CHAPTER 1

Product Overview

This chapter provides an introduction to the Cisco Internet Streamer Content Delivery System (CDS). This chapter has the following major topics:

Overview

The Cisco Content Delivery System (CDS) is a distributed network of Content Delivery Engines (CDEs) running Content Delivery Applications (CDAs) that collaborate with each other to deliver multi-format content to a variety of client devices. The client devices supported in Releases 2.0–2.3 are personal computers and Wi-Fi-enabled mobile devices, such as personal digital assistants (PDAs).

The CDS supports a variety of mechanisms to accelerate the distribution of content within the content delivery network. The CDS offers an end-to-end solution for service providers to ingest and stream entertainment-grade content to subscribers.

The CDS functionality can be separated into four areas:

- Ingest
- Distribution
- Delivery
- Management

Each CDE in the CDS contributes to one or more of these functions as determined by the CDAs running on it. Table 1-1 describes the relationship between the CDA names and the Internet Streaming Content Delivery System Manager (CDSM) device names.

Table 1-1 CDA Mapping to Functionality and CDSM

CDA Name	Functionality	CDSM Device Name
Internet Streamer (+ Content Acquirer)	Ingest, distribution, and delivery	Service Engine (SE)
Service Router	Redirect client requests for delivery	Service Router (SR)
Internet Streaming Content Delivery System Manager	Management	CDSM

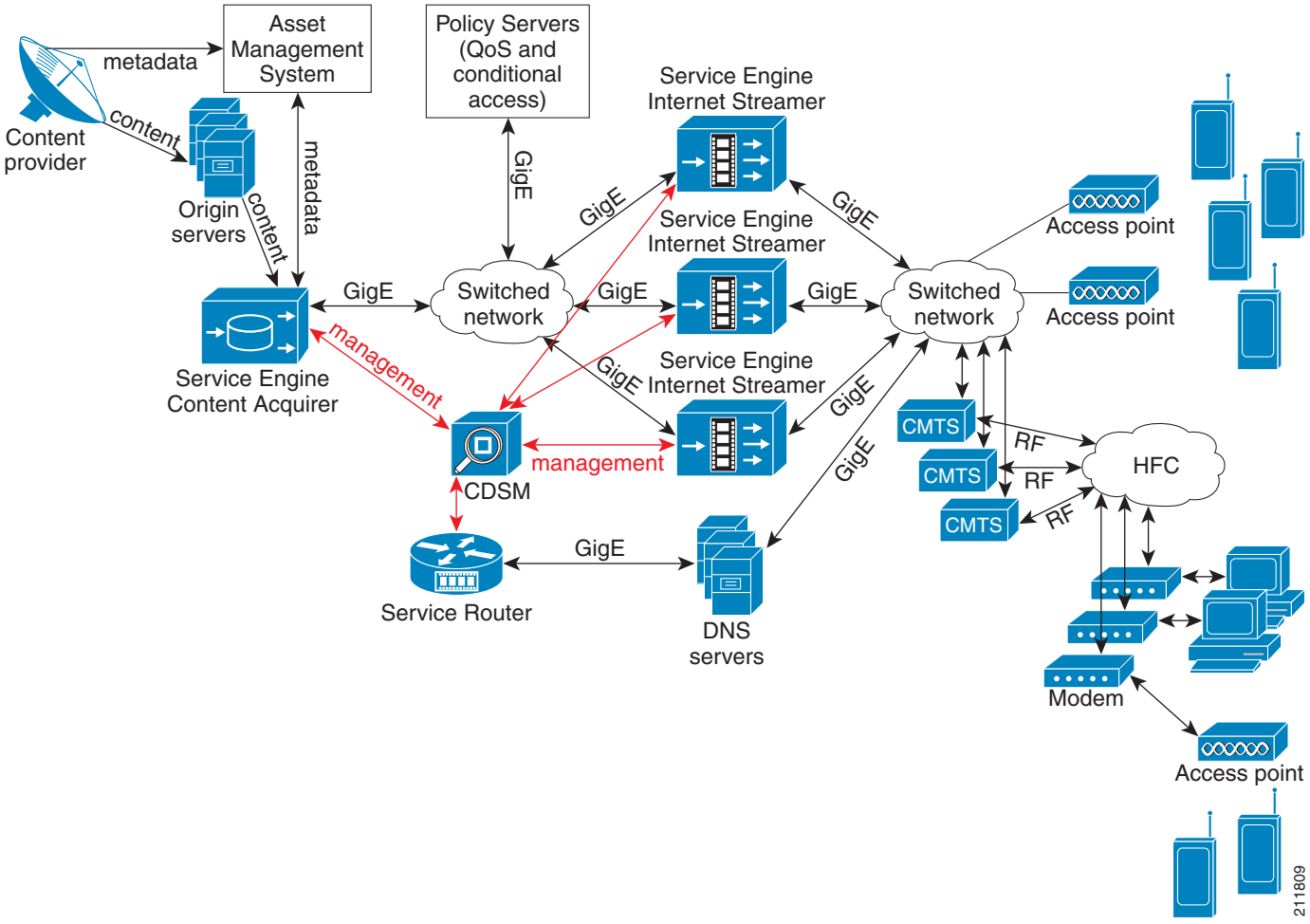
The Service Engine can function as a Content Acquirer and Internet Streamer, or just as an Internet Streamer.

Figure 1-1 shows the major elements of a CDS network. How content flows, from ingest to distribution within the CDS, to delivery to client devices, is dictated by the content delivery services defined for each Content Origin. A delivery service is a configuration defined by using the CDSM and consists of configuration parameters that dictate how content is ingested and distributed, and what content is delivered to the client devices. Some of the primary delivery service definition parameters are:

- Origin server
- Service routing domain name
- Service Engines participating in the delivery service
- Service Engine designated as the Content Acquirer

The Content Acquirer is only active on one Service Engine in each delivery service.

Figure 1-1 High-Level View of the Cisco CDS



The following sections briefly describe the elements of the CDS. For more detailed information, see the "Content Delivery System Architecture" section on page 1-7.

Ingest and Distribution

The Service Engine designated as the Content Acquirer for a delivery service is the ingest device. Cisco Internet Streamer CDS Releases 2.0–2.3 supports the following methods of content ingest:

- Prefetch ingest
- Dynamic ingest
- Hybrid ingest
- Live stream ingest and split

The distribution of content within the CDS is determined by the method of ingest used.

Note The recommended maximum number of prefetched content items is 200,000.

Prefetch Ingest

The Content Acquirer receives metadata from the backoffice in the form of an XML-formatted Manifest file and, using the information in the file, pulls the content into storage on the Content Acquirer. The content can be ingested by using different protocols. The supported protocols are FTP, HTTP, HTTPS, CIFS, as well as local files, which are files copied to the Service Engine. The ingested content is then distributed to all Service Engines in the content delivery service. The content is stored on each Service Engine's hard disk for a configurable amount of time or until the content entry gets deleted from the Manifest file. This is called *content pinning*.

The Manifest file can be used to specify different policies for content ingest and also for streaming the prefetched content. For example, the policy could include specifying the expiry of the content, setting time windows in which the content is made available to users, and so on.

Dynamic Ingest

Content can be dynamically ingested into the CDS. Dynamic ingest is triggered when a Service Engine's Internet Streamer application does not find a client's requested content in its local hard disk storage. All Service Engines participating in the content delivery service coordinate to form a content distribution tunnel starting at the origin server and ending at the Service Engine responding to the client request. As the content flows through this tunnel, the participating Service Engines cache a copy of the content. Subsequent requests for the same content are served off the CDS network. Content ingested and distributed by this method is deleted if clients do not request it frequently.

The Internet Streaming CDSM manages this ingest method internally, not by instructions embedded in a Manifest file, and manages the storage automatically. The Internet Streaming CDSM also provides the ability to purge any dynamically ingested content out of the Service Engines. Content is identified by a URL, which is also used to delete the content.

Hybrid Ingest

The hybrid ingest method provides a very powerful solution by combining the features of the prefetch ingest and the dynamic ingest methods. The metadata and control information about the content, defined in the Manifest file, is propagated and pinned to all Service Engines participating in the content delivery service. However, the content is not prefetched. Ingest occurs upon user request for the content. Content

that is cached on the Service Engines by using this method is subject to the same deletion rules as the dynamic ingest method. The metadata that is propagated can be used to specify explicit controls and policies for streaming the content.

