산업기술연구(강원대학교 산업기술연구소 논문집), 제21권 A호, 2001.
Journal of Industrial Technology, Kangwon Nat'l Univ., Korea, No. 21 A, 2001.

리눅스 클러스터 서버 상에서 RTSP 기반의 VOD 스트리밍 시스템의 설계 및 구현

Design and Implementation of RTSP Based VOD Streaming System On a Linux Cluster Server

홍 기 호* 김 영 진* 최 황 규**
Hong, Ki-Ho Kim, Young-Jin Choi, Hwang-Kyu

Abstract

Linux cluster Server makes a high performance PC cluster system, and provides scalablity and availability as compared with an expensive single server. In this paper, we design and implement a VOD streaming service system to serve video and audio streaming services on the Linux cluster server. Our Streaming server supports the MS Window Media Format based on the RTSP streaming protocol.

키워드 : *윈도우미디어포맷, 리눅스클러스터, 스트리밍서비스, RTSP, VOD*
Keywords : *Windows Media Format, Linux cluster, Streaming service, RTSP, VOD*

1. 서론

최근 통신 기술의 급속한 발전으로 통신망 서비스 사업이 다각화됨에 따라 초고속 통신망의 보급이 가정으로까지 빠르게 확대되고 있다. 이에 따라 인터넷을 기반으로 하는 웹 서비스 업체들이 기하급수적으로 증가하고 있고, 인터넷을 기반으로 하는 서비스도 다양하게 변화하고 있으며, 이를 이용하려는 서비스 가입자들도 폭발적으로 증가하고 있는 추세이다. 인터넷 서비스 제공 업체들이 많은 가입자들에게 동시에 안정된 서비스를 제공하려면 비교적 고가의 고성능의 대용량 서버를 설치 운영하여야 한다.

비교적 저가의 고성능 PC 또는 워크스테이션을 빠른 네트워크로 연결한 클러스터 시스템은 강 결합의 단일 서버와 비교하여 확장성과 신뢰성이 높고, 가격 대 성능 면에서 우수한 것으로 나타났다. 이중 리눅스 기반의 PC 클러스터는 과학 계산용의 슈퍼컴퓨팅 분야에 많이 활용될 뿐만 아니라 웹 서버, 게임 서버, VOD 서버 등 고부하와 고 가용성 응용분야에 활용하고자 하는 많은 연구가 이루어지고 있다[1,2,3].

한편, 비디오/오디오 스트리밍 서비스는 최근 들어 여러 분야에서 폭 넓게 사용이 증가하고 있으며, 다양한 상용의 제품들이 스트리밍 서비스에 이용되고 있다. 그러나 대부분의 상용 스트리밍 서버는 특정한 미디어 타입만을 지원하며, 서비스 비용이 높다는 단점을 갖는다.

따라서 본 논문에서는 최근 활발히 진행중인 리눅스 기반의 PC 클러스터를 이용한 확장성과 가격 대 성능비가 우수한 스트리밍 서버를 설계·구현한다.

* 강원대학교 컴퓨터정보통신공학과 석사과정
** 강원대학교 전기전자정보통신공학부 교수, 공학박사

본 논문은 2장에서는 리눅스 가상 서버 기반의 부하 분산 서버의 설계에 대하여 알아보고, 3장에서는 스트리밍 서비스의 종류와 폭 넓게 쓰이고 있는 상용의 스트리밍 서버에 대해서 기술한다. 4장에서는 클러스터형 스트리밍 서비스에 대한 부하 분산 방법과 서버의 설계 및 구현에 대한 결과를 보인다.

2. 리눅스 클러스터 구성

리눅스 가상 서버[5]는 리눅스 운영체제를 기반으로 고속 네트워크에 연결된 서버들의 클러스터로 성능이 뛰어나고 가용성이 높은 서버를 쉽게 구축할 수 있다[1].

