12) ACK SDP A+B------|
    |                          |                          |
    | ************************* | ************************* |
    |*         MEDIA          *|*         MEDIA          *|
    | ************************* | ************************* |
```
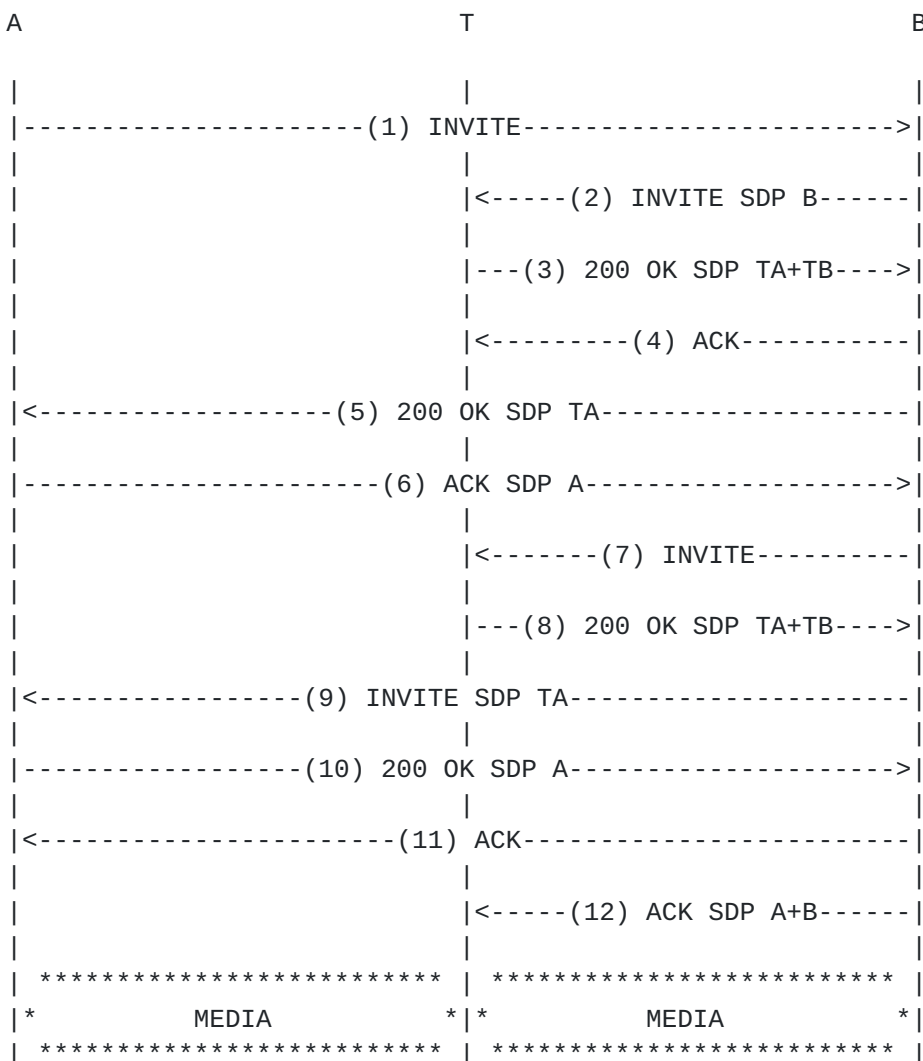
        Figure 2: Callee's invocation after initial INVITE without SDP

   (2) INVITE SDP A+B

           m=audio 20000 RTP/AVP 0
           c=IN IP4 0.0.0.0
           m=text 40000 RTP/AVP 96
           c=IN IP4 B.example.com
           a=rtpmap:96 t140/1000

   (3) 200 OK SDP TA+TB

           m=audio 30000 RTP/AVP 0
           c=IN IP4 T.example.com
           m=text 30002 RTP/AVP 96
           c=IN IP4 T.example.com
           a=rtpmap:96 t140/1000

(5) 200 OK SDP TA

        m=audio 30000 RTP/AVP 0
        c=IN IP4 T.example.com

(6) ACK SDP A

        m=audio 20000 RTP/AVP 0
        c=IN IP4 A.example.com

(8) 200 OK SDP TA+TB

        m=audio 30004 RTP/AVP 0
        c=IN IP4 T.example.com
        m=text 30006 RTP/AVP 96
        c=IN IP4 T.example.com
        a=rtpmap:96 t140/1000

(9) INVITE SDP TA

        m=audio 30004 RTP/AVP 0
        c=IN IP4 T.example.com

(10) 200 OK SDP A

        m=audio 20002 RTP/AVP 0
        c=IN IP4 A.example.com

(12) ACK SDP A+B

        m=audio 20002 RTP/AVP 0
        c=IN IP4 A.example.com
        m=text 40000 RTP/AVP 96
        c=IN IP4 B.example.com
        a=rtpmap:96 t140/1000

   Four media streams (i.e., two bi-directional streams) have been
   established at this point:

        1. Audio from A to T.example.com:30004

        2. Text from T to B.example.com:40000

        3. Text from B to T.example.com:30006

        4. Audio from T to A.example.com:20002

3.3. Caller's Invocation

   In this scenario, A wishes to establish a session with B using a
   transcoding service. A uses 3pcc to set up the session between T and
   B. The call flow we provide here is slightly different than the ones
   in [2]. In [2], the controller establishes a session between two
