Release Notes for Cisco Internet Streamer CDS 3.0.0

These release notes cover Cisco Internet Streamer CDS Release 3.0.0-b156.

Revised: July 2012, OL-27070-01

Contents

The following information is included in these release notes:

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

New Features

Release 3.0.0 of the Cisco Internet Streamer CDS introduces the following features:

IPv6 Support for Client Interfaces

In Release 3.0.0, IPv6 is implemented on the client interfaces for the Web Engine, the Windows Media Streaming Engine, the Flash Media Streaming engine, the Authorization Server, and the Service Router. Movie Streamer does not support IPv6.

Communication among the Internet Streamer CDS devices, between the CDS and the Origin server, and with the CDSM GUI still uses IPv4; these communications includes CMS, Service Router keep-alive, live routing, cache routing, and acquisition and distribution of content.

Because the Internet Streamer CDS supports dual stack (IPv4 and IPv6) for client interfaces, both IPv4 clients, IPv6 clients, and dual-stack clients can interact with the Internet Streamer CDS.

In Release 3.0.0, the following rules apply to configuring IP addresses:

- Manually configured IPv6 addresses are only used to communicate with the clients.
- Unique local IPv6 address and global IPv6 address can be configured for each interface.
- Multiple IPv6 and IPv4 addresses can be assigned to each network interface.

Note If the streaming interface configuration is changed (addition, deletion, modification), the Web Engine is restarted; therefore, we recommend offloading the SE before changing the streaming interface configuration.

Note The following are not supported for IPv6 addresses:

- IP Security (IPSec) implementation
- DHCP configuration
- Flash Media Streaming Administration
- Movie Streamer
- Geo-Location servers
- Proximity-based routing

- Last-resort redirection
- External servers (such as, FTP servers, syslog servers, and NTP servers)

Logical Interfaces

For the Service Engine, the logical interface configured as a primary interface must have an IPv4 address, because the intra-CDS device communication is only through IPv4. If the logical interface is configured as both a primary and a streaming interface, it must have both IPv4 and IPv6 addresses assigned, to serve IPv4 and IPv6, or dual stack clients.

For the Service Router, the primary interface must be configured with IPv4 and IPv6 address to serve IPv4 and IPv6 dual-stack clients.

ICMP6, MLD, and Neighbor Discovery Messages

The following Internet Control Message Protocol version 6 (ICMPv6) messages are supported in Release 3.0.0:

- ICMPv6 error messages
- Destination unreachable message
- Packet too big message
- Time exceeded message
- Parameter problem message
- Echo request message
- Echo reply message

The following Neighbor Discovery for IPv6 are supported in Release 3.0.0:

- Router solicitation
- Router advertisement
- Neighbor solicitation
- Neighbor advertisement
- Redirect message

DNS Configuration

The IPv6 address name server must be configured by using the **ip name-server** *ip-address* command.

Note The Service Router acts as the authoritative DNS server, and supports IPv6 DNS extensions.

If an IPv6 address is configured on the Service Router for DNS, the communication between the Service Router and the DNS server is over the IPv6 transport. The IPv4 address of the Service Router must be configured in the DNS server, so that the Service Router can respond to both A and AAAA queries. In this case, the communication between the DNS Server and the Service Router is over IPv4 transport.

QoS

CDS supports DSCP marking for QoS of outbound IPv4 or IPv6 traffic. The IPv4 header field is the Type of Service (ToS) or differentiated services code point (DCSP) value. The IPv6 header field is the Traffic Class (TCLASS).

ACL Setting

Access control lists (ACLs) for IPv6 are separate from IPv4, and use the **ipv6 access-list** command. An ACL permit or deny policy for IPv6 traffic is based on source and destination IPv6 address, plus other IPv6 protocol factors, such as TCP, UDP, ICMPv6, and GRE, or a specific port number. There are two groups for IPv6 ACLs: Standard ACL and Extended ACL.

Service Router

Communication between the Service Engine and Service Router is through the IPv4 stack, including the keep-alive message. If IPv6 is enabled, then the keep-alive message includes the IPv6 address of the SE in the keep-alive message payload. This enables the Service Router to resolve the SE's IPv6 address correctly.

The Service Router operates as a DNS Server for the requests that belong to the delivery service to which the SR is associated. The Service Router accepts DNS queries from IPv6 clients by way of the IPv6 transport and replies through the IPv6 AAAA record.

The Service Router accepts the HTTP, RTSP, and RTMP requests and sends back the response by way of the IPv6 transport. The Service Router also supports the IP-based redirection, and includes the IPv6 address of the SE in the redirect URL. If the redirect URL has the SE host name, the client sends a DNS query to the Service Router, and the Service Router responds with the SE's IPv4 address for the "A" query and the SE's IPv6 address for the "AAAA" query.

In Release 3.0.0, the Coverage Zone file supports IPv6 and IPv4 addresses. The network and subnetwork addresses in the Coverage Zone file support CIDR format (IP address with a prefix).

Note The Geo-Location servers do no support IPv6 client configuration; therefore, in Release 3.0.0, location-based routing only supports IPv4 addresses.

Authorization Server

The Authorization Server supports the following policies:

- IP address-based
- Geographic location-based
- Service rules-based

The Geo/IP file contains information on the allowed client IP addresses and geographic locations, and denied client IP addresses and geographic locations. The Authorization Server blocks client requests based on the Geo/Ip file uploaded for the delivery service. For IP address-based authorization, Release 3.0 supports both IPv4 and IPv6 addresses in the Geo/Ip file.

For geographic location-based authorization, the SE communicates with the Geo-Location server, which maps IP addresses to geographic locations. The Geo-Location server, which is the same Geo-Location sever used for location-based routing on the Service Router, identifies the geographic location of a client request by the country, state, and city of the client.

Note The Geo-Location servers do no support IPv6 client configuration; therefore, in Release 3.0.0, geographic location authorization of client requests are only supported for IPv4 addresses.

Service Rules

For the Web Engine and Flash Media Streaming, service rules are configured by creating a Service Rule XML file and uploading it for the delivery service. The SrcIp pattern type supports both IPv4 and IPv6 addresses.

For Windows Media Streaming and Movie Streamer, service rules are configured on a per-device basis, either through the CDSM GUI or through the CLI. The **src-ipv6** pattern-list is used to configure IPv6 source patterns for Windows Media Streaming, and src-ip pattern-list is used to configure IPv4 source patterns for Windows Media Streaming and Movie Streamer.

Note Movie Streamer does not support IPv6 addresses.

Windows Media Streaming Multicast

Windows Media Streaming multicast support provides a multicast service to distribute media efficiently to multiple clients using IP multicast. Before a client can tune into a channel and listen to or watch a stream, a multicast station has to be set up first. For IPv6 clients, the Windows Media Streaming multicast station should also multicast on an IPv6 multicast IP address. This requires that an IPv6 multicast IP address be configured for the live or rebroadcast program in the CDSM GUI (**Services > Live Video > Live Programs > Live Streaming** for live programs and **Services > Live Video > Live Programs > Streaming** for rebroadcast programs).

The client fetches an NSC file to get the multicast IP address and port information. For IPv4 and IPv6 clients, Windows Media Streaming must generate two different NSC files. When Windows Media Streaming receives the request, the client type (IPv4 or IPv6) is detected and the corresponding NSC file is sent in the response.

CLI Commands

The following IPv6 commands have been added:

- **ipv6 access-list** {**extended** <*extended_access_list_name*> | **standard** <*standard_access_list_name*>}
- **ipv6 default-gateway** <*X:X:X:X::X*>
- **ipv6 route** <*destination_ipv6_addr*> <*gateway_ipv6_addr*>

The following command keywords and keyword value have been modified for IPv6:

- [**no**] **interface** {<**GigabitEthernet | Portchannel | Standby | TenGigabitEthernet**>} **ipv6 address** {**range** <*low-range-ipv6addr*> <*high-range-ipv6addr*> | <*ipv6addr*>[/<*prefix-length*>]}
- [**no**] **interface** {<**GigabitEthernet | Portchannel | Standby | TenGigabitEthernet**>} **ipv6 access-group** <*access list_no* | *access_list_name*> <**in | out**>
- [**no**] **ip name-server** <*A.B.C.D or X:X:X:X::X*>

The following show and clear commands have been added for Ipv6:

- **show ipv6 access-list** [<*1-99*> | <*100-199*> | *access_list_name* | *output modifiers*]
- **show ipv6 route** [*output modifiers*]
- **clear ipv6 access-list counters** [<*1-99*> | <*100-199*> | a*ccess_list_name*]
- **show interface**, **show running-config**, and **show service-router routes** display IPv6 addresses

The following exec commands have been added for IPv6:

- **traceroute6** <*remote_host or ip_address*> {**traceroute6** [**options**] <*host or ip_address*>}
- **ping6** <*host or ip_address*> {**ping6** [**options**] <*host or ip_address*>
- **dnslookup**, **tcpdump** and **netstat** display IPv6 addresses

Syslog, error logs, and transaction logs all support IPv6 address information.

CDSM

The following CDSM GUI pages support IPv4 and IPv6 addresses:

- **Devices > Devices > General Settings > Network > Network Interfaces**
- **Devices > Devices > General Settings > Network > DSR VIP**
- **Services > Live Video > Live Programs > Live Streaming**
- **Services > Live Video > Live Programs > Streaming**

The **Devices > Devices > General Settings > Network > IPv6 ACL** is a new CDSM GUI page for IPv6 ACLs.

The **Devices > Devices > General Settings > Network > Static IPv6 Routes** is a new CDSM GUI page for IPv6 static routes.

Note **Devices > Devices > General Settings > Network > IP Routes** has been changed to **Devices > Devices > General Settings > Network > Static IPRoutes**.

Validation for the following XML files includes IPv6 addresses:

- Service Rule
- Coverage Zone
- Manifest

Delivery Services

To support Origin server redirection for IPv6 clients and dual-stack clients, do not use the IP address of the Origin server when configuring the Content Origin for a delivery service; instead, use the domain name associated with the Origin server.

If the dual-stack client intent is to use either (IPv4 or IPv6) transports, map both the IPv4 address and IPv6 address of the Service Engine to the Delivery Service (**Services > Service Definition > Assign IP Address**).

While using dual-stack clients (IPv4 and IPv6) and the sessions are in progress, do not remove or add '"ip name-server"' configurations on the SE.

