# Solution Reference Network Design for Cisco MediaSense

Release 8.5(2)

*September 2011*

**Americas Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0833

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCBs public domain version of the UNIX operating system. All rights reserved. Copyright 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

# Cisco MediaSense, Release 8.5(2) SRND

## Product Overview

Cisco MediaSense is a SIP-based network level service whose function is to provide voice and video media recording capabilities for other network devices. Fully integrated into Cisco's Unified Communications architecture, Cisco MediaSense automatically captures and stores every Voice over IP (VOIP) conversation which traverses appropriately configured Cisco IP phones. In addition, an IP phone user may call the Cisco MediaSense system directly in order to leave a recording consisting of only media generated by that user. Such recordings may include video as well as audio, offering a simple and easy method for recording video blogs and podcasts.

Recordings may be accessed in several ways. While a recording is still in progress, it can be streamed live ("monitored") through a computer which is equipped with a media player such as VLC, QuickTime or RealPlayer. Once completed, recordings may be played back in the same way, or downloaded in raw form via HTTP. They may also be converted into .mp4 and downloaded in that format. All access to recordings, either in progress or completed, is through web-friendly URIs.

Media recordings occupy a fair amount of disk space, so space management is a significant concern. Cisco MediaSense offers two modes of operation with respect to space management: Retention Priority and Recording Priority. Selected at installation time, these modes are designed to address two opposing and incompatible use cases: one in which all recording sessions must be retained until explicitly deleted, even if it means new recording sessions cannot be captured; and one in which older recording sessions can be deleted if necessary to make room for new ones. A sophisticated set of events and APIs is provided in order to enable client software to automatically control and manage disk space.

Cisco MediaSense also maintains a metadata database, in which information about all recordings is maintained. A comprehensive Web 2.0 API is provided which allows client equipment to query and search the metadata in various ways, to control recordings that are in progress, to stream or download recordings, and to bulk-delete recordings which meet certain criteria, as well as to apply custom tags to individual recording sessions. A Symmetric Web Services (SWS) eventing capability is also provided, which allows server-based clients to be notified when recordings start and stop, and when disk space usage exceeds thresholds, and when meta-information about particular recording sessions is updated. Clients may use these events to keep track of system activities and to trigger their own actions.

Taken together, these Cisco MediaSense capabilities target three basic use cases:

1. Recording of conversations for regulatory compliance purposes ("compliance recording")
2. Capturing media for transcription and speech analytics purposes
3. Capturing of individual recordings for podcasting and blogging purposes ("video blogging")

Compliance recording may be required in any enterprise, but is of particular value in contact centers where all conversations conducted on designated agent phones must be captured and retained, and where supervisors need an easy way to find, monitor and play conversations for auditing, training, or dispute resolution purposes. Speech analytics engines are also particularly well served by the fact that Cisco MediaSense maintains the two sides of a conversation as separate tracks and provides access to each track individually, greatly simplifying the analytics engine's need to identify who is saying what.

# Characteristics and Features

This section provides design-level details on compliance recording, direct inbound/outbound recording, and monitoring using Cisco MediaSense.

## Compliance Recording

In *compliance recording*, all calls which are received by or initiated by designated phones are recorded. Cisco MediaSense does not itself control which calls are recorded. Individual lines on individual phones are enabled for recording by configuring them with an appropriate Recording Profile in Unified Communications Manager. Compliance recording is distinguished from *selective recording*, in which the recording server would have the ability to determine which calls it will record, or even to select a new or existing call and initiate recording on it. Cisco MediaSense does not currently support selective recording.

Recording is accomplished by "media forking" – the phone in effect sends a copy of the incoming and outgoing media streams to the Cisco MediaSense recording server. When a call originates or terminates at a recording-enabled phone, Unified CM sends a pair of SIP Invites to both the phone and the recording server. The recording server then establishes a pair of connections between the phone and itself.

This procedure has several implications:

- Each recording session consists of two media streams: one for media which arrives at the phone, and one for media which emanates from the phone. These two streams are captured separately on the recorder, though both streams (or "tracks") are guaranteed to end up on the same Cisco MediaSense recording server. (Further implications of this dual track architecture are discussed below.)
- Most, but not all Cisco IP phones support media forking (a list is provided below). Those which do not support media forking cannot be used for recording.
- Though the phones can fork copies of media, they cannot transcode. This means that whatever codec is negotiated by the phone during its initial call setup will be the codec used in recording. Cisco MediaSense supports a limited set of codecs; if the phone negotiates a codec which is not supported by Cisco MediaSense, the call will not be recorded.
- The recording streams are set up only *after* the phone's primary conversation is fully established, and could take some time to complete. There is therefore a possibility of clipping the beginning of each call. Clipping is typically limited to less than two seconds however, but it can be affected by overall Unified CM and Cisco MediaSense load. Cisco MediaSense carefully monitors this latency and raises alarms if it exceeds certain thresholds.

As mentioned above, Cisco MediaSense does not itself initiate compliance recording; it only receives SIP Invites from Unified CM and is not involved in deciding which calls should or should not be recorded. Unified CM on the other hand, decides on behalf of each individual phone line whether it should be recorded, without considering the other endpoints in the call. This gives rise to the possibility that some calls may be recorded twice, with neither Unified CM nor Cisco MediaSense being aware of that fact while it is happening. Such would be the case if, for example, all contact center agent phones are configured for recording, and one agent calls another agent. That said, Cisco MediaSense stores enough metadata that a client can invoke a query to locate duplicate calls, and selectively delete one copy if so desired.

Also, note that only audio streams can be forked by Cisco IP phones at this time. Compliance recording of video media is not supported.

### Transfers and Conferences

Cisco MediaSense recordings are made up of one or more *sessions*. As described above, each phone-based media forking session contains two media streams, one for the incoming stream and one for the outgoing stream. A simple scenario consisting of a straightforward two-party conversation is represented entirely by a single session. A multi-party conference is no different: the forking phone still receives only one incoming stream (representing the other parties on the conference bridge) and sends one outgoing stream. There is an indication in the metadata that one of the streams represents a conference bridge, but Cisco MediaSense does not have a record of the full list of conference participants.

Transfers can cause a conversation to be split into multiple sessions. If the far end (the phone which is not forking media) transfers to a different endpoint, then the same session continues, and the conversation still consists of only one session. However if the near end device (the forking phone) transfers, a new session starts. The metadata contains all the information necessary to identify based on API queries which sessions were part of the same conversation.

Finally, when a phone executes a Hold operation followed by a Resume, one session is terminated and a new session is created. As with other multi-session scenarios, the metadata contains the information necessary to associate these two session.

