Compaq ActiveAnswers

Technical Guide

July 2001
155C-0701A-WWEN
Prepared by:
Enterprise Applications Solutions and Services
Compaq Computer Corporation

Compaq TaskSmart C-Series Streaming Servers Deployment Guide

Abstract: State-of-the-art websites continue to attract users with increasingly rich content that expands the viewer experience and delivers denser value than traditional text-based websites. Websites, ranging from education to entertainment, are turning to streaming media technologies as the next generation of rich content. Streaming media invites levels of interaction and satisfaction that even the most complex hypertext-based websites cannot generate.

Traditionally, streaming media has been difficult to implement because of scalability limitations and delivery uncertainty. The latest generation of TaskSmart C-Series servers delivers industry-defining support for distributed media delivery networks and edge media caches. This guide will demonstrate and clarify the common scenarios for implementing this technology.

This guide provides an introduction to the available streaming technologies, explains the benefit delivered by the addition of a TaskSmart C-Series server, and demonstrates the functional and architectural principles of the deployment.

Contents

Notice

Streaming Media

It seems that everything from Napster to satellite TV is called "streaming media." However, for the purposes of this deployment guide, streaming media will refer only to the audio and/or video delivered to a computer over IP networks via one or all of the three dominant Internet streaming packages:

- Windows Media Technologies (WMT)
- RealMedia (Real)
- Apple QuickTime (QT)

These three formats are supported by the Compaq *TaskSmart*™ C-Series Model 50 server and represent virtually all of the streaming content on the Internet today.

Introduction

Streaming media as network traffic presents many of the familiar deployment issues associated with traditional Web servers, but amplifies and complicates the problems by requiring bandwidth that is not only orders of magnitude greater than what is required for a less-rich content but also more reliable and more consistent. Consider the effect of network latency and congestion on HTML and embedded binaries. To an end user, this network appears to be "slow." But, in a few seconds, all of the text will be displayed and all of the images will load; everything will be legible and is presented as the author intended. For streaming media, this same adverse network condition may deliver content that is, at best, annoying and jerky; at worst, useless and frustrating. Delay and packet loss for streamed content can lead to jerky pauses, unviewable audio/video and rebuffering during playback. While a "slow" site is certainly not the most enjoyable user experience, it is considerably better than a site that delivers unintelligible content.

Streaming Formats and Protocols

To understand how the TaskSmart C-Series servers and caching can ensure the highest quality streaming media, it may be helpful to examine the protocols and components of streaming traffic. While there has been a tremendous effort in creating ubiquitous standards for streaming media, currently, all three dominant streaming formats are, in some way, proprietary and independent (although, a RealServer is capable of distributing QuickTime content). Despite the differences, all streaming protocols operate similarly in that they all employ separate data and control channels. The control channel carries the client requests and commands to the server and the server replies to the client. The data channel moves the actual encoded media stream from server to client.

Table 1 details the control and data protocols used by the three streaming formats.

Table 1. Streaming Formats and Associated Protocols

Streaming Format	Control Protocol	Data Transfer Protocol	Protocol Ownership	Underlying Transport Protocol
RealNetworks	RTSP	RDT (formerly known as RDP)	RTSP is Internet RFC-based RDT is proprietary	UDP or TCP
WMT (currently)	MMS	MMST	Proprietary	TCP
WMT (currently)	MMS	MMSU	Proprietary	UDP
WMT (planned)	RTSP	RTP	Internet RFC-based	UDP or TCP
QuickTime	RTSP	RTP	Internet RFC-based	UDP or TCP
Note: Currently only Model 50 of the TaskSmart C-series servers supports streaming media. In the future, other models will support streaming media.				

QuickTime seems to have the most standards compliance – QuickTime uses its own Real-Time Streaming Protocol (RTSP) implementation to send requests and set playback parameters. When a QuickTime client issues a request for a QuickTime multi-bit-rate media clip, information describing the client's QuickTime configuration is sent to the server. The server can respond with the most appropriate version of the media, based on the client configuration: QuickTime can use parameters such as client bandwidth, client language, QuickTime player version, computer platform and even computer speed to determine which clip should be sent to the client. While all of these parameters are sent over the control connection or control channel, they are not part of the RTSP specifications. For the streaming data channel, QuickTime uses another standard, Real-time Transport Protocol (RTP), to provide data sequencing and lost media packet detection.

RealMedia relies on a proprietary protocol, RealNetworks Data Transport (RDT), for the data channel, but uses a modified RTSP implementation for requests and control. While both QuickTime RTSP and RealNetworks' RTSP are based on the same standard, they are not interchangeable because of the proprietary components to support each format's multi-bit-rate implementation.

Microsoft Windows Media Technologies (WMT) is currently entirely proprietary, that is the data channel communications, the control channel communications and the packet format are not based on any industry standard. The Microsoft Media Services (MMS) protocols and formats define all Windows streaming communication and operation for both the control and the data channels. Microsoft has expressed intentions to migrate to the standards RTSP and RTP for the upcoming Microsoft Windows Media 8.

Multi-bit-rate streaming

Like Real and QuickTime, WMT relies on the proprietary formats to implement features that extend beyond the current version of the standards.

One such feature is support for multi-bit-rate streaming. With multi-bit-rate streaming, information exchanged between the client and server or between the client and cache allows the client to receive the most appropriate version of the stream for the current network environment between the client and the server. In most cases, the 'current network environment' will simply refer to the client's connection bandwidth. However, with Windows Media Technologies' *Intelligent Streaming* and RealNetwork's *SureStream*, the server can change the bit rate of the stream during playback to respond to transient network congestion. Multi-bit-rate streaming starts when the content is created. When the content producer is encoding the source material into either Windows Media or Real formats, they must configure the encoder to embed multiple versions of the media into a single file. When a client requests this file, the proprietary protocols negotiate and select the highest bit-rate stream that the client can effectively receive.

QuickTime's implementation of multiple bit-rates is done differently. Instead of embedding multiple versions of a stream into a single file, QuickTime uses separate files for each bit-rate. QuickTime performs stream selection by publishing a link to a QuickTime master file, instead of the media itself. The master file allows the server to respond to a client request, based on the QuickTime client configuration, with the version that most closely matches the client capability.

The concept of multi-bit-rate files is important when understanding caching behavior. To the cache, each bit-rate version of the media stream is considered a separate object – even if the versions are different rates in the same file on the origin. Therefore, if a client requests the lowest bit-rate available in a multi-bit-rate file, any higher bandwidth versions will not be cached as a result of this request – only the version that is requested is cached. Each version of a file will have to be requested individually before held in cache.

The ABCs of TCP and UDP

All three streaming formats currently prefer to use both the User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP) protocols to deliver streaming media services.

