

IMS망에서 PTT서비스의 통화 처리 성능 향상을 위한 프로토콜 압축 기법

정인환[†]

요약

본 논문에서는 유/무선 통합 패킷 망인 IMS(IP Multimedia Subsystem)에서 대표적인 응용 서비스인 PTT(Push-To-Talk) 서비스의 통화 처리 성능 향상을 위한 프로토콜 압축 기법을 제안한다. IMS망에서 PTT서비스를 하기 위해서는 현재 상용화 되어있는 Mobile망과 TRS(Trunked Radio System)망과 동일한 호 설정을 가져야 하며 다수의 단말기들 사이에 동일한 내용을 실시간으로 교환할 수 있어야하기 때문에 빠른 호 설정 및 통화 시에 대역폭이 충분해야한다. 본 논문에서 제안한 A+SigComp기법은 PTT의 호 설정 프로토콜인 SIP의 세션 설정 단계의 지연 시간을 약 10% 줄였으며 A+ROHC기법은 음성 통화시 교환되는 RTP 헤더의 크기를 줄임으로써 전송 효율 약 5% 높였다.

A Protocol Compression Scheme for Improving Call Processing of Push-To-Talk Service over IMS

In Hwan Jung[†]

ABSTRACT

In this paper, we propose a protocol compression scheme for enhancing the performance of call processing of PTT(Push-to-Talk) which is one of the important services in IMS(IP Multimedia Subsystem), a next generation integrated wired/wireless packet communication network. To service the PTT on an IMS network, it should use the same call setup procedure as legacy Mobile and TRS(Trunked Radio System) networks and have a fast call setup time and enough communication bandwidth because a number of terminals should be able to exchange same data in real time. The proposed *A+SigComp* scheme reduces the initial call setup delay of SIP by about 10%, which is used by PTT service for call setup. In addition, the *A+ROHC* scheme is proposed to compress the header of RTP packets transferred during PTT voice transmission and, as a result, about 5% of increase in communication efficiency is observed.

Key words: IMS(IP 멀티미디어 망), PTT(푸시투토크), Protocol Compression(프로토콜압축)

1. 서론

인터넷의 보급과 확산에 따른 인터넷 기술의 발전과 이동통신 사용자들의 고속 데이터 Traffic과 다양한 서비스 기능에 대한 요구가 증가함에 따라 현재 이동 통신망은 기존의 TDM(Time Division Multi-

plexing)방식의 코어 망에서 IP(Internet Protocol) 방식의 코어 망으로 이동하고 있다. 전 세계 대부분의 이동 통신업체들은 이동통신망의 발전 및 광대역 코드 분할 다중 접속(WCDMA : Wide Code Division Multiple Access), 초고속 데이터 패킷 접속(HSDPA : High Speed Data Packet Access),

※ 교신저자(Corresponding Author) : 정인환, 주소 : 서울시 성북구 삼선동 3가 389(136-792), 전화 : 02)760-4136, FAX : 02)760-4346, E-mail : ihjung@hansung.ac.kr

접수일 : 2008년 12월 1일, 완료일 : 2009년 2월 10일

[†] 종신회원, 한성대학교 컴퓨터공학과 부교수

※ 본 연구는 2007년도 한성대학교 교내연구비 지원과제임

(Charging Data Record)수집 및 전송 등의 운용 및 유지, 보수 관련 기능의 마스터 기능을 수행한다[2].

IP 기반인 IMS망에서 PTT서비스를 하기 위해 고려해야 할 두 가지 사항이 있다. 첫 번째는 현재 상용화 되어 있는 Mobile망, TRS망과 동일한 콜 셋업을 제공해야 한다. 현재 국내에서 상용화되고 있는 대표적인 PTT서비스는 주파수 공용 통신(TRP : Trunked Radio System)방식이다. 2005년 1사분기까지 가입자 30만 이상을 확보하여 사용 중인 TRS는 이동통신망이 복합된 형태로 두 개의 망에 접속 할 수 있는 겸용 단말기를 사용해야 한다. PTT서비스는 Hot-Key버튼을 이용하여 1:1또는 1:N 그룹통화가 가능하며, 콜 셋업 시간이 0.5~1초 정도이다. IMS망 PTT서비스의 경우 SIP프로토콜을 이용하며, IMS망에서 음성을 무선데이터와 같이 무선 패킷 망에 실어 전송하는 방식으로 다수의 그룹 원에게 동시에 음성 패킷을 One-Way로 전송하는 데, 다른 통신망에 비해 콜 셋업이 느리다는 단점이 있다. 콜 셋업이 느린 단점에 비해 투자비가 저렴하고 PTT서비스 이외 영상 PTT서비스, 데이터 전송 서비스, 프레즌스 서비스 등 다양한 서비스를 제공받을 수 있다[3,4].

두 번째는 IMS망에서의 주파수 자원 사용이 충분해야 한다. 2003년 8월 세계 최초로 PTT상용화를 한 Verizon사는 최근 신규 가입자 유치를 잠정적으로 중단하였다. 중단의 가장 큰 원인은 기존 음성통화 대비 PTT주파수 자원의 과다 사용으로 인한 기존 이동전화 서비스에 영향을 미쳤기 때문이다[4].

IMS기반 망에서 PTT서비스를 하기 위해서는 위에 언급한 두 가지 사항이 중요하며, 본 논문에서는 콜 셋업 시간의 단축과 통화 성능 향상을 위한 방안으로 프로토콜 압축 방법을 제시 하고자 한다.

PTT서비스는 1:N의 통화가 순간적으로 이루어져야 하기 때문에 기존의 1:1음성 및 영상 서비스 보다 훨씬 더 많은 음성 대역폭을 요구하게 된다. 또한, IMS망에서 사용하는 SIP의 경우 대역폭이 풍부한 네트워크에서의 사용을 전제로 개발되었고, SIP메시지들은 텍스트 기반이어서 크기가 수백에서 수천 바이트로 매우 크다. SIP를 대역폭이 풍부하지 못한 3G와 같은 무선 이동 통신 네트워크에서 사용 할 경우 유선망에서는 발생하지 않았던 콜 셋업 지연과 대역폭 부족의 문제들이 나타나고 있다. 콜 셋업 지연과 대역폭 부족의 문제를 해결하기 위하여 SIP, RTP,

UDP, IP와 같은 프로토콜 헤더의 오버헤드를 압축하여 대역폭을 절약 할 수 있는 헤더 압축 기술이 요구되고 있다. SIP프로토콜 압축 방법에는 Deflate메시지 압축 방법을 이용한 SigComp(Signalling compression)[5] 알고리즘을 이용한다. IP/UDP/RTP 압축방법에는 대표적으로 Header removal/Generation, Header Striping/Regeneration, 헤더 압축[6] 등이 있으며, 그 중 헤더 압축 방법이 가장 효율적이다. 헤더 압축 연구는 유선 망 환경에서는 TCP/IP프로토콜 헤더 압축[7,8], RTP/UDP/IP 헤더 압축[9]이 연구되었으며, 무선망에서는 ROHC(RObust Header Compression)[10] 헤더 압축 기법이 연구되고 있다. 본 논문에서는 IMS망 PTT서비스의 성능 향상을 위해 통화 설정 대기 시간 단축 및 대역폭 확보를 위한 프로토콜 압축 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장 관련연구에서는 프로토콜 압축 기법과 다양한 호 처리 및 통화 성능 향상 기법 등을 기술하고, 제 3장에서는 IMS망에서 PTT서비스를 위한 호 설정 및 통화 성능 향상을 위한 새로운 프로토콜 압축 기법 방안을 제시하였다. 4장에서 성능평가를 하고, 마지막으로 결론을 장에 기술하였다.

2. 관련연구

2.1 프로토콜 헤더 압축

2.1.1 기존 프로토콜의 문제점

기존의 프로토콜 헤더의 크기는 IP 20byte, TCP 20byte, UDP 8byte, RTP 12byte이다. IP/UDP/RTP헤더의 크기를 제외하면 전송되는 데이터 및 패딩의 크기가 헤더에 비해 작다. 이와 같은 헤더들은 고정적인 값을 가지는 필드, 일정한 규칙에 맞게 변하는 필드, 몇몇 필드를 통하여 추론할 수 있는 필드를 포함하고 있다. FTP와 같이 헤더의 비해 데이터가 큰 경우에는 문제가 되지 않지만, 대부분의 실시간 서비스에서와 같이 헤더에 비해 데이터가 작은 서비스의 큰 오버헤드는 네트워크 효율을 저하 시키는 원인이 된다. 이와 같은 데이터에 비하여 헤더가 큰 경우, 헤더 압축 기술을 이용하면 많은 오버헤드를 줄일 수 있으며, 네트워크 대역폭 확보에 큰 도움이 된다.