Live Stream Ingest and Split

The live stream ingest method distributes a live content feed to all the Service Engines participating in the content delivery service and helps to scale the content delivery to a very large audience. This method leverages the live stream splitting capabilities of the Internet Streamer application and optimizes the access by doing a one-to-many split to all Service Engines in the content delivery service. The Internet Streaming CDSM provides the necessary interface to schedule the streaming of live programs. Advanced techniques are used to enhance the performance of live streaming.

Delivery

The Service Router handles client requests for content and determines the best Service Engine to deliver it based on proximity, load and health states.

Once the best Service Engine has been determined, the content is delivered to the client device by means of one of the following mechanisms:

- **Static Content Download using HTTP**—Content is downloaded by the client device before it can be rendered to the user.
- **Progressive Content Download using HTTP**—Content is rendered in segments to the user before it has been fully downloaded.
- **Content Streaming using HTTP, RTMP, RTSP, or RTP**—Content is streamed to the client device, Service Engines collect feedback and can fine-tune streaming. Advanced error recovery can also be performed. This is a very common method of streaming video content to client devices.

Table 1-2 lists the content types and formats, content transport protocols, and client types supported by the CDS.

Table 1-2 Supported Content Types

Content Types and Formats	Transport Protocols	Typical Client Types	Access Network Type
Windows Media (WMA, WMV, ASF, and others) VC-1	RTP, RTSP, HTTP	Windows Media Player 9, 10, 11 on PC Windows Media Player 9 for Mac Windows Media Player 9 on PDA running Windows Pocket PC 2002/2003 Windows Media Player 10 on PDA running Windows Mobile 5 Windows Media Technology (WMT) Silverlight	Wired Wi-Fi Cellular
QuickTime (MOV), hinted (3GP) files	RTP, RTSP, HTTP	On PC: QuickTime Player, QuickTime Pro 6 or 7, RealPlayer 10 or 11 (3GP only), VLC player On Mac: QuickTime Player, QuickTime Pro 6 or 7, RealPlayer 10 for Mac OS X (3GP only), On Mobile: RealPlayer on Nokia N series phones (3GP only), PackerVideo player	Wired Wi-Fi Cellular

Table 1-2 Supported Content Types (continued)

Content Types and Formats	Transport Protocols	Typical Client Types	Access Network Type
Other Hypertext and image files (HTML, JPEG, and so on)	HTTP	Web browsers and other HTTP clients	Wired Wi-Fi Cellular
MPEG (MP1, MP2, MP4)	RTP, RTSP	MPEG clients **Note** For Flash Media Streaming, the Adobe Flash Media Player 9 update 3, Adobe Media Player, and Adobe Air, are the only players that support MPEG-4.[1]	Wired
Adobe Flash (SWF, FLV, MP3)	RTMP, HTTP	Adobe Flash Player 9 for Windows, Mac OS, and Linux	Wired Wi-Fi Cellular
H.264[2]	RTMP, HTTP	H.264 clients **Note** For Flash Media Streaming, the Adobe Flash Media Player 9 update 3, Adobe Media Player, and Adobe Air, are the only players that support H.264.	Wired

1. Support for MPEG-4 in the Flash Media Streaming Engine is a Release 2.2 feature.
2. Support for H.264 in the Flash Media Streaming Engine is a Release 2.2 feature.

Note RTMP is part of the Flash Media Streaming feature and is part of Releases 2.0–2.3

3-Screen Session Shifting

Note 3-Screen Session Shifting is a Release 2.2 feature and supports RTSP streaming for the Windows Media Engine and Movie Streamer Engine.

The 3-Screen Session Shifting feature unifies the user interactions with different content streaming engines and enables seamless movement of user sessions to and from different client devices (PC, Mac, Mobile, and TV). With this feature, users can pause a streaming session on one device and resume it on a different device. This feature provides the intelligence for client detection and format selection. The 3-Screen Session Shifting interacts with the TV CDS as well.

3-Screen Session Shifting integrates with the delivery service configuration by linking three XML files that interact with content management and transcoding systems to obtain device-type mappings and content format mapping. 3-Screen Session Shifting works with entitlement servers to get the user identification and uses the user identification as a key to store user sessions.

Session shifting from one client device to another device for a particular user requires the subscriber ID, content ID, and last play time.

The Service Router acts as the centralized session manager for session shifting and maintains the session database. The session database is populated by operator-generated 3-screen maps (XML files), consisting of the Content map, Subscriber map, and Profile map. The XML schema for these different

XML files is available upon request. The operator can enable session shifting on a delivery service basis and also configure a TV streamer IP address and port for each delivery service. The operator also needs to specify which Service Router has the session database.

The Content map aggregates same semantic content across the TV CDS and Internet Streamer CDS systems. Similarly, the Subscriber map helps to correlate different identifiers of the same users between the TV CDS and Internet Streamer CDS.

Note The Fast Cache feature for the Windows Media Engine is not supported in session shifting.

3-Screen Web Services Infrastructure

Note Release 2.2 supports web-based programming APIs. These APIs are based on the REpresentational State Transfer (REST) architecture. For more information, see Appendix D, "Creating and Manipulating Session Shifting Files." In Release 2.2, 3-Screen Session Shifting is the only application that is supported through the Web Services infrastructure.

Following are the Web Services infrastructure features that are supported in this release:

- Content Manager
- Subscriber Manager
- Profile Manager
- Stream Session Event Manager

Content Manager

The Content Manager facilitates the management of 3-screen content for session shifting. Content lists can be posted to the Web Services interface for content addition or deletion. The provisioned content is visible by using a web browser and entering a simple URL. The schema for the ContentList XML file is available upon request.

Subscriber Manager

The Subscriber Manager facilitates the management of 3-screen users for session shifting. Subscriber lists can be posted to the Web Services interface to add or delete subscribers. The provisioned subscribers are visible by using a web browser and entering a simple URL. The schema for the SubscriberList XML file is available upon request.

Profile Manager

The Profile Manager facilitates the management of different client devices and player types for session shifting. This is called *Capability Exchange*. Capability Exchange refers to the process of mapping the logical URL to a specific content format inside the group. One profile groups together multiple clients that have similar characteristics, such as supported file format, and desired resolution or bandwidth. Profile lists can be posted to the Web Services interface for addition or deletion. The profiles are visible by using a web browser and entering a simple URL. The schema for the ProfileList XML file is available upon request.

Stream Session Event Manager

The Stream Session Event Manager allows creation and retrieval of last play times for specific pairs of content and subscribers. The schema for the StreamSessionEvent XML file is available upon request.

Management

The Internet Streaming CDSM, a secure Web browser-based user interface, is a centralized system management device that allows an administrator to manage and monitor the entire CDS network. All devices, Service Engines and Service Routers, in the CDS are registered to the Internet Streaming CDSM.

Service Engines can be organized into user-defined device groups to allow administrators to apply configuration changes and perform other group operations on multiple devices simultaneously. One device may belong to multiple device groups.

The Internet Streaming CDSM also provides an automated workflow to apply a software image upgrade to a device group.

Content Delivery System Architecture

The CDS consists of an Internet Streaming CDSM, one or more Service Engines, and one Service Router. For full redundancy, a CDS would include an additional CDSM and Service Router. The Service Engine handles content ingest, content distribution within the CDS, and content delivery to client devices. The Service Router handles client requests and redirects the client to the most appropriate Service Engine. The Internet Streaming CDSM manages and monitors the CDS, the delivery services, and all the devices in the CDS.

This section covers the following topics:

Service Engine

Each Service Engine can function as a Content Acquirer and Internet Streamer, or just as an Internet Streamer. Based on the Service Engines' assignments to different delivery services, the right set of applications supporting the functions is enabled. For example, only one Service Engine is assigned the role of Content Acquirer in each delivery service. In addition, the Service Engine assigned as the Content Acquirer in a delivery service also includes the functions of an Internet Streamer.

Both the Content Acquirer and the Internet Streamer applications have storage and distribution functions within the CDS, which include the following:

- Management of the physical storage of content and metadata. Content URLs are translated into their physical file paths for content retrieval, deletion, and update.
- Management of dynamically ingested content and periodic replacement of content not accessed frequently. Content replacement is performed by means of sophisticated content-replacement algorithms. The algorithms add "weight" to the content according to size, frequency of access, and other attributes to produce the list of content that needs to be purged.
- Ingest of prefetched content and retrieval of such content for distribution to other Service Engines in the same delivery service.