## Direct Inbound Recording

In addition to Compliance Recording, controlled by a Unified CM Recording Profile, recordings can be initiated by directly dialing a number which is associated with Cisco MediaSense. Such recordings are not carried out through media forking technology, and are therefore not limited to Cisco IP phones, and not limited to audio media.  This is in fact how the Video Blogging use case is accomplished.  Even a non-IP phone such as a home phone can be recorded, if it is converted to IP via some sort of gateway.

## Direct Outbound Recording

Using the Cisco MediaSense API it is possible for a client to request Cisco MediaSense to call a phone number. When the recipient answers, his call will be recorded just as if he had dialed the recording server as in direct Inbound Recording. The client can be any device capable of issuing an HTTP request to Cisco MediaSense; for example a button on a web page could cause Cisco MediaSense to call the user's phone, at which point the system would begin recording his message.

## Monitoring

While a recording is in progress, Cisco MediaSense allows it to be *monitored* through a publicly available streaming media player. In order to monitor a call from a streaming media player, a client must specify an RTSP URI. It can obtain that URI either by querying the metadata for it, or by capturing session events. Cisco MediaSense offers an HTTP query API which allows suitably authenticated clients to search for recorded sessions based on many criteria, including whether the recording is active. Alternatively, a client may subscribe for session events, whereby Cisco MediaSense will send it SWS events whenever a recording is started (among other conditions). In both cases, the body passed to the client includes a great deal of metadata about the recording, including the RTSP URI to be used for streaming.

The streaming media players that Cisco has tested with are VLC, QuickTime and RealPlayer. Each of these has tradeoffs however, which should be taken into account when selecting which one to use. Recall that recording sessions are usually made up of two audio tracks. Cisco MediaSense receives and stores them that way and does not currently support real time mixing. VLC is capable of playing only one track at a time. The user can alternate between tracks, but cannot hear both simultaneously. On the other hand, VLC is open source, and is easy to embed into a browser page. QuickTime and RealPlayer can play the two streams as stereo – one stream in each ear – but their buffering algorithms for slow connections sometimes result in misleading periods of silence for the listener. People are more or less used to such delays when playing recorded media, but monitoring is expected to be real time, and significant buffering delays are inappropriate for that purpose. Also, none of these players can render g.729 or g.722 audio. A custom application must be created in order to monitor or play streams in those forms. And finally, keep in mind that only calls which are being recorded are available to be monitored. Customers who require live monitoring of unrecorded calls, or who cannot live with these other restrictions, may wish to consider Unified CM's Silent Monitoring capability instead, as driven through a contact center or other CTI application.

# Playback

Once a recording session has completed, it can be played back on a streaming media player (see discussion above). The process is similar to that for Monitoring: an RTSP URI must first be obtained either through a query or an event.

### Silence Suppression

While recording a call, it is possible to create one or more segments of silence within the recording, for example by invoking the pauseRecording API.  Upon playback, there are various ways to represent that silence.  Different representations are suitable for different purposes.  Via a set of custom header parameters on the RTSP PLAY command, Cisco MediaSense allows the requesting client to specify one of the following:

1. RTP stream pauses for the full silent period, then continues with a subsequent packet whose mark bit is set and whose timestamp reflects the elapsed silent period;
2. RTP stream does not pause, timestamp reflects the fact that there was no pause, but RTP packets contain "TIME" padding which includes the absolute UTC time at which the packet was recorded;
3. RTP stream compresses the silent period to roughly half a second, and in all other respects acts exactly like bullet #1 above.  This is the default behavior.

A detailed explanation of these options is included in the Cisco MediaSense Developer Guide.

In all cases, note that the file duration as returned by the RTSP DESCRIBE command reflects the original, record time duration. It is simply the time the last packet ended minus the time the first packet began. Furthermore, the session duration as returned by the MediaSense API and session events may also differ, since these are based on SIP activity rather than on media streaming activity.

Commercial media players such VLC, QuickTime and RealPlayer will always end up eliciting the default behavior, which is #3 above. However, these players are designed to play music and podcasts; they are not designed to handle media streams which include silence. They will therefore evidence various ill effects, such as hanging, disconnecting, or an inability to properly seek backward and forward in the stream.

## Conversion and Download

Completed recording sessions can also be converted on demand to .mp4 format via an HTTP request. Files in this format can actually carry two audio tracks not as a mixed stream, but as stereo. Alternatively, .mp4 files can also carry one audio and one video track. Once converted, .mp4 files remain in Cisco MediaSense's storage along with their raw counterparts, and are accessible via their own URIs. As with streaming, browser or server-based clients may obtain these URIs by either querying the metadata or monitoring recording events. The URI may then be invoked by the client for playing and/or for downloading. .Mp4 conversion therefore offers a convenient and standards-compliant way to package and export recorded sessions, either for a more natural listening experience, or for long term archiving purposes.

Large scale conversion to .mp4 would take a fair amount of processing power on the recording server however, and may impact performance and scalability. To meet the archiving needs of some organizations, as well as to serve the purposes of those speech analytics vendors who would rather download recordings than stream them in real time, Cisco MediaSense also offers a "low overhead" download capability. This allows clients, again using specific URIs, to download individual tracks in their raw g.711 or g.729 format, unmixed and unpackaged. The transport is HTTP 1.1 Chunked, which leaves it up to the client to reconstitute and package the media into whatever format best meets its requirements.

# Codecs Supported

Cisco MediaSense can accept audio in g.711 ulaw/alaw or g.729a/b, and video in h.264. Note that off-the-shelf streaming media players typically do not support g.729 codecs.

Unified CM does not perform a full codec negotiation with Cisco MediaSense. It first negotiates codecs among the conversation endpoints, and only then initiates a connection to Cisco MediaSense. Cisco MediaSense indicates which codecs it can support, and then Unified CM will only complete the recording setup if the previously selected codec is one of the ones that Cisco MediaSense supports. If it is not, then the call will not be recorded. Therefore, for all phones that need to be recorded, it is important to configure them such that the codec that gets selected for the phones can only be one of the codecs that Cisco MediaSense supports.

Some of the newer Cisco IP phones support g.722, and for those phones Unified CM prefers to negotiate the g.722 codec if at all possible. However, since Cisco MediaSense does not yet accept g.722, it must be disabled for recording enabled devices in Unified CM's service parameter settings.

Once the recording codec has been established, it cannot be changed. Unified CM currently does not support codec change in the middle of a recording session. If a remote party transfers the call to a phone which cannot accept the previously selected codec, then Unified CM will automatically insert a transcoder between the two phones. If no transcoder is available, it will drop the transferred call and terminate the recording.

