CHAPTER 2

Network Design

Provisioning the Cisco CDS consists of two stages:

- Register the devices to the Internet Streaming CDSM and define the network topology and device groups.
- Configure the delivery services that deliver content to the clients.

This chapter describes the details of the two stages of provisioning a Cisco CDS network and how metadata and content flow through the Cisco CDS. This chapter has the following major topics:

Note In order to achieve the best throughput, we recommend you configure a port channel for the four Gigabit Ethernet ports on the line card. For more information, see the "Configuring Port Channel" section on page E-1.

Cisco CDS Topology

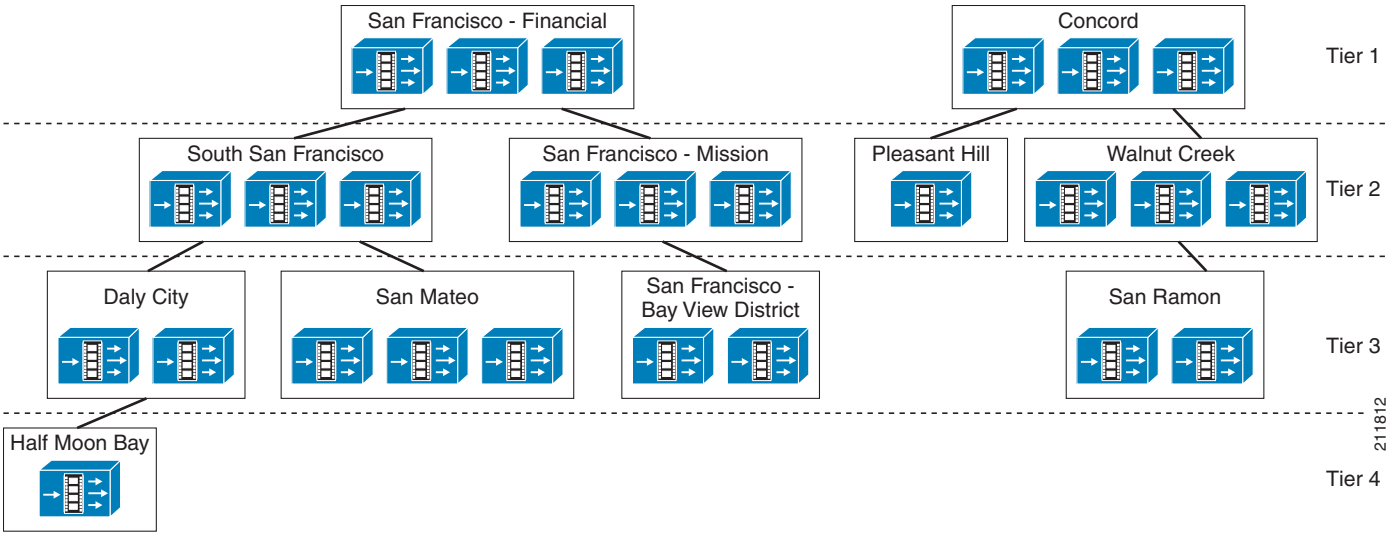
In the Cisco CDS the Service Engines are grouped together into locations, such that a Location Tree is a set of locations organized in the form of a tree. The Location Tree represents the network topology configuration that is based on parent-child relationships. Locations are well connected and have similar connectivity properties to the outside world. A location generally implies topological proximity. Each location can have a parent relationship and multiple child relationships, such that each location can have zero to one parent locations and zero to many child locations. These relationships guide how content flows among locations but does not restrict content flow in any direction.

Locations are also classified into tiers. Each tier consists of locations belonging to the same tier. All locations with no parents belong to Tier 1. All locations that are children of Tier 1 locations belong to Tier 2.

The Cisco CDS can consist of one or more topological Location Trees. A Cisco CDS network is limited by the maximum depth of four tiers.

Figure 2-1 illustrates two location trees, with the parent-child relationship of each location indicated by a solid line and each tier indicated by a dotted line.