- Maintenance of information about the entire CDS topology and all the delivery services. This includes upkeep of a list of Service Engines in the same delivery service that is used for distributing prefetched, dynamic, and live stream content.
- Maintenance of the database that stores and distributes metadata about the content, and the topology and delivery service information.
- Distribution of content on a per-delivery service basis, where the flow path of content could differ from one delivery service to another.

Content Acquirer

Every delivery service requires a Content Acquirer, which is a CDA that resides on every Service Engine. The Content Acquirer CDA becomes active when the Service Engine is designated as the Content Acquirer in a delivery service. The Content Acquirer has the following functions and capabilities:

- Fetches content from origin servers using HTTP, HTTPS, FTP, or CIFS (Dynamic ingest supports HTTP only).
- Supports the NT LAN Manager (NTLM) and basic authentication for ingesting content from the origin servers.
- Creates and distributes the metadata for each of the prefetched contents according to the Manifest file and the information returned by the origin server.

Once the Content Acquirer has ingested the content and distributed the metadata, it creates a database record for the metadata and marks the content ready for distribution. All other types of ingest (dynamic, hybrid, and live stream) are handled by the Content Acquirer as well.

Internet Streamer

All Internet Streamers participating in a delivery service pull the metadata from a peer Internet Streamer called a *forwarder*, which is selected by the internal routing module. Each Internet Streamer participating in a delivery service has a forwarder Internet Streamer. The Content Acquirer is the top-most forwarder in the distribution hierarchy. In the case of prefetched ingest, each Internet Streamer in the delivery service looks up the metadata record and fetches the content from its forwarder. For live or cached content metadata, only the metadata is distributed.

The content associated with the metadata for live and cached content is fetched by the specified protocol engine, which uses the dynamic ingest mechanism. When a request for a non-prefetched content arrives at an Internet Streamer, the protocol engine application gets the information about the set of upstream Internet Streamers through which the content can be acquired. In the case of dynamic ingest, the Internet Streamer uses the cache routing function to organize itself as a hierarchy of caching proxies and performs a native protocol cache fill. Live stream splitting is used to organize the Internet Streamers into a live streaming hierarchy to split a single incoming live stream to multiple clients. The live stream can originate from external servers or from ingested content. Windows Media Engine, Movie Streamer Engine, and Flash Media Streaming Engine support live stream splitting.

The Internet Streamers use service control to filter and control incoming requests for content. The service rules and the PacketCable Multimedia (PCMM) Quality of Service (QoS) control are some of the functions that are encapsulated under the Service Control option in the Internet Streaming CDSM.

The Internet Streamers send keep-alive and load information to the Service Router that is participating in the same delivery service. This information is used by the Service Router to choose the most appropriate Internet Streamer to handle the request.

The Internet Streamer function is implemented as a set of protocol engine applications. The protocol engine applications are:

- Web Engine
- Windows Media Engine
- Movie Streamer Engine
- Flash Media Streaming Engine

Web Engine

All HTTP client requests that are redirected to a Service Engine by the Service Router are handled by the Web Engine. On receiving the request, the Web Engine uses its best judgment and either handles the request or forwards it to another component within the Service Engine. The Web Engine, using HTTP, can serve the request from locally stored content in the CDS or from any upstream proxy or origin server.

An HTTP client request that reaches the Service Engine can either be from a Service Router redirect or from a direct proxy request.

On receiving an HTTP request for content, the Web Engine decides whether the content needs to be streamed by the Windows Media Engine, and if so, hands the request over to the Windows Media Engine, otherwise the request is handled by the Web Engine.

Policy Server Integration

The policy control server, a third-party PCMM-compliant system, ensures guaranteed bandwidth for multimedia data delivered over broadband networks. The Web Engine communicates with the policy server by means of Internet Content Adaptation Protocol (ICAP) and HTTP to set and monitor QoS attributes for each client session and whether access should be denied.

Using ICAP, the Web Engine determines whether the bandwidth reservation can be allocated for this client or access should be denied. The policy server uses the cookie generated by the web portal and the client's IP address to make the decision and replies accordingly to the Web Engine.

Using HTTP, the Web Engine communicates the start and teardown of the request to the policy server. Bandwidth reservation is performed when the download starts, and once the download is complete, the Web Engine sends a teardown message to the policy server.

The Web Engine uses PCMM to interact with the policy server to allocate guaranteed bandwidth for authenticated client requests for content. The PCMM integration allows the granting of QoS for the session as well as conditional access protection of the content.

Upon receiving permission from the policy server, the Web Engine generates a URL signature and appends it to the requested URL; it then embeds the new URL in an .asx file and sends the file back to the client. The file consists of the signed URL with RTSP and HTTP options. For RTSP and HTTP streaming, the Windows Media Engine communicates with the policy server for bandwidth commitment. If RTSP and HTTP streaming fail, the client device begins HTTP progressive download of the file. The Web Engine handles the QoS for HTTP progressive download requests.

The signed URL adds additional security. The URL signature generation is based on a key that is a shared secret between the component generating the URL signature and the component validating the URL signature. The URL signature can be generated by the Web Engine, another component external to the Service Engine, or the web portal.

Cache Fill Operations

The Web Engine interfaces with the storage function in the Service Engine to determine whether the content is present locally or whether the content needs to be fetched from either an upstream Service Engine or the origin server.

The Web Engine communicates to the upstream Service Engine for cache fill operations. This interaction is based on HTTP. This cache fill operation is on demand and hence only occurs when the content is not stored locally. The upstream Service Engine can be selected dynamically by means of the Hierarchical Cache Routing Module, or can be configured statically through the Internet Streaming CDSM. The Hierarchical Cache Router generates a list of upstream Service Engines that are alive, ready to serve the request, and part of the delivery service. If the Web Engine is unsuccessful in locating the content on one of these Service Engines, the content is retrieved from the origin server.

Whether the content is found locally or retrieved and stored through the cache fill operation, the Web Engine serves the content based on the following:

- **Freshness of content**—The freshness of prefetched content is governed by a time-to-live (TTL) value set for the content in the delivery service configuration. The TTL specifies the rate at which content freshness is checked. For cached content, which is content ingested by means of the dynamic ingest or hybrid ingest method, the freshness check is performed by the Web Engine in compliance with RFC 2616. For expired cached content, the local copy is deleted and fresh content is ingested.
- **Rate of data transfer**—The rate at which the content is sent can be configured on a per-delivery basis. By default, LAN bandwidth is used.
- **Content completeness**—Prefetched content is stored locally in the CDS in its entirety. For cached content, there are two cases when the content is not complete:
  - The Web Engine process halts or the Service Engine experiences a failure in the process of caching the content. In this case, the subsequent request starts the cache fill anew.
  - The content is in the process of being cached by another request. In this case, the subsequent request is served from the cached content.

Authentication

The Web Engine supports a pass-through mode of authentication, whereby the origin server negotiates authentication and the Web Engine passes the requests and responses between the client device and the origin server. Content that requires authentication is not cached by the Service Engine, so all requests for authenticated content are retrieved from the origin server.

Service Rules

Service rules can be configured that dictate how the Web Engine responds when client requests match specific patterns. The patterns can be a domain or host name, certain header information, the request source IP address, or a Uniform Resource Identifier (URI). Some of the possible responding actions are to allow or block the request, generate or validate the URL signature, send an ICAP request to the policy server, or rewrite or redirect the URL.

Windows Media Engine

The Windows Media Engine uses Windows Media Technology (WMT), a set of streaming solutions for creating, distributing, and playing back digital media files across the Internet. WMT includes the following applications:

- Windows Media Player—End-user application
- Windows Media Server—Server and distribution application
- Windows Media Encoder—Encodes media files for distribution
- Windows Media Codec—Compression algorithm applied to live and on-demand content
- Windows Media Rights Manager (WMRM)—Encrypts content and manages user privileges

The Windows Media Engine streams Windows Media content, with the capability of acting both as a server and as a proxy. It streams prefetched content to the Windows Media Player, acts as a proxy for client requests, splits a live stream into multiple live streams, and caches content requested from remote servers.

