韓國 컴퓨터情報學會 論文誌
第10卷 第5號, 2005. 11.

2005-10-5-2-10

차세대 IP체계에서 효율적인 데이터 전송을 위한 확장된 ROHC 알고리즘

김 경 신*, 강 문 식**

Extended ROHC Compression Algorithm for The Efficient Data Traffic Transmission in the IPv6

Kyung-shin Kim*, Moon-sik Kang**

요 약

본 논문에서 차세대 IP 네트워크인 IPv6에서의 효율적인 데이타 전송을 위한 헤더압축 기법을 제안하고자 한다. 기존의 네트워크와 최근 각광받는 무선링크의 대역폭은 제한적인 물리적, 지역적 한계로 인해 점차 줄어들고 있으며 이것은 사용자의 처리율(Throughput)을 감소시키는 큰 요인으로 들 수 있다. 본 논문에서는 기존의 헤더압축방식인 RFC2507 기법과 확장된 형태의 헤더압축 기술과 에러 감지시 신속하고 적절한 복구기술에 관하여 중점을 두고 있는 ROHC(RFC3095) 기법에 관하여 기술하였고, 특히 이 기법의 개선에 역점을 두었다. RTP 헤더에 포함된 SN, TS의 압축효율과 견고성(Robust)을 향상시키는 확장된 ROHC(Extended ROHC) 알고리즘을 연구하였고, 그 알고리즘을 위한 확장 헤더의 표준을 제시하였으며, 개선된 알고리즘을 시뮬레이션을 통하여 증명하기위해 비디오패킷을 대상으로 실험하였다.

Abstract

In this paper, we propose the enhanced header compression scheme for the efficient data traffic transmission in IPv6 networks. The bandwidth of wireless links and IP networks will probably be limited due to properties of the physical medium and regulatory limits on the use of frequencies for radio communications. That is major cause of user throughput reduction. Therefore, We discuss the IPHC(RFC2507) and ROHC(RFC3095) scheme. IPHC is simple header compression scheme and ROHC is enhanced header compression that have fast optimal recovery scheme. We have studied the enhanced header compression scheme in ROHC. We will show that indication of compression context values preventing from packet losses can provide the fast recovery of compression state. Computer simulations show that the proposed scheme has better performance than the previous one.

▸ Keyword : 헤더압축(Header Compression), Robust헤더압축(ROHC)

• 제1저자 : 김경신
• 접수일 : 2005.10.10, 심사완료일 : 2005.10.29
* 청강문화산업대학 컴퓨터네트워크과 교수, ** 강릉대학교 정보전자공학부 교수

I. 서 론

패킷 네트워크를 통해서 음성이나 비디오 같은 실시간 데이터를 전송할 때 사용되는 RTP(Real time protocol)에서 전송되는 패킷은 IP 헤더(20 octets), UDP(User datagram protocol) 헤더(8 octets), RTP(12 octets) 등으로 캡슐화 되어, 헤더부분의 크기가 40 바이트를 차지하게 된다[1]. IPv6를 사용할 경우에는 60 바이트의 크기를 갖게 되며[2], 또한 여러 가지 확장 헤더들은 라우팅 정보나 인증 정보 등 추가정보를 제공하기 위해서는 기본헤더에 부가된다. 대역폭을 보다 더 적절하게 사용하는 다양한 방법 중의 하나는 각 패킷의 중복되는 헤더의 오버헤드를 제거하여 헤더의 사이즈를 줄이는 것이다. (그림 1)은 IPv6의 헤더를 나타낸 것이다.

같은 패킷 스트림에 속한 패킷들은 version, flow label, next header, 송신자주소, 수신자주소가 같다. 또한 IPv6 헤더의 payload 길이 필드는 캡슐화(Encapsulation)되는 링크 계층의 길이 필드에 영향을 받는다. 이러한 분석을 통해, 목적지 노드는 IPv6 패킷을 수신 하고나서 (이전 패킷의 정보가 저장되어 있다는 가정 하에) 헤더 필드의 많은 부분을 추론 후 복원 가능하다는 것을 알 수 있다.

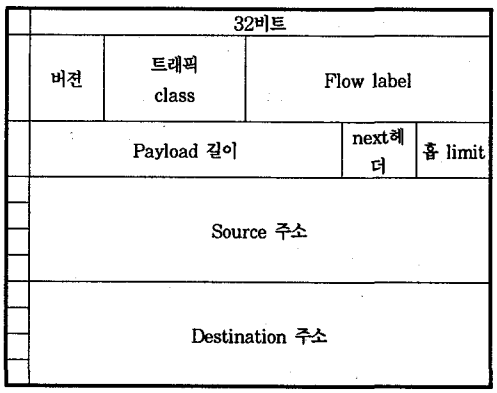

그림 1. 40-옥텟 IPv6 헤더
Fig 1. 40-Octet IPv6 Header

(그림 2)는 UDP 헤더를 나타내고 있다. 많은 헤더필드들이 같은 패킷 스트림내의 연속적인 패킷간에 static한 부분을 가지고 있다. UDP에서 같은 스트림내의 두 개의 연속적인 패킷은 같은 송신자주소와 수신자 주소를 가질 것이다. 더 나아가서 UDP 길이 필드는 하위 계층의 프로토콜 스택으로부터 유추 가능할 것이다. 그러나 checksum 필드는 불규칙적(random)이며 각 패킷마다 다양한 값을 가질 것이다.

UDP와 유사하게, TCP헤더도 static하거나 random한 필드 뿐 아니라 dynamic한 필드(예를 들면 TCP sequence number)를 가질 수 있는데, 이것은 2개의 연속적인 패킷간에 특정 값으로 증가하는 것을 말한다.

헤더 압축 방법은 IPHC와 ROHC로 대표되는데, 이들은 프로토콜의 헤더필드의 중복성을 활용하게 되며, 보다 더 효율적인 통신을 가능하게 한다.

