목 차

그림 목차

표 목차

I. 서론

인터넷 사용자가 전세계적으로 급격히 증가하고, 네트워크 기반의 응용 프로그램이 다양하게 개발되고 사용됨에 따라 네트워크 트래픽이 급격히 증가하고 있다. 이러한 수요 증대에 맞추어 네트워크가 계속해서 증설되고 새로운 네트워크 기반 서비스들이 생겨나고 있다. 이런 상황에서 네트워크 상의 트래픽을 관찰하여 사용자들이 얼마나 트래픽을 발생시키고 어떤 서비스나 어플리케이션을 사용하는지에 대한 경향을 파악하는 일들이 요구된다. 따라서 네트워크상에 구성된 자원의 구체적인 사용 현황을 분석할 수 있는 정보를 제공하는 일은 네트워크 관리에 있어 상당히 중요한 일임에 틀림없다. 이처럼 네트워크 트래픽 모니터링은 네트워크의 현재 상황 및 문제점을 파악하고 네트워크 회선에 대한 계획 수립, 서비스 제공 평가 등을 위한 정보를 제공할 수 있다 [1][2][3].

최근의 네트워크 트래픽의 두드러진 추세 중 하나는 스트리밍이나 멀티미디어 컨퍼런싱과 관련된 트래픽의 급격한 증가이다. 본 논문에서는 이 두가지의 트래픽을 멀티미디어 서비스 트래픽이라 칭하기로 한다. 현재 많은 사이트에서 방송 프로그램, 영화, 음악 등의 스트리밍 컨텐츠를 제공하고 있고 이에 대한 사용자의 수요도 증가 추세이다. 또한 화상 회의와 같은 양방향성의 멀티미디어 전송 서비스도 도입되고 점차 널리 사용되어지고 있다. 이러한 경향은 인터넷 사용자 증가, 초고속 네트워크 보급, PC 성능 향상 등으로 인해 더욱 가속화되고 있다. 이렇게 멀티미디어 서비스 트래픽이 네트워크에 미치는 영향이 더욱 커지고 있고, 따라서 멀티미디어 트래픽 분석은 네트워크 이용 현황을 파악하는데 중요한 부분을 차지하게 된다.

그러나 아직까지 대부분의 기업이나 인터넷 서비스 제공 사업자(Internet Service Provider, ISP)들은 멀티미디어 서비스 트래픽에 대해서 사용자들이 어떤 서비스를 얼마나 사용하는지에 대한 정보를 파악하지

못하고 있다. 이것은 네트워크 트래픽이 대용량화 되고 기존의 텍스트나 이미지 위주의 트래픽이 멀티미디어 위주의 트래픽으로 변하고 있는 특성에 맞는 모니터링 및 분석 시스템이 존재하고 있지 않기 때문이다. 기존 대부분의 모니터링 시스템에 주로 쓰인 트래픽 분석 방법은 잘 알려진 포트 번호(well-known port)를 갖고서 그 상위 어플리케이션을 판단하던 것이었다. 그러나 멀티미디어 서비스의 경우 제어 프로토콜에서 동적으로 데이터를 전송할 세션에 대한 협상을 한 후 멀티미디어 서비스 컨텐트를 전송하게 된다. 따라서 멀티미디어 서비스를 전송할 때는 잘 알려진 포트 번호가 아니라 동적으로 할당된 프로토콜과 포트 번호를 사용하기 때문에 이러한 트래픽은 기존의 분석 방법으로 판단했을 경우 어플리케이션을 알 수 없는 트래픽으로 나타나게 되어 트래픽 분석이 어렵게 된다[3].

멀티미디어 서비스 트래픽을 분석함에 있어서 또 하나의 문제점은 프로토콜의 다양성과 비공개성이다. 현재 멀티미디어 서비스 플랫폼별로 사용하는 전송 방식과 프로토콜에 차이가 있고, 이들 중에는 각자 나름대로의 비공개 프로토콜과 전송 기술을 사용하는 곳이 있다. 따라서 멀티미디어 서비스와 그 사용 프로토콜별로 따로 조사와 분석을 해야만 하는 어려움이 있다.

본 논문에서는 이러한 문제를 해결하기 위해 플로우 기반의 일반 모니터링 시스템에 적용되어 응용될 수 있는 멀티미디어 서비스 트래픽의 모니터링 및 분석 방법을 제시한다. 모니터링 및 분석은 현재 가장 많이 쓰이고 있는 스트리밍 서비스, 멀티미디어 컨퍼런싱 및 IP 네트워크에서의 전화 서비스(VoIP)의 트래픽을 그 대상으로 한다. 이 방법들은 멀티미디어 서비스를 위한 제어 프로토콜과 관련된 패킷의 페이로드를 분석하여 멀티미디어 서비스 데이터를 전송하는데 사용될 프로토콜과 포트번호를 추출한다. 이 정보를 이용해 기존 방법에서는 어떤 어플리케이션인지 알 수 없었던 멀티미디어 서비스 트래픽을 판별할 수 있게 된다. 이 분석을 통해서 현재 네트워크에서의 멀티미디어 서비

스 트래픽 사용 현황을 파악할 수 있다. 또한 이러한 분석 방법을 이용한 시스템을 설계 및 구현하고, 시스템 적용 결과를 제시함으로써 제시된 분석 방법을 평가한다.

본 논문의 구성은 다음과 같다. 2장에서는 멀티미디어 서비스에 관련된 배경 지식과 관련 연구를 정리하고, 3장에서는 멀티미디어 서비스 트래픽 모니터링을 위한 분석 방법을 제안한다. 4장에서는 이러한 분석 방법을 적용할 시스템의 요구 조건과 이를 만족시키기 위한 시스템을 디자인하고, 5장에서는 실제 시스템을 구현해서 적용한다. 마지막으로 6장에서는 결론과 향후 과제에 대해서 설명한다.

II. 관련 연구

이 장에서는 우선 배경 지식으로 멀티미디어 서비스와 관련 프로토콜에 대해서 기술한다. 그리고 멀티미디어 서비스 트래픽 분석 방법을 적용시킬 플로우 기반의 네트워크 트래픽 모니터링 시스템에 대해서 살펴본다. 그 후 멀티미디어 서비스 트래픽을 모니터링 하기 위해 도입되었던 기존의 분석 방법을 소개한다.

2.1 멀티미디어 서비스 관련 프로토콜

이 절에서는 멀티미디어 서비스로 스트리밍과 멀티미디어 컨퍼런싱 서비스 두 가지를 소개하고, 이와 관련된 프로토콜들에 대해서 기술한다.

2.1.1 스트리밍 서비스

스트리밍은 전송되는 데이터를 마치 끊임없고 지속적인 물 흐름처럼 처리할 수 있는 기술을 의미한다. 스트리밍 기술을 이용하면, 파일이 모두 전송되기 전에라도 클라이언트 브라우저 또는 플러그인이 데이터의 표현을 시작할 수 있다. 스트리밍 기술은 인터넷의 성장과 함께 더욱더 중요해지고 있는데, 그 이유는 대부분의 사용자들이 가지고 있는 접속 회선 환경이 대용량 멀티미디어 파일들을 즉시 다운로드할 만큼은 빠르지 못한 대신 물 흐름처럼 흐르는 데이터를 받는 것은 가능하기 때문이다[5]. 현재 많은 스트리밍 서비스가 시도되고 실용화되어 왔지만, 그 중에서도 가장 많이 쓰이고 있는 서비스 플랫폼으로는 Windows Media Technology(WMT)[6], RealMedia[7], QuickTime[8]을 들 수 있다. 본 논문에서는 이 서비스들의 트래픽을 모니터링 대상으로 한다. 표 1은 이 서비스들에서 이용되는 프로토콜들을 보여준다.

스트리밍 서비스 플랫폼	제어 프로토콜	데이터 전송 프로토콜	제안기관
RealMedia	RTSP	RDT	RealNetworks
QuickTime	RTSP	RTP	Apple
Windows Media Technology	MMS	MMST/MMSU	Microsoft

표 1 스트리밍 서비스 이용 프로토콜

표 1에서 제어 프로토콜은 스트리밍 서비스에 있어서 서비스에 대한 요구, 응답이나 서비스 연결 설정, 스트림 재생과 관련된 각종 제어를 담당한다. 현재 사용되고 있는 제어 프로토콜로 RTSP(Real Time Streaming Protocol)[9]와 MMS(Microsoft Media Server)[6]를 들 수 있다. RTSP는 인터넷 상에서 스트리밍 데이터를 제어하는 방법에 대한 표준안으로 미국 컬럼비아 대학과 넷스케이프 및 RealNetworks 등에 의해 수행된 작업으로부터 비롯되었으며, IETF(Internet Engineering Task Force)에서 표준으로 지정되었다. 반면 MMS는 Microsoft사에서 스트리밍 서비스를 위해 독자적으로 개발해서 사용하고 있는 제어 프로토콜로 그 동작과 스펙이 비공개되어 있다.

실제로 스트리밍 데이터 전송을 담당하는 것은 표 1에서 보이는 데이터 전송 프로토콜이다. RTP(Real Time Protocol)[10]는 오디오와 비디오와 같은 실시간 데이터를 전송하기 위한 인터넷 프로토콜로서 일반적으로 UDP 프로토콜 상에서 실행된다. IETF에서 표준화가 되었고, 산업계의 광범위한 지지를 받고 있다. RDT(RealNetworks Data Transport)[7]도 실시간 스트리밍 데이터 전송을 위한 프로토콜로 RealNetworks사에서 자체적으로 개발되어 RealNetworks[7] 서비스에

사용되고 있다. MMS가 스트리밍 서비스를 제어할 때는 MMST(MMS over TCP)[6]나 MMSU(MMS over UDP)[6]를 데이터 전송 프로토콜로 사용한다. 이 프로토콜들은 Microsoft사에서 자체적으로 개발하고, 독점적인 사용을 하고 있는 비공개 프로토콜이다.

2.1.2 멀티미디어 컨퍼런싱 서비스

이 절에서는 멀티미디어 컨퍼런싱 및 IP 기반 전화 서비스 어플리케이션들이 기반으로 하는 H.323[10]과 SIP(Session Initiation Protocol)[12]에 대해서 설명한다.

H.323은 멀티미디어 컨퍼런싱 데이터를 TCP/IP와 같은 패킷 교환 방식의 네트워을 통해 전송하기 위한 ITU-T(International Telecommunications Union - Telecommunication Standardization Sector)의 표준이다. 이 서비스에 있어서 종단 요소(Terminal Entity) 사이의 호 설정(Call setup)과 접속해제(teardown)를 위한 시그널링 프로토콜(signaling protocol)이 Q.931[10]이다. Q.931에 의해 호출 설정이 되고 난 후에는, H.245[10]가 연결 제어(connection control)를 위해 사용된다. H.245는 시스템 사이의 제어정보를 교환하기 위한 프로토콜로 두 양 종단(Endpoint) 사이의 주종(master-slave) 관계를 결정하고, 두 종단 사이의 논리적 채널(logical channel)을 열고 닫는다. 이렇게 열린 논리 채널을 통해 오디오, 비디오, 데이터 등의 미디어 스트림들이 전달된다. 이러한 데이터들은 TCP/IP를 기반으로 하여 실시간 전송을 위한 RTP 프로토콜 상위에서 전송되어진다.

SIP는 ITU-T의 H.323에 대응되는 프로토콜로서 멀티미디어 컨퍼런싱 서비스에서 단말간 또는 사용자들간에 호를 설정(call setup)하는 프로토콜이다. SIP는 매우 간단한 텍스트 기반의 응용계층 제어 프로토콜로서, 하나 이상의 참가자들이 함께 세션을 만들고, 수정하고 종료할 수 있게 한다. SIP는 텍스트 기반의 HTTP(Hypertext Transfer Protocol)[13]

를 본받아 확장 시켜서 설계된 것으로, 클라이언트가 호출을 시작하면 서버가 그 호출에 응답을 하는 클라이언트-서버 구조에 기반을 두고 있다. SIP는 이러한 기존의 텍스트 기반 인터넷 표준들에 따름으로써, 고장 수리와 네트웍 디버깅 등이 쉽다[14].

2.1.3 멀티미디어 서비스 데이터 전송 과정

멀티미디어 서비스를 위해 두 종류의 세션이 형성된다. 하나는 서비스 동안 연결 설정이나 데이터 전송에 있어서의 제어를 담당하는 세션이다. 스트리밍 서비스에서는 RTSP나 MMS와 같은 제어 프로토콜을 사용하고, 멀티미디어 컨퍼런싱 서비스에서는 SIP, Q.931이나 H.245 세션을 이용하게 된다. 이 때 이용되는 세션을 본 논문에서는 *제어 세션 (control session)*이라고 부르기로 한다. 나머지 하나는 RDT, RTP, MMST/MMSU와 같은 전송 프로토콜을 사용해서 서버에서 클라이언트에게 혹은 양 종단간에 실제로 멀티미디어 데이터를 전송하는 세션이다. 이후 이 세션을 *데이터 세션(data session)*이라 칭한다.