Figure 2-1 Location Trees Example



The Location Trees define preferred distribution routes. The Tier 1 locations are located closest to the Internet or backbone. Tier 1 locations can communicate with all other Tier 1 locations.

Note The CDS does not support network address translation (NAT) configuration, where one or more CDEs are behind the NAT device or firewall. The workaround for this, if your CDS network is behind a firewall, is to configure each internal and external IP address pair with the same IP address.

The CDS does support clients that are behind a NAT device or firewall that have shared external IP addresses. In other words, there could be a firewall between the CDS network and the client device. However, the NAT device or firewall must support RTP/RTSP.

Device Groups

Device groups offer a way to group similar devices and configure all the devices in a group at one time. Service Engines can be assigned to multiple device groups when the Device Group Overlap feature is enabled.

A device in a device group can have individual settings different from other devices in the group, and its settings can revert back to the group settings. The last configuration submitted for the device, whether group or individual, is the configuration the device uses.

In addition to group configuration and assignment, the CDSM allows the following:

- Hiding configuration pages of a device group
- Adding all newly activated devices to a device group
- Forcing device group settings onto all devices assigned to a group

A device can be assigned to a device group in one of two ways:

1. From the Device Assignment page
2. From the Device Group Assignment page

Baseline Groups

A baseline group is a special type of device group that denotes a group of devices for a particular service. There are three baseline groups:

- Web Baseline Group—Used for web-based content
- Video Baseline Group—Used for video content
- Platform Baseline Group—Used for platform-specific configurations

A device group can be configured as a baseline group. A device can be assigned to a baseline group in the following three ways:

1. From the Device home page.
2. From the Device Assignment page.
3. From the Device Group Assignment page.

Delivery Service

A delivery service is a configuration that defines how content is acquired, distributed, and stored in advance of a client request (prefetch), and after a client request (cached). Content from a single origin server is mapped to a set of devices by means of a delivery service. Content objects associated with a specific delivery service have a common domain name; in other words, the content in a specified delivery service resides in a single location on an origin server. Each delivery service maps service routing domain names to origin servers one-to-one for Service Router DNS interception.

For each delivery service, there is only one Content Acquirer but multiple Service Engines. The location that has the Content Acquirer for a delivery service is called the *root location*. Other Service Engines in the root location that are assigned to the same delivery service can act as backup Content Acquirers if the configured Content Acquirer fails.

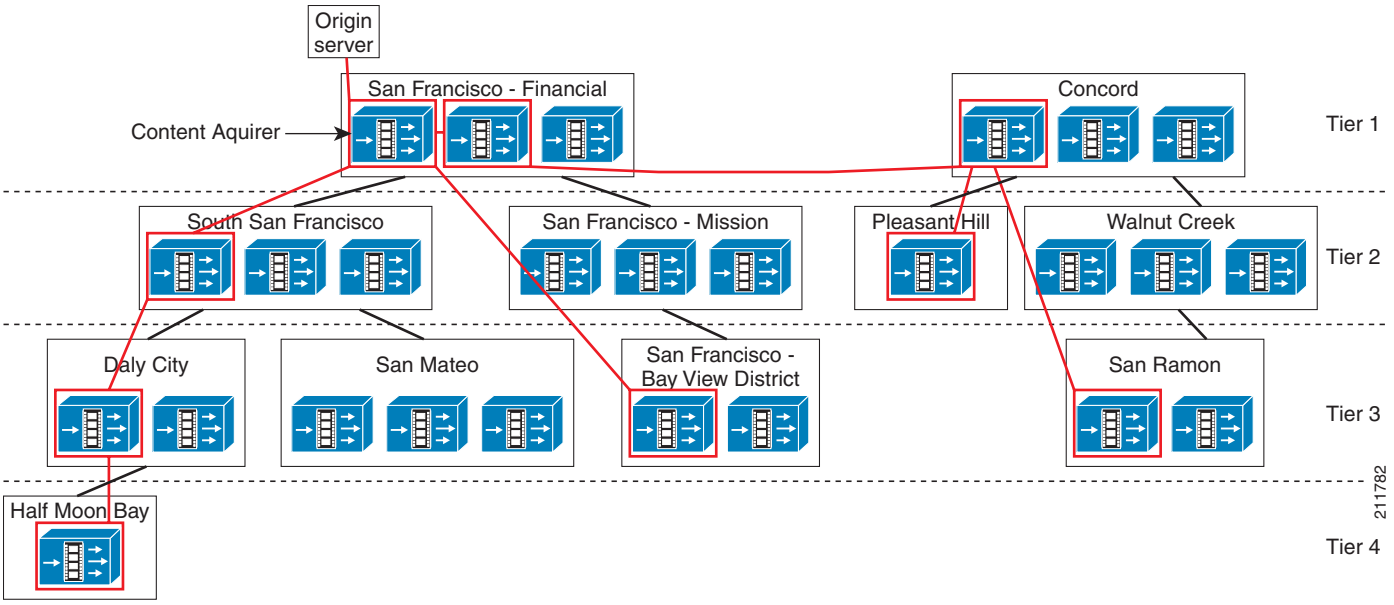
Delivery services form logical routes for content to travel from an origin server through the Content Acquirer to all the Service Engines in the delivery service. Logical routes for content distribution are based on the device location hierarchy or Location Tree.

