An Architectural Framework for Media Server Control

Status of This Memo

Copyright Notice

Abstract

   This document describes an architectural framework for Media Server
   control. The primary focus will be to define logical entities that
   exist within the context of Media Server control, and define the
   appropriate naming conventions and interactions between them.

Table of Contents

1. Introduction

   Application Servers host one or more instances of a communications
   application. Media Servers provide real-time media processing
   functions. This document presents the core architectural framework
   to allow Application Servers to control Media Servers. An overview
   of the architecture describing the core logical entities and their
   interactions is presented in Section 3. The requirements for Media
   Server control are defined in [RFC5167].

   The Session Initiation Protocol (SIP) [RFC3261] is used as the
   session establishment protocol within this architecture. Application
   Servers use it both to terminate media streams on Media Servers and
   to create and manage control channels for Media Server control
   between themselves and Media Servers. The detailed model for Media
   Server control together with a description of SIP usage is presented
   in Section 4.

   Several services are described using the framework defined in this
   document. Use cases for Interactive Voice Response (IVR) services
   are described in Section 5, and conferencing use cases are described
   in Section 6.

2. Terminology

   The following terms are defined for use in this document in the
   context of Media Server control:

   Application Server (AS): A functional entity that hosts one or more
      instances of a communication application. The application server
      may include the conference policy server, the focus, and the
      conference notification server, as defined in [RFC4353]. Also, it
      may include communication applications that use IVR or
      announcement services.

   Media Functions: Functions available on a Media Server that are used
      to supply media services to the AS. Some examples are Dual-Tone
      Multi-Frequency (DTMF) detection, mixing, transcoding, playing
      announcement, recording, etc.

   Media Resource Broker (MRB): A logical entity that is responsible
      for both the collection of appropriate published Media Server (MS)
      information and supplying of appropriate MS information to
      consuming entities. The MRB is an optional entity and will be
      discussed in a separate document.

   Media Server (MS): The media server includes the mixer as defined in
      [RFC4353]. The media server plays announcements, it processes
      media streams for functions like DTMF detection and transcoding.
      The media server may also record media streams for supporting IVR
      functions like announcing conference participants. In the
      architecture for the 3GPP IP Multimedia Subsystem (IMS) a Media
      Server is referred to as a Media Resource Function (MRF).

   Media Services: Application service requiring media functions such
      as Interactive Voice Response (IVR) or media conferencing.

   Media Session: From the Session Description Protocol (SDP)
      specification [RFC4566]: "A multimedia session is a set of
      multimedia senders and receivers and the data streams flowing from
      senders to receivers. A multimedia conference is an example of a
      multimedia session."

   MS Control Channel: A reliable transport connection between the AS
      and MS used to exchange MS Control PDUs. Implementations must
      support the Transport Control Protocol (TCP) [RFC0793] and may
      support the Stream Control Transmission Protocol (SCTP) [RFC4960].
      Implementations must support TLS [RFC5246] as a transport-level
      security mechanism although its use in deployments is optional.

MS Control Dialog: A SIP dialog that is used for establishing a
   control channel between the user agent (UA) and the MS.

MS Control Protocol: The protocol used for by an AS to control an
   MS. The MS Control Protocol assumes a reliable underlying
   transport protocol for the MS Control Channel.

MS Media Dialog: A SIP dialog between the AS and MS that is used for
   establishing media sessions between a user device such as a SIP
   phone and the MS.

The definitions for AS, MS, and MRB above are taken from [RFC5167].

3. Architecture Overview

A Media Server (MS) is a network device that processes media streams.
Examples of media processing functionality may include:

o Control of the Real-Time Protocol (RTP) [RFC3550] streams using
   the Extended RTP Profile for Real-time Transport Control Protocol
   (RTCP)-Based Feedback (RTP/AVPF) [RFC4585].

o Mixing of incoming media streams.

o Media stream source (for multimedia announcements).

o Media stream processing (e.g., transcoding, DTMF detection).

o Media stream sink (for multimedia recordings).

An MS supplies one or more media processing functionalities, which
may include others than those illustrated above, to an Application
Server (AS). An AS is able to send a particular call to a suitable
MS, either through discovery of the capabilities that a specific MS
provides or through the use of a Media Resource Broker.

The type of processing that a Media Server performs on media streams
is specified and controlled by an Application Server. Application
Servers are logical entities that are capable of running one or more
instances of a communications application. Examples of Application
Servers that may interact with a Media Server are an AS acting as a
Conference 'Focus' as defined in [RFC4353], or an IVR application
using a Media Server to play announcements and detect DTMF key
presses.

Application servers use SIP to establish control channels between
themselves and MSs. An MS Control Channel implements a reliable
transport protocol that is used to carry the MS Control Protocol. A

SIP dialog used to establish a control channel is referred to as an
MS Control Dialog.

Application Servers terminate SIP [RFC3261] signaling from SIP User
Agents and may terminate other signaling outside the scope of this
document. They use SIP Third Party Call Control [RFC3725] (3PCC) to
establish, maintain, and tear down media streams from those SIP UAs
to a Media Server. A SIP dialog used by an AS to establish a media
session on an MS is referred to as an MS Media Dialog.

Media streams go directly between SIP User Agents and Media Servers.
Media Servers support multiple types of media. Common supported RTP
media types include audio and video, but others such as text and the
Binary Floor Control Protocol (BFCP) [RFC4583] are also possible.
This basic architecture, showing session establishment signaling
between a single AS and MS is shown in Figure 1 below.

```
       +-------------+                +-------------+
       |             | SIP (MS Control Dialog) |    |
       | Application |<---------------------->|   Media   |
       |   Server    |                |        |   Server  |
       |             |<---------------------->|           |
       +-------------+ SIP (MS Media Dialog)   +-------------+
              ^                                    ^   ^
               \                                   |  | RTP/SRTP
                \                                  |  |  audio/
                 \                                 |  |  video/etc)
                  \                                |  |
                   \                               |  v
                    \             SIP          +-------------+
                     \                         |             |
                      +-------------->|        |     SIP     |
                                      |        |  User Agent |
                                      |        |             |
                                      |        +-------------+
```
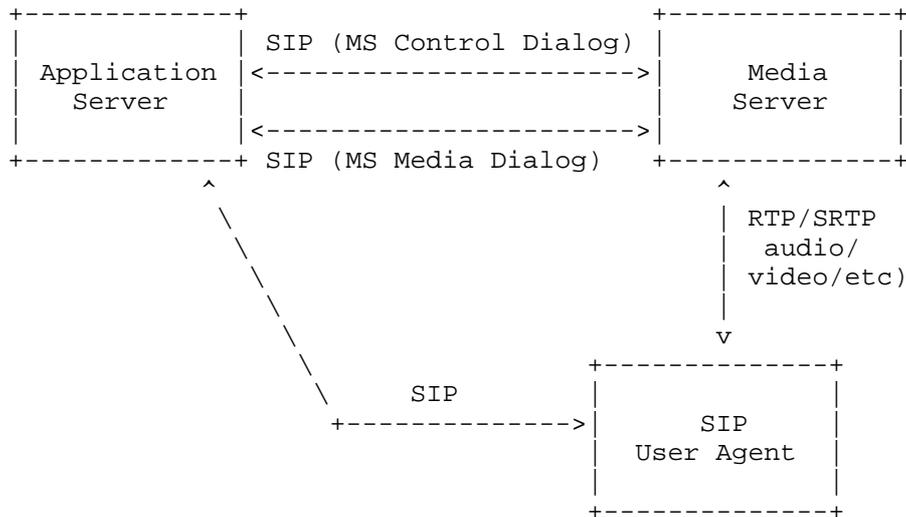
                  Figure 1: Basic Signaling Architecture