Windows Media Engine acts as Windows Media Server for prefetched or cached content stored locally. The request is served by means of RTSP and HTTP. Windows Media Engine checks with the storage function on the Service Engine to see whether the content is stored locally; if the content is not found, the Windows Media Engine engages the Windows Media Proxy.

The WMT Proxy works like the cache fill operation in the Web Engine. There are two options:

- Hierarchical Caching Proxy—If content is not found locally, the Windows Media Engine checks the upstream Service Engines first before pulling the content from the origin server.
- Static Caching Proxy—The administrator statically configures Service Engines as upstream proxies.

The WMT Proxy accepts and serves streaming requests over RTSP and HTTP.

Fast Start

Fast Start provides data directly to the Windows Media Player buffer at speeds higher than the bit rate of the requested content. After the buffer is filled, prefetched, cached, or live content stream at the bit rate defined by the content stream format. Fast Start does not apply to content that is dynamically ingested. Only Windows Media 9 Players that connect to unicast streams using MMS-over-HTTP or RTSP can use Fast Start. The Fast Start feature is used only by clients that connect to a unicast stream. With live content, the Windows Media Engine needs to hold the content in its buffer for a few seconds. This buffer is used to serve Fast Start packets to subsequent clients that request the same stream as the initiating first client request. The first client triggers the process, with the subsequent clients benefitting from Fast Start.

Fast Cache

Fast Cache allows clients to buffer a much larger portion of the content before rendering it. Fast Cache is supported only for TCP. It is not supported for RTP. The Windows Media Engine streams content at a much higher data rate than specified by the stream format. For example, using Fast Cache, the Windows Media Engine can transmit a 128-kilobit per second (Kbps) stream at 700 Kbps. This allows the client to handle variable network conditions without perceptible impact on playback quality. Only MMS-over-HTTP and RTSP requests for prefetched or cached content support Fast Cache. The speed is determined by the client's maximum rate and the configured Fast Cache rate—whichever is smaller.

Fast Stream Start

The first client requesting a live stream often experiences the longest wait time for the content to begin playing. Users can experience long wait times because of the full RTSP or HTTP negotiation that is required to pull the live stream from the source. Delays can also occur if the edge Service Engine has not buffered enough stream data to fill the player's buffer at the time the content is requested. When the buffer is not filled, some data to the client might be sent at the linear stream rate, rather than at the Fast Start rate. With Fast Stream Start, when a live stream is primed, or scheduled and pulled, a live unicast-out stream is pulled from the origin server to a Service Engine before a client ever requests the stream. When the first request for the stream goes out, the stream is already in the delivery service.

Live Stream Splitting

Live stream splitting is a process whereby a single live stream from the origin server is split and shared across multiple streams, each serving a client that requested the stream. When the first client that requested the stream disconnects, the Windows Media Engine continues to serve the subsequent requesting clients until all requesting clients have disconnected. Live stream splitting using content that

is already stored locally is generally better than using content from the origin server; this is because the Service Engine is typically closer to the requesting clients, and therefore network bandwidth to the origin server is freed up.

Note When using Windows Media Server 2008 as the origin server, the source content type must be a playlist or encoder type.

Live stream splitting can either be unicast or multicast, depending on the configuration, capabilities and limitations of the network. The Windows Media Engine can receive and deliver Windows Media content over IP multicast or unicast transmission in the following combinations:

- Unicast-In Multicast-Out
- Multicast-In Multicast-Out
- Unicast-In Unicast-Out
- Multicast-In Unicast-Out

Multicast-Out

The Windows Media Engine can act as a multicast station to deliver multicast streams to client devices. The source of the stream can be multicast, unicast, or a local file. The station can be scheduled, continuous, or play once. The content can be either live or rebroadcast. The Windows Media Engine creates a Windows Media Station file (.nsc) that contains session information including the multicast IP address, port, time-to-live (TTL), and so on. The client requests the .nsc file using HTTP. Once the file is downloaded, the client parses it and sends an Internet Group Management Protocol (IGMP) join to receive the multicast stream. A client can start and stop the stream, but cannot pause, fast-forward, or rewind it. Multicast logging is a configurable option when setting up a multicast station. When multicast logging is enabled, the Windows Media Player automatically collects and sends the statistics to the multicast logging URL using the HTTP POST request method. The statistics collection is accomplished either by a remote server or by the Service Engine itself.

Unicast-Out

The Windows Media Engine can act as a broadcast publishing point to deliver live streams, prefetched/cached content, or content from dynamic ingest, to a requesting client. The source of the stream can be multicast, unicast, or a local file. The Windows Media Engine can also perform live stream splitting if more than one client requests the same content. The broadcast alias can be used to simulate an experience similar to viewing a TV program even if the source of the stream is a Video On Demand (VOD) file. A client can start and stop the stream but cannot pause, fast-forward, or rewind it. When a broadcast alias is configured, a client makes a request to the Windows Media Engine, which is acting as the Windows Media Server, and the Windows Media Engine checks to see whether the incoming stream is present. If it is, the Windows Media Engine joins the stream and splits it to the new client. If the request is the first client request for this stream, the Windows Media Engine sends the request to the origin server and then serves it to the new client.

Authentication

The Windows Media Engine supports pass-through authentication. The following authentication mechanisms are supported in pass-through mode:

- Anonymous
- NTLM
- Negotiate (Kerberos)
- Digest access authentication

With pass-through authentication, the Windows Media Engine establishes a tunnel between the client and the origin server so that the origin server can authenticate the client.

Bandwidth Management

Bandwidth management of Windows Media content can be controlled by setting limits for incoming and outgoing bandwidth and session bit rate and Fast Start maximum bandwidth. In addition, in the case of live streaming, contributing origin servers can by identified to allow incoming content to exceed the bandwidth check to support high demand scenarios. The Windows Media bandwidth management capabilities are described in Table 1-3.

Table 1-3 Bandwidth Management Capabilities

Bandwidth Management	Description
Incoming Bandwidth	The bandwidth for Windows Media content coming into the Service Engine, from either an upstream Service Engine or from the origin server.
Outgoing Bandwidth	The bandwidth for streaming Windows Media content to the end user from the Service Engine.
Incoming Session Bit Rate	The maximum bit rate per session that can be delivered to the Service Engine from the origin server or upstream Service Engine.
Outgoing Session Bit Rate	The maximum bit rate per session that can be delivered to a client.
Incoming Bandwidth Bypass List	The list of identified hosts allowed to bypass the incoming bandwidth check for broadcast or multicast live content.
Fast Start Maximum Bandwidth	Maximum bandwidth allowed per player when Fast Start is used to serve packets to each player. Increased bandwidth initially used by the Fast Start feature can overburden a network if many players connect to the stream at the same time. To reduce the risk of network congestion caused by the Fast Start feature, limit the amount of bandwidth the Fast Start feature uses to stream to each player.

Policy Server Integration

The Windows Media Engine uses HTTP and RTSP to send start, stop, and pause messages to the policy server.

Movie Streamer Engine

Note In Release 2.0, Movie Streamer Engine was in a demonstration state for live streaming, prefetched, cached, and dynamically cached content.
In Releases 2.1 on, live streaming is in full production. Prefetched, cached, and dynamically cached content remain in a demonstration state.
For details of live streaming performance for Movie Streamer Engine, please refer to the Release 2.1 performance bulletin.

The Movie Streamer Engine is an open-source, standards-based, streaming server that delivers hinted MPEG-4, hinted 3GP, and hinted MOV files to clients over the Internet and mobile networks using the industry-standard RTP and RTSP. Hinted files contain hint tracks, which store packetization information that tell the streaming server how to package content for streaming.

The Movie Streamer Engine is an RTSP streaming engine that supports Third Generation Partnership Project (3GPP) streaming files (.3gp). Support of 3GPP provides for the rich multimedia content over broadband mobile networks to multimedia-enabled cellular phones.

Note The streaming capability of Movie Streamer Engine only depends on the movie file format or stream transport type. It is independent of codec types. Movie Streamer supports any client player that can fetch media streams by way of RTSP or RTP. However, the client player must have the correct codec in order to render the stream correctly.

The Movie Streamer Engine can act as both a server and a proxy. It streams prefetched or RTSP-cached content to RTSP clients, acts as a proxy for client requests, splits a live stream into multiple live streams, and caches content requested from remote servers.