2.1 일반적인 클러스터 시스템 구조

일반적인 가상 서버의 구조로 하나의 부하 분산기와 여러 대의 실제 서버를 가진다(그림 1). 실제

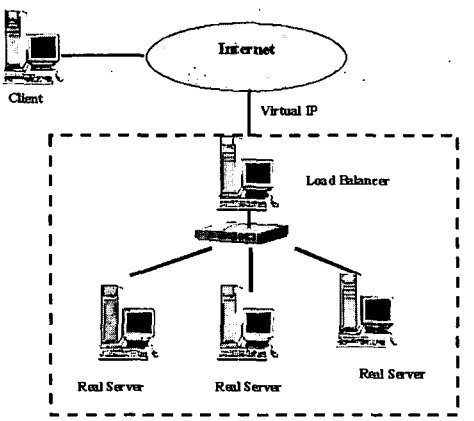

그림 1. 일반적인 클러스터 시스템 구조

서버들은 동일한 서비스를 운영하도록 구성되며, 서비스 내용은 각 서버의 지역 디스크에 복제되거나, 분산 파일 시스템으로 공유됨으로써 제공된다. 부하 분산 서버는 실제 서버의 서비스 내용에 대하여 클라이언트의 요청이 있을 때마다, 적절한 스케줄링 방법(예, Round Robin)에 의해서 상호 연결되어 있는 실제 서버에 클라이언트의 요청을 발송한다. 점선의 직사각형으로 표시된 클러스터의 실제 서버들은 빠른 LAN/WAN으로 연결되며, 클라이언트에게는 하나의 단일 이미지의 시스템으로 보이도록 투명하게 설정된다. 즉, 실제 서버들의 외부 연결부로서의 부하 분산 서버는 클라이언트의 요구들을 각각 다른 노드에게 할당하고 클러스터의 병렬 서비스를 하나의 IP 주소상의 가상 서비스로 보이도록 한다. 높은 가용성은 노드 또는 데몬의 장애나 시스템의 구조 변화를 적절히 감지하여, 노드를 제거하거나 대치하여 이루어진다.

리눅스 가상 서버의 클라이언트 요구 처리 흐름에 따라 부하 분산 방식은 크게 두 가지 유형을 보인다.

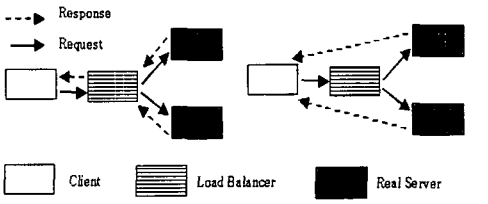

그림 2 요구 처리 흐름에 따른 부하 분산 유형

첫 번째 유형은 클러스터 노드에서의 클라이언트 요구 처리를 중앙의 부하 분산 서버를 통해 클라이언트에 전달하는 방법이다. 단순하면서 클러스터 노드들의 수정 없이 클라이언트에게 클러스터 시스템의 투명성을 제공한다. 하지만, 중앙 집중적인 방식으로 부하 분산기가 클라이언트 요구를 클러스터 노드에 할당하는 부하 분산과 노드의 요구 처리 결과를 다시 클라이언트에게 전달해야 하므로, 부하 분산기의 오버 헤드가 증가하여 쉽게 병목구간이 될 수 있는 단점을 가진다. 두 번째 유형은 요구의 중앙 집중적인 처리로 인한 확장성의 저하를 해결하기 위한 방법으로 부하 분산 서버는 단순히 요구를 클러스터 내 노드에게 할당하는 임무를 가지며, 실제 각 노드에서 요구를 처리하여 클라이언트에게 전송한다. 요구 처리 결과를 직접 클라이언트에게 전송함으로써 부하 분산기의 오버 헤드가 크게 감소하여 확장성을 크게 향상시킬 수 있다.

2.2 리눅스 클러스터

지금까지 리눅스 가상 서버는 세 가지 방법의 IP 계층 부하 분산 방식으로 클러스터의 병렬 서비스를 단일 IP 주소 상에 하나의 가상 서비스로 보이도록 개발되었다. 이때 가상 서버 구성 방법은 크게 세 가지로 1)네트워크 주소 번역에 의한 가상 서버, 2)IP 터널링에 의한 가상 서버, 3)다이렉트 라우팅에 의한 가상 서버 등이 있다.[1]

네트워크 주소 번역에 의한 가상 서버는 부하 분산기와 실제 서버들이 스위치나 허브로 연결되어 있으며, 실제 서버들은 사적 인터넷 주소체제를 사용한다. 다만 서비스를 위한 가상 IP 만이 외부와 연결된 부하 분산 서버에 사용된다. 따라서, 클라이언트로 하여금 하나의 IP 주소 위에 하나의 가상 서비스로 보이도록 할 수 있는 서버를 구축할 수 있게 된다.