   user agents, which are the ones deciding the characteristics of the
   streams. Here, A wants to establish a session between T and B, but A
   wants to decide how many and which types of streams are established.
   That is why A sends its session description in the first INVITE (1)
   to T, as opposed to the media-less initial INVITE recommended by [2].
   Figure 3 shows the call flow for this scenario.

   We do not include the session descriptions of this flow, since they
   are very similar to those in Figure 2. In this flow, if T returns
   the same SDP TA+TB in (8) as in (2), messages (9), (10), and (11) can
   be skipped.

3.4. Receiving the Original Stream

   Sometimes, as pointed out in the requirements for SIP in support of
   deaf, hard of hearing, and speech-impaired individuals [5], a user
   wants to receive both the original stream (e.g., audio) and the
   transcoded stream (e.g., the output of the speech-to-text
   conversion). There are various possible solutions for this problem.
   One solution consists of using the SDP group attribute with Flow
   Identification (FID) semantics [3]. FID allows requesting that a
   stream is sent to two different transport addresses in parallel, as
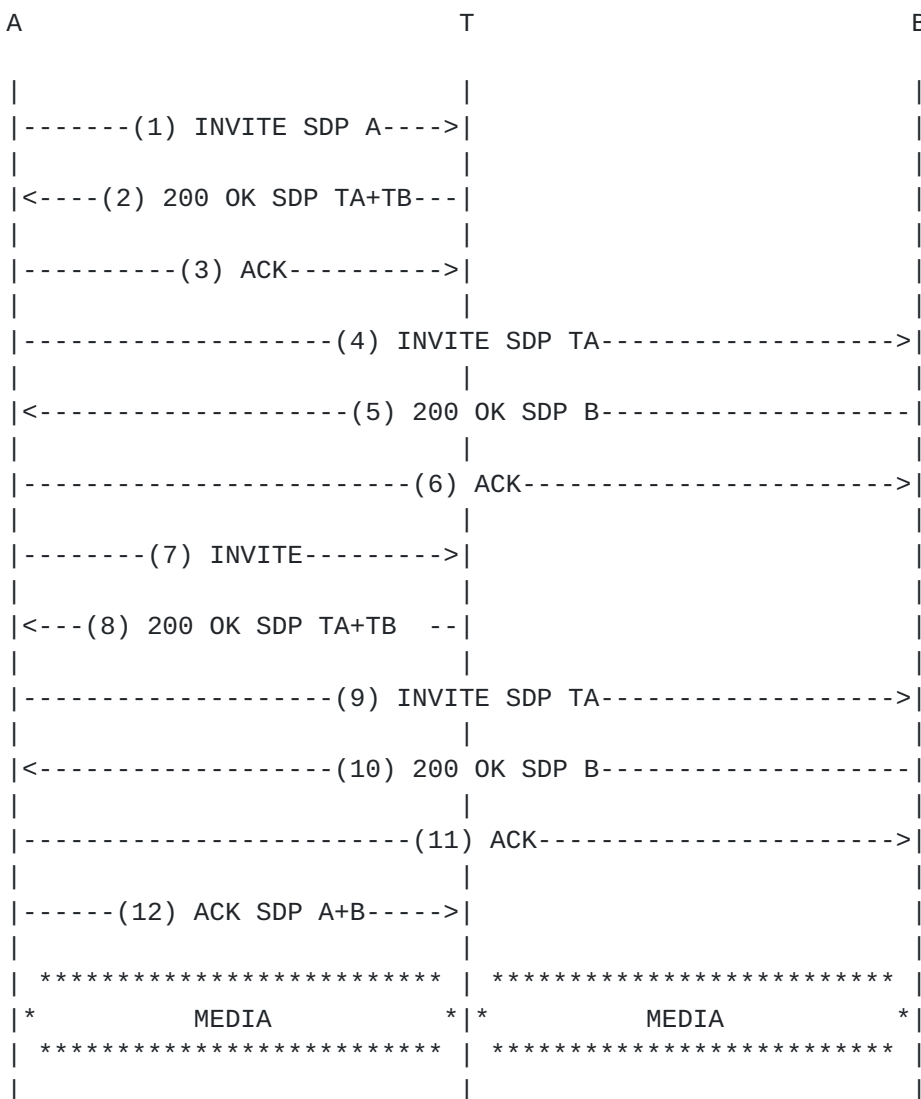   shown below:

```
   A                              T                              B

   |                              |                              |
   |-------(1) INVITE SDP A---->|                              |
   |                              |                              |
   |<----(2) 200 OK SDP TA+TB---|                              |
   |                              |                              |
   |----------(3) ACK---------->|                              |
   |                              |                              |
   |-------------------(4) INVITE SDP TA------------------->|
   |                              |                              |
   |<------------------(5) 200 OK SDP B--------------------|
   |                              |                              |