2.1.2 프로토콜 헤더 압축 기법

헤더 압축 프로토콜은 RFC1144의 Van Jacobson 헤더 압축(VJHC : Van Jacobson Header Compression)을 기본으로 출발 하였다. 이 프로토콜은 패킷스트림에 그 개념을 두고 있다, 동일한 세션 내에 존재하는 헤더 영역의 대부분은 동일한 값을 갖게 된다. 예를 들어, IP나 UDP/TCP/RTP 헤더 내에 존재하는 주소 영역과 포트 영역은 같은 세션 내에서는 동일하다. 또한, RTP의 Timestamp나 Sequence number와 같은 영역은 값이 순차적으로 일정한 크기를 가지고 변하며, UDP 길이 영역은 링크계층의 영역과 중복된다. 이러한 패턴 분석을 기반으로 IP/UDP/RTP 헤더영역의 중복되는 부분은 제거하고 일정하게 증가하는 영역의 값에 대해서는 변화량만 전송하면 헤더압축이 가능하다는 것이 현재 RFC 2508로 승인되어있다.

2.2 호 처리 향상 기법

2.2.1 Always On

Always On은 단말의 Online 데이터 착신을 위하여 활성화(Active or Dormant) 상태가 영원히 지속 되도록 하는 기능이다[11]. 활성화 상태가 영원히 지속되면 통화 접속 대기 시간을 단축시키는데 효율적이다. 이러한 단말 상태는 표 1과 같이 3가지로 구분할 수 있다.

구현 방안으로는 두 가지 방법이 제안되었다. 첫 번째 방법은 단말의 패킷 데이터 서비스망(PDSN : Packet Data Switch Network) 사이의 세션 종료 타이머(SCT : Session Closure Timer)를 조정하는 방

표 1. 단말의 상태 구분

단말 상태	IP 주소 할당 여부	트래픽 채널 할당 여부
Idle	×	×
Dormant	○	×
Active	○	○

법이다. 세션 종료 타이머는 단말이 패킷 데이터 서비스망과 세션을 유지하고 있는 상황에서 무선 환경의 저하 및 단말이 PTT 서비스를 지속 할 수 없을 경우 세션 자원의 효율성을 위해서 일정 시간동안 응답이 없으면 강제로 세션을 종료시키는 역할을 한다. 이때 단말의 Active 상태를 유지하기 위하여 세션 종료 타이머 시점을 무한대로 설정하는 것이다. 두 번째 방법은 세션 종료 타이머는 기존 그대로 유지하며, 타이머가 종료되기 전에 keep alive 메시지를 전송하여 세션을 유지하는 방법이 있다.

그림 3은 위의 두 가지 방안을 절충하여서 Always On 기능 동작 시 단말과 기지국(BS : Base Station), PDSN의 상태 천이를 나타내었다. 현재 국내의 상용망인 1x EV-DO(Evolution Data Only)에서 사용 중인 세션 종료 타이머는 600초이며, 이 시간을 적용할 시에는 단말과 PDSN의 타이머 불일치로 인하여 상호간의 상태가 불일치 될 수 있다.[12] PTT에서는 이러한 점을 보완하여 세션 종료 타이머를 12배가량 늘린 7200초를 사용하여 첫 번째 제시한 방법 중 하나인 무한대의 효과를 가져 올 수 있다. 또한, 세션 종료 타이머가 종료 될 시에 PDSN에서 SDB(Short Data Burst)[13] 기능을 사용한 keepalive(LCP ECHO Request /Response)메시지를 전송하여 두 번

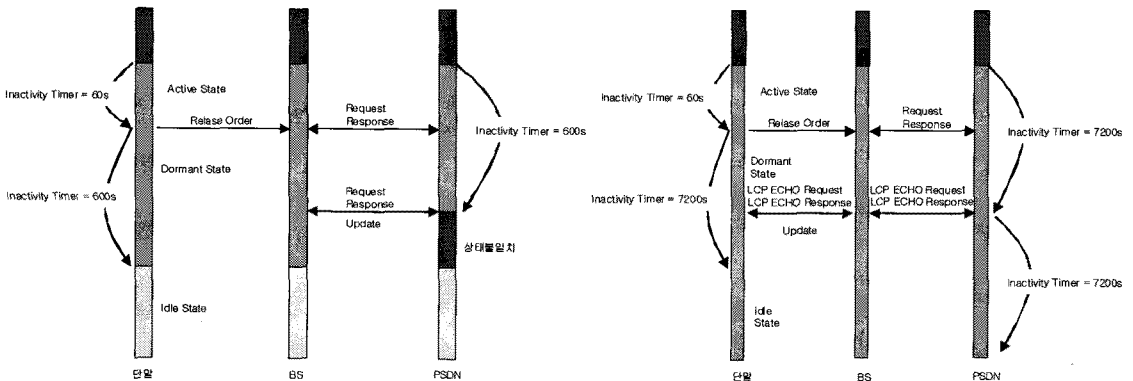


그림 3. 1xEV-DO 망과 Always On 상태 천이도 비교

째 제시한 방법인 타이머를 갱신함으로써 세션을 유지할 수 있다.

2.2.2 Dormant 페이징과 Multiple Access 채널[12]

Dormant 페이징 방법은 단말이 무선접속 시 액세스 채널을 통하여 통화를 시도하게 되는데 액세스 채널보다는 트래픽 채널을 사용하는 것으로써 SIP Signaling(INVITE / 200 OK)을 트래픽 채널로 전송하는 방법을 제안한다.

트래픽 채널 사용에 따라 자연히 액세스 채널의 사용은 감소하게 되고, 무선 접속 충돌(Access Collision)에 의한 재전송(Re-transmission) 발생은 크게 줄어들 수 있다. 또한 시스템 내부에서 발생하는 PPP Layer의 Encapsulation으로 인한 SDB (Short Data Burst) 사이즈의 증가로 인한 SDB 처리 불가 및 PCF(Packet Control Function)에서 버퍼링되는 메시지, 확인 메시지 지연에 따라 버퍼링되는 메시지의 페이징에 의한 지연현상을 대부분 줄일 수 있게 된다. 이러한 효과로 착신 가입자의 입장에서는 세션 참여시간이 짧아지며, 편차도 크게 줄어들게 된다.

Multiple 액세스 채널은 기존의 한 개로 운영되는 액세스 채널수를 여러 개로 늘려주어 다수의 액세스 채널을 사용함으로써 무선 접속 충돌을 방지 할 수 있는 방법이다. Multiple 액세스 채널을 사용한다는 개념 자체는 역방향 링크(Reverse Link)에 간섭(Interference)을 증가시키지는 않는다. 다수의 액세스 채널이 존재하는 경우 단말은 Hashing 함수를 사용하여 특정 액세스 채널을 사용하게 되는데, 이 경우 액세스 채널이 수용하게 되는 단말의 수가 분산되는 효과를 발생하므로 역방향 링크 무선 환경에 따른 액세스 충돌에 의해 재전송하는 단말의 액세스 Probe Sequence 횟수가 줄어들어 결과적으로 역방향 링크 상의 간섭 발생량이 줄어드는 효과를 가져다준다.

Multiple 액세스 채널을 사용함으로써 기존 서비스에 미치는 악영향은 없으나, Hardware적으로 한정된 트래픽 채널 자원을 액세스 채널로 용도 변경함에 따라 통화량 증가에 따른 트래픽 채널의 자원 부족 현상이 발생할 수 있다. 따라서 액세스 채널수를 늘리는 것은 해당 기지국의 통화 용량을 검토한 후에 적용을 하는 것이 바람직하다.

2.2.3 페이징 부하 개선[12]

기존 PTT 호 설정 과정에 이용하는 GAN(Global

Action Network)단위의 SDB를 전송하는 방법은 호 설정 시간을 최대한 줄이기 위해 고안되었으나, 반면에 페이징 부하를 증가시키는 문제가 발생한다. CDMA 2000 1X 망의 경우 GAN 단위의 SDB (INVITE) 페이징이 발생하므로 전체 시스템이 미치는 부하의 영향은 상당하다. 이에 아래와 같은 4가지 방안을 제시한다.

첫 번째 방안은 GAN 단위의 GPM(General Page Message) 전송 방법으로써 SDB를 이용하여 INVITE 메시지를 전송하는 방법이다. INVITE 메시지를 수신한 측에서는 BS/PCF가 GAN 단위의 GPM을 전송하고 이에 응답하는 특정 Cell/Sector로 INVITE 메시지를 전송하여 Page 채널의 부하를 줄이는 방법이다. 그림 4는 첫 번째 방안을 나타내었다.