The content distribution route follows the general tree structure of the Location Tree, where content is distributed from the root of the tree (Content Acquirer) to the branches (Service Engines associated with the delivery service). A delivery service distribution tree is constructed for each delivery service.

By excluding it from the Coverage Zone file, a Service Engine in a delivery service can be configured only to forward content and metadata, and not deliver the content to client devices.

Figure 2-2 shows an example of a delivery service distribution tree. The Service Engines participating in the delivery service are marked in red. Possible content and metadata routes are indicated by red lines. The actual route may differ among the participating Service Engines as determined by the Service Router routing method.

Figure 2-2 Delivery Service Distribution Tree Example



The Cisco CDS supports two types of delivery services:

1. Prefetch/caching delivery services

   For prefetch delivery services, called content delivery services in the CDSM, content is forwarded from Service Engine to Service Engine through the delivery service distribution tree until all Service Engines in the delivery service have received it. The delivery service distribution architecture provides unicast content replication using a hop-by-hop, store-and-forward methodology with the forwarder Service Engines systematically selected on the basis of the manually configured location hierarchy. For caching delivery services, the content need not be fully stored before forwarding.

2. Live delivery service

   The live delivery services are only used for managed live stream splitting. The prefetch/caching delivery services are used for prefetch ingest, dynamic ingest, and hybrid ingest.

There are two methods that can be used to configure a delivery service:

1. Specifying the content by using an externally hosted Manifest file.
2. Specifying the content by using the Internet Streaming CDSM.

   The Internet Streaming CDSM provides a user-friendly interface for adding content and configuring crawl tasks. All entries are validated and a Manifest file is generated. The Internet Streaming CDSM offers the most frequently used parameters, a subset of the Manifest parameters. For a complete set of parameters, use a Manifest file.

The following sections describe the main building blocks of a delivery service:

- Origin Servers
- Manifest File
- Content Acquirer
- Internet Streamer

Origin Servers

Content is stored on origin servers. Each delivery service is configured with one content origin. The same origin server can be used by multiple live delivery services. However, only one prefetch/caching delivery service is allowed per content origin. Each Content Origin is defined in the Internet Streaming CDSM by the following:

- Origin server
- Service routing domain name

The origin server is defined by the domain name that points to the actual origin server. The origin server domain name is used to fetch content that resides outside the delivery service, and to request redirection in case of a failure. The origin server must support at least one of the following protocols in order for the CDS to be able to ingest content:

- HTTP
- HTTPS
- FTP
- CIFS
- SMB

Content can also originate from a local file on the CDS.

The service routing domain name is an FQDN and is used for content redirection. Each content that is ingested by means of the Manifest file is published using the service routing domain name. The service routing domain name configured for the Content Origin must also be configured in the DNS servers, so client requests can be redirected to a Service Router for request mediation and redirection.