The architecture must support a many-to-many relationship between
Application Servers and Media Servers. In real world deployments, an
Application Server may interact with multiple Media Servers and/or a
Media Server may be controlled by more than one Application Server.

Application Servers can use the SIP URI as described in [RFC4240] to
request basic functions from Media Servers. Basic functions are
characterized as requiring no mid-call interactions between the AS
and MS. Examples of these functions are simple announcement-playing

or basic conference-mixing where the AS does not need to explicitly
control the mixing.

Most services however have interactions between the AS and MS during
a call or conference. The type of interactions can be generalized as
follows:

o commands from an AS to an MS to request the application or
   configuration of a function. The request may apply to a single
   media stream, multiple media streams associated with multiple SIP
   dialogs, or to properties of a conference mix.

o responses from an MS to an AS reporting on the status of
   particular commands.

o notifications from an MS to an AS that report results from
   commands or notify changes to subscribed status.

Commands, responses, and notifications are transported using one or
more dedicated control channels between the Application Server and
the Media Server. Dedicated control channels provide reliable,
sequenced, peer-to-peer transport for Media Server control
interactions. Implementations must support the Transport Control
Protocol (TCP) [RFC0793] and may support the Stream Control
Transmission Protocol (SCTP) [RFC4960]. Because MS control requires
sequenced reliable delivery of messages, unreliable protocols such as
the User Datagram Protocol (UDP) are not suitable. Implementations
must support TLS [RFC5246] as a transport-level security mechanism
although its use in deployments is optional. A dedicated control
channel is shown in Figure 2 below.

```
      +------------+                     +-------------+
      |            |                     |             |
      | Application |   MS ctrl channel  |    Media    |
      |   Server   |<------------------->|   Server    |
      |            |                     |             |
      +------------+                     +-------------+
                                              ^ ^ ^
                                  RTP/SRTP | | |
                                   (audio/ | | |
                                 video/etc) | | |
                                          | | v
                               +---|-v-------+
                             +-|---v-------+ |
                           +-|----------+ | |
                           |            | | |
                           |     SIP    | | |
                           | User Agent | |-+
                           |            |-+
                           +------------+
```
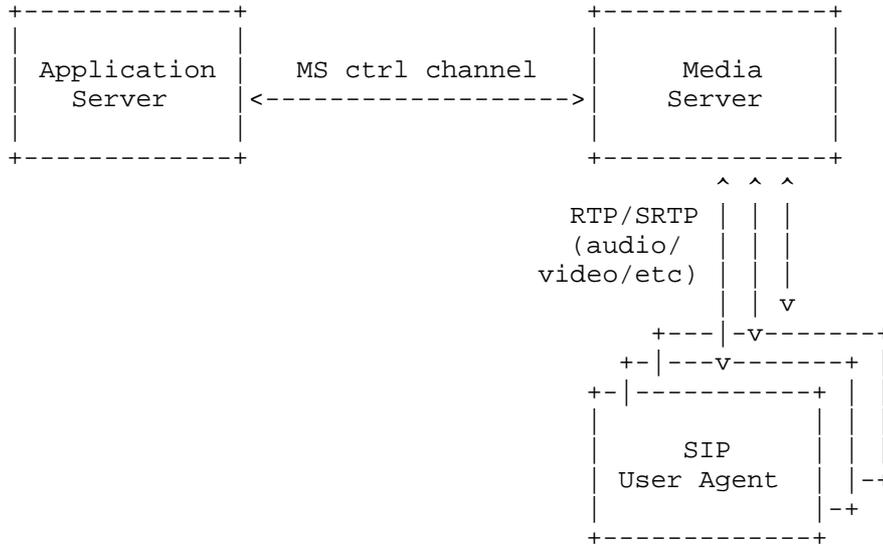
                Figure 2: Media Server Control Architecture

   Both Application Servers and Media Servers may interact with other
   servers for specific purposes beyond the scope of this document. For
   example, Application Servers will often communicate with other
   infrastructure components that are usually based on deployment
   requirements with links to back-office data stores and applications.
   Media Servers will often retrieve announcements from external file
   servers. Also, many Media Servers support IVR dialog services using
   VoiceXML [W3C.REC-voicexml20-20040316]. In this case, the MS
   interacts with other servers using HTTP during standard VoiceXML
   processing. VoiceXML Media Servers may also interact with speech
   engines (for example, using the Media Resource Control Protocol
   version 2 (MRCPv2)) for speech recognition and generation purposes.

   Some specific types of interactions between Application and Media
   servers are also out of scope for this document. MS resource
   reservation is one such interaction. Also, any interactions between
   Application Servers, or between Media Servers, are also out of scope.

4. SIP Usage

   The Session Initiation Protocol (SIP) [RFC3261] was developed by the
   IETF for the purposes of initiating, managing, and terminating
   multimedia sessions. The popularity of SIP has grown dramatically
   since its inception and is now the primary Voice over IP (VoIP)
   protocol. This includes being selected as the basis for
   architectures such as the IP Multimedia Subsystem (IMS) in 3GPP and

included in many of the early live deployments of VoIP-related
systems. Media servers are not a new concept in IP telephony
networks and there have been numerous signaling protocols and
techniques proposed for their control. The most popular techniques
to date have used a combination of SIP and various markup languages
to convey media service requests and responses.

As discussed in Section 3 and illustrated in Figure 1, the logical
architecture described by this document involves interactions between
an Application Server (AS) and a Media Server (MS). The SIP
interactions can be broken into "MS media dialogs" that are used
between an AS and an MS to establish media sessions between an
endpoint and a Media Server, and "MS control dialogs" that are used
to establish and maintain MS control channels.

SIP is the primary signaling protocol for session signaling and is
used for all media sessions directed towards a Media Server as
described in this document. Media Servers may support other
signaling protocols but this type of interaction is not considered
here. Application Servers may terminate non-SIP signaling protocols
but must gateway those requests to SIP when interacting with a Media
Server.