HTTPS Streaming

The HTTPS Streaming feature provides delivery service-based HTTPS support for incoming requests to the SE and outgoing requests to the Origin server. The CDSM GUI offers the ability to enable HTTPS or HTTP for streaming to clients as well as ingesting from the Origin server for each delivery service. When the HTTPS feature is enabled, the inter-SE communication continues to use HTTP.

Note HTTPS support for user equipment (UE) sessions is not supported by the Service Router in Release 3.0.0, so the Service Router cannot be used to load-balance HTTPS sessions. DNS-based redirection must be used to redirect client requests.

DNS-based redirection means that service-aware routing and content-based routing cannot be used. For more information about DNS-based redirection, see the "Product Overview" chapter in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

The HTTPS feature supports SSL 3.0 and TLS 1.0 protocols to tunnel HTTP.

Certificates

SEs associated with the delivery service that has HTTPS streaming enabled have a certificate installed. The SEs do not validate certificates of downstream clients. SEs, as a client of the Origin server, do not send their certificates to the Origin server.

Certificate Authority's (CA's) root certificates are expected to be available to all clients initiating HTTPS communication; most browsers are installed with well-known CA root certificates. Trusted CA certificates are expected to be provided for the purpose of Origin server certification validation.

CDS does not support certificate enrollment (SCEP) nor certificate status verification (OCSP). The Internet service provider or third-party service provides enrolled certificates and installs them through the CDSM.

Note A single subject alternative name (SAN) certificate will be installed for all delivery services in the CDS.

RSA Key Pair

An RSA key pair (public, private) is generated and used for certificate signing requests (CSRs). The private key cannot be encrypted.

Certificate and Key Pair Uploads to CDSM

The CDSM GUI provides new pages under the **System > Configuration > HTTPS Streaming** menu, for uploading certificate files and key files.

Updating Certificates

When a new HTTPS delivery service is added or the Service Router domain name is changed on an existing HTTPS delivery service, the client certificate and key file must be updated. This requires that new certificate and key files are uploaded to the CDSM and a schedule is created to notify the Web Engines associated with the affected HTTPS delivery service.

Traffic Separation for HTTPS

Prior to Release 3.0.0, port 443 was used for Acquisition and Distribution (A&D), all intra-CDS control, and management. With the introduction of the HTTPS feature, port 443 also needs to be used for streaming. It therefore becomes mandatory to have separate interfaces, one for the primary and one for streaming. The management interface, if configured separately, must not share the same interface with the streaming interface either.

Note The HTTPS feature supports the Multiple Logical IP Addresses feature for multiple IP addresses for the streaming interface and one IP address for the primary interface. However, combining the streaming interface and the primary interface on one physical interface is not supported in the HTTPS feature.

Primary Interface

The primary interface is mandatory on all CDS devices and consists of one or more physical interfaces. The primary interface on the CDS devices (SE, SR, and CDSM) is used for the following communication over port 443:

- Communication among SEs
- All intra-CDS control and data traffic
- All prefetched traffic from the Origin server to the SEs by way of the location tree
- All dynamic ingest and all cache miss traffic to the SEs by way of the location tree
- Finding routes to an Origin server for the cache router module and live stream module
- Keep-alive information from the SEs to the SR
- All management communication between the SEs and the CDSM, by default

  Alternatively, a management IP address and port can be configured manually on the SEs and SRs that is used for all management communication to the CDSM. For redundancy, a port channel can be configured. Streaming traffic uses the primary interface, and management traffic between the SE or SR and CDSM use the manually configured IP address and port, and if configured, the port channel and static route created for it.

Note In Release 3.0, the Management Communication Port on the Device Activation page for an SE is hard-coded to port 443. The SR is not affected because the HTTPS feature does not support the SR. DNS-based redirection is the only routing redirection supported.

On more higher-end CDEs, the primary interface can be configured as one-gigabit Ethernet interfaces bonded as a port channel.

Streaming Interface

After a primary interface has been configured and the device is online, the SE is ready to serve streaming traffic. By default, the traffic is served by the primary interface. Optionally, one or more streaming interfaces can be configured on an SE, which designate that all client-facing traffic goes through the streaming interface. Effective client throughput can be measured as a sum of traffic on all streaming interfaces.

The SE streaming interfaces have the following properties:

- If the HTTPS feature is not enabled, the streaming interface is optional, and can have the same IP address as the primary interface
- Number of physical interfaces configured as streaming interfaces is not limited
- They can be configured as a port channel
- Multiple IP addresses in the same subnet can be configured for a streaming interface
- Same IP address can be used for both a primary interface and a streaming interface.

The CDSM, SR, and other SEs do not know the IP addresses of the streaming interfaces on the SE; they only know the primary interface IP address. When the SE sends the SR keep-alive messages, it sends the streaming interface IP addresses as well, which the SR uses to redirect requests to.

HTTPS Enabled

To enable the HTTPS feature for a delivery service, the following configuration must exist:

- All SEs in the delivery service must be running Release 3.0 or later
- All SEs must have at least one streaming interface configured
- Streaming interface must be a different IP address than the primary interface (or management interface if configured)

Note Before making configuration changes to the primary interface or management IP address on an SE, make sure the CDSM is not performing updates to the SE, and there are no prefetching activities going on for the SE.

Note When the HTTPS feature is enabled, and configuration changes (addition, deletion, modification) are made to the streaming interfaces, the Web Engine is restarted; therefore, we recommend offloading the SE before changing the streaming interface configuration.

Web Engine

By default, the Web Engine uses port 80 of the primary interface on the SE for serving HTTP clients and communicating with the Origin server.

If a streaming interface is configured on the SE, the Web Engine uses port 80 of the streaming interface for serving HTTP clients and communicating with the Origin server.

If HTTPS is enabled on an SE, the Web Engine uses port 443 of the streaming interface for serving HTTPS clients.

Internally, the Web Engine on the SEs continue to use HTTP to communicate with each other.

Note If the HTTPS feature is disabled for a delivery service, the Web Engine on the SEs associated with that delivery service continue to use port 443.

Configuring HTTPS

Configuring the HTTPS feature consists of the following procedures:

- Uploading the Certificate and Key Files
- Separating the HTTPS Traffic
- Enabling HTTPS for a Delivery Service

For more information, see the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

Uploading the Certificate and Key Files

Uploading certificate and key files consists of the following pages:

- **Root CA File Registration**—Upload or import the certificates for the Origin servers participating in HTTPS
- **HTTPS Certification Files Registration**—Upload client certificate and key file for all SEs

- **HTTPS Certification File Scheduling**—Schedule client certificate and key file notification to the Web Engine on each SE that is participating in an HTTPS delivery service

The procedures involved in uploading certificate and key files consist of the following:

- Uploading or Importing a Root CA File
- Uploading Client Certificate and Key Files
- Scheduling Web Engine Notification of Client Certificate and Key Files

Separating the HTTPS Traffic

To enable the HTTPS feature for a delivery service, all participating SEs must have at least one streaming interface configured and the streaming interface must be a different IP address than the primary interface (or management interface if configured).

Note Before making configuration changes to the primary interface or management IP address on an SE, make sure the CDSM is not performing updates to the SE, and there are no prefetching activities going on for the SE.

When using the CLI to make configuration changes to the SE, it takes up to one datafeed poll, which has a default of five minutes, for the CLI change to synchronize with the CDSM. Do not make any changes to the HTTPS setting, or SE assignment or device group assignment to the delivery service until the CDSM has been synchronized with the configuration change.

Note When the HTTPS feature is enabled, and configuration changes (addition, deletion, modification) are made to the streaming interfaces, the Web Engine is restarted automatically; therefore, we recommend offloading the SE before changing the streaming interface configuration.

To add a streaming interface to an SE, use the streaming-interface command. For more information, see the *Cisco Internet Streamer CDS 3.0 Command Reference*. For more information about separating traffic, see the "Traffic Separation for HTTPS" section on page 7.

Enabling HTTPS for a Delivery Service

To enable the HTTPS feature for a delivery service, all participating SEs must be running Release 3.0 or later, and must be configured with at least one streaming interface that has a different IP address than the primary interface (or management notifies if configured). For more information, see Separating the HTTPS Traffic.

API Support for HTTPS

API support is provided for the HTTPS feature through the following APIs:

- ChannelApiServlet—Add the OsProtocol and StreamingProtocol parameters
- FileMgmtApiServlet—Add fileType setting of 26 for root certificate files
- CertKeyFileMgmtApiServlet—New Certificate Key File Management API

Note All parameters, except actions, are case sensitive.

If the action parameter is missing or cannot be recognized, an error code and the API usage syntax is returned.

Delivery Service Provisioning API Parameters for the HTTPS Feature

The Delivery Service Provisioning API uses the ChannelApiServlet. Release 2.6.3 introduced the following new actions for ChannelApiServlet:

- createDeliveryServiceGenSetting
- modifyDeliveryServiceGenSettings

The OsProtocol and StreamingProtocol parameters were added in Release 3.0.0 to support the HTTPS feature.

Table 1 lists the required parameters for the createDeliveryServiceGenSetting action.

Table 1 createDeliveryServiceGenSettings Required Parameters

Name	Description	Required	Value Type
deliveryService	Delivery service ID.	Yes	String (Channel_231)
Bitrate	Maximum bitrate limit per session for HTTP.	Yes	Integer, range is 0-2000000
OsProtocol	What protocols are selected for upstream (origin server to SEs)	Yes	0 means HTTP only support 1 means HTTPS only support
StreamProtocol	What protocols are selected for downstream (SEs to end-users)	Yes	0 means HTTP only support 1 means HTTPS only support
HashLevel	URL Hash Level for Cache Routing.	Yes	Integer, range is 0-10
TmpfsSize	Memory Cache Size.	Yes	Integer, range is 1-10
OsHttpPort	Origin Server HTTP Port (web-engine only, default 80).	Yes	Integer, range is 1-65535 except well known port numbers
ReadTimeout	HTTP Response Read Timeout.	Yes	Integer, range is 1-60

For more information, see the *Cisco Internet Streamer CDS 3.0 API Guide*.