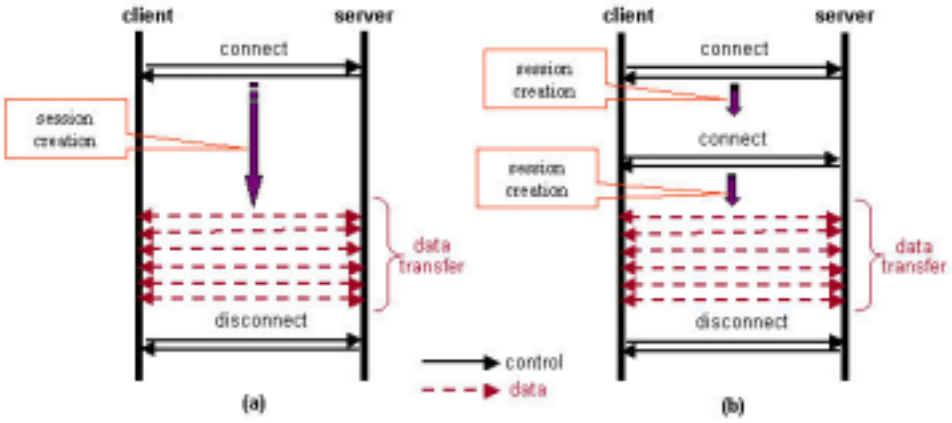


그림 1 멀티미디어 서비스 제어 및 데이터 세션

그림 1은 제어 세션과 데이터 세션이 생성되어 데이터가 전달되

는 과정을 보여준다. 우선 처음에는 잘 알려진 포트번호를 통해서 제어 세션이 생성된다. 이 때 (a)의 스트리밍 서비스나 SIP 기반 서비스의 경우에는 한 개의 제어 세션이 생기지만, (b)의 H.323 같은 경우에는 제어 세션이 한 단계 더 생긴다. 즉, 호출자와 응답자 사이에 호 설정(call setup) 메시지가 교환되고 난 후 생성된 첫번째 제어 세션에서 자신이 전송 받을 포트 번호를 상대방에게 알림으로써 두 번째 제어 세션을 생성시킨다. 이렇게 제어 세션이 생성되고 나면 제어 프로토콜을 이용해 전송 프로토콜과 포트 번호 등과 같이 멀티미디어 데이터 전송에 대한 사항들을 협상한 후 데이터 세션을 생성시켜 멀티미디어 데이터를 전송하게 된다. 여기서 본 논문은 (a), (b)의 데이터 세션이나 (b)의 두 번째 제어 세션과 같이, 이전에 성립된 제어 세션의 협상에 의해서 동적으로 할당된 전송 프로토콜과 포트 번호를 이용하는 세션을 *다이나믹 세션(dynamic session)* 이라고 부르기로 한다.

이 때, 제어 세션에 의해 다이나믹 세션의 정보를 협상할 경우 제어 세션 패킷의 페이로드에는 새로운 세션에서 사용할 전송 프로토콜이나 포트 번호 등의 협상 내용이 나타나게 된다. 따라서 이러한 제어 세션 패킷의 페이로드를 분석함으로써 다이나믹 세션에 대한 정보를 찾아낼 수 있게 된다.

2.2 플로우 기반 네트워크 모니터링 시스템

이 절에서는 본 논문에서 제시할 멀티미디어 서비스 트래픽 분석 방법이 적용될 플로우 기반 모니터링 시스템에 대해서 살펴보기로 한다.

2.2.1 플로우의 개념

플로우(flow)란 특정 두 종단(end points)간에 지나다니는 일련의 패

킷(packet)들을 모아 부르는 것으로[15], 일반적으로 일정 시간 동안 근원지 IP 주소, 목적지 IP 주소, 근원지 포트 번호, 목적지 포트 번호 및 전송 프로토콜 번호 5개 항목이 같은 패킷들의 집합으로 정의된다[16].

플로우 기반의 모니터링 시스템이 개발되고 각광을 받는 이유는 데이터의 압축성에 있다. 최근 네트워크 트래픽의 대용량화 추세에 맞추어 보았을 때 패킷 단위로 분석을 하고 저장하는 것은 트래픽 처리에 있어서 많은 부담이 된다. 따라서 같은 플로우에 속하는 패킷들을 하나의 플로우로 모아서 분석하고 저장함으로써 처리해야 할 데이터를 줄일 수 있다. 압축의 정도는 패킷들을 모아 하나의 플로우로 만들어 두는 타임 아웃 값에 따라 다양하게 변한다[17]. 포항공대 인터넷 회선에서의 적용 결과 약 18:1 정도의 압축 비율을 얻을 수 있었다[18].

이렇게 패킷들을 모아 플로우 정보를 얻는 방법은 크게 두 가지로 나뉘어진다. 첫째는 라우터로부터 생성된 플로우 정보를 받아오는 방법이다[19]. 라우터가 패킷의 전송 이동 경로를 결정하는 도중 패킷 헤더 정보를 저장하면서 플로우로 모아두었다가 일정시간마다 특정 시스템으로 보내주게 된다. 모니터링 시스템에서 패킷으로부터 플로우 정보를 모으는데 필요한 부담을 줄일 수 있다는 장점이 있다. 그러나 이러한 방법은 라우터로부터 도움을 받아야만 한다는 제한 사항이 있다. 따라서 이러한 기능을 라우터로부터 지원받지 못할 수도 있고, 라우터의 이러한 기능을 이용하기 위한 비용도 고려되어야 한다. 또한 라우터가 플로우 정보를 만들고 그 정보를 다른 시스템에 보내는 부담을 떠맡게 된다. 따라서 두번째로, 모니터링 시스템 자체에서 네트워크 링크를 지나다니는 패킷을 수집해서 패킷 헤더로부터 필요한 정보를 추출해내고 관련 패킷 정보를 모아 직접 플로우 정보를 생성해내는 방법이 이용되기도 한다[18][20][21].

2.2.2 넷플로우 기반 모니터링 시스템

현재 라우터로부터 플로우 정보를 받아오는 가장 대표적인 것으로는 넷플로우(NetFlow)[19]를 들 수 있다. 이것은 시스코 계열의 라우터(Cisco Router)로부터 받아올 수 있는 플로우 정보 포맷이다. 비록 표준화되지는 않았지만, 그 효용성으로 인해 많은 모니터링 시스템에서 이러한 기능 지원을 지원하고 있거나 고려중이다. 이 절에서는 넷플로우를 이용하는 대표적인 모니터링 시스템들을 간단히 소개하기로 한다.

2.2.2.1 FlowScan

FlowScan[15]은 넷플로우로부터 얻은 플로우 정보를 수집, 분석하여 그 결과를 보여주는 대표적인 플로우 기반의 모니터링 시스템 패키지이다. CAIDA(Cooperative Association for Internet Data Analysis)[22]에서 만들어진 이 패키지 안에는 플로우 정보를 이용하기 위해 다음과 같은 몇 가지 툴들을 가지고 있다. cflowd[23]는 기본적으로 수집 기능만 담당하며 넷플로우로부터 얻은 플로우 정보를 수집하여 파일로 저장한다. 또한 별도의 툴을 이용해 몇 가지 정해진 통계 정보를 분석하여 텍스트 형태로 얻어낼 수도 있다. flowscan은 프리젠테이션 모듈을 실행함으로써 미리 정해진 형태의 통계 자료를 그래프로 그려낼 수 있다. 그리고 RRDtool[24]은 데이터들을 요약 정리해서 장기간 동안의 트래픽 데이터를 분석 가능한 형태로 생성시켜 준다.

2.2.2.2 CISCO 상업용 툴

넷플로우를 이용하기 위해 CISCO사에서 개발해서 내놓은 몇 가지 이용 가능한 상업용 툴들이 있다.

NetFlow FlowCollector[19]는 넷플로우 트래픽을 라우터로부터 수집하고 가공하여 저장하는 역할을 한다. 저장된 데이터는 NetFlow Data Analyzer[19]와 같은 다른 넷플로우 클라이언트 응용(NetFlow Client Application)에 의해 사용될 수 있다. 플로우 데이터를 사용자가 원하는

방향으로 다양하게 정리 및 통합(aggregation)해서 원하는 정보를 정확, 간결하게 저 용량으로 저장할 수 있다.

NetFlow Data Analyzer는 FlowColletor에 의해 수집, 통합, 저장된 넷플로우 데이터를 받아와서 사용자에게 시각적으로 보여준다. FlowCollector에게서 데이터를 얻어오는 형식으로 작동 하므로 FlowCollector에서 어떻게 설정을 하여서 데이터를 수집했는지가 중요하다. 여러 가지 통합(Aggregation)으로 데이터를 볼 수 있게 하며 시간 별 추이, 찾기 등의 특수한 기능도 제공 한다.

2.2.3 NG-MON

이 절에서는 라우터의 도움없이 패킷들을 직접 수집하여 플로우 정보를 만드는 모니터링 시스템의 하나로 NG-MON[18]을 소개한다.

NG-MON은 플로우에 기반해서 IP 네트워크 트래픽을 모니터링하기 위한 시스템이다. 대용량 네트워크 트래픽을 처리하기 위해 트래픽 모니터링 단계를 5 단계로 구분한 후, 각 단계를 파이프 라인 구조로 연결하였다. 각 단계마다 한 대 이상의 시스템에서 수행이 가능하도록 병렬 구조를 적용하였다. NG-MON의 전체 구조는 다음과 같다.

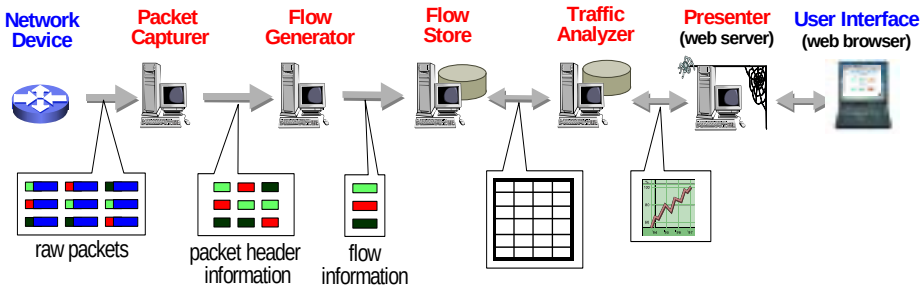


그림 2 NG-MON 전체 구조

그림 2 에서 보는 바와 같이 플로우 정보 생성을 위해 패킷 수집,

플로우 생성, 플로우 저장의 세 단계로 나누어 처리하도록 하였다. 그리고 이러한 플로우 정보에 기반해 트래픽을 분석하는 트래픽 분석 단계와 프레젠테이션 단계를 통해 트래픽 정보로 가공되어 사용자에게 분석 결과를 보여준다. 각 단계를 각기 다른 하나의 시스템이나, 혹은 로드를 감당하지 못할 경우에는 여러 시스템에 설치되어, 병렬로 동시에 수행 가능한 유연성을 제공함으로써 전체 로드를 분산할 수 있다.

2.3 기존 멀티미디어 서비스 트래픽 모니터링 분석 방법

기존의 네트워크 트래픽 모니터링 시스템에서 멀티미디어 서비스 트래픽을 모니터링하기 위한 시도들이 있었다. 이 절에서는 그러한 시스템에서 멀티미디어 서비스 트래픽을 판별하기 위해 사용한 분석 방법을 소개한다.

2.3.1 추측법

어플리케이션을 알 수 없는 트래픽 중에서 멀티미디어 서비스 트래픽임을 추측하는 방법은 Flowscan[15]이나 Fluxoscope[25]에서 사용되었다. 이것은 수집된 패킷에서 멀티미디어 서비스 제어와 관련된 잘 알려진 포트 번호가 감지되면 그 이후 두 호스트 사이에 지나다니는 트래픽 중에서 어플리케이션을 판단할 수 없는 트래픽들을 모두 멀티미디어 서비스 트래픽이라고 간주하여 분석하는 방법이다. Flowscan이나 Fluxscope의 본 목적은 멀티미디어 서비스 트래픽뿐만 아니라 네트워크의 전체 트래픽을 분석하는데 있다. 그러므로 멀티미디어 서비스 트래픽 모니터링을 위해 부가적인 처리과정을 거치지 않고 분석시에만 멀티미디엇 서비스 트래픽이라고 추측을 할 뿐이다. 이러한 방법은 멀티미디어 서비스 트래픽 분석을 위한 별도의 처리 오버헤드가 없이 쉽게 분석을 할 수 있다는 장점이 있다. 그러나 두 호스트 사이에 멀티미디어 서비스와 관련된 포트 번호가 감지되고 난 후에 두 호스트 사이에

발생하는 트래픽 중에는 멀티미디어 서비스 트래픽이 아닌 경우도 있다. 그리고 이 방법은 처음에 쓰였던 서버와는 다른 서버를 이용해 멀티미디어 데이터를 전송할 경우에도 적용될 수가 없다. 따라서 멀티미디어 서비스와 관련된 잘 알려진 포트 번호를 가진 패킷이 지나가고 나서 이후에 발생되는 트래픽이 멀티미디어 서비스라는 가정 때문에 분석 결과가 부정확한 단점이 있다.

2.3.2 선택적 멀티미디어 트래픽 패킷 수집 방법

2.3.1에서 제시되었던 분석 방법과는 달리 mmdump[26]에서는 멀티미디서 서비스 트래픽을 판별하기 위해 별도의 처리 과정을 두고 있다. 여기서는 패킷 수집 필터를 변경해가며 멀티미디어 서비스 관련 패킷만 수집해서 분석하는 방법이 이용된다. 이 때 필터에는 수집될 패킷의 포트 번호가 등록되어 있어서, 등록된 포트 번호를 가진 패킷만 수집된 후 분석할 수 있다. 이를 위해서 mmdump는 제어 세션 패킷들을 분석해서 데이터 세션을 위해 새롭게 할당된 포트 번호를 찾아낸다. 이후 새로 찾아낸 포트 번호에 해당하는 패킷들을 잡아낼 수 있도록 수집 필터를 바꾸어 적용시킨다. 이 방법은 다이나믹 세션에 사용되어질 정보를 찾아냄으로써 멀티미디어 데이터 전송 관련 트래픽을 판별할 수 있다는 점에서 의의가 있다. 그러나 이러한 방법을 응용하는데 있어서 다음과 같은 문제점들을 가진다.

첫째는 모니터링 대상에 있어서 한계점이 있다. mmdump는 현재 스트리밍 서비스에 많이 이용되는 있는 Windows Media Technology 서비스 트래픽은 분석하지 않았다. 이 서비스에 이용되는 프로토콜이 비공개되어 있어서 제어 세션을 정확하게 분석 하기가 어렵기 때문이다. 그러나 더욱 근본적인 문제는, 만일 다이나믹 세션의 정보를 알아내었다고 해도 이 정보를 이용할 수 없는 경우가 생기는 데 있다. 이 선택적 패킷 수집 방법에서 필터는 등록된 포트 번호를 가진 패킷들만을 잡아

낸다. 그런데 Windows Media Technology에서는 응용 계층에서 네트워크 전송을 할 때 한번에 많은 메시지를 보내게 되므로, 데이터 전송시에 IP 단편화(IP fragmentation)가 발생하게 된다. 관찰 결과 Windows Media Technology 데이터 전송에 쓰이는 패킷 중에서 약 30~70%의 IP 단편화 된 패킷들을 발견할 수 있었다. 이러한 패킷들은 그 자체만을 보고서는 포트 번호를 확인할 수가 없기 때문에 필터에서 인식을 하지 못하고 버려지게 된다. 이것은 비단 Windows Media Technology뿐만 아니라 데이터 전송 시에 IP 단편화가 발생하는 경우 모두에 발생할 수 있는 문제이므로, 이러한 서비스의 경우에는 트래픽 모니터링을 할 수가 없게 된다.