   |-----------------------(6) ACK------------------------>|
   |                              |                              |
   |--------(7) INVITE--------->|                              |
   |                              |                              |
   |<---(8) 200 OK SDP TA+TB  --|                              |
   |                              |                              |
   |-------------------(9) INVITE SDP TA------------------->|
   |                              |                              |
   |<------------------(10) 200 OK SDP B-------------------|
   |                              |                              |
   |-----------------------(11) ACK----------------------->|
   |                              |                              |
   |------(12) ACK SDP A+B----->|                              |
   |                              |                              |
   | ************************* | ************************* |
   |*          MEDIA          *|*          MEDIA          *|
   | ************************* | ************************* |
   |                              |                              |
```

         Figure 3: Caller's invocation of a transcoding service

          a=group:FID 1 2
          m=audio 20000 RTP/AVP 0
          c=IN IP4 A.example.com
          a=mid:1
          m=audio 30000 RTP/AVP 0
          c=IN IP4 T.example.com
          a=mid:2

   The problem with this solution is that the majority of the SIP user
   agents do not support FID. Moreover, only a small fraction of the
   few UAs that support FID, also support sending simultaneous copies of
   the same media stream at the same time. In addition, FID forces both
   copies of the stream to use the same codec.

Therefore, we recommend that T (instead of a user agent) replicates
the media stream. The transcoder T receiving the following session
description performs speech-to-text and text-to-speech conversions
between the first audio stream and the text stream. In addition, T
copies the first audio stream to the second audio stream and sends it
to A.