Root Certificate File Management API

The File Management API uses the FileMgmtApiServlet. The FileMgmtApiServlet actions are used to manage external XML files registered with the CDSM. The following servlet actions can be used to manage the root certificate file:

- listTypes
- registerFile
- refetchFile
- modifyFile
- deleteFile
- listFile

In Release 3.0.0, the fileType setting in the registerFile, refetchFile, modifyFile, deleteFIle, and listFile actions for root certificate files is 26.

For information about the FileMgmtApiServlet actions, see the *Cisco Internet Streamer 3.0 API Guide*.

Certificate and Key File Management API

The Certificate and Key File Management API is introduced in Release 3.0.0, and uses the CertKeyFileMgmtApiServlet. This API is used to upload the certificate and key files to the CDSM, where they are distributed to all SEs. Uploading new certificate and key files overwrites the existing files.

Note The CertKeyFileMgmtApiServlet only uploads the certificate and key files to the CDSM and distributes them to all SEs. The Web Engine on each SE participating in an HTTPS delivery service needs to be notified of the new certificate and key files. To schedule Web Engine notification of new certificate and key files, use the CDSM GUI. See the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide* for more information.

The servlet performs one or more of the following actions:

- registerFile—Uploads the certificate and key files
- modifyFile—Uploads and overwrites the certificate and key files
- deleteFile—Deletes the certificate and key files from the CDS. Only use this action if you want to disable the HTTPS feature on all delivery services.
- listFile—Provides information about uploaded certificate file and the key file

Session-Based Encryption and Session Tracking

Release 3.0.0 offers the Session-Based Encryption and Session Tracking feature for Apple HTTP Live Streaming (HLS) and Microsoft HTTP Smooth Streaming (HSS) Adaptive Bit Rate (ABR) protocols. Current content workflows may require content have digital rights management (DRM) and encryption applied prior to being sent to authoritative storage or CDN delivery, and often multiple formats and DRM schemes must be supported by the service provider, which results in high storage capacity requirements both within the authoritative store as well as the CDS cache hierarchy. Session-Based Encryption at the edge can dramatically reduce the overall storage requirements.

The Session-Based Encryption and Session tracking (SBE) feature introduces the *Session Construct* term to create the concept of session within the CDS, even though HTTP ABR is inherently sessionless. The uses of the Session Construct are threefold:

- Session Tracking— Useful for reporting, billing, and troubleshooting purposes. The Session Construct allows the CDS to mark transaction log entries for the many individual ABR fragment requests with a common parameter for correlation to the broader user session between the user and the service provider CDN.
- Access Protection—Through the use of optionally signed cookies that have expiration times or URL query strings, along with the Session Construct, only the intended clients (IP address validation) may successfully request content from the CDS.
- Session-Based Encryption—Ensures that the CDS can encrypt with the correct key for the specific client session. The Session Construct is fundamental to the implementation of session-based encryption.

Note SBE is supported per user device and per content request. So, if a user has two HLS content requests from the same device for the same content, there are two separate sessions that are tracked and each has its own key.

Session-based encryption supports encryption key message (creation, request, rotation, and deletion) communication over HTTP and HTTPS with an external key management server (KMS).

The ABR Session Tracking and Session-Based Encryption feature for HLS and HSS protocols addresses the following requirements:

- Support ABR session tracking across multiple TCP sessions and content access protection using HTTP cookies.
- Session tracking across bitrate shifts.
- Support SBE for HLS as per the Apple HTTP Live Streaming specification.
- Function irrespective of the underlying storage mechanism (for example, NAS, prefetched content, and cached content).
- Support out-of-band manifest use-case for HLS and HSS, where the manifest handling is done outside the SE and the SE receives only the video segment or fragment requests
- Support HSS with out-of-band manifests, and with whole fragment encryption (non-PIFF)
- Supports HSS configurable encryption key messages (create, request, rotation, delete) to different KMSs
- Supports periodic key rotation for HLS
- Support media session resiliency across SE failover

The following subsections provide more information on this feature:

- HLS Session Based Encryption
- HSS Session-Based Encryption
- Session Tracking
- Configuring Session-Based Encryption and Session Tracking
- Transaction Logs for Session-Based Encryption and Session Tracking

HLS Session Based Encryption

Session-based-encryption encrypts HLS video content with a unique key per device and content, The key consists of AES-128 CBC cipher for encryption and decryption. The EXT-X-KEY tag in the m3u8 playlist file is used by clients to retrieve the key and decrypt the content.

HLS Solution Components

A typical media distribution and delivery systems has the following components:

- Encapsulator—Responsible for packaging the content into MPEG2-TS segments and creating the m3u8 playlist files.
- Authorization/Authentication Server—Responsible for user authentication and media entitlement. Typically this consists of a URL signature with identifiers (user, device, media, session) or an authentication token that may need to be validated by the SE, and the URL for the manifest (m3u8 playlist) file.
- Key Management Server (KMS)—Creates and provides keys used for encryption or decryption to the streaming SE.
- SE—Provides edge caching and streams content to the client. When session-based-encryption is enabled, the content is encrypted here. The edge SE interacts with the KMS to retrieve the encryption keys, and updates the manifest file with the key URI that is provided to the client.

- Client—Downloads the playlist and segments, retrieves the key URI, and decrypts the segments for playing the content.

HLS Out of Band Manifests

In systems where there is a separate manifest generator or session ID generator, the SE receives only the video segment requests. In such cases, the session is created on the first video segment request and session encryption is performed identical to the case where the SE received the manifest request. Session tracking is performed based on the URL query string, which typically has a session ID generated by the manifest generator. We do not recommend using a cookie-based session tracking with out-of-band manifest.

HSS Session-Based Encryption

Session-based encryption for HSS supports the following:

- Out-of-band manifests (only)
- Whole fragment encryption
- Caching only the small fragments for both VOD and live streaming
- AES-128 CBC cipher for encryption and decryption
- One key per session (no key rotation)

With out-of-band manifest for HSS, the IIS Smooth Streaming Client Manifest (ISMC) file (also known as the client manifest file) and the key URI are sent to the Windows Silverlight client by the manifest generator. The SE only receives the fragment requests from the client; not the client manifest file. After the SE receives the client manifest file, it gets the key from the KMS, encrypts the video fragments, and sends them to the client.

The same statistics counters used for HLS are also used for HSS.

Note Protected Interoperable File Format (PIFF) encryption is not supported in Release 3.0.0.

Session Tracking

For ABR session tracking, both external and internal session IDs are supported. External session IDs are provided by the entitlement server, DRM system, or manifest generator, and are delivered to the CDS in the client request. In other scenarios, the session ID is generated by a session state manager and delivered to the CDS as part of the publish-URL. The session ID is stored in the session cookie. When the client downloads the ABR manifest file at the start of content delivery, the SE sends a cookie along with the manifest file response to the client. The cookie is sent back to the SE in all subsequent manifest and video fragment requests. If an external session ID is not received, an internal session ID is created by the CDS for internal purposes.

Session Cookie

The session cookie includes session information (session ID, key parameters, expiry, and MD5 hash of cookie), and expires, path, and domain control of the cookie. When the SessionSinfoTimeout is configured in the Service Rule XML file for the delivery service, an expiry time stamp is added to the cookie. If the SessionSinfoTimeout is reached, then the client requests are considered invalid. For this mechanism to be effective for content access protection, the SessionSinfoTimeout is set for a few

minutes and refreshed periodically. The cookie expiry is refreshed when a manifest or fragment request comes in after 50 percent of the SessionSinfoTimeout has elapsed but has not expired. Following is an example of a session cookie:

```
=sid=403-I-617F4464804374693FB7C91665DC8E003636~et=1330546573~kp=1_20~md5=2A36FB095849DA68
8A8D8D07A84B3333;
```

Client IP Address Validation

When the SessionClientIpCheckEnable parameter in the Service Rule XML file is set to enable, the SE validates the client IP address for each fragment request in the following cases:

- Client always resides behind a managed client, and the client IP address known to the SE does not change during the playout session.
- For URL query string, a unique manifest file is generated for each client and the client IP address is sent in each fragment request URL. For session cookie, there is no need for a unique manifest, unless the encryption requires it.

Key Parameters

- Key Content-ID—Four-byte number that identifies a user session to the KMS. Content ID is not related to session tracking, but is used as an alternative to the session ID when interfacing with the KMS. Some KMSs only accept a 4-byte session ID, while others use a full session iD (external or internal).
- Key Rotation—Variables stored in the session cookie are used to calculate the key rotation, such that even if an SE failover or configuration change occurs during a session, the correct key is retrieved from the KMS for encryption.

Configuring Session-Based Encryption and Session Tracking

The Service Rule file has been enhanced in Release 3.0.0 to provide configuration settings for the Session-Based Encryption and Session Tracking feature. The Service Rule file is an XML configuration file that specifies settings for all the SEs in a delivery service.

In addition to the Service Rule XML configuration file, the following must be enabled for each delivery service (**Services > Service Definition > Delivery Services > Location Settings**):

- Enable HSS Session Tracking—Enables HSS session tracking at the edge location
- Enable HSS Streaming from NAS—Enables HSS from Network-attached Storage (NAS) devices (not supported in Release 3.0.0)
- Enable HLS Session Tracking—Enables HLS session tracking at the edge location

Service Rule Configuration for Session-Based Encryption and Session Tracking

A new action, Rule_SetAction, has been added to the Rule_Actions subelement in the Service Rules schema file (CDSRules.xsd).

For more information about the Service Rules, see the “Creating Service Rules File” appendix in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*. The Rule_SetAction subelement is used to specify the configuration parameters for Session-Based Encryption.

For more information about ABR Session-Based Encryption, see the “Configuring Session-Based Encryption and Session Tracking” appendix in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

Transaction Logs for Session-Based Encryption and Session Tracking

Transaction logs for Session-Based Encryption and Session Tracking consist of the following:

- Application Session Manager Transaction Logs
- Key Client Manager Transaction Logs
- ABR session transaction logs

For more information on ABR session transaction logs, see the "ABR Transaction Logs" section on page 17.

Application Session Manager Transaction Logs

The Application Session Manager (ASM) module maps the content playout session to a particular client. Client and MediaAsset together identifies a unique session object. The ASM module is generic and provides interfaces for applications to integrate. For HLS Session based encryption, the ASM interacts with HLS Helper and Key Client Manager.