Proxy Server

When the Content Acquirer cannot directly access the origin server because the origin server is set up to allow access only by a specified proxy server, a proxy server can be configured. The proxy server is configured through the Internet Streaming CDSM for fetching the Manifest file, and through the Manifest file for fetching the content. Proxy configurations made in the Manifest file take precedence over proxy configurations in the CLI.

Manifest File

The Manifest file contains XML tags, subtags, and attributes used to define how content is ingested and delivered. Each delivery service has one Manifest file. The Manifest file can specify attributes for content playback and control. Attributes for specifying metadata only, without fetching the content, are supported. If special attributes are set, only the metadata and control information are propagated to the Service Engines. The control data is used to control the playback of the content when it gets cached by dynamic ingest. The Manifest file format and details are described in Appendix B, “Creating Manifest Files.”

Crawling

For HTTP, HTTPS, FTP, SMB, or CIFS, a single item can be fetched by specifying a single URL in the CDSM or Manifest file, or content can be fetched by using the crawler feature. The crawler feature methodically and automatically searches acceptable websites and makes a copy of the visited pages for

later processing. The crawler starts with a list of URLs to visit, identifies every web link in the page, and adds every link to the list of URLs to visit. The process ends after one or more of the following conditions are met:

- Links have been followed to a specified depth.
- Maximum number of objects has been acquired.
- Maximum content size has been acquired.

The crawler works as follows:

1. The Content Acquirer requests the starting URL that was configured for the delivery service.
2. The crawler parses the HTML at that URL for links to other files.
3. If links to other files are found, the files are requested.
4. If those files are HTML files, they are also parsed for links to additional files.

In this manner, the Content Acquirer "crawls" through the origin server.

Note The crawler cannot parse JavaScript or VBScript to get the links, nor does it work with HTTP cookies.

A website that has indexing enabled and the default document feature disabled generates HTML that contains a directory listing whenever a directory URL is given. That HTML contains links to the files in that directory. This indexing feature makes it very easy for the crawler to get a full listing of all the content in that directory. The crawler searches the folders rather than parsing the HTML file; therefore, directory indexing must be enabled and the directory cannot contain index.html, default.html, or home.html files.

In FTP acquisition, the crawler crawls the folder hierarchy rather than parsing the HTML file. Content ingest from an SMB server for crawl jobs is similar to FTP ingest; that is, the crawler crawls the folder hierarchy rather than parsing the HTML file.

Content Acquirer

The Content Acquirer parses the Manifest file configured for the delivery service and generates the metadata. If the hybrid ingest attributes are not specified, the Content Acquirer ingests the content after generating the metadata. The Content Acquirer can be shared among many delivery services; in other words, the same Service Engine can perform the Content Acquirer role for another delivery service.

SMB Servers

The CDS supports file acquisition from Windows file servers with shared folders and UNIX servers running the SMB protocol. The Content Acquirer first mounts the share folder. This mount point then acts as the origin server from which the content is fetched. The Content Acquirer fetches the content and stores it locally.

Note With SMB, files greater than two gigabytes cannot be ingested.

HTTP Servers

The no-cache directive in an HTTP server response header tells the client that the content requested is not cacheable. When an HTTP server responds with a no-cache directive, the Content Acquirer behaves as follows:

- If the content to be ingested is specified in an <item> tag in the Manifest file, the Content Acquirer ignores the no-cache directive and fetches the content anyway.
- If the content to be acquired is specified in a <crawler> tag in the Manifest file, the Content Acquirer honors the directive and does not fetch the content.

Internet Streamer

The Internet Streamer application on the Service Engine participates in the delivery service by distributing content within the CDS and delivering content to the clients. The Service Engines can be shared among other delivery services.

HTTP Download—Disabling

Note The ability to disable HTTP downloads on a per-delivery service basis is a feature of Release 2.1.

In some instances, for example when there are contractual obligations to prevent clients from downloading content, it may be necessary to disable HTTP downloads on a delivery service. When HTTP download is disabled, the Web Engine returns a 403 forbidden message.