SIP will also be used for the creation, management, and termination
of the dedicated MS control channel(s). Control channel(s) provide
reliable sequenced delivery of MS Control Protocol messages. The
Application and Media Servers use the SDP attributes defined in
[RFC4145] to allow SIP negotiation of the control channel. A control
channel is closed when SIP terminates the corresponding MS control
dialog. Further details and example flows are provided in the SIP
Control Framework [SIP-CTRL-FW]. The SIP Control Framework also
includes basic control message semantics corresponding to the types
of interactions identified in Section 3. It uses the concept of
"packages" to allow domain-specific protocols to be defined using the
Extensible Markup Language (XML) [W3C.REC-xml-20060816] format. The
MS Control Protocol is made up of one or more packages for the SIP
Control Framework.

Using SIP for both media and control dialogs provides a number of
inherent benefits over other potential techniques. These include:

1. The use of SIP location and rendezvous capabilities, as defined
    in [RFC3263]. This provides core mechanisms for routing a SIP
    request based on techniques such as DNS SRV and NAPTR records.
    The SIP infrastructure makes heavy use of such techniques.

2. The security and identity properties of SIP; for example, using
    TLS for reliably and securely connecting to another SIP-based

entity. The SIP protocol has a number of identity mechanisms
that can be used. [RFC3261] provides an intra-domain digest-
based mechanism and [RFC4474] defines a certificate-based inter-
domain identity mechanism. SIP with S/MIME provides the ability
to secure payloads using encrypted and signed certificate
techniques.

3. SIP has extremely powerful and dynamic media-negotiation
    properties as defined in [RFC3261] and [RFC3264].

4. The ability to select an appropriate SIP entity based on
    capability sets as discussed in [RFC3840]. This provides a
    powerful function that allows Media Servers to convey a specific
    capability set. An AS is then free to select an appropriate MS
    based on its requirements.

5. Using SIP also provides consistency with IETF protocols and
    usages. SIP was intended to be used for the creation and
    management of media sessions, and this provides a correct usage
    of the protocol.

As mentioned previously in this section, media services using SIP are
fairly well understood. Some previous proposals suggested using the
SIP INFO [RFC2976] method as the transport vehicle between the AS and
MS. Using SIP INFO in this way is not advised for a number of
reasons, which include:

o INFO is an opaque request with no specific semantics. A SIP
   endpoint that receives an INFO request does not know what to do
   with it based on SIP signaling.

o SIP INFO was not created to carry generic session control
   information along the signaling path, and it should only really be
   used for optional application information, e.g., carrying mid-call
   Public Switched Telephone Network (PSTN) signaling messages
   between PSTN gateways.

o SIP INFO traverses the signaling path, which is an inefficient use
   for control messages that can be routed directly between the AS
   and MS.

o [RFC3261] contains rules when using an unreliable protocol such as
   UDP. When a packet reaches a size close to the Maximum
   Transmission Unit (MTU), the protocol should be changed to TCP.
   This type of operation is not ideal when constantly dealing with
   large payloads such as XML-formatted MS control messages.

5. Media Control for IVR Services

   One of the functions of a Media Server is to assist an Application
   Server that is implementing IVR services by performing media
   processing functions on media streams. Although "IVR" is somewhat
   generic terminology, the scope of media functions provided by an MS
   addresses the needs for user interaction dialogs. These functions
   include media transcoding, basic announcements, user input detection
   (via DTMF or speech), and media recording.

   A particular IVR or user dialog application typically requires the
   use of several specific media functions, as described above. The
   range and complexity of IVR dialogs can vary significantly, from a
   simple single announcement play-back to complex voice mail
   applications.

   As previously discussed, an AS uses SIP [RFC3261] and SDP [RFC4566]
   to establish and configure media sessions to a Media Server. An AS
   uses the MS control channel, established using SIP, to invoke IVR
   requests and to receive responses and notifications. This topology
   is shown in Figure 3 below.

```
      +-------------+            SIP          +-------------+
      | Application |<--------------------------->|   Media     |
      |   Server    | (media & MS Control dialogs) |   Server    |
      |             |                              |             |
      |             | MS Control Protocol (IVR)    |             |
      |             |<--------------------------->| (IVR media  |
      | (App logic) |      (CtrlChannel)           | functions)  |
      +------------+                              +------------+
           ^                                          ^^
            \                                         ||   R
             \                                        ||   T
              \                                       ||   P
               \                                      ||   /
                \                                     ||   S
                 \                                    ||   R
                  \                                   ||   T
                   \                                  ||   P
                    \                                 vv
                     \    call signaling        +-----------+
                      --------------------------->|   User    |
                          (e.g., SIP)            | Equipment |
                                                 +-----------+
```
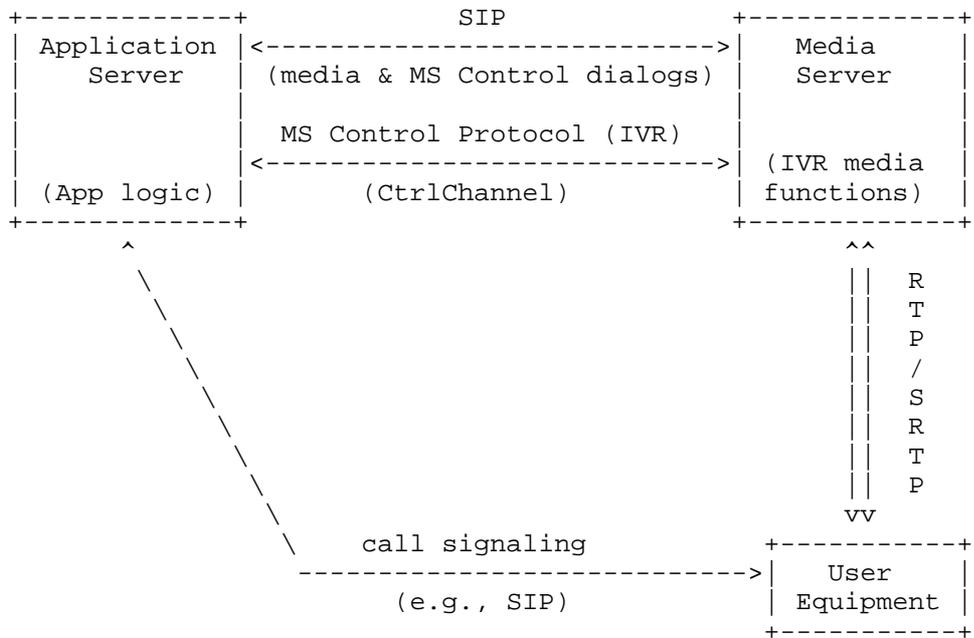
                       Figure 3: IVR Topology

The variety in complexity of Application Server IVR services requires
support for different levels of media functions from the Media Server
as described in the following sub-sections.