HTTP owes its reliability to the underlying transport mechanism –TCP. Every packet that an HTTP server sends to a client is done so over a dedicated TCP connection between the client and the server. Over TCP connections, every packet that is sent between two nodes is acknowledged and checked for validity. TCP packets are placed in correct order and sent to the browser with the guarantee that what is shown in the browser is exactly what was sent from the server. TCP is called a reliable transport protocol; using TCP over IP, delivery between nodes is guaranteed (given that routing and security policy does not dictate otherwise).

Streaming media formats prefer to use the streamlined but unreliable UDP as the default transport mechanism for delivering the content to the player. Like TCP, UDP is a Layer 4 protocol in the Open Systems Interconnection (OSI) network model. Also like TCP, UDP provides network port addressing. But it is the differences between TCP and UDP that make UDP the preferred choice for streaming media data. UDP does not establish and maintain connections for data transfer, nor does UDP sequence packets before passing them through the OSI layers to the requesting application. Most importantly, UDP does not guarantee delivery of data. When TCP packets are lost between source and destination, TCP delivery will request a retransmission. All of the other packets containing data associated with the missing packet will not be forwarded to the application until all origin packets have been successfully delivered to the client. In the case of HTTP, this manifests itself as the delay before every object is displayed in the client browser. For streaming media, however, losing packets on a TCP stream would have the potential to create annoying delays throughout playback.

But, isn't it important to ensure that all streaming data is delivered to the client, ensuring clear and crisp video? Yes and no. The linear sequence and timing of streaming media make delayed packets worthless; for example, if a packet in a live broadcast is delayed for three seconds, the video sequence may have progressed beyond the point at which the delayed packet may have been displayed. At this point, to retransmit the packet would be a waste of server resources and bandwidth. Therefore, minimizing packet loss and latency are more critical for streaming media than conventional HTTP traffic. But eliminating packet loss with increased latency, which is what the TCP retransmission accomplishes, either wastes bandwidth by resending packets that contain "old" (useless) data or causes delays and pauses in the media playback.

Through buffering and advanced coding/decoding algorithms (codecs) streaming media protocols account for and absorb minor packet loss and latency while still rendering very acceptable playback. Buffering works to counteract the effect of latency. When the server initially responds to a request with a media stream, the client streaming application will delay rendering the data to screen until the player has received several seconds of media. This buffer will allow the player to continue rendering the stream during brief periods of latency without being interrupted. Media players compensate for packet loss during decompression. Most codecs allow the recomposition of a media stream even when some of the media data is lost. Motion prediction and interpolation, as well as other advanced encoding/decoding techniques, can recreate parts of the currently displayed image based on the previous and following images.

In networks where UDP is not an acceptable transport mechanism (most often because of firewall restrictions), streaming media can still be delivered via TCP. However, the load on the streaming server and the bandwidth required to deliver the same data may be substantially higher for TCP-based streams. For a stream to be delivered over TCP, both the client and server must be configured to support the TCP stream.

Another alternative is to have a user download the entire media clip via a TCP connection, via HTTP or FTP, and view the clip as a file stored on their local machine. While this alternative may guarantee the highest playback quality, it does so by requiring the entire clip to download before playback can begin.

Streaming and Proxy Services

A key function of a proxy is the ability to act upon, alter, record, or retain the data that it gathers on behalf of another node. A streaming proxy is no different from any other proxy – it can record the content requests, providing detailed log files. A streaming proxy can retain, or cache, a copy of the content, and then serve that local copy to fill subsequent requests for the same data. And, unique to streaming media, the streaming proxy can split a single live stream into multiple identical streams to serve multiple clients while consuming the bandwidth and resources of only one stream. When deploying a TaskSmart C-Series streaming media cache, consider which services and benefits the TaskSmart server can provide.

Streaming Proxy Services Provided by TaskSmart C-Series Servers

Logging

Much like the logging service of an HTTP proxy, the streaming proxy logging service will record all incoming client requests for streaming media. Each log file type will enable an administrator to gather and analyze data that can lead to configuration or deployment optimizations, such as actively pre-caching or pinning popular content in the streaming cache before the clients request the data.

Caching

Obviously, one of the most compelling streaming proxy services is the ability to cache and vend media streams from a local object store. As with traditional caching, serving the object locally can substantially improve response time as well as provide tremendous bandwidth savings by storing large streaming media files. However, caching as a streaming proxy service can actually deliver content that is more palatable than the same content served across the Internet from the origin server. By reducing or eliminating latency and packet loss in the media stream, the TaskSmart media caching service can deliver content that has fewer visual and auditory artifacts.

Splitting

The TaskSmart C-Series Model 50 server leverages the fact that all clients connected to a live stream should see the same content at roughly the same time to provide the basis for the splitting proxy service. Stream splitting takes a single live stream from the origin media server and sends the same stream to multiple clients. Stream splitting applies only to live streams, and not to on-demand content. Hierarchies of TaskSmart C-Series servers can even split streams within the hierarchy. For example, a parent proxy can split a stream to ten child proxies, each of which may then split the stream to twenty clients. By scaling and splitting live streams through a hierarchy, TaskSmart C-Series servers allow a virtually unlimited number of clients to view a live stream from only a single origin stream. Both scalability and bandwidth savings are addressed by this proxy service.

Client Acceleration

Introduction

The TaskSmart C-Series server appliance improves the following:

- Streaming media quality. Latency and packet loss are eliminated.
- Network efficiency. Redundant connections and downloads from streaming servers are eliminated.
- Streaming media scalability. Distributed streaming media networks are simplified and enabled.

The advantages of caching HTTP and FTP traffic have been employed by some of the largest enterprises and providers for years. The principles that have allowed caching to reshape data distribution for Web servers apply more directly and with greater benefit to streaming media. The caching mantra "*Get the data closer to the clients"* delivered on two promises – improving response time for cached requests and eliminating wasted bandwidth for redundant downloads. However, caching HTTP or FTP traffic did little to improve the content quality. Streaming media with the aid of distributed caches combines reliable quality with speed and efficiency to deliver a third promise – improving the end-product quality.

By using edge media caches to eliminate or reduce the latency and potential for packet loss on the media stream, the rendered image on the client's screen can actually be displayed more clearly, with fewer visual and audible artifacts. Service providers, such as Digital Island, have employed this technology to build worldwide media distribution networks that provide unparalleled streaming quality. Attempting to deliver the same content throughout the same reach from a single point of presence would present an unpredictable and fluctuating experience for the audience.

If network plans do not yet include a globally distributed presence, not only do the benefits of caching still apply to the network, but the benefits of caching streaming media may even be more compounded. Enterprises and service providers have shown that caching can pay for itself in bandwidth savings for HTTP and FTP traffic alone. Consider that research has shown that the average HTTP object is roughly 13 KB, whereas a 30-second clip of 100 Kbps QuickTime streaming video is roughly 550 KB – over 42 times larger than the average HTTP object. The potential for bandwidth savings from streaming media caches dwarfs the already compelling value proposition of caching HTTP traffic. For service providers, imagine the bandwidth savings possible by having currently popular movie trailers and commercials cached at the POP closest to the client. At the same time, the clients on the network to whom the content is delivered will delight with the improved quality of reduced or eliminated packet loss and latency.