Source 주소	Destination 주소
길 이	Checksum

그림 2. 8-바이트 UDP 헤더
Fig 2. 8-byte UDP Header

II. IP 헤더압축(Internet Protocol Header Compression) 기법

IP 헤더압축은 RFC2507로 정의되어 있으며[7], TCP/IP(v6 포함) UDP/IP(v6 포함) 그리고 ESP/IP(v6 포함)에서의 헤더압축을 구현하며 주로 point-to-point 연결을 기본으로 하고 있다. IPHC 프레임웍은 또한 RTP/UDP 헤더의 압축을 위한 확장을 포함한다[2].

IPHC는 패킷헤더의 특성, 즉 많은 패킷헤더들이 연속적인 패킷 스트림내에 변하지 않는 값을 유지한다는 것을 이용한다. 이런 식으로 연속적인 패킷간에 중복적으로 정보가 유지되고 있는 사실을 기초로, 헤더압축 프로토콜은 Compressor와 Decompressor를 정의한 후 보내지 않은 데이터를 추론 복원 한다. 이러한 전송할 헤더의 량의 감소의 결과는 사용자 처리율(throughput)을 증가시킨다.

2.1 헤더압축 프로토콜의 동작원리

헤더압축 프로토콜은 RFC1144의 Van Jacobson 헤더 압축(VJHC)을 기본으로 출발 하였다[3].

이 프로토콜은 패킷스트림(Packet Stream)에 그 개념을 두고 있는데, 동일한 스트림(세션) 내에 존재하는 헤더 영역의 대부분은 동일한 값을 갖게 된다. 예를 들어, IP나 UDP/TCP헤더 내에 존재하는 주소영역과 포트영역은 같은 스트림내에서는 같다. 또한 RTP의 TimeStamp나 Sequence Number같은 영역은 값이 순차적으로 일정한 크기를 가지고 변하며, UDP 길이영역은 링크계층의 영역과 중복된다. 이러한 패턴분석을 기반으로, RTP/UDP/IP 헤더영역의 중복되는 부분은 제거하고, 일정하게 증가하는 영역의 값에 대해서는 변화량만을 전송하면 헤더압축이 가능하다는 방법이 현재 RFC 2508로 승인되어 있다[2].

Compressor는 항상 변하지 않는 필드와 그것과 관련된 context들 (예를 들면 context identifier인 CID, 이것은 최초 헤더의 길이 필드와 교신한다) 을 이용하여 static 헤더를 작성한다. 반면 수신측의 Decompressor는 이러한 context를 저장한 후, 그것을 이용하여 같은 CID를 가지는 연속적인 패킷을 decompress 한다.

만일 이러한 static 헤더 필드가 변화 된다면(혹은 context 동기를 잃게 된다면) 각각의 필드는(혹은 모든 full 헤더) refresh를 위해 재전송 되어야 한다.

VJHC가 TCP/IPv4 스트림만을 유일하게 적용하는 반면, 헤더압축 프로토콜은 UDP/IP, TCP/IP 패킷 헤더의 압축을 제공한다. 그리고 RTP/UDP/IP의 압축으로도 확장 가능하다.

2.2 Dynamic 헤더필드의 압축

static 헤더필드의 압축의 경우와 같이, 헤더압축 프로토콜은 delta-based differential encoding scheme을 사용한다.

delta-based differential encoding scheme이 사용된다면, Compressor는 Decompressor가 dynamic 필드의 모든 값을 전송하지 않고도 dynamic 헤더 필드의 증가분의 정보를 계산해 낼 수 있도록 충분한 정보를 send 해야 한다. (예를 들면 TCP sequence number)

예를 들면, 2개의 연속적인 TCP sequence number가 각각 306080과 306180 이라고 가정해 보자.

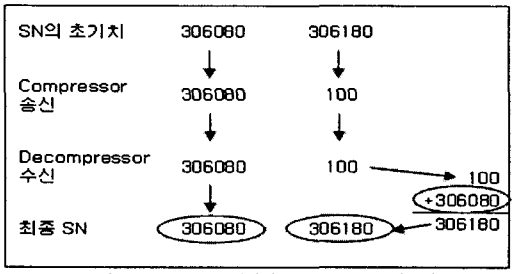

그림 3. dynamic 헤더의 differential 인코딩
Fig 3. differential encoding of dynamic header

Compressor와 Decompressor의 context를 설정하기 위해서, Compressor는 최초의 패킷에는 full header를 전송하게 된다.

다른 static 필드와 함께 Decompressor는 첫 번째 패킷의 TCP sequence number를 306080을 저장하게 된다. 두 번째 패킷이 전송되는 동안 Compressor는 306180 대신에 이것을 압축(이것을 differential encode 라고 하자)하여 sequence number를 100으로 전송하게 된다. 이것은 단지 3자리수의 정수값으로 전송되어지는 결과를 낳게 된다. differentially encode된 필드를 받고나면, Decompressor는 두 번째 패킷의 sequence number의 실제적인 값을 계산해 내어 306180(306080 + 100 = 306180)을 복원하게 된다. 이러한 시나리오를 (그림 3)에 나타내었다.

III. 개선된 IP 헤더압축 기법

본 절에서는 앞에서 기술한 RFC2507로 알려진 헤더압축 알고리즘을 보다 개선한 알고리즘인 ROHC(Robust Header Compression Protocol)에 대하여 기술하였다. RFC 3095으로 명명이 된 이 알고리즘은 무선 링크상에서 IP 프로토콜을 구현하는 핵심기술이다. 특히 무선 링크에서는 일반 유선 링크와 달리 높은 에러 전송율은 나타내데, 헤더 부분의 에러는 특히 치명적인 결과를 낳을 수 있어 에러에 강한 헤더 처리 기술이 필요한 상황이다. 또한 여기에 IP 패킷의 헤더의 크기가 너무 커 대역폭이 한정되어 있는 무선 링크에서는 기존의 헤더를 이용한 프로토콜이 최적의 솔루션이 될 수가 없다. 예를 들어 IPv4를 사용해 VoIP를 구현하게 된다면 66퍼센트의 무선 주파수 대역폭이 헤더를

전송하는데 낭비될 것이고 IPv6는 헤더 크기가 확장돼서 더욱더 높은 대역폭의 손실을 가져오게 되는 결과를 낳게 된다.