After the RTSP request comes into the Movie Streamer, the URI in the RTSP request is modified to reflect the result of the mobile capability exchange. The Movie Streamer checks with the storage function on the Service Engine to see whether the content is stored locally. If the content is not found or if an RTSP-cached content version needs freshness validation, the Movie Streamer engages the Movie Streamer proxy.

In the case of an RTSP-cached content version verification, the Movie Streamer proxy forwards the DESCRIBE request to the origin server for a response containing the Last-Modified-Time header in the response. If the Last-Modified-Time matches the cached version, the Movie Streamer streams the cached content; otherwise, the Movie Streamer proxy forwards the request to the origin server for RTSP negotiation. Then, a client session and a server session are created.

- The server session is responsible for connecting to the origin server to fetch the content and cache it locally. The server session generates the media cache file and the linear hint files.
- The client session is responsible for streaming the locally cached file to the client.
- The client and server sessions are separated so that multiple server sessions can be spawned for the same URL to cache content from different starting points or at faster speeds, or both. This increases the speed of fetching the content. The client session starts to stream from the cached content that the server session is writing.

The Movie Streamer proxy works like the cache fill operation in the Web Engine and the Windows Media Engine. There are two options:

1. Hierarchical Caching Proxy—If content is not found locally, the Movie Streamer checks the upstream Service Engines first before pulling the content from origin server.
2. Static Caching Proxy—The administrator statically configures Service Engines as upstream proxies.

The Movie Streamer supports basic pass-through proxy mode for certain conditions where caching cannot be performed. Such conditions include, but are not limited to, the Service Engine running out of disk space.

Transport Types

Prefetched content can be delivered by the non-accelerated method or the accelerated method. Non-prefetched content (proxied or cached content) is always delivered by the accelerated method. The content is delivered to the client device by means of one of the following mechanisms:

- **Non-Accelerated**—This method has limited concurrent streams and total throughput, but supports many transport formats. The non-accelerated method supports the following transport formats:
  - RTP over UDP
  - Reliable UDP

- **Accelerated**—This method supports only RTP over UDP. Content must be reprocessed by the Movie Streamer Linear Hinter. The linear hinter process can be initiated manually by the administrator or dynamically triggered by the first request for the content.

The Movie Streamer Linear Hinter process may take a while, so the first request that triggers this process is served by means of the non-accelerated method. All subsequent requests are served by the accelerated method.

The first client request for content that requires proxying or caching experiences a delay, because all proxying and caching requires the accelerated method.

Live Stream

The Movie Streamer Engine supports multicast reference URLs (Announce URLs) for programs that are created through the Internet Streaming CDSM. The multicast reference URL, which is in the form of http://*Service Engine IP address*/*Program ID*.sdp, is resolved by the Movie Streamers that are serving the live program.

QuickTime live typically has a UDP socket pair (for RTP and RTCP) per track, and each client session typically has two tracks (audio and video).

Note The following rules apply to live splitting:

1. For unicast streaming, the client request must be sent by means of RTSP.
2. For multicast streaming, the client request must be sent by means of HTTP.

Authentication

The Movie Streamer Engine supports the following authentication modes:

- Basic
- Digest

URL Signing

Support of URL signing for Movie Streamer content requests is a Release 2.2 feature. For more information see the "URL Signing" subsection, under the "Flash Media Streaming Engine"section on page 1-17.

Flash Media Streaming Engine

Note Flash Media Streaming is a Release 2.1 and Release 2.2 feature.

The Flash Media Streaming Engine incorporates the Adobe Flash Media Server technology into the CDS platform. The Flash Media Streaming Engine is capable of hosting Flash Media Server applications, such as VOD (prefetched content, or dynamic or hybrid ingested content) and live streaming, that are developed using ActionScripts.

Upon receiving a client request for VOD content, the edge Service Engine does the following:

- If the content is present, the edge Service Engine streams it using RTMP.
- If the content is not present, the edge Service Engine uses HTTP to fetch the content from the origin server and serves it using RTMP.

No client information is sent to the origin server. No per-client control connection is present between edge to origin server for VOD streaming.

Note The Cisco CDS Flash Media Streaming Engine supports the Adobe Flash Media Rights Management Server (FMRMS) for VOD content; it is not supported for live streaming. Adobe FMRMS protects media content delivered to Adobe Media Player and Adobe AIR applications. FMRMS is also available for proxied content, if Adobe supports the content type. For more information about the Adobe Flash Media Rights Management Server, see www.adobe.com.

Note Release 2.3 supports the Adobe Flash Media Server Administration APIs and the Administration Console that was built using the Administration APIs. These APIs can be used to monitor and manage the Adobe Flash Media Server running on a Cisco CDS Service Engine. See the "Configuring Flash Media Streaming" section on page 4-42 for more information.

HTTP Requests

Flash Media Streaming encompasses all flash applications, from simple Flash Video (FLV) files to more complex Small Web Format (SWF) files. All HTTP client requests for SWF files, that are redirected to a Service Engine by the Service Router, are handled by the Web Engine. The Web Engine, using HTTP, serves the request from locally stored content in the CDS or from any upstream Service Engine or origin server. See the "Web Engine" section on page 1-9 for more information.

RTMP Requests

The SWF file is a compiled application that runs on the Adobe Flash Player, and may contain Real Time Media Protocol (RTMP) calls to FLV, MPEG-4 (H.264), or MP3 files. RTMP calls, in the form of URL requests, are routed to a Service Engine by the Service Router.

Flash Media Streaming supports RTMP and RTMPE on port 1935 only. RTMPE is the secure flash streaming technology from Adobe. Encrypted RTMP (RTMPE) is enabled on Flash Media Streaming by default, and allows you to send streams over an encrypted connection without requiring certificate management.

In Release 2.3, Flash Media Streaming also supports RTMPT and RTMPTE on port 80. RTMP Tunneled (RTMPT) encapsulates the RTMP data within HTTP requests in order to traverse firewalls. RTMP Tunneled Encrypted (RTMPTE) encrypts the communication channel, tunneling over HTTP.

Note The Service Router uses RTMP redirection to direct the client's Flash Player to the best Service Engine based on load balancing and resiliency. RTMP redirections are supported only by Adobe Flash Player 9. All older Flash Players do not support RTMP redirection.

Note For VOD streams, all RTMP calls in the SWF file must be in the following format:

```
rtmp://rfqdn/vod/path/foo.flv
```

In this format, *rfqdn* is the routing domain name of the Service Router, *vod* is the required directory, and *path* is the directory path to the content file that conforms to the standard URL specification.

For prefetched and cached content, the Flash Media Streaming Engine uses RTMP or RTMPE over port 1935. In Release 2.3, the Flash Media Streaming Engine also supports RTMPT and RTMPTE over port 80. For content that is not found locally, the Flash Media Streaming Engine communicates with the Web Engine, that in turn communicates with the upstream Service Engine for cache fill operations. See the

"Cache Fill Operations" section on page 1-9. This interaction uses HTTP. Once the content is in the process of being retrieved by the Web Engine, the Flash Media Streaming Engine uses RTMP to begin streaming the content.

The following describes the characteristics of caching content using HTTP for RTMP client requests;

1. Origin server-based cache validation is still honored for the cached content.
2. Client-side Web Engine rules are bypassed for the RTMP client request.
3. If HTTP headers from the origin server have the "no-cache" attribute set, content is not cached, and transparent proxy is performed to stream RTMP.
4. Transparent proxy from HTTP to RTMP is supported. Flash Media Streaming Engine begins RTMP streaming while content is still being fetched using HTTP proxy mode.

Any HTTP configuration that prevents content from being cached still applies for RTMP requests. The Flash Media Streaming Engine uses multiple HTTP-based range requests in such cases.

Flash Media Streaming Proxy

The Flash Media Streaming Engine can deliver content acting as an origin server or as a proxy server. The Flash Media Streaming Engine acts as a proxy server when content cannot be cached due to the origin server's configuration or due to the Service Engine's Web Engine configuration. Content is ingested and distributed using HTTP, whether the client request for the content used HTTP or RTMP.

Note Any content that does not contain "live" or "vod" in the path is automatically proxied.

Unicast Streaming

The Flash Media Streaming Engine supports unicast flash streaming.

