SIP 기반 음성 통신 환경에서의 실시간 모니터링 플랫폼 개발

우호진, 이원석
연세대학교 컴퓨터과학과

The Real-time Monitoring for SIP-based VoIP Network

Woo Ho Jin, Lee Won Suk
Yonsei University
E-mail : {judas, leewo}@database.yonsei.ac.kr

요 약

고속 인터넷 망 구축과 멀티미디어 통신 수요의 증가에 따라 VoIP는 기존의 PSTN 망의 대체 혹은 확장 기술로서 지속적으로 검증되어 왔다. 음성 데이터 처리 규약들 중 SIP는 다른 규약에 비해 신호 처리 단계가 간단하기 때문에 이를 기반으로 RTP를 활용하여 음성 통신 시스템을 구축하는 사례가 늘어나고 있다. 그러나 RTP의 특성상 패킷을 처리할 때마다 복원 과정이 필요하며, 다중 세션으로 통신이 발생할 경우 전체 패킷들의 관리가 복잡해지므로 이들 간에 혼선 없이 데이터를 처리 및 유지할 수 있는 방법론이 요구된다. 본 논문에서는 SIP 기반의 IP 전화를 통해서 고객과 상담원 간의 통화 이벤트가 발생하는 일반 콜센터 환경에서 RTP 음성 데이터를 처리하는 다중 세션 어플리케이션의 구축 사례를 제시한다. 구현한 시스템은 IP 전화에서 발생하는 통화 내역을 통합 스위치 서버에서 포트 미러링하여 녹취 및 녹음 서버로 전송하며, 전송된 패킷 정보들의 세션이 유지되고 있는 동안 음성 데이터를 실시간으로 모니터링한다.

1. 서론

VoIP(Voice over Internet Protocol)는 고속 인터넷 망 구축과 멀티미디어 통신 수요의 증가에 따라 기존의 PSTN(Public Switched Telephone Network) 망의 대체 혹은 확장 기술로서 지속적으로 검증되어 왔다. 음성 데이터 처리를 비롯하여 인터넷 방송, 원격 영상 회의 등의 어플리케이션에서 요구되는 실시간 멀티미디어 응용 데이터 전송은 일반적으로 RTP(Real-time Transport Protocol) 및 RTCP(Real-time Transport Control Protocol)를 따라 이루어지며, 이러한 데이터들 간의 호 처리(call/signal processing)는 SIP(Session Initiation Protocol), MGCP (Media Gateway Control Protocol), H.323 등의 규약을 따르는 경우가 대부분이다. 특히, SIP는 다른 규약

이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국과학재단의 국가지정연구실사업으로 수행된 연구임 (No.R0A-2006-000-10225-0)

에 비해 신호 처리 단계가 간단하기 때문에 이를 기반으로 RTP를 활용하여 음성 통신 시스템을 구축하는 사례가 늘어나고 있다. 그러나 RTP는 연결 설정이나 데이터 품질, 원본 순서를 보장하지 않는 특성을 가지고 있기 때문에 패킷(packet)을 처리할 때마다 타임스탬프(timestamp) 및 순서 정보(sequence)를 기반으로 손실을 최소화하여 복원하는 과정이 필요하다. 또한 다중 세션(multi session)으로 통신이 발생할 경우 전체 패킷들의 페이로드(payload) 관리는 더욱 복잡해지므로 이들 간에 혼선 없이 데이터를 처리 및 유지할 수 있는 구조 및 방법이 필요하다.