ECRTP와 같은 압축기법이, 여러개의 IPHC와 그 유사 기법에 대한 대안을 제공하고 있지만, 손실률이 높고 latency 시간이 적은 무선링크상에서의 효율적인 헤더압축 기법의 기반은 ROHC라고 할 수 있으며 이것은 다양한 헤더 압축방법을 제공한다.

3.1 ROHC의 FSM 상태

FSM은 Finite State Machines의 약자로 Compressor와 Decompressor간의 상태도를 의미한다.

기능적으로 ROHC 기법은 Compressor에 위치한 Compressor FSM과 Decompressor에 위치한 Decompressor FSM간의 상호작용을 기본으로 한다.

Compressor의 FSM은 초기화와 refresh(IR) 상태, first order(FO) 상태, 그리고 second order(SO) 상태로 구성된다.

- IR: 초기화와 context의 static한 부분들(예를 들어 source address, destination address, 등)의 초기화를 담당하고 헤더는 비 압축된 형태이다.
- FO: FO 상태는 Decompressor와 Compressor 간의 부분적 context를 만든다. FO 상태는 패킷스트림에서 예외적인 부분들에 대한 효율적인 통신을 제공한다. 그래서 Compressor에 의해 전송되는 헤더는 부분적으로 압축된 상태이다.
- SO: 링크간 가장 적절한 헤더압축상태를 나타낸다. Decompressor와 Compressor 간 full context를 만들고 Decompressor는 대부분의 헤더 패턴을 이해하기에 충분한 정보를 갖게 된다.

(그림 4)는 이러한 Compressor의 상태와 상태전이를 요약해 놓은 것이다.

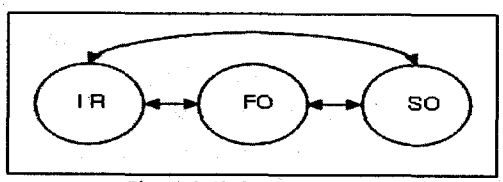

그림 4. ROHC Compressor FSM
Fig 4. ROHC Compressor FSM

Decompressor의 FSM은 NC, SC, FC상태로 구성된다.

- NC : no context 상태
- SC : static context 상태
- FC : full context 상태

NC는 Decompressor의 초기상태이다. 이것은 Decompressor가 압축된 헤더를 성공적으로 decompression 할 수 없다는 것을 의미한다.

헤더의 성공적인 decompression 이후에 Decompressor는 FC상태로 된다. static과 dynamic 헤더 필드들의 연속적인 decompression이 성공하면 Decompressor는 각각 SC나 NC 상태로 back 하게 된다.

(그림 5)는 Decompressor의 상태와 transition을 요약해 놓았다. 다음의 예는 낮은 error rate의 링크상에서 Decompressor와 Compressor 상태간의 상호작용(interaction)을 보여준다. 우선 Decompressor가 FC상태라고 가정하자.

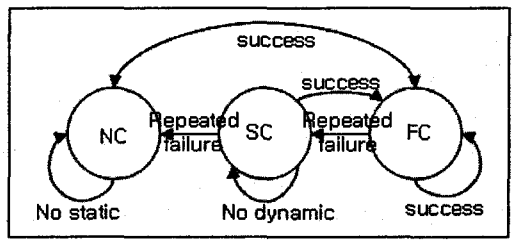

그림 5. ROHC decompressor FSM
Fig 5. ROHC decompressor FSM

Decompressor가 FC상태에서 decompression이 실패하면, SC상태로 가게 된다. 이 시점에서 압축된 FO 헤더가 성공적으로 decompression되면 Decompressor는 FC상태로 가게 된다. 그러나 압축된 FO헤더의 decompression이 실패하면 Decompressor는 NC상태로 간다.

3.2 ROHC 작동 Mode

Compressor와 Decompressor간의 작동은 다음과 같은 3가지 상태의 FSM으로 구성된다[4].

U-mode(Uniderectional mode)는 단방향 압축모드 방식은 Compressor로부터 Decompressor간에 엄격하게 패킷이 전송되는 단방향 링크상의 압축을 말한다. 헤더필드/패턴에 대한 Decompressor의 지식에 의존(Compressor에 의해 판단되어지는) 한다고 본다.

Compressor는 IR에서 FO 상태 또는 "optimistically" SO 상태로 upward하게 된다. Compressor의 downward 상태로의 transition은 주기적인 timeout과 헤더패턴의 불규칙성에 기인한다고 볼 수 있다. 주기적인 timeout에 의한 상태저하는 (Compressor가 SO로부터 FO, IR 상태로 transition된다는 것은 full header를 전송하게 된다는 것이다.) Compressor가 Decompressor로 부터의 context synchronization 피드백을 못 받기 때문이다.

O-mode(Bidirectional optimistic mode)는 Compressor와 Decompressor 간의 양방향의 ROHC 방법을 말한다. 이 모드에서 피드백 채널이 Decompressor로부터 Compressor 간에 존재한다. Decompressor는 단지 decompression이 잘 안되었을 경우 이것을 인지시켜 준다거나, 피드백 메커니즘을 사용하게 된다.

예를 들자면 통신에서 중요한 context의 update들에 대한 acknowledgement를 들 수 있다. O-mode의 목적은 헤더의 압축을 최대화 하게 하거나 피드백 채널의 사용을 최소화 하게 하는데 있다.