Methods of Deployment

Despite the increased complexity of the content, a streaming media cache is not much more complex to deploy than an HTTP cache. Just as with deploying a TaskSmart C-Series server as an HTTP content accelerator, a TaskSmart C-Series server streaming media cache acts as an application-level proxy. That is, the TaskSmart C-Series server receives the request for streaming content from the client and fills the request on behalf of the client. To the origin server, the TaskSmart C-Series server is the client. The TaskSmart C-Series server caches the content in local memory and disk storage while forwarding the content to the original requestor.

Just as one would deploy the HTTP caching proxy, the TaskSmart C-Series streaming cache must be deployed as a:

- Client accelerator as a forward proxy
- Client accelerator as a transparent proxy
- Server accelerator as a reverse proxy

Client accelerators are closer to the client edge of the network; they serve requests from a select group of clients to any and all allowed websites. Reverse proxies are deployed closer to the server edge of the network; reverse proxies service requests from any and all clients to a select website or websites, as configured by the TaskSmart administrator.

The definitions of forward and transparent proxies for streaming media are the same as those for Web traffic. A forward proxy is explicitly configured on the client; the client application is aware of and actively directing requests towards the proxy server. A transparent proxy does not have any configuration stored on the client indicating the presence of a proxy. Instead, active network components – Layer 4 switches and Cisco routers with WCCP 2.0 – will examine the network traffic and redirect any requests for streaming media to the TaskSmart C-Series server.

This release of the *Compaq TaskSmart C-Series Streaming Servers Deployment Guide* contains only the information pertaining to deploying the server as a client-accelerating forward proxy. This document will be updated shortly with the relevant information for client-accelerating transparent proxies and server-accelerating reverse proxies.

Forward Proxy

This section of the guide will provide all of the information that a system administrator or network architect would need to make an effective deployment of a TaskSmart server as a streaming media forward proxy. This section discusses in detail the methods and architecture for deploying a TaskSmart server as a streaming cache forward proxy. This section assumes that either the network is small enough that manually configuring the clients is possible or that an automation tool is present that will allow the proxy configuration to be "pushed" to the clients.

The term "forward" implies that every client streaming application has been configured to use a proxy to access the Internet. Clients will then actively *forward* all requests to the proxy.

Introduction

By default, TaskSmart C-Series Model 50 servers are configured to act as forward proxies for streaming media. When deploying a TaskSmart C-Series media cache, only basic network configuration through the TaskSmart C-Series System Administration Utility is required to enable the full functionality of the streaming cache. Server installation is so simple that most of the configuration must be done on the clients.

Forward proxy deployments are suitable only in environments where there are so few clients that each client can be configured manually or, in networks with hundreds or more clients, where client management software can alter the client configuration across the network. Forward proxy deployments are not suitable for Internet Service Providers (ISPs). There are many reasons why forward proxy is not appropriate for ISP deployments, but most reasons center on the requirement of the client configuration. ISPs should consider transparent proxy deployments. Until this guide is revised to include transparent proxy deployments, consult the *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide*.

Forward proxy is the simplest architecture to understand and implement.

Forward Proxy Request Direction

WMT, Real and QT players can all be configured to use a proxy. The menu path to manually change the proxy for each of the players is shown in Table 2.

Table 2. Streaming Client Proxy Configuration

Windows Media Player 7	RealPlayer 8 Basic	QuickTime
1. Click **Tools**.	1. Click **View**.	1. Click **Edit**.
2. Click **Options**.	2. Click **Preferences**.	2. Click **Preferences**.
3. Click the **Network** tab.	3. Click the **Proxy** tab.	3. Select **Streaming Proxy**.
4. Highlight **MMS:** in the protocol list and click **Configure**.	4. Place check marks and addresses in the PNA and RTSP proxy fields.	4. Place check marks and proxy IP addresses in the appropriate fields.

Larger environments that may already be using client management software, such as System Management Server (SMS) from Microsoft, can automate the configuration of the client player by pushing registry entries to the clients at logon. Some of the common registry keys for Windows Media Player and RealPlayer 8 Basic on Windows 2000 are shown in Table 3 and Table 4. To discover which registry keys may need to be changed on the clients in the network, manually configure a client while running a registry monitor, such as REGMON.EXE. If REGMON.EXE captures too many registry accesses for the user to discern which are necessary to configure the streaming proxy, try opening the REGMON log in a spreadsheet program and find the accesses marked by the "SetValue" command.

Table 3. Windows Media Player 7 – Streaming Proxy Registry Keys

HKCU\SOFTWARE\MICROSOFT\MediaPlayer\Preferences\ProxySettings\MMS\ProxyStyle
HKCU\SOFTWARE\MICROSOFT\MediaPlayer\Preferences\ProxySettings\MMS\ProxyName
HKCU\SOFTWARE\MICROSOFT\MediaPlayer\Preferences\ProxySettings\MMS\ProxyPort
HKCU\SOFTWARE\MICROSOFT\MediaPlayer\Preferences\ProxySettings\MMS\ProxyExclude
HKCU\SOFTWARE\MICROSOFT\MediaPlayer\Preferences\ProxyChanged

Table 4. RealPlayer 8 Basic – Streaming Proxy Registry Keys

HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\PNAProxySupport\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\RTSPProxySupport\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\PNAProxyHost\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\PNAProxyPort\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\RTSPProxyHost\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\RTSPProxyPort\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\HTTPProxyHost\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\HTTPProxyPort\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\NoProxyFor\(Default)
HKCR\Software\RealNetworks\RealPlayer\6.0\Preferences\GetHTTPProxyFromBrowser\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\HTTPProxySupport\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\RTSPProxySupport\(Default)
HKCR\Software\RealNetworks\RealMediaSDK\6.0\Preferences\PNAProxySupport\(Default)

Forward Proxy Operation

When configured to use a proxy, streaming applications behave very similarly to proxied Web browsers. The streaming application no longer assumes any responsibility for filling the request from the origin server; instead, the streaming application simply forwards the raw request to the proxy. The TaskSmart server is monitoring simultaneously on TCP ports 9231, 1091 and 1755 for streaming media requests. Because each streaming format implements control commands and requests differently, there has to be a different daemon for each. Port 9231 is the port that the RealMedia RTSP proxy daemon is listening on; the QuickTime RTSP proxy daemon listens on 1091, and the WMT MMS proxy listens on the default port of 1755.