두 번째 방안은 GAN 단위로 GPM 메시지를 전송하는 방법인데 트래픽 채널을 이용하여 INVITE 메시지를 전송한다. INVITE 메시지를 수신한 착신 측 BS/PCF가 GAN단위의 GPM을 전송하고 이에 응답하는 특정 Cell/Sector의 MS(Mobile Station)와 트래픽 채널 설정 후, 트래픽 채널을 통해 INVITE 메시지를 전송하여 페이징 채널의 부하를 줄이는 방법이다. 그림 5는 두 번째 방안에 대한 호 처리 절차를 도식화 하였다.

세 번째 방안은 BSC(Base Station Controller) 단위로 GPM 메시지를 전송하는 방법인데 INVITE 메시지를 수신한 측 BS(Base Station)/PCF(Packet

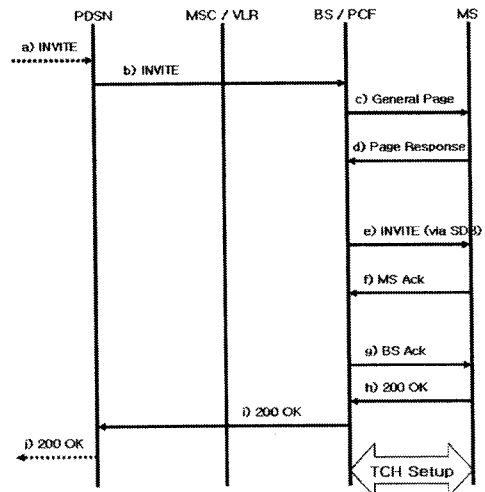


그림 4. GAN 단위의 GPM 전송방법 - TCH(Traffic Channel) 이용 INVITE 전송

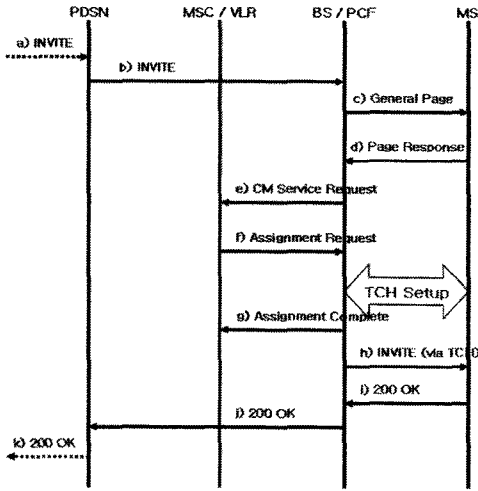


그림 5. GAN 단위의 GPM 전송 방법 - TCH이용 INVITE 전송

Control Function)가 이를 MSC(Mobile Switching Center)/VLR(Visitor Local Registry)로 전달하고, MSC/VLR의 착신 단말의 위치정보에 따라 특정 BSC 단위의 페이징을 수행한 후 이에 응답하는 특정 Cell/Sector로 INVITE 메시지를 전송하여 페이징 채널의 부하를 줄이는 방법이다.

마지막으로 BSC 단위의 SDB 전송 방법이 있다. 이것은 INVITE 메시지를 수신한 착신 측 BS/PCF가 이를 MSC/VLR로 전달하고 MSC/VLR의 착신 단말의 위치정보에 따라 특정 BSC 단위로 INVITE 메시지를 전송함으로써 페이징 채널의 부하를 줄이는 방법이다.

2.2.4 단말기의 버퍼 추가[14]

PoC 서버와 단말기 사이의 호 설정 시간 지연의 문제점은 무선 환경이나 호 설정 메시지 문제가 아닌 네트워크 전송 환경에 보다 많이 영향을 받고 있다는 것을 알 수 있다. 따라서 네트워크상에서 데이터 전송 중에 발생하는 호 설정 지연 문제 해결을 단말기 상에 버퍼를 추가하여 호 연결 시간을 단축시키는 방법을 제시한다.

호 설정 초기화 및 종료, Floor Control 절차에서 단말기가 PoC 서버로부터 200 OK를 받은 후 호 설정되어 Vocoder 측에서 음성 데이터를 PoC 서버를 통해서 상대방에게 전달하게 되는 방법이다. 이러한 기존의 방법을 기지국과 데이터 채널이 연결 된 후

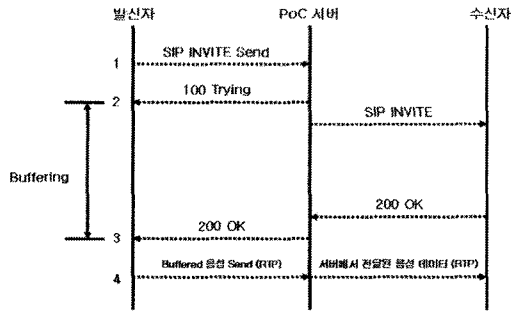


그림 6. 단말기 음성 버퍼 메시지 흐름

INVITE 메시지를 보내는 시점에 단말기 내부에서 호 연결 상태로 인식하여 Vocoder 음성 데이터를 임시 버퍼 상에 저장한 후 PoC 서버로부터 200 OK를 받는 순간 RTP 패킷에 단말기 임시 버퍼에 저장된 음성 데이터를 발신하는 방법이다. 이 방법은 이론상으로 단말기 상에서 RTP 한 패킷은 6개의 프레임으로 구성되어 있기에 약 120ms 정도의 음성 데이터 전송 시간을 줄일 수 있다. 그러나 INVITE 메시지를 발신 후 발신 측에서 음성 데이터를 버퍼에 저장하기 때문에 보다 더 많은 시간이 단축되는 것으로 사용자에게 인식된다. 그림 6은 단말기 음성 Buffering 메시지 흐름이다.

1번 INVITE 메시지를 보낸 시점에서 3번 200 OK를 받은 시점 사이에 Vocoder 단에서 마이크를 통하여 들어온 음성 데이터를 20ms 프레임 내부 음성 버퍼에 저장한다는 원리이다[15]. 단말기 내부 동작은 INVITE 메시지를 발신하는 시점과 동시에 Vocoder에서는 마이크 단에 입력된 음성 데이터를 20ms 프레임 단위로 Encoding 한 후 버퍼로 전달하며, PoC 서버 측에서 200 OK를 받은 즉시 이미 단말기 버퍼에 저장된 음성 데이터를 120ms RTP 패킷 데이터에 실어서 PoC서버로 발신하도록 구현하는 것이다 [16,17].

2.3 통화 성능 향상 기법

채널의 효율성을 높이기 위하여 입력 음성 신호로부터 음성구간을 구분하고 일정시간 이상 목음이 지속되는 구간에서는 패킷을 전송하지 않고 일정한 시간 간격으로 패킷을 전송하는 방법이 있다. H.323 프로토콜을 이용하여 개발한 OpenPhone의 경우 G.711 코덱을 사용하면 초당 64Kbit를 보내게 된다. 사용자

가 말을 하지 않을 경우도 초당 64Kbit를 보내게 되어 불필요한 패킷 낭비 및 통화 대역폭의 낭비가 우려된다. 이러한 문제점을 해결하기 위해서 묵음 구간 신호를 효율적으로 코딩하는 기법이 필요하다.

음성 코덱을 사용하는 주된 이유는 음성 신호를 전송 또는, 저장하는데 필요한 비트 수를 낮추는데 있으며, 시스템의 특성에 따라 가변 Bit Rate 또는, 고정 Bit Rate 음성 코덱을 사용한다. 고정 Bit Rate 코덱에서는 입력 신호의 특성에 상관없이 항상 같은 Bit Rate로 코딩하고, 가변 Bit Rate 코덱에서는 입력 신호의 특성에 따라 다른 Bit Rate로 코딩한다. 멀티미디어 서비스를 위해 채널을 다른 유형의 데이터와 공유할 경우에는 가변 Bit Rate 코덱을 사용하는 것이 효율적이다. 양방향 통화중인 경우에 입력 신호를 분석해 보면 약 60% 이상이 말을 하고 있지 않는 묵음 구간이다. 이 묵음 구간에는 정보가 거의 없으므로 적은 Bit Rate로 표현 가능하며, 이로 인해 가변 전송률 음성 코덱을 사용할 경우에는 거의 음질의 저하 없이 평균 전송률을 낮출 수 있다.