R-mode(Bidirectional reliable mode)는 Compressor와 Decompressor간 양방향 ROHC 방법을 말하는데 R-mode는 O-mode 보다 더 신뢰성 있는 ROHC 방법이라고 할 수 있다. R-mode는 양단간 손상된 context를 최소화 시켜서 손실/손상된 패킷 시나리오에 대한 견고성(Robustness)을 극대화하는데 그 목표를 두고 있다.

이것은 피드백 채널의 빈번한 사용과 보다 더 엄격한 로직(logic)에 의해 성취된다. 그러나 이러한 방법으로 신뢰성을 얻는 대신에 압축되는 양은 O-mode보다 적다.

이것을 분석하여보면 O-mode와 R-mode를 보면 transition이 Decompressor로부터 수신된 피드백의 기능이 주를 이루고 있고, 반면 U-mode는 주로 timer에 의한 기능임을 알 수 있다.

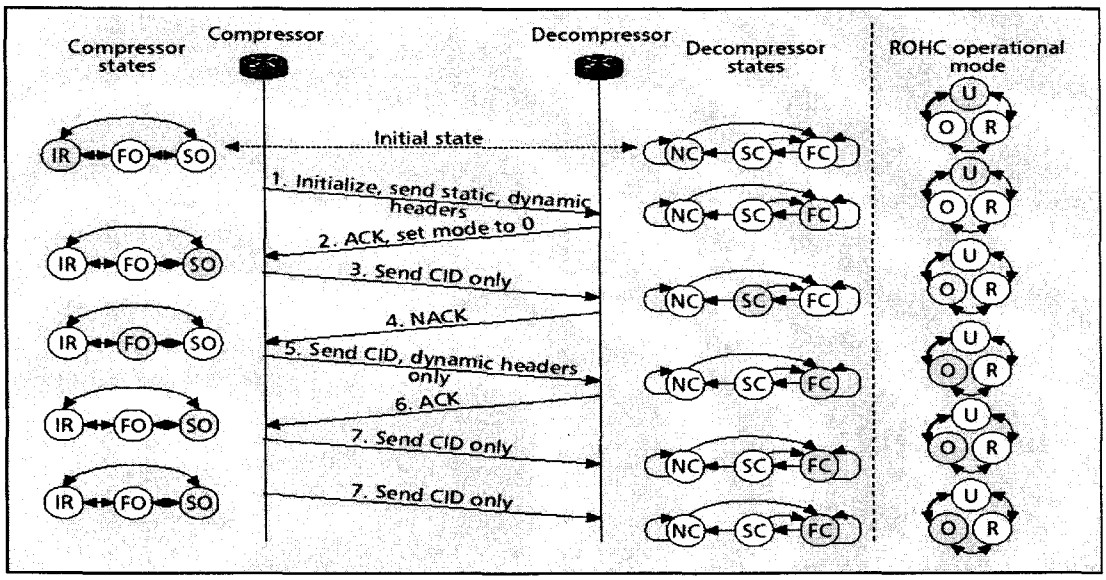

그림 7. ROHC 초기화 과정과 동작절차
Fig 7. ROHC initial processing and operation

3.3 윈도우기반의 LSB 인코딩

ROHC는 dynamic 헤더 필드의 압축을 높이기 위해 IPHC의 delta-based 인코딩에 비해 개선된 윈도우 기반의 LSB 인코딩 방법을 제안하였다.

W-LSB 인코딩 방법은 여러 개의 연속된 패킷이 손실되는 경우 장점을 찾을 수 있는데, 이때 context는 즉시로 resynchronized 될 수 있다. W-LSB 인코딩 방식을 쓸 때, 헤더의 LSB k-bit를 전송한다. 이 k-LSB를 수신한 후, Decompressor는 헤더필드의 실질적 값을 재구축(재생)가능하다.

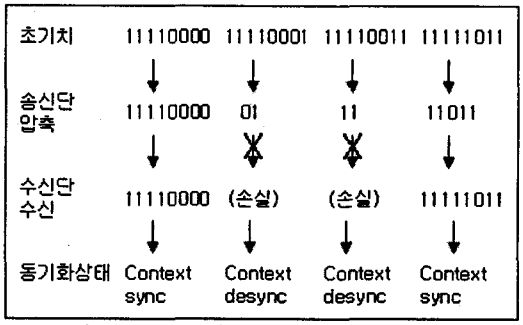

그림 6. W-LSB를 사용한 dynamic 헤더필드의 계산
Fig 6. dynamic header cal. using W-LSB

(그림 6)은 IPv6 헤더의 dynamic traffic class 필드에 대한 W-LSB 인코딩 기법을 응용한 예이다.

AF(asured forwarding) class 2 medium drop precedence로부터 AF class2 high drop precedence로의 traffic class 필드의 변화분(change)에 대한 통신 예이다. Compressor와 Decompressor는 연속적인 패킷 손실 이후에라도 resynchronize가 가능하다.

3.4 ROHC 작동

아래에 나열된 step과 (그림 7)은 단순한 ROHC 세션의 한 실례를 들은 것이다. context resynchronization을 위한 Compressor-Decompressor의 초기화 과정과 피드백 메커니즘에 대해 자세히 나타내었다[4].

