

# Network Load Balancing using Streaming Servers for VoD Applications

## **Design of Clustered Streaming Servers to Balance the Network Load for VoD Applications**

Ms. S.V.Kolekar\*, Mr. D.M.Thakore\*\*

\*Student of M.Tech (IT), Bharati Vidyapeeth Deemed University College of Engineering, Pune-411043, Maharashtra, India. Email: [suchu\\_it@yahoo.com](mailto:suchu_it@yahoo.com)

\*\*A.P., Computer and Information Technology Department, Bharati Vidyapeeth Deemed University College of Engineering, Pune-411043 Maharashtra, India.  
Email: [deventhakur@yahoo.com](mailto:deventhakur@yahoo.com)

### ***Abstract***

*Ideally, video and audio are streamed across the Internet from the server to the client in response to a client request for a Web page containing embedded videos. The client plays the incoming multimedia stream in real time as the data is received. Quite a few video streamers are starting to appear and many pseudo-streaming technologies and other potential solutions are also in the pipeline. Generally streaming video solutions may work on a closed-loop intranet, but for mass-market Internet use, they're simply dysfunctional. However current transport protocol, codec and scalability research will eventually make video on the Web a practical reality. The basic goal of this paper is the same of developing and stream video and audio servers which will provides its client media data on a real time basis and balance the load in the cluster of servers. Video transmission requires relatively high bandwidth with strict Quality of Service (QoS) properties (e.g., guaranteed bandwidth, bounded jitter and delays). Therefore, as any application involving video transmission, our service is best provided using QoS reservation mechanisms. However, if bandwidth is abundant and jitter rarely occurs, e.g., in a relatively not loaded LAN or small scale WAN, and then some buffer space and a flow control mechanism can account for jitter period.*

**Keywords:** Media Streaming, Content Delivery Network, Clustering, Load Balancing, Mirroring, Protocol Rollover.

### **1. Introduction:**

Applications such as news on demand, distance learning, e-commerce, and scientific visualization all store, maintain, and retrieve large volumes of real-time data over a network. These data are denoted collectively as *continuous media*, or CM. Video and audio objects are popular examples; *haptic* and *avatar* data are less familiar types. CM data require a streaming architecture that can, first, manage real-time delivery constraints. Failure to meet these constraints on CM data disrupts the display with "hiccups." Second, the architecture must address the large size of CM objects. A two-hour MPEG-2 video with a bandwidth requirement of 4 megabits per second is 3.6 gigabytes in size. The currently available commercial implementations of CM servers fall into two broad categories:

- ✓ Low-cost, single-node, consumer-oriented systems serving a limited number of users; and
- ✓ Multinode, carrier-class systems such as high-end broadcasting and dedicated video-on-demand systems.

RealNetworks, Apple Computer, and Microsoft product offerings fit into the consumer-oriented category, while SeaChange and nCube offer solutions oriented toward carrier-class systems. While commercial systems ordinarily use proprietary technology and algorithms, making it difficult to compare their products with research prototypes, we have designed and developed a second-generation CM server that demonstrates several advanced concepts.

The research thing of this paper distinguishes itself from other similar research efforts in the following:

- ✓ Complete distribution with all nodes running identical software and no single points of failure;
- ✓ Efficient online scalability allowing disks to be added or removed without interrupting CM streams;
- ✓ Synchronization of several streams of audio, video, or both within one frame (1/30 second);
- ✓ Independence from media types;
- ✓ Compliance with industry standards;
- ✓ Selective retransmission protocol; and
- ✓ Multithreshold buffering flow-control mechanism to support variable bit-rate (VBR) media.

This paper is also providing a complete end-to-end system that uses an IP network with several supportable client types and cluster of servers.