URL Signing

Note Support of URL signing for Flash Media Streaming content requests is a Release 2.2 feature.

Flash Media Streaming supports signed URLs, which adds additional security. The URL signature generation is based on a key that is a shared secret between the component generating the URL signature and the component validating the URL signature. The URL signature can be generated by the Service Engine, another component external to the Service Engine, or the web portal.

For more information about the URL signatures, see the "Configuring URL Signing" section on page 4-23.

Codecs

The Flash Media Streaming codec that is supported in Release 2.1 is On2 VP6. The Flash Media Streaming codecs supported in Release 2.2, in addition to On2 VP6, are listed in Table 1-4.

Table 1-4 Codecs Supported in CDS Releases 2.2 and 2.3 for Flash Media Streaming

Standard	Details
ISO/IEC 14496-3	MPEG-4 Part 3, also known as AAC+, HE-AAC. A set of compression codecs for perpetual coding of audio signals, including some variations of Advanced Audio Coding (AAC), as well as AAC Main, AAC LC, and SBR.
ISO/IEC 14496-10	Advanced Video Coding (AVC), also known as H.264/AVC. All levels of applications are supported, Base (BP), Main (MP), High (HiP), High 10 (Hi10P), and High 4:2:2 Profile (Hi422P). This standard is technically identical to the ITU-T H.264 standard.
ISO/IEC 14496-12	ISO Base Media File Format. A file format for storing media content containing one audio track (either ISO/IEC 14496-3 [AACPlus] or MP3), and one video track (either ISO/IEC 14496-10 [H.264 or AVC] or VP6).
3GPP TS 26.245	Time text format

Live Streaming

Note Live streaming using Flash Media Streaming is a Release 2.2 feature.

Flash Media Streaming uses RTMP to stream live content by means of dynamic proxy. Configuration of live or rebroadcast programs is not required. When the first client requests live streaming content, the stream is created. There are no limits to the number of live streams other than the system load. Live streaming uses distributed content routing to distribute streams across multiple Service Engines.

Upon receiving a client request for live content, the edge Service Engine does the following:

- If the live stream is already present, the edge Service Engine attaches the new client to the existing stream. No message is sent to the origin server and no connection is set up.
- If the live stream is not present, CDS creates a connection to the origin server to get the stream. No client information is sent to the origin server.

No per-client control connection is present between edge to origin server for live streaming.

For Flash Media Streaming, a delivery service can be used for prefetched content, cached content, dynamically cached content, and live content. Because Flash Media Streaming uses dynamic proxy to stream live content, no disk space is used to store content. A Service Engine can act as the origin server origin server for streaming live content, provided the SE designated as the origin server is not assigned to the delivery service that is streaming the live content.

The Flash Media Streaming Engine automatically retries a connection to the origin server if the upstream live-splitting connection fails. This failover does not require any additional retries from the client side. Clients see a subsecond buffering, after which video continues to play. This feature does not address failover when the Service Engine that is streaming to the client fails. The primary advantage is increased resiliency in the CDS infrastructure. In other words, if a Service Engine fails, the downstream Service Engine automatically connects to the origin server.

The Adobe Flash Media Encoder can publish the streams to any Adobe Flash Media Server acting as the origin server. Clients use the RFQDN to get the live content. The request from the client for "streamname" is mapped to origin_appinst_streamname internally in the CDS to differentiate between two streams with the same name in two different delivery services.

Note All RTMP calls for live content in the SWF file must be in the following format:
`rtmp://rfqdn/live/path/foo.flv`
In this format, *rfqdn* is the routing domain name of the Service Router, *live* is the required directory, and *path* is the directory path to the content file that conforms to the standard URL specification.

Flash Media Streaming supports live stream splitting. For more information about live stream splitting, see the "Live Stream Splitting" section on page 1-11.

Interactive Applications

Release 2.3 supports pass-through (proxy) support for interactive applications (non-VOD or non-live). The interactive applications are hosted on a Flash Media Interactive Server that is external to the CDS.

Direct routing from the Service Engine, acting as the Flash Media Streaming edge server proxy, to the origin server (the Flash Media Interactive Server) is supported. Using the delivery service framework, the origin server is abstracted from the client request by using the Service Router Domain Name (SRDN), which resolves to the Service Engine that accepts the user connection and forwards the request to the origin server. In Release 2.3, Flash Media Streaming includes the edge server (proxy) mode, and by default, all non-live and non-VOD applications are proxied by using the edge server. Flash Media Streaming selectively picks connections for processing in edge server mode and aggregates connections to the origin servers.

CDS supports implicit URI as the method that allows the client to connect with the edge server without exposing the origin server. The URI would look like this: `rtmp://edge1.fms.com/ondemand`.

Request routing based on SWF files or using RTMP redirection is supported. However, RTMP redirection requires more changes in the client code. SWF file-based redirection is recommended. SWF redirection works as follows:

1. SWF files and associated HTML pages are either prefetched or hosted in the origin server.
2. Client uses a web browser to access the HTML page, which also loads the SWF file.
3. The SWF file is accessed using the SRDN.
4. The Service Router redirects the request to a Service Engine.
5. The SWF file is downloaded to the web browser.
6. The ActionScript in the SWF file attempts to connect to the same host from where the SWF file was downloaded. This is an RTMP connection that reaches the Service Engine.
7. The Service Engine checks for the application type in the URI, if it is not VOD or live, the processing is moved to the edge server mode and the connection is forwarded to the origin server.
8. The Service Engine tunnels the data between the client and the origin server.

Note Changes to a delivery service do not affect existing connections to the Flash Media Interactive Server (origin server). Only new connections are affected by changes to a delivery service.

Service Router

The Service Router mediates requests from the client devices and redirects the requests to the most appropriate Service Engine. It monitors the load of the devices and does automatic load balancing.

The Service Router is the authoritative Domain Name System (DNS) server for the routed request for the Fully Qualified Domain Name (FQDN) of the origin server. In other words, the Service Router responds to any DNS queries for that domain. The Service Router chooses the Service Engine based on two scenarios:

- Client is directly connected to the service provider's network (on-net).
- Client is roaming outside the home network (off-net).

When clients are connected to the service provider's network, the Service Engine is chosen based on the requested FQDN, the client's IP address, and the responsiveness of the Service Engine. The Service Router compares the source IP address of the client against a table of address ranges assigned to the Service Engines, known as the *Coverage Zone file*. The Coverage Zone file provides information on the proximity of the client to the Service Engine based on each client's IP address.

If the client is not connected to the service provider's network and location-based routing is enabled, the Service Engine that is geographically closest to the client is selected. The Service Router uses a Geo-Location server to find the geographical location of the client. Geographical distance is calculated between the client and the Service Engine. The closest Service Engine is chosen based on the distance.

IP-Based Redirection

Note IP-based redirection is a Release 2.3 feature.

In Release 2.3, there are two ways for client requests to get routed to the Service Router and on to the Service Engine:

- Router Fully Qualified Domain Name (RFQDN) redirection
- IP-based redirection

RFQDN redirection is the default configuration. With RFQDN redirection, client requests are resolved to the Service Router by the DNS server and the Service Router redirects the request to the Service Engine based on route tables created from the Coverage Zone file and the current load of the Service Engines. The redirected URL is http://SENAME.SE.RFQDN/relative_path_of_content, where SENAME is the hostname of the Service Engine.

Note The redirected URL for Flash Media Streaming requests is rtmp://SENAME.SE.RFQDN/application_name/encoded (relative_path_of_streamname), where SENAME is the hostname of the Service Engine.

When IP-based redirection is enabled, the Service Router uses the IP address of the Service Engine in the URL instead of the hostname. The redirected URL is http://<se ip addr>/ipfwd/<rfqdn>/<path>. The IP-based redirection method avoids the extra DNS lookup that was required in the RFQDN redirection.

Service Router Workflow

The Service Router workflow for clients connected to the service provider's network is as follows:

1. The client sends the DNS query for the routed FQDN to the local DNS server.
2. The DNS server replies with the Service Router IP address.
3. The client issues an HTTP or RTSP request to the Service Router.
4. If the Service Router finds the client's subnet in the Coverage Zone file, the following occurs:
   a. The Service Router chooses the appropriate Service Engine and performs a protocol-specific redirection.
   b. The client issues an HTTP or RTSP request to the Service Engine.
   c. The Service Engine serves the content.