본 논문에서는 이러한 다중 세션 어플리케이션의 한 예로 SIP 기반의 IP 전화를 통해서 고객과 상담원 간의 통화 이벤트가 발생하는 일반 콜센터(call center) 환경에서 RTP 음성 데이터를 처리하는 플랫폼 구축 사례를 제시한다. 구현한 시스템은 G.711 표준 기반으로 동작하는 IP 전화를 통해 발생하는 두 종단(terminal) 간의 통화 내역을 통합 스위치 서버에서 수신하여 자동 포트 미러링(port mirroring)을 통해 별도의 녹취/녹음 서버로 전송한다. 전송된 음성 패킷 정보들은 동시에 발생하는 약 수십 개 이상의 세션으로서, 관리자가 이러한 음성 데이터를 실시간으로 수신하여 녹취할 수 있는 기능을 지원한다. 이를 위해서 구현한 시스템에서는 패킷 정보들을 순차적으로 유지하며 세션이 종료되면 개별적인 음성 파일로 저장하는 한편, 해당 세션이 유지되고 있는 동안에는 음성 데이터를 실시간 모니터링(real-time monitoring)하는 시스템을 제안, 구축하였다.

2. 본론

2.1. 호 처리

IP 전화 장비를 기반으로 하는 통화는 전술(前述)한 바와 같이 SIP 및 RTP 규약에 따라 이루어진다. SIP는 멀티미디어 세션의 설정, 수정 및 종료를 위해 사용되는 응용 계층 신호 규약으로서, 호 설정 및 전송 처리 프로세스는 가장 단순한 경우 다음과 같은 순서로 진행된다.[1,2]

① INVITE (request) [User A → User B]
② 180 Ringing (response) [User A ← User B]
③ 200 OK (response) [User A ← User B]
④ ACK (request) [User A → User B]
⑤ 통화 진행 (RTP media)
⑥ BYE (request) [User A ← User B]
⑦ 200 OK (response) [User A → User B]

콜 발생 시의 사용자 식별은 단순히 SIP 세션의 URI(Uniform Resource Indicator) 및 IP 주소를 이용한다. 관리되는 SIP 패킷의 구조는 그림 1과 같다.

2.2. 통화 데이터 처리

통화 데이터, 즉 실제 페이로드의 처리와 운용의 기반이 되는 RTP는 음성을 포함한 멀티미디어 데이터를 실시간으로 전송하기 위한 규약이다. RTP는 주로 미디어 스트림의 전송을 목적으로 설계된 UDP 기반의 프로토콜이기 때문에, 네트워크의 신뢰성을 가정하지 않는다. 즉, 데이터 수신부에서 받은 패킷이 적합한 순서로 전송되었는지 또는 손실된 패킷이 있는지의 여부를 알 수 없다.

```
typedef struct sip_pkt
{
  u_int8   sip_message_type : 1;          //0:요청, 1:응답
  u_int8   *sip_type;          //SIP 종류(INVITE, BYE 등)
  u_int8   *src_ip_addr;                  //출발지 IP 주소
  u_int8   *dst_ip_addr;                  //도착지 IP 주소
  u_int8   *src_phone_num;                //송출 전화번호
  u_int8   *dst_phone_num;                //수신 전화번호
  u_int8   *call_id;                //호 별로 부여되는 ID
  long   sip_length;                      //SIP 패킷의 길이
  long   rtp_port_num;          //RTP 미디어의 포트번호
  struct sip_pkt   *next;     //다음 SIP를 가리키는 포인터
}sip_pkt;
```

그림 1. 수집 SIP 패킷 구조

따라서 모든 RTP 처리 과정에서는 각 패킷마다 기술되어 있는 타임스탬프와 순서 정보를 이용하여 전역 클락(clock)에 대한 동기화, 원본 순서의 확인 및 재정렬, 수신 단에서의 손실 검출 및 복구 등의 작업을 수행해야 한다.[3] 본 논문의 구현에서는 두 종단 간의 호 연결, 즉 1:1 통화가 대량으로 동시에 발생하는 콜센터 환경을 가정하였으므로 별도의 믹서(mixer)/트랜스레이터(translator) 기능이나 다중 미디어 동기화는 고려하지 않았다. 그 외에 실제로 통화 식별 및 구분을 위해서 사용되는 IP 주소 및 포트, 순서 번호 필드들 외의 헤더(header) 값들[4] 역시 제외하여 간소화하였다. 다음의 그림 2에 수집되는 RTP 구조를 나타내었다.

```
typedef struct rtp_pkt
{
  rtp_hdr *RTP_header;                          //RTP 헤더
  rtp_ext *RTP_extension;              //RTP 확장(extension)
  u_int8   *payload;          //패킷의 실제 저장 페이로드
  char   src_addr[20];                    //출발지 IP 주소
  char   dst_addr[20];                    //도착지 IP 주소
  long   payload_len;                    //페이로드의 길이
  int   srcPort;                          //출발지 포트번호
  int   dstPort;                          //도착지 포트번호
  struct rtp_pkt *next;       //다음 RTP를 가리키는 포인터
  struct rtp_pkt *prev;      //이전 RTP를 가리키는 포인터
} rtp_pkt;

typedef struct rtp_info_pkt
{
  char   *session;             //현재 RTP 패킷의 세션 정보
  struct rtp_pkt   *first;        //현재 호의 시작 RTP 주소
  struct rtp_pkt   *last;       //현재 호의 마지막 RTP 주소
  struct rtp_info_pkt   *next;          //다음 세션의 포인터
}rtp_info_pkt;
```

