제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003. 5)

RTP/RTSP 스트리밍 서버와 클라이언트에서의 Pay-Per-View 설계 및 구현

권순만, 주한규
한림대학교 컴퓨터공학과
e-mail: {smkwon, hkjoo}@hallym..ac.kr

Design and Implementation of Pay-Per-View in RTP/RTSP Streaming Server and Client

Sun-Man Kwon, Hankyu Joo
Dept of Computer Engineering, Hallym University

요 약

인터넷이 활성화되면서 다양한 서비스가 제공되어지고 있다. 그 가운데 하나가 MOD(Multimedia on Demand) 서비스이다. MOD는 다양한 프로그램을 사용자가 원하는 시점에 볼 수 있도록 한다. MOD 서비스를 유료화할 경우 시청한 양만큼 요금을 지불하는 pay-per-view 방법을 생각할 수 있다.

본 논문에서는 RTP(Realtime Transport Protocol)/RTSP(Real Time Streaming Protocol)를 이용한 스트리밍 서버와 클라이언트를 구현하고 여기에 PPV(Pay-Per-View)를 적용해 본다.

1. 서론

인터넷이 활성화되면서 다양한 웹 기반 서비스가 제공되어진다. 그 가운데 하나가 MOD 서비스이다. MOD는 다양한 프로그램을 사용자가 원하는 시점에 시청 할 수 있도록 한다. MOD를 유료화할 경우 다양한 지원방법이 필요하다. 이를 지원하는 방법은 월 회비를 지불하고 자유롭게 이용할 수 있는 정액 방법과 시청한 양만큼 요금을 지불하는 pay-per-view 방법으로 나눌 수 있다. 정액 사용 방법의 경우 월 회비를 납부하는 회원만이 프로그램을 이용할 수 있도록 하는 기법이 필요하다. pay-per-view 방법의 경우에는 허용된 사람만이 프로그램을 이용할 수 있도록 하여야 하며 사용자의 사생활 보호뿐만 아니라 프로그램을 이용한 사람에게는 이용한 분량에 따라 요금을 부과할 수 있는 방법이 필요하다. pay-per-view 방식의 경우 정액 방식보다 복잡하나, 서비스의 유연성이 훨씬 뛰어나다.

본 논문에서는 RTP와 RTSP에 기반한 스트리밍 서버와 클라이언트를 구현한다. 그리고 PPV 프로토콜을 설계하고 앞에서 구현한 스트리밍 서버와 클라이언트에 적용하여 유연성을 갖는 스트리밍 서비스를 구현하고자 한다.

본 논문의 구성은 다음과 같다. 2 절에서는 RTP[1]와 RTSP[2]에 대하여 기술하고, 3 절에서는 제안된 PPV 프로토콜에 대하여 기술한다. 4 절에서는 설계 및 구현 정보를 기술하고 마지막으로 5 절에서는 결론 및 향후 연구를 제시한다.

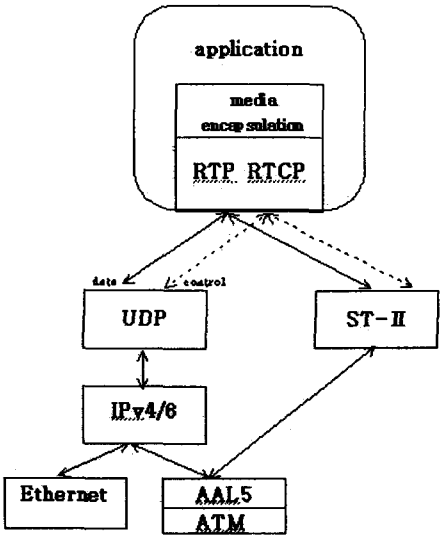
2. RTP 와 RTSP

본 절에서는 스트리밍 서버와 클라이언트에 이용되는 RTP와 RTSP 프로토콜에 대해서 알아본다.

2.1 RTP(Real-Time Transport Protocol)

RTP는 IETF(Internet Engineering Task Force)의 AVT working group에 의해 제안된 프로토콜로 RFC1889 표준안으로 승인되었다. RTP는 기본적으

로 실시간 특성을 가지는 데이터를 전송하는 기능을 수행하며, 전송품질을 모니터링해서 피드백하는 RTCP와 함께 동작한다. RTP 패킷의 Sequence Number는 패킷손실여부를 확인하고 패킷간의 순서를 재조정할 수 있게 하며 Time Stamp는 데이터의 재생시점을 결정하는데 이용된다. RTCP는 RTP와 결합하여 작동하는 제어프로토콜로써 동작하며 데이터 전송품질에 대한 피드백과 멤버쉽정보를 전달한다. 주요한 피드백정보는 패킷손실률, 지터, 라운드 트립지연등이 있으며, 이를 바탕으로 서버는 흐름제어나 에러제어 등을 수행한다. [그림 1]은 프로토콜상의 RTP를 나타내고 있다.