둘째는 정확성의 문제이다. 이 방법은 수집하고자 하는 패킷만 잡아내도록 필터를 수시로 변경한다. 그런데 새로운 세션이 생성되고 사라지게 됨에 따라 빈번하게 필터를 새로 컴파일해서 변경하는 동안에는 네트워크를 지나가는 패킷을 수집할 수가 없다. 이 지연 시간동안 새로 생겨나는 세션의 패킷이나, 다이나믹 세션 정보를 담고 있는 제어 세션 패킷이 지나갈 수 있다. 그리고 서비스가 끝났다고 판단되어 그와 관련된 패킷을 캡쳐하지 않도록 필터를 변경한 후에도 라우팅 상의 문제로 데이터 세션의 패킷이 지나갈 경우에는 그 패킷은 수집되지 않는다. 이럴 경우 이 트래픽들에 대한 모니터링을 할 수가 없게 된다. 따라서 네트워크 상에 지나다니는 멀티미디어 서비스 트래픽을 판별해서 정보를 파악하기에는 부정확하거나 부족한 면이 있다.

셋째로 이 방법을 일반 모니터링 시스템에 그대로 적용시키기에는 난항들이 존재한다. 일반적으로 대부분의 트래픽 모니터링 시스템들은 네트워크 사용 현황을 파악하기 위해 모두 혹은 임의로 선택된 패킷들을 수집해서 분석한다. 따라서 특정 포트 번호에 해당되는 패킷들만을 골라서 수집하고 분석할 경우에는 다른 종류의 트래픽 정보는 수집할 수 없다. 만일 모든 포트 번호에 대해서 수집을 한다고 했을 경우에는 시스템에 많은 부하를 줌으로써 일반적인 트래픽을 모니터링까지

어렵게 할 수 있다. 왜냐하면 다이나믹 세션 정보를 추출하기 위해서는 패킷의 페이로드까지 잡아서 분석해야 하기 때문이다. 대부분의 모니터링 시스템에서 시스템 부하를 줄이기 위해 패킷의 헤더만 잡아서 분석하는 실정을 고려했을 때, 모든 포트 번호의 패킷 전체를 수집하는 것은 시스템에 부담이 된다.

III. 멀티미디어 서비스 트래픽 모니터링 및 분석 방법

이 장에서는 플로우 기반 네트워크 모니터링 시스템에 적용될 수 있는 멀티미디어 서비스 트래픽 모니터링 및 분석 방법을 제시한다. 이를 위해 패킷을 수집하고 처리하는 과정이 그림 3에 나타나 있다. 전체적인 과정은 다음과 같이 크게 세 단계로 나누어 살펴볼 수 있다. 플로우 생성(Flow Generation) 과정은 패킷 헤더로부터 추출된 정보를 모아 플로우를 만들고, 이와 동시에 세션 분석(Session Analysis) 과정에서는 다이나믹 세션 플로우를 판단할 때 필요한 정보를 제공하기 위해 제어 세션의 패킷을 분석한다. 이렇게 만들어진 이 두 종류의 정보를 바탕으로 멀티미디어 서비스 트래픽 분석 과정(Multimeida Service Traffic Analysis)에서는 멀티미디어 서비스 관련 플로우를 판별한다.

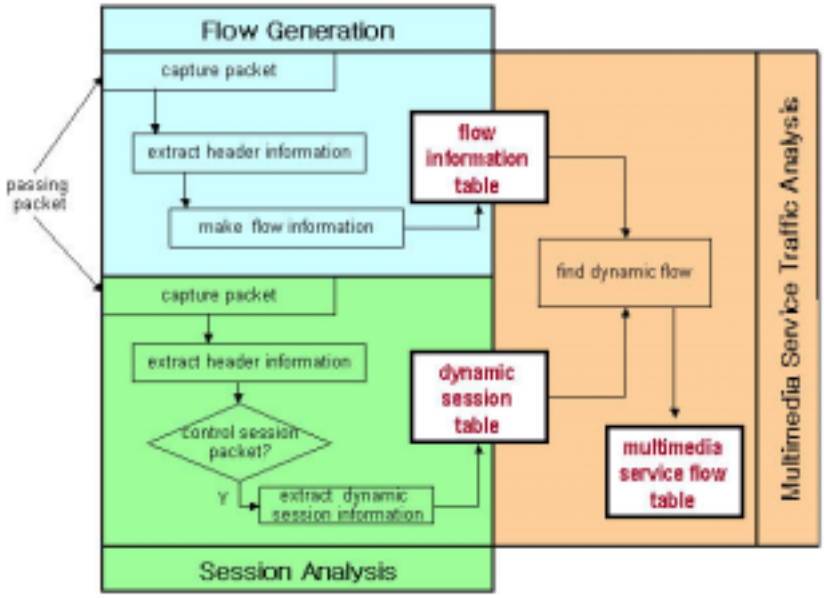


그림 3 패킷 수집 및 처리 과정

3.1 플로우 기반 정보 생성

일반적인 플로우 기반 모니터링 시스템은 패킷을 분석하여 플로우 정보를 저장해 둔다. 제시된 분석 방법에서는 이렇게 일반 모니터링 시스템으로부터 생성된 플로우 정보를 그대로 이용한다. 따라서 플로어 생성 과정에 대한 기본 과정만 간단히 언급하고자 한다.

일반 모니터링 시스템에서는 네트워크 링크를 지나는 패킷을 수집하면 우선 패킷 헤더를 분석해서 트래픽 분석에 필요한 정보를 추출한다. 추출된 패킷의 헤더 정보를 이용해 일정 시간 동안 같은 플로우에 속하는 패킷끼리 정보를 모으고 통합한다. 즉, 플로우를 결정짓는 5개 항목이 같은 플로우가 있는지 살펴서, 해당 플로우에 패킷이 추가될 때마다 그 플로의 패킷 수와 플로우 길이를 갱신시킨다. 그러나 이 패킷이 속하는 플로우가 없다면 수집된 패킷으로 새로운 플로우를 만들어 두고 추가할 패킷이 올 때마다 플로우 정보를 갱신하다. 일정 시간이 지나면 모여진 플로우 정보들을 데이터베이스나 파일 형태로 저장하고, 다시 새롭게 플로우 정보를 만드는 과정을 반복한다.

3.2 다이나믹 세션 정보 분석

다이나믹 세션 패킷들을 모아 플로우를 만들 경우에는 포트번호만을 가지고서는 그 어플리케이션을 알 수가 없다. 왜냐하면 다이나믹 세션은 잘 알려진 포트 번호를 사용하는 것이 아니라 동적으로 할당된 포트 번호를 사용해 데이터를 전송하기 때문이다. 따라서 다이나믹 세션 플로우들을 분석하기 위해서는 포트 번호 이외에 추가적인 정보가 필요하게 된다. 이러한 다이나믹 세션 정보를 찾아내기 위한 과정이 다음 각 절에서 기술된다.

3.2.1 제어 세션 패킷의 선택 및 분석

다이나믹 세션에 대한 정보는 제어 세션 패킷의 페이로드에서 찾

아낼 수 있다. 따라서 수집된 패킷 중에서 제어 세션의 패킷을 골라내고, 각 제어 프로토콜에 따라서 패킷의 페이로드를 분석해 다이나믹 세션 정보를 찾아야 한다. 그림 4에는 이렇게 다이나믹 세션 정보를 찾고 그 정보를 저장하는 알고리즘이 기술되어 있다.

```
procedure SessionAnalysis
    ( var Msg : CONTROL SESSION MESSAGE,
      var Table : SET of DYNAMIC SESSION RECORD )
  var
    result : boolean ;
    dsRec : DYNAMIC SESSION RECORD ;
    Q931Rec : DYNAMIC SESSION RECORD ;
  begin
    if protocol of Msg =RTSP then
      if TCP Flag.FIN of Msg is set  then  goto DELETE_SESS;
      else result := ParseRTSP (Msg, dsRec);
    else if  protocol of Msg =MMS then
      if TCP Flag.FIN of Msg is set  then  goto DELETE_SESS;
      else result := ParseMMS (Msg, dsRec);
    else if  protocol of Msg =Q.931 then
      if. TCP Flag.FIN of Msg is set  then  goto DELETE_SESS;
      else result := ParseQ931 (Msg, dsRec);
    else
      if  DYNAMIC SESSION RECORD from Msg ∈ Table  then
        Q931Rec := DYNAMIC SESSION RECORD in Table;
        if  TCP Flag.FIN of Msg is not set  then  goto DELETE_SESS;
        else
          Q931Rec := DYNAMIC SESSION RECORD ;
          result := ParseH245 (Msg, dsRec, Q931Rec);
      else  goto END_ALGO;
    if result = TRUE then  add to dsRec to Table ;
    goto END_ALGO;
    DELETE_SESS:
        set FIN time of  DYNAMIC SESSION RECORD from Msg in Table;
    END_ALGO:
end ; {SessionAnalysis }
```

그림 4 제어 세션 패킷의 선정 및 분석

그림 4에서 CONTROL SESSION MESSAGE는 수집된 패킷의 헤더

정보와 트랜스포트 계층의 페이로드를 말한다. 본 논문에서는 이 정보를 합쳐 *제어 세션 메시지*라 칭한다. 이 메시지로부터 패킷의 어플리케이션 계층의 프로토콜을 판단하고, 제어 세션의 경우에는 페이로드로부터 다이나믹 세션 정보를 찾아낸다. 그리고 DYNAMIC SESSION RECORD는 다이나믹 세션에 대한 정보를 찾았을 때 해당 정보를 담을 레코드이다. 이 메시지 타입에 대한 정의가 그림 5에 기술되었다.

```
type
    CONTROL SESSION MESSAGE = record
        source address : integer ;
        destination address : integer ;
        source port : integer ;
        destination port : integer ;
        transport portocol : integer ;
        TCP Flag : integer ;
        time stamp : integer ;
        payload : array of [1..1500] of char ;

    DYNAMIC SESSION RECORD = record
        data client address : integer ;
        data client port : integer ;
        control server address : integer ;
        control client address : integer ;
        transport portocol : integer ;
        data application : integer ;
end
```

그림 5 다이나믹 세션 관련 메시지 타입

우선 수집된 패킷이 제어 세션에 관련된 패킷인지를 골라내어야 한다. 대개의 제어 세션 패킷의 경우는 잘 알려진 포트 번호를 사용하기 때문에, 제어 세션에 사용되는 잘 알려진 포트 번호를 등록해 놓고, 등록된 포트 번호를 이용하는 패킷을 골라내면 된다. 현재 RTSP, MMS, SIP, H.323의 Q.931가 이 경우에 해당한다. 그러나, 그림 1의 (b)처럼 제어 세션 역시 다이나믹 세션일 수도 있기 때문에 제어 세션 패킷 중에는 잘 알려진 포트 번호만으로는 찾을 수 없는 것도 있다. 따라서 잘

알려진 번호에 속하지 않은 포트 번호를 가진 패킷의 경우에는 활성화된 다이나믹 세션들의 정보를 참고해 비교함으로써 제어 세션 패킷인지를 파악해야 한다. H.323의 H.245가 이 경우에 해당된다.

이렇게 제어 세션 패킷을 선정한 후에는 먼저 메시지에서 TCP FIN flag가 설정되어 있는지를 살핌으로써 세션 연결을 해제하는 패킷인지를 판단한다. 연결 해제 패킷에는 다이나믹 세션 정보가 들어있지 않으므로, 그 이외의 패킷에 대해서만 페이로드를 분석해서 다이나믹 세션 정보를 찾아본다. 이 때 각 제어 프로토콜별 다이나믹 세션 형성 과정과 분석 방법에 대해서는 다음 절에서 자세히 기술하도록 한다.

분석 결과 다이나믹 세션에 대한 정보를 찾게 되면 이 정보들을 현재 활성화되어 있는 다이나믹 세션 정보를 갖고 있는 $Table$에 저장한다. 그러나 세션이 끝났을 경우에는 이 정보는 필요 없는 정보가 되므로 $Table$에서 제거되어야 한다. 따라서 처음 TCP flag 확인 결과 세션을 종료하는 제어 세션 메시지이면 다이나믹 세션 테이블에 있는 세션 정보의 FIN time을 설정해줌으로써 세션이 종료되었음을 알린다. 이를 통해 활성화된 다이나믹 세션 정보를 유지할 수 있다.

그러나 이러한 연결 해제 과정이 없거나 종료 패킷이 손실된 경우가 발생하게 되면 FIN 플래그(flag)를 확인할 수 없게 된다. 이런 점을 대비하기 위해 주기적으로 데이터 전송 세션 정보들을 체크해서 일정 시간동안 패킷이 교환되지 않은 세션은 제거시킴으로써 세션 정보를 관리할 수 있다. 따라서 이 테이블의 내용을 참조함으로써 현재 활성중인 다이나믹 세션들에 대한 정보를 파악할 수 있고 이를 바탕으로 멀티미디어 서비스 트래픽을 판별할 수 있다.

3.2.2 제어 프로토콜 별 정보 추출 방법

이 절에서는 각 프로토콜별로 세션 형성 과정과 패킷의 페이로드로부터 다이나믹 세션 정보를 찾아내는 방법에 대해 설명하도록 한다.

3.2.2.1 RTSP

그림 6은 제어 프로토콜로 RTSP가 이용되었을 때 스트리밍 서비스를 위해 연결이 설정되고 데이터가 전송되는 과정을 나타낸 것이다.