   If the Service Router does not find the client's subnet in the Coverage Zone file and location-based routing has been enabled, the following occurs:
   a. The Service Router communicates with a Geo-Location server and gets the geographical coordinates of the client's IP address.
   b. The distance is calculated between the client and the Service Engines, and the Service Engine closest to the client is selected.
   c. The Service Router performs a protocol-specific redirection with the closest Service Engine.
   d. The client issues an HTTP or RTSP request to the Service Engine.
   e. The Service Engine serves the content.

When a Service Router is registered with the Internet Streaming CDSM, the CDSM propagates the Service Router's IP address to all the registered devices. The Service Engine sends a keep-alive message to the Service Router on a periodic interval, which consists of information about the disk health, whether the Internet Streamer application is enabled or not, and the load of the Internet Streamer application on the Service Engine. The Service Router uses the Service Engine's load and liveness information for generating the routes.

The Internet Streamer CDS can have more than one Service Router in order to support Service Router failover. In line with failover, the DNS server should be configured with multiple Service Routers for the same routed FQDN.

Note DNS entries for all FQDNs must be delegated to the Service Router. In the DNS server's database file, a name server record must be entered for each FQDN that routes to the Service Router.

Coverage Zone File

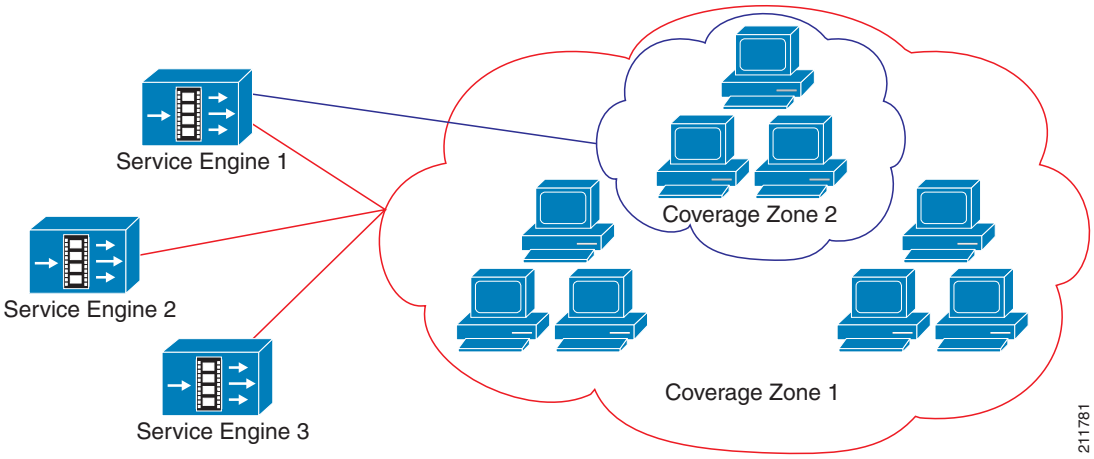
When a Service Engine is registered to the Internet Streaming CDSM, it is assigned a default Coverage Zone file that is created by the CDSM using the interface IP address of the Service Engine. The default Coverage Zone file can be unassigned, and a custom coverage zone can be created using the Coverage Zone file.

A Coverage Zone file is an XML file containing coverage zone entries for each client IP address range, the Service Engine serving that range, the latitude and longitude of the Service Engine, and a metric value. The Coverage Zone file can be referenced by a URL and imported into the Internet Streaming CDSM, or uploaded from a local machine. The Coverage Zone file can be set as the default for a specific Service Router or for all Service Routers in the CDS network.

When content is requested by a client, the Service Router checks the client's IP address to find the coverage zone that contains that IP address. The Service Router then selects the Service Engine that serves this coverage zone. If a specific IP address is in multiple coverage zones, the one with the more specific range is selected. If no match is found in the coverage zone data and if location-based routing is enabled on the Service Router, the Service Router looks up the best Service Engine closest to the client. If the Service Router is unable to redirect the request, the Service Router sends an error response to the client.

A coverage zone can be associated with one or more Service Engines. Each Service Engine can have its own unique coverage zone, or the Service Engines can be associated with more than one coverage zone and have over lapping coverage zones. In Figure 1-2, all Service Engines serve Coverage Zone 1, and Service Engine 1 is specifically associated with Coverage Zone 2, a subset of Coverage Zone 1.

Figure 1-2 Coverage Zone Example



If a coverage zone is served by multiple Service Engines, all Service Engines are put in the routing table. The metric value, entered in the Coverage Zone file, indicates the proximity of the Service Engine to the client. When multiple Service Engines serving a coverage zone are on the same subnet and have the same metric value, and load-based routing is not enabled, the Service Router uses round-robin routing to redirect the client. If load-based routing is enabled, the load of the Service Engines are used to determine the best Service Engine to redirect the client.

Routing Methods

The Service Router chooses the best Service Engine based on whether the Service Engine is participating in the delivery service for which the origin server matches that of the requested domain and whether the Service Engine is assigned to serve the client's network region.

The Service Router uses four methods:

- Simplified hybrid routing (round-robin-based routing) (Releases 2.0 and 2.1)
- Load-based routing (least loaded)
- Last-resort routing (all Service Engines are overloaded)
- Location-based routing (off-net clients)
- Service aware routing
- Content-based routing

Note In Release 2.2, the keepalive messages between the Service Router and Service Engine are transmitted and received on port 2323. However, the Release 2.2 software interoperates with older software releases that do not use port 2323 for keepalive messages. If a firewall is configured between the Service Engine and the Service Router, port 2323 (UDP) must be opened for the keepalive message to go through.

Simplified Hybrid Routing

Note This feature is supported only in Releases 2.0 and 2.1.

The Service Router uses simplified hybrid routing to route requests to Service Engines. In simplified hybrid routing, the Service Engines are chosen in a round-robin fashion if multiple Service Engines are serving the client network region.

Load-Based Routing

Note This is a Release 2.0 and Release 2.1 feature. In Release 2.2 and later, load-based routing is enabled by default and cannot be disabled.

If load-based routing is enabled, the routing decision is made according to the capacity and load of the Service Engines instead of selecting the Service Engines in a round-robin fashion. This routing method is enabled by default.

The load of the Service Engine is determined by different parameters, such as processor use, memory usage, disk usage, the number of current Windows Media streams being served, and so on. The current load is compared with the thresholds configured for the Service Engine. If a threshold has been exceeded for a Service Engine it is excluded from the routing table.

Last-Resort Routing

Last-resort routing is applicable when load-based routing is enabled and all Service Engines have exceeded their thresholds or all Service Engines in the domain are offline. The Service Router can redirect requests to a configurable alternate domain when all Service Engines serving a client network region are overloaded.

Last-resort routing works dynamically when Service Engines are overloaded or deactivated. When the load of one or more Service Engines in the original host domain is reduced below threshold limits or the Service Engines are reactivated, new requests are routed to the original host domain automatically.

If a last-resort domain is not configured and the Service Engine thresholds are exceeded, requests are redirected to the origin server.

Last-resort routing supports requests from RTSP, HTTP (including MMS-over-HTTP), and RTMP clients.

Location-Based Routing

Location-based routing is used for off-net clients. Off-net clients are clients that are not directly connected to the service provider's network. Location-based routing is designed to work with load-based routing. When both are enabled, the Service Router first looks up the client's IP address in the Coverage Zone file. Then, if there is no subnet match, the client's geographical location is compared to the

geographical location of the Service Engines listed in the Coverage Zone file, and the closest and least-loaded Service Engine is selected. Geographically locating a client is used when users roam outside of their home network.

In order to provide routing to off-net clients, the Service Router communicates with a Geo-Location server, which maps IP addresses to a geographic location. For redundancy, the Internet Streaming CDSM can be configured with a primary and secondary Geo-Location server.

The Geo-Location Server identifies the geographical location of an off-net client by the latitude and longitude of the client. The Service Router compares the client's location with the location of the Service Engines participating in that delivery service and chooses the best Service Engine to serve the content.

Service Aware Routing

Note This is a Release 2.2 feature. Service aware routing is always enabled and is not configurable.

In service aware routing, the Service Router redirects the request to the Service Engine that has the required protocol engine enabled, the required protocol engine is functioning properly and has not exceeded its threshold, and the SE has not exceeded its thresholds as configured. See the "Setting Service Monitor Thresholds" section on page 4-77 for more information.