# Metadata Database and the Cisco MediaSense API

Cisco MediaSense maintains a database containing a great deal of information about recorded sessions. The database is stored redundantly on the Primary and Secondary servers. The data includes, among other things:

- Various track, participant, call and session identifiers
- Time stamps and durations
- Real time session state
- URIs for streaming and downloading in various formats
- Server address where recorded files are stored

## Session Tags

Along with the above information, Cisco MediaSense also stores *tags* with each session. Tags are brief, arbitrary text strings that a client can specify and associate to individual sessions using the Web 2.0 APIs, and optionally to specific time offsets within those sessions. Timed session tags are useful for identifying points in time when something happened, such as when a caller became irate or an agent gave erroneous information. Untimed session tags may be used to attach notes which are applicable to the entire session, such as a contact center agent ID, or to mark or categorize some sessions with respect to other sessions. Cisco MediaSense itself also uses the tagging facility to mark when certain actions occurred, such as pause and resume.

# Cisco MediaSense API

The Cisco MediaSense API offers a number of methods to search and retrieve information in the metadata database. Suitably authenticated clients may perform simple lookups, such as finding all sessions which have been deleted by the automatic pruning mechanism, or all sessions which are tagged with a certain string. The API also supports much more complex queries, as well as a sorting and paging scheme by which only a selected subset of the result set will be returned.

The API provides access to a number of other Cisco MediaSense functions as well. Clients can use the API to subscribe for events, to manage disk storage, to manipulate recording sessions which are in progress, and to invoke operations such as conversion to .mp4. Lengthy operations are supported as well through a remote batch job control facility. The API is described in detail in the Cisco MediaSense Developer Guide.

Cisco MediaSense API interactions are conducted entirely over HTTPS, and require that clients be authenticated. Depending on the type of request, clients will use either POST or GET methods. Response bodies are always delivered in JSON format.

API requests may be addressed to either the Primary or the Secondary server, though the client needs to authenticate to each server separately, and to provide the HTTP session identifier which was obtained from the server being addressed.

# Events

The Cisco MediaSense eventing mechanism is designed to provide server-based clients with immediate notification when actions of interest to them take place. The following types of events are supported:

- Session Events - when recording sessions are started, ended, updated, deleted or pruned
- Tag Events - when tags are attached to or removed from recorded sessions
- Storage Threshold Events - when disk space occupancy rises above or falls below certain preconfigured thresholds

Session events provide all of the critical information about a session given its current state, which a client could use to offer various interesting and powerful applications. A client could for example, use the URIs provided in these events in order to offer monitoring and control buttons to an auditor or contact center supervisor. It might also implement a form of *selective* recording (as opposed to *compliance* recording), by deleting, after the fact, sessions which it decides did not need to be recorded.

Tag events might be used as a form of inter-client communication: when a session is tagged by one client, all other subscribed clients hear about it.

Storage Threshold events are useful in allowing a server-based client application to manage disk usage. The client would typically subscribe for these events, and selectively delete older recordings when necessary, according to its own rules. For example, during the normal course of operations, the client might use the tagging facility to mark selected sessions for retention, and then when a threshold event is received, delete all sessions older than a certain date, but skipping over those that have been tagged for retention.

Events are populated with JSON formatted payloads, and delivered to clients using a Symmetric Web Services protocol (SWS), which is essentially a predefined set of HTTP requests sent from Cisco MediaSense to the client (note that HTTPS is *not* currently supported for eventing). When a client subscribes, it provides a URL to which Cisco MediaSense will address its events, as well as a list of event types or categories in which it has an interest. Any number of clients may subscribe, and clients may even subscribe *on behalf of* other recipients (i.e., the subscribing client may specify a host other than itself as the event recipient). The only restriction is that there cannot be more than one subscription to the same URL.

Events are always generated by either the Primary or the Secondary server. When both are deployed, each event is generated on one server or the other, but not both. This has implications for high availability which will be discussed below; for now suffice it to say that customers must choose one of two modes of event delivery – one which favors reliability, and one which favors convenience.

# Disk Space Management

As on any recording device, disk space is a critical resource. Cisco MediaSense provides a number of features designed to meet various, sometimes conflicting space management needs.

At a high level, two space management operating modes are available, to be selected once per deployment:

- Recording Priority
- Retention Priority

Recording Priority mode is designed for customers who would rather lose an old recording than miss a new one. In this mode, Cisco MediaSense automatically prunes recordings which age beyond a configurable number of days, or when the percentage of available disk space falls to dangerous levels. Retention Priority mode focuses on media retention. In this mode Cisco MediaSense will not automatically prune recordings for any reason. In either mode, Cisco MediaSense will stop accepting new calls if necessary in order to protect the space remaining for calls which are currently in progress. Affected calls will be automatically redirected to another Cisco MediaSense recording server if one is available.

The algorithms are as follows:

Retention Priority behavior

- No automatic pruning takes place
- When a node enters warning condition (75%) an alarm will be raised
- When a node enters critical condition (90%) it will redirect new calls
- When a node enters emergency condition (99%) it will drop active recordings
- When a node exits critical condition (drops below 87%) it will start taking new calls

Recording Priority behavior

- Automatic age-based pruning is in effect: recordings older than a configurable number of days are automatically pruned
- When a node enters warning condition (75%) an alarm will be raised
- When a node reaches critical condition (95%) older recordings (even if younger than the age threshold) will be pruned to make room for new ones
- When node enters emergency condition (99%) it will redirect new calls and forcibly drop ongoing recordings
- When a node exits the critical condition (drops below 87%) it will start taking new calls

Any automatic pruning applies only to raw recording files. Converted .mp4 recordings, as well as any metadata associated with pruned recordings, should be deleted explicitly.

For this and other reasons, clients have the option of managing disk usage directly, and in fact *must* do so under Retention Priority mode. Cisco MediaSense therefore takes progressively more aggressive action when storage levels reach successively more dangerous levels, but as each stage is entered or exited, it publishes an event to subscribed clients. These events inform the client when space management actions are necessary, and the Cisco MediaSense API offers a number of ways in which the client can choose which recordings should be deleted, including an option to issue a customized bulk delete operation which is then carried out without client involvement.

The ability to explicitly delete old recorded sessions is not limited to automatic operations performed by a server based client. A customer may wish to take a completely manual approach, designing a web page which fetches and displays appropriate meta-information about older recordings, and allows an administrator to selectively delete those which he considers to be expendable. This web page would use the same API as the server-based client would use.

# System Resiliency and Overload Protection

Cisco MediaSense keeps track of a number of measurements and statistics about its own performance, and raises alarms when certain thresholds are exceeded. The system also limits the number of simultaneous API requests that it will process, simultaneous recordings that it will accept, and the percentage of disk usage it will allow. When these limits are exceeded, Cisco MediaSense rejects new incoming recording requests, causing Unified CM to deliver the call to another recording server if one is available.