[그림 1] 프로토콜상의 RTP

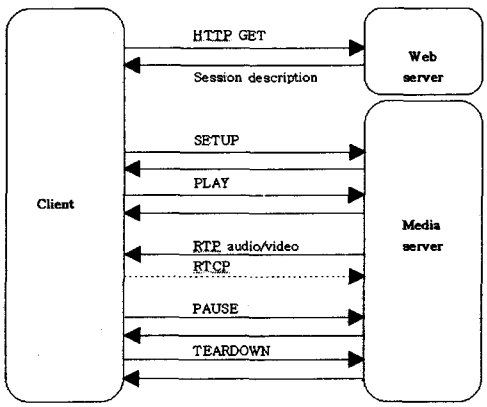
2.2 RTSP(Real-Time Streaming Protocol)

RTSP는 실시간 특성을 가진 데이터 전송을 위한 응용계층 제어 프로토콜이다. 따라서 전송은 RTP/UDP와 같은 다른 프로토콜을 사용하여 이루어진다. RTSP는 VCR과 같은 즉, play, forward, rewind, pause, stop, 그리고 record 명령어들을 제공하여 멀티미디어 서버에 대하여 network remote control과 같은 역할을 한다.

RTSP는 HTTP/1.1과 매우 흡사하다. 따라서 HTTP의 확장이 대부분의 경우에 RTSP에도 추가될 수 있다. RTSP에서 제공하는 메소드(method)는 아래와 같다.

OPTIONS	사용가능한 메소드를 얻음
SETUP	트랜스포트 연결을 설정
ANNOUNCE	미디어 개체의 설명을 변경
DESCRIBE	미디어 개체의 설명을 얻음
PLAY	재생을 시작
REDIRECT	클라이언트를 새로운 서버로 리다이렉트
PAUSE	스트림 전송을 중단
SET_PARAMETER	장치 또는 인코딩을 제어
TEARDOWN	세션 상태를 지움

[그림 2]는 위의 메소드들을 이용한 RTSP의 기본적인 동작 과정을 보여준다. 클라이언트는 HTTP의 GET 메소드를 통해 웹서버에서 연속매체의 정보, Session Description을 가져온다. 그리고 이 정보를 이용하여 멀티미디어서버에서 SETUP 요구를 보낸다. 멀티미디어서버가 SETUP을 허용하면 클라이언트는 PLAY요구를 보낼 수 있다. PLAY요청이 받아들여지면 실제 데이터는 RTP/RTCP프로토콜을 사용하여 클라이언트에게 전송되어진다. 일시적인 재생중지를 위해서는 PAUSE 메소드를 사용하고 세션을 완전히 닫기 위해서는 TEARDOWN 메소드를 사용한다.



[그림 2] RTSP 기본 동작

3. PPV

pay-per-view 방법은 J.Zhou와 K-Y. Lam에 의하여 제안되었다[3]. 다중해쉬[4]를 사용하여 사용자의 시청 양을 제공자가 증명할 수 있는 방법이다. 본 논문에는 J.Zhou와 K-Y. Lam에 의하여 제안된 프로토콜이 가진 사생활보호 문제를 보완하여 기밀성을 갖는 프로토콜을 개발하였다.

PPV 서비스를 다음 4단계로 나눌 수 있다.

· 1단계(Subscription) : 사용자는 제공자의 서버에 접속해서 사용자 인증 단계를 거침. 이 단계에서 요금청구서를 받을 (e-mail)주소를 제공자에게 알림.
· 2단계(Browse) : 사용자는 제공자로부터 멀티미디어 정보(제목, 재생시간, 가격, 등)을 받음. 원하는 멀티미디어를 선택한 후 제공자에게 서비스를 요청함으로써 3단계로 들어감.
· 3단계(Request and View) : 제공자는 사용자의 요청 정보를 확인하고 나서 서비스를 제공함.
· 4단계(Payment and Dispute Resolution) : 사용자는 제공자에게 요금을 지불하게 됨.

여기에서 중요한 단계가 3단계이다. 3단계의 구체적인 설명을 위해 다음 표기법을 사용하기로 한다.