그림 2. 수집 RTP 패킷 구조

2.3. 관리자 모니터링

녹취/녹음 서버에 도착한 패킷들은 메모리에 순차적으로 적재되며, 각 통화가 종료된 경우 및 관리자의 모니터링 요청이 발생한 경우 해당하는 패킷의 페이로드를 mp3 형식으로 인코딩(encoding)하게 된다. 인코딩 과정에서는 공개 라이브러리인 Lame[5] 패키지를 이용하였다.

인코딩이 완료된 mp3 데이터는 두 가지 목적에 따라 사용된다. 첫 번째는 통화가 종료되었음을 알리는 SIP 패킷의 BYE 메시지가 확인되었을 때 물리적인 mp3 파일로 생성하는 것이다. 생성되는 파일명은 일괄적으로 “[통화시작일시][통화종료일시]_[발신자번호][수신자번호].mp3”의 형식을 가지도록 한다.

두 번째 경우는 관리자가 현재 상황의 모니터링을 요청하였을 때 인코딩된 음성 데이터를 전송하여 관리자가 들을 수 있도록 지원하는 것으로, 다음과 같은 순서로 진행된다.

① 관리자 클라이언트(client) X가 통화 C에 대한 감청 요청
② 통화 C에 대응하는 IP 집합 (A,B) 검색
③ (A,B) 세션이 존재하면 RTP 패킷 페이로드 집합 rtp_pkt를 first부터 last까지 인코딩
④ 인코딩 데이터를 주소지 X로 UDP 전송
⑤ 관리자 클라이언트 X에서 데이터 디코딩

실시간 모니터링을 지원하는 구현 시스템의 전체 구성도는 그림 3과 같다. N명의 사용자(user), 즉 상담원들이 각각 1명씩 고객과의 통화가 발생하였을 때 각 사용자의 콜 프로세싱 모듈(call processing module)에서는 소켓을 통해서 발생한 sip_pkt 및 rtp_pkt를 미리 지정한 IP 주소의 서버로 전송한다. 각 콜 프로세싱 모듈 프로그램은 상담원마다 개별적인 다수의 PC에서 구동시킬 수도 있으나, 본 구현에서는 스위칭 성능 및 트래픽 확인을 위해 스위치 서버에 N개의 IP 전화 단말기를 물리적으로 접속하여 포트 미러링하고 데이터를 녹취 서버로 자동 전송하도록 구성하였다.