When the proxy receives the request from the client, the proxy performs a lookup to determine if the requested media is already stored in cache. The proxy will use one of its IP addresses to open a TCP connection to the origin content server for every streaming client. If the object is already in cache – a cache "hit" – the connection will simply be used as an accounting connection. The accounting connection assures that streaming proxies do not break logging and licensing on origin servers. For example, if a RealServer license allows only 50 connections, placing a proxy in front of the server will not mask the true number of clients viewing the content – the license will still allow only 50 connections. If eight clients are accessing streams through the TaskSmart C-Series server, the origin will still see eight occupied connections.

If the object is not already in cache – a cache "miss" – the TCP connection will be used to transport the streaming media from the server to the cache. As previously mentioned, there is considerably more effort involved in streaming over TCP than the preferred UDP. However, because the cache may send out this media object potentially to hundreds of users, the guaranteed delivery of TCP between the cache and the server ensures that the cache does not propagate and repeat transient errors during the initial download.

Again, because each streaming format has different ports for their server processes – 554 for Real, 554 for QT and 1755 for WMT – each origin server will respond on the appropriate port. Real and QT both use modified versions of RTSP as the command and request protocol and, therefore, share the same RTSP port for requests.

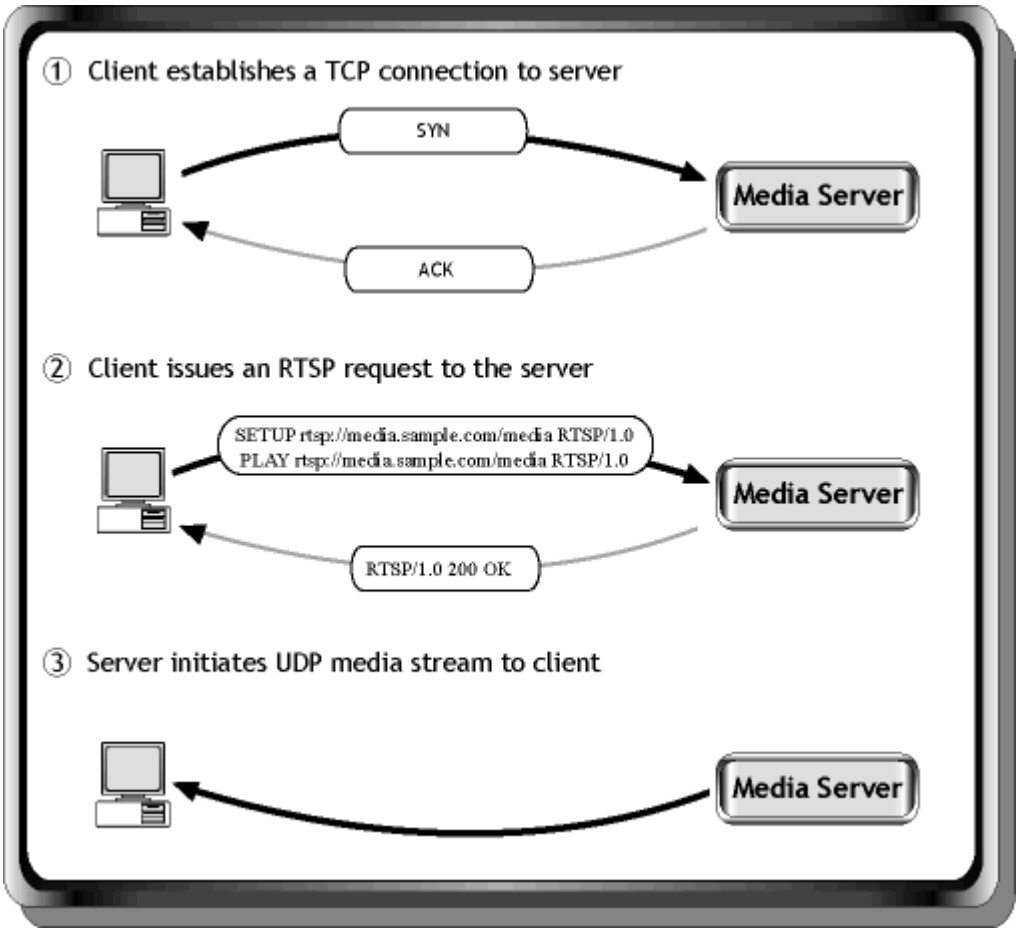

Figure 1: Simplified sequence of an RTSP request

If an object is found in cache or as the proxy fills the request from the origin server, the streaming forward proxy will begin to send the media stream to the client over UDP connections, if available. If UDP data cannot be exchanged between the server and the client, the client can be configured to receive streaming data over TCP. Unless the client is configured to use TCP, all client streaming data will be received over UDP. The exception to this rule is QuickTime; QuickTime player cannot be configured to request a UDP stream. All media data sent between the cache and the server is done so via TCP.

Network Architecture and Placement

The TaskSmart server should be placed logically between the clients and the Internet.

Do not place the streaming cache in the network so that it increases the distance that media objects will travel on their way to the client. Remember that reducing latency and eliminating potential packet loss will have the most dramatic effect on media quality.

The TaskSmart server should be placed as close to the clients as possible.

The most obvious way to reduce latency and potential packet loss is to place the TaskSmart server on the same network segment as the clients. Where possible, this should be the point of choice for streaming media cache deployments. However, cost and performance requirements may demand a less-distributed architecture where the cache may be one or more hops from the clients.

The next most advantageous locations in the network would be access points and aggregation points. Access points are where one network connects to another. The ideal access point is the point at which the local client network meets the primary provider's network. Aggregation points exist wherever large numbers of users converge through a single network node. This node could be a building-wide router that leads to a central office (CO) gateway or it could be at the Digital Subscriber Line Access Multiplexer (DSLAM) or concentrator in a remote POP.

The TaskSmart server should be placed as far from the clients as performance and savings allow.

Placing a TaskSmart C-Series streaming media cache on every desktop would certainly improve network performance and content quality, but it is doubtful that this deployment scenario would be within the budget for a caching architecture.[klm1]

Firewalls and DNS Servers

The most common and most complex consideration for a forward proxy deployment is the media cache's interaction with the local firewall. The first issue to examine is the firewall's policy towards UDP packets. Most firewalls are configured to drop UDP packets. There are two solutions to deploying a TaskSmart C-Series server media cache in an environment that does not allow UDP packets through the firewall:

- Place the TaskSmart C-Series server inside of the firewall (recommended).
- Configure the clients to use TCP as the transport protocol (not recommended).

Placing the TaskSmart C-Series server inside of the firewall allows the clients to continue using UDP as the transport protocol. As mentioned previously, UDP is much more efficient at streaming media delivery than the alternative, TCP. The communication between the TaskSmart C-Series server and the origin media server will always occur over a TCP connection. This requirement is to ensure that the TaskSmart C-Series server does not cache a media object with dropped packets and missing data, thereby propagating the "damaged" file from its cache. At the same time, this simplifies deployment by alleviating the requirement that the firewall be reconfigured to pass UDP packets.