```
m=audio 40000 RTP/AVP 0
c=IN IP4 B.example.com
m=audio 20000 RTP/AVP 0
c=IN IP4 A.example.com
a=recvonly
m=text 20002 RTP/AVP 96
c=IN IP4 A.example.com
a=rtpmap:96 t140/1000
```

3.5. Transcoding Services in Parallel

Transcoding services sometimes consist of human relays (e.g., a
person performing speech-to-text and text-to-speech conversions for a
session). If the same person is involved in both conversions (i.e.,
from A to B and from B to A), he or she has access to all of the
conversation. In order to provide some degree of privacy, sometimes
two different persons are allocated to do the job (i.e., one person
handles A->B and the other B->A). This type of disposition is also
useful for automated transcoding services, where one machine converts
text to synthetic speech (text-to-speech) and another performs voice
recognition (speech-to-text).

The scenario described above involves four different sessions: A-T1,
T1-B, B-T2 and T2-A. Figure 4 shows the call flow where A invokes T1
and T2.

Note this example uses unidirectional media streams (i.e., sendonly
or recvonly) to clearly identify which transcoder handles media in
which direction. Nevertheless, nothing precludes the use of
bidirectional streams in this scenario. They could be used, for
example, by a human relay to ask for clarifications (e.g., I did not
get that, could you repeat, please?) to the party he or she is
receiving media from.

(1) INVITE SDP AT1

         m=text 20000 RTP/AVP 96
         c=IN IP4 A.example.com
         a=rtpmap:96 t140/1000
         a=sendonly
         m=audio 20000 RTP/AVP 0
         c=IN IP4 0.0.0.0
         a=recvonly

(2) INVITE SDP AT2

         m=text 20002 RTP/AVP 96
         c=IN IP4 A.example.com
         a=rtpmap:96 t140/1000
         a=recvonly
         m=audio 20000 RTP/AVP 0
         c=IN IP4 0.0.0.0
         a=sendonly

(3) 200 OK SDP T1A+T1B

         m=text 30000 RTP/AVP 96
         c=IN IP4 T1.example.com
         a=rtpmap:96 t140/1000
         a=recvonly
         m=audio 30002 RTP/AVP 0
         c=IN IP4 T1.example.com
         a=sendonly

(5) 200 OK SDP T2A+T2B

         m=text 40000 RTP/AVP 96
         c=IN IP4 T2.example.com
         a=rtpmap:96 t140/1000
         a=sendonly
         m=audio 40002 RTP/AVP 0
         c=IN IP4 T2.example.com
         a=recvonly

(7) INVITE SDP T1B+T2B

         m=audio 30002 RTP/AVP 0
         c=IN IP4 T1.example.com
         a=sendonly
         m=audio 40002 RTP/AVP 0
         c=IN IP4 T2.example.com
         a=recvonly

```
     A                      T1                    T2           B