1. Compressor는 ROHC 메커니즘을 작동케 하기위해 CID와 함께, static, dynamic 헤더필드와 초기화 패킷을 보낸다.
2. Decompressor는 이 패킷을 수신한 후, FC상태로 바꾼후 ACK 신호를 응답한다. 이때 Decompressor는 자신이 O-mode로 간다는 것을 표시한다.
3. Compressor는 ACK를 접수한 후, ROHC O-mode로 간다. 그리고 Compressor FSM을 SO 상태로 한다. 다음 패킷과 함께 Compressor는 헤더와 패킷 Payload로 구성된 CID만을 보낸다.
4. 그러나 Decompressor는 수신한 패킷 헤더를 decompress 하지 못하였다. Decompressor는 패킷을 폐기하고 SC 상태로 간 이후, NACK 신호를 송신한다.
5. NACK를 수신하면, Compressor는 FO 상태로 downward 한다. Compressor는 다음 패킷에 CID와 dynamic 헤더필드로 구성한 패킷을 보낸다.
6. Decompressor가 압축된 헤더를 decompress 하는데 성공하고, FC상태로 바꾸며 ACK 를 응답한다.
7. Compressor는 ACK를 받고, SO 상태로 복귀한다. Compressor는 다음패킷(헤더와 패킷 Payload로 구성된 CID 단독)만 전송한다.
8. Compressor는 다른 패킷을 송신한다. 이것은 CID로만 구성한다.(헤더 + 패킷 Payload) 왜냐하면 ROHC O-mode에서 Compressor는 매 패킷마다 각각의 ACK를 기다리지 않는다.

3.5 ROHC와 IPHC의 비교

〈표 1〉은 IPHC와 ROHC 프로토콜의 장점과 단점을 비교분석한 표이다.

이러한 trade-off는 어느 장비에 특정 헤더압축 알고리즘을 구현할 것인가를 결정할 때 필요하게 된다.

표 1. IPHC와 ROHC의 비교
Table 1. Comparision of IPHC and ROHC

	장 점	단 점
IPHC	- TCP/IP, UDP/IP, ESP/IP 헤더 압축가능 - 구현이 비교적 단순 - RTP/IP헤더의 압축을 위한 확장 프로토콜 제공 - 패킷손실과 순서조정을 통한 견고성을 제공키위한 확장형 프로토콜 제공	- 긴 RTT를 가지는 링크상에서 context 재구축의 지연 - 프로세싱과 메모리가 필요
ROHC	- ESP/IP, UDP/IP, RTP/UDP/IP, TCP/IP 헤더압축 가능 - Robust 기법 사용 - 신속한 context 재구축을 위한 메커니즘의 통합 - 동적으로 변화하는 헤더의 필드들의 압축을 위한 개선된 인코딩 방법 제공	- 프로세싱과 메모리 자원을 이용한 구현의 복잡 - 프로토콜의 표준화가 미확립

Ⅳ. 제안된 확장 W_LSB 인코딩 ROHC

본 연구에서는 비디오 트래픽에 대한 효율적인 헤더압축 기법을 적용하기 위해 MPEG 스트림을 사용하였다. RTP를 사용해 Video를 압축할 경우, 한 개의 패킷은 하나 이상

의 그림을 포함하지 않으며, 한 개의 그림은 두 개 이상의 RTP 패킷으로 나누어 질 수도 있다. RTP Marker bit은 하나의 그림을 가지고 만들어진 패킷 중 마지막 패킷에서 1로 설정된다. 비디오 정보와 관련된 헤더압축 프로파일은 참조클럭으로 90kHz를 사용한다. 본 연구에서는 헤더의 compression을 위해 RTP/UDP/IP 구조를 이용하였고, TimeStamp, Sequence Number의 변화를 어떻게 표현할 것인지와, 이를 위한 확장헤더를 설계하였다[1]. 다음 (그림 8)은 UDP의 유료부하인 RTP 헤더와 RTP 유료부하의 형태이다.

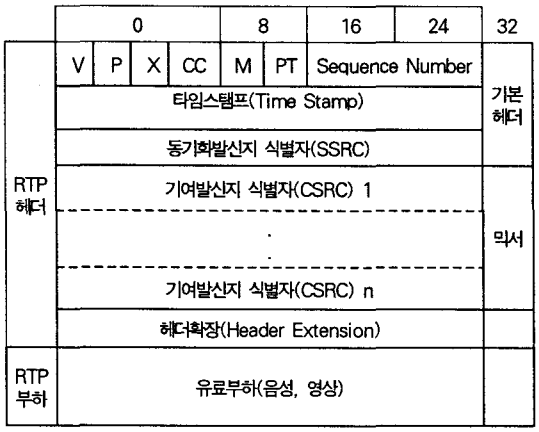

그림 8. RTP 헤더와 유료부하
Fig 8. RTP Header and Payload

4.1 Sequence Number 압축

16비트 순서번호의 압축과정은 다음과 같다. SEQ7 은 순서번호에 대한 모듈로-7연산을 수행한 결과이며, SEQR 은 7에 의해 나누어진 후의 정수부분을 의미한다. 그러면 순서번호의 압축은 SeqNr = SEQR*7 + SEQ7 수식에 의해 간략화 될 수 있다. SEQ7은 압축헤더의 첫 3비트에 표시되어 전송된다. 기존의 방법은 헤더 압축 전에 1개의 reordering을 처리하기 위해, 사용하는 코드의 순서번호 변화로 [-1, 5]범위를 다루고 있다. 이 방법은 순서번호의 동기화를 잃지 않고도 연속적인 4개의 패킷 손실을 복구할 수 있다.

본 연구에서는 보다 확장된 방법으로, SEQRLSB를 이용하여 확장헤더가 전송되도록 한다. 여기서 SEQRLSB는 SEQR의 LSB(least significant bit)를 의미한다. 확장헤더를 이용한 후에 순서번호의 변화는 [-3, (7*2SEQR_NB)-4]과 같은 수식을 갖게 된다. 여기서 SEQR_NB는 Decompressor에 전송된 SEQR 비트수 이며, 이처럼 범위를 확장함으로써 다중패킷 손실에 대한 강인성을 제공할 수 있다. 순서번호를 Decompressor 측에서 해석하기 위한 절차로, S(n)을 RTP의 순서번호라 하고, S(n'-1)은 가장 최근에 디코드된 순서번호라 하면, 제안된 기법에 의해 순서번호의 범위를 확장했을 때 아래와 같은 형태로 표현할 수 있다.

$$S(n') = S(n'-1) - SEQRLSB(n'-1)*7 - S(n'-1) \bmod 7 + SEQRLSB(n')*7 + S(n') \bmod 7 \quad \cdots\cdots (4.1)$$