Another reason to place the TaskSmart C-Series server inside of the firewall is to eliminate the persistent and prolonged traffic of redundant streams consuming resources on the firewall. When the TaskSmart C-Series server is inside of the firewall, it establishes only one connection to retrieve a single instance of the stream.

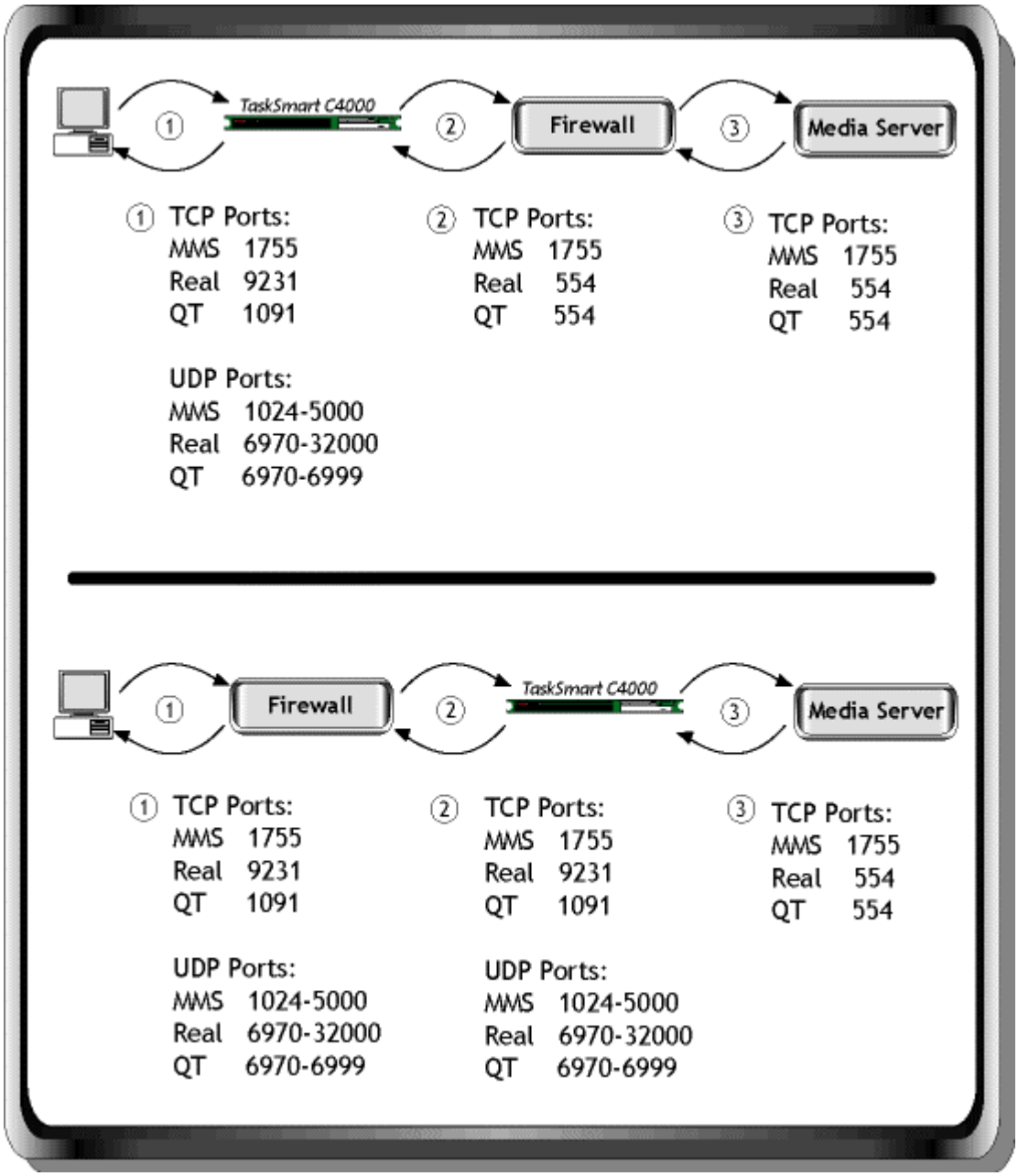

Figure 2: Port utilization with the TaskSmart C-Series server on either side of a firewall

The other solution, configuring clients to use TCP as the transport protocol and placing the TaskSmart C-Series server outside of the firewall, is not recommended. The additional overhead of TCP packet delivery and verification will have a substantially negative impact on any streaming server. At the same time, the firewall must now monitor and maintain the sustained throughput of all the client streams, both cached and uncached. In many cases, this additional load on the firewall may cause delay and degraded network performance for all traffic traversing this taxed firewall.

In either case, there are specific configuration parameters that the firewall will have to obey for the streams to reach the clients. The ports used for the streaming media are shown in Table 5. The data in the table combined with the figures that follow the table should provide administrators with the information necessary to decide which deployment and firewall configuration best meets the security policy.

Table 5. Streaming Formats and Port Usage

Format	Transport Protocol	Port(s) Used	Connection	Description
Windows Media	**TCP/IP**	**1755**	Player-to-cache	MMS requests and responses MMS statistics MMS media data only if the client is configured not to use UDP as the data transport protocol
		1755	Cache-to-server	MMS requests and responses MMS statistics TCP delivery of streaming data to the cache from the origin
		1755	Player-to-server, if there is no proxy	MMS requests and responses MMS statistics MMS media data only if the client is configured not to use UDP as the data transport protocol
	UDP/IP	**1024 – 5000**	Cache-to-player	MMS media data
		1024 – 5000	Server-to-player, if there is no proxy	MMS media data
Real Media	**TCP/IP**	**9231**	Player-to-cache	Real RTSP requests and responses Real Statistics MMS media data only if the client is configured not to use UDP as the data transport protocol
		554	Cache-to-server	Real RTSP requests and responses Real Statistics
		554	Player-to-server, if there is no proxy	Real RTSP requests and responses Real Statistics Real media data only if the client is configured not to use UDP as the data transport protocol
		7802, 7878	Cache-to-server	TCP delivery of streaming data to the cache from the origin
	UDP/IP	**6970 – 32000**	Cache-to-player	Real media data
		6970 – 32000	Server-to-player, if there is no proxy	Real media data

continued

Table 5. Streaming Formats and Port Usage *(continued)*

QuickTime	**TCP/IP**	**1091**	Player-to-cache	QT RTSP requests and responses
		554	Cache-to-server	QT RTSP requests and responses TCP delivery of streaming data to the cache from the origin
		554	Player-to-server, if there is no proxy	QT RTSP requests and responses
	UDP/IP	**6970 - 6999**	Cache-to-player	QT RTP media data
		6970 – 6999	Server-to-player, if there is no proxy	QT RTP media data

While Table 5 provides detailed information that might be used to configure particularly unique scenarios, the following figures illustrate proper configuration for the most common deployments.