묵음 구간 신호를 효율적으로 코딩하기 위해서는 두 가지 알고리즘이 필요하다. 첫 번째는 입력 신호를 음성과 묵음 구간으로 분류하는 VAD(Voice Activity Detection) 알고리즘이다. 만일 입력 신호가 묵음 구간으로 분류되면 약 1Kbit/s의 낮은 Bit Rate로 압축하여 전송하며 DTX(Discontinuous Transmission) 기법을 이용하여 전송을 하지 않는 경우도 있다. DTX 기법이란 이동 전화나 휴대 무선 전화기에 음성 신호가 없을 때 순간적으로 출력을 줄이거나 묵음 상태로 만드는 방법이다. 두 사람이 통화를 할 때 각자의 통화시간은 각각 전체의 절반 이하에 해당하므로 만약, 송신기에 음성 입력 시간만 연결된다면 송신 시간이 50% 이하가 될 수도 있다. 이는 전지 소모 절약, 송신기 증폭기의 부하 감소, 시분할 다중화(TDM : Time Division Multiplexing)에서 다른 신호와 채널 공유 등의 이점을 살릴 수 있다. DTX 회로는 음성 구동과 검파(VAD : Voice Activity Decetion)를 이용해 동작하며, 복스(VOX : Voice Operated Transmitting)라고 부른다. 두 번째는 수신 단에서 사람의 귀에 편안한 잡음을 재생하는 CNG(Comfortable Noise Generation) 알고리즘이다. GSM에서는 MS(Mobile Station)에서 전력 소모를 줄이고 전체적인 간섭을 줄이기 위한 목적으로

묵음 구간을 검출하여 일정 시간마다 묵음 신호에 대한 정보를 전송하기 위해 VAD/DTX/CNG 알고리즘을 사용한다. 또한 평균 Bit Rate를 낮출수록 더 많은 가입자를 수용할 수 있는 CDMA에서는 시스템의 특성상 DTX는 지원하지 않고 묵음 구간을 약 1Kbit/s로 압축하여 전송한다. 이와 같은 방법은 무선 주파수 자원 절감에도 효과가 있으며 기존 대비 2/3이상 오버헤드를 줄일 수 있다[18,19].

3. 호 설정 및 통화 성능 향상을 위한 프로토콜 압축 기법

IMS망에서 PTT서비스를 상용화 할 경우 두 가지 사항을 고려해야 한다. 첫 째, 현재 상용화 되고 있는 기존의 PTT서비스인 TRS망에서의 동일한 호 설정 시간을 가져야 한다. 둘째, 통화 연결이 된 후 통화 품질 또한 기존 서비스와 동일해야 하며 # of User에 대한 충분한 트래픽 채널이 할당 되어야 한다. 기존 연구에서는 호 설정 향상을 위한 SDB 메시지[13], Always On 기능 활성화[11], 페이징 부하 개선 방안 [12] 등을 통하여 해결하였다. 본 논문에서는 하드웨어 문제나 네트워크 트래픽 채널에서의 호 설정 문제를 해결하기 전에 멀티미디어 프로토콜인 SIP에 문제가 있음을 발견하였다. IMS망에서 PTT서비스를 할 때 호 설정에는 SIP가 사용되며, 호 설정이 완료된 후 통화 시에는 RTP가 사용된다. 두 프로토콜은 멀티미디어 통신을 위해 고안된 프로토콜이며[20] 대역폭이 충분히 확보된 상태에서 텍스트 기반으로 되어있다. 많은 멀티미디어 통신 프로토콜(SIP, RTP, H.323)은 그 결과 프로토콜 메시지는 충분히 최적화 되어 있지 않는 상태이다. 예를 들어 SIP의 경우를 보면 메시지 길이는 몇 백 바이트에서 2000바이트까지 이거나 혹은 그 이상의 길이를 가지고 있다.[20] RTP의 경우는 IP(20) + UDP(8) + RTP(12) + Payload Data(24 : G.723.1)로써 헤더의 길이가 음성 데이터의 길이보다 16바이트 더 길다. 멀티미디어 프로토콜은 대역폭이 풍부하지 못한 2.5G, 3G, IMS와 같은 무선 이동 통신 네트워크에서 사용 할 경우 유선망에서 발생하지 않았던 문제들이 나타난다. 무선망에서는 일반 유선망과는 달리 높은 에러 전송률을 나타내는데 헤더 부분의 에러는 치명적인 결과를 낼 수 있어, 효율적인 헤더 처리 기술이 필요하다.

또한 SIP와 같은 큰 메시지로 인하여, 무선망에서 통신하기 위한 멀티미디어 프로토콜로써는 최적의 솔루션이 될 수 없다.

본 논문에서는 헤더가 긴 멀티미디어 프로토콜인 SIP를 압축하여 문제를 해결하고자 한다. 호 설정 압축 프로토콜은 RFC3320에 명시된 SigComp(Signalling Compression)[5] 기법을 향상 시킨 A+SigComp(Advanced Signalling Compression) 기법을 이용하고, IP/UDP/RTP의 경우는 RFC3095에 명시된 ROHC(RObust Header Compression)[10] 기법을 향상 시킨 A+ROHC(Advanced RObust Header Compression)기법을 이용하여 프로토콜을 압축한다.

3.1 A+SigComp : 호 설정 향상을 위한 프로토콜 압축 기법

IETF(Internet Engineering Task Force)의 SIP(Session Initiation Protocol)는 HTTP(HyperText Transfer Protocol)와 유사한 텍스트 기반의 메시지를 사용함으로써 이를 구성하는 헤더의 확장이 용이하고 간단하게 세션을 설정하고 수정 및 종료 할 수 있기 때문에 인터넷 응용 분야로의 적용이 가능한 장점이 있다. 확장이 용이하고 세션 설정이 간단한 장점으로 인하여 다수의 표준화 단체들이 차세대 호 설정 프로토콜로써 SIP를 채택함에 따라 SIP를 기반으로 한 많은 응용 연구가 진행되고 있다.

초고속 무선 네트워크를 통해 멀티미디어를 제작, 전달 및 재생을 논의하는 표준화 단체인 3GPP(3rd Generation Partnership Project)와 동기식 IMT-2000 이동통신 규격을 의하는 단체로 최근 멀티미디어 서비스를 위한 규격들을 제정하고 있는 3GPP2(3rd Generation Partnership Project 2)의 경우, 멀티미디어 서비스를 위한 표준 프로토콜로 SIP를 채택하면서, 멀티미디어 서비스, 팩스, 홈 네트워킹, 인스턴트 메시징, 망 관리, 웹 및 SMTP(Simple Mail Transport Protocol) 등의 응용 서비스들이 SIP를 기반으로 통합되는 구조를 보이고 있다[21].

SIP는 대역폭이 풍부한 네트워크에서의 사용을 전제로 개발되었고, SIP메시지들은 텍스트 기반이어서 크기가 매우 크다. SIP를 대역폭이 풍부하지 못한 3G와 같은 무선 이동 통신 네트워크에서 사용할 경우 유선망에서 발생하지 않았던 문제들이 나타나고

있다. 그 중 가장 큰 문제점은 주파수 사용의 낭비를 들 수 있다. 풍부하지 못한 대역폭에서 텍스트 기반의 큰 메시지는 다른 통신에 지장을 주게 된다. 그 문제점을 해결하기 위해 RFC3320에 명시된 SigComp(Signalling Compression)을 이용하여 SIP 메시지를 압축하는 방법이 있다.

SIP는 응용 계층에서 SIP, SDP스택을 포함한 메시지를 TCP/UDP로 전송한다. SigComp압축 알고리즘은 응용 계층(SIP, SDP)을 TCP/UDP에 포장하기 전에 Deflate 압축 알고리즘을 이용하여 압축 한 후, TCP/UDP계층으로 보내게 되며, IP/TCP/UDP 단에서는 다른 헤더 압축 알고리즘을 이용하여 압축 한 뒤 패킷을 전송하게 된다.

그림 7은 SigComp Interface 모델을 나타낸다. SIP는 응용 계층에서 데이터가 만들어지고, 만들어진 데이터는 전송 계층(TCP or UDP 캡슐화)으로 내려가게 된다. 응용 계층에서 만들어진 SIP메시지가 전송계층으로 내려가는 중간 단계에 SigComp 알고리즘을 이용하여 SIP메시지를 압축하게 된다. SigComp Stack에는 Compressor Dispatcher, De-Compressor Dispatcher, State Handler, UDVM(Universal Decompressor Virtual Machine)등으로 구성된다.

SigComp는 OpenSigComp[21]에서 0.9.3 Version 까지 Released 되었으며, 압축 알고리즘은 Deflate Algorithm[22]을 사용하였다. Deflate Algorithm은 LZ77 Algorithm을 이용하여 압축하며, 중복되는 내용에 대한 포인터를 허프만 부호화를 이용하여 한번 더 압축한다. Deflate Algorithm은 압축률에 비해

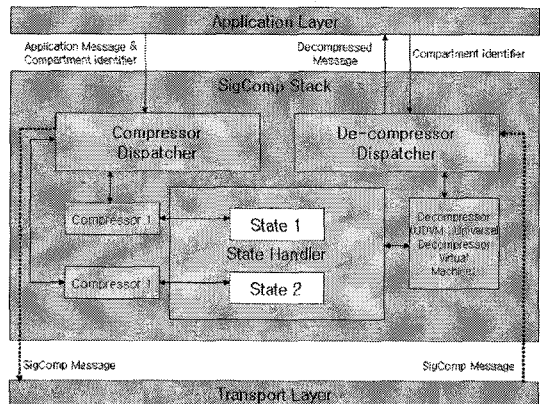


그림 7. SigComp Interface 모델

압축과 해제 속도가 빠른 것으로 알려져 있다[22]. SIP는 호 설정을 위한 패킷으로 속도가 빨라야 하므로, 압축률 보다는 압축속도에 비중을 둔 Deflate Algorithm을 이용하였다.