네트워크 주소 번역에 의한 가상 서버가 실제 서버들이 외부 인터넷에서 사용할 수 없는 사적

인터넷 주소 체제를 사용할 수 있다는 장점을 가지지만, 부하 분산 서버가 클라이언트의 요청 패킷과 실제 서버들의 응답 패킷 모두를 처리해야 하므로 부하 분산 서버 자체가 병목구간이 될 수 있다. 이런 단점을 해결할 수 있는 방법으로 IP 터널링과 다이렉트 라우팅 방식을 적용할 수 있다.

IP 터널링은 요청 패킷을 IP화된 정보안에 감싸 넣어 실제 서버로 전송하고, 실제 서버는 패킷을 풀어 요청을 처리하여, 결과를 직접 요청한 클라이언트에게 돌려준다.

다이렉트 라우팅은 실제 서버와 부하 분산기가 하나의 서비스를 위한 가상 IP를 공유한다. 부하 분산 서버는 가상 IP로 설정된 하나의 인터페이스를 가지며 요청 패킷을 스케줄링 방법에 의해 선택된 실제 서버에 직접 라우팅 한다. 다이렉트 라우팅 방식은 본 연구에서 VOD 스트리밍 서비스를 위한 가상 서버 구현에 사용되었다.

3. 스트리밍 서버 구성

현재 통신망 상에서 비디오와 오디오를 실시간으로 전송하기 위하여 사용하고 있는 멀티미디어 서비스 시스템들은 아래와 같이 크게 두 부류로 나눌 수 있다.

① 인터넷 기반 스트리밍 서비스

웹 상에서의 멀티미디어 서비스를 위해서 독자적인 압축 방법을 주로 사용하여 낮은 통신 속도에도 신축적으로 대응한다. 대표적인 예는 RealSystem, Streamworks, WMT (Window media Technologies), Vivo, VDO Live 등이 사용되며, Real Video의 경우 웹 서비스에서 거의 표준적으로 사용되고 있다.

② 인트라넷/랜 기반 스트리밍 서비스

원격교육과 같은 응용에서 높은 통신 속도를 기반으로 주로 MPEG I,II 등의 표준화된 압축방식을 사용하여 고화질 서비스가 가능하다. 현재까지는 광범위하게 사용되지는 않으나 국내/외에 개발된 다수의 제품이 있다.

또한, 멀티미디어 서비스는 클라이언트와 서버간의 대화 형태에 따라 다음과 같이 구분할 수 있다.

1) **VOD(Video On Demand) 방식** : 저장된 비디오 및 오디오 정보의 스트리밍 전송

2) **Live Broadcasting 방식** : 실시간 압축 및 실시간 전송

3) **Video Conference System** : 양방향 실시간 비디오/오디오 전송

본 논문에서는 인터넷을 기반으로 하는 스트리밍 서비스 시스템을 구현한다. 따라서 인터넷 기반의 상용 스트리밍 서버를 소개하고 비교한다. 현재 가장 많이 쓰이고 있는 스트리밍 서비스는 RealNetworks의 RealSystem, Microsoft의 WMT, Darwin Streaming Server를 들 수 있다.

① RealNetworks의 RealSystem

세계 최초의 스트리밍 솔루션이며 한때 전세계 시장 점유율 70%이상이었다. 또 최근 버전에서는 브로드캐스팅을 기능을 지원 웹 브라우저 없이도 디지탈 인터넷 방송, 웹 프리젠테이션, 애니메이션, 스트리밍 비디오, 스트리밍 오디오를 모두 감상할 수 있다. 그러나 고가의 서버프로그램을 구입해야 하고 동시 접속자수가 제한된다는 단점이 있다.

② Microsoft의 Windows Media Technologies

Microsoft의 스트리밍 솔루션이며 최근들어 Real Network사의 RealSystem의 시장 점유율을 앞지르고 있다. 폭 넓은 스트림 대역폭, 여러 가지 플레이 백 미디어 포맷 지원하며 비디오 재생시, 화면의 사이즈가 픽셀 단위로 자유롭게 조절 가능하다. 또 고 대역폭의 라이브 스트림 지원하며 강력한 멀티캐스트 기능을 가지고 있다.

③ Darwin Streaming Server

비디오 스트리밍 서버로서는 최초로 오픈 소스 애플리케이션이라는 장점을 갖으며 이런 오픈 소스 정책으로 인하여 사용자수가 급격히 증가하고 있다. 또 윈도우와 애플 OS를 모두 지원한다. 또 업계 표준 스트리밍 프로토콜인 RTP/RTSP를 지원하며, 음성과 이미지의 고 품질로 인기가 높다 [6].