The ASM transaction log keeps track of all transactions of the ASM. This is the same log as the Web Engine custom format transaction logs located in the /local1/logs/webengine_clf/ directory. The custom log formats for the ASM transaction log consist of the following:

%a %U %O %b %I %m %>s %t %D %y %S %i %E %k %B

For more information see the "New Custom Formats for Web Engine Transaction Logs" section on page 17.

Key Client Manager Transaction Logs

The Key Client Manager (KCM) transaction logs capture the details of the transactions between the KMS and the SE. When transaction logging is enabled, transaction logs between the SE and KMS are logged in the KeyClientManager log file in the /local1/logs/KeyClientManager/ directory.

Table 2 describes the KCM transaction log fields.

Table 2 KCM Transaction Log Fields

Field	Description
Timestamp	Time stamp of the transaction.
Session_ID	ID of the session.
Key_Server	Domain or IP address of the KMS.
Key-Profile#DsId	Key profile name used in this transaction.
HTTP-Method	HTTP method used in the transaction (POST, GET, or DELETE).
Key-Request-URL	URL of HTTP call sent to the KMS in this transaction.
Response-Status	Status of transaction.
Key-URI	Client key URI that is inserted in manifest file. The end-user client uses this URL to acquire the key from the KMS.

ABR Transaction Logs

The ABR Transaction Log feature consists of two aspects:

1. Adding optional fields that provides information on ABR HTTP transactions to the custom format transaction logs for the Web Engine.
2. Enabling the new log file, the ABR Per Session log, which significantly reduces the log message volume by combining per-transaction logs for each session.

Enabling ABR Per Session Log

To enable ABR Per Session log, the following needs to be configured:

- Each device needs to have transaction logs and ABR Per Session logs enabled
- Each delivery service must have session tracking enabled
- Service Rule XML file must have a SessionResolveRule configured

Enabling ABR Per Session Log on a Device

The transaction logs and ABR Session Log must be enabled on each SE participating in ABR session logging.

For more information, see the "Configuring Devices" chapter in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

Enabling Session Tracking for a Delivery Service

To enable session tracking for a delivery service, the following must be enabled:

- **Enable HSS Session Tracking**—Enables HSS session tracking at the edge location
- **Enable HLS Session Tracking**—Enables HLS session tracking at the edge location

For more information, see the "Configuring Services" chapter in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

Adding a SessionResolveRule to the Service Rule XML File

At least one SessionResolveRule is required to enable the ABR Session log. The Origin server (OFQDN) and Service Routing Domain Name (RFQDN) must be specified as the pattern lists to match for the SessionResolveRule.

For more information, see the "Configuring Session-Based Encryption and Session Tracking" appendix in the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*.

New Custom Formats for Web Engine Transaction Logs

Each ABR streaming session consists of multiple HTTP transactions. Reflected in the transaction logs for Web Engine, there are multiple transaction log entries for one streaming session.

The following custom format tokens have been added to the Web Engine custom format to provide information on ABR HTTP transactions:

- %i—Session ID
- %S—Session status
- %y—ABR protocol (HLS or HSS)

- %E—Encryption type (none, AES256CTR, AES256CBC)
- %k—Method of session tracking (cookie, URL query)
- %B—Bitrate in bits per second (bps) (only for HSS)

Use the **transaction-logs format custom** command to modify the custom log format.

The custom transaction logs are located in the /local1/logs/webengine_clf/ directory.

ABR Per Session Log

Typically, ABR transaction log volume is very large because of the number of fragments for each session. The ABR Per Session log reduces log message volume by consolidating per-transaction log entries for each session. All ABR-related HTTP transactions are logged in the Per HTTP transaction log and session log entries are only generated on session events.

ABR Session Events

HTTP is a stateless protocol. HTTP ABR does not have a counterpart of states as does the traditional streaming protocol such as RTSP. The session close event generates a transaction log entry.

ABR Session Log Fields

The ABR Per Session log is located in the /local1/logs/webengine_abr directory. Table 3 describes the ABR Per Session transaction log fields.

Table 3 ABR Per Session Transaction Log Fields

Field	Description
client-ip	Client IP address.
abr-protocol	Type of ABR protocol.
session-id	Unique string generated by server to identify the session.
manifest-uri	URI of manifest file. If it is a failover session and the CDE does not receive request for master manifest file, this field is the asset URL (URL without asset filename).
asset-id	Not supported in Release 3.0.0. This field always returns a dash (-).
bytes-sent	Bytes sent to client. If this is session close log directly following a session stop, the field is 0 as nothing is sent since last log.
bytes-recvd	Bytes received from client. If the log entry consists of a session stop directly followed by a session close, the bytes-recvd field is 0.
status	Status of the session.
time-recvd	If this is the first log entry for the transaction, it is the time stamp of when this session is received; otherwise, it is the time stamp of when the first HTTP transaction is received since last log message. If this is a session close log entry directly following a session stop log entry, the field is blank (-) because no request was received since last log.
time-to-serve	If this is the first log entry for the transaction, it is the time, in microseconds, taken since this session started; otherwise, it is the time when the first HTTP transaction is received since the last log message. If this is session stop log entry directly followed by a session stop, the field is 0.

Table 3 ABR Per Session Transaction Log Fields (continued)

Field	Description
bitrate	Current stream bitrate in bits per second (bps). A value of zero (0) means "not applicable" or "not available." For HSS, the video bitrate is used as the overall stream bitrate. **Note** The bitrate field is not supported for HLS in Release 3.0.0.
encryption	Indicate the session encryption type.
session-tracking-mode	Method of tracking the session.

Service Router Last-Resort URL Translator

Release 3.0.0 introduces the URL translator for configuring the Service Router Last Resort feature. The URL translator provides a way to dynamically translate the client request URL in order to redirect the client to a different CDN.

Previously, the last-resort routing allowed redirecting a request to an alternate domain or Origin server for one of the following conditions:

- All SEs in the delivery service have exceeded their thresholds
- All SEs in the delivery service are unavailable
- Client is unknown

Last-resort routing could also be configured to redirect a client to an error domain and filename.

With the URL translator option, the following occurs if the SR uses last resort routing for a client request:

1. The SR contacts the third-party URL translator through the web service API. The Web Service API is described in the *Cisco Internet Streamer CDS 3.0 API Guide*.
2. The third-party URL translator sends the translated URL in the response to the SR.
3. The SR sends a 302 redirect message to the client with the translated URL it received from the third-party URL translator.

The timeout for connecting to the URL translator server is 500 milliseconds. There are no retries if the URL translator cannot be reached.

If there is no configuration on the URL translator for the requested domain or the connection timeout threshold has been reached, the SR last-resort routing falls back to the alternate domain configuration.

Client Support

The SR Last-Resort URL Translator feature fully supports RTSP and HTTP client requests.

For Flash Media Streaming clients, the client must be able to handle redirects to a different application name. Most Flash clients cannot support a stream name change; so the filename returned by the translator is ignored.

Configuring the SR Last-Resort URL Translator

The SR Last-Resort URL Translator feature can be configured through the CDSM GUI or through the SR CLI.

One translator can be configured per domain (content Origin server).

If there is no configuration on the URL translator for the requested domain or the connection timeout threshold has been reached, the SR last-resort routing falls back to the alternate domain configuration.

Stream and Cache-Fill Performance

Release 3.0.0 improves CDS performance of streams and cache-fill in the following ways:

- QoS support. Using QoS ensures that sessions receive either best effort or a guaranteed rate while not exceeding overall system capacity.
- File-level hole management is supported, allowing partially-filled files to be streamed, and multiple-parallel fills and streams to be attached to the same file at various offsets.
- Higher performance data transmission engine provides better throughput

Small ABR File, Encrypted HLS and HTTPS Traffic

Hole management is not used for small files, because the sessions are over quickly, and the entire file is always downloaded from the Origin server.

Encrypted HLS traffic and HTTPS traffic do not use the Stream and Cache-Fill components, because with HTTPS traffic is encrypted in the user space, and with encrypted HLS traffic, the situation is similar to small ABR files.

Large ABR File

With ABR large files, files are either stitched from fragments or they are natively large files. This type of traffic is similar to progressive download traffic in the way it uses the Stream and Cache-Fill feature, but the usage pattern is different. Specifically, clients are more likely to stop streaming from a file when they shift bit-rates, so files may have holes.

Hole management and QoS optimize the serving of large ABR files. There is a large improvement in performance with ABR large files and the Stream and Cache-Fill feature.

Large File Progressive Download

Large file progressive download traffic is similar to large ABR files, but the client is likely to stay on a bit rate longer because it does not automatically adjust its rate. This traffic type also sees a large performance improvement, for the same reasons as large ABR files.

Stream and Cache-Fill Feature Components

The Stream and Cache-Fill feature consists of the following components:

- QoS Types
- Hole Management
- Admission and QoS statistics can be viewed by using the show statistics admission command.

QoS Types

The Stream and Cache-Fill feature adds support for the following QoS classes:

- Hard Guaranteed (HG)—Flows assigned a bit rate that is maintained under any circumstances. The bandwidth allocated for these sessions is never reused by other sessions. HG is not directly selectable.

Note HG is not supported in Release 3.0.0.

- Soft Guaranteed (SG)—Flows assigned a fixed bit-rate, unlike HG, any unused bandwidth assigned can be reused by other sessions. SG and best effort (BE) flows can continue to be admitted even if the total requested SG rate exceeds system capacity, as long as the total measured rate does not exceed the total system capacity. This is a *statistical guarantee* in the sense that it is expected to be guaranteed in most circumstances.
- Best Effort (BE)—Depending on whether the traffic is VOD or live, the bandwidth allocation behaves differently.
  - In the VOD case, all BE streams are given an equal share of any disk bandwidth left over after guaranteed sessions are satisfied.
  - In the live case, each BE client is allowed to stream at a rate limited only by CPU and network interface bandwidth.

  The system has an administratively-defined minimum best-effort rate for VOD BE sessions. New sessions are only admitted if the global best-effort rate does not fall below the minimum. This way the SE does not stream countless sessions at very low bit-rates.
- On any given SE, live BE traffic cannot be mixed with any other type of QoS, but VOD BE can be mixed with guaranteed QoS types.

Table 4 summarizes the different QoS types in the CDS. The types listed in bold are introduced with this feature.