$H(X)$: 메시지 X의 일방향 해쉬함수.
$H^k(X) = H(H^{k-1}(X)), H^0(X) = X$: X를 k번 해쉬함.
SK_A: A의 개인키
PK_A: A의 공개키
$E_{PK_A}(X)$: A의 공개키로 메시지 X를 암호화 함.
$SIG_{SK_A}(X)$: A의 개인키로 메시지 X에 서명 함.
K_{AB}: A와 B 사이에 사용될 세션키
Id: 선택한 프로그램 제목
$Program$: 선택한 프로그램
Pr: 선택한 프로그램에 대한 가격
L: PPV 서비스의 요금 단위. (1초, 1분, 10분 등)
m: PPV 서비스 요금 단위의 전체 수.
$A \rightarrow B: X$: A가 B에게 메시지 X를 보냄.

C는 S에 등록되었고 가탈로그를 훑어본 것으로 가정한다. 3단계 즉, 요청과 뷰 단계에서 요청은 다음과 같이 작동한다.

1. $C \rightarrow S: E_{PK_S}(K_{CS}), K_{CS}(Id, \Pr, L, m)$
2. $S \rightarrow C: K_{CS}(Id, SIG_{SK_S}(Id, \Pr, L, m))$
3. $C \rightarrow S: K_{CS}(Id, SIG_{SK_C}(Id, \Pr, L, m, H^m(n)))$

C는 K_{CS}를 PK_S로 암호화하여 전송하고 Id, Pr, L, m을 K_{CS}로 암호화해서 S에게 보낸다. S는 SK_S를 이용하여 K_{CS}를 복원한 후 K_{CS}를 이용해서 Id, Pr, L, m을 획득한다. 그 후 S는 Id, Pr, L, m을 SK_S로 전자 서명을 한 후 K_{CS}로 암호화를 해서 C에게 보낸다. C는 K_{CS}로 복호화를 하고 S의 전자 서명을 확인하여 키 교환이 성립되었음을 확인한다. 그 후 C는 랜덤하게 생성한 정수 n을 m번 해쉬한 $H^m(n)$을 Id, Pr, L, m과 함께 SK_C로 전자 서명을 한 후 K_{CS}로 암호화를 해서 S에게 보낸다. 이 때 C는 n을 저장해 놓는다. S는 K_{CS}로 복호화를 하고 전자 서명을 보고 올바른 C인가를 확인한다. 확인 후 S는 $H^m(n)$을 저장해 놓는다. 여기까지가 요청 작동 설명이다. 뷰는 다음과 같이 작동한다.

1. $S \rightarrow C: K_{CS}(Program)$
2. $C \rightarrow S: K_{CS}(Id, H^{m-j}(n))$

S는 Program을 K_{CS}로 암호화를 해서 C에게 전송한다. C는 K_{CS}로 복호화하여 Program을 시청한다. 매 L당 C는 S에게 $H^{m-j}(n)$을 전송하여 C가 현재 프로그램을 시청하고 있음을 알린다. 즉, 프로그램을 선택하여 제일 처음 프로그램을 보기 시작할 때 $H^m(n)$을 보내며 그로부터 L만큼 시간이 흐르면 $H^{m-1}(n)$을 보낸다. $H^{m-1}(n)$을 보낸 후 다시 L만큼 시간이 흐르면 $H^{m-2}(n)$을 보낸다. S는 항상 가장 최근의 해쉬값을 보관하며, $H^{m-j+1}(n)$을 받은 후 L만큼 시간이 경과하면 $H^{m-j}(n)$을 기대한다. $H^{m-j+1}(n)$을 받은 후 L만큼 시간이 흐른 다음 $H^{m-j}(n)$을 받지 못하면 C가 더 이상 프로그램을 시청하지 않는 것으로 간주하여 프로그램 전송을 중단한다.

4. 설계 및 구현

RTSP 서버는 클라이언트의 요구를 분석하여 처리하며 미리 저장되어 있는 멀티미디어 데이터를 클라이언트에게 스트리밍하는 기능을 수행하도록 한다.

RTSP 클라이언트는 서버에서 전송되는 스트림을 초기지연을 최소화시켜 재생하는 기능을 수행한다. 이때 RTP 패킷의 헤더정보를 이용하여 여러 가지 QoS를 분석하고 이를 서버측에 피드백 하도록 한다.

PPV 는 클라이언트와 서버의 연결이 확립된 후 클라이언트는 일정한 시간이 흐르면 그에 맞는 해쉬값을 서버에 전송하도록 한다.