## 2. Methods and Material:

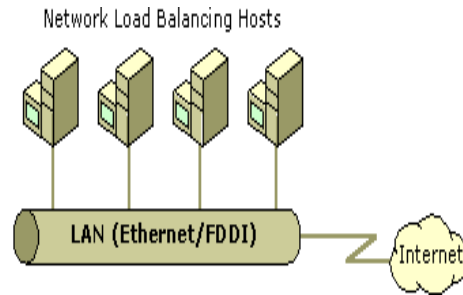
This paper discussed the solution which is a scalable, real-time streaming architecture that enables applications such as video-on-demand and distance learning on a large scale. While this incorporates lessons learned from first generation research prototypes, it also complies with industry standards in content format (MPEG-4) and communication protocols (RTP/RTSP). The given solution improves upon both research and commercial approaches by using a *bipartite* design and alternative approaches to handling variable-bit-rate (VBR) video. We also integrated a selective retransmission protocol into RTP server to recover from packet loss. Lastly this is an operational a

### 2.1 Performance and Reliability: Ensuring the Availability of Applications and Services

Network Load Balancing clusters a group of computers together that run server and supports a variety of display bandwidths. r programs using the TCP/IP networking protocol. Network Load Balancing service enhances the availability and scalability of Web servers, File Transfer Protocol (FTP) servers, streaming media servers, virtual private network (VPN) servers, and other mission-critical programs.

Network Load Balancing provides these enhancements using a cluster of two or more host computers (servers that are members of the cluster) working together.

A single computer running Windows 2000 Advanced Server can provide a limited level of server reliability and scalable performance. However, by combining the resources of two or more computers running Windows 2000 Advanced Server into a single cluster, Network Load Balancing can deliver the availability that Web servers and other mission-critical programs need to maintain top performance. Figure represents a Network Load Balancing cluster containing four hosts.



**Fig.1 Hosts within a Network Load Balancing Cluster**

Each host runs separate copies of the desired server programs, such as Web server, FTP, Telnet, and messaging. For some services, such as a Web server, a copy of the program runs on all hosts within the cluster, and Network Load Balancing distributes the workload among the servers. For other services, such as messaging, only one copy of the service handles the workload within the cluster. Instead of equalizing the loads of these services, Network Load Balancing allows network traffic to flow to a single host, moving the traffic to another host only in cases of server failure. Network Load Balancing allows all computers in the cluster to be addressed by the same set of cluster Internet Protocol (IP) addresses — while also maintaining their existing, dedicated IPs. Network Load Balancing distributes incoming client requests as TCP/IP traffic, including TCP connections and UDP streams, across the hosts.

To scale server performance, Network Load Balancing balances the load of incoming TCP/IP connections across all hosts in the cluster. You can configure the load size for each host as necessary. You can also add hosts to the cluster dynamically to manage increased load. In addition, Network Load Balancing can direct all TCP/User Datagram Protocol (UDP) traffic (not configured to be load balanced) to a designated single host, called the "default host." This is advantageous because it allows all services not explicitly configured for load balancing to run on a single host. Network Load Balancing manages the TCP/IP traffic to maintain high availability for server programs.

When a host fails or goes offline, Network Load Balancing automatically reconfigures the cluster to redirect client requests to the remaining computers. For load-balanced programs, the load is automatically redistributed among the computers that are still operating. Programs running on a single server have their traffic redirected to a specific host. Connections to the failed or offline server are lost. After necessary maintenance is completed, the offline computer can transparently rejoin the cluster and regain its share of the workload.

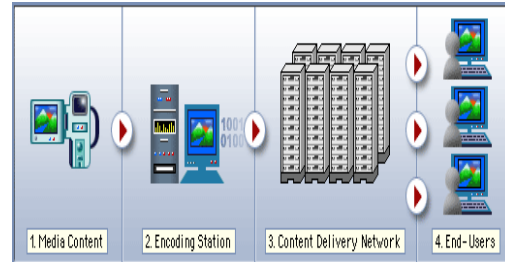
Network Load Balancing does not detect application failures. Instead, it is designed to be controlled by application monitoring programs that check for and ensure correct behavior of their associated applications. For example, if an application monitor determines that its service has failed, it can instruct Network Load Balancing to remove the affected host from the cluster until the problem is corrected. Additionally, Network Load Balancing detects whether a cluster host has had an orderly or disorderly shutdown or if the network adapter has failed.