A+SigComp 알고리즘은 기존의 SigComp의 두 가지 문제점을 해결하기 위하여 고안되었다. 첫 번째는 호 설정 시 SIP메시지를 압축함으로써 생기는 프로세스 처리 시간이다. 두 번째는 기존 SigComp Stack에서 ROHC압축 기법으로 인하여 메시지 처리 속도가 느린 단점이 있다.

SigComp에서 Stack은 응용계층에서 만들어 진 메시지를 전송 계층에 보내기 전에 State에 SIP/SDP메시지 형태를 저장하게 된다. 저장을 하는 이유는 다음 Compressor와 Decompressor에서 빠른 메시지 복원을 위해서이다. 따라서 초반에 새로운 메시지 형태를 가진 경우 메시지 형태를 State에 저장해야 하는 과정으로 인하여 처리 속도가 감소된다. 초기 메시지의 State저장 과정에서 빠른 처리를 위하여 SIP Directory를 추가하여 방법을 해결 하고자 한다.

SIP Directory는 RFC3485[23]에 명시되어 있으며, 잘 알려진 String 을 Static으로 Directory 화 한 코드이다. 그림 8은 SIP의 메시지 구조를 보여준다. SIP는 Start Line, SIP Header, SDP(Session Description Protocol) 헤더로 구성되어 있다. Start Line은 INVITE, 200 OK, TRYING, RINGING 등, SIP헤더의 From, To, Call-ID 등과, SDP 헤더의 session, IP4등 같이 이미 Static한 String이 존재한다. 그리고 메시지를 송/수신시 자주 사용되는 String이 존재하게 된다. Static한 String과 자주 사용되는 String을 이용하여 SIP Directory를 만들게 된다. 자주 사용되는 String에 대해서는 우선권을 높게 주고, 자주 사용되지 않는 String에 대해서는 낮은 우선권을 부여한다. 따라서 메시지 복원 시 SIP

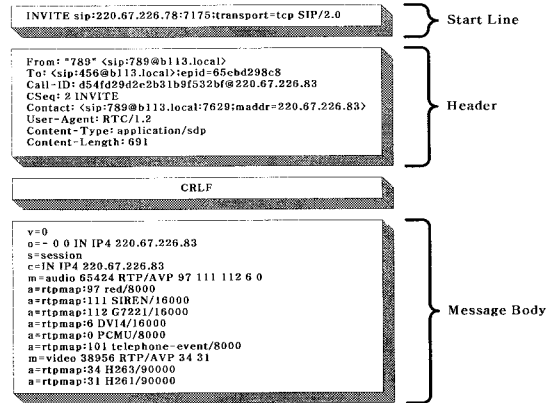


그림 8. SIP 메시지의 구조

Directory를 우선순위로 검색한 후 offset값과 String 길이를 이용하여 메시지를 받은 후, 복구 String에 포함시켜 빠른 복구를 구현 할 수 있다.

두 번째는 기존 SigComp 프로토콜 Stack에서 포함된 ROHC 압축 스택 부분을 제외함으로써 메시지 처리 속도를 빠르게 하는 방법이다. 또한, 기존의 SIP 경우는 TCP통신을 사용 하지만, PTT 서비스를 위하여 UDP스택을 사용하는 방법이다. 그림 9는 기존의 SigComp 스택[24]과 확장된 A+SigComp 스택을 보여준다.

SIP 메시지 압축 방법은 헤더 압축 방법과는 다른 압축 알고리즘을 이용한 압축 방법이다. IP/UDP/RTP 헤더 압축 방법은 STATIC필드와 DYNAMIC 필드를 보낸 후 다음 패킷은 이전 패킷과의 차이를 전송하는 방법을 사용하였다. 하지만 SIP 메시지 압축의 경우 String 메시지가기 때문에 String을 압축할 수 있는 Deflate 알고리즘을 이용하여 메시지를 압축하였다. 따라서 SigComp 압축기법과 IP/UDP/RTP 헤더 압축과는 다른 방식을 사용하기 때문에 이중 압축을 이용할 경우 빠른 통화설정을 위해서는

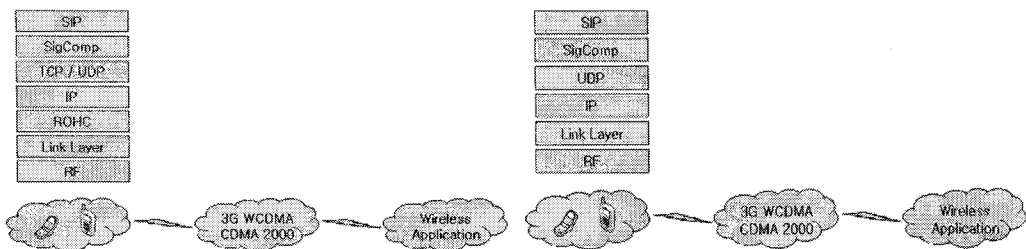


그림 9. 기존 SigComp와 A+SigComp Stack

속도가 느린 단점을 가지고 있다. SIP 메시지의 크기가 크기 때문에 압축 한 결과 50~100byte의 크기를 가지고 있기 때문에 IP/UDP 헤더 크기인 28byte에 비하여 큰 크기를 가지고 있다. IP/UDP헤더를 압축할 경우 최대 2byte까지 압축되지만 전체 데이터 크기를 비교해 볼 때 약 26byte의 차이를 가지고 있기 때문에 압축된 SIP 메시지 크기에 비해서는 큰 차이를 가지고 있지 않다. 따라서 기존의 IP/UDP 헤더를 ROHC 기법을 이용하지 않고 원본 데이터를 보낼 경우 처리속도는 더 빨라 질 것이다.

기존의 SIP의 경우 TCP헤더에 메시지를 실어 보냈었다. PTT서비스의 경우는 신뢰성이 있는 프로토콜인 TCP를 쓰지 않고, UDP를 사용하여도 IMS망에서 오류처리를 하기 때문에 UDP 헤더를 이용하여도 문제가 되지 않는다. 또한, INVITE 메시지 이후 200 OK 응답 메시지를 받기 때문에 TCP를 쓰지 않아도 오류처리가 가능하다. UDP헤더를 사용함으로써 TCP헤더의 복잡한 헤더 필드를 처리 하지 않아도 되며, 8byte의 헤더 크기를 줄일 수 있다

3.2 A+ROHC : 통화 성능 향상을 위한 프로토콜 압축 기법

ROHC(RObust Header Compression)는 기존의 헤더 압축 방법들과는 다르게 높은 비트 에러율과 긴 왕복소요시간이 있는 무선망에서 효율적이고 강인한 헤더 압축을 위한 방법이다. 기존의 헤더 압축 방법들은 IP, TCP, UDP 헤더를 STATIC, DYNAMIC, INFERRABLE 클래스로 분류하여 압축하는 방식을 사용하였다. 압축한 패킷을 송/수신 시 네트워크에 따라 피드백의 유무나, 신뢰성, 최적화에 따르지 않고 보내기 때문에 네트워크 대역폭이 풍부한 유선망에 적합한 프로토콜 압축 방법이다. ROHC 경우는 압축 상태와 압축 모드라는 개념을 도입하여 네트워크 상태 및 피드백의 유무에 따라 압축방법이 변하게 된다. 따라서 가장 낮은 압축 상태에서 가장 높은 압축 상태로 변하게 되며, 패킷 헤더들 변화,

Decompressor부터의 ACK/NACK, 주기적인 Timeout 등의 요인에 의해 압축 상태가 변하게 된다.

RFC1889에 의하면 RTP패킷은 압축되지 않은 상태에서 Ethernet Header(14byte), IP Header(20byte), UDP Header(8byte), 그리고 RTP Header(12byte)를 합쳐서 54byte의 헤더 크기를 가진다. 하지만 ROHC 알고리즘을 적용하여 RTP패킷을 압축할 경우 최대 Ethernet Header(14byte)와 ROHC(8byte) 즉, 22byte만을 가지면 되기에 약 60% 압축률을 보여준다. 만일 Payload Data를 더하여 계산할 경우 약 41% 가량의 크기를 절약 할 수 있다.