Note however, that these hard limits are significantly higher than Cisco MediaSense's rated capacity. It is necessary to provide this headroom in order to allow for the possibility of natural asymmetry in balancing load across the cluster. So, while Cisco MediaSense nodes will individually protect themselves from *severe* overload, the effects of such protection may include loss of recordings. **It is therefore still the customer's responsibility to ensure that the overall rated cluster-wide capacity is not exceeded.** Capacity specifications can be found in the Scalability and Sizing section below.

# Deployment Models

## Server Models Supported

Cisco MediaSense may only be deployed on top of a VMware hypervisor, which must be running on a Cisco B-series server with fiber channel-attached SAN drives, or a C-series server with directly attached hard disk drives (see the Compatibility Matrix section below for versions and model numbers).

When ordering C-series servers, be sure to include battery backed cache. Their RAID controllers have an optional battery backup for the write cache. If the battery backup is not present or not operational, the write cache is disabled on these controllers. When the write cache is disabled, write throughput is significantly reduced. (see the Compatibility Matrix section for detailed disk requirements.)

Finally, the Primary and Secondary server nodes (see below) *must* be based on identical hardware, or at least have identical specifications in terms of CPU speed and disk I/O throughput. They must also be using the same version of VMware ESXi. Any asymmetry will cause accumulating database latency in one server or the other. Expansion servers do not need to be running on the identical hardware.

### Server Node Types

Cisco MediaSense is deployed on up to five nodes depending on the capacity and degree of redundancy required. (For the purposes of this discussion, "node" refers to a virtual machine, not necessarily a physical machine.) There are three types of nodes:

- Primary: Supports all database operations as well as media operations.
- Secondary: Provides high-availability for the database. Also supports all database operations as well as media operations.
- Expansion: Provides additional capacity for media operations, but not for database operations.

Only the Primary node is required, and indeed a small lab setup would likely consist of only that node. The following diagram depicts this deployment model:

Unable to render embedded object: File (SRND graphics - Deployments-1.png) not found.

Customers who require database redundancy would then deploy a Secondary node as well, as shown below:

Unable to render embedded object: File (SRND graphics - Deployments-2.png) not found.

If additional recording capacity is required, Expansion nodes would be deployed, as follows:

Unable to render embedded object: File (SRND graphics - Deployments-3.png) not found.
Unable to render embedded object: File (SRND graphics - Deployments-4.png) not found.
Unable to render embedded object: File (SRND graphics - Deployments-5.png) not found.

All nodes are deployed on the same hardware and virtual machine, and all are provided with the same installed software. They differ only in function. The Primary node is always the first node to be installed, and is identified as such during the installation process. Secondary and Expansion node are identified during the initial configuration process, after installation has completed.

Each node adds both simultaneous activity capacity – the ability to perform more parallel recording, monitoring, download and playback activities – and storage capacity. Recordings are always stored on the disks which are attached to the node which initially captured the media.

On two-server deployments it is possible to replace the Secondary node with an Expansion node. However, there is little reason to do so, except perhaps in lab environments.  The amount of recording capacity to be gained at the expense of database redundancy is minimal, and usually does not outweigh the resulting loss of data safety. Beyond two servers, both a Primary and a Secondary node are required.

## Very Large Deployments

Customers who require capacity which exceeds that of a single Cisco MediaSense cluster must deploy multiple independent MediaSense clusters. The phones should be partitioned among MediaSense clusters, such that any given phone will be recorded by one specific MediaSense cluster; its recordings will never be captured by a server in another cluster.

## Virtual Machine Configuration

Cisco provides several .ova virtual machine template files for your use, specifying the supported VMware virtual machine configurations in this release for Cisco MediaSense servers. These templates specify, among other things, an 8GB memory footprint, and a requirement for seven of the eight available CPUs on a C-series or B-series server (the 8th CPU is dedicated to VMware). Only one virtual machine is allowed per server.

It is a requirement that you use these Cisco-provided templates in your deployment.

Templates also specify the amount of disk space to reserve for use by the database and software binaries, but they do not specify the media storage partitions themselves.  Media storage should be provisioned after the templates has been applied, but before the Cisco MediaSense software is installed.

Four templates are provided.  One of these is to be used for the Expansion nodes, and because it does not itself specify the media storage, it is media-storage-size agnostic and should be used for all Expansion nodes.

The other three templates are for use on the Primary or Secondary nodes.  Each is designed for a specific maximum amount of media partition space cluster-wide, and configures the metadata database partitions according to that value.  One of these templates should be selected, and applied to both the Primary and Secondary nodes.

*Note: After Cisco MediaSense software installation it is not possible to add storage or change any storage-related settings.* Be sure to perform a thorough analysis of all storage requirements prior to installation.

### Storage Alternatives

Two storage alternatives are currently supported:

- Physically attached disks, offering up to 4TB of storage per node in two virtual disks; and
- Fiber channel-attached SAN, offering up to 12TB of storage per node, in six virtual disks

Across five nodes, a maximum of 60TB of storage is available. It is not necessary for all nodes to be configured with the same number or type of virtual disks.

## Geographical Specifications

All Cisco MediaSense nodes must be in a single *campus network*. A *campus network* is defined as a network in which the maximum round-trip delay between any pair of Cisco MediaSense servers is less than 2 milliseconds. (Some Metropolitan Area Networks (MANs) may fit this definition as well.) Media forking phones, however, may be connected over a WAN. Some or all of the nodes in the Unified CM cluster may also communicate over a WAN with the Cisco MediaSense servers, but it should be expected that affected calls may evidence additional clipping at the beginning of recordings, due to the increased round trip delay.  API and administrator sign-in times may also be delayed if the Unified CM server which hosts the AXL authentication service is located over a WAN. Also, from a high availability standpoint, be aware that API sign-in has a dependency on the WAN being functional.

# High Availability

Cisco MediaSense implements a redundant, highly available architecture. Under normal operation, all deployed servers are always fully active. The following sections describe various aspects of this design.

## Recording Server Redundancy - New Recordings

As mentioned elsewhere, a Cisco MediaSense cluster may contain up to five servers, each capable of recording up to a specific number of simultaneous calls. Unified CM should be configured such that it sends recordings to each node in succession, in round robin fashion. This ensures that recording nodes are fully equal to each other in terms of initial SIP Invite preference, and avoids situations in which one node receives the bulk of Invites, and therefore causes a disproportionate amount of impact in case it fails.