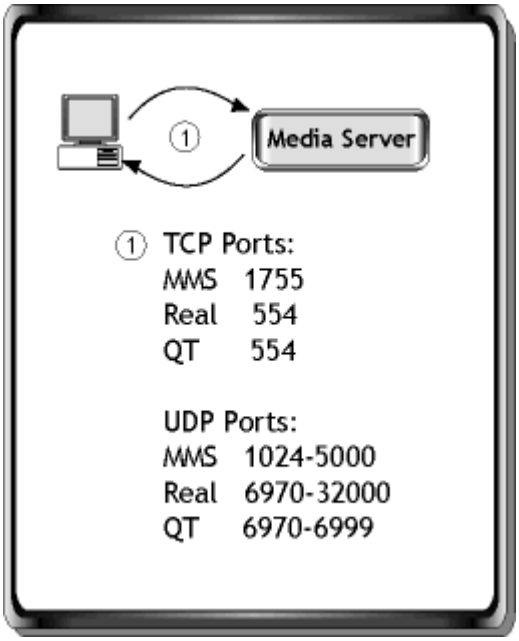

Figure 3: Port utilization without proxy or firewall

Figure 3 shows the ports used by the clients when communicating directly with the origin media server. While there is no TaskSmart C-Series server in this deployment, it may be helpful to understand the un-proxied architecture without a firewall before examining the needs of the network.

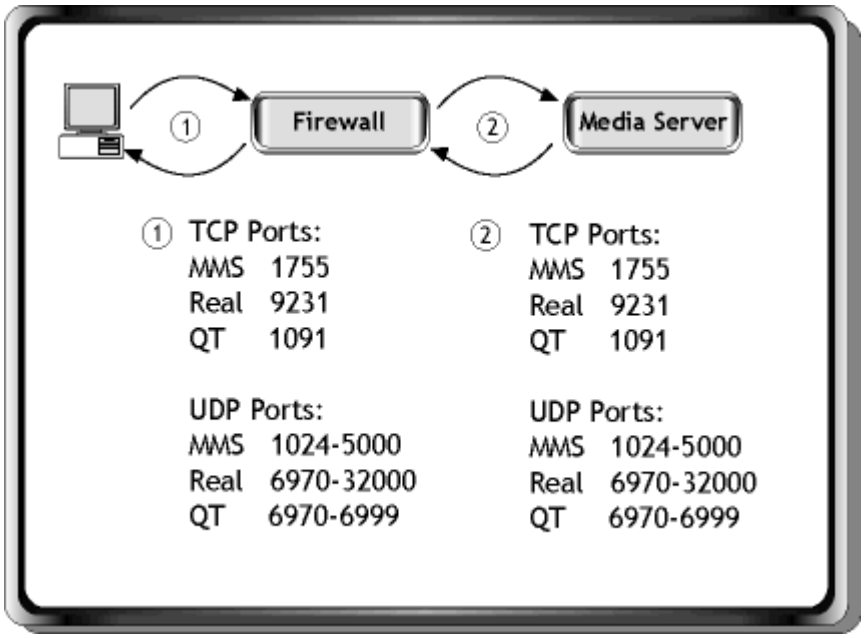

Figure 4: Port utilization with a firewall

In Figure 4, there is no proxy; however, the addition of the firewall will require that either the clients be configured to conform to the firewall policy or the firewall be configured to allow the streaming media applications to communicate on their native ports over their native protocols. If the firewall in Figure 2 does not allow UDP traffic to pass through, the clients must be configured to use TCP as their default transport for streaming media. As mentioned, requiring clients to use TCP as the streaming data transport protocol can have a substantial impact on the performance of the media server and the quality of the presentation viewed by the client. If at all possible, allow clients to retain UDP as the default transport mechanism for streaming data.[GMM2]

When possible, place the TaskSmart C-Series server inside of the firewall. The primary reasons for doing so are:

- The clients can communicate with the TaskSmart server using the preferred UDP without configuring the firewall to allow UDP to all clients.
- The TaskSmart C-Series server will shield the firewall from redundant, sustained streaming traffic if the requested media is in cache.

The firewall must be configured to allow the TaskSmart C-Series server access to origin servers outside of the firewall. For all three streaming formats – WMT, Real and QT – this access takes place over TCP connections. Therefore, the firewall should be configured to allow TCP connections initiated by the TaskSmart C-Series server on ports 1755 (for WMT), 554 (for Real RTSP and QT RTSP), 7802 and 7878 (for Real).

By placing the TaskSmart C-Series server outside of the firewall, either the client will have to be configured to use the less-preferred TCP transport mechanisms or the firewall will have to be configured to pass UDP packets to the clients. Obviously, the preferred solution in this case is to configure the firewall to pass the UDP packets. Thanks to the nature of the application-level proxy, the firewall can be configured to pass UDP packets without severely affecting the security of the firewall. To the firewall, all streaming UDP packets destined for the clients will seem to originate at the TaskSmart server. Therefore, by allowing UDP packets *only* from the TaskSmart IP or MAC address, the firewall can maintain a very tight UDP policy and allow the efficient UDP streaming transport to the clients.

Be wary of this configuration, however. While it may allow the preferred method of transport for streaming media, this configuration has the potential to stress a firewall beyond acceptable limits. If the cache is connected closely to the firewall, for example, via a 100-Mbps/Full-duplex switch, high-cache hit ratios for streaming data could ask the firewall to pass substantially higher traffic than the link the firewall was deployed to serve.

The firewall must still be configured to allow clients to initiate TCP connections to the TaskSmart C-Series server on ports 1755 (for WMT), 9231 (for Real RTSP) and 1091 (for QT RTSP).

If configuring the firewall to pass UDP packets is not acceptable, the client players must be configured to use TCP as the underlying transport mechanism. In addition to the aforementioned difficulty of transmitting streaming media over TCP, access to QuickTime media may be severely constrained. QT does not have a mechanism to move the RTP streaming data over TCP. Instead, QT reverts to serving RTP packets encapsulated in HTTP packets and served on port 80. While this makes the firewall configuration practically a non-issue (it is more likely that the firewall allows port 80 TCP connections), serving QT over HTTP requires the streaming server administrator to have configured the QuickTime server to support HTTP streaming.