본 논문에서는 업계 표준으로 자리잡아가는 스트리밍 프로토콜인 RTP/RTSP를 구현하여 리눅스 클러스터 상에 스트리밍 서버를 구현한다. 4장에서는 실제 클러스터를 구성하기 위한 IP 부하 분산 방식을 살펴보고, RTP/RTSP를 지원하는 스트리밍 서버의 설계와 이를 재생하기 위한 클라이언트 설계에 대하여 자세한 구현 사항을 기술한다.

4. 클러스터형 스트리밍 서버 설계 및 구현

이 장에서는 먼저 빠른 패킷 전송과 부하 분산기의 병목 현상을 제거하여 성능을 높이고 확장이용이하도록 하기 위하여 하나의 가상 IP를 공유하는 다이렉트 라우팅에 의한 부하 분산 서버 설계 방법을 보인다. 그리고 이를 바탕으로 하여 스트리밍 서비스를 위한 서버 설계 방법과 클라이언트 재생기의 구현 사례를 통해 결과를 기술한다.

4.1 부하 분산 서버 설계

그림 3에서와 같이 클러스터 서버는 리눅스 가상 서버의 다이렉트 라우팅을 사용하여 리눅스 상

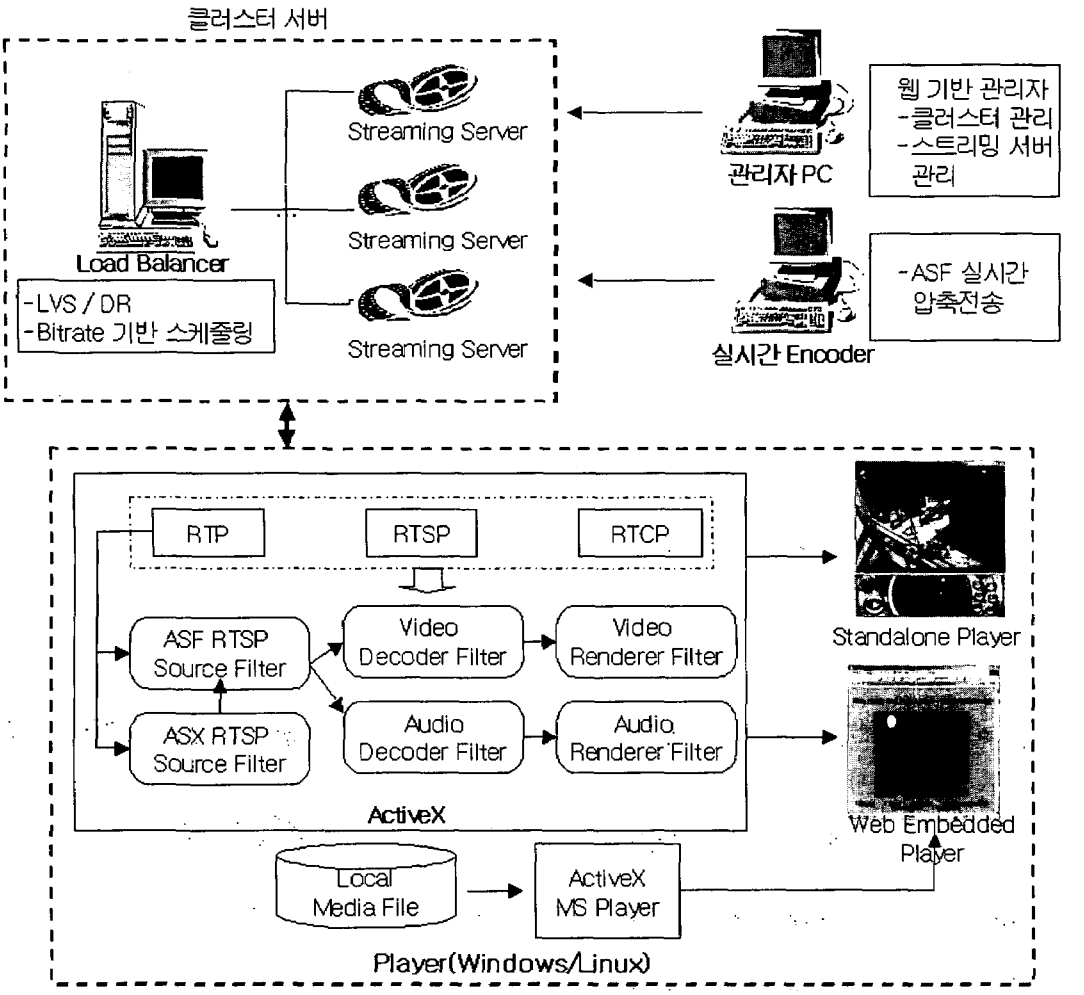

그림 3. 클러스터형 스트리밍 서버 구조

에 스트리밍 서비스를 위한 실제 서버로 구성된다.

많은 인터넷 서비스들은 요청 패킷에 비하여 응답 패킷은 더 큰 경우가 많다. 스트리밍 서비스 경우 또한 작은 요청 패킷에 비하여 상대적으로 큰 응답 패킷을 처리해야 한다. 리눅스 가상 서버의 부하 분산 방식 중, 네트워 주소 번역(NAT)에 의한 방법은 요청의 작업 할당과 응답 패킷을 모두 부하 분산 서버에서 처리하기 때문에, 서버가 증가하는 경우, 부하 분산 서버가 병목 구간이 된다. IP 터널링과 다이렉트 라우팅에 의한 부하 분산 방식은 실제 서버가 클라이언트에게 직접 응답 패킷을 전송하므로, NAT에 의한 방법 부하 분산기의 병목 구간에 대한 문제를 해결할 수 있다. 그리고 다이렉트 라우팅은 터널링에 대한 오버헤드가 없으므로 보다 확장성이 좋은 클러스터 시스템을 구축할 수 있다.