When a request reaches the Service Router, the Service Router generates a hash from the URI. The Service Router first generates a list of Service Engines to best serve the request based on service aware routing. The Service Router then reorders the list based on the hash and selects the best Service Engine. Because the hash generated for the same URI is equal, typically the same Service Engine is selected. If the Service Engine is overloaded, the next Service Engine in the list is selected.

Note For service aware routing, some of the services running on a Service Engine are protocol based. When protocol-based services associated with a protocol engine are stopped on a Service Engine, the Service Router excludes this Service Engine from the list of possible Service Engines that can serve requests for this type of content. The Service Router identifies the protocol engine that will serve the request based on the user-agent in the request. For example, if some Windows Media Engine-related services are stopped, the Service Engine can still serve Web Engine requests. However, if the request for Web Engine content is sent from a Windows Media Player, the Service Router excludes the Service Engine from the list of possible Service Engines that can serve the request.

Note For service aware routing, when a threshold is exceeded for all Service Engines, the Service Engine redirects the client request to the origin server if an alternate domain is not configured. If an alternate domain is configured, the alternate domain takes precedence. For a managed-live URL, if the origin server does not match the source of the live program, the above case fails. For the above case to work, the origin server host must be configured to match the live program source. In addition, the origin server stream name must be the same as the live program name.

Content-Based Routing

Note This is a Release 2.2 feature.

In content-based routing, the Service Router redirects the request based on the Uniform Resource Identifier (URI). Requests for the same URI are redirected to the same Service Engine, provided the Service Engine's thresholds are not exceeded.

The number of redundant copies of content is configurable for content-based routing. The same content can be stored in more than one Service Engine if the number of redundant copies is set to more than one. Redundancy is used to maximize the cache hit ratio. If redundancy is configured with more than one copy, multiple Service Engines are picked for a request with the same URI hash.

Content-based routing is best suited for cache, prefetched, and live program requests in order to maximize the cache hit ratio.

Request Redirection

The Service Router supports the following redirections:

- HTTP ASX Redirection — Used if the requested file has an.asx extension. This redirection method is used for Windows Media Technology.
- HTTP 302 Redirection — Used if the protocol is HTTP and the file extension is not .asx. This is the native HTTP redirection.
- RTSP 302 Redirection — Used if the protocol is RTSP and the client is QuickTime or Windows Media. This is the native RTSP redirection.
- RTMP 302 Redirection — Used if the protocol is RTMP and the client is Adobe Flash Player, Adobe Media Player, or Adobe Flash Lite Player.

Content Delivery System Manager

The Internet Streaming Content Delivery System Manager (CDSM) is a web browser-based user interface. The Internet Streaming CDSM allows the administrator to configure, manage, and monitor delivery services and devices in the Cisco Content Delivery System (CDS). Application programming interfaces (APIs) are provided for backoffice integration with the Internet Streaming CDSM.

Authentication, Authorization, and Accounting

The Internet Streaming CDSM uses HTTPS to secure the administrator's session. Multiple users can perform administrative operations by using the Internet Streaming CDSM. The administrator can configure certain users to have either view-only rights for monitoring the CDS, or full rights that allow configuration changes as well as monitoring capabilities.

User accounts and groups can be added to the Internet Streaming CDSM and given roles and rights for accessing configuration information. It is also possible to segregate and group objects and give access to a limited group of users.

User authentication can be configured to use RADIUS servers when available, otherwise the Internet Streaming CDSM provides its own authentication server.

The CDS-wide policy and status information is maintained in a relational database on the Internet Streaming CDSM. This information is propagated and synchronized with all devices in the CDS network.

As part of the network management process, the administrator can perform basic administration operations on the Internet Streaming CDSM database, including backup and restore.

Device Management

The Internet Streaming CDSM sends device configuration changes to the selected device or group of devices once the change has been submitted. The device sends any configuration changes that were made locally to the CDSM, and also provides periodic status information.

Devices can be organized into user-defined device groups, which allow administrators to apply configuration changes and perform other group operations on multiple devices simultaneously. Because a device can belong to multiple device groups, this reduces the management overhead of the administrator. Device groups allow for a single instance of management thus eliminating the need to repeat the same step for each device.

The Internet Streaming CDSM also provides an automated workflow to apply software upgrades to the devices in a device group.

Higher Storage Utilization of CDS

Storage across multiple Service Engines is virtually divided into buckets where each Service Engine serves only a subset of the total content. Both the local storage and RAM of the Service Engines can function as an aggregated distributed service, providing unlimited scalability. Linear scaling of the CDS storage is accomplished by adding more Service Engines to one location. This addresses the demands of the "Long Tail" use case relevant to the Service Engines. The Long Tail is the realization that the sum of many small markets is worth as much, if not more, than a few large markets. Long-tail distribution is the possibility that extremely infrequent occurrences in traffic are more likely than anticipated.

This higher storage utilization provides the following:

- Overall better system performance
- Higher in-memory cache hit ratio
- Deterministic resiliency in case of failures or overload due to very popular content (This is useful when customers have live, prefetched, and cached assets more than 4.5 terabytes of content on one Service Engine.)

The content distribution is resilient and stateless. If the load of all content mapped to one Service Engine increases, the load is automatically spread to other Service Engines without requiring any administrator intervention.

Delivery Services Management

The Internet Streaming CDSM provides the configuration and monitoring of delivery services, which defines how content is ingested, stored, cached, and published. The Internet Streaming CDSM provides the Service Engines with information about the delivery services and which Service Engines are participating in the delivery service.

In addition to using the Internet Streaming CDSM to define delivery services, an XML file called a *Manifest file* can be used to define a delivery service. The Manifest file and APIs serve as the basis for backoffice integration. For more information about the Manifest file, see the "Manifest File" section on page 2-5.

Resiliency and Redundancy

A CDS that is designed with full redundancy and no single point of failure includes redundant Internet Streaming CDSMs and Service Routers. The redundancy mechanisms for the Content Acquirer and Internet Streamer applications running on the Service Engines operate differently.

Content Acquirer Redundancy

In the event of a primary failure on the Content Acquirer, the failover mechanism supports the election of a backup Content Acquirer. A failover requires that both the primary and backup Content Acquirer be located in the root location of the delivery service.

Live Programs

If the Content Acquirer receives a live program as a multicast stream from the origin server, upon failure of the primary, the backup Content Acquirer assumes control of that program's streaming and the program continues without interruption. This process is transparent to the end user. When the primary Content Acquirer comes back online, it receives the live stream from the active secondary Content Acquirer and does not fall back (regain its primary status) until the live program has finished or has been restarted.

If the Content Acquirer receives the program as a unicast stream from the origin server, the failover mechanism is not supported. If the primary Content Acquirer fails while a program is playing, the person viewing the program must re-request the program.

Internet Streamer Redundancy

If a Service Engine running the Internet Streamer application fails, the Service Router stops receiving keep-alive messages from that Service Engine. When a new request comes in, the Service Router does not redirect the request to that Service Engine; instead, it redirects the request to other Service Engines within the same delivery service. All the existing sessions on the failed Service Engine terminate and the affected end users must re-request the content.

Service Router Redundancy

If the CDS network is designed with multiple Service Routers, all Service Routers are aware of all Service Engines in the CDS. The DNS servers must be configured with multiple Service Routers and the failover is handled by the DNS servers.

Internet Streaming CDSM Redundancy

The Internet Streaming CDSM can operate in two different roles: primary and standby. The primary role is the default. There can only be one primary active in the CDS network; however, you can have any number of Internet Streaming CDSMs operating in standby to provide redundancy and failover capability.

Primary and standby CDSMs must be running the same version of software. We recommend that the standby CDSM be upgraded first, followed by the primary CDSM.

The Internet Streaming CDSM design principle is that the management device is never in the service delivery path. When the CDSM fails, the rest of the CDS continues to operate. A CDSM failure does not affect any services delivered to end users, and all content ingest continues. The only negative effect is

that the administrator cannot change configurations or monitor the CDS. As soon as a failure to connect to the CDSM is noticed, the administrator can activate the standby CDSM. For information on making the standby CDSM the primary CDSM, see the "Changing a Standby to a Primary CDSM" section on page 3-9.