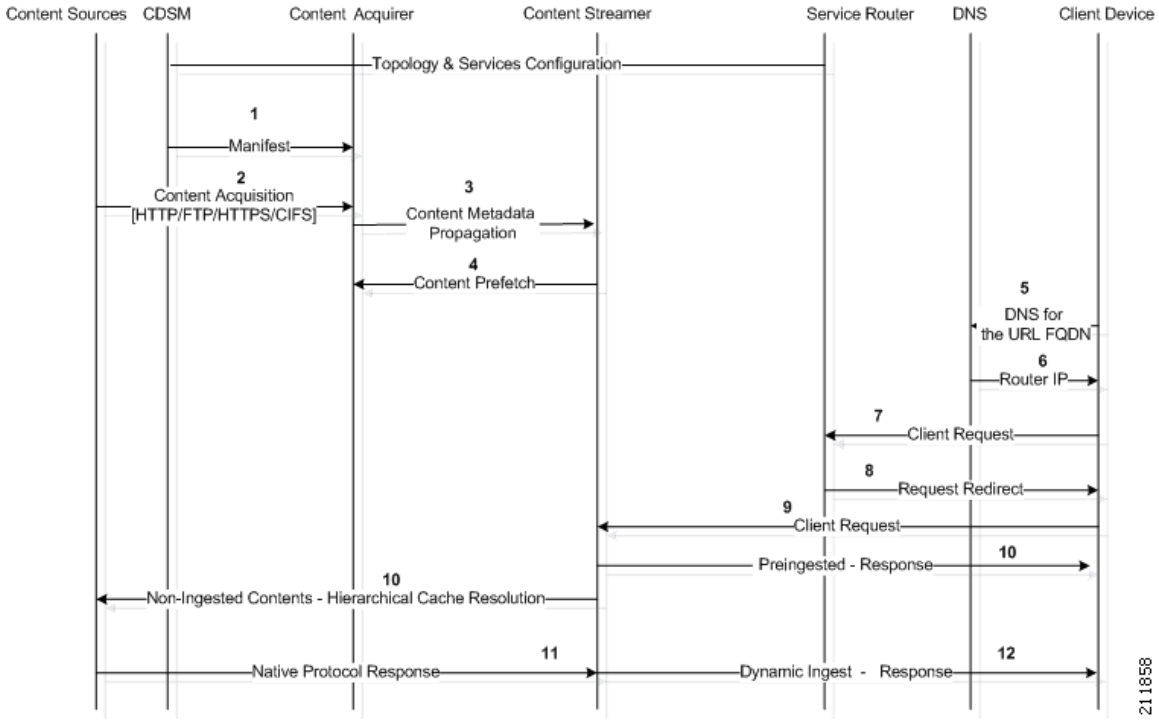
Service Workflow

What follows is a description of the workflow of a delivery service. Table 2-1 shows sample values for for the delivery service workflow described in Figure 2-3. The delivery service workflow is described in detail following Figure 2-3.

Table 2-1 Delivery Service Parameters Example

Parameter	Value
Type	Caching/Prefetch
Origin Server	www.ivs-internal.com
Service Routing Domain Name	cr-ivs.videonet.com
Delivery Service Contents	http://www.ivs-internal.com/video/wmv-152 http://www.ivs-internal.com/video/wmv-92 http://www.ivs-internal.com/video/wmv-212 http://ww.ivs-internal.com/video/wmv-59 type="cache" http://www.ivs.internal.com/video/wmv-6 type="cache"