At the same time, Cisco MediaSense recording nodes also attempt to distribute the load as evenly as possible among themselves. Each node automatically redirects incoming Invites to other nodes as necessary so that all nodes receive a more or less equal number of recordings. The algorithm used is aware of the state of all recording servers within the cluster and will not direct recordings to failed nodes. It also ensures that the two media streams associated with a given call are recorded on the same node.

If any recording server is down or its network is disconnected, it cannot respond to Unified CM's SIP Invite. Unified CM's usual SIP processing in this case is to deliver the Invite to the next node in the list, thereby also implementing a form of redundancy. However, Unified CM must wait for a timeout to expire before determining that it must try another node. The SIP specification actually envisions it trying the same node several times, with progressively growing timeouts, before determining that the targeted node is unavailable. Since Unified CM only involves recording servers *after* the primary media path has already been established, such operations can clearly take much too long for the resulting recording to be useful. Unified CM in fact sets a time limit beyond which, if the recording hasn't begun, it will stop trying. The net result is that if Unified CM selects a recording server which is not responding, the call in question will most likely not be recorded.

To reduce the likelihood of losing recordings due to a recording server failure, Cisco MediaSense and Unified CM also support a facility known as "SIP Options Ping". This allows Unified CM to periodically probe each recording server to make sure it is up and running, without having to do so while a conversation is literally waiting to be recorded. Once Unified CM is aware that a given Cisco MediaSense server is not running, it will skip that server as it traverses its round robin list of recording servers, thereby distributing the incoming load across the remaining servers. The Cisco MediaSense Installation and Configuration Guide contains instructions for configuring the SIP Options Ping facility as well as other Unified CM SIP parameters.

In contrast, if they are at all able to do so, recording servers which are unable to accept calls due to low disk space or other conditions will actively *redirect* SIP Invites onto alternative servers. There is very little penalty in terms of delayed start of the recording.

Finally, from a sizing perspective, be sure to provision enough recording ports so that if one node fails, you still have enough capacity to capture all the expected concurrent calls. Similarly, the amount of storage space available for recording session retention will also be impacted.

## Recording Server Redundancy - Recordings in Progress

If a recording server fails, all calls which are currently being captured on that server are changed from an ACTIVE state to an ERROR state, and the contents are discarded. Note that the detection of such failed calls, and therefore the state change, may not occur for some time, on the order of an hour or two.

There is currently no ability to continue in-progress recordings on an alternate server.

## Recording Server Redundancy - Saved Recordings

After a recording is completed, Cisco MediaSense retains that recording on the same server which captured it. If that server goes out of service, then none of its recordings will be available for playback, conversion, or download during that period, even though information about them can still be found in the metadata.

## Metadata Database Redundancy

The Primary and Secondary nodes (which we will call the "metadata nodes" in this section) each maintain a database for metadata and configuration data. They also each implement the Cisco MediaSense API, including the ability to publish events to subscribed clients. Once deployed, the two metadata nodes are fully symmetric: the databases are fully replicated such that writing to either one causes the other to be updated as well. Clients may also address their HTTP API requests to either node, and use the alternate node as a fallback in case of failure.

If either the Primary or Secondary does fail, then the surviving node remains available for use. Once the failed node returns to service, the data replication mechanism automatically begins its catch-up operation without any user intervention required. Depending on the duration of the outage and the amount of churn during the outage, this could take some time, during which some irregularities and inconsistencies between the two nodes may be noticed.  It is therefore not advisable to use the recovering node until the catch-up operation has completed, a state which can be determined manually via CLI commands.  Also, there is a limit to how much data can be transferred in this way, and there is no easy way to determine the amount of time which can pass before that limit is reached. If at all possible, keep server outages to within 24-hours in duration.

The eventing mechanism also deserves some discussion. An event is generated by one metadata node as a result of an action which took place on that node. It is not generated by both metadata nodes. For example, if a recording server begins a recording, it initiates a session record in *one* of the metadata nodes. Though the database update is shared with its peer, only that one metadata node generates the event. This holds true for all types of events, from recording session events to disk storage threshold events.

A client cannot know ahead of time which node will generate the events it is interested in. Each client must therefore subscribe to *both* metadata nodes in order to be sure it receives all events (the two subscriptions may designate the same target URI, however, which does simplify things somewhat).

As a convenience, Cisco MediaSense provides the ability for each metadata node to itself subscribe to events which are generated by the other, and forward them to subscribers almost as if they had been generated locally (a flag is included in the event body which identifies such forwarded events). This capability can be enabled in the Cisco MediaSense Administration facility. If enabled, a client need only subscribe to one metadata node or the other. That said, doing so may sacrifice reliability. If the client's chosen metadata node goes down, the client must quickly subscribe to the alternate node in order to avoid any missed events. The risk here should not be underestimated, especially considering that there is no reliable way for the client to detect such a loss without periodically issuing subscription verification requests.

## Backup and Restore

As with most Cisco Unified Communications products, Cisco MediaSense provides a cluster-wide backup and restore capability. The backup includes the entire metadata database, but it does not include any of the media files. Customers are encouraged to use VM-level backups in order to achieve full protection; however use of VMware's snapshot restore capability is not currently supported. If selective preservation of media files is all that is required, then those files can be converted to .mp4 and downloaded to a separate (customer-provided) server for backup.

## Network Redundancy

Network redundancy capabilities such as NIC Teaming may be provided at the hardware level, and are managed by the hypervisor. Cisco MediaSense itself plays no role in network redundancy, and is completely unaware of it.

# Security

User Administration and Authentication

Cisco MediaSense makes use of Unified CM's user administration. Any users configured as End Users in Unified CM may be selectively enabled as Cisco MediaSense API users, and once signed in any such user can access all API functions. Unified CM's AXL service is used for authentication. There is only one Cisco MediaSense administration and serviceability user however, whose credentials are configured during installation. Cisco MediaSense does not currently offer support for multiple roles and authorization.

Cisco MediaSense API and Events

Cisco MediaSense API interactions are conducted entirely over secure HTTPS. All API requests must be issued under the auspices of an authenticated session, denoted through a JSESSIONID header parameter. Authentication is accomplished through a special sign-in API request; user IDs and passwords are configured through the Cisco MediaSense Administration web pages. However, SWS events may only be delivered to clients using HTTP; HTTPS is not currently supported for eventing. By default, Cisco MediaSense uses self-signed certificates; however customers may install their own certificates if desired.

Internal Intracluster Communication

For their own purposes, various components in a Cisco MediaSense cluster communicate with each other over unencrypted HTTP. The specifications for these interactions are not publicly documented, but they nevertheless cannot be considered to be secure.