Table 4 Supported QoS Types

QoS Type	Minimum Guaranteed Rate	Maximum Rate	Other Compatible QoS Types
Hard Guaranteed (HG) (not supported in Release 3.0.0)	Protocol Engine requested rate	Protocol Engine requested rate	SG, BE
Soft Guaranteed (SG)	Delivery service bitrate	Delivery service bitrate	HG, BE
Best Effort VOD (BE-VOD)	Globally configured minimum	(Total disk rate - Total (SG + HG) rate) / number BE sessions	SG
Best Effort Live (BE-live)	None	Determined by CPU and network cards	None
Fixed Bit Rate	None	Delivery service bitrate	None
Best Effort	None	Determined by CPU, network cards and disk	None

There is no performance penalty for using any QoS type. The QoS types are defined indirectly through the delivery services.

Admission and QoS statistics can be viewed by using the **show statistics admission** command.

Note Only admission statistics can be cleared. QoS statistics are dynamically measured quantities rather than counters; and therefore, cannot be cleared.

Hole Management

Although hole management is not directly visible as a feature to the user, it has a great impact on system behavior. The basic ideas of hole management are as follows:

- Multiple fills can run on a single file at different offsets
- Play request is considered a hit either if the entire request range is filled, or a currently active fill will eventually fill that range
- Maximum number of holes per file is limited for file system robustness reasons

Holes in a file are created in two cases:

1. If the last client aborts the session and fills are still going to the file.
2. When fills are aborted for some other reason like the Origin server drops the connection.

In either case, hole management is equipped to handle these holes by starting fills as needed to bridge holes and limit the total number of holes in a file if required.

Abort Behavior

When a client aborts a session, any associated fill task normally continues to completion, until either the entire hole is filled or until the end of file. The exception to this is when no more clients are playing the file. In this case, all fills are aborted.

Alarms

Alarms are generated when the system is exceeds the threshold. This happens when more than three disks in the system are over the limit. The alarm triggers the SR to stop directing traffic to the SE.

New Hardware Platforms

Release 3.0 supports three additional CDE250 models and two Unified Computing System (UCS) models. All CDEs ship with the Cisco Internet Streamer CDS software. The Cisco UCS models (UCS C200 and UCS C210) and the Cisco Internet Streamer Release 3.0 software are sold separately and ship independently of each other.

The new CDE250 models (CDE250-2S8, CDE250-2S9, and CDE250-2S10) have four interfaces at 10 gigabit Ethernet speeds and four interfaces at gigabit Ethernet speeds (plus two additional gigabit ethernet interfaces for management).

The new CDE250 models only support the SE device mode and have the following storage capacities:

- CDE250-2S8—24 x 300 GB 2.5 SSD
- CDE250-2S9—12 x 600 GB 2.5 SSD
- CDE250-2S10—24 x 600 GB 2.5 SSD

Release 3.0 supports golden configuration of the UCS C200 and the UCS C210.

Enhancements

The following enhancements have been added in Release 3.0.0:

- Telnet Button Removed
- Last-Resort Routing Redirects to Origin Server

Telnet Button Removed

Telenet button missing in the Device>Device home menu.

It was removed for the following two reasons:

- IE7/8 and higher versions of Firefox does not support Telnet.
- Telnet is not secure to login to devices.

Last-Resort Routing Redirects to Origin Server

By default, last-resort routing redirects known client requests (and unknown client requests if **Allow Redirect All Client Request** is enabled) to the Origin server if the alternate domain is unreachable or is not configured, and the translator domain is not reachable or times out and there is no alternate domain configured.

The option to redirect client requests to the Origin server is now configurable on a per Content Origin basis. The default for last-resort routing is to redirect client requests to the Origin server if the alternate domain is not available. To disable redirects to the Origin server, in the **Services > Service Definition > Content Origins > Definition** page, uncheck the **Enable Origin Server Redirect**.

New MIB Objects

Table 5 describes the new object IDs that were added to CISCO-SERVICE-ENGINE-MIB for Session-Based Encryption (SBE).

Table 5 Object IDs for Session-Based Encryption Counters

Group	Object ID	Description
cdsHssGroup	cdsHssAppSessionActive(1)	Number of active media sessions.
	cdsHssAppSessionCreated(2)	Number of sessions created.
	cdsHssAppSessionCreatedInternalSID(3)	Number of sessions created-Internal sessID.
	cdsHssAppSessionFailover(4)	Number of session failovers.
	cdsHssAppSessionDeletedInactive(5)	Number of sessions deleted-inactive.
	cdsHssAppSessionDeletedError(6)	Number of sessions deleted-internal error.
	cdsHssAppSessionDeletedExpired(7)	Number of session deleted-expired request.
	cdsHssAppSessionDeletedSIDError(8)	Number of sessions deleted-session ID error.
	cdsHssAppSessionReqRejectInvalidIP(9)	Number of requests rejected-client IP invalid.
	cdsHssAppSessionReqRejectSIDCollision(10)	Number of requests rejected-SessID collision.
	cdsHssAppSessionReqRejectFailToTrack(11)	Number of requests rejected-failed to track.
cdsHlsGroup	cdsHlsAppSessionActive(1)	Number of active media sessions.
	cdsHlsAppSessionCreated(2)	Number of sessions created.
	cdsHlsAppSessionCreatedInternalSID(3)	Number of sessions created-Internal sessID.
	cdsHlsAppSessionFailover(4)	Number of session failovers.
	cdsHlsAppSessionDeletedInactive(5)	Number of sessions deleted-inactive.
	cdsHlsAppSessionDeletedError(6)	Number of sessions deleted-internal error.
	cdsHlsAppSessionDeletedExpired(7)	Number of session deleted-expired request.
	cdsHlsAppSessionDeletedSIDError(8)	Number of sessions deleted-session ID error.
	cdsHlsAppSessionReqRejectInvalidIP(9)	Number of requests rejected-client IP invalid.
	cdsHlsAppSessionReqRejectSIDCollision(10)	Number of requests rejected-SessID collision.
	cdsHlsAppSessionReqRejectFailToTrack(11)	Number of requests rejected-failed to track.
	cdsHlsAppSessionInlineKeyReqs(12)	Number of inline key requests.
cdsKcmGroup	cdsKcmKeyProfilesLocalUriGenerated(1)	Number of URIs generated locally.
	cdsKcmKeyProfilesKeyCreateSent(2)	Number of key create messages sent.
	cdsKcmKeyProfilesKeyRequestSent(3)	Number of key request messages sent.
	cdsKcmKeyProfilesKeyDeleteSent(4)	Number of key delete messages sent.
	cdsKcmKeyProfilesKeyReqInvalidOrRejected(5)	Number of key requests that are invalid or are rejected.
	cdsKcmKeyProfilesKeyReqPendingOnDns(6)	Number of key requests pending on DNS resolve.
	cdsKcmKeyProfilesKeyReqRespRcvdSuccess(7)	Number of received success responses for key request.
	cdsKcmKeyProfilesActive(8)	Number of active key profiles.

Table 6 describes the OIDs in the UCD-SNMP-MIB that are used to monitor system core statistics. The UCD-SNMP-MIB is located in the spced/bfc/systems/snmp/src/snmpd/src/snmpd/extend/ce/ directory.

Table 6 Object IDs in UCD-SNMP-MIB

Group	Object ID	Description
memory Group	memIndex(1)	This index always returns the integer 0.
	memErrorName(2)	This parameter always return the string “swap.”
	memTotalSwap(3)	Total amount of swap space configured for this host.
	memAvailSwap(4)	Amount of swap space currently unused or available.
	memTotalReal(5)	Total amount of real, physical memory installed on this host.
	memAvailReal(6)	Amount of real, physical memory currently unused or available.
	memTotalFree(11)	Total amount of memory free or available for use on this host. This value typically covers both real memory and swap space or virtual memory.
	memMinimumSwap(12)	Minimum amount of swap space expected to be kept free or available during normal operation of this host. If this value (as reported by memAvailSwap(4)) falls below the specified level, then memSwapError(100) is set to 1 and an error message is made available (memSwapErrorMsg(101)).
	memBuffer(14)	Total amount of real or virtual memory currently allocated for use as memory buffers. This object is not implemented on hosts where the underlying operating system does not explicitly identify memory as specifically reserved for this purpose.
	memCached(15)	Total amount of real or virtual memory currently allocated for use as cached memory. This object is not implemented on hosts where the underlying operating system does not explicitly identify memory as specifically reserved for this purpose.
	UCDErrorFlag memSwapError(100)	Indicates whether the amount of available swap space (as reported by memAvailSwap(4)), is less than the desired minimum (specified by memMinimumSwap(12)).
	memSwapErrorMsg(101)	Describes whether the amount of available swap space (as reported by memAvailSwap(4)), is less than the desired minimum (specified by memMinimumSwap(12)).
laTable Group	laIndex(1)	Reference index/row number for each observed loadave.
	laNames(2)	List of loadave names being watched.
	laLoad(3)	Load averages (1,5, and 15 minute), one per row.
	laConfig(4)	Watch point for load-averages to signal an error. If the load averages rises above this value, the laErrorFlag is set.
	laLoadInt(5)	Load averages (1, 5, and 15 minute) as an integer. This is computed by taking the floating point load average value and multiplying it by 100, then converting the value to an integer.
	laLoadFloat(6)	Load averages (1, 5, and 15 minute) as an opaquely-wrapped floating point number.
	UCDErrorFlag laErrorFlag(100)	Error flag to indicate the load-average has crossed its threshold value defined in the snmpd.conf file. It is set to 1 if the threshold is crossed, 0 otherwise.
	laErrMessage(101)	Error message describing the load-average and its surpassed watch-point value.