Figure 2-3 Delivery Service Workflow Diagram



1. The topology is propagated to all the devices registered and activated in the Internet Streaming CDSM. The delivery service configuration is propagated to all the Service Engines subscribed to the delivery service. The Manifest file information is sent to the Content Acquirer for the delivery service.
2. The Content Acquirer parses the Manifest file and generates the metadata. All content listed in the Manifest file, except for non-cache content types, is fetched.
3. The Content Acquirer propagates the metadata to all other Service Engines.
4. The Service Engines receive the metadata and associated prefetched content. The Service Engines do not prefetch content that is “wmt-live” or “cache” types. The “wmt-live” type corresponds to the Windows Media live streaming and the “cache” type corresponds to the hybrid ingest content.
5. The client request for a URL first performs a DNS resolution. The Service Router is configured as the authoritative DNS server for the hosted, or service routing, domain. The URLs that are published to the users have the service routing domain names as the prefix.
6. The Service Router resolves the service routing domain name to its own IP address.
7. The client sends the request to the Service Router and the Service Router uses its routing method to determine the best Service Engine to stream the requested content.
8. The Service Router redirects the client to the best Service Engine.
9. The client sends the request to the Service Engine.

The following are the possible scenarios after the request reaches the Service Engine:

- **Prefetched/Pinned Content**

  Flow 10, “Pre-ingested response.”

  The content is prefetched using the URL: http://www.ivs-internal.com/video/wmv-152

  The actual user request is: http://cr-video.videonet.com/video/wmv-152

The Service Engine processes the user request, and based on the metadata, determines the content was prefetched and pinned in its local storage. The Service Engine looks up the policies for the content and streams the content to the user.

- **Dynamic Ingest/Cached Content**

  Flows 10, 11, 12, "Non-ingested contents—Hierarchical cache resolution," "Native Protocol Response," and "Dynamic ingest response."

  If the request for content is not specified in the Manifest file, dynamic ingest is used.

  The user request is: http://cr-video.videonet.com/video/wmv-cached.wmv

  The Service Engines in the delivery service form a hierarchy, pull the content into the CDS, and cache it. The Service Engine streams the content to the user.

- **Hybrid Ingest/Metadata Only Content**

  (no content flow)

  The request for content is specified in the Manifest file as "cache."

  The user request is: http://cr-video.videonet.com/video/wmv-59

  The Service Engine fetches the content, similar to the dynamic ingest method, but the metadata attributes (for example, serveStartTime, serveStopTime) are honored by the Service Engines and the content is served only if the request falls within the defined time interval.

Programs

A program in the CDS is defined as a scheduled live or rebroadcast event that streams content to client devices. The CDS streams live or rebroadcast content by using the Movie Streamer or the Windows Media Engine.

Movie Streamer live and rebroadcast programs can have multiple tracks (1–3 tracks).

Live Programs

Live events are streamed from third-party encoders (such as Windows Media Encoder Version 9 or the QuickTime encoder) or from streaming servers (such as Windows Media Server). The live stream is ingested by the Content Acquirer and transmitted to all Service Engines using either unicast or multicast. The live stream is transmitted to end users by using either multicast or multicast/unicast live splitting. The live stream is only available to end users during its scheduled times.

With live stream splitting, administrators do not have to create scheduled multicast events, because the Service Engines automatically split the stream.

Unicast to multicast streaming is a solution similar to live stream splitting, except that in the final delivery segment the stream is converted to multicast to minimize the bandwidth demand on the CDS network and to minimize the load on the Service Engines.

Each live program can have up to ten different playtimes scheduled. The program is broadcast from all Service Engines simultaneously.

Rebroadcasts

In a scheduled rebroadcast, prefetched content is scheduled to be streamed from the Service Engines using multicast. Content can only be selected from one delivery service. The Service Engines and device groups assigned to the delivery service are automatically selected when the content files are chosen for the program.

API Program File

Programs can be defined through the Internet Streaming CDSM or through an API. Programs created through APIs are based on a program file. A program file is an XML file that resides on an external server and contains the elements that define the schedule, content, and presentation parameters. The Internet Streaming CDSM gets the program file, parses it, and saves the program file to the database. The program is automatically updated at intervals by refetching the program file and reparsing it. RTSP is the only protocol supported in the program file.

Programs created using an API can be viewed in the Internet Streaming CDSM as read-only, and modifications to the API programs can be accomplished through the API. The API program can also be edited using the Internet Streaming CDSM; however, the information about the API program file is deleted and the program can no longer be modified through the API. A third option is to copy the API program using the Copy Program feature.