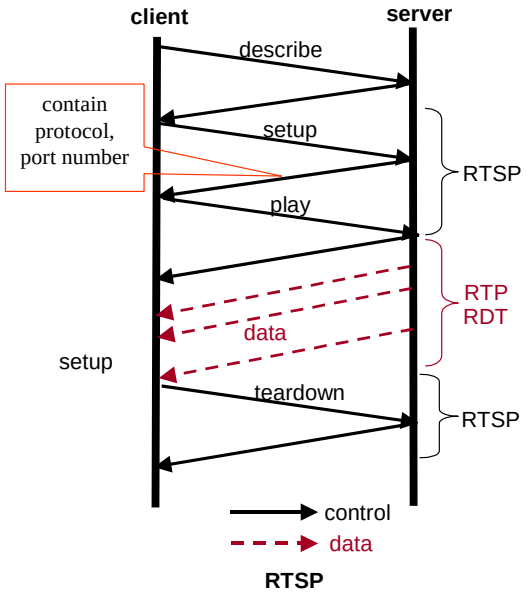


그림 6 RTSP를 이용한 데이터 전송 과정

우선 클라이언트가 describe 메시지를 통해 서비스 받기를 원하는 멀티미디어 스트림을 요구한다. 그러면 서버는 이 스트림에 대한 인코딩 방식이나 평균 전송 속도 등의 정보를 담아서 응답을 보낸다. 다음에는 setup 메시지에 클라이언트가 데이터를 받을 수 있는 여러 쌍의 전송 프로토콜과 포트 번호를 담아 서버에게 요청한다. 이 요청을 받은 서버는 여러 쌍의 전송 프로토콜과 포트 번호 중에서 하나를 골라 응답 메시지에 실어 클라이언트에게 알려주게 된다. 이러한 정보 교환 후에 클라이언트가 스트리밍을 시작하는 play 요구를 하고 서버에서는 선택된 클라이언트의 포트 번호로 데이터를 전송하는 서비스를 제공한다.

서비스 제공 중에 pause나 stop과 같은 스트림 재생과 관련된 제어를 서버에게 요청할 수 있다. 이러한 스트리밍 서비스가 끝나게 되면 teardown 요청을 보내 연결을 해제한다.

```
Procedure ParseRTSP
    ( var Msg : CONTROL SESSION MESSAGE,
      vardsRec : DYNAMIC SESSION RECORD)
  var
    msgHeader : array[1..256] of character ;
    result : boolean ;
  begin
    if source port of Msg = RTSP server port number then
      msgHeader := read a line from Msg;
      if  msgHeader = RESPONSE MESSAGE   then
        while   (there is a line to read in Msg ) do begin
          msgHeader := read a next line from Msg;
          if msgHeader = TRANSPORT MESSAGE   then
            data client port of dsRec := PARSED PORT;
            transport protocol of dsRec := PARSED PROTOCOL;
            result : =TRUE;
          else if msgHeader = SERVER MESSAGE   then
            data application of dsRec := PARSED APPLICATION;
        end
    if result = TRUE then
      data client address of dsRec := destination address  of Msg;
      control server address of dsRec := source address of Msg;
      control server port of dsRec := source port of Msg;
      control client address of dsRec := destination address of Msg;
      control client port of dsRec := destination port of Msg;

 end ; {ParseRTSP}
```

그림 7 RTSP 다이나믹 세션 분석 방법

위에서 기술된 바와 같이 제어 세션에 의해 협상된 데이터 세션에 대한 정보가 setup 요구와 응답 메시지에 담겨져 있게 된다. 그런데, setup 요청에는 클라이언트가 스트리밍 데이터를 받을 수 있는 여러 쌍의 전송 프로토콜과 포트번호가 들어있다. 그러므로 setup 요청만을 분석하는 것은 스트리밍 데이터가 정확히 어떤 전송 프로토콜의 포트 번

호을 통해 전달되는지 알 수 없다. 따라서 setup 요구에 대한 응답 메시지를 분석함으로써 데이터 전달에 쓰이는 정확한 전송 프로토콜과 포트번호를 찾아낼 수 있게 된다. 이렇게 서버의 응답 메시지로부터 다이나믹 세션 정보를 찾아내는 방법이 그림 7에 기술되었다.

메시지의 각 내용들은 라인단위로 구분되어 있기 때문에 라인단위로 나누어 읽어서 메시지 내용을 분석하게 된다. 우선 성공적인 응답 메시지를 골라내기 위해서 근원지 포트 번호(source port)가 RTSP의 포트번호인 554인지를 확인하고 한 라인을 읽어서 RESPONSE MESSAGE의 표현이 있는지 확인한다. 표 2 에는 RESPONSE MESSAGE와 그 외 페이로드 분석에 사용되는 메시지의 정규 표현식을 담고 있다.

RTSP RESPONSE MESSAGE
“[Rr][Tr][Ss][Pp]/[0-9]+\.[0-9]+ 200 OK.*”
SERVER MESSAGE
“[Ss][Ee][Rr][Vv][Ee][Rr]: [a-zA-Z]+/]/[0-9]+\.[0-9]+”
TRANSPORT MESSAGE
“[Tt][Rr][Aa][Nn][Ss][Pp][Oo][Rr][Tt]: [a-zA-Z]+([/-][a-zA-Z])*.* ;client_port=[1-9][0-9]{3,4}+(-[1-9][0-9]{3,4}){0,1};.*\n”

표 2 RTSP 메시지의 정규 표현식

성공적인 서버 응답 메시지를 골라낸 후에는 페이로드의 끝부분까지 라인단위로 메시지를 읽으면서 어플리케이션이나 데이터 세션에 대한 정보를 찾는다. 어플리케이션의 경우에는 SERVER MESSAGE가 있는지를 살펴서 RTSP를 사용하는 어플리케이션이 무엇인지를 확인한다. 데이터 전송과 관련된 정보는 transport 헤더에서 찾는다. Transport 헤더

는 “Transport: (전송프로토콜);(클라이언트포트);(서버포트)” 형태로 되어 있는데 TRANSPORT MESSAGE의 표현식을 갖는지 확인하고 전송 계층의 프로토콜과 포트 번호를 찾아낼 수 있다.

분석 결과 다이나믹 세션 정보를 찾아낸 경우에는 이를 저장한다. 이 메시지가 서버로부터 오는 패킷이므로 제어 및 데이터 전송의 서버 정보에는 근원지의 주소와 포트 정보를 저장해 주고, 클라이언트에는 목적지의 주소와 포트 정보를 저장해준다.

3.2.2.2 MMS

MMS는 Microsoft사에서 개발해서 독자적으로 사용하고 있는 스트리밍 데이터 제어 프로토콜이다. 이 프로토콜은 비공개 되어 있어서 정확한 프로토콜의 동작이나 스펙을 알 수는 없다. 그러나 비록 정확하다고 규정할 수는 없지만, Windows Media Technology 관련 패킷을 다수 분석함으로써 추정될 수 있는 연결 설정 과정을 그림 8에 나타내었다.

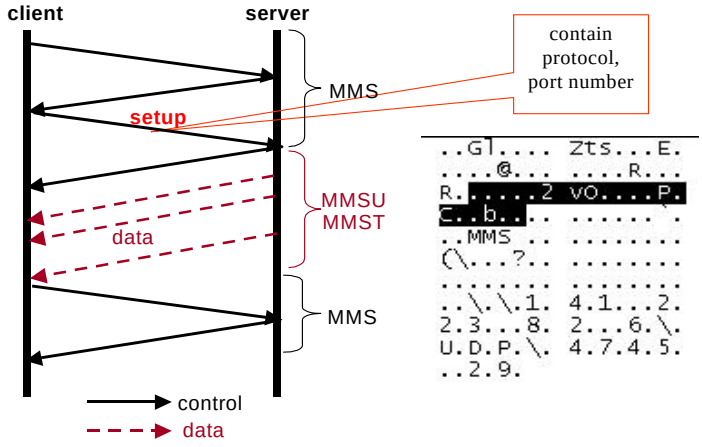


그림 8 MMS를 이용한 데이터 전송 과정

```
procedure ParseMMS
    ( var Msg : CONTROL SESSION MESSAGE,
      vardsRec : DYNAMIC SESSION RECORD)
  var
    msgHeader : array[1..256] of  character ;
    url : array[1..256] of  character ;
  begin
    if destination port of Msg = MMS server port number then
      msgHeader := read  8 bytes from Msg;
      if msgHeader = MMS START  MESSAGE  then
        msgHeader := read  next 2 bytes from Msg;
        if msgHeader = MMS SETUP MESSAGE  then
        while  (there is a byte to read in Msg ) do begin
          msgHeader := next byte from Msg;
          if msgHeader := URL CHARACTER  then
            add  msgHeader to url ;
        end
        if  (URL FORM MESSAGE is in url ) then
          if  PARSED ADDRESS = source  address of dsRec then
            data port of dsRec := PARSED PORT;
            data client address of dsRec := source address  of Msg;
            control server address of dsRec := destination address of Msg;
            control server port of dsRec := destination port of Msg;
            control client address of dsRec := source address of Msg;
            control client port of dsRec := source port of Msg;

 end ; {ParseMMS}
```

그림 9 MMS 다이나믹 세션 분석 방법

우선 클라이언트가 윈도우 미디어 플레이어 버전과 같은 자신의 정보와 함께 서버에게 서비스를 요청한다. 다음에는 클라이언트가 RTSP에서의 setup과 같은 기능을 하는 요청을 서버에게 보낸다. 즉 자신이 데이터를 받을 전송 프로토콜과 포트번호를 담아 서버에게 요청하게 된다. 이 요청이 수락되면 다시 클라이언트가 서비스를 받을 스트림 파일을 서버에게 요구한다. 이렇게 데이터 세션에 대한 협상이 끝나면 RTSP에서와 마찬가지로 클라이언트가 스트리밍 서비스에 관련된 재생, 일시정지, 멈춤 등의 각종 제어 요구를 서버에게 요청하면서 데이터를 전송받거나 서비스를 중단할 수 있다. 그런데 MMS에서는 RTSP에

서와는 달리 클라이언트의 요구 메시지에서 클라이언트가 받을 오직 한 쌍의 전송 프로토콜과 포트 번호만이 확인된다. 또한 이 정보가 클라이언트의 요청 메시지에서 쉽게 찾을 수 있는 형태로 되어 있기 때문에 이 메시지를 분석함으로써 데이터 세션 관련 정보를 얻어낼 수 있다. 그림 9는 이 메시지를 분석해서 데이터 세션 정보를 얻어내기 위한 방법을 기술한 것이다. 그리고 표 3 은 MMS 프로토콜을 이용하는 패킷들의 페이로드를 관찰해서 찾아낸 패턴들을 정의한 것이다.

MMS START MESSAGE
Hexa decimal form (01 00 00 00 CE FA 0B B0)
MMS SETUP MESSAGE
Hexa decimal form (60 00)
URL CHARACTER
Regular expression of “[\\\.0-9a-zA-Z]”
URL FORM MESSAGE
Regular expression of “\\\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\\[([Tt][Cc][Pp])|([Uu][Dd][Pp])]\\[1-9][0-9]{1,2}[0-9][0-9].*”

표 3 MMS 메시지 타입 및 값

우선 첫 8바이트를 읽어 들여 MMS START MESSAGE와 같은지 비교한다. 이것은 MMS 제어의 시작에 나타나는 공통된 값들이다. 다음

은 MMS SETUP MESSAGE의 값과 비교해서 RTST의 setup와 같은 기능을 하는 요청 메시지를 가려낸다. 그리고 나서 URL의 문자로 들어갈 수 있는 값들을 모은다. 모아진 스트링에서 URL FORM MESSAGE의 정규식에 맞는 패턴이 있는지 찾아 전송 프로토콜과 포트 번호 정보를 알아낸다. 이 때 얻어진 URL 폼을 찾아낸 후 추출된 IP 주소가 발신자 주소와 같은지 재확인함으로써 페이로드에 우연히 URL 형태를 가졌지만 데이터 세션 정보가 아닌 경우를 가려낼 수 있다.

분석 결과 다이나믹 세션 정보를 찾아낸 경우에는 이를 저장한다. 이 메시지가 클라이언트에서 서버로부터 가는 패킷이므로 제어 및 데이터 전송의 서버 정보에는 목적지의 주소와 포트 정보를 저장해 주고, 클라이언트에는 근원지의 주소와 포트 정보를 저장해준다.

3.2.2.3 H.323의 제어 프로토콜

그림 10은 H.323 상의 호출자와 응답자 사이에서 연결이 설정되고 데이터가 전송되기까지의 메시지 교환을 간단히 보여준다.

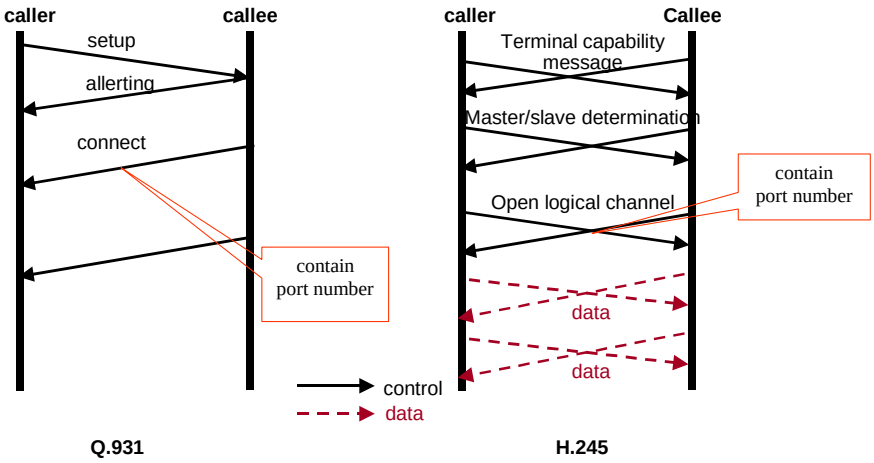


그림 10 H.323에서의 데이터 전송 과정

먼저 호출자와 응답자 사이에 생기는 세션은 Q.931 세션이다. 이 세션에서 호출자가 잘 알려진 포트 번호를 통해 setup 메시지를 보내면, 응답자는 이에 대해 alerting 메시지로 답변한다. 다음에는 응답자에서 connect 메시지를 보내게 되는데, 이 메시지 안에는 새로 맺어지게 될 TCP 연결의 포트번호가 담겨져 있다. 여기에서 connect 메시지를 분석함으로써 다이나믹 세션 정보를 파악할 수 있다.

```
procedure ParseQ931
    ( var Msg : CONTROL SESSION MESSAGE,
      var dsRec : DYNAMIC SESSION RECORD)
  var
    msgHeader : array[1..256] of   character ;
  begin
    if  protocol discriminator of Msg = Q.931 then
      if message type   = CONNECT then
      while   (there is a byte to read in Msg ) do begin
        msgHeader := next element information from Msg;
        if msgHeader = user-user info   then
          if   IP address of H.245 address = source  address of Msg then
            data client port of dsRec := port of H.245 address;
            transport protocol of dsRec := UDP;
            data client address of dsRec := source address  of Msg;
            control server address of dsRec := source address of Msg;
            control server port of dsRec := source port of Msg;
            control client address of dsRec := destination address of Msg;
            control client port of dsRec := destination port of Msg;
      end

 end ; {ParseQ931}
```

그림 11 Q.931 다이나믹 세션 분석 방법

이후, 새로운 세션이 열려 멀티미디어 컨퍼런싱 서비스에 관련된 제어를 담당하게 되는데 이것을 H.245 세션이라 한다. 이 세션에서는 두 호출, 응답자의 능력(capability)에 대한 정보와 역할을 결정하는 메시지를 교환하게 된다. 다음에는 다이나믹 세션에 대한 정보를 open logical channel 메시지를 상대방에게 보내게 되고, 데이터들은 여기에서 생성된 데이터 세션을 통해 전송된다. 따라서 open logical channel의 요

구와 응답 메시지를 분석함으로써 다이나믹 세션에 쓰일 정보를 찾아낼 수 있다. 여기에는 데이터를 전송 받을 주소와 포트번호가 담겨져 있게 된다.