스트리밍 서버의 구현을 위해서 실행 플랫폼으로써 Linux 2.4.18 을 사용하였고, 프로그래밍 툴로는 GCC 3.2 를 사용하였다.

[그림 3]은 서버의 실행 모습을 보여주고 있다. 클라이언트가 서버에 연결할 때 HELLO 응답을 보내고, 멀티미디어 정보를 요청할 때 DESCRIBE 응답을 보내며, PLAY 요구를 보낼 때 데이터를 주기적으로 전송하게 된다. 또한, [그림 4]는 클라이언트의 실행모습을 보여주고 있다. 서버의 3456번 포트를 사용하여 서버에 연결한 다음 서버로부터 멀티미디어 정보를 받고 나서 다중해쉬를 수행한다. 다중해쉬 값을 서버에게 보내고 나서 서버상의 멀티미디어를 스트리밍하여 재생한다. 재생이 끝나고 나면 요금이 출력되게 되어 있다. 요금에 대한 논란이 있을 경우에는 초기에 생성한 다중해쉬값에 의해 요금 논란을 없앨 수 있다.

```
[smkwon@sunman server]$ ./ppv-server ../etc/example.cfg
Control channel port set to 3456.
BasePath set to "/home/smkwon/test/ppv/etc"
RTSP port 3456.
HELLO request received.
HELLO response sent.
HELLO sent.
HELLO reply received.
DESCRIBE request received.
Describe response message:
RTSP/0.1 200 2 OK
Date: 19 Mar 2003 13:12:18 GMT
Content-type: application/sdp
Content-Length: 214

v=0
o=- 2890844256 2890842807 IN IP4 220.66.152.176
s=RTSP Session
i=PPV
u=rtsp://220.66.152.176/korea_song.wav
t=0 0
m=audio 0 RTP/AVP 101
a=MaxBitRate:88200
a=MaxPktSize:551
a=TypeSpecificData:"AAEAAQAAKxEACAAB"

DESCRIBE response sent.
SETUP request received.
Setup response message:
Session: 1
Content-Length: 0
Transport: rtp/udp;port=2610


SETUP response sent.
PLAY request received.
PLAY response sent.
TEARDOWN request received.
TEARDOWN response sent.
```

[그림 3] 서버의 실행 화면

```
Command :
o rtsp://220.66.152.176:3456

handle_command: o rtsp://220.66.152.176:3456

HELLO
g korea_song.wav

handle_command: g korea_song.wav

New session: 0
PLAY started.
WAV File
Frequency: 11025
Channels: 1
Bits Per Sample: 8
PLAY done.
c

handle_command: c

        * Pay-Per-View *
 * View Time   : 76 second
 * Charge : 76 x 10(won) = 760(won)
CLOSE
```

[그림 4] 클라이언트의 실행 화면

5. 결론 및 향후연구

본 논문은 요금부과에 있어서 유연성을 갖는 PPV를 적용한 스트리밍 서버와 클라이언트를 구현하였다.

현재 구현된 스트리밍 서버와 클라이언트는 forward, rewind의 기능이 없다. 이 기능의 추가가 필요하다. 이 기능이 추가되고 나면 PPV가 더 큰 역할을 할 수 있다. 위치 변경(forward, rewind)후의 play에서 요금부과는 다르게 적용될 필요가 있기 때문이다. 즉, play 모드로 수행하던 중 replay 영역을 만나면 replay 모드로 변환하여 기존의 요금보단 저렴하게 부과될 수도 있다. 지금은 가장 단순한 미디어 포맷인 WAV만 지원하지만 향후 MPEG-4(Moving Picture Expert Group-4)와 같은 비디오 스트림이 지원될 필요가 있다.

참고문헌

[1] H.Schulzrinne, S.Casner, R.Frederick, and V.Jacobson, "RTP: A transport protocol for real-time applications", RFC-1889, 1996

[2] H.Schulzrinne, A. Rao, R.Lanphier, "Real Time Streaming Protocol(RTSP)", RFC-2326, 1998

[3] Zhou, A. and Lam, K-Y., "A Secure Pay-per View Scheme for Web-Based Video Service," Proceedings of the 2nd International Workshop on Practice and Theory of Public Key Cryptography, LNCS 1560, pp.315 - 326, 1999, Kamakura, Japan.

[4] Pedersen, T., "Electronic Payments of Small Amounts", Proceedings of Cambridge Workshop on Security Protocols, LNCS 1189, pp. 59-68, 1996, Cambridge, U.K.