만약 이전의 순서번호에 대한 SEQR 비트의 전송이 없다면, 디코더는 순서번호를 디코드하기 위해 선행하는 순서번호 값을 계산해야 한다. 이러한 경우, Modulo wrap-around 검사는 순서번호의 Delta값인 [-3, (7*2SEQRLSB_NB)-4] 사이의 간격을 처리할 수 있도록 바뀐다. 여기서RTP 순서번호 Wrap-around란, 16비트 크기의 순서번호가 가질 수 있는 최대값인 65535에서 다시 0으로 넘어갈 때 SEQRLSB의 값이 잘못 인식되는 것을 의미한다. Wrap-around에서 패킷 손실이 발생하면, 순서번호의 디코드된 값이 잘못될 확률이 증가하는데, 이런 현상을 피하기 위해서 동적 패킷이나 압축 패킷의 확장을 사용해서 순서번호를 코드화 하였다.

4.2 TimeStamp(TS) 압축

압축효율을 위해 현재의 RTP TS변화는 고정된 값의 배수로 변하도록 설계하였다. 이는 현재의 PCF (Picture clock frequency)와 같은 값이다. 같은 그림 내에서는 TS의 차이가 '0'이 된다. 그리고 B를 뒤따르는 B뿐 아니라, 원래의 Intra(I)나 Inter(P) picture에 대해서는 TS의 값이 PCF의 몇 배가 되거나 0보다 큰 값이 될 것이다. H.261, H.263 ver 1의 PCF 는 29.97Hz 으로, 이는 두 코드화된 그림 사이의 증가가 3003이 된다는 의미이다. 또한 H.263 ver 2, MPEG-4 의 PCF는 각각 25Hz, 30Hz가 되는데, 이 값은 각각 3600, 3000배의 TS에서의 증가와 같은 값이다. 패킷 손실시에도 TS의 증가는 PCF의 배수이므로 이런 성질은 PCF 배수인 TS의 증가분을 아는 데 충분하므로 압축에서 사용하면 이득을 얻을 수 있다. 상업용 코덱이나, 이에 따르지 않는 코덱들도 지금까지 설명한 비디오 프로파일의 TS값을 확장헤더를 사용해서 보낼 수 있다.

B-picture 가 사용되는 경우, 두 그림 사이에서 TS가 감소하게 되는 경우가 발생한다. 이러한 경우에 대처하기 위해서 다음과 같은 방법을 사용하였다.

TS = TSQ * PCTSI + TSR ..... (4.2)
TS = RTP timestamp,
PCTSI = PiCTure clock interval in TS
(예를들면 90000/25Hz = 3600)
TSQ = TimeStamp Quotient,
TSR = TimeStamp Residual(=TS modulo PCTSI)

만약 두 패킷 사이의 TS의 변화가 PCTSI의 배수이면, TSR은 같은 값이 유지되면서, TSQ만 바뀔 것이다. TSQ를 인코드하기 위해 5 LSB비트가 기본압축 패킷에서 전송된다. TS값이 감소되는 경우를 처리하기 위해서 TSQ 비트는 [-6, 25]의 범위와 일치하는 Delta TSQ의 범위를 처리하는 방법을 이용한다. 이 방법을 확장했을 경우 [-10, 2N+5-11] 을 사용할 것을 제안한다. 여기서 N은 확장부분에 전송된 TSQ의 비트수 이다. 간단한 예로, 압축헤더에 5 개의 TSQ 비트가 사용되면 [-6, 25]를 처리하게 되고, 여기에 확장헤더를 사용하여 추가의 2비트 TSQ 를 만들면 [-10, 117]를 처리할 수 있다.

PCTSI값은 동적 패킷이나 압축 패킷의 확장부분에 있는 TS Delta영역을 사용하여 전송하면 된다. 만약 TS가 PCTSI의 배수로 바뀌지 않는다면 압축 패킷을 사용하여 여러개의 TS LSB를 확장부분에 넣어서 전송할 수 있다. TS_LSB_NB를 TS_LSB영역의 크기라 놓으면, TS가 폐기되는 것을 방지하기 위해서 TS LSB는 마지막으로 전송된 TS로 부터 [216, 2TS_LSB_NB - (216 -1)]의 범위를 처리해야 한다.

4.3 확장헤더의 설계

본 연구에서는 static, dynamic 패킷을 고려하였다. 유료부하가 없는 static구조는 (그림 9)와 같다. 단지 한 개의 static패킷이 각각의 경우에 전송되는데, 만약 Decompressor가 dynamic이나 static 패킷이 없는 헤더압축을 받는다면, Decompressor는 static패킷을 요구해야 한다.

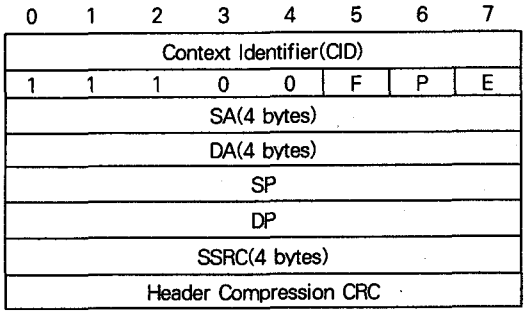

그림 9. static 패킷 설계
Fig 9. static Packet design

여기서 F는 Fragment, P는 Padding, E는 RTP Extension을 말한다.

두번째 패킷은 dynamic 정보패킷(Information Packet)이다. 원래의 헤더에서 모든 바뀌는 부분을 포함하는 헤더이며 Payload를 전송한다. 이 패킷은 우선 Decompressor측의 컨텍스트(Context)를 설정하기 위해 처음에 전송된 static 패킷 후에 전송된다. (그림 10)은 동적 패킷의 구조이다.