     |                       |                     |            |
     |----(1) INVITE SDP AT1--->|                  |            |
     |                       |                     |            |
     |----------------(2) INVITE SDP AT2--------------->|       |
     |                       |                     |            |
     |<-(3) 200 OK SDP T1A+T1B--|                  |            |
     |                       |                     |            |
     |---------(4) ACK--------->|                  |            |
     |                       |                     |            |
     |<---------------(5) 200 OK SDP T2A+T2B-----------|        |
     |                       |                     |            |
     |---------------------(6) ACK------------------->|         |
     |                       |                     |            |
     |---------------------(7) INVITE SDP T1B+T2B----------------->|
     |                       |                     |            |
     |<--------------------(8) 200 OK SDP BT1+BT2-----------------|
     |                       |                     |            |
     |------(9) INVITE--------->|                  |            |
     |                       |                     |            |
     |------------------(10) INVITE------------------>|         |
     |                       |                     |            |
     |<-(11) 200 OK SDP T1A+T1B-|                  |            |
     |                       |                     |            |
     |<------------(12) 200 OK SDP T2A+T2B-------------|        |
     |                       |                     |            |
     |------------------(13) INVITE SDP T1B+T2B-------------------->|
     |                       |                     |            |
     |<----------------(14) 200 OK SDP BT1+BT2--------------------|
     |                       |                     |            |
     |------------------------(15) ACK---------------------------->|
     |                       |                     |            |
     |---(16) ACK SDP AT1+BT1-->|                  |            |
     |                       |                     |            |
     |------------(17) ACK SDP AT2+BT2--------------->|         |
     |                       |                     |            |
     | ********************** | ******************************** |
     |*       MEDIA        *|*          MEDIA               *|
     | ********************** | ******************************** |