URIs

A number of session- and track-specific URIs may be associated with each recorded session. These URIs are not secure, however. Once any client knows the URI, it does not need to be authenticated in order to use it. This leaves open the possibility of URIs being transmitted insecurely by people and equipment which are out of Cisco MediaSense's control, and then used inappropriately.

Media

Media encryption is currently not supported. Media is not stored on disk in any encrypted form, nor is it transmitted as Secure RTP (sRTP).

# Reporting and Call Correlation

Unified MediaSense does not offer a direct reporting interface for historical data, nor does it offer SQL access into its schema. However, most historical information is available via the MediaSense API query interface and through its eventing, and is thereby suitable for both browser-based and server-based client use.

The information available in Cisco MediaSense's metadata database is limited to a) that which is provided in Unified CM's SIP Invite; b) tags which are inserted by client applications; and c) information which is generated within Cisco MediaSense itself. The best way to correlate calls between Unified CM's CDR records and Unified MediaSense is to key on the Unified CM Leg IDs. In Unified CM's CDR table, these fields are known ad orig- and destLegCallIdentifier. In Unified MediaSense, they are the xRefCi fields in each Participant object.

In Unified CCE contact center environments, the best call identifier to use is the 32-character string which is known as the CallReferenceID in the UCCE's TerminationCallDetail table. This is also available to the live routing script under the name Call.CallGUID. Unified CM knows this identifier as the incomingProtocolCallRef in its CDR records. However, Unified MediaSense does not have access to this identifier. Therefore, only an indirect correlation can be performed, by joining the Leg IDs to the CallReferenceID through a CDR lookup in Unified CM.

For real time correlation, calls can be associated using the device extension. This value is available in both Unified MediaSense and Unified CCE. However, the correlation is somewhat non-deterministic in complex environments. This is because in Unified CM, *lines* are configured for recording, not *devices* or *extensions*, and there is not necessarily a one-to-one correspondence. If a given phone happens to have two lines or two extensions, or a given extension happens to be assigned to two phones, then some ambiguity can result.

# Serviceability and Administration

Cisco MediaSense offers a web-based user interface for administrative activities such as adding and configuring Cisco MediaSense servers, managing users, checking and configuring storage management parameters, and so forth.

It also provides a number of entry points to support system *serviceability* functions, covering all the functions necessary to service the product. Cisco MediaSense offers most of these functions through the Real Time Monitoring Tool (RTMT), which is similar to Unified CM and other Cisco voice products. RTMT is a thick client which can be downloaded onto any Microsoft Windows system (other operating systems are not supported) from the Cisco MediaSense serviceability web pages. It provides the following capabilities:

- Collecting log files of specific types and specific time periods from some or all Cisco MediaSense nodes. Remote log browsing is also available so that logs can be viewed without having to download them
- Displaying Alerts, including the current set of System Conditions. System Conditions are service impacting conditions, such as temporary or permanent outage of a server or critical subsystem, an overload condition, or loss of connectivity to a dependent service. Events which raise or clear System Conditions may also be sent to a SYSLOG server, or trigger proactive notifications to be sent by email.
- Displaying and graphing a large array of both system and application level counters, statistics, and performance measurements including, for example, the amount of space in use on the media partitions. RTMT also allows thresholds to be configured for these values; crossing such a threshold creates an entry on the Alerts screen. As with System Conditions, these alerts may also be sent to a SYSLOG server, or trigger proactive notifications to be sent by email.

Separate from RTMT, Cisco MediaSense provides a specialized browser-based Serviceability User Interface which provide administrators with the following capabilities:

- Starting, stopping, activating and deactivating individual services;
- Selecting the level and type of information which gets written to log files;
- Requesting heap memory and thread dumps;
- Accessing other Cisco MediaSense nodes in the cluster; and
- Downloading RTMT for Windows

Finally, Cisco MediaSense supports a command line interface (CLI) for many additional service functions. Administrators of the Unified CM will already be familiar with most of these functions.

Note that SNMP is not supported at this time.

# Best Practices

This section describes some best practices for consideration when preparing a Cisco MediaSense deployment.

## Proactive Storage Management

As mentioned elsewhere in this document, Cisco MediaSense offers both Retention Priority and Recording Priority storage retention modes. Under Retention Priority, clients are required to manage the space available for recordings, because the system will not perform any automatic pruning. Under Recording Priority, clients are not required to manage space explicitly. However, as a best practice, some amount of proactive management is recommended. Specifically:

- When sessions are pruned, the metadata associated with those sessions remains in the database, though they are marked as

"pruned". This metadata does not take a large amount of storage space compared to the recordings themselves, but it does take some, and over time it will grow unbounded if not periodically cleaned up. These pruned records generate other unnecessary overhead as well, such as data sorting, searching and filtering resources within both the database and the API client. Once these records are no longer needed, they should be periodically removed.
- Also, converted .mp4 files are not automatically removed by this pruning process. Clients should generally contrive to delete them once they have been offloaded to long term storage, or their retention is otherwise no longer required.
- To aid in both of these activities, clients may periodically issue an API request for pruned sessions, or may elect to receive session pruned events, and explicitly delete those it no longer needs.

These session management activities should be invoked using the Cisco MediaSense API; details can be found in the Cisco MediaSense Developer Guide. If these activities are going to be performed regularly, it is advisable to schedule them for low usage periods in order to minimize possible impact on normal operations.

## Codec Configuration for Phones

Configure phones to negotiate g.711 or g.729 only. Some of the newer Cisco IP phones support g.722, and for those phones Unified CM prefers to negotiate the g.722 codec if at all possible. However, since Cisco MediaSense does not yet accept g.722, it must be disabled for recording enabled devices in Unified CM's service parameter settings.

## Using Scalable Queries

Cisco MediaSense offers an API for searching the metadata in a very flexible manner. While many queries will execute with little or no impact on the normal operation of the Cisco MediaSense servers, it is possible to formulate queries that have a significant impact. Cisco MediaSense limits the *number* of simultaneous queries it will process, but does not consider the relative cost of each individual query. Customers who use the query APIs should therefore read and adhere to the guidelines for writing scalable queries, which can be found in the Cisco MediaSense Developer Guide.

## Alarm Monitoring

Various situations which require administrator attention cause alarms to be raised in the form of System Conditions. These can be observed in the system logs, as well as in RTMT's alarms page. Using RTMT, it is also possible to configure these alarms to be sent to a SYSLOG server, and to send email messages to a designated email address.  Cisco MediaSense does not currently support SNMP alarms.

Cisco recommends that one or more of these methods be used to actively monitor the state of the Cisco MediaSense servers.

# Solution Environment

The following diagram depicts the Cisco MediaSense solution environment:

Unable to render embedded object: File (SRND graphics - Solution Environment.png) not found.