5.1. Basic IVR Services

For simple basic announcement requests, the MS control channel, as
depicted in Figure 3 above, is not required. Simple announcement
requests may be invoked on the Media Server using the SIP URI
mechanism defined in [RFC4240]. This interface allows no digit
detection or collection of user input and no mid-call dialog control.
However, many applications only require basic media services, and the
processing burden on the Media Server to support more complex
interactions with the AS would not be needed in that case.

5.2. IVR Services with Mid-Call Controls

For more complex IVR dialogs, which require mid-call interaction and
control between the Application Server and the Media Server, the MS
control channel (as shown in Figure 3 above) is used to invoke
specific media functions on the Media Server. These functions
include, but are not limited to, complex announcements with barge-in
facility, user-input detection and reporting (e.g., DTMF) to an
Application Server, DTMF and voice-activity controlled recordings,
etc. Composite services, such as play-collect and play-record, are
also addressed by this model.

Mid-call control also allows Application Servers to subscribe to IVR-
related events and for the Media Server to notify the AS when these
events occur. Examples of such events are announcement completion
events, record completion events, and reporting of collected DTMF
digits.

5.3. Advanced IVR Services

Although IVR services with mid-call control, as described above,
provide a comprehensive set of media functions expected from a Media
Server, the advanced IVR services model allows a higher level of
abstraction describing application logic, as provided by VoiceXML, to
be executed on the Media Server. Invocation of VoiceXML IVR dialogs
may be via the "Prompt and Collect" mechanism of [RFC4240].
Additionally, the IVR control protocol can be extended to allow
VoiceXML requests to also be invoked over the MS control channel.
VoiceXML IVR services invoked on the Media Server may require an HTTP
interface (not shown in Figure 3) between the Media Server and one or
more back-end servers that host or generate VoiceXML documents. The
back-end server(s) may or may not be physically separate from the
Application Server.

6. Media Control for Conferencing Services

   [RFC4353] describes the overall architecture and protocol components
   needed for multipoint conferencing using SIP. The framework for
   centralized conferencing [RFC5239] extends the framework to include a
   protocol between the user and the conferencing server. [RFC4353]
   describes the conferencing server decomposition but leaves the
   specifics open.

   This section describes the decomposition and discusses the
   functionality of the decomposed functional units. The conferencing
   factory and the conference focus are part of the Application Server
   described in this document.

   An Application Server uses SIP Third Party Call Control [RFC3725] to
   establish media sessions from SIP user agents to a Media Server. The
   same mechanism is used by the Application Server as described in this
   section to add/remove participants to/from a conference, as well as
   to handle the involved media streams set up on a per-user basis.
   Since the XCON framework has been conceived as protocol-agnostic when
   talking about the Call Signaling Protocol used by users to join a
   conference, an XCON-compliant Application Server will have to take
   care of gatewaying non-SIP signaling negotiations. This is in order
   to set up and make available valid SIP media sessions between itself
   and the Media Server, while still keeping the non-SIP interaction
   with the user in a transparent way.

```
         +-----------+           +-----------+
         |           | SIP (2m+1c) |         |
         | Application|-------------|  Media   |
         |   Server  |             |  Server  |
         |  (Focus)  |-------------|  (Mixer) |
         |           | CtrlChannel |          |
         +-----------+           +-----------+
              |    \                  ..  .
              |     \\         RTP...    .
              |      \\            ..    .
              |  H.323 \\        ...     .
          SIP |         \\ ...          .RTP
              |         ..\             .
              |      ...   \\           .
              |     ...      \\         .
              |    ..          \\       .
              |   ...           \\      .
              \  ..               \     .
         +-----------+           +-----------+
         |Participant|           |Participant|
         +-----------+           +-----------+
```
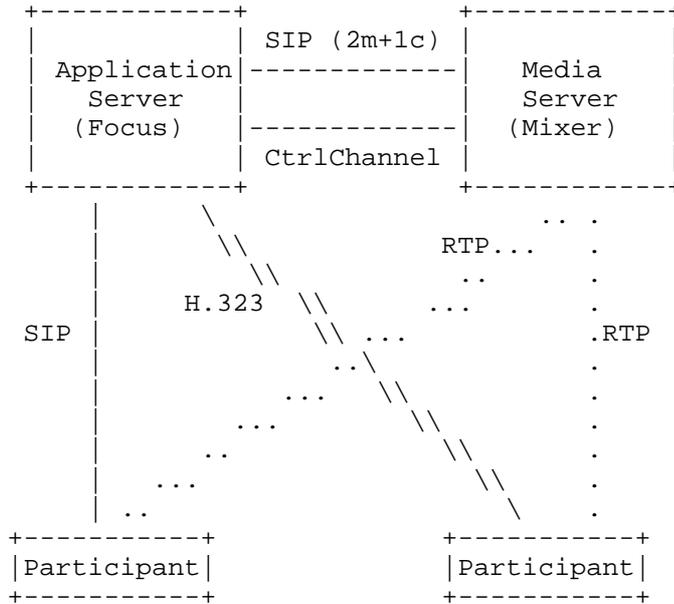
                  Figure 4: Conference Topology

To complement the functionality provided by 3PCC and by the XCON
control protocol, the Application Server makes use of a dedicated
Media Server control channel in order to set up and manage media
conferences on the Media Server. Figure 4 shows the signaling and
media paths for a two-participant conference. The three SIP dialogs
between the AS and MS establish one control session (1c) and two
media sessions (2m) from the participants (one originally signaled
using H.323 and then gatewayed into SIP and one signaled directly in
SIP).

As a conference focus, the Application Server is responsible for
setting up and managing a media conference on the Media Servers, in
order to make sure that all the media streams provided in a
conference are available to its participants. This is achieved by
using the services of one or more mixer entities (as described in RFC
4353), whose role as part of the Media Server is described in this
section. Services required by the Application Server include, but
are not limited to, means to set up, handle, and destroy a new media
conference, adding and removing participants from a conference,
managing media streams in a conference, controlling the layout and
the mixing configuration for each involved media, allowing per-user
custom media profiles, and so on.

As a mixer entity, in such a multimedia conferencing scenario, the
Media Server receives a set of media streams of the same type (after
transcoding if needed) and then takes care of combining the received
media in a type-specific manner, redistributing the result to each
authorized participant. The way each media stream is combined, as
well as the media-related policies, is properly configured and
handled by the Application Server by means of a dedicated MS control
channel.

To summarize, the AS needs to be able to manage Media Servers at a
conference and participant level.

6.1. Creating a New Conference

When a new conference is created, as a result of a previous
conference scheduling or of the first participant dialing in to a
specified URI, the Application Server must take care of appropriately
creating a media conference on the Media Server. It does so by
sending an explicit request to the Media Server. This can be by
means of an MS control channel message. This request may contain
detailed information upon the desired settings and policies for the
conference (e.g., the media to involve, the mixing configuration for
them, the relevant identifiers, etc.). The Media Server validates
such a request and takes care of allocating the needed resources to
set up the media conference.