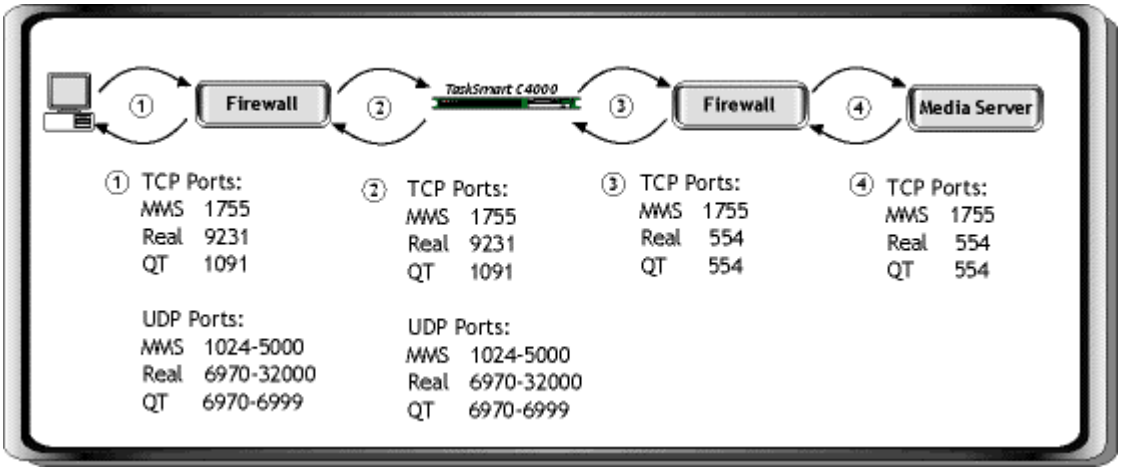

Figure 5: Port utilization with the TaskSmart C-Series server in a DMZ

Figure 5 shows the TaskSmart C-Series server between two firewalls, in what is commonly called a demilitarized zone (DMZ). This is simply a combination of the two previous examples. By following the same logic used to make prior recommendations, the optimal configuration would have the outer firewall, between the TaskSmart server and the Internet, configured to block UDP packets. However, the ports necessary for the TaskSmart server to retrieve the streaming data over TCP would have to be opened – namely 1755 for WMT, 554 for RTSP (Real and QT), 7802 and 7878 for Real. The inner firewall, between the TaskSmart server and the clients, should be configured to allow UDP packets from the TaskSmart server to pass to the clients as well as the bi-directional TCP communication for each of the protocols. For the inner firewall, the TCP communication initiated by the clients would happen on port 1755 for WMT, on port 9231 for Real and on port 1091 for QT.

In the previous cases, no effort was given on constraining the UDP access by port. This is intentional as only Windows Media Player allows administrators to configure which UDP ports are used for streaming data. By default, Windows Media Player will randomly select a UDP port between 1024 and 5000 for streams sent to the client. The client and server (or client and cache) negotiate the UDP port for the streaming data during the client request. Both Real and QT perform the same random UDP port selection during client request. The port ranges, however, are not configurable. RealPlayer 6.0 and higher will select a UDP port between 6970 and 32000 (between 6970 and 6999 for RealPlayer 5.0 and previous), whereas QT will select a port between 6970 and 6999. Because of the difficulty in determining a definitive UDP port scheme, UDP security is restricted at the firewall to the IP or MAC address of the UDP origin – in these cases, the TaskSmart C-Series server.

Forward Proxy Streaming Hierarchies

Streaming hierarchies operate and are configured similar to HTTP proxy hierarchies. For details on deploying TaskSmart C-Series server hierarchies, see the *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide*. The only aspect of streaming hierarchies that is not covered in the *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide* is stream splitting.

Multiple tiers of streaming caches can deliver enormous scalability through stream splitting. When multiple children in a hierarchy request the same live stream from a parent cache, the TaskSmart C-Series server parent is intelligent enough to automatically feed the data stream to all of the children caches – without opening another data channel connection to the parent. Note that the aforementioned control or "accounting connection" will still traverse the hierarchy, thereby reporting all clients that are currently viewing the stream.

From a configuration perspective, only the child caches require any special configuration. For information on configuring a TaskSmart C-Series hierarchy, see the *Compaq TaskSmart C-Series Administrator's Guide*.

If a firewall is placed between a child and parent cache, the child and parent will communicate as though they were cache and origin server. To understand how this will affect port utilization and firewall configuration consult the "Cache to Server" fields in Table 5.

Load Balancing and Load Distribution

Aside from upgrading to the latest and most powerful ProLiant server, the most effective way to scale network performance is to distribute the processing of requests. Generally, the effectiveness of the scaling is related to the effectiveness of the mechanism that distributes the requests. Both load balancing and load distribution enable this performance scaling. Which one is right for a network will depend on performance requirements and, possibly, budget. Load distribution is any passive implementation of load allocation; that is, a load distribution does not monitor and respond to poorly balanced loads between nodes. Load balancing adds this active component through specialized network hardware. Load balancing switches and dedicated load balancing devices direct and monitor the loads on a group of nodes. As a result, load balancers are aware of any uneven distribution of load and can respond by directing any new, incoming requests to the more lightly loaded nodes.

Load Balancing

Scaling performance by deploying more than one TaskSmart C-Series streaming media cache may require an external load-balancing mechanism. In most cases, an L4 switch or dedicated load-balancer should be able to provide fairly linear scalability. But, because the technology is relatively new, make sure that any load-balancing mechanism being considered is capable of handling streaming media traffic.

Load balancing is handled, for the most part, outside of the TaskSmart C-Series server. An intelligent network device monitors the connections and traffic being sent to and from each node in a load-balanced configuration. When a new request reaches the load balancer, the request is sent to the node that is the most lightly loaded. The algorithm that determines which server is the most lightly loaded can vary from manufacturer to manufacturer. Before purchasing a load balancer for streaming media, make certain that the device is capable of handling all required protocols and has been tested with streaming media solutions.

The TaskSmart C-Series server does not require any special configuration to work with a load-balancing device. By simply enabling the streaming proxy services, the caching appliance is ready to respond to the requests forwarded by the load balancer. The clients will still require their media player applications to be configured to use the proxy – in this case, the proxy is usually specified as the address of the load balancer. After the TaskSmart server, the clients, and the load-balancing device have been configured, the number of clients and streams served should remain fairly evenly distributed between the nodes.

Load Distribution

Unlike load balancing, load distribution does not necessarily require an external device to manage the loads between nodes. Instead, load distribution relies on more passive network traffic management to divide the load. Common load distribution mechanisms are DNS “round robin” or manually distributing clients by client configuration. Generally, load balancing is preferred over load distribution, but costs may dictate otherwise. With load distribution, because there is no device to monitor and respond, it may be advisable to monitor the loads on the streaming proxies to determine if they are capable of handling the loads placed upon them. If a particular node is being overwhelmed, this will reintroduce the potential for packet loss and poor quality video or audio.