Though this diagram shows only one Cisco MediaSense server and one Unified CM server, each should be considered as a cluster of such servers. That is, one cluster of Cisco MediaSense servers interacts with one cluster of Unified CM servers. From a Unified CM perspective, there is no concept of a hierarchy of recording servers. SIP Trunks should be configured to point to all Cisco MediaSense servers.

For compliance recording applications, call recordings are initiated via a SIP Invite from Unified CM to Cisco MediaSense, once the initial call has been established between two parties. Inbound blog recordings are initiated in a similar way: a SIP Invite is sent from Unified CM to Cisco MediaSense. Outbound blog recordings are initiated via an API request to Cisco MediaSense, which triggers an *outbound* SIP Invite from Cisco MediaSense to Unified CM. The processing of the Invite results in one or more RTP media streams being established between the phone being recorded and Cisco MediaSense. These call flows are depicted in the following figures (Note: these figures are illustrative only and are not intended to show the detailed message flow.)

| | |
|---|---|
| **Compliance Recording** | Unable to render embedded object: File (SRND_Diagrams_Compliance_Recording_1.png) not found. |
| **Inbound Blogging** | Unable to render embedded object: File (SRND_Diagrams_Inbound_Blogging_1.png) not found. |
| **Outbound Blogging** | Unable to render embedded object: File (SRND_Diagrams_Outbound_Blogging_1.png) not found. |

Live monitoring is accomplished when a workstation running a streaming media player sends an RTSP:// URI to Cisco MediaSense, specifying a "live" media address; the RTP media stream is then established between Cisco MediaSense and the player. This stream is actually a *copy* of one of the streams that Cisco MediaSense is receiving from the phone. The media does not come from the disk.

Playback is initiated when a workstation running a streaming media player sends an RTSP:// URI to Cisco MediaSense, specifying an "archive" media address. The resulting media stream between Cisco MediaSense and the player is read from the disk.

Live monitoring and playback callflows are illustrated below. (Again, these figures are illustrative only and are not intended to show the detailed message flow.)

| | |
|---|---|
| **Live Monitoring** | Unable to render embedded object: File (SRND_Diagrams_Live_Monitoring_1.png) not found. |
| **Playback** | Unable to render embedded object: File (SRND_Diagrams_Playback_1.png) not found. |

The Cisco MediaSense API is accessed from either a server-based or a browser-based client. Server-based clients may subscribe for asynchronous events as well.

## Compatibility Matrix

Platforms

| Cisco Servers | Corresponding VMware Templates |
|---|---|
| B200 M2 TRC #1 (DAS+FC SAN) | T12, T24, T60 |
| B200 M2 TRC #2 (Diskless) | T12, T24, T60 |
| B200 M1 TRC #1 (DAS+FC SAN) | T12, T24, T60 |
| B200 M1 TRC #2 (Diskless) | T12, T24, T60 |
| C210 M2 TRC #1 (DAS) | T12, T24 |
| C210 M1 TRC #1 (DAS, single-VM) | T12, T24 |
| C210 M1 TRC #2 (DAS) | T12, T24 |

Hypervisor

| ESXi Version | Notes |
|---|---|
| ESXi v4.0 | -- |
| ESXi v4.1 | With LRO disabled |

*(Key: DAS=Direct Attached Storage; FC=Fiber-Channel; SAN=Storage Area Network)*

Note that a hypervisor his required. Cisco MediaSense is not designed to run on bare metal hardware.

When ordering these servers, be sure to include battery backup for the write cache.

Storage

Up to 60 terabytes of media storage is supported per cluster, divided into up to six 2TB virtual disks in each of five servers. This assumes that all nodes are B-series servers, for which Cisco MediaSense supports SAN drives. C-series servers, for which only direct attached drives are currently supported, are limited to two 2TB media storage partitions each.

- Only fiber-channel attached SAN is supported, and the SAN must be on Cisco's list of supported SAN products (see "Cisco Unified Communications on the Cisco Unified Computing System" at http://www.cisco.com/go/swonly, under the heading "Required Storage").
- Non-media disks (which are the those configured as part of the VMware template) must be at least 15,000 RPM fiber-channel attached disks, running RAID 5 or faster
- Media disks must use disks of at least 7200 RPM, running RAID 5 or faster

Applications

The only application version which Cisco MediaSense depends on, even in complex Contact Center deployments, is Unified CM.

| Application | Releases Supported |
|---|---|
| Unified CM | 8.5, 8.0 with caveats below |

Unified CM Release 8.0 is supported, with the following caveats:

- The metadata does not include extension, device and xRefci fields for the far end device (the device which is *not* the forking device), or the isConference flag
- If the far end of a call is transferred or conferenced, there is no Update event, and the metadata will not reflect the change in participants
- SIP OPTIONS PING is not implemented in Unified CM. This could result in a substantial number of missed recordings if one Cisco MediaSense server fails.

As a result, Cisco does not recommend use of Cisco MediaSense with Unified CM Release 8.0 in production environments.

Phones

All Cisco phones which support media forking and/or video features are supported. Following is a partial list of those devices.

| Endpoint Category | Models |
|---|---|
| Hard Phones | 7906, 7911, 7921, 7925, 7941, 7942, 7945, 7961, 7962, 7965, 7970, 7971, 7975<br>An up to date list may be found under "Unified CM Silent Monitoring and Recording Device Support" at http://developer.cisco.com/web/sip/wikidocs |
| Soft Phones | Cisco IP Communicator (CIPC) v7.0(1) or later |
| Video (for Direct recording only; media forking is not supported) | 9971, 9951 and 7985, plus any audio phone when paired with Cisco Unified Video Advantage (CUVA). A complete list may be found at http://www.cisco.com/go/cuva. |

# Configuration Requirements for Other Solution Components

## Unified CM

Unified CM must be configured appropriately to direct recordings to the Cisco MediaSense recording servers. This includes configuring a Recording Profile, as well as various SIP parameters. Phone zones must be configured to avoid use of g.722 codec on phones whose media will be recorded.  Note that for Cisco MediaSense, SIP over UDP is not supported.

Also, since Cisco MediaSense uses AXL to authenticate users, Unified CM's AXL service must be enabled on at least one of its nodes.

Detailed instructions are available in the Installation and Administration Guide.

## Streaming Media Players

Cisco MediaSense has been tested with the following off-the-shelf media players:

- VLC version 1.0.5 only
- QuickTime
- RealPlayer

Note that none of these players support g.729.  A custom media player is required in order to play media which has is encoded using that codec.  Also, be aware that these media players are not designed to handle silence.  Playback of recordings which include silent segments may produce unpredictable behavior.