ROHC 알고리즘은 Compressor상에서 IR(Initialization and Refresh), FO(First Order), SO(Second Order)등의 3가지 상태를 Decompressor상에서 NC(No Context), SC(Static Context), FC(Full Context)등의 3가지 상태를 가지고 동작한다.

그림 10은 Compressor에서의 상태 전이를 나타낸다. Compressor에서는 가장 낮은 압축 상태인 IR에서 시작하여, 전체 헤더와 CID를 전송한다. Decompressor에서 ACK를 받을 경우 Compressor는 FO 상태로 전이 한 후에 STATIC Field 및 DYNAMIC Field를 제외한 후, CID와 함께 패킷을 전송한다. Decompressor에서 ACK를 받으면 Compressor에서는 SO상태로 전이되며, STATIC Field와 DYNAMIC Field, 추론 가능한 Field를 제외한 CID와 그 이외의 Field를 포함하여 전송한다.

그림 11은 Decompressor에서의 상태 전이를 나타낸다. Decompressor에서는 전체 헤더를 받은 NC 상태에서 STATIC Fields, CID or STATIC Fields, DYNAMIC Field, CID를 저장한다. 성공적으로 저장한 뒤에 FC 상태로 전이한다. 다음 패킷이 CID, Irregular Fields를 받은 경우 Success 메시지와 함께 FC상태를 유지하며, Decompression이 실패 할 경우 NACK패킷을 Compressor에서 보내며, SC 상태로 변하여 Compressor에게 DYNAMIC Field를 요구하게 된다. Compressor가 CID, DYNAMIC Field를 전

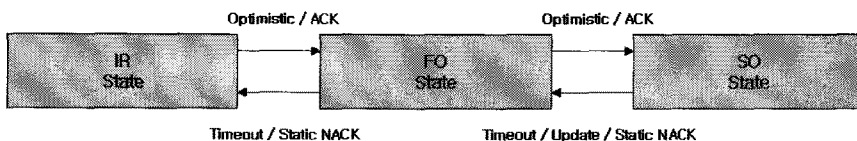


그림 10. Compressor 상의 상태 전이

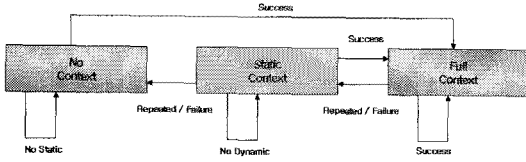


그림 11. Decompressor에서 상태 전이

송 할 경우 다시 FC상태로 전이하게 된다.

Compressor와 Decompressor 사이에는 3가지 상태로 변하게 되며, 작동 할 수 있는 3가지 모드가 있다. U-Mode(Unidirectional Mode)는 패킷이 오직 한 방향(Compressor에서 Decompressor)으로 전송되며, 주기적인 Timeout에 기초를 두고 있다. 시작 에러 복구를 위한 주기적인 Refresh와 Feedback이 없기 때문에 압축은 비효율적이다. ROHC 알고리즘은 U-Mode로 시작한다. O-Mode(Bi-directional Optimistic Mode)는 U-Mode와 유사하지만 Feedback 채널이 Send error Recovery request와 Decompressor로부터 Compressor로 중요한 context Update ACK로 사용된다는 것이다. 주기적인 Refresh들은 사용되지 않지만 적당한 견고성을 유지하는 동안에 최적의 압축 효율과 적은 수의 Feedback 채널 사용을 위해 설계되었다. R-Mode(Bi-directional Reliable Mode)가 가지는 앞의 두 가지 모드와의 구현과 기능에 관한 가장 큰 차이점은 Feedback 채널의 과다한 사용이다. Compressor와 Decompressor사이에 모든 context Update에 ACK를 보내게 된다. 따라서 패킷 손실에 대한 부분은 완벽한 견고성으로 context 손실을 최소화 한다.

그림 12는 ROHC 초기화 과정 및 동작 절차를 나타낸다. Compressor는 ROHC 알고리즘에 의해 hcrl에 CID, STATIC, DYNAMIC 헤더 Fields와 초기화 Field를 보낸다. Decompressor는 이 패킷을 수신한 후, FC State 전이 한 후 ACK 신호를 응답한다. De-Compressor는 자신이 O-Mode로 간다는 것을 표시한다. Compressor는 ACK를 수신한 뒤 O-Mode로 전이 한 후, Compressor는 SO State로 전이한다. 다음 패킷과 함께 Compressor는 헤더와 패킷 Payload로 구성된 CID를 보낸다. 그러나 Decompressor는 수신한 패킷 헤더를 Decompresses 하지 못하여 패킷을 폐기 한 후 SC State로 전이한다. 전이와 동시에 NACK 신호를 송신한 후 NACK를 수신한 Compressor는 FO State로 전이한다.

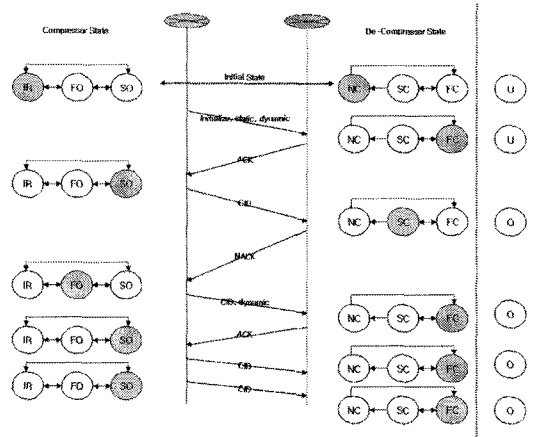


그림 12. ROHC 초기화 과정 및 동작 절차

Compressor는 다음 패킷에 CID, Dynamic 헤더 Field를 구성하여 보낸다. Decompressor가 Compression된 헤더를 Decompresses하는데 성공하고 FC State로 바꾼 후 ACK로 응답한다. Compressor는 ACK를 받은 후 SO State로 전이하고 다음 패킷 (CID, Payload Data)을 전송한다. ROHC는 O-Mode에서 Compressor는 매 패킷마다 각각의 ACK를 전송하지 않고, 기다리지도 않는다.

멀티미디어 전송에서 가장 중요한 프로토콜은 RTP이다. 특히 헤더에 포함된 Sequence Number, Timestamp는 음성 데이터의 순서와 첫 번째 옥테트가 샘플링 된 순간부터 순차적 증가를 가진 Field로써, 실시간 데이터에서 중요한 부분에 속한다. PTT 서비스의 경우 실시간 음성 데이터 전송으로써 패킷이 손실되거나 오류가 발생할 경우 ROHC 동작과정에 의해 압축 상태가 낮은 상태로 이동하게 된다. 이 동원 상태에서 현재 STATIC, DYNAMIC Field를 전송해야 하기 때문에 음성이 끊기는 현상이 발생하게 된다. 이와 같이 실시간성 데이터 전송 시 패킷의 손실이나 오류 등을 방지하기 위하여 고안된 A+ROHC(Advanced ROHC)기법을 제안한다.

$$Sequence\ Number = SEQR \times 7 + SEQ7$$

(SEQR : 7로 나눈 몫, SEQ7 : 7로 나눈 나머지)

Sequence Number는 RTP 패킷이 송/수신 될 때마다 1씩 증가한다. 수신측은 이 Field를 이용하여 패킷 손실을 감지하고 패킷 순서를 재 저장한다. 16bit를 가진 Sequence Number의 압축 과정은

Modulo-7 연산을 통하여 정수부분과 나머지 부분의 bit를 W-LSB Encoding을 이용하여 보내게 된다.

위와 같은 수식을 통하여 간략화 될 수 있으며, SEQ7은 압축헤더의 첫 3bit에 표시되어 전송된다. 이때 3bit를 패킷에 담아 보내기 전에 1개의 reordering을 처리하기 위해 사용하는 코드의 순서번호는 [-1, 5]범위를 다루고 있다. 이러한 기존의 방법을 확장시킨 방법으로는 SEQR, SEQ7부분을 고려하여 다중 패킷 손실에 대한 강인성을 제공할 수 있다.

$$S(n') = S(n'-1) - SEQR_{LSB}(n'-1) \times 7 - S(n'-1) \% 7 + SEQR_{LSB}(n') \times 7 + S(n') \% 7$$

$SEQR_{LSB}$: SEQR의 LSB
 $S(n)$: 현재 순서번호
 $S(n-1)$: 이전 순서번호

위와 같이 식을 이용할 경우 확장된 Header의 순서번호의 변화는 [-3, (7×2×SEQR_NB) - 4]를 가진다. SEQR_NB는 De-Compressor에 전송된 SEQR Bit 수이다. Wrap-around에서 패킷 손실이 발생하면 순서번호의 Decoding된 값이 잘못될 확률이 증가하는데, 이러한 현상을 피하기 위하여 동적 패킷이나 압축 패킷의 확장을 사용해서 순서번호를 코드화 하였다. Wrap-around란, 16bit 크기의 순서번호가 가질 수 있는 최대값인 65535에서 다시 0으로 넘어갈 때 SEQR_LBS의 값이 잘못 인식되는 것을 의미한다. 이와 같은 확장된 SN으로 인하여 다중 패킷에 대한 손실을 이전의 압축 방법보다 더 효율적으로 줄일 수 있다.