     |                       |                     |            |
     | ********************************************** ********** |
     |*                      MEDIA                *|* MEDIA  *|
     | ******************************************** | ********** |
     |                       |                     |            |
```
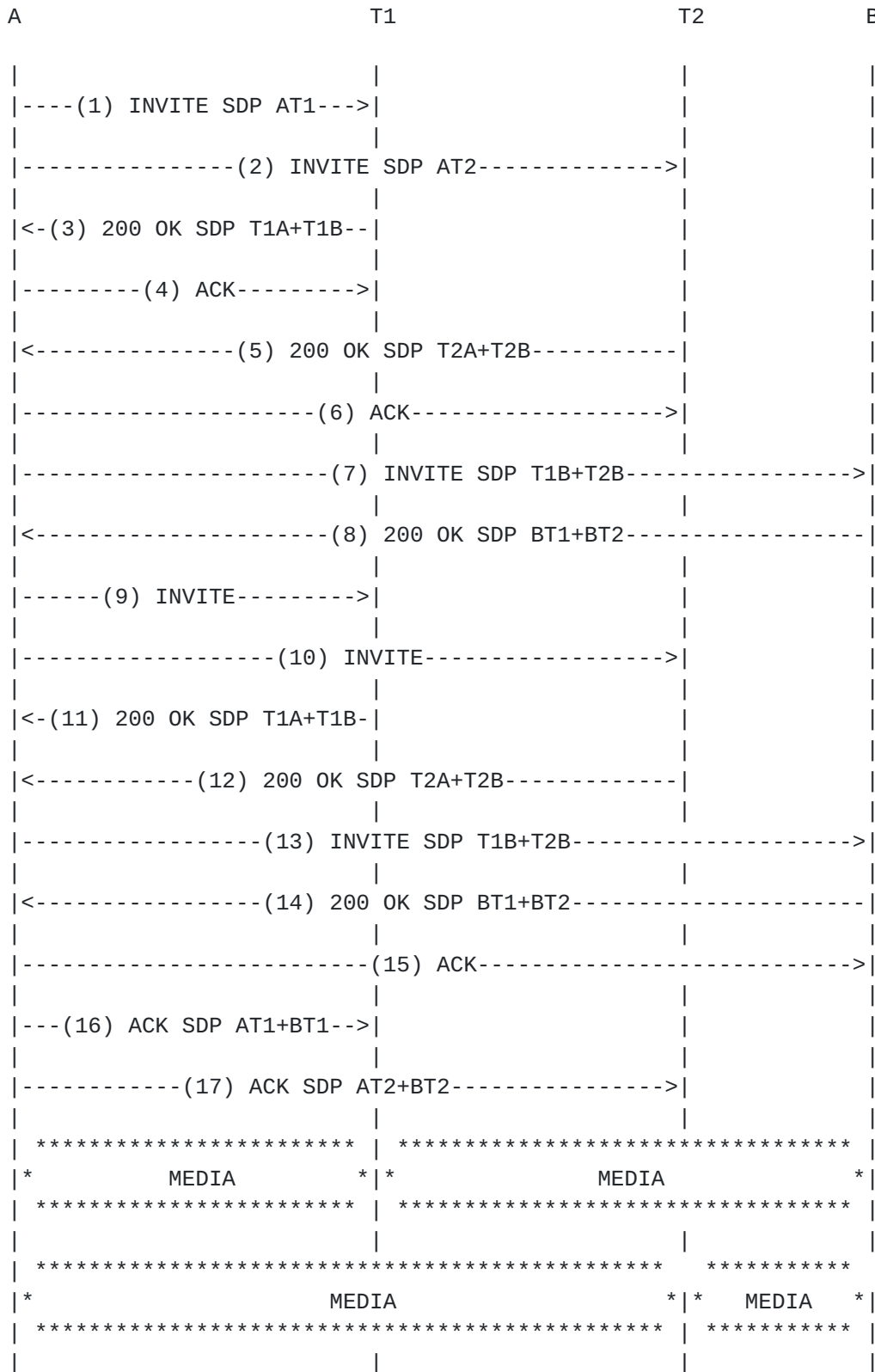
Figure 4: Transcoding services in parallel

(8) 200 OK SDP BT1+BT2

        m=audio 50000 RTP/AVP 0
        c=IN IP4 B.example.com
        a=recvonly
        m=audio 50002 RTP/AVP 0
        c=IN IP4 B.example.com
        a=sendonly

(11) 200 OK SDP T1A+T1B

        m=text 30000 RTP/AVP 96
        c=IN IP4 T1.example.com
        a=rtpmap:96 t140/1000
        a=recvonly
        m=audio 30002 RTP/AVP 0
        c=IN IP4 T1.example.com
        a=sendonly

(12) 200 OK SDP T2A+T2B

        m=text 40000 RTP/AVP 96
        c=IN IP4 T2.example.com
        a=rtpmap:96 t140/1000
        a=sendonly
        m=audio 40002 RTP/AVP 0
        c=IN IP4 T2.example.com
        a=recvonly

Since T1 have returned the same SDP in (11) as in (3), and T2 has
returned the same SDP in (12) as in (5), messages (13), (14) and (15)
can be skipped.

(16) ACK SDP AT1+BT1

        m=text 20000 RTP/AVP 96
        c=IN IP4 A.example.com
        a=rtpmap:96 t140/1000
        a=sendonly
        m=audio 50000 RTP/AVP 0
        c=IN IP4 B.example.com
        a=recvonly

(17) ACK SDP AT2+BT2

            m=text 20002 RTP/AVP 96
            c=IN IP4 A.example.com
            a=rtpmap:96 t140/1000
            a=recvonly
            m=audio 50002 RTP/AVP 0
            c=IN IP4 B.example.com
            a=sendonly

Four media streams have been established at this point:

    1. Text from A to T1.example.com:30000

    2. Audio from T1 to B.example.com:50000

    3. Audio from B to T2.example.com:40002

    4. Text from T2 to A.example.com:20002

Note that B, the user agent server, needs to support two media
streams: sendonly and recvonly. At present, some user agents,
although they support a single sendrecv media stream, do not support
a different media line per direction. Implementers are encouraged to
build support for this feature.

3.6. Multiple Transcoding Services in Series

In a distributed environment, a complex transcoding service (e.g.,
English text to Spanish speech) is often provided by several servers.
For example, one server performs English text to Spanish text
translation, and its output is fed into a server that performs text-
to-speech conversion. The flow in Figure 5 shows how A invokes T1
and T2.