그림 11은 Q.931에서 다이나믹 세션 정보를 찾는 과정을 보여준다. 우선 제어 메시지의 protocol discriminator가 Q.931인지 확인한 후에 그 중에서도 다이나믹 세션 정보를 담고있는 CONNECT 메시지 타입일 경우에만 페이로드로부터 정보를 찾기 위한 파싱을 한다.

Q.931에는 각 element information의 종류의 그 길이를 나타내는 필드가 있다. 따라서 각 element information의 길이만큼 읽어가면서 user-user info라는 element information을 찾는다. 여기에는 전송될 미디어 데이터를 받을 주소와 포트 번호가 담겨져 있기 때문에 DYNAMIC SESSION RECORD에 해당 다이나믹 세션 정보를 넣어주게 된다. 이 때 H.323의 멀티미디어 컨퍼런싱은 클라이언트-서버 개념이 아니기 때문에, 본 논문에서는 클라이언트와 서버를 다음과 같이 정해서 정보를 저장한다. 여기서 협상된 포트 번호는 응답자에서 데이터나 접속을 받기 위해서 사용될 것이기 때문에, 데이터 세션의 클라이언트 주소에는 응답자의 주소를 넣어준다. 그리고 잘 알려진 포트 번호를 사용해서 호출을 받은 응답자를 제어 세션의 서버, 호출자를 클라이언트로 하고 각각 근원지와 목적지의 주소와 포트 번호를 저장한다.

그림 12은 Q.931에서 다이나믹 세션 정보를 찾아내기 위한 방법을 기술한 것이다. H.245 세션에서는 다이나믹 세션에 대한 정보를 갖고 있는 것은 open logical channel 메시지를 교환할 때이다. 따라서 제어 세션 메시지에서 open logical channel 교환 메소드인지를 먼저 판별해서 페이로드를 파싱할 것인지를 선택한다. 선택된 제어 메시지의 페이로드에서 forward나 reverse 채널의 파라미터, 혹은 network access 파라미터 안에 새로운 세션에 대한 정보가 있기 때문에 이러한 파라미터를 찾아 IP 주소와 포트 번호를 읽어온다.

```
procedure ParseH245
    ( var Msg : CONTROL SESSION MESSAGE,
      var dsRec : DYNAMIC SESSION RECORD,
      var Q931Rec : DYNAMIC SESSION RECORD)
  var
    msgHeader : array[1..256] of   character ;
  begin
    if. ( method of Msg = request of H.245 open logical channel or
          method of Msg = response of H.245 open logical channel ) then
      while  (there is a byte to read in Msg ) do begin
        msgHeader := next logical parameter from Msg;
        if  (msgHeader =  forwardLogicalChannelParameters or
             msgHeader = reverseLogicalChannelParameters or
             msgHeader = networkAcessParameters)  then
          data client port of dsRec := tsapIdentifier of H.245 address;
          transport protocol of dsRec := UDP;
          data client address of dsRec := source address  of Msg;
          control server address of dsRec := control server address of Q931Rec;
          control server port of dsRec := control server port of Q931Rec;
          control client address of dsRec := control client address of Q931Rec;
          control client port of dsRec := control client port of Q931Rec;
      end

 end ; {ParseH245}
```

그림 12 H.245 다이나믹 세션 분석 방법

분석 결과 다이나믹 세션 정보를 찾아낸 경우에는 이 정보를 저장한다. 이 메시지가 자신이 앞으로 데이터를 받을 정보를 포함하는 것이기 때문에 데이터 세션의 클라이언트에는 근원지의 정보를 넣어준다. 그리고 제어 세션의 서버와 클라이언트 정보에는 이 H.245를 생성해 낸 Q.245 세션의 제어 서버와 클라이언트 정보를 저장한다.

3.2.2.4 SIP

그림 13에는 SIP에 의해 세션이 초기화되고 데이터가 전송되는 과정이 나타나 있다. 우선 호출자가 SIP의 잘 알려진 포트 번호 5060으로 invite 메시지를 통해 접속을 요구한다. 이 때 SDP[27]를 이용해 전

송 받을 미디어의 타입과 전송 프로토콜 및 포트 번호를 지정해준다. 이에 대한 응답으로 응답 상태와 응답자 쪽에서 받을 미디어 형태와 전송 프로토콜 및 포트 번호를 함께 보내준다. 이렇게 다이나믹 세션에 대한 정보를 교환하고 난 후에 양쪽 방향에서 멀티미디어 데이터를 전송하다가 전송이 끝나면 bye 메시지와 함께 세션 연결이 해제된다. 따라서 invite 요구, 응답 메시지를 분석함으로써 데이터 전달 세션에서 쓰이는 전송 프로토콜과 포트번호를 찾아낼 수 있게 된다.

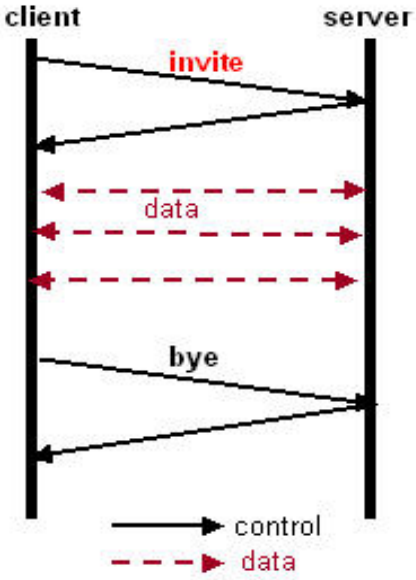


그림 13 SIP를 이용한 데이터 전송 과정

이들 메시지에서 포트 번호를 찾아내는 알고리즘이 그림 14에 제시되어 있다. 메시지의 각 내용들은 라인단위로 구분되어 있다. 따라서 라인단위로 읽어가면서 메시지 내용을 분석하게 된다. 우선 가장 먼저 확인할 것은 이 메시지가 invite 요청이거나 이에 대한 성공적인 응답 메시지를 담고 있는지의 여부이다. 이는 메시로부터 읽어 온 내용이 INVITE MESSAGE이나 RESPONSE MESSAGE의 정규 표현을 포함하는지를 확인함으로써 가능하다. 그리고 표 4에는 이들을 포함한 SIP의 분

석에 필요한 정규식들이 표현되어 있다.

```
procedure ParseSIP
    ( var Msg : CONTROL SESSION MESSAGE,
      vardsRec : DYNAMIC SESSION RECORD)
  var
    msgHeader : array[1..256] of    character ;
    msgMethod : array[1..256] of    character ;
  begin
    if destination port of Msg = SIP server port number then
      msgMethod := read a line from Msg;
      if   (msgMethod = INVITE MESSAGE) or
           (msgMethod = RESPONSE MESSAGE) then
        while   (there is a line to read in Msg ) do begin
          msgHeader := read a next line from Msg;
          if msgHeader = CONNECT FORMAT    then
            data client address of dsRec := PARSED ADDRESS;
          else if msgHeader = MEDIA FORMAT    then
            data client port of dsRec := PARSED PORT;
            if msgMethod = INVITE MESSAGE then
              control server address of dsRec := destination address of Msg;
              control server port of dsRec := destination port of Msg;
              control client address of dsRec := source address of Msg;
              control client port of dsRec := source port of Msg;
            else
              control server address of dsRec := source address of Msg;
              control server port of dsRec := source port of Msg;
              control client address of dsRec := destination address of Msg;
              control client port of dsRec := destination  port of Msg;
        end

 end ; {ParseSIP}
```

그림 14 SIP 다이나믹 세션 분석 방법

이후에는 페이로드의 끝부분까지 라인단위로 메시지를 읽으면서 미디어 전송에 관련된 다이나믹 세션 정보를 찾는다. SDP에 있어서 데이터 세션에 대한 정보는 media 헤더에 있다. media 헤더는 “M= (미디어 타입) (포트번호) (전송프로토콜) (페이로드 타입)” 형태로 되어 있다. 따라서 CONNECT나 MEDIA FORMAT의 정규식을 갖는지를 확인해서 데

이터를 전송할 세션의 전송 프로토콜과 포트번호를 추출해낸다.

분석 결과 다이나믹 세션 정보를 찾아낸 경우에는 이를 저장한다. 이 메시지가 클라이언트에서 서버로부터 가는 패킷이므로 제어 및 데이터 전송의 서버 정보에는 목적지의 주소와 포트 정보를 저장해 주고, 클라이언트에는 근원지의 주소와 포트 정보를 저장해준다.

MEDIA FORMAT
[Cc]= [Ii][Nn] \.+ ([1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2})|(([a-zA-Z]+\.)+[a-zA-Z])]

표 4

INVITE MESSAGE
“[Ii][Nn][Vv][Ii][Tt][Ee] [0-9a-zA-Z]+@ [0-9a-zA-Z]+ [Ss][Ii][Pp].*”
RESPONSE MESSAGE
“[Ss][Ii][Pp]/[0-9]+\.[0-9]+ 200 OK.*”
CONNECT FORMAT
“[Mm]= [a-zA-Z]+ [1-9][0-9]{3,4} [a-zA-Z]+([/-][a-zA-Z])*.*”
MEDIA FORMAT
[Cc]= [Ii][Nn] \.+ ([1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2}\.\\[1-9][0-9]{1,2})|(([a-zA-Z]+\.)+[a-zA-Z])]

표 4 SIP 메시지 정규 표현식

3.3 멀티미디어 서비스 플로우 판별 및 분석

이 절에서는 3.2에서 추출해 낸 다이나믹 세션 정보를 참고로 전

체 플로우 중에서 멀티미디어서비스 플로우를 판별하는 방법에 관해 기술하도록 한다. 저장되어 있는 플로우 정보 중에서 멀티미디어 서비스 트래픽 전송을 위한 다이나믹 세션 관련 플로우들은 포트 번호만으로는 어떤 응용과 관련된 플로우인지를 알 수 없다. 따라서 저장된 플로우와 제어 세션을 분석함으로써 추출해 낸 다이나믹 세션 정보를 서로 연관시킴으로써 기존에는 그 어플리케이션을 알 수 없었던 플로우들의 응용을 밝혀 낼 수 있다.

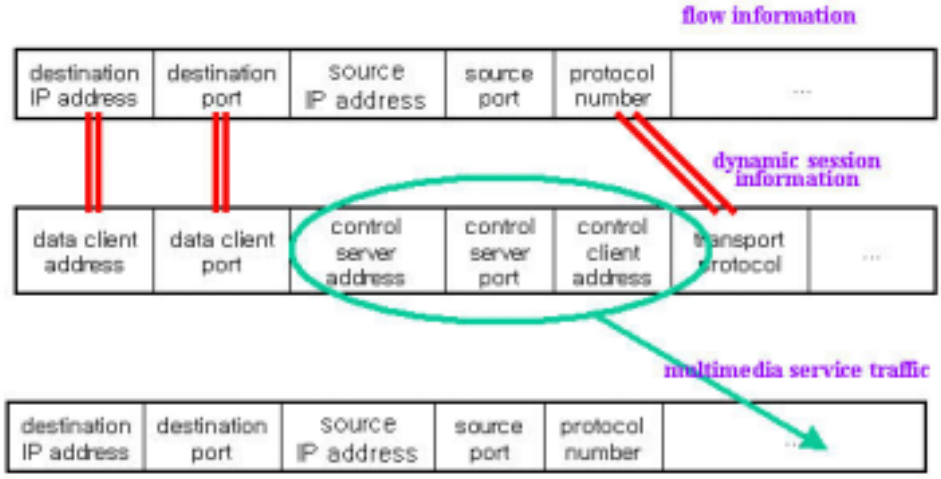


그림 15 멀티미디어 서비스 플로우의 판별

그림 15는 서로 연관시킬 항목을 나타낸다. 스트리밍의 서비스의 경우에는 스트리밍 데이터들이 서버에서 클라이언트 방향으로 일방적으로 흘러간다. 따라서 다이나믹 세션에 있는 [데이터 클라이언트 주소, 포트번호, 전송 프로토콜]의 쌍이 저장된 플로우의 [목적지 주소와 포트번호, 전송 프로토콜]의 쌍과 동일한지를 확인한다. 이 세 개의 쌍만으로 플로우를 결정지을 수 있는 것은 플로우 전송 당시 활성화되어 있는 포트번호가 전혀 다른 응용에 동시에 쓰이지는 않기 때문이다. 그러나 멀티미디어 컨퍼런싱의 경우에는 상호 양방으로 데이터 교환이 이루어지기 때문에 [데이터 클라이언트 주소,포트번호, 전송 프로토콜]의

쌍과 저장된 플로우의 [목적지 주소와 포트번호, 전송 프로토콜]의 쌍이나 [목적지 주소와 포트번호, 전송 프로토콜]과 동일한지를 모두 살펴보아야 한다.