본 논문에서는 그림 2와 같이 스트리밍 서버가 수행되는 펜티엄 II 233MHz, 리눅스 환경의 서버 세대와 하나의 펜티엄 II 400MHz의 다이렉트 라우팅 방식의 부하 분산 서버로 클러스터형 스트리밍 서버를 구현하였다.

4.2 스트리밍 서버 설계

스트리밍 서버는 리눅스 상에서 구현되었으며, ASF 파일 형식을 지원하며 RTSP와 RTP 프로토콜을 구현하였다. ASF(Advanced Streaming Format)는 윈도우즈 계열의 스트리밍 서비스에서 사용하는 파일 형식이며 윈도우즈 계열의 스트리밍 서비스를 위하여 마이크로소프트사에 의해 독자적으로 개발된 스트리밍 전용 포맷이다. 따라서 현재는 Windows NT, Windows 2000같은 운영체제가 있어야만 서비스가 가능하다. 하지만 본 연구

에서는 리눅스 상에서 ASF 파일 형식을 스트리밍 할 수 있는 서버를 자체적으로 개발하였다. 따라서 WMT를 이용하는 스트리밍 서비스를 할 경우 요구되는 높은 비용(운영체제의 구입비용)을 대폭 절감할 수 있도록 하였다. 또 기존의 윈도 계열의 스트리밍 서버는 자체 프로토콜을 사용하였지만, 본 연구에서는 표준으로 자리잡은 RTP/RTSP를 사용하여 범용성을 높였다.

(1) RTSP 메소드

RTSP는 제어와 실시간 전송을 위해서 RTP의 상위 계층으로 설계되었다. 따라서 RTP가 수정되거나 기능이 추가되어도 RTSP에서 계속적으로 사용할 수 있다는 장점이 있다. RTSP는 기능적으로 HTTP와 매우 유사하다. 구문과 작동방법이 유사하고 대부분의 경우 HTTP확장은 RTSP에 추가된다. RTSP는 네트워크상에서 아래와 같은 Method들을 정의하며, 이것은 재생기가 서버에게 특정 역할을 요청하는 Remote-Controller의 기능을 가능하게 한다. 따라서 RTSP에서 제공되는 메소드를 이용하여 다양한 사용자 피드백이 가능하다. 본 연구에서는 이런 RTSP의 메소드를 이용하여 PLAY, PAUSE, Random Access가 모두 가능하도록 서버를 구현하였다. 다음 표 1.의 내용은 본 연구에서 구현한 RTSP 메소드이다.