Table 6 Object IDs in UCD-SNMP-MIB

Group	Object ID	Description
systemStats Group	ssIndex(1)	This index should always return the integer 1.
	ssErrorName(2)	This parameter should always return the string "systemStats."
	ssSwapIn(3)	Average amount of memory swapped in from disk, calculated over the last minute.
	ssSwapOut(4)	Average amount of memory swapped out to disk, calculated over the last minute.
	ssIOSent(5)	Average amount of data written to disk or other block device, calculated over the last minute. This object has been deprecated in favor of ssIORawSent(57), which can be used to calculate the same metric, but over any desired time period.
	ssIOReceive(6)	Average amount of data read from disk or other block device, calculated over the last minute. This object has been deprecated in favor of ssIORawReceived(58), which can be used to calculate the same metric, but over any desired time period.
	ssSysInterrupts(7)	Average rate of interrupts processed (including the clock) calculated over the last minute. This object has been deprecated in favor of ssRawInterrupts(59), which can be used to calculate the same metric, but over any desired time period.
	ssSysContext(8)	Average rate of context switches, calculated over the last minute. This object has been deprecated in favor of ssRawContext(60), which can be used to calculate the same metric, but over any desired time period.
	ssCpuUser(9)	Percentage of CPU time spent processing user-level code, calculated over the last minute. This object has been deprecated in favor of ssCpuRawUser(50), which can be used to calculate the same metric, but over any desired time period.
	ssCpuSystem(10)	Percentage of CPU time spent processing system-level code, calculated over the last minute. This object has been deprecated in favor of ssCpuRawSystem(52), which can be used to calculate the same metric, but over any desired time period.
	ssCpuIdle(11)	Percentage of processor time spent idle, calculated over the last minute. This object has been deprecated in favor of ssCpuRawIdle(53), which can be used to calculate the same metric, but over any desired time period.
	ssCpuRawUser(50)	Number of "ticks" (typically 1 per 100 seconds) spent processing user-level code. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssCpuRawNice(51)	Number of "ticks" (typically 1 per 100 seconds) spent processing reduced-priority code. This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssCpuRawSystem(52)	Number of "ticks" (typically 1 per 100 seconds) spent processing system-level code. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors). This object may sometimes be implemented as the combination of the ssCpuRawWait(54) and ssCpuRawKernel(55) counters, so care must be taken when summing the overall raw counters.
	ssCpuRawIdle(53)	Number of "ticks" (typically 1 per 100 seconds) spent idle. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).

Table 6 Object IDs in UCD-SNMP-MIB

Group	Object ID	Description
systemStats Group (continued)	ssCpuRawWait(54)	Number of "ticks" (typically 1 per 100 seconds) spent waiting for IO. This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. This time may also be included within the ssCpuRawSystem(52) counter. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssCpuRawKernel(55)	Number of "ticks" (typically 1 per 100 seconds) spent processing kernel-level code. This object is not implemented on hosts where the underlying operating system doe s not measure this particular CPU metric. This time may also be included within the ssCpuRawSystem(52) counter. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssCpuRawInterrupt(56)	Number of "ticks" (typically 1 per 100 seconds) spent processing hardware interrupts. This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssIORawSent(57)	Number of blocks sent to a block device.
	ssIORawReceived(58)	Number of blocks received from a block device.
	ssRawInterrupts(59)	Number of interrupts processed.
	ssRawContexts(60)	Number of context switches.
	ssCpuRawSoftIRQ(61)	Number of "ticks" (typically 1 per 100 seconds) spent processing software interrupts. This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssRawSwapIn(62)	Number of blocks swapped in.
	ssRawSwapOut(63)	Number of blocks swapped out.
	ssCpuRawSteal(64)	Number of "ticks" (typically 1 per 100 seconds) spent by the hypervisor code to run other VMs even though the CPU in the current VM had something runnable. This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
	ssCpuRawGuest(65)	Number of "ticks" (typically 1 per 100 seconds) spent by the CPU to run a virtual CPU (guest). This object is not implemented on hosts where the underlying operating system does not measure this particular CPU metric. On a multiprocessor system, the ssCpuRaw* counters are cumulative over all CPUs, so their sum is typically N*100 (for N processors).
version Group	versionIndex(1)	Index to MIB (always 0).
	versionTag(2)	CVS tag keyword.
	versionDate(3)	Date string from RCS keyword.
	versionCDate(4)	Date string from ctime().

System Requirements

The Internet Streamer CDS runs on the CDE205, CDE220, and the CDE250 hardware models.

Table 7 lists the different device modes for the Cisco Internet Streamer CDS software, and which CDEs support them.

Table 7 Supported CDEs

Device Mode	CDE205	CDE220-2G2	CDE220-2S3i	CDE250 (all models)	UCS C200	UCS C210
CDSM	Yes	No	No	No	Yes	No
SR	Yes	Yes	No	No	Yes	No
SE	Yes	Yes	Yes	Yes	No	Yes
SR—Proximity Engine standalone	Yes	Yes	No	No	No	No

The new CDE250 models (CDE250-2S8, CDE250-2S9, and CDE250-2S10) have four interfaces at 10 gigabit Ethernet speeds and four interfaces at gigabit Ethernet speeds (plus two additional gigabit ethernet interfaces for management).

The new CDE250 models only support the SE device mode and have the following storage capacities:

- CDE250-2S8—24 x 300 GB 2.5 SSD
- CDE250-2S9—12 x 600 GB 2.5 SSD
- CDE250-2S10—24 x 600 GB 2.5 SSD

The Cisco UCS models (UCS C200 and UCS C210) and the Cisco Internet Streamer Release 3.0 software are sold separately and ship independently of each other.

CDE250-2S6 and CDE250-2M0 platforms have four interfaces at 10 gigabit Ethernet speeds and four interfaces at gigabit Ethernet speeds (plus two additional gigabit ethernet interfaces for management).

The CDE220-2S3i platform has a total of 14 gigabit Ethernet ports in this CDE. The first two ports (1/0 and 2/0) are management ports. The remaining 12 gigabit Ethernet ports can be configured as two port channels. See the *Cisco Content Delivery Engine CDE205/220/250/420 Hardware Installation Guide* for set up and installation procedures for the CDE220-2S3i and the *Cisco Internet Streamer CDS 2.6 Software Configuration Guide* for information on configuring the Multi Port Support feature.

The CDE220-2G2 platform has a total of ten gigabit Ethernet ports. The first two ports (1/0 and 2/0) are management ports. The remaining eight gigabit Ethernet ports can be configured as one port channel. See the *Cisco Content Delivery Engine CDE205/220/250/420 Hardware Installation Guide* for set-up and installation procedures for the CDE220-2G2.

The CDE205 can run as the CDSM, SR or SE. See the *Cisco Content Delivery Engine CDE205/220/250/420 Hardware Installation Guide* for set-up and installation procedures for the CDE205.

Note For performance information, see the release-specific performance bulletin.

Limitations and Restrictions

This release contains the following limitations and restrictions:

- There is a 4 KB maximum limit for HTTP request headers. This has been added to prevent client-side attacks, including overflowing buffers in the Web Engine.
- Standby interface is not supported for Proximity Engine. Use port channel configuration instead.
- There is no network address translation (NAT) device separating the CDEs from one another.
- Do not run the CDE with the cover off. This disrupts the fan air flow and causes overheating.

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.

System Limits and Thresholds

This release has the following limits and thresholds:

- Service Router Limits and Thresholds
- Service Monitor Limits and Thresholds
- Web Engine Limits and Thresholds
- CDSM Limits and Thresholds
- RTSP Gateway and Movie Streamer
- Windows Media Streaming
- Flash Media Streaming

Service Router Limits and Thresholds

The Service Router has memory-related limits and thresholds. Memory usage of the Service Router depends on the number of coverage zone entries, the number of Content Origin servers, the distribution of subnets in the Coverage Zone file, and the number of Service Engines in the CDS. From our tests using a sample Coverage Zone file, we have observed that we can support 20,000 Coverage Zone entries with 26 SEs, and 40 Content Origins servers.

Note The number of Coverage Zone entries, SEs, and Content Origin servers are subject to change depending on the Coverage Zone configured.

We recommend keeping the memory usage (both virtual and resident) below 1.5 GB.

Frequent configuration updates could cause memory fragmentation, which raises the memory usage.

Service Monitor Limits and Thresholds

When the Service Monitor thresholds are exceeded, an alarm is raised on the respective device and an SNMP trap is sent to the CDSM. The parameters monitored and thresholds for each component or protocol engine can be modified. The default thresholds are as outlined below.

Following are the parameters that are monitored on each device (SE, SR, and CDSM) and the default threshold setting of each parameter:

- CPU—80 percent
- Memory—80 percent
- Kernel memory—50 percent
- Disk usage—0 percent
- Disk failures—75 percent
- Augmentation alarms—80 percent

Following are the parameters that are monitored only on the SE, along with default threshold setting of each parameter:

- Windows Media Streaming thresholds—90 percent
- Flash Media Streaming thresholds—90 percent
- Movie Streamer—90 percent%
- Maximum number of concurrent sessions—200
- Maximum Bandwidth—200,000 kbps
- NIC bandwidth—90 percent
- Burst Count—1

Web Engine Limits and Thresholds

The Web Engine has the following limits and thresholds:

- Memory Usage
- Session Limits
- CAL Limits

Memory Usage

In Release 2.5.9, the memory threshold on each SE is 3.2 GB. If the threshold is exceeded, the memory_exceeded alarm is raised and trickle mode is enabled. In Release 2.5.9, the admission control is based on 30,000 session and 3.2 GB of memory.

In Release 2.6.1, the memory threshold on each SE is 3.2 GB. If the threshold is exceeded, the memory_exceeded alarm is raised. In cases where the memory reaches 3.7 GB, trickle mode is enabled and eventually the Web Engine is restarted. The above memory values, and the 20,000–60,000 sessions and 100,000 open file/socket descriptor (FD) limit are used for admission control in Release 2.6.1.

Session Limits

Web Engine supports the following session-threshold limits:

- 49,800 session count for the CDE250
- 15,000 session count for all other CDEs

The max_session_exceeded alarm is raised if the session-threshold limit is reached. If further requests are sent to the SE even when the session threshold is reached, the Web Engine attempts to process the requests but does not accept any more requests when the request count reaches 60,000 on a CDE250, and 20,000 on all other CDEs.

CAL Limits

Outstanding CAL Lookup threshold is 25,000 on the CDE250 and 15,000 on all other CDEs. The WebCalLookupThreshold alarm is raised on reaching this threshold limit.

Outstanding CAL disk Write threshold is 3,000 CAL requests (create, update, delete, popularity update) on the CDE250, and 1,500 on all other CDEs. The WebCalDiskWriteThreshold alarm is raised on reaching this threshold.

Other CAL thresholds are as follows:

- File Descriptor usage threshold is 85 percent
- TEMPFS usage threshold is 80 percent
- Active datasource threshold is 2,000

Note CAL-related thresholds and the File Descriptor-related thresholds are introduced in Release 2.6.1.