## SIP Proxy Servers

Cisco MediaSense is not currently supported with SIP Proxy Servers situated between its recording servers and the Unified CM servers.

## Cisco Unified Session Manager Edition

In Cisco Unified Session Manager Edition (CUSME) deployments, Cisco MediaSense may only be placed at the "leaf" Unified CM cluster level. It is not currently supported at the centralized CUSME level. This means that each leaf cluster requires its own Cisco MediaSense cluster.

## Contact Center Environments

- Cisco MediaSense does not explicitly interact with or support Unified Contact Center Enterprise (Unified CCE) or Unified Contact Center Express (Unified CCX). The recording functions which are available with these products' Agent/Supervisor Desktop clients utilize different mechanisms for initiating and capturing recordings and require their own established recording solutions.
- For supervisor whisper feature, Cisco MediaSense does not record the whisper call between agent and supervisor. This is because the agent phone's build-in-bridge doesn't include the supervisor-to-agent whisper in the forked media stream that it delivers to the recorder.  On the other hand, if the *supervisor's* phone is configured for forking, then the whisper will be included in the supervisor's phone recording.
- Equipment which monitors agent conversations by listening to a span port output and filtering on the agent's phone MAC or IP address may not function properly while the phone is also forking media for recording.  This is because every RTP packet will be emitted from the phone twice, and the listening device may not exclude those packets which are destined for the recording server from its capture.  The result will be an apparent echo observed by the listener.  Please discuss this with your silent monitoring vendor if the application calls for monitoring of conversations which are also being recorded.

# Scalability and Sizing

## Performance

Each Cisco MediaSense server supports up to 340 media streams simultaneously (170 calls), at a sustained busy hour call arrival rate of two calls per second, on up to 12 terabytes of disk space. This number is a total off all streams used for recording, live monitoring, playback, and HTTP download, all of which may occur in any combination. (HTTP download is not a streaming activity, strictly speaking, but it has equal weight from the perspective of resources required.)

Essentially, the number of servers required would simply equal the number simultaneous streams needed divided by 340, or the number of busy hour call arrivals per second divided by two, or the space required for retained recording sessions divided by 12 terabytes, whichever is greater, and rounded up.

Other factors which significantly impact performance are the number of sessions being converted to .mp4 format, and the number of Cisco MediaSense API requests in progress.  The latter (number of API requests) is limited to 15 at a time, with the ability to queue up to 10 more.

Cisco MediaSense currently does not enforce these limits, with the exception of the number of concurrent API requests.  Three of the operations -- monitoring, playback and HTTP download -- are entirely under client control however, so these limits should be enforced by the client. The remaining operation, call recording, is not under client control.  However, the deployment can be sized such that the overall recording load will not exceed a desired maximum cluster-wide (leaving room for an enforceable number of monitoring, playback and HTTP download operations).  It can be assumed that the recording load will be balanced across all nodes.  (Perfect balance will not always be achieved, but each node has enough headroom to accommodate most disparities.)

## Storage

The amount of storage space required depends on a number of factors, such as the mix of codecs in use, the number of such calls, the call arrival rate, duration and duty cycle, and the retention period desired. Since most of these parameters are very difficult to estimate, we will focus here on only the number of recording session hours, and the retention period. Essentially, we will answer this question, "How much disk space do I need in order to retain *h* hours of recordings for *d* days?"

We begin by selecting the codec. We will assume g.711, which requires about 1MB per minute of dual-stream recording. G.729 uses a variable rate compression, which means that the space requirement depends on the content, which is not predictable. On the other hand, for estimation purposes it is generally safe to assume that it requires about one eighth the space needed by G.711, or 128 kilobytes per minute of dual-stream recording. H.264 video is even less predictable; not only does it use a variable rate compression, but it also depends on video resolution, screen dimensions, and a number of other factors. This is best evaluated empirically.

Given g.711 then, we have a rate of 1MB per minute, or 60MB per hour. Assuming the maximum direct-attached storage, 4TB of disk space therefore can store about 70,000 hours of dual-stream recordings. If you have 100 phones which are in active use 80% of the time, 24 hours a day, then you are recording at a rate of 80 hours per hour of elapsed time. That will use up 70,000 hours in 875 hours of elapsed time, or a little over 36 days, after which the oldest calls will need to be pruned. Therefore, your retention period will be 36 days.

Let's say all of the same parameters apply, except that your business is only open 12 hours per day. That will give you a retention period of 72 days.

If you only have 50 agents, active 12 hours a day, then your retention period rises to 144 days.

Finally, all of the above assumes 4TB of storage space available. If you deploy five Cisco MediaSense servers, each with its maximum SAN storage allocation, then you have in effect, 60TB available. In that scenario, the retention period for 50 agents at 12 hours per day with an 80% active usage ratio would be 2160 days, or about six years.

Here is the formula:

```
Codec bit rate (B) in MB/hour for two streams
Number of phones (P)
Average Usage ratio of each phone (U) in hours per day

o Write Rate (W) = B * P * U, in hours of storage per hour of of elapsed time

Total Storage available across all servers (S) in GB

o Retention (R) in hours = S * 1024 / W
```

There is one more factor to consider however: conversions into .mp4. If you expect to be converting a significant number of recorded sessions to .mp4 and leaving them on the server, then you must increase the Write Rate (W) to account for it. In anecdotal trials, .mp4 averaged about 18 MB/hour for dual-channel audio, and about 180 MB/hour for audio+video. (Note that the .mp4 files use AAC, which is another variable rate encoding, so the actual space used may vary considerably.) If you convert and retain an average of 50% of your recorded sessions for example, then you must increase the Write Rate by 50% times the .mp4 bit rate in MB/hour, which obviously reduces the retention period. Thus the Write Rate now becomes:

```
Proportion (M) of recorded session hours which are converted to .mp4 and retained
.mp4 average bit rate (K) in MB/hour

o Write Rate (W) = (B * P * U) * (1 + K * M)
```

## Bandwidth Provisioning

If Call Admission Control (CAC) is enabled, Unified CM automatically estimates whether there is enough available bandwidth between the forking device and the recording server so that media quality for either the current recording or for any other media channel along that path is not impacted. If sufficient bandwidth does not appear to be available, then Unified CM will not record the call; however the call itself does not get dropped. There is also no alarm raised in this scenario. The only way to determine why a call did not get recorded in this situation is to examine its logs and CDR records.

It is important to provision enough bandwidth so that this does not happen. In calculating the requirements, the Unified CM administrator should include enough bandwidth for *two two-way media streams*, even though the reverse direction of each stream is not actually being used.

Bandwidth requirements also depend on the codecs in use, and in the case of video, on the frame rate, resolution and dimensions of the image.