Using DNS "round robin" may be easier to implement, depending on the number of clients and how their streaming application configuration is managed. DNS "round robin" requires no more configuration than a simple forward proxy; that is, only the client streaming application needs to be configured to use a proxy. In the client configuration, this should be a valid DNS entry, such as "streamer.compaq.com." Most DNS servers are capable of cycling through the multiple entries for streamer.compaq.com. In effect, this will direct some clients to one proxy and other clients to another proxy. DNS "round robin," however, cannot and will not make any effort to balance the loads evenly between the nodes.

Manually configuring the clients simply means that in the configuration of some clients, the streaming application is referred to one node, while other clients are referred explicitly to another node. Generally, this is more difficult to implement than "round robin" when dealing with larger numbers of clients, but can yield similar distribution results.

When trying to decide whether to use load-balancing or load distribution, try to understand the expected loads on the streaming proxies and which load-sharing method meets the performance needs and budgetary restrictions of your network. In most cases, only one of the two methods will meet your criteria; that is, either the cost of the load-balancing device will be acceptable or a load-distribution method should be used.

Clustering and Fault Tolerance

TaskSmart C-Series servers powered by Inktomi can be configured as members of a cluster to extend the functionality and stability of a single system. As members of a cluster, TaskSmart C-Series nodes can be configured collectively, allowing an administrator to treat all of the systems as a single entity. To take full advantage of the clustered capabilities, it is important to understand the clustered services and how they relate to the fault tolerant virtual IP address (VIP) failover.

All TaskSmart C-Series servers can be configured to operate in either management-only cluster mode or full-cluster mode. In full cluster mode, TaskSmart C-Series servers create an amalgam in which the cache object store of one node is available to and accessible by another member of the cluster. In this manner, TaskSmart C-Series servers can scale to provide massive cache storage. However, this aggregated virtual cache does not extend to the streaming media cache. Unlike HTTP objects, streaming media clips may be stored in the cache of every member of a cluster. In other words, a full cluster may share HTTP objects between nodes but they will not share streaming media objects.

If the TaskSmart C-Series server supports streaming media (Model 50), the cluster services will still present a single administration node – this is called *management-only* clustering. Management-only clustering allows TaskSmart C-Series server nodes to share configuration information, allowing an administrator to configure all cluster nodes simultaneously. For more information about management-only clustering, see the *Compaq TaskSmart C-Series Administrator's Guide.*

Clustering, however, does not address fault tolerance without enabling the VIP failover component of the TaskSmart cluster. The VIP failover mechanism allows the streaming proxy service associated with an address to migrate to another cluster node in the event that a node fails. Because the service may take as long as 30 seconds to detect the cluster node failure, migrate the virtual IP address, and start the proxy service on this new node, it is possible that users may experience lost connections during a failover. If this temporary loss of streaming services is unacceptable, any of a number of load-balancing devices are equally suited to the task of detecting a node failure and distributing requests between remaining servers with little or no interruption of service. For more information about VIP failover, see the *Compaq TaskSmart C-Series Administrator's Guide*.

Consider administrative requirements and the impact of as long as 30 seconds of down time as the VIP addresses migrate to the remaining members of the cluster. If the streaming services on the network require a more responsive or nearly instantaneous failover, use of an intelligent network device to monitor the status of the TaskSmart C-Series servers and distribute requests between operational members of the cluster should be considered. Note that if such a device is used, there may be no need to enable the VIP failover on the TaskSmart server itself.

Security and Authentication

A properly deployed TaskSmart C-Series server can increase the security of any network. As a streaming media cache, the primary concern around security is the handling of licensing, content ownership, and digital rights.

All three caching protocols require that the origin media server grant the client the rights to view content, even when served from cache. The simplest form of granting rights is to simply accept the accounting connection from the media proxy server. When a client connects to the TaskSmart server and makes a request for an object already in cache, the TaskSmart server *must* establish a connection with the origin server before the content can be served – this connection is often called an *accounting connection*. This connection is used to ensure the freshness of the content, verify the client's rights to view the content and maintain licensing on the origin server. As mentioned in a previous example, if an origin RealServer has a license that allows the server to deliver 25 streams, any clients that receive the media stream from the cache still consume a connection on the origin server. As a result, the server licensing is enforced. Should a client close its connection to the media proxy, the proxy will close the corresponding accounting connection to the origin server.

In many cases, the authorization to view content requires more than an available accounting connection between the TaskSmart server and the origin media server. Many streams require a user to authenticate their request with a valid user name and password. The TaskSmart C-Series server supports origin media server authentication and Lightweight Directory Access Protocol (LDAP) authentication. However, because of the differences between the streaming protocols, the authentication may cause the media cache to handle the content differently. Because authentication may be critical to a particular deployment, it is important to clarify how the TaskSmart server will handle different types of authenticated media streams.

LDAP authentication is supported only with Real media. However, using LDAP as the authentication protocol for streaming Real data has the drawback of requiring TCP as the underlying data transport protocol. Where possible with RealMedia, use the origin server to authenticate users through the proxy. This allows for the usage of the preferred transport protocol, UDP, and allows the data to be cached by the TaskSmart server. Subsequent requests for the cached data will still require the authentication credentials, as the origin server will still require the credentials to establish the accounting connection before the data can be served.

For Windows Media Technologies, any authentication challenge issued by the origin server in response to a request is forwarded to the client. The client is then prompted to enter a user name and password; the TaskSmart server will forward the client authentication response (the user name and password) to the origin server. If the origin server authenticates the user, the TaskSmart C-Series server will simply pass the traffic between the client and origin server without writing the information to cache. In short, WMT streams that require authentication are not cached.

For more information about HTTP security and the TaskSmart C-Series server, see the *Compaq TaskSmart C-Series Administrator's Guide* and the *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide.*

Conclusion

Ideally, by understanding the protocols and their operation, the administrator will have a greater understanding of how to deploy a TaskSmart C-Series server as a streaming media cache. There are endless possibilities for the placement and configuration for the streaming media proxy. The information contained herein details the most common forward proxy considerations, allowing an administrator to analyze the network and determine the best solution.

This document is intended to complement the information in the *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide* by providing the additional concerns and details that must be considered when deploying a TaskSmart C-Series as a streaming media cache. For more details on deploying HTTP caches, refer to *Compaq TaskSmart C-Series Web Content Acceleration Servers Deployment Guide* and the *Compaq TaskSmart C-Series Administrator's Guide*.

Shortly, this document will be updated to cover the other deployment scenarios – client acceleration as a transparent proxy and server acceleration as a reverse proxy. Until that time, details on the configuration and operation of transparent and reverse proxies can be found in the *Compaq TaskSmart C-Series Administrator's Guide.*

Page: 13
[klm1]We need to finish this sentence. What is the train of thought?
Page: 17
[GMM2] If that's the case, what happens when the cache attempts to do its TCP cache fill of the origin stream?