Application Servers may use mechanisms other than sending requests
over the control channel to establish conferences on a Media Server,
and then subsequently use the control channel to control the
conference. Examples of other mechanisms to create a conference
include using the Request-URI mechanism of [RFC4240] or the
procedures defined in [RFC4579].

Once done, the MS informs the Application Server about the result of
the request. Each conference will be referred to by a specific
identifier, which both the Application Server and the Media Server
will include in subsequent transactions related to the same
conference (e.g., to modify the settings of an extant conference).

6.2. Adding a Participant to a Conference

As stated before, an Application Server uses SIP 3PCC to establish
media sessions from SIP user agents to a Media Server. The URI that
the AS uses in the INVITE to the MS may be one associated with the
conference on the MS. More likely however, the media sessions are
first established to the Media Server using a URI for the Media
Server and then subsequently joined to the conference using the MS

Control Protocol. This allows IVR dialogs to be performed prior to
joining the conference.

The AS as a 3PCC correlates the media session negotiation between the
UA and the MS, in order to appropriately establish all the needed
media streams based on the conference policies.

6.3. Media Controls

The XCON Common Data Model [XCON-DM] currently defines some basic
media-related controls, which conference-aware participants can take
advantage of in several ways, e.g., by means of an XCON conference
control protocol or IVR dialogs. These controls include the
possibility to modify the participants' own volume for audio in the
conference, configure the desired layout for incoming video streams,
mute/unmute oneself, and pause/unpause one's own video stream. Such
controls are exploited by conference-aware participants through the
use of dedicated conference control protocol requests to the
Application Server. The Application Server takes care of validating
such requests and translates them into the Media Server Control
Protocol, before forwarding them over the MS Control Channel to the
MS. According to the directives provided by the Application Server,
the Media Server manipulates the involved media streams accordingly.

```
      +------------+          +------------+
      |            | 'Include audio |          |
      | Application|  sent by user X |   Media  |
      |   Server   |  in conf Y mix' |  Server  |
      |   (Focus)  |---------------->|  (Mixer) |
      |            |    (MS CtrlChn) |          |
      +------^-----+          +-----------+
             |                      ..
             |                     ...
             | 'Unmute me'      ... RTP
             |   (XCON)      ...
             |            ...
             |         ...
      +----------+    ...
      |Participant|...
      +----------+
```
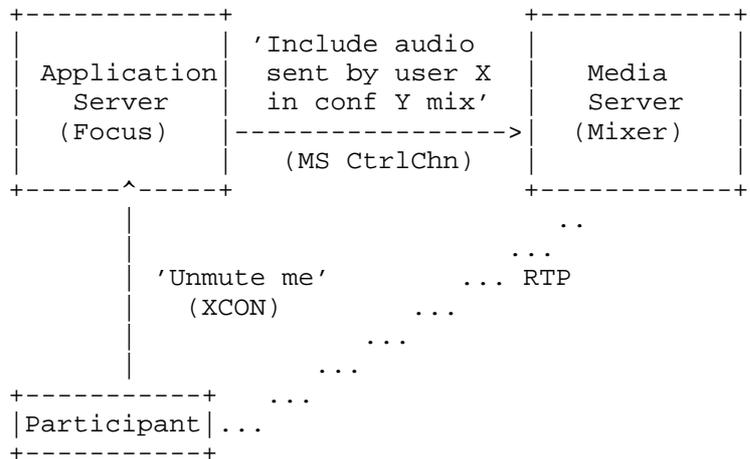
              Figure 5: Conferencing Example: Unmuting A Participant

The Media Server may need to inform the AS of events like in-band
DTMF tones during the conference.

6.4. Floor Control

   The XCON framework introduces "floor control" functionality as an
   enhancement upon [RFC4575]. Floor control is a means to manage joint
   or exclusive access to shared resources in a (multiparty)
   conferencing environment. Floor control is not a mandatory mechanism
   for a conferencing system implementation, but it provides advanced
   media input control features for conference-aware participants. Such
   a mechanism allows for coordinated and moderated access to any set of
   resources provided by the conferencing system. To do so, a so-called
   floor is associated to a set of resources, thus representing for
   participants the right to access and manipulate the related resources
   themselves. In order to take advantage of the floor control
   functionality, a specific protocol, the Binary Floor Control
   Protocol, has been specified [RFC4582]. [RFC4583] provides a way for
   SIP UAs to set up a BFCP connection towards the Floor Control Server
   and exploit floor control by means of a Connection-Oriented Media
   (COMEDIA) [RFC4145] negotiation.

   In the context of the AS-MS interaction, floor control constitutes a
   further means to control participants' media streams. A typical
   example is a floor associated with the right to access the shared
   audio channel in a conference. A participant who is granted such a
   floor is granted by the conferencing system the right to talk, which
   means that its audio frames are included by the MS in the overall
   audio conference mix. Similarly, when the floor is revoked, the
   participant is muted in the conference, and its audio is excluded
   from the final mix.

   The BFCP defines a Floor Control Server (FCS) and the floor chair.
   It is clear that the floor chair making decisions about floor
   requests is part of the application logic. This implies that when
   the role of floor chair in a conference is automated, it will
   normally be part of the AS.

   The example makes it clear that there can be a direct or indirect
   interaction between the Floor Control Server and the Media Server, in
   order to correctly bind each floor to its related set of media
   resources. Besides, a similar interaction is needed between the
   Floor Control Server and the Application Server as well, since the
   latter must be aware of all the associations between floors and
   resources, in order to opportunely orchestrate the related bindings
   with the element responsible for such resources (e.g., the Media
   Server when talking about audio and/or video streams) and the
   operations upon them (e.g., mute/unmute a participant in a
   conference). For this reason, the Floor Control Server can be co-

located with either the Media Server or the Application Server, as
long as both elements are allowed to interact with the Floor Control
Server by means of some kind of protocol.

In the following text, both the approaches will be described in order
to better explain the interactions between the involved components in
both the topologies.