```
A                        T1                    T2                B

|                        |                     |                 |
|----(1) INVITE SDP A----->|                   |                 |
|                        |                     |                 |
|<-(2) 200 OK SDP T1A+T1T2- |                   |                 |
|                        |                     |                 |
|----------(3) ACK-------->|                   |                 |
|                        |                     |                 |
|-----------(4) INVITE SDP T1T2------------------>|              |
|                        |                     |                 |
|<-----------(5) 200 OK SDP T2T1+T2B-------------|              |
|                        |                     |                 |
|--------------------(6) ACK-------------------->|              |
|                        |                     |                 |
|------------------------(7) INVITE SDP T2B---------------->|
|                        |                     |                 |
|<-----------------------(8) 200 OK SDP B-------------------|
|                        |                     |                 |
|------------------------------(9) ACK---------------------->|
|                        |                     |                 |
|---(10) INVITE----------->|                   |                 |
|                        |                     |                 |
|-----------------(11) INVITE------------------->|             |
|                        |                     |                 |
|<-(12) 200 OK SDP T1A+T1T2-|                   |                 |
|                        |                     |                 |
|<-------------(13) 200 OK SDP T2T1+T2B----------|             |
|                        |                     |                 |
|---(14) ACK SDP T1T2+B--->|                   |                 |
|                        |                     |                 |
|----------------------(15) INVITE SDP T2B-------------------->|
|                        |                     |                 |
|<---------------------(16) 200 OK SDP B----------------------|
|                        |                     |                 |
|----------------(17) ACK SDP T1T2+B------------->|           |
|                        |                     |                 |
|------------------------(18) ACK-------------------------->|
|                        |                     |                 |
| *********************** | ****************** *********** |
|*        MEDIA        *|*       MEDIA      *|*  MEDIA   *|
| *********************** | ****************** | *********** |
|                        |                     |                 |
```
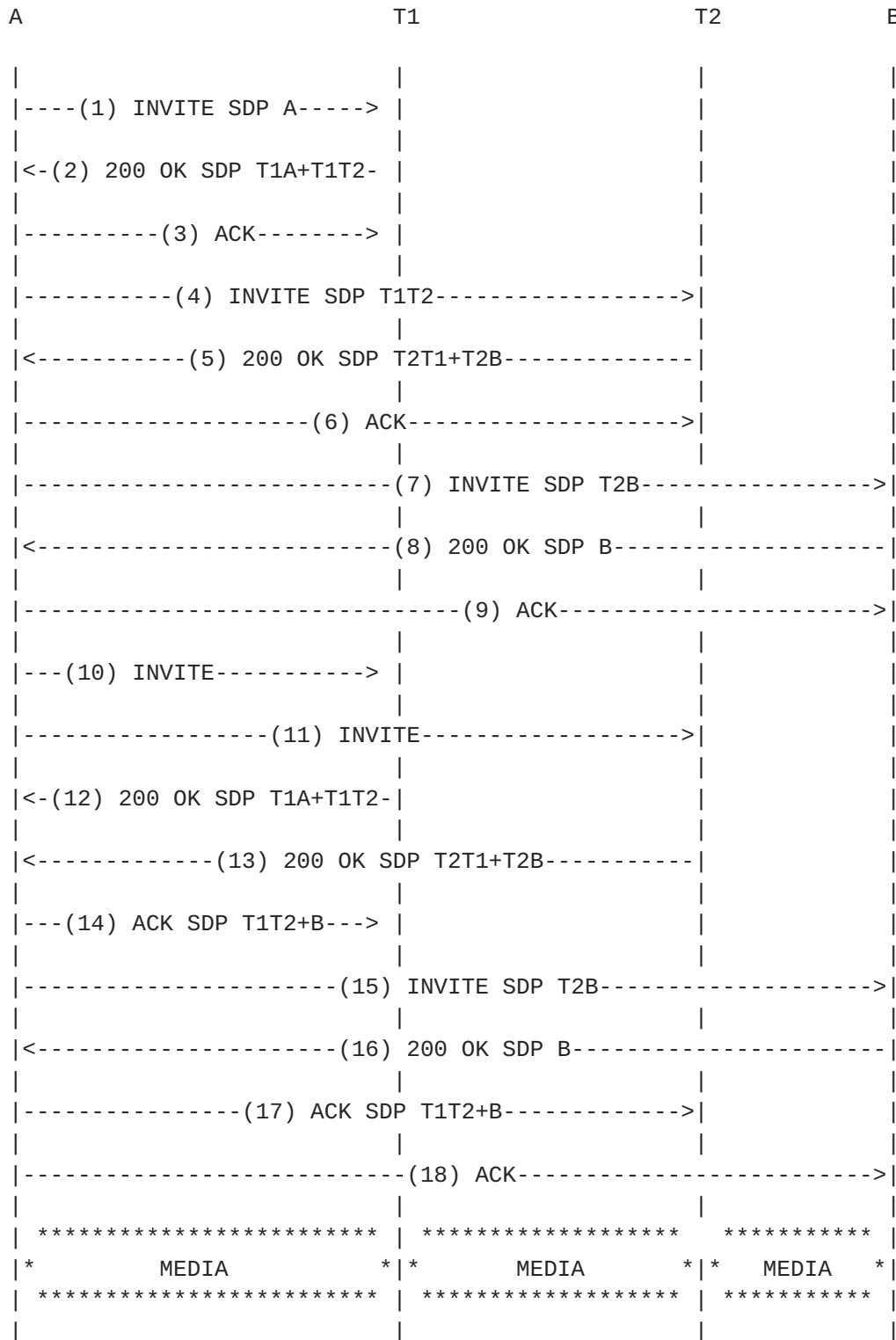
Figure 5: Transcoding services in serial

4. Security Considerations

   RFC 3725 [2] discusses security considerations which relate to the