You know that your organization requires Network Load Balancing if you host a TCP/IP service (such as a Web server) that must scale its performance to meet increasing client demand and that must be continuously available.

## **2.2 Content Delivery Network**

Using Content Delivery Network for delivering video on-demand is easy. Capture your media content then encode it into the format of choice. Upload your digitized file into our Content Delivery Network and your content is replicated to the edge of the network in real-time. If time or experience is an issue,

PlayStream can capture and encode your media for you in any format we deliver at a competitive rate. After uploading your content to PlayStream's Content Delivery Network, your content is available to embed or link-to within your website, allowing complete flexibility over how you integrate media into your web presence.



**Fig. 2 Content Delivery Network**

### **2.3 Uploading Your Media To Our Content Delivery Network**

After your content is converted to a streaming format you must upload it to our servers. We offer two ways to do this, either through the content management area when you are logged in to your PlayStream account or via FTP at <ftp.playstream.com> with your user name and password. Links to your content are generated via our globally load balanced easy link system and you put these links into the HTML of your Web page. When someone visits your Web site and clicks a link for streaming audio or video, that request is directed to the data center closest to them and they stream your content from our servers.

By directing your audience member to the closest data center, we cut down on the time it takes to deliver your stream to them. Once the closest data center has been established, the streaming request is sent down one of the many backbones we are connected to via Internet's advanced "Synchronous-When-Optimal" routing service.

The system delivers your media by avoiding the often congested public and private peering points of the Internet, thus greatly increasing delivery performance and reducing problems such as rebuffering.

### **2.4 Delivering Your Media To Your Audience**

When a streaming media Player makes a connection to one of our streaming servers via our easy link technology or a reference file (RAM for RealMedia, ASX; WVX or WAX for Windows Media; or QTL for QuickTime), the server sends the player data via User Datagram Protocol (UDP) by default. If the streaming media player cannot accept this data, the server tries resending it via the TCP protocol to several different ports, with port 80 as its final attempt if each attempt fails. This process is called protocol rollover. If this negotiation is successful, and the server is either Windows Media or the RealServer, a check is made for the connection speed configured in the connection settings of the streaming media player. Based on the connection speed returned to the server from the player, the server will send the highest play supported stream from a multiple-bit-rate streaming file, the whole file in the case of a single-bit-rate streaming file, or just the audio portion of an audio/video streaming file if the player connection speed is not fast enough to accept all video data.

### 3. System Architecture:

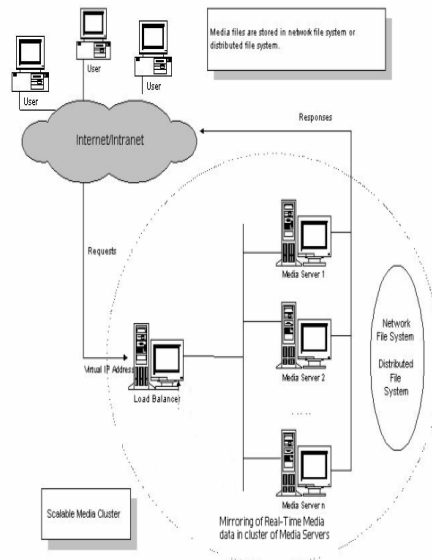


Fig. 3 System Architecture of N/W Load balancing Using Streaming Media Server for VOD