When the AS and the FCS are co-located, the scenario is quite
straightforward. In fact, it can be considered as a variation of the
case depicted in Figure 5. The only relevant difference is that in
this case the action the AS commands on the control channel is
triggered by a change in the floor control status instead of a
specific control requested by a participant himself. The sequence
diagram in Figure 6 describes the interaction between the involved
parties in a typical scenario. It assumes that a BFCP connection
between the UA and the FCS (which we assume is co-located with the
AS) has already been negotiated and established, and that the UA has
been made aware of all the relevant identifiers and floors-resources-
associations (e.g., by means of [RFC4583]). It also assumes that the
AS has previously configured the media mixing on the MS using the MS
control channel. Every frame the UA might be sending on the related
media stream is currently being dropped by the MS, since the UA still
isn't authorized to use the resource. For a SIP UA, this state could
be consequent to a 'sendonly' field associated to the media stream in
a re-INVITE originated by the MS. It is worth pointing out that the
AS has to make sure that no user media control mechanisms, such as
mentioned in the previous sub-section, can override the floor
control.

```
   UA                                   AS                         MS
   (Floor Participant)                  (FCS)
   |                                    |                          |
   |<=================== One-way RTP stream =====================|
   |                                    |                          |
   | FloorRequest(BFCP)                 |                          |
   |----------------------------------->|                          |
   |                                    |                          |
   |    FloorRequestStatus[PENDING](BFCP) |                        |
   |<-----------------------------------|                          |
   |                                    |--+ apply                 |
   |                                    | | policies               |
   |                                    |<-+ to request            |
   |                                    |                          |
   |   FloorRequestStatus[ACCEPTED](BFCP) |                        |
   |<-----------------------------------|                          |
   |                                    |                          |
   .                                    .                          .
   .                                    .                          .
   |                                    |                          |
   |    FloorRequestStatus[GRANTED](BFCP) |                        |
   |<-----------------------------------|                          |
   |                                    | 'Unmute UA' (CtrlChn)    |
   |                                    |------------------------->|
   |                                    |                          |
   |<=================== Bidirectional RTP stream ===============>|
   |                                    |                          |
   .                                    .                          .
   .                                    .                          .
```
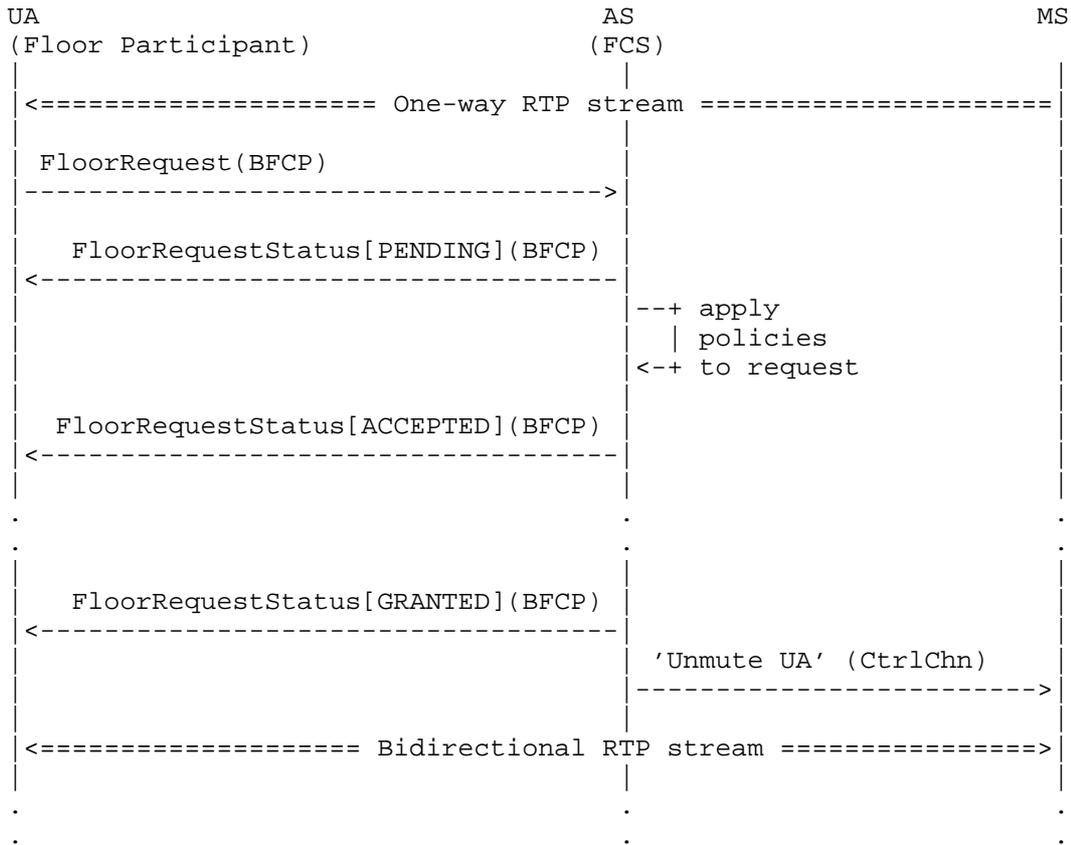
           Figure 6: Conferencing Example: Floor Control Call Flow

   A UA, which also acts as a floor participant, sends a "FloorRequest"
   to the floor control server (FCS, which is co-located with the AS),
   stating his will to be granted the floor associated with the audio
   stream in the conference. The AS answers the UA with a
   "FloorRequestStatus" message with a PENDING status, meaning that a
   decision on the request has not been made yet. The AS, according to
   the BFCP policies for this conference, makes a decision on the
   request, i.e., accepting it. Note that this decision might be
   relayed to another participant in case he has previously been
   assigned as chair of the floor. Assuming the request has been
   accepted, the AS notifies the UA about the decision with a new
   "FloorRequestStatus", this time with an ACCEPTED status in it. The
   ACCEPTED status of course only means that the request has been
   accepted, which doesn't mean the floor has been granted yet. Once
   the queue management in the FCS, according to the specified
   algorithms for scheduling, states that the floor request previously

made by the UA can be granted, the AS sends a new
"FloorRequestStatus" to the UA with a GRANTED status, and takes care
of unmuting the participant in the conference by sending a directive
to the MS through the control channel. Once the UA receives the
notification stating his request has been granted, he can start
sending its media, aware of the fact that now his media stream won't
be dropped by the MS. In case the session has been previously
updated with a 'sendonly' associated to the media stream, the MS must
originate a further re-INVITE stating that the media stream flow is
now bidirectional ('sendrecv').

As mentioned before, this scenario envisages an automated floor chair
role, where it's the AS, according to some policies, which makes
decisions on floor requests. The case of a chair role performed by a
real person is exactly the same, with the difference that the
incoming request is not directly handled by the AS according to its
policies, but it is instead forwarded to the floor control
participant that the chair UA is exploiting. The decision on the
request is then communicated by the chair UA to the AS-FCS by means
of a 'ChairAction' message.