Method	설 명
SETUP	세션 설정을 요구하며, 서버는 필요한 리소스를 할당
DESCRIBE	스트리밍 되는 미디어에 대한 description 정보를 요구한다.
PLAY	스트리밍 서비스의 시작을 요구한다.
PAUSE	스트리밍 서비스의 일지 중단을 요구한다.
TEARDOWN	스트리밍 서비스의 종료와 세션 종료를 요구한다.

표 1. RTSP Method

(2) RTP의 이용

RTP는 스트리밍이 이루어지는 네트워크 상의 실제 패킷 단위이며, 간단한 헤더와 미디어 데이터로 이루어지며 UDP를 이용하여 전달된다. RTP 패킷은 12Byte의 헤더와 Payload로 구성된다. Payload는 또한 encoding-sepecific layer에 의해 싸여진다. 본 연구에서는 RTP의 Payload 부분에 ASF 데이터를 넣어 RTP 패킷을 구성하였다. 구성된 패킷 정보는 다음 그림과 같다. 그림과 같이 패킷을 구성함으로써 ASF파일을 RTP형식으로 전송이 가능하게 된다. 패킷을 받은 클라이언트측에서는 RTP 헤더를 검사하여 시간 정보 등을 추출하여 재생에 이용하게 된다. 이런 일련의 작업은 RTP Service Thread에서 처리된다. RTP Service Thread에 관한 정보는 다음절에 자세히 알아보도록 한다.

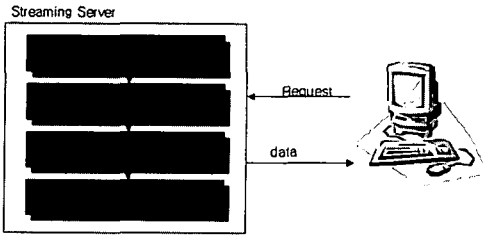

그림 4. 스트리밍 서버 구조

(3) 서버 구조

그림 4는 스트리밍 서버의 개략적인 구조를 나타낸다. TCP Streaming Server는 클라이언트의 접속을 기다리는 모듈이다. 클라이언트가 접속하면 TCP Streaming Server는 클라이언트와 연결을 설정한 후 클라이언트의 요청을 RTSP Parser에게 전달한다. RTSP Parser는 클라이언트가 보내온 RTSP 메소드들을 판단하며, 분석된 내용에 맞는 응답을 클라이언트에 보내게 된다. 또 클라이언트와 세션을 설정한다. 세션이 설정되면, Session Manager가 클라이언트와의 고유 세션을 관리하게 되며. 실제 비디오/오디오 데이터를 RTP Service Thread 모듈을 통해 클라이언트에 전달한다. RTP Service Thread는 서버에 데이터를 실제로 클라이언트에 전달하는 모듈이며, RTP 프로토콜을 이용하여 데이터를 전송하게 된다.

4.3 스트리밍 클라이언트 설계

클라이언트 재생기의 설계는 DirectShow의 Filter와 ActiveX Controller를 기본으로 설계되었으며, ASF/ASX RTSP Source Filter와 Video/Audio Decoder Filter, Video/Audio Renderer Filter로 구성된다. 또한 RTSP를 담당하는 ActiveX Controller가 구현되었다. 각 구성요소에 대하여 기술한다.

(1) RTSP Source Filter

RTSP Source Filter는 서버로부터 들어오는 RTP 패킷 정보를 분석하여 ASF 패킷을 분리해내고, 분리된 ASF 패킷을 다시 오디오 데이터와 비디오 데이터로 분리하여 오디오 데이터일 경우에는 Audio Decoder로 비디오 데이터일 경우에는 Video Decoder 모듈로 전송한다. 또 소스 필터에서는 RTP 패킷의 오류검사 및 ASF 패킷의 오류검사도 하게 된다.

(2) Video/Audio Decoder

Video/Audio Decoder 모듈은 소스 필터로부터 출력된 압축된 비디오/오디오 데이터를 압축 해제하여 Video/Audio Render에 전달한다. 본 연구에서는 소스필터와 비디오/오디오 Decoder를 자체 개발하였으며, 개발된 각 모듈을 모두 묶어 새로운 ActiveX Controller로 구성하였다.

(3) Video/Audio Renderer

Video/Audio Decoder로 부터 압축이 풀린 비디오/오디오 데이터를 랜더링하여 실제로 사용자에게 보여주는 모듈이다. 본 연구에서 Video/Audio Decoder 모듈은 DirectShow에서 제공되는 필터 모듈을 그대로 사용하였다. 이렇게 함으로써 별도의 추가 작업 없이 윈도우즈에서 기본적으로 제공되는 Active Movie 컨트롤을 사용하여 비디오/오디오 데이터를 재생할 수 있다.