- **Cluster of VODServers for Real Time Media delivery:** This is main feature of paper, which offers highest cost / performance, bandwidth availability, manage load by distributing the data and finally manages the scalability.
- **Streaming of Real-Time Media at Server Side:** Content delivery networks (CDN) represent a server-side solution to streaming that is proactive in nature. If time or experience is an issue, PlayStream can capture and encode your media for you in any format we deliver at a competitive rate. After uploading your content to PlayStream's Content Delivery Network, your content is available to embed or link-to within your website, allowing complete flexibility over how you integrate media into your web presence.
- **Supported Formats:** ASF, AVI, MPEG-I, MPEG-2 PS, MPEG-2 TS, MP3, HDTV, CODEC Independent distribution and delivery supports multiple media formats concurrently for mixed media deployments. Project can try to support above formats of Real-Time data at the time of Streaming and Content Delivery.
- **Load Balancing using Round-Robin Algorithm in the Cluster:** To balance the load of cluster system can use Round-Robin Method of load balancing in cluster because this method is easy to implement and effective where all the nodes in the cluster are identical in capacity and performance.
- **Managed Delivery of Media to Clients by considering QOS factors:** Media Server can send data via UDP or TCP protocol. In this project implements protocol rollover concept by changing the protocol from UDP to TCP or vice-versa. Based on connection speed returned to the server from the player the server will send the highest play supported stream from a multiple-bit-rate streaming file, the whole file in the case of a single-bit-rate streaming file, or just the audio portion of an audio/video streaming file if the player connection speed is not fast enough to accept all video data.
- **Splitter and Mirroring:** Splitter splits real-time data on cluster of servers in efficient manner so that accessibility of data manages properly. Software architecture for continuously mirroring streaming data received by one node of a cluster-based server to other cluster nodes. The intent is

to distribute the load on the server generated by the data's processing and distribution to many clients. This is particularly important when the server not only processes streaming data, but also performs additional processing tasks that heavily depend on current application state. This feature is important if any one server goes fail in the cluster then mirroring of data of another server can easily handle the client's request.

#### 4. Implementation:

This design is implemented using Java™ for all of the components. The VoD client is based on the **Java™ Media Framework (JMF)** a package that supports the replay of audio and video streams (the system utilizes Sun's JMF implementation). JMF supports a wide range of video and audio formats, including MPEG. The VoD client consists of two main components, a player and a data source. The player is responsible for replaying the audio and/or video stream. The data source is responsible for retrieving the data. Data sources exist that retrieve data from disk, or retrieve data over the network using a variety of protocols. The main protocols require for this implementation are (Real-Time Transport Protocol) RTP and RTSP (Real-Time Transport Streaming Protocol).

RTP provides end-to-end network delivery services for the transmission of real-time data. RTP is network and transport-protocol independent, though it is often used over UDP. While RTP does not provide any mechanism to ensure timely delivery or provide other quality of service guarantees, it is augmented by a control protocol (RTCP) that enables you to monitor the quality of the data distribution. RTCP also provides control and identification mechanisms for RTP transmissions. RTSP provides Streaming features for Real-Time Data so that when media content is streamed to a client in real-time, the client can begin to play the stream without having to wait for the complete stream to download.

#### 5. Results and Discussion:

We have conducted several end-to-end video playbacks to test the key design features of the VoD system: fault-tolerance and performance. This section presents results of some experiments that we carried out over our system implementation. Fig. 4 demonstrates the results of an experiment that measures the downloading rate of a video file delivered to a client machine using different number of data servers.

Three servers have been used in this experiment. We measured the rate to download the whole file from each server, and then we distributed the same file across two and three servers. We measured the rate taken to download and merge the blocks in parallel from the servers. Fig 4 shows the advantage of our approach of utilizing multiple concurrent threads for retrieving video blocks.

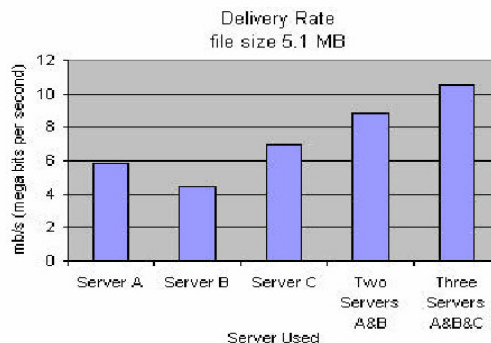


Fig. 4 Download rate using multiple Servers

**Visual Results:** The visual quality of an MPEG-1 encoded movie is surprisingly excellent with and without failed servers. Our test movie, encoded at 640x480 resolutions and about 1.2 Mb/s, displayed perfectly without any jitter. Due to parallel data read and prefetching, the visual user experience and the quality of played video was clearly superior to that provided by the traditional single server approach.