헤더영역에서 덜 불규칙적인 변화는 기본적인 헤더에 추가하여 확장헤더를 요구한다. 처음 3비트는 확장부분의 타입을 위해서 사용된다. TS Delta, TS LSB, TSQ, SEQR, TSC, Bit mask등이 확장부분을 이용하여 전송할 수 있는 것들이다. TSQ의 추가적인 LSB는 TS범위를 증가시키기 위해서 사용된다.

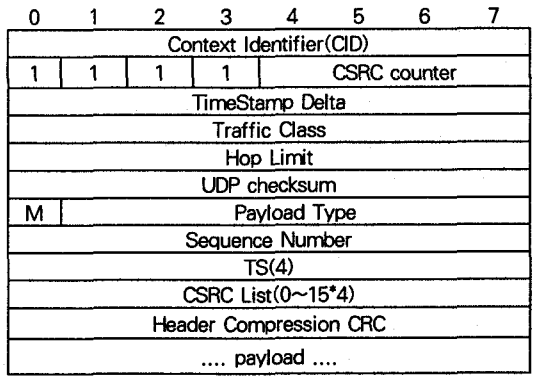

그림 10. dynamic 패킷
Fig 10. dynamic packet design

기본헤더의 비트는 항상 LSB이다. SEQR 영역은 SN 나머지를 표현하는 LSB값 이다. TSQ LSB와 TS LSB가 같은 헤더 내에 존재한다면, TSQ LSB는 무시되고 TS LSB만 고려가 된다.

TSC는 큰 TS Delta 영역 대신에 확장헤더 부분에서 사용된다. (그림 10)의 RTP 헤더에서 확장을 위한 헤더선언부는 4바이트이고, 헤더의 내용도 4바이트 단위로 구성하면 된다[10][11]. (그림 12)는 확장헤더의 형식이고, (그림 11)에 제안된 내용을 나타내었다.

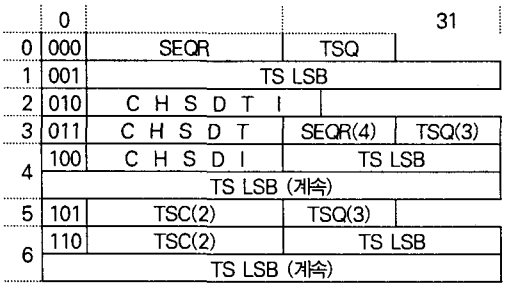

그림 11. 확장헤더의 제안
Fig 11. Extension of packet

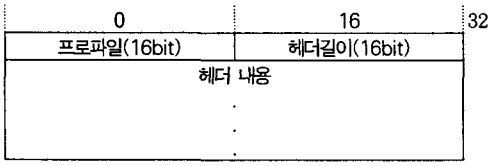

그림 12. 확장헤더의 Format
Fig 12. Format of Extension Header

V. 시뮬레이션

5.1 모델 및 파라메터

안된 기법의 타당성을 검증하기 위하여 다음과 같은 기본 가정을 하였다.

Forward 채널 상에서 전송된 패킷은 손실될 수 있고, 패킷 에러의 특별한 형태가 없다. 에러검출기법을 이용하여 손상된 정보가 전달되지 않고, 에러검출은 링크에 의해서 제공된다. 패킷의 순서는 Compressor와 Decompressor 사이에서 유지된다. 즉 수신기는 항상 전송측에서 보낸 순서대로 패킷을 받는다. 피드백 채널 상에서 엄격한 지연이나 에러요구사항이 없다. ACK는 손실 및 지연될 수 있으며, 에러와 지연의 특성은 시간에 따라 변동될 수 있다. 패킷순서는 피드백채널 상에서 유지된다. 제안한 알고리즘은 RTT의 어떤 특별한 분포를 가정하지 않는다. 동적으로 RTT에 대한 변화에 적응할 수 있으며, 헤더압축은 인터리빙과 채널코딩에 영향을 미치지 않는다.

제안된 헤더압축상태의 빠른 복구과정을 분석하기 위해서, 앞에서 기술한 가정하에 Network Simulator-2를 사용하여 시뮬레이션을 수행하였다. 링크 레벨 시뮬레이션은 WCDMA 채널모델을 사용하였다. 사용된 파라미터는 다음과 같다.

먼저 패킷은 초당 100개가 생성되며 비디오 code rate는 384 kb/s이다. 비디오는 320 비트 프레임이며 헤더의 프로파일은 1~2바이트를 가진다. RTT는 120 ms이고 이동노드는 20개라고 가정했다. 마지막으로 비디오 프로파일의 참조클럭은 90KHz로 설정하였다.

5.2 패킷 손실률 분석

본 연구에서는 FEC, RF 변조 및 다중액세스 기법에 대한 효과는 제외하고, 핸드오프만 고려하였다. RTP/UDP/IP Agent기능을 가진 호스트는 비디오 트래픽을 발생시키고, Compressor까지 전송되어 온 패킷은 압축 초기화 과정을 거친 후에 RTP 프로토콜이 지원하는 세션 내에서 2바이트 크기의 헤더를 만들어 전송하게 되고, 이를 수신한 Decompressor 호스트에서 압축헤더를 복구하게 된다.

(그림 13)은 비트에러율이 주어졌을 때, 제안된 압축기법과 기존 방법과의 패킷손실율을 비교한 것이다. 주어진 BERs에서 제안한 기법은 패킷 손실률이 5% 이내로 제한됨을 확인하였다. 기존의 RFC-2508에서 제안된 기법의 경우는 패킷 손실률이 최대 30~35%까지 발생하므로, 제안된 기법이 패킷 손실률에서 기존의 방법에 비해 보다 좋은 성능을 보임을 확인하였다.