(4) RTSP ActiveX Controller

RTSP ActiveX Controller는 클라이언트 재생기 내의 모든 모듈을 통합 관리하는 메인 모듈이다. 클라이언트가 플레이어가 시작되면 먼저 콘트롤러가 시작되며, 사용자가 스트리밍 할 파일을 선택하면 콘트롤러는 서버와 통신하여 세션을 설정한 후 Source Filter, Video/Audio Decoder, Video/Audio Renderer를 메모리에 로딩한다. 서버로부터 온 데이터는 각 모듈들을 거쳐 처리되며, 이런 과정을 콘트롤러가 관리하게 된다.

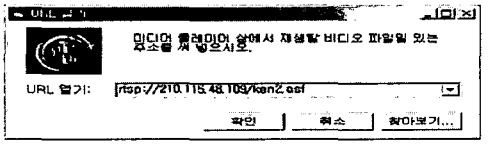

그림 5. URL 열기 대화상자

그림 5는 클라이언트 플레이어에서 서버의 주소와 플레이 할 파일을 입력하는 화면이다. 적절한 URL과 파일명을 입력하면 그림5과 같이 선택된 파일이 실시간으로 재생된다. 클라이언트 GUI는 윈도 미디어 플레이어와 비슷하게 구성하여 사용자에게 친숙하고, 사용이 용이하도록 하였다.

5. 결론

본 논문에서는 리눅스 클러스터 시스템을 기반으로 하는 비디오/오디오 스트리밍 서버를 구현함으로써, 가격 대 성능이 우수한 스트리밍 서버를 실현하였다. 아울러, 클라이언트가 스트리밍 서비스에 접근할 수 있도록 전용 재생기를 구현하였다.

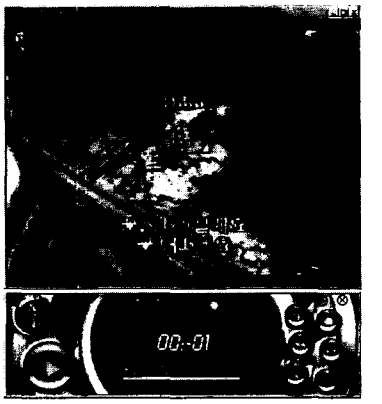

그림 6. 클라이언트 재생기 실행 화면

다이렉트 라우팅에 의한 클러스터 서버구조는 부하 분산 서버가 라우팅 기능만을 갖고, 클라이언트 요청의 처리는 클러스터 내의 스트리밍서버가 처리함으로써, 확장성이 더 뛰어나다. 부하 분산 서버의 스케줄링 알고리즘은 현재 구현된 시스템에서는 일반적인 Round-Robin 스케줄링 또는 Least-Connection 스케줄링을 사용하였다. Round-Robin 스케줄링으로도 좋은 부하 분산 성능을 보이나 좀더 낳은 성능 향상을 위한 멀티미디어 데이터의 특징에 따른 Bitrate 기반의 스케줄링의 연구가 필요하겠다. 또한, 향후 연구에서는 멀티미디어 데이터의 특징을 보이는 대용량의 VOD 서비스에 대하여 리눅스 클러스터형 스트리밍 서버에서의 성능을 테스트하여 분석하고 그 시스템에 반영할 것이다.

감사의 글

본 논문은 정보통신부의 정보통신우수대학원 지원사업의 일부 지원으로 이루어 졌음

참 고 문 헌

[1] W. Zhang, "Linux Virtual Server for Scalable Network Services", Linux virtural server project.
[2] 권세오, 김상식, 김동승, "리눅스 클러스터형 웹서버 설계", 정보과학회지, vol. 18, 2000.
[3] 최재영, 최종명, 김은희, 김민석, "고가용성 리눅스", 정보처리학회지, vol. 6, 1999.
[4] 서대화, "리눅스 클러스터 환경에서의 분산 파일 시스템", 정보처리학회지, vol. 6, 1999.
[5] W. Zhang and et al., "Linux virtual server project", *http://www.linuxvirtualserver.org*, 1998.
[6] Darwin Streaming Server, *http://www.publicsource.apple.com/projects/streaming*.