## 6. Conclusion:

Media streaming is a complex process involving client and server that are connected by the Internet in an end-to-end system. Both the client and the server can view the Internet as a black box without knowing what is happening in the opposite end. In the client's view, the problems with streaming are constantly changing conditions such as bandwidth fluctuation and packet loss. The server sees other problems, such as server overload and aggregate demand for bandwidth throughput.

The main features of this paper are:

- Network Load Balancing using Clustering of Streaming Servers.
- Content Delivery Network (CDN).
- Increasing delivery performance and reducing problems such as rebuffering.
- Protocol rollover.
- Multi Bit Rate streaming file transmission.
- Mirroring of data on Video Servers of Cluster.
- Dynamic insertion of video server.

A CDN helps improve QoS and server-side scalability for streaming, a CDN cannot completely solve the problems that we reviewed in Section 2. This is because any CDN has a limited scale and reach. For a client to benefit from a CDN, a surrogate server must be present in a location closer to the client than the origin server. Unfortunately, this cannot be guaranteed for every client. This situation will be improved with the recent development in CDN peering that aims at extending the scale and reach of individual CDNs by interconnecting them using standard protocols.

## 7. References:

- [1] Y.C. Tay, H.H. Pang, "**Load sharing in distributed multimedia-on-demand systems**", IEEE Trans. on Knowledge and Data Engineering, vol. 12, no. 3, pp. 410-428, June 2000.
- [2] Mohamed M. Hefeeda and Bharat K. Bhargava CERIAS and Department of Computer Sciences Purdue University West Lafayette, IN 47907. "**On-Demand Media Streaming Over the Internet**"
- [3] Aidong Zhang, Yuqing Song, Markus Mielke. "**NetMedia: Streaming Multimedia Presentations in Distributed Environments**," *IEEE MultiMedia*, vol. 09, no. 1, pp. 56-73, January-March 2002.
- [4] D. L. Eager, M. K. Vernon, and J. Zahorjan, "**Minimizing Bandwidth Requirements for On-Demand Data Delivery**", *Proc. MIS'99*, Indian Wells, CA, Oct.1999.
- [5] "**Design & implementation of a low-cost clustered video Server using a network of personal computers**" *Apostolos Papagiannis Dimitrios Lioupis Stylianos Egglezos* Department of Computer Engineering & Informatics University of Patras, 265 00 Rio, Patras, Greece &Computer Technology Institute, 61 Riga Feraiou St., Patras, Greece

[6] **"Distributing Streaming Media Content Using Cooperative Networking"** *IEEE Multimedia* by Venkata N. Padmanabhan Helen J. Wang Philip A. Chou *Microsoft Research* Kunwadee Sripanidkulchai *Carnegie Mellon University* April 2002

[7] T. P. Nguyen and A. Zakhor. **"Distributed Video Streaming over the Internet"**, *Multimedia Computing and Networking (MMCN)*, January 2002.

[8] **"Reactive and Proactive Approaches to Media Streaming: From Scalable Coding to Content Delivery Networks"** by Jian Lu EnjoyWeb, Inc. *Coding and Computing*, April 2001

[9] **"Architecture of a modular streaming media server for content delivery networks"** *Sumit Roy, John Ankcorn, and Susie Wee* Streaming Media Systems Group Mobile and Media Systems Laboratory Hewlett-Packard Laboratories, Palo Alto

[10] **"Java™ Media Framework"** Home Page, <http://java.sun.com/products/java-media/Jmf/index.html>