따라서 현재 활성화되어 있는 다이나믹 세션 정보를 유지하는 것이 중요하다. 이 때 패킷이 수집되고 나서 멀티미디어 서비스 분석기에 플로우 정보가 되어 도착할 때까지의 시간만큼의 시간을 t라고 하고 다이나믹 세션 정보가 분석되어 즉시 다이나믹 세션 테이블에 저장된다고 가정한다면, 다이나믹 세션 정보는 t 분 후부터 적용된다. 그리고 FIN 메시지가 오고 난 후 테이블에서 다이나믹 세션 정보를 삭제하는 것은 t 분에 여분의 시간을 추가하도록 한다. 이 여분의 시간은 라우팅 등의 문제로 제어 세션이 끝난 뒤에 도착할 수 있기 때문에 두는 시간으로 시스템 관리자가 설정할 수 있다. 또한 해당 세션으로 트래픽이 계속해서 오가는지를 확인해 일정시간동안 트래픽이 전혀 없을 경우에는 다이나믹 세션 정보를 지워 버림으로써 활성화되어 있는 다이나믹 세션 정보를 유지할 수 있도록 한다.

IV. MST-MON 설계

이 장에서는 앞 장에서 제시된 멀티미디어 서비스 트래픽 분석 방법을 플로우 기반 모니터링 시스템인 NG-MON에 적용시킨다. 이 모니터링 시스템을 MST-MON(Multimedia Service Traffic MONistoring system)이라 명명하기로 한다. 우선 이 시스템에 대한 요구 사항을 분석한 후, 이를 만족시키기 위한 시스템 구조 및 설계 사항을 기술한다.

4.1 요구사항

MST-MON에 있어서 요구사항은 크게 두 가지로 살펴볼 수 있다. 하나는 멀티미디어 서비스 트래픽을 모니터링 할 수 있어야 하고, 다른 하나는 이 시스템이 적용되는 NG-MON의 트래픽 분석기로서 가져야 할 요구 사항이다. 각각의 세부 사항에 대해서는 다음 각 절에서 기술한다.

4.1.1 멀티미디어 서비스 트래픽 모니터링

MST-MON의 모니터링 대상 트래픽을 표 5에 정리하였다. 서비스별로는 스트리밍과 멀티미디어 컨퍼런싱으로 크게 두 가지로 나누어볼 수 있다. 스트리밍 서비스에 있어서는 제어 프로토콜로 RTSP를 이용한 응용으로 QuickTime과 RealNetworks, MMS를 이용한 응용인 Windows Media Technology와 관련된 트래픽을 모니터링하고 분석한다. 멀티미디어 컨퍼런싱과 IP 기반 전화 서비스에 있어서는 해당 서비스를 제공하는 응용이나 서비스 플랫폼이 아직은 분명하게 드러나 있지 않으므로, H.323이나 SIP를 기반으로 한 트래픽인지를 모니터링하고 분석한다. 이렇게 멀티미디어 서비스 트래픽을 모니터링 하기 위해서는 다음과 같은 기능적 세부 사항들을 필요로 한다.

서비스 구분	제어프로토콜	응용
스트리밍	RTSP	RealMedia
		QuickTime
	MMS	Windows Media Technology
멀티미디어 컨퍼런싱	H.323 기반 응용	
	SIP 기반 응용	

표 5 모니터링 대상 서비스

4.1.1.1 플로우 정보 획득

멀티미디어 서비스 패킷들을 캡쳐해서 일정시간 동안 모아 만든 플로우 정보를 가져올 수 있어야 한다. 이 때 IP 단편화(IP fragmentation)를 고려해서 만든 플로우 정보를 가져와야 한다. MST-MON에서는 NG-MON에서 생성한 플로우 정보를 가져와 이용할 수 있도록 한다.

4.1.1.2 다이나믹 세션 정보 추출

멀티미디어 서비스 도중 생겨나는 다이나믹 세션들을 분석하기 위해서는 이들이 사용하는 프로토콜 및 포트 번호에 대한 부가적인 정보가 필요하다. 이 정보는 제어 세션에서 협상 결과 생겨나는데, 제어 세션 패킷의 페이로드에 일정한 패턴으로 나타나게 된다. 따라서 수집된 패킷 중에서 패킷의 헤더 정보를 참고하여 제어 세션 패킷을 선별해 내고, 선택된 패킷의 페이로드로부터 다이나믹 세션 정보를 추출해 내어 이용할 수 있어야 한다.

4.1.1.3 멀티미디어 서비스 플로우 판별 및 종합 분석

멀티미디어 서비스 트래픽 분석을 위해서는 저장된 플로우들 중에서 서비스 제어를 위한 제어 트래픽 뿐만 아니라 데이터 전송을 위한 트래픽도 분석해 낼 수 있어야 한다. 이 때 제어 트래픽의 경우는 잘 알려진 포트 번호를 이용해서 쉽게 분석할 수 있다. 그러나 데이터 전송을 위한 트래픽을 파악하기 위해서는 서비스 도중 생겨나는 다이나믹 세션을 판별하고 분석할 수 있어야 한다. 이를 위해 제어 세션으로부터 추출되어진 다이나믹 세션 정보와 저장된 플로우 정보를 비교함으로써 다이나믹 세션 관련 플로우들을 가려낸다. 그리고 하나의 멀티미디어 서비스를 위해 여러 개의 플로우들이 발생하게 되는데, 이렇게 연관된 플로우들의 상관 관계를 밝힐 수 있어야 한다.

또한 추출되어진 다이나믹 세션 정보가 플로우를 판별하는데 있어서 유효하게 적용될 수 있도록 관리되어야 한다. 다이나믹 세션은 생성되었다가 데이터 전송이 끝나거나 사용자의 요구에 의해 세션이 종료될 수 있다. 따라서 종료된 다이나믹 세션을 찾아냄으로써 활성화된 다이나믹 세션 정보만이 멀티미디어 서비스 플로우 판별에 사용되어질 수 있도록 한다.

4.1.2 NG-MON 분석기로서의 요구 사항

4.1.2.1 실시간 온라인 분석

실시간 온라인 분석은 NG-MON에서 계속 고려하고 있는 기본적인 요구 사항이다. 하지만, 대상으로 하고 있는 네트워크의 성능이 향상됨에 따라 대용량 트래픽 분석을 실시간 온라인 분석하기 위해서 다음과 같은 사항을 필요로 하게 된다.

첫째는 거의 실제 시간에 가깝게 (near-real-time)분석을 해야 한다. 이어야 한다. 플로우 기반의 분석을 한다고 할 때, 플로우는 일정 시

간 동안의 패킷들을 모아 생성되므로, 실제로 분석이 가능한 시간은, 패킷이 캡처 되고 나서 최소한 플로우 타임 아웃 시간 만큼 지난 후가 된다. 따라서 비록 따라서 플로우 생성 과정의 특성 상 완전한 실시간은 아니더라도 실시간에 가까운 분석이 가능해야 한다.

둘째는 로드 분산 구조를 가져야 한다는 것이다. 트래픽 분석기가 동작할 시스템 사양이 범용 PC 시스템이 되든, 고 사양의 서버 시스템이 되든지 간에 하나의 시스템으로 감당할 수 없을 만큼의 대용량 트래픽을 분석하기 위해서는 여러 시스템에 로드를 분산 할 수 있는 방안이 필요하다. 이를 위해 트래픽 분석기 시스템들을 병렬 구조로 구성하는 방법을 고려하였다.

셋째는 분석 결과를 요청하고 결과를 확인하는 데까지 걸리는 응답 시간의 단축이다. 응답 시간은 사용자가 웹 인터페이스를 통해서 요청을 하면, 요청된 분석을 위한 쿼리를 트래픽 분석기의 데이터베이스에 요청하고 데이터베이스가 해당 쿼리를 실행하여 응답한 결과를 가지고 웹 페이지를 생성하여 보여주는 데까지 걸리는 시간의 총합이다. 이러한 일련의 과정 중에서 시간이 가장 많이 소요되는 과정은 데이터베이스가 요청된 쿼리에 응답하는 시간이며, 이 시간을 줄이는 것이 응답시간을 향상 시키는 가장 효과적인 방법이다. 이를 위해 본 논문에서는 시간 및 분석 아이템에 따라 데이터베이스 및 테이블을 구성하는 방법을 고려하였다.

4.1.2.2 제한된 저장 방식의 고려

온라인으로 트래픽을 분석하기 위해서는 단기간 혹은 장기간 동안의 트래픽 정보를 분석 요청 시 즉시 처리할 수 있는 매체에 저장하고 있어야 한다. 오프라인 분석을 위해서라면 값 싸고 대용량의 저장공간을 제공하는 테이프에 저장할 수도 있다. 그러나, 현재 고려하고 있는 분석 시스템의 목적이 실시간 온라인 분석이라는 점을 감안하면

트래픽 정보들을 제한된 저장 용량에 저장하기 위한 사항들이 요구된다.

4.2 시스템 구조

그림 16은 멀티미디어 서비스 트래픽을 모니터링하고 분석하기 위한 MST-MON의 전체 시스템 구조를 나타낸다. MST-MON은 플로우 기반 모니터링 시스템인 NG-MON에 멀티미디어 서비스 트래픽 분석 방법을 적용시킨 것이다. 이에 있어서 멀티미디어 서비스 트래픽 분석을 위해 기존의 시스템을 변경해야 하거나, 기존 시스템에 부하를 주지 않도록 MST-MON의 전체 구조를 설계한다. 다음 각 절에서는 이러한 구조가 4.1에서 제시된 요구 사항을 어떻게 만족시키는지와 시스템 각 구성 요소의 기능에 대해서 설명하기로 한다.

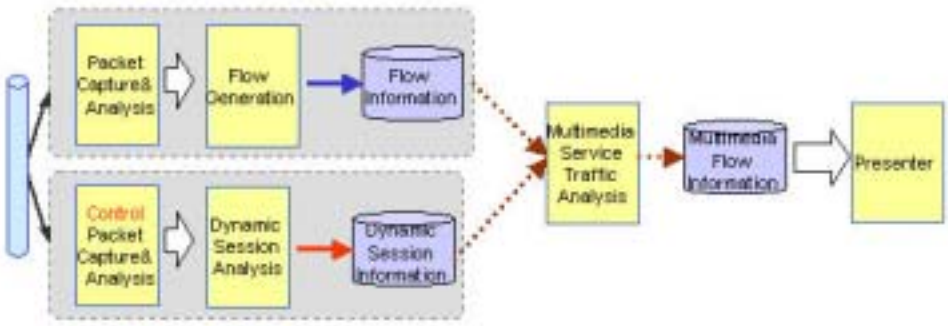


그림 16 MST-MON 전체 시스템 구조

4.2.1 플로우 정보 획득

MST-MON은 플로우 정보 생성을 위해 NG-MON에서 만든 플로우 정보를 그대로 이용한다. 이를 위해 특정 시간마다 플로우 저장기(flow store)에 저장된 플로우 정보를 분석기(analyzer)에서 읽어올 수 있도록 한다. 이 경우 NG-MON의 플로우 생성기(flow generator)는 단편화된 IP

패킷들까지 고려해서 플로우 정보를 만들어 플로우 저장기에 저장하기 때문에, 여기에 저장된 플로우 정보를 읽어옴으로써 IP 단편화로 인해 플로우의 포트 번호를 판단할 수 없게 되는 문제점을 해결할 수 있다.

4.2.2 다이나믹 세션 정보 추출

MST-MON은 멀티미디어 서비스 트래픽을 판별하는데 필요한 정보를 제공해 줄 수 있다. 이를 위해 제어 세션의 패킷들을 분석해서 다이나믹 세션의 정보를 찾아낸다. 이 때 패킷의 페이로드 전체를 캡쳐해서 분석해야 하는데, NG-MON에서 모든 패킷의 페이르드까지 모두 캡쳐해서 처리하는 것은 기존 시스템 성능을 떨어뜨리게 된다. 따라서 별도로 제어 세션의 패킷을 페이로드까지 수집하는 제어 패킷 수집기(Control Packet Capturer)를 둠으로써 기존 시스템이 가질 수 있는 부담을 제거한다.

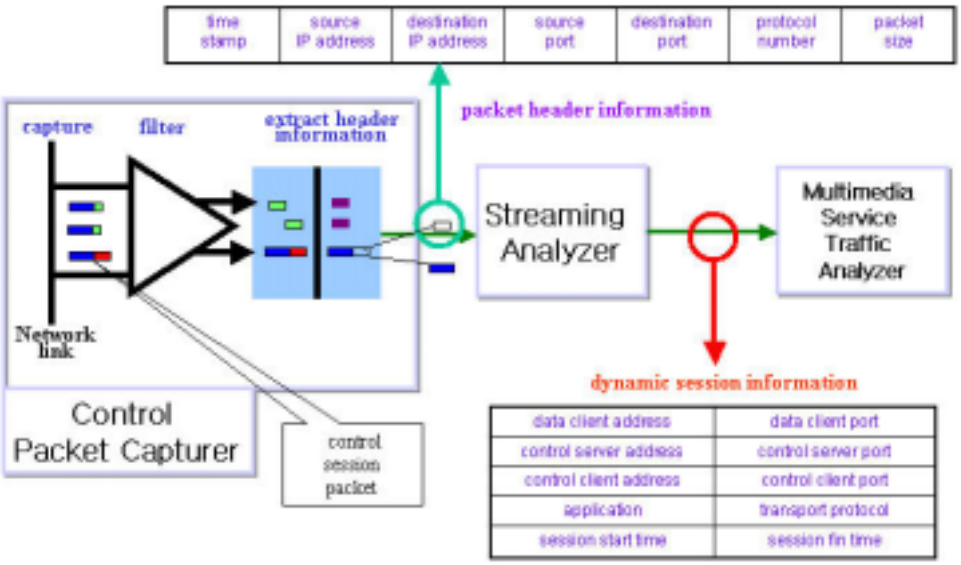


그림 17 제어 패킷 수집 및 다이나믹 세션 정보 추출 과정

그림 17에서 제어 패킷 수집기는 제어 세션 패킷을 페이로드까지

캡쳐한다. 우선 캡쳐된 패킷의 헤더를 분석함으로써 패킷 헤더 정보를 만들고 페이로드 부분과 함께 다이나믹 세션 분석기로 넘긴다. 이 때 전달되는 패킷 헤더 정보 형태가 그림 17에 나타나 있다. 항목 중 Time stamp는 패킷을 캡쳐한 시각이다. 상위 어플리케이션을 구별하기 위한 source/destination port만 transport protocol header에서 정보를 추출하고, 나머지 항목은 IP header 헤더에서 추출한다. 여기서 protocol number은 트랜스포트 계층의 프로토콜을 나타낸다.