Web Engine thresholds are also applicable to adaptive bit rate (ABR) streaming.

CDSM Limits and Thresholds

The CDSM has the following limits and thresholds:

- RPC Connections
- File Synchronization
- CDSM Availability (primary and standby)
- SE Configuration Change Synchronization

RPC Connections

A maximum of 40 RPC connections are supported among the managed devices (SE, SR, standby CDSM, and primary CDSM). The RPC connection maximum is defined in the httpd.conf.rpc configuration file located in the /state directory.

File Synchronization

The primary CDSM checks for file updates and synchronization with the managed devices (SE, SR, and standby CDSM) every ten minutes.

CDSM Availability (primary and standby)

The SE and SR check for the availability of the primary and standby CDSM on a regular interval; however, if the CDSM does not respond, the SE and SR use an exponential-backoff call for retrying the connection.

The exponential backoff call means that if the CDSM does respond to the first attempt, the SE or SR sleep for ten seconds before trying again. If the second attempt does not succeed, the wait time doubles (20 seconds), if that attempt does not succeed, the wait time doubles again (40 seconds). The wait time doubles every attempt (10, 20, 40, 80, and so on) until the maxWaitingTime of 320 seconds.

SE Configuration Change Synchronization

The period of time before the local configuration manager (LCM) on an SE sends a configuration change to the primary CDSM is a maximum of 2.25 times the polling rate. The polling rate is configurable through the CDSM GUI (**System > Configuration > System Properties**, System.datafeed.pollRate).

RTSP Gateway and Movie Streamer

The default RTSP Gateway transactions per second (tps) is 40. There are no other limits to the RTSP Gateway.

The Movie Streamer default maximum concurrent session is 200 and the default maximum bandwidth is 200 Mbps.

Windows Media Streaming

Windows Media Streaming has the following limits and thresholds:

- Windows Media Streaming recommended concurrent remote server sessions 300

Note Regarding concurrent remote server sessions, if all requests are unique cache-miss cases, Windows Media Streaming can reach up to 1000 sessions of 1 Mbps file each. Windows Media Streaming can sustain 1000 remote server sessions at most if the Content Origin server can respond, but the recommended value is 300.

- Windows Media Streaming transactions per second is 40 (because of the RTSP Gateway limitation).
- Memory threshold 3 GB
- CPU threshold is 80 percent

Flash Media Streaming

With the basic license, Flash Media Streaming the default maximum concurrent sessions is 200 and the default maximum bandwidth is 200 Mbps.

Buying more licenses can increase the concurrent sessions and maximum bandwidth as follows:

- CDE220-2G2 and CDE220-2S3—15,000 concurrent sessions and 8 Gbps maximum bandwidth
- CDE250-2M0—40,000 concurrent sessions and 40 Gbps maximum bandwidth

We recommend that the Flash Media Streaming process memory usage not exceed 3 GB resident set size (RSS). If the memory usage for Flash Media Streaming exceeds 3 GB RSS, a threshold exceeded alarm is raised.

Note RSS is the portion of a process that exists in physical memory (RAM), as opposed to virtual memory size (VSIZE), which includes both RAM and the amount in swap. If the device has not used swap, the RSS number is equal to VSIZE.

Important Notes

To maximize the content delivery performance of a CDE205, CDE220, or CDE250, we recommend you do the following:

1. Use port channel for all client-facing traffic.

   Configure interfaces on the quad-port gigabit Ethernet cards into a single port-bonding interface. Use this bonding channel, which provides instantaneous failover between ports, for all client-facing traffic. Use interfaces number 1 and 2 (the two on-board Ethernet ports) for intra-CDS traffic, such as management traffic, and configure these two interfaces either as standby or port-channel mode. Refer to the *Cisco Internet Streamer CDS 2.6 Software Configuration Guide* for detailed instruction.

2. Use the client IP address as the load balancing algorithm.

   Assuming ether-channel (also known as port-channel) is used between the upstream router/switch and the SE for streaming real-time data, the ether-channel load balance algorithms on the upstream switch/router and the SE should be configured as "Src-ip" and "Destination IP" respectively. Using this configuration ensures session stickiness and general balanced load distribution based on clients' IP addresses. Also, distribute your client IP address space across multiple subnets so that the load balancing algorithm is effective in spreading the traffic among multiple ports.

   Note The optimal load-balance setting on the switch for traffic between the Content Acquirer and the edge Service Engine is dst-port, which is not available on the 3750, but is available on the Catalyst 6000 series.

3. For high-volume traffic, separate HTTP and WMT.

   The CDE205, or CDE220 performance has been optimized for HTTP and WMT bulk traffic, individually. While it is entirely workable to have mixed HTTP and WMT traffic flowing through a single server simultaneously, the aggregate performance may not be as optimal as the case where the two traffic types are separate, especially when the traffic volume is high. So, if you have enough client WMT traffic to saturate the full capacity of a server, we recommend that you provision a dedicated server to handle WMT; and likewise for HTTP. In such cases, we do *not* recommended that you mix the two traffic types on all CDE servers which could result in suboptimal aggregate performance and require more servers than usual.

4. For mixed traffic, turn on the HTTP bitrate pacing feature.

   If your deployment must have Streamers handle HTTP and WMT traffic simultaneously, it is best that you configure the Streamer to limit each of its HTTP sessions below a certain bitrate (for example, 1Mbps, 5Mbps, or the typical speed of your client population). This prevents HTTP sessions from running at higher throughput than necessary, and disrupting the concurrent WMT streaming sessions on that Streamer. To turn on this pacing feature, use the HTTP bitrate field in the CDSM Delivery Service GUI page.

   Please be aware of the side effects of using the following commands for Movie Streamer:

   ```
   Config# movie-streamer advanced client idle-timeout <30-1800>
   Config# movie-streamer advanced client rtp-timeout <30-1800>
   ```

   These commands are only intended for performance testing when using certain testing tools that do not have full support of the RTCP receiver report. Setting these timeouts to high values causes inefficient tear down of client connections when the streaming sessions have ended.

   For typical deployments, it is preferable to leave these parameters set to their defaults.

5. For ASX requests, when the Service Router redirects the request to an alternate domain or to the origin server, the Service Router does not strip the .asx extension, this is because the .asx extension is part of the original request. If an alternate domain or origin server does not have the requested file, the request fails. To ensure requests for asx files do not fail, make sure the .asx files are stored on the alternate domain and origin server.

Open Caveats

This release contains the following open caveats:

Content Manager

- CSCua08680

  Symptom:

  When the **clear-cache-all** command is entered, sometimes the content is not completely deleted. This is because the Content Manager is not aware of all the content cached by the Web Engine.

  Conditions:

  When the Web Engine creates more than three million objects and the slow scan (slowscan) process has just finished running, some content is not known by the Content Manager.

  Workaround:

  The next round of slowscan (every 12 hours) picks up the content that was unknown by the Content Manager. After that, the **clear-cache-all** command works.

Web Engine

- CSCub00586

  Symptom

  Session-based Encryption with a single key per session (default configuration) does not work for HLS use case. HSS use case has no impact because the manifest for HSS is out-of-band.

  Conditions:

  When HLS encryption is enabled with a single key for a session (user).

  Workaround:

  To enable HLS Session-based Encryption with a single key, when the HLSEncryptionEnable parameter is set to 1 in the Service Rule XML file, configure HLSMaxKeysPerSession =1 and HLSKeyRotationFragmentInterval =1. This enables HLS encryption with a single key for the entire session.

- CSCtz10789

  Symptom:

  Web Engine process fails to start.

  Conditions:

  Web Engine fails to start when restarted manually.

Workaround:

Retry. Reload the Web Engine.

- CSCtz27194

  Symptom:

  When 24 IPv6 addresses are added to one interface and 12 or more IPv6 addresses are added another interface using a script (that is, apply all of them together), a Web Engine crash is seen occasionally.

  Conditions:

  This is seen only when a large number of IPv6 addresses (more than 24) are applied to more than one interface using a script. It is not seen when 24 IPv6 addresses are applied to just one interface. Or, when addresses are applied using the CLI.

  Workaround:

  If more than 24 addresses are required to be applied to an interface, then one of the following could be done:

  - Apply all the IPv6 addresses using CLI
  - Apply 24 addresses to one interface using a script, wait for short duration to see if all the addresses are applied; then apply successive IPv6 addresses to other interfaces either through a script or through CLI (preferred mechanism).

Platform

- CSCua50262

  Symptom:

  When converting a CDE250-2S10 to a CDE250-2S9, the **show disk details** command still show the 24 SSDs for a CDE250-2S10, instead of the 12 SSDs for the CDE250-2S9.

  Conditions:

  This happen when changing the CDE250 from a CDE250-2S10 to a CDE250-2S9.

  In a CDE250-2S9, there are 12 INTEL SSDSA2BW60 600G external SSD from disk00 slot to disk11 slot.

  In a CDE250-2S10, there are 24 INTEL SSDSA2BW60 600G external SSD in disk00 slot to disk23 slot.

  The **show disk details** command displays the disk slots and hard drive type installed in each slot.

  Workaround:

  Upgrade the software after changing from a CDE250-2S10 to a CDE250-2S9 resolves this issue. You can use the same software image that is currently installed. It is the software upgrade process that allows the system to recognize the new hardware configuration.

Resolved Caveats

The following caveats have been resolved since Cisco Internet Streamer CDS Release 3.0.0. Not all the resolved issues are mentioned here. The following list highlights the resolved caveats associated with customer deployment scenarios.

Flash Media Streaming

- CSCtz57201

  Symptom:

  Flash Media Streaming cache miss goes to the Content Acquirer instead of serving from the disk, when the Stream and Cache-Fill Performance feature is disabled.

  Conditions:

  This only happens when the Stream and Cache-Fill Performance feature is disabled and there is a Flash Media Streaming request.

- CSCtz56592

  Symptom:

  With Flash Media Streaming cache miss, coredumps were observed with around 100 simulated users.

  Conditions:

  Run a stress test with 100, or so, all unique cache miss requests. Start noticing Web Engine coredumps.

Platform

- CSCtz08128

  Symptom:

  Power unit removal may not generate alarm.

  Conditions:

  Some older power units may not trigger an alarm after removal. The issue is on Supermicro, third-party hardware.