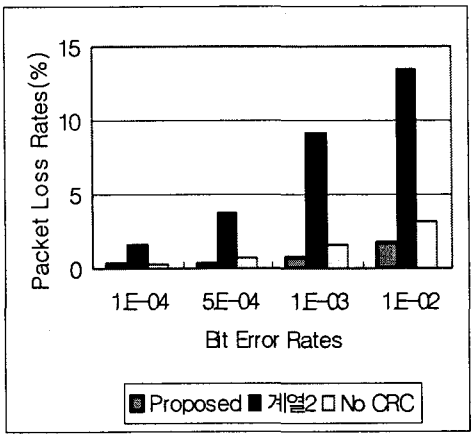

그림 13. 패킷손실률 분석
Fig 13. Analysis of packet loss

5.3 Context 재설정 시간 분석

핸드오프 동작에 의해 이동호스트가 위치를 변경했을 때 새로운 IGSN으로 부터 서비스를 받는데 걸리는 시간을 분석한 결과 평균 130ms정도, 최대 250ms의 시간이 소요되는 것을 확인할 수 있었다. 20ms 프레임이 7개 정도 손실될 수 있는 시간이지만 우리가 설계한 비디오 헤더압축 프로파일에서는 최대 25까지의 패킷 손실을 복구 할 수 있기 때문에 문제가 되지 않는다. 이동호스트가 비디오 스트림에 대한 RTP SN(Sequential Number)이나 TS(Time Stamp)차이 값을 예측함으로써 25개 이상의 연속된 패킷을 복구할 수 있는 기법을 사용하였다.

MPEG-4 비디오의 경우, 종단간 지연값이 최대 400ms이기 때문에, 핸드오프 재설정 시간인 130ms는 양호한 값이 된다.

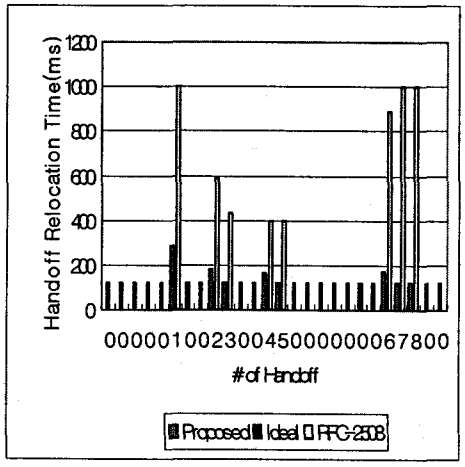

그림 14. 재설정 시간 분석
Fig 14. Analysis of context resync

Ⅵ. 결론

비디오 패킷의 효율적인 전송은 제한된 대역폭을 갖는 네트워크에서 매우 중요하며, 대역폭의 효율적인 사용과 원하는 서비스품질은 효율적인 헤더압축 기법과 함께 제공되어야 한다. 제안된 기법은 IP 네트워크에서 헤더의 크기를 획기적으로 줄이는 하나의 방법을 제시한 것이다. 본 연구에서 설계한 확장된 LSB 방식을 가지고 시뮬레이션을 수행한 결과, 링크의 대역폭 효율성 context sync등에 대해서 좋은 성능을 얻을 수 있었다. 비디오 트래픽의 효율적인 전송을 위해 링크계층에서 RTP/UDP/IP에 대해서 BER-resistent 헤더압축 알고리즘의 제공이 요구된다고, 또한 긴 RTT를 가진 셀룰러 환경에서 동작도 원할 해야 한다. 단일 무선링크 보다는 다중 링크환경이 일반적이어서, RTT값이 커질 수 있기 때문이다. 또한 유료부하에서 비트에러가 발생했을 때 패킷 폐기가 최소화되어야 한다.

제안된 기법은 사용자의 수가 작은 모델에서는 좋은 성능을 보였지만, 사용자의 수가 늘어나고, 서비스를 요구하는 노드가 많아지는 경우에 대해서 연구가 계속될 것이다.

참고문헌

[1] H. Schulzrimme et al., "RTP: A Transport Protocol for Real Time Applications," IETF RFC 1889, Jan. 1996.

[2] Stephen Casner, Van Jacobson,"Compressing RTP/UDP/IP Headers for Low-Speed Serial Links", RFC 2508, February 1999.

[3] Van Jacobson, "Compressing TCP/IP Headers for Low-Speed Serial Links", RFC 1144, February 1990.

[4] Emre Ertekin, "Internet Protocol Header Compression, Robust Header Compression and Their Applicability in the GIG" IEEE Communications Magazine Nov 2004.

[5] 서영건, "멀티미디어 통신", 인솔미디어, pp281

[6] 조재수, "Standards for Multimedia Communications", 한국기술교육대학교 멀티미디어통신 2003.11, pp17-24.

[7] Mikael Degermark, Bjorn Nordgren, Stephen Pink, "IP Header Compression", RFC 2507, February 1999.

[8] 홍종준, "RTP를 위한 보안제어 프로토콜 구현, 한국컴퓨터정보학회 논문지 제8권 제3호, 2003.9.

[9] 노경택, "무선 비디오 통신을 위한 피드백 채널 기반의 에러복구 알고리즘의 개발", 한국컴퓨터정보학회논문지 *Journal of the Korea Society of Computer and Information 2002,1 v.007, n.002, pp.95-100 1229-9324

[10] Koren et al., "Enhanced Compressed RTP(CRTP) for Links with High Delay, Packet Loss and Reordering" RFC 3545, June 2001.

[11] C. Bormann, Ed., "Robust Header Compression (ROHC)" RFC 3095, June 2001.

[12] M. Degermark. "Evaluation of CRTP Performance over Cellular Radio Networks" IEEE Pers. Commun., vol.7, no. 4, Aug. 2000, pp. 20-25.

저 자 소 개

김 경 신

1993년 2월 연세대학교 대학원 전자공학과 졸업

2000년~현재 청강문화산업대학 컴퓨터네트워크과 교수

〈관심분야〉 근거리전산망, 라우팅프로토콜, 헤더압축

강 문 식

현재 강릉대학교 정보전자공학부 교수