The rest of this section will instead explore the other scenario,
which assumes that the interaction between AS-FCS happens through the
MS control channel. This scenario is compliant with the H.248.19
document related to conferencing in 3GPP. The following sequence
diagram describes the interaction between the involved parties in the
same use-case scenario that has been explored for the previous
topology: consequently, the diagram makes exactly the same
assumptions that have been made for the previously described
scenario. This means that the scenario again assumes that a BFCP
connection between the UA and the FCS has already been negotiated and
established, and that the UA has been made aware of all the relevant
identifiers and floors-resources-associations. It also assumes that
the AS has previously configured the media mixing on the MS using the
MS control channel. This time it includes identifying the BFCP-
moderated resources, establishing basic policies and instructions
about chair identifiers for each resource, and subscribing to events
of interest, because the FCS is not co-located with the AS anymore.
Additionally, a BFCP session has been established between the AS
(which in this scenario acts as a floor chair) and the FCS (MS).
Every frame the UA might be sending on the related media stream is
currently being dropped by the MS, since the UA still isn't
authorized to use the resource. For a SIP UA, this state could be
consequent to a 'sendonly' field associated to the media stream in a
re-INVITE originated by the MS. Again, it is worth pointing out that
the AS has to make sure that no user media control mechanisms, such
as mentioned in the previous sub-section, can override the floor
control.

```
      UA                        AS                           MS
   (Floor Participant)    (Floor Chair)                    (FCS)
      |                         |                            |
      |<==================== One-way RTP stream ====================|
      |                         |                            |
      | FloorRequest(BFCP)      |                            |
      |------------------------------------------------------------>|
      |                         |                            |
      |                         | FloorRequestStatus[PENDING](BFCP) |
      |<----------------------------------------------------------- |
      |                         | FloorRequestStatus[PENDING](BFCP) |
      |                         |<--------------------------------- |
      |                         |                            |
      |                         | ChairAction[ACCEPTED] (BFCP)      |
      |                         |--------------------------------->|
      |                         |     ChairActionAck (BFCP)        |
      |                         |<--------------------------------- |
      |                         |                            |
      |                         | FloorRequestStatus[ACCEPTED](BFCP) |
      |<----------------------------------------------------------- |
      |                         |                            |
      .                         .                            .
      .                         .                            .
      |                         |                            |
      |                         | FloorRequestStatus[GRANTED](BFCP) |
      |<----------------------------------------------------------- |
      |                         | 'Floor has been granted' (CtrlChn) |
      |                         |<--------------------------------- |
      |                         |                            |
      |<================== Bidirectional RTP stream ===============>|
      |                         |                            |
      .                         .                            .
      .                         .                            .
```
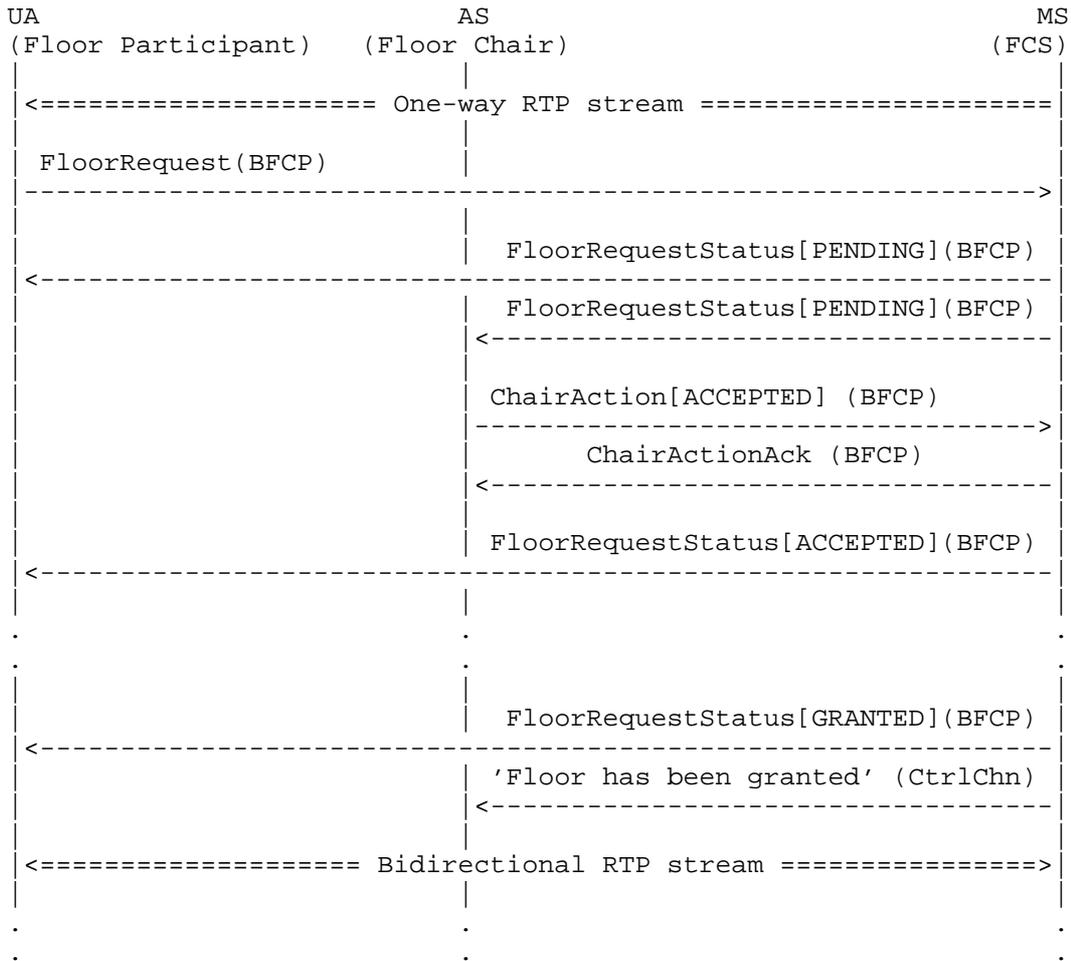
           Figure 7: Conferencing Example: Floor Control Call Flow

   A UA, which also acts as a floor participant, sends a "FloorRequest"
   to the floor control server (FCS, which is co-located with the MS),
   stating his will to be granted the floor associated with the audio
   stream in the conference. The MS answers the UA with a
   "FloorRequestStatus" message with a PENDING status, meaning that a
   decision on the request has not been made yet. It then notifies the
   AS, which in this example handles the floor chair role, about the new
   request by forwarding there the received request. The AS, according
   to the BFCP policies for this conference, makes a decision on the
   request, i.e., accepting it. It informs the MS about its decision
   through a BFCP "ChairAction" message. The MS then acknowledges the
   'ChairAction' message and then notifies the UA about the decision

with a new "FloorRequestStatus", this time with an ACCEPTED status in
it. The ACCEPTED status of course only means that the request has
been accepted, which doesn't mean the floor has been granted yet.
Once the queue management in the MS, according to the specified
algorithms for scheduling, states that the floor request previously
made by the UA can be granted, the MS sends a new
"FloorRequestStatus" to the UA with a GRANTED status, and takes care
of unmuting the participant in the conference. Once the UA receives
the notification stating his request has been granted, he can start
sending its media, aware of the fact that now his media stream won't
be dropped by the MS. In case the session has been previously
updated with a 'sendonly' associated to the media stream, the MS must
originate a further re-INVITE stating that the media stream flow is
now bidirectional ('sendrecv').

This scenario envisages an automated floor chair role, where it's the
AS, according to some policies, which makes decisions on floor
requests. Again, the case of a chair role performed by a real person
is exactly the same, with the difference that the incoming request is
not forwarded to the AS but to the floor control participant that the
chair UA is exploiting. The decision on the request is communicated
by means of a 'ChairAction' message in the same way.