- CSCty81853

  Symptom:

  If a UCS platform (for example, UCS model 210) contains a MegaRAID controller, there is currently a limitation of the Release 3.0.0 software wherein the corresponding SMART health statistics for all drives behind the MegaRAID controller cannot be reported using the **show disks SMART-info** command and the s**how tech-support** command.

  For example, the **show disks SMART-info** command produces the following results:

```
U10-210-1#show disks SMART-info details
=== disk00 ===
smartctl 5.40 2010-10-16 r3189 [i686-pc-linux-gnu] (local build)
Copyright (C) 2002-10 by Bruce Allen, http://smartmontools.sourceforge.net
```

```
Device: LSI MR9261-8i Version: 2.12
Serial number: 00cd15660d02b1fc1640e3c403b00506
Device type: disk
Local Time is: Wed Mar 21 16:20:58 2012 UTC
Device does not support SMART
... etc ...
```

In addition, there is a secondary issue, if a disk sector I/O error occurs on a drive, a “badsector” alarm is only temporarily raised against this drive. The corresponding “badsector” alarm is cleared when the next disk health monitoring poll cycle occurs (scheduled every 30 minutes).

Conditions:

A UCS-based system containing a MegaRAID controller (for example, MegaRAID model 9261-8i).

Service Router

- CSCtz62040

  Symptom:

  After modifying the IP address on the SR primary interface, it is not responding to A and AAAA queries.

  Conditions:

  The IP address on the SR primary interface is modified.

- CSCty71678

  Symptom:

  TCP connection half-closed when Quova server restarts.

  Conditions:

  If Geo-Location server restarted when the SR is processing a request, the SR skips location routing and falls back to the next routing method.

Windows Media Streaming

- CSCtz47599

  Symptom:

  Windows Media Streaming multicast log file may have truncated username fields.

  Conditions:

  Windows Media Streaming playback.

- CSCtz47575

  Symptom:

  There is coredump for the wmt_mbe process.

  Conditions:

  Multicast program playing for more than two days.

Web Engine

- CSCtz57653

  Symptom:

  High-session alarm is triggered for HTTPS at 2,000 session and 1.8 GB memory usage.

  Conditions:

  While running stress test for HTTPS.

- CSCtz59286

  Symptom:

  Content assets and content files are not freed up when SE fails to serve range requests for .aac files.

  Conditions:

  Play a video asset using VLC player, with cache-fill disabled and session resolve rules not configured to allow requests for .aac files. As a result, SE does not serve range requests for .aac files to the client player.

- CSCtz57552

  Symptom:

  During an HTTP streaming session, if the Origin server fails or if HTTP server on the Origin server fails, Web Engine may core dump.

  Conditions:

  This is a race condition that happens if something catastrophic happens to the Origin server socket that is being used to stream data to the SE.

- CSCtz26952

  Symptom:

  The output from the **show stat web-engine detail** command shows non-zero number of sessions for "Active HTTPSession," even when there are no active HLS requests coming to the SE.

  Conditions:

  This happens, when HLS session tracking with Session-Based Encryption are enabled on an SE, and the SE has a large number of HLS requests coming in.

- CSCtz33187

  Symptom:

  Web Engine reloads intermittently.

  Conditions:

  If revalidation is mandated by using the **web-engine revalidation must-validate** command on the SE, or if 100 simulated users are created for HLS VOD content.

- CSCtz07692

  Symptom:

  There is throughput fluctuations.

  Conditions:

  Create a new port channel and assign an IP address to it.

Upgrading to Release 3.0.0

Release 3.0.0 supports upgrades from Release 2.5.9, Release 2.5.11, Release 2.6.1, and Release 2.6.3.

When upgrading from Release 2.5.9 or 2.5.11, all content is erased. For Service Engines, this means that prefetched metadata and content need to be redistributed from upstream SEs after the upgrade, and that cached content is not preserved. Additionally, Flash Media Streaming service rules must be converted from device-based service rules to the Service Rule XML file. For more information on upgrading from Release 2.5.9 and 2.5.11, see the *Cisco Internet Streamer CDS 2.6 Software Upgrade Guide* (http://www.cisco.com/en/US/docs/video/cds/cda/is/2_6/upgrade_guide/upgrade.html).

Note If your CDS software is older than Release 2.6.1 and you have CDE205 and CDE220 platforms in your system, you must check that the partition size (specifically, disk 00/02), on each CDE205 and CDE220 in your system is larger than 0.5 GB. To check the partition size, enter the **show disks detail** command. If the disk00/02 partition is not larger than 0.5 GB, you must upgrade the CDE to Release 2.6.1 before upgrading to Release 3.x.

If your CDS is running an older release than Release 2.5.9, you need to upgrade to Release 2.5.9, or 2.5.11 before upgrading to Release 3.0.0.

We strongly recommend that you upgrade your CDS network devices in the following order:

1. Multicast sender Service Engines
2. Multicast receiver Service Engines
3. Edge Service Engines
4. Middle-tier Service Engines
5. Content Acquirers
6. Service Routers
7. Standby CDSMs (Upgrade before primary when using the GUI only.)
8. Primary CDSM

Note When using the CDSM GUI to upgrade from Release 2.5.9, 2.5.11, or 2.6.1 to Release 3.1.0, after you upgrade the standby CDSM, if you switch roles of the standby CDSM and primary CDSM to maintain an active CDSM, the old primary CDSM is now the standby CDSM, and the old standby CDSM is now the primary CDSM. At this point, you must use the CLI to upgrade the new standby CDSM. The primary CDSM GUI cannot upgrade the standby CDSM.

Alternatively, if you do not switch roles of the standby CDSM and primary CDSM, you can use the CDSM GUI to upgrade the primary CDSM. The primary CDSM loses connectivity with the CDS devices for a short time during the upgrade, but this is not service affecting.

When using the CDSM GUI to upgrade from Release 2.6.3 and later releases to Release 3.10, after you upgrade the standby CDSM, if you switch roles of the standby CDSM and primary CDSM to maintain an active CDSM at all times, the new primary CDSM GUI can be used to upgrade the new standby CDSM.

For information on the upgrade procedure from Release 2.6.x to Release 3.0.0, see the *Cisco Internet Streamer CDS 3.0 Software Configuration Guide.*

After the upgrade procedure starts, do not make any configuration changes until all the devices have been upgraded.

Downgrading from Release 3.0.0 to Release 2.6.x

For software downgrades from Release 3.0.0 to Release 2.6.x on systems with primary and standby CDSMs, you need to do the following:

Step 1 If you are using the CDSM GUI, downgrade the standby CDSM first, followed by the primary CDSM.

If you are using the CLI, downgrade the primary CDSM first, followed by the standby CDSM.

Step 2 After downgrading the primary and standby CDSMs, using the CLI, log in to each CDSM and run the following commands:

```
cms database downgrade
cms database downgrade script downgrade/Downgrade3_0_to_2_6
cms enable
```

Step 3 Downgrade the software on the Service Routers, followed by the Service Engines.

Note If you are downgrading the CDSM from Release 3.0.0 to Release 2.6.x, after running the cms database downgrade command, run the downgrade/Downgrade3_0_to_2_6 command.

Documentation Updates

The following document has been added for this release:

- *Release Notes for Cisco Internet Streamer CDS 3.0.0*
- *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*
- *Cisco Internet Streamer CDS 3.0 Command Reference Guide*
- *Cisco Internet Streamer CDS 3.0 Quick Start Guide*
- *Cisco Internet Streamer CDS 3.0 Alarms and Error Message Guide*
- *Cisco Internet Streamer CDS 3.0 API Guide*
- *Cisco Internet Streamer CDS 3.0 Software Installation Guide for non-CDEs*
- *Cisco Content Delivery Engine 205/220/250/420 Hardware Installation Guide*

Related Documentation

Refer to the following documents for additional information about the Cisco Internet Streamer CDS 3.0:

- *Cisco Internet Streamer CDS 3.0 Software Configuration Guide*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/configuration_guide/is_cds30-cfguide.html
- *Cisco Internet Streamer CDS 3.0 Quick Start Guide*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/quick_guide/ISCDSQuickStart.html
- *Cisco Internet Streamer CDS 3.0 API Guide*
  http://www.cisco.com/en/US/docs/video/cds/cda/is3_0/developer_guide/is_cds_30_apiguide.html
- *Cisco Internet Streamer CDS 3.0Command Reference Guide*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/command_reference/Command_Ref.html
- *Cisco Internet Streamer CDS 3.0 Alarms and Error Messages Guide*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/message_guide/messages.html
- *Cisco Internet Streamer CDS 3.0 Software Installation Guide for non-CDEs*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/install_guide/Non_CDE_IS_3_0_Software_Install.html
- *Cisco Content Delivery System 3.x Documentation Roadmap*
  http://www.cisco.com/en/US/docs/video/cds/overview/CDS_Roadmap3.x.html
- *Open Source Used in Cisco Internet Streamer CDS 3.0*
  http://www.cisco.com/en/US/docs/video/cds/cda/is/3_0/third_party/open_source/OL-25149-01.pdf
- *Cisco Content Delivery Engine 205/220/250/420 Hardware Installation Guide*
  http://www.cisco.com/en/US/docs/video/cds/cde/cde205_220_420/installation/guide/cde205_220_420_hig.html
- *Regulatory Compliance and Safety Information for Cisco Content Delivery Engines*
  http://www.cisco.com/en/US/docs/video/cds/cde/regulatory/compliance/CDE_RCSI.html
- *Cisco UCS C200 Installation and Service Guide*
  http://www.cisco.com/en/US/docs/unified_computing/ucs/c/hw/C200M1/install/c200M1.html
- *Cisco UCS C210 Installation and Service Guide*
  http://www.cisco.com/en/US/docs/unified_computing/ucs/c/hw/C210M1/install/C210M1.html

The entire CDS software documentation suite is available on Cisco.com at:

http://www.cisco.com/en/US/products/ps7127/tsd_products_support_series_home.html

The entire CDS hardware documentation suite is available on Cisco.com at:

http://www.cisco.com/en/US/products/ps7126/tsd_products_support_series_home.html

The Cisco UCS C-Series Rack Servers documentation is available on Cisco.com at:

http://www.cisco.com/en/US/products/ps10493/prod_installation_guides_list.html

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

This document is to be used in conjunction with the documents listed in the "Related Documentation" section.