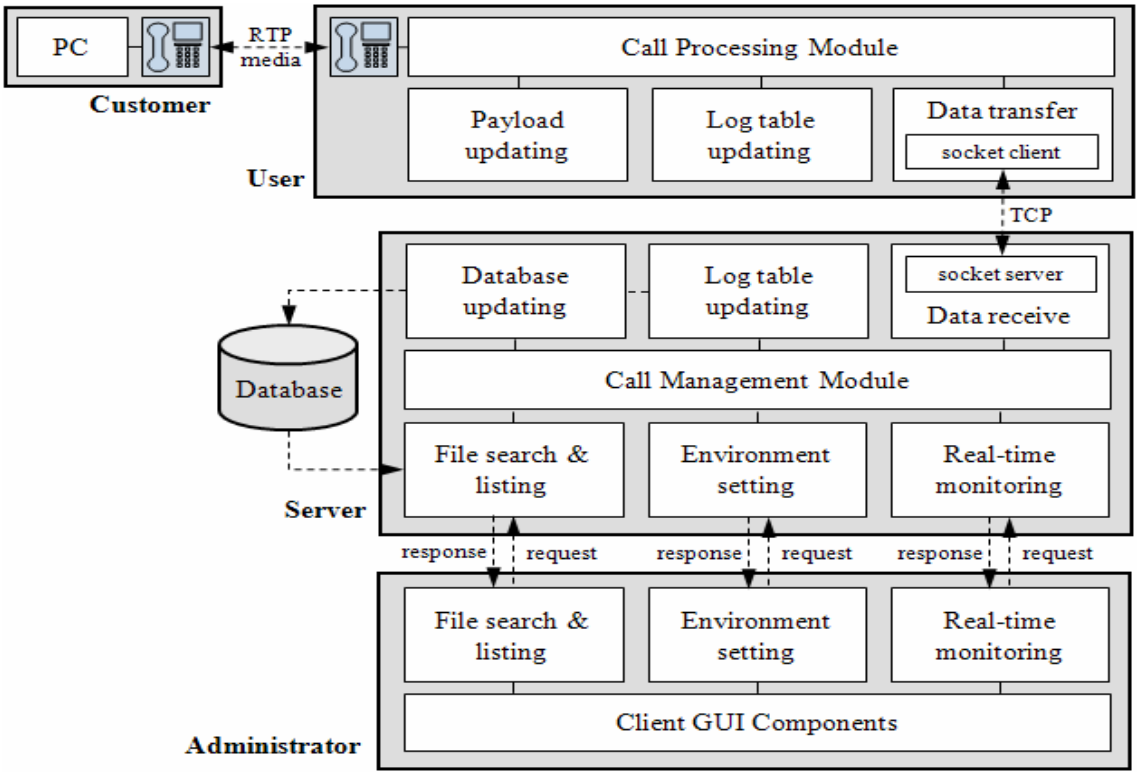

그림 3. 실시간 모니터링 플랫폼 구성도

이에 따라 동시에 최대 N개의 다중 세션이 녹취 서버 주소지로 들어오게 되며, 서버는 sip_pkt의 IP 주소지 값들을 확인하고 이에 대응하는 rtp_pkt를 검색하여 rtp_info_pkt의 first부터 적재한다. 콜 관리 모듈(call management module)은 메모리 상에 이 구조체 리스트를 유지하다가 관리자의 요청 또는 통화 종료에 따라서 관리자 클라이언트에 인코딩 데이터를 전송하거나 데이터베이스에 파일을 기록하는 작업을 수행한다.

3. 결론

본 논문에서는 방대한 음성 통신이 발생하는 다중 세션 어플리케이션으로서 SIP 기반의 IP 전화를 통해 운용되는 콜센터 환경에 대해 실시간 모니터링을 지원하는 시스템 사례를 보였다. 이는 관리자에게 현재 통화 중인 다수의 상담원들의 통화 내역을 선택적으로 실시간 청취 및 관리할 수 있도록 지원한다. 단, 구현한 시스템은 전체 네트워크 전송 속도가 낮아지거나 사용자 수가 늘어남에 따라 그에 비례하여 부하(overhead)가 증가하게 된다. 향후 이러한 요소들에 의존성이 적은 플랫폼으로 모듈화 및 추가적인 개발이 필요하다.

[참고문헌]

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, 2002.

[2] A. Johnston, S. Donovan, R. Sparks, C. Cunningham and K. Summers, "Session Initiation Protocol (SIP) Basic Call Flow Examples", RFC 3665, 2003.

[3] C. Perkins, "RTP: Audio and Video for the Internet", Addison-Wesley, pp 71-88, 2003.

[4] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, 1996.

[5] Lame MP3 Encoder, http://lame.sourceforge.net