Timestamp는 RTP 패킷의 첫 번째 Octet이 샘플링 된 순간을 나타낸다. 그 샘플링 시점은 일정하게 증가하는 Clock으로부터 생성된다. 이것은 실시간 데이터의 동기화와 지터 계산에 이용된다. 초기 값은 Sequence Number와 동일하게 무작위 수로 설정된다. 16bit를 가진 Field인 Timestamp의 TS값 변화는 고정적이다. G.711 Codec, 64 Kbit/s, 30ms 경우 TS의 간격은 $64 \text{ Kbit/s} \div 8 \times 0.03 = 240\text{Byte}$ 라는 값이 나온다. 즉, G.711 Codec의 경우 TS 변화폭은 고정적으로 240Byte씩 증가하게 된다. 이와 같은 Timestamp Field값은 SN과 Payload Data Size에 의해서 충분히 유추될 수 있다. 안정된 SN을 통하여 손실을 줄인 RTP 패킷을 이용하는 SIP Module에서는 SN에 맞추어 Payload Data를 디코딩 한 후에 스

피커를 통하여 음성 데이터를 흘려보내게 된다. TS는 고정적인 값을 가지고 있기 때문에 기존의 5-LSB Bit 전송되던 Field를 생략할 수 있다. 단, 목음처리를 할 경우 SN은 +1간격으로 늘어나는 반면에 TS 간격은 × TS Interval 배수로 늘어나기 때문에 이에 대한 부분을 고려하여 1byte의 Field가 필요하게 된다. 1bit는 목음처리로 인해 갑자기 늘어난 TS의 유무를 판별하기 위한 Flag이며 7bit는 TS Interval 배수 Field이다. 따라서 첫 Bit가 1일 경우 TS는 다음과 같은 공식에 의해 계산된다.

$$TS = TSQ \times TSInterval + TSR$$

TS : Timestamp
 TSQ : Timestamp Quotient
 TSR : Timestamp Residual

4. 성능 평가

본 논문에서 제안한 A+SigComp 기법과 A+ROHC 기법의 성능 평가에 대하여 기술한다. A+SigComp 기법의 성능 평가는 PTT 단말기가 IMS 망에서 세션을 설정하는 단계에서 주고 받는 SIP 메시지의 종류별 크기를 측정하였으며 A+ROHC 기법의 성능 평가는 음성 통화시 교환되는 헤더의 크기를 측정하여 대역폭 증가량을 계산하였다.

성능 평가는 그림 13과 같은 환경을 가정하였으며 PTT Client, IMS 망의 Proxy 를 구성요소로 갖는 모의실험 환경을 구축하여 실시하였다. 모의실험을 위해 OpenSigComp[21]의 SigComp 프로그램을 Windows C++ 용으로 수정하여 사용하였다.

4.1 호 처리 향상 성능 평가

SIP 메시지 압축 크기를 확인하기 위하여 Open SigComp의 소스를 이용하여 압축전의 SIP메시지, SigComp 기법을 이용하여 압축된 메시지,

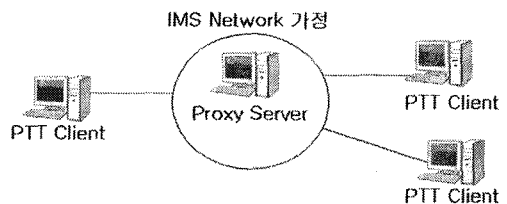


그림 13. 성능평가 환경

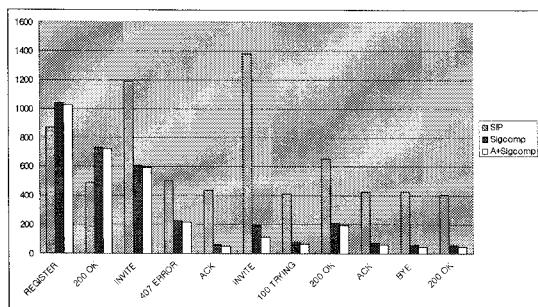


그림 14. 패킷 크기 비교

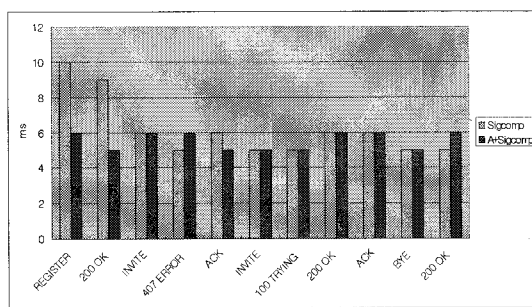


그림 15. 메시지 압축 처리 시간 비교

A+SigComp 기법을 이용하여 압축된 메시지 순으로 그림 14에 비교해 보았다.

최소 서버에 접근하여 등록 할 시의 메시지는 REGISTER, 200OK이다. 이 경우 SIP 메시지 크기에 비해 SigComp, A+SigComp 압축 기법을 이용한 메시지가 더 크을 알 수 있다. 이 경우는 최초 메시지 전송 시 Compressor와 Decompressor 사이에 현재 메시지 상태를 미리 state에 저장해 둬므로써 다음 압축률을 높이기 위해서이다. 그 이후 INVITE 메시지에서 압축률이 증가함을 볼 수 있다. 동일한 형태를 가진 메시지를 계속적으로 보낼수록 state에 저장된 내용을 검색하여 압축 및 복원 과정을 거치기 때문에 압축률이 높아짐을 알 수 있다. 위의 패킷 크기를 비교 해 본 결과 A+SigComp기법을 이용한 메시지가 더 압축률이 좋다는 것을 알 수 있다.

이번 실험에서는 메시지 압축 시 소요되는 프로세스 시간을 알아보았다. SIP 메시지 경우 압축하지 않은 상태에서 메시지를 보내기 때문에 프로세스 시간을 0으로 측정하였다. 압축 시간 측정은 한 PC에서 SIP메시지를 압축 한 후에 복원까지 걸리는 시간을 측정하였다. 그림 15는 SigComp와 A+SigComp의 메시지 압축처리 시간을 비교하였다.

REGISTER, 200K 메시지의 경우 약 10, 9ms 정도의 처리시간을 보인 SigComp 기법과는 달리 A+SigComp 기법의 경우는 SIP Directory를 사용함으로써 기존의 SigComp기법보다 약 4ms이상 단축된 것을 볼 수 있다. 초기 state 저장 과정 이후는 비슷한 처리 시간을 보여준다. 결과적으로 A+SigComp 기법은 세션 설정 단계에 지연시간 단축 효과가 크게 관측되었으며 이는 다수의 단말들과 세션 설정이 필요한 PTT 서비스에서 상당한 성능 향상이 이루어질 것이다.

4.2 통화 성능 향상 성능 평가

IP/UDP/RTP 패킷 오버헤드 측정 시 ROHC 기법에서 3가지 상태를 고려해야 한다. IR상태는 전체 헤더를 전송하며, FO상태는 부분적 동적헤더를 받은 상태이다. 가장 최적 압축률 상태인 SO상태는 1byte CID와 함께 이전 패킷과 현재 패킷간의 차이를 k-LSB인코딩을 통하여 헤더에 실어 보내게 된다. 따라서 ROHC기법과 A+ROHC기법에서 대상이 되는 압축 헤더 데이터 크기는 SO상태의 데이터 크기가 된다. IP/UDP/RTP 패킷 데이터 크기는 IP(20byte), UDP(8byte), RTP(12byte) 헤더 합인 40byte이다. SO 상태에서 ROHC 기법의 경우 IP(20bit), UDP(4bit), RTP(8bit) 헤더 합인 5byte이다. A+ROHC 기법의 경우는 IP(4bit), UDP(4bit), RTP(4bit) 헤더 합인 3byte이다. 압축 기법에는 CID의 1byte 크기의 헤더가 가장 앞에 붙게 된다. 모든 Payload Data의 경우 전송속도를 30ms로 제한하였으며, G.711의 경우 64Kbit/s로 240byte, G.729의 경우 8Kbit/s로 30byte, G.723.1의 경우 6.3Kbit/s로 24byte의 Payload Data를 가진다. 음성 데이터가 클수록 패킷 오버헤드 값이 감소하는 것을 볼 수 있다.