다이나믹 세션 분석기는 넘겨 받은 패킷의 헤더 정보를 보고 3.2.1에서와 같이 다이나믹 세션 정보를 가질 수 있는 패킷인지를 판단하고, 각 프로토콜별로 3.2.2의 분석방법에 따라 페이로드로부터 다이나믹 세션이 사용할 포트번호 등에 대한 정보를 추출해 낸다. 분석 결과 얻어진 정보들은 다이나믹 세션 테이블에 저장되어, 트래픽 분석기가 참조할 수 있도록 한다. 이 때 테이블에 기록되는 정보 항목이 그림 17에 나타나 있다. data client address와 port는 다이나믹 세션에서 새롭게 만들어서 데이터를 전송 받을 주소와 포트 번호이다. control server 및 client의 address와 port는 해당 다이나믹 세션을 생성해 낸 제어 세션에서의 서버 및 클라이언트의 IP 주소와 포트 번호이다. 이 정보를 이용해 다이나믹 세션과 관련 있는 제어 세션을 찾아낼 수 있다. 그리고 application은 해당 세션을 이용하고 있는 멀티미디어 서비스의 종류를 나타낸다.

4.2.3 멀티미디어 서비스 플로우 판별

MST-MON의 분석기는 플로우 생성기(flow generator)에서 생성된 플로우 중에서 멀티미디어 서비스 플로우를 판별할 수 있다. 멀티미디어 서비스 도중 생겨나는 제어 세션은 주로 잘 알려진 포트 번호를 사용한다. 따라서 플로우의 근원지나 목적지의 포트 번호만을 보고서도 쉽게 멀티미디어 서비스 플로우인지 찾아낼 수 있다. 또한 다이나믹 세

션에 속하는 플로우인 경우에는 현재 활성화 되어 있는 다이나믹 세션 정보를 참고로 3.3에 기술된 방법에 따라 멀티미디어 서비스와 관련 플로우를 골라낼 수 있다.

비교 결과 멀티미디어 서비스 트래픽이라고 판단되어진 플로우는 그림 18와 같은 정보를 가지고 저장되어진다. 플로우 정보의 각 항목은 플로우 저장기로부터 가져온 플로우 정보에 그림 17의 다이나믹 세션 정보를 종합한 것이다. 이 중에서 control server 및 client의 address와 port 항목은 다음과 같은 처리한다. 제어 관련 플로우는 그대로 서버와 클라이언트의 정보를 넣고, 다이나믹 세션의 플로우는 다이나믹 세션 테이블의 내용을 참조해서 해당 세션을 만들어낸 제어 세션의 정보를 저장해 준다. 이 정보들을 이용해서 같은 서비스에서 파생된 여러 개의 플로우들의 관계를 파악할 수 있다.

source address	destination address
source port	destination port
control server address	control client address
control server port	control client port
transport protocol	application
packet count	packet total size

그림 18 멀티미디어 서비스 플로우 정보

그리고 3.3에 제시된 방법에 따라서 다이나믹 세션 테이블에는 현재 활성화되어 있는 세션 정보만을 유지시킴으로써 관련 없는 플로우를 멀티미디어 서비스 플로우라고 잘못 판단하거나 다이나믹 세션 패킷임에도 불구하고 잡아내지 못하는 문제가 생기지 않도록 한다.

4.2.4 NG-MON에 부합되는 트래픽 분석기의 설계

플로우 저장기에 저장된 플로우 중에서 멀티미디어 서비스와 관련되었다고 판단된 플로우는 트래픽 분석기에 바로 저장된다. 따라서 사용자의 질의가 있을 경우에 거의 실시간에 가깝게 분석되어진다. 이때 로드 분산을 위해 2개 이상의 MST-MON 트래픽 분석기가 플로우저장기에 있는 내용을 쿼리해서 가져다가 병렬적으로 처리할 수도 있다.

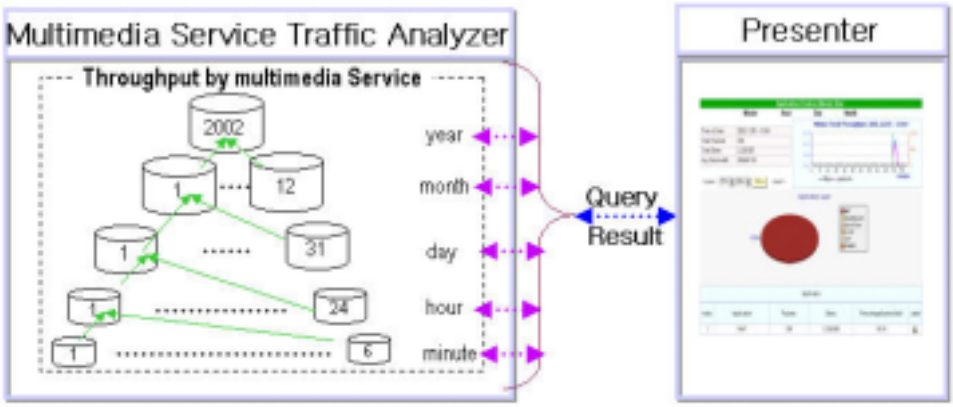


그림 19 테이블 구성 및 프리젠테이션

또한 그림 19에서 보이듯 시간에 따른 데이터만을 갖는 작은 단위의 데이터베이스 테이블을 구성하도록 한다. 플로우 데이터들을 저장하는 데이터베이스의 테이블의 크기가 커지면 해당 테이블에 대한 검색시간 또한 증가하게 된다. 또한 모든 플로우 정보를 하나의 테이블에 저장한 후 분석 요청이 있을 때마다 분석을 하는 방식은 매 분석 시에 전체 플로우 정보를 검색해야 하므로 응답 시간이 많이 걸리게 된다. 따라서 분석 요청이 있을 때, 그에 대응되는 작은 단위의 테이블에 대해 쿼리를 요청함으로써 응답 시간을 단축할 수 있다 [18].

트래픽 분석기에 있어서 또 하나의 고려사항은 제한된 저장공간을 고려한다는 점이다. 이를 위해 일정 기간 동안만 데이터를 보존하고 일정 기간 후에는 그 데이터를 버리는 타임 슬라이딩 윈도우 방식을 적용한다. 그리고, 일정시간 이후에 데이터가 사라져 버림으로써 장기

간 트래픽 분석을 위한 데이터 유지가 어려운 타임 슬라이딩 윈도우 방식을 보완하기 위해 RRDtoo의 요약 방식을 사용한다. RRDtool은 시간에 따라 연속적으로 변하는 데이터들을 가지고, 그 데이터들의 변화 추이를 나타내는 그래프를 생성하기 위한 툴로써, 시간에 따른 모든 데이터를 저장하지 않고 총합에 대한 평균만을 저장함으로써 데이터를 요약 한다. 따라서 그림 19에서 MST-MON 트래픽 분석기 역시 매 분마다 플로우의 정보를 저장하고, 1시간 동안 저장된 플로우 정보를 한꺼번에 모아 요약함으로써 1시간 데이터를 새로 생성해내고, 이러한 과정이 일, 월로 계속해서 이어진다. 이로써 제한된 저장 공간에 분석 데이터들을 저장하고 유지할 수 있게 된다.

V. MST-MON 시스템 구현

이 장에서는 4장에서 설계한 MST-MON을 실제로 구현하고 적용한 결과에 대해서 기술한다. 시스템은 100Mbps 네트워크 링크를 미러링하여 트래픽할 수 있도록 구축되었다. 본 시스템을 실제 적용함으로써 멀티미디어 서비스 트래픽을 모니터링하고 분석한 결과를 제시한다.

5.1 시스템 구조

MST-MON이 구현된 환경은 그림 20과 같다. 포항공대 DPNM연구실에서 백본 망으로 통하는 100Mbps 네트워크 링크의 트래픽을 미러링해서 모니터링 할 수 있도록 했다.

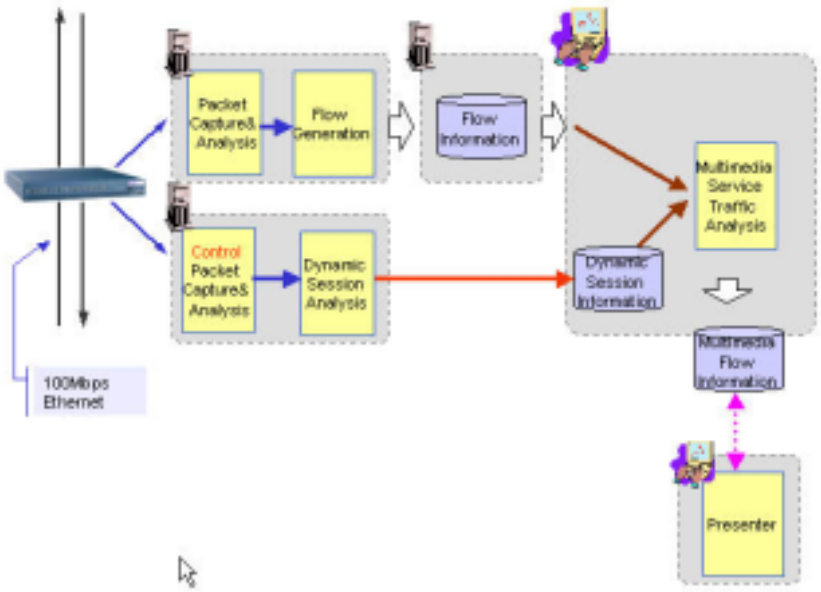


그림 20 MST-MON 시스템 구조도

동일한 트래픽이 스위치의 미러링 포트를 통해 두 개의 시스템으로 전달되지만, 각 시스템의 기능에 따라서 달리 처리된다. 그 중 패킷

수집기는 네트워크 인터페이스 카드로부터 들어오는 모든 패킷을 캡처하여 1 분간 패킷들을 모아 플로우를 생성한 후 이더넷 링크를 통해 플로우 저장기로 플로우 데이터를 전송한다. 이 때 동일한 패킷들이 제어 패킷 수집기로 넘겨지지만, 다이나믹 세션 정보를 위해 필요한 제어 세션 패킷들만을 선별되어 분석되어진다. 만일 선택된 페이로드로부터 다이나믹 세션의 포트 정보 등을 찾게 되면 멀티미디어 서비스 트래픽 분석기에 있는 다이나믹 세션 테이블에 저장시킨다.

멀티미디어 서비스 트래픽 분석기는 매 분마다 플로우 저장기에 쿼리를 보내 플로우 정보를 가져온다. 그 중에서 다이나믹 세션 테이블의 정보를 바탕으로 멀티미디어 서비스 트래픽 플로우를 판별해 데이터베이스에 저장시킨다. 이후 분석된 결과는 사용자의 요구에 의해 프리젠터를 통해 제공되어진다.

System	CPU	Memory	OS
Packet Capturer Flow Generator	Pentium II 450MHz	128 Mbytes	Redhat Linux 7.0
Flow Store	Pentium III 800MHz	256 Mbytes	Redhat Linux 7.2
Control - Packet Capturer D.S. Analyzer	Pentium II 450MHz	128 Mbytes	Redhat Linux 7.0
M.S.T. Analyzer	Pentium III 800MHz	256 Mbytes	Redhat Linux 7.2
Presenter	Pentium III 800MHz	256 Mbytes	Redhat Linux 7.2

표 6 구축 시스템 사양

표 6는 구축된 각 시스템의 사양을 나타낸다. 각 시스템간에는 플로우를 생성하고 판별하는데 있어서 시간 동기화가 필요하다. 이를 위해 LAN(Local Area Network) 환경에서 밀리 초 단위의 동기화를 제공하는 NTP[28](Network Time Protocol)를 사용하였다. 그리고 플로우나 세션 정보를 저장하기 위한 데이터베이스로는 MySQL 3.23.51[29]이 이용되었다.

5.2 제어 패킷 수집기 및 다이나믹 세션 분석기

MST-MON에서는 NG-MON의 플로우 저장기로부터 가져온 플로우 중에서 멀티미디어 서비스 트래픽을 판별하기 위한 정보를 필요로 한다. 이를 위해 제어 패킷 수집기와 다이나믹 세션 분석기를 구현하였다.

패킷 수집기를 구현하기 위해서 C 언어와 libpcap[30] 라이브러리를 사용했다. libpcap 라이브러리는 유닉스 계열의 다양한 운영체제에서 쓸 수 있도록 운영 체제에 독립적인 API(Application Programming Interface)를 제공하므로 운영 체제에 상관 없는 패킷 수집기를 구현할 수 있다.

패킷 수집기에서는 lipcap에서 제공하는 함수를 이용해 패킷 수집 모드를 promiscuous로 설정했다. 대개의 네트워크 인터페이스 카드는 네트워크를 오가는 많은 패킷 중에서 목적지 주소가 해당 호스트인 패킷만 받아들여 상위 계층으로 올려준다. 그러나 이 promiscuous로 설정된 수집 모드는 네트워크 인터페이스로 오는 모든 패킷들을 받아들여 사용자 메모리에서 패킷의 내용을 읽을 수 있도록 한다. 따라서 스위치 포트 미러링 기능을 이용해 네트워크 링크를 오가는 트래픽을 패킷 수집기로 보냄으로써 모니터링 하고자 하는 네트워크 상의 패킷들을 분석해 낼 수 있게 되는 것이다.