Another typical scenario is a BFCP-moderated conference with no chair
to manage floor requests. In such a scenario, the MS has to take
care of incoming requests according to some predefined policies,
e.g., always accepting new requests. In this case, no decisions are
required by external entities, since all are instantly decided by
means of policies in the MS.

As stated before, the case of the FCS co-located with the AS is much
simpler to understand and exploit. When the AS has full control upon
the FCS, including its queue management, the AS directly instructs
the MS according to the floor status changes, e.g., by instructing
the MS through the control channel to unmute a participant who has
been granted the floor associated to the audio media stream.

7. Security Considerations

This document describes the architectural framework to be used for
Media Server control. Its focus is the interactions between
Application Servers and Media Servers. User agents interact with
Application Servers by means of signaling protocols such as SIP.
These interactions are beyond the scope of this document.
Application Servers are responsible for utilizing the security
mechanisms of their signaling protocols, combined with application-
specific policy, to ensure they grant service only to authorized
users. Media interactions between user agents and Media Servers are

also outside the scope of this document. Those interactions are at
the behest of Application Servers, which must ensure that appropriate
security mechanisms are used. For example, if the MS is acting as
the FCS, then the BFCP connection between the user agent and the MS
is established to the MS by the AS using SIP and the SDP mechanisms
described in [RFC4583]. BFCP [RFC4582] strongly imposes the use of
TLS for BFCP.

Media Servers are valuable network resources and need to be protected
against unauthorized access. Application Servers use SIP and related
standards both to establish control channels to Media Servers and to
establish media sessions, including BFCP sessions, between an MS and
end users. Media servers use the security mechanisms of SIP to
authenticate requests from Application servers and to ensure the
integrity of those requests. Leveraging the security mechanisms of
SIP ensures that only authorized Application Servers are allowed to
establish sessions to an MS and to access MS resources through those
sessions.

Control channels between an AS and MS carry the MS control protocol,
which affects both the service seen by end users and the resources
used on a Media Server. TLS [RFC5246] must be implemented as the
transport-level security mechanism for control channels to guarantee
the integrity of MS control interactions.

The resources of an MS can be shared by more than one AS. Media
Servers must prevent one AS from accessing and manipulating the
resources that have been assigned to another AS. This may be
achieved by an MS associating ownership of a resource to the AS that
originally allocates it, and then insuring that future requests
involving that resource correlate to the AS that owns and is
responsible for it.

8. Acknowledgments

The authors would like to thank Spencer Dawkins for detailed reviews
and comments, Gary Munson for suggestions, and Xiao Wang for review
and feedback.

9. Contributors

This document is a product of the Media Control Architecture Design
Team. In addition to the editor, the following individuals
constituted the design team and made substantial textual
contributions to this document:

Chris Boulton: cboulton@ubiquity.net

Martin Dolly: mdolly@att.com

Roni Even: roni.even@polycom.co.il

Lorenzo Miniero: lorenzo.miniero@unina.it

Adnan Saleem: Adnan.Saleem@radisys.com

10. Informative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7,
           RFC 793, September 1981.

[RFC2976] Donovan, S., "The SIP INFO Method", RFC 2976,
           October 2000.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston,
           A., Peterson, J., Sparks, R., Handley, M., and E.
           Schooler, "SIP: Session Initiation Protocol", RFC 3261,
           June 2002.

[RFC3263] Rosenberg, J. and H. Schulzrinne, "Session Initiation
           Protocol (SIP): Locating SIP Servers", RFC 3263,
           June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model
           with Session Description Protocol (SDP)", RFC 3264,
           June 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
           Jacobson, "RTP: A Transport Protocol for Real-Time
           Applications", STD 64, RFC 3550, July 2003.

[RFC3725] Rosenberg, J., Peterson, J., Schulzrinne, H., and G.
           Camarillo, "Best Current Practices for Third Party Call
           Control (3pcc) in the Session Initiation Protocol (SIP)",
           BCP 85, RFC 3725, April 2004.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat,
           "Indicating User Agent Capabilities in the Session
           Initiation Protocol (SIP)", RFC 3840, August 2004.

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

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

   [RFC4353] Rosenberg, J., "A Framework for Conferencing with the
              Session Initiation Protocol (SIP)", RFC 4353,
              February 2006.

   [RFC4474] Peterson, J. and C. Jennings, "Enhancements for
              Authenticated Identity Management in the Session
              Initiation Protocol (SIP)", RFC 4474, August 2006.

   [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session
              Description Protocol", RFC 4566, July 2006.

   [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session
              Initiation Protocol (SIP) Event Package for Conference
              State", RFC 4575, August 2006.

   [RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol
              (SIP) Call Control - Conferencing for User Agents",
              BCP 119, RFC 4579, August 2006.

   [RFC4582] Camarillo, G., Ott, J., and K. Drage, "The Binary Floor
              Control Protocol (BFCP)", RFC 4582, November 2006.

   [RFC4583] Camarillo, G., "Session Description Protocol (SDP) Format
              for Binary Floor Control Protocol (BFCP) Streams",
              RFC 4583, November 2006.

   [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey,
              "Extended RTP Profile for Real-time Transport Control
              Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585,
              July 2006.

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

   [RFC5167] Dolly, M. and R. Even, "Media Server Control Protocol
              Requirements", RFC 5167, March 2008.

   [RFC5239] Barnes, M., Boulton, C., and O. Levin, "A Framework for
              Centralized Conferencing", RFC 5239, June 2008.

   [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.2", RFC 5246, August 2008.

    [SIP-CTRL-FW]
              Boulton, C., Melanchuk, T., and S. McGlashan, "Media
              Control Channel Framework", Work in Progress,
              February 2009.

    [W3C.REC-voicexml20-20040316]
              Carter, J., Tryphonas, S., Danielsen, P., Burnett, D.,
              Rehor, K., McGlashan, S., Ferrans, J., Porter, B., Lucas,
              B., and A. Hunt, "Voice Extensible Markup Language
              (VoiceXML) Version 2.0", World Wide Web Consortium
              Recommendation REC-voicexml20-20040316, March 2004,
              <http://www.w3.org/TR/2004/REC-voicexml20-20040316>.

    [W3C.REC-xml-20060816]
              Sperberg-McQueen, C., Paoli, J., Bray, T., Maler, E., and
              F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fourth
              Edition)", World Wide Web Consortium Recommendation REC-
              xml-20060816, August 2006,
              <http://www.w3.org/TR/2006/REC-xml-20060816>.

    [XCON-DM] Novo, O., Camarillo, G., Morgan, D., and J. Urpalainen,
              "Conference Information Data Model for Centralized
              Conferencing (XCON)", Work in Progress, April 2009.

Author's Address

    Tim Melanchuk (editor)
    Rain Willow Communications

    EMail: tim.melanchuk@gmail.com