패킷 오버헤드는 대역폭의 효율 상에서도 같은 효과를 나타낸다. 오직 헤더 전송에만 필요한 대역폭이 헤더 압축이 없을 때 전체 서비스에 필요한 대역폭에서 차지하는 비율로 대역폭의 점유율을 표시할 수 있다. 수식 1.을 이용하여 절약할 수 있는 대역폭의 값을 구할 수 있다.

그림16에 절약할 수 있는 대역폭 수식을 이용하여 ROHC, A+ROHC기법을 비교하였다. 음성 데이터가 클수록 절약되는 대역폭이 많음을 알 수 있으며 약 5%의 성능 향상이 관찰된다. 따라서 A+ROHC 기법

$$\begin{aligned}
 Bandwidth_{saving} &= Bandwidth_{wc} - Bandwidth_c \\
 &= (Header.Size + PayloadData.Size) \times F_{sf} \\
 &\quad - (Compression.Header.Size + PayloadData.Size) \times F_{sf} \\
 &= (Header.Size - Compression.Header.Size) \times F_{sf}
 \end{aligned}$$

Bandwidth_{wc} : 헤더 압축을 하지 않았을 때 전체 서비스에 필요한 대역폭

Bandwidth_c : 헤더 압축을 했을 때 전체 서비스에 필요한 대역폭

F_{sf} : 소스 프레임 율

수식 1. 대역폭 절약 계산

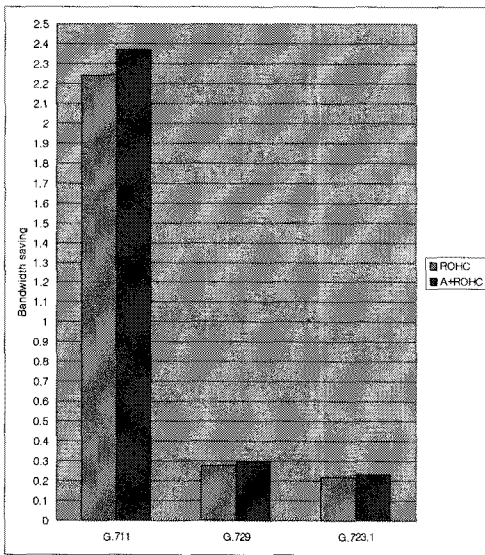


그림 16. 절약할 수 있는 대역폭 비교

으로 기존 기법에 비해 큰 대역폭을 확보할 수 있으므로 PTT의 음성 통화시 성능 향상이 이루어진다.

5. 결 론

본 논문에서 IMS망에서 PTT서비스 구현 시 호 설정 시간의 지연과 통화 성능의 향상을 위하여 프로토콜 압축 방법을 제안하였다. PTT서비스는 1:N의 통화가 순간적으로 이루어져야 하기 때문에 기존의 1:1음성 및 영상 서비스보다 많은 대역폭을 요구하게 된다. IP기반 IMS망에서 PTT서비스의 호 설정 프로토콜인 IP/UDP/RTP의 경우 헤더의 길이가 음성 데이터보다 크다는 단점을 가지고 있다. 높은 에러율과 긴 왕복 소요시간을 갖고 있는 무선망에서는 큰 패킷은 부적합하다고 판단하였다. 따라서 호 설정 프로토콜인 SIP메시지를 A+SigComp기법을 이용하여 메

시지를 압축하였고, 음성 통화 프로토콜은 A+ROHC 기법을 이용하여 헤더 크기를 줄임으로써 전송 시의 효율성을 높였다.

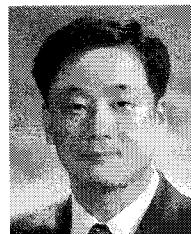
SigComp기법을 이용할 경우 SIP메시지 압축 시 초기의 state를 저장하는 과정에서 처리시간이 많이 지연되므로 SIP Directory방법을 이용한 A+SigComp 기법을 이용하여 초기 state저장 시간을 단축시킴으로써 SIP의 세션 설정 단계의 지연 시간을 약 10% 줄였다.

음성 통화 프로토콜인 IP/UDP/RTP의 경우 ROHC기법을 적용함으로써 헤더의 크기를 줄였다. 하지만 ROHC기법의 경우 패킷 오류 및 손실이 발생할 경우 Full헤더를 전송해야 하는 단점이 있다. 패킷 오류나 손실이 발생할 경우를 최소화하기 위해서 A+ROHC기법을 제안하였다. A+ROHC 경우 Sequence number와 Timestamp의 확장된 LSB방식을 이용함으로써 비트에러 발생 시 패킷 폐기를 최소화 하였다. 결과적으로, A+ROHC기법은 음성 통화 시 교환되는 RTP 헤더의 크기를 줄임으로써 전송 효율 약 5% 높였다.

참 고 문 헌

- [1] 3GPP TR 23.228, "IP Multimedia Subsystem (IMS); Stage 2," Mar. 2007.
- [2] 김규연, "PoC(Push-To-Talk over Cellular) 시스템의 설계와 구현
- [3] SKTelecom, "해외 사업자들의 PTT 서비스 현황 및 전망," 2004.
- [4] 정성구, "VoIP 기반의 CDMA PTT(Push To Talk) 서비스 구현방안, 2003. 05.
- [5] R. Price Siemens/Roke Manor, C. Bormann TZI/Uni Bremen, "Signaling Compression

- (SigComp),” RFC3320, Jan. 2003.
- [6] TSG-SA-WG2, TSG-GERAN and TSG-RAN, “Header compression for optimized voice bearers,” Joint Meeting, 3GPP, Aug. 2001.
- [7] Van Jacobson, “Compressing TCP/IP Headers for Low Speed Serial Links,” RFC 1144, Feb. 1990.
- [8] M. Degermark, B. Nordgern and S. Pink, “IP Header Compression,” RFC2507, Feb. 1990.
- [9] Stephen Casner, Van Jacobson, “Compressing RTP/UDP/IP Headers for Low-Speed Serial Links, RFC2508, Feb. 1999.
- [10] C. Borman et. al, “Robust Header Compression (ROHC),” RFC3095, July 2001.
- [11] KTF, “Always On Standard,” 2004, 04.
- [12] 정의명, “이동통신망 기반 Push-to-Talk 서비스의 호 설정 성능 향상에 관한 연구,” 2005, 06.
- [13] KTF, “SDB Standard,” 2004, 04.
- [14] 류봉석, “CDMA 1x 망에서 PoC서비스의 Call Setup Delay 개선 방법에 관한 연구,” 2004.
- [15] Audio-Video Transport Working Group:1889 H. Schulzrinne, GMD Fokus:S. Casner, Precept Software, Inc.:R. Frederick, Xerox Palo Alto Research Center:V. Jacobson, Lawrence Berkeley National Laboratory, “RTP: A Transport Protocol for Real-Time Applications,” RFC:1889, Jan. 1996.
- [16] Motorola, “M-PTT Protocol-Developer’s Guide,” Motorola Inc., Mar. 2004.
- [17] Verizon Wireless, “Push to Talk (PTT) System Call Flows Version 1.1,” Verizon Wireless, Feb. 2004.
- [18] 차세대 VoIP 음성 코덱 위원회, “차세대 VoIP 음성 코덱 규격(안),” 2003.
- [19] 이미숙 외, “음성 코덱의 특성과 협대역 음성 코덱 표준 기술 분석,” 2005.
- [20] J. Rosenberg, “SIP : Session Initiation Protocol,” RFC 3261, June 2002.
- [21] OpenSigComp project, <http://www.opensigcomp.org>
- [22] Deutch, P., “DEFLATE Compressed Data Format Specification version 1.3,” RFC 1951, May 1996.
- [23] M. Garcia-Martin, Ericsson, C. Bormann, “The Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Static Dictionary for Signaling Compression (SigComp),” RFC 3485, Feb. 2003.
- [24] Haipen Jin, AC Mahendran: “Using SigComp to compress SIP/SDP messages,” IEEE International Conference on Communications, 2005, 3107-3111.A. Bruna et. al., “JPEG rate control algorithm for multimedia,” 2004 IEEE International Symposium on Consumer Electronics, pp. 114-117, Sept. 1-3, 2004.



정인환

1984년 한양대학교 원자력 공학과 공학사
 2000년 한국과학기술원 정보및통신공학과 박사
 1985년 1월~1998년 8월 삼성전자 컴퓨터시스템사업부 부장

2000년 1월~2001년 1월 (주)두루넷 개발팀장
 2001년 3월~현재 한성대학교 컴퓨터공학과 부교수
 관심분야 : 분산시스템, IMS, IPTV, VoIP