   use of third party call control in SIP. These considerations apply
   to this document, since it describes how to use third party call
   control to invoke transcoding service.

   In particular, RFC 3725 states that end-to-end media security is
   based on the exchange of keying material within SDP and depends on
   the controller behaving properly. That is, the controller should not
   try to disable the security mechanisms offered by the other parties.
   As a result, it is trivially possible for the controller to insert
   itself as an intermediary on the media exchange, if it should so
   desire.

   In this document, the controller is the UA invoking the transcoder,
   and there is a media session established using third party call
   control between the remote UA and the transcoder. Consequently, the
   attack described in RFC 3725 does not constitute a threat because the
   controller is the UA invoking the transcoding service and it has
   access to the media anyway by definition. So, it seems unlikely that
   a UA would attempt to launch an attack against its own session by
   disabling security between the transcoder and the remote UA.

   Regarding end-to-end media security from the UAs' point of view, the
   transcoder needs access to the media in order to perform its
   function. So, by definition, the transcoder behaves as a man in the
   middle. UAs that do not want a particular transcoder to have access
   to all the media exchanged between them can use a different
   transcoder for each direction. In addition, UAs can use different
   transcoders for different media types.

5. Normative References

[1]   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.

[2]   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.

[3]   Camarillo, G., Eriksson, G., Holler, J., and H. Schulzrinne,
      "Grouping of Media Lines in the Session Description Protocol
      (SDP)", RFC 3388, December 2002.

6. Informative References

[4]   Camarillo, G., "Framework for transcoding with the session
      initiation protocol", August 2003, Work in Progress.

[5]   Charlton, N., Gasson, M., Gybels, G., Spanner, M., and A. van
      Wijk, "User Requirements for the Session Initiation Protocol
      (SIP) in Support of Deaf, Hard of Hearing and Speech-impaired
      Individuals", RFC 3351, August 2002.

Authors' Addresses

    Gonzalo Camarillo
    Ericsson
    Advanced Signalling Research Lab.
    FIN-02420 Jorvas
    Finland

    EMail: Gonzalo.Camarillo@ericsson.com

    Eric Burger
    Brooktrout Technology, Inc.
    18 Keewaydin Way
    Salem, NH 03079
    USA

    EMail: eburger@brooktrout.com

    Henning Schulzrinne
    Dept. of Computer Science
    Columbia University
    1214 Amsterdam Avenue, MC 0401
    New York, NY 10027
    USA

    EMail: schulzrinne@cs.columbia.edu

    Arnoud van Wijk
    Viataal
    Research & Development
    Afdeling RDS
    Theerestraat 42
    5271 GD Sint-Michielsgestel
    The Netherlands

    EMail: a.vwijk@viataal.nl