제어 세션 패킷을 선택함에 있어 lipcap의 필터를 적용시켰다. 필

터를 적절히 설정함으로써 원하는 호스트나 프로토콜 및 포트번호에 해당되는 패킷만을 수집할 수 있다. 대개 제어 세션은 잘 알려진 포트 번호를 사용하기 때문에 제어 세션에 해당되는 포트 번호를 가진 패킷만을 수집함으로써 시스템 오버헤드를 줄일 수 있다. 그러나 MST-MON의 구현에 있어서는 H.323의 H.245 제어 세션은 동적으로 할당된 포트 번호를 사용하기 때문에 특정 포트 번호만을 수집하는 것은 불가능하다. 대신 제어 세션이 사용하는 프로토콜을 가진 패킷만 수집함으로써 시스템 부하를 일부나마 줄이도록 했다. 수집된 제어 패킷에서 추출된 헤더 정보는 페이로드와 함께 다이나믹 세션 분석기로 넘겨지게 된다.

다이나믹 세션 분석기는 C언어로 구현되었고, 멀티미디어 서비스 트래픽 분석기에 있는 다이나믹 세션 테이블에 접근하기 위해 MySQL 라이브러리를 이용하였다. 다이나믹 세션 분석기는 3.2.2에서 제시된 방법에 따라 각 제어 프로토콜별로 다이나믹 세션 정보를 갖는 패킷을 선별하고 페이로드를 분석한다. 분석 결과 추출해 낸 다이나믹 세션 정보는 멀티미디어 서비스 트래픽 분석기로 넘겨진다.

5.3 멀티미디어 서비스 트래픽 분석기 및 프리젠터 시스템

이 절에서는 멀티미디어 서비스 플로우를 찾아 저장하는 멀티미디어 서비스 트래픽 분석기와 분석 결과를 웹을 통해 사용자에게 보여주는 프리젠터의 구현에 대해서 설명한다.

멀티미디어 서비스 트래픽 분석기는 C 언어와 각종 데이터 베이스 생성 및 제어를 위한 MySQL 라이브러리를 이용해 구현되었다. 매 분마다 플로우 저장기로부터 플로우 정보를 가져와 다이나믹 세션 테이블의 내용을 참조로 3.3의 방법에 따라 멀티미디어 서비스 트래픽을 가려낸다. 이때 테이블에 있는 다이나믹 세션의 활성 시간은 플로우 정보가 만들어져 분석기에 도착하기까지 2분이 소요되므로, 제어 세션 시작 후 2분부터 제어 세션 종료 후 3분까지 적용되도록 했다. 이렇게 찾

아낸 플로우 정보는 대용량 트래픽을 처리하기 위해서는 빠른 시간 안에 데이터 베이스에 저장되어야 한다. 이를 위해서 멀티미디어 서비스 플로우를 찾아낼 때마다 그 정보를 파일에 저장했다가 한꺼번에 데이터베이스에 넣는 방법을 선택했다.

테이블의 생성 및 갱신, 유지와 관련된 구현은 NG-MON 트래픽 분석기의 구현을 따랐다. 테이블 명명 규칙과 RRDtool의 요약 방식 및 타임 슬라이딩 윈도우가 이에 해당된다.

데이터베이스 내에는 시간별로 각각 테이블들이 존재하게 된다. 테이블 이름을 지을 경우, 일정한 규칙에 따라 이름을 지어 필요한 데이터를 요청할 때 테이블 전체 목록을 검색해 보지 않고도 그 규칙에 따라 해당하는 테이블 이름을 생성하여 해당 테이블에 쿼리를 보낼 수 있도록 하였다[18]. 테이블 이름을 인코딩 하는 방법은 표 7와 같다. 분석 대상으로 ‘mst_’(multimedia service traffic)를 공통적으로 사용하고 뒤에는 시간 정보를 가지고 테이블 이름을 결정한다. ‘time2_unit’에서 각 시간 단위를 지정하고, ‘time1’은 상위 시간을, ‘time2’는 해당 단위의 시간을 표시한다. 예를 들어 15시 3분에는 분 단위로 ‘mst_miniute_15_03_table’와 같이 이름을 지정해 테이블을 생성한다.

Mst_[time2 unit]_[time1]_[time2]_table
mst_[minute|hour|day|month]_[00-23|01-31|01-12]_[00-59|00-23|01-31|01-12]_table

표 7 테이블 이름 인코딩 규칙

이러한 테이블들은 제한된 저장공간을 고려했을 때 요약 및 삭제 되어야 한다. 이를 위해 NG-MON의 트래픽 분석기에서는 RRDtool 방식의 요약을 위해 스레드와 세마포어를 사용하여 구현하였다[18]. 메인 프로세스는 스레드를 생성하고 세마포어를 이용해 스레드를 실행시키는 역할을 하며, 스레드는 실제로 RRDtool 방식의 요약을 하는 역할

을 한다. 또한 타임 슬라이딩 윈도우를 설정해 일정 시간동안만 테이블이 유지되고 그 이후에는 삭제되도록 했다[18].

프리젠터에서는 사용자의 요청에 따라 데이터 베이스의 내용을 찾아 트래픽 분석 결과를 웹을 통해 사용자에게 보여준다. 이를 위해 웹 서버로 Apache 1.3.26[31]을 사용하였다. 그리고 멀티미디어 서비스 트래픽 분석기의 데이터베이스 내용을 가져오기 위한 스크립트 언어로 PHP [32](Professional Hypertext Preprocessor)를 사용하였으며, 트래픽의 분석 결과를 그래프로 그려 주기 위해 JpGraph[33] 라이브러리를 사용하여 구현하였다.

5.4 시스템 적용

이 절에서는 MST-MON을 실제 적용시켜 분석한 결과를 다룬다. MST-MON은 저장된 멀티미디어 서비스 플로우 정보를 바탕으로 각 시간 단위, 멀티미디어 서비스 종류, 이용된 개별 서비스별로 트래픽 정보를 제공한다. 또한 시간의 변화에 따른 멀티미디어 서비스 트래픽 추이를 쉽게 알아볼 수 있게 한다. 다음 각 절에서는 각 분석 항목 별로 얻어진 결과에 대해서 예를 들어 설명한다.

5.4.1 시간 단위 별 멀티미디어 서비스 트래픽 정보

MST-MON은 각 시간 단위 별로 멀티미디어 서비스의 트래픽 양을 분석해서 제공한다. 그림 21은 DPNM 연구실에서의 1분 단위의 멀티미디어 서비스 트래픽 분석 예를 보여준다.

그림 21의 상단에는 멀티미디어 서비스 트래픽이 나타난 시간의 전체 정보를 보여준다. 왼쪽 표에서 트래픽 발생 시간, 전체 패킷 수, 전체 트래픽 양, 평균 이용 대역폭을 알 수 있다. 오른쪽 그래프에서는 선택된 시간 단위별로 전체 발생 트래픽의 변화를 보여준다. 예를 들어

분 단위로 분석 시간을 선택했을 경우에 한 시간 동안 각 분단위로 발생한 트래픽 양을 그래프로 보여주게 되어 있다.

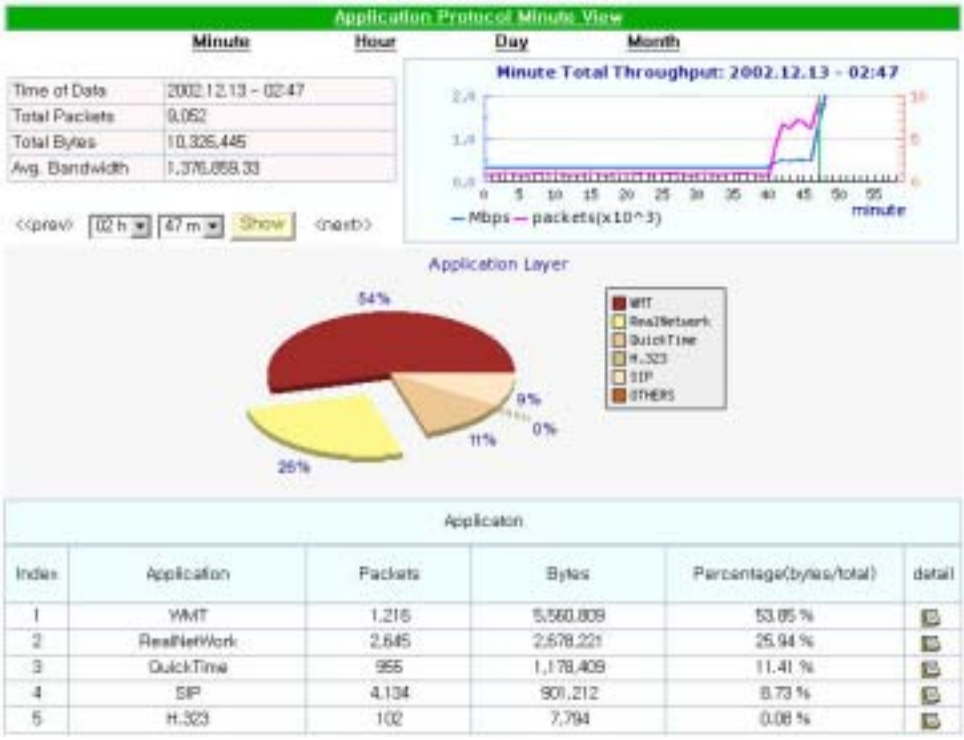


그림 21 시간 단위 별 트래픽 분석

왼쪽 상단 전체 통계 정보 밑에서는 시간 단위별로 분석 시간을 결정할 수 있다. 이 예는 분석 시간으로 2시 47분을 선택했다. 그리고 시간 선택 창 좌우의 <prev, next> 버튼을 통해서 이전 시간 단위나 이후 시간 단위로 이동이 가능하다. 여기서는 1분 전이나 1분 후에 발생한 트래픽을 볼 수 있게 이동할 수 있다.

중간의 원형 그래프는 멀티미디어 서비스 종류별로 발생한 트래픽 비율을 한 눈에 볼 수 있도록 나타낸 것이다. 하단의 표는 해당 시간에 발생한 각 멀티미디어 서비스 종류 별 트래픽 정보를 트래픽 양이 순서에 따라서 보여준다. 각 서비스 종류 별로 발생한 트래픽의 패

킷 수, 바이트 및 전체에서의 비율이 나타나 있다. 해당 멀티미디어 서비스에 대한 정보를 원할 경우에는 <detail> 버튼을 눌러 다음 절에서 제공하는 분석 결과를 볼 수 있다.

5.4.2 멀티미디어 서비스 종류 별 트래픽 정보

MST-MON은 선택된 멀티미디어 서비스 종류에 대한 트래픽 정보를 분석해서 제공한다. 그림 22는 그림 21에서 WMT를 선택했을 경우 나타나는 1분 단위의 트래픽 분석 예이다.

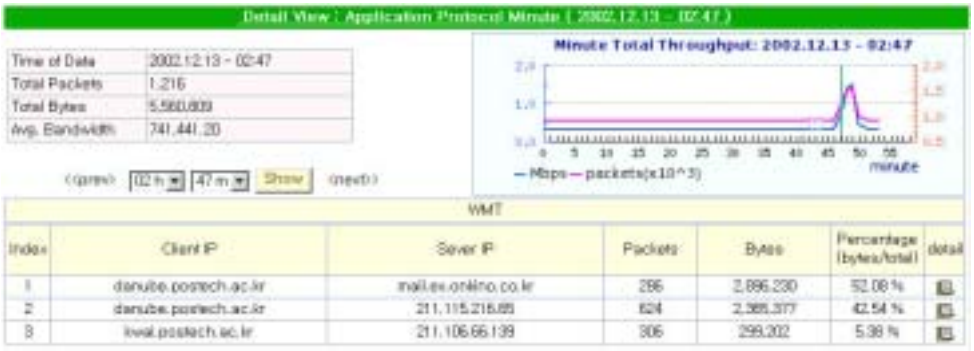


그림 22 멀티미디어 서비스 종류별 분석

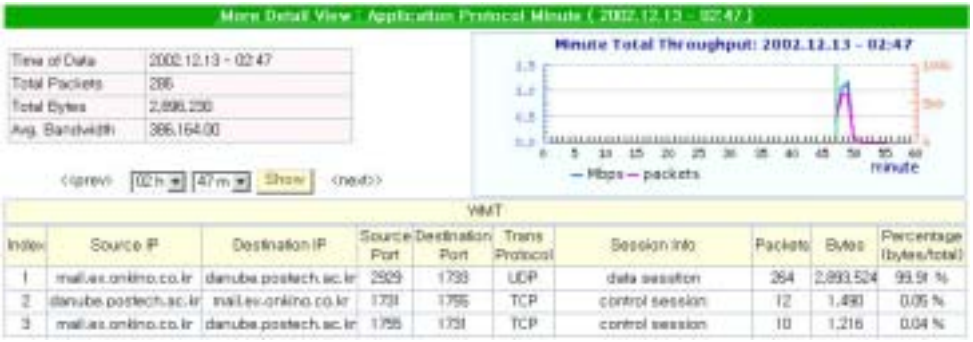
전체적인 화면 구성 및 항목은 앞 절에서 설명한 그림 21과 유사하다. 상단에는 선택된 멀티미디어 서비스 트래픽이 나타난 시간의 전체 정보를 보여주고, 오른쪽 그래프에서는 선택된 서비스 종류에 해당하는 트래픽의 변화 상태를 보여준다. 그리고 원하는 분석 시간을 입력하는 부분이 있다.

화면 하단에 있는 표를 통해서 선택된 서비스 종류를 이용한 트래픽 정보를 알 수 있다. 이 예에서는 2시 47분에 WMT를 이용한 클라이언트와 서버 정보, 그리고 이 때 발생한 트래픽 정보가 나타나있다. WMT 중에서도 특정 클라이언트와 서버 사이에서 발생한 트래픽 정보를 원할 경우에는 <detail> 버튼을 눌러 다음 절에서 제공하는 상세 분

석 결과를 볼 수 있다.

5.4.3 개별 이용 멀티미디어 서비스 별 트래픽 정보

MST-MON은 특정 호스트 사이에서 발생한 멀티미디어 서비스 트래픽 정보를 분석할 수 있다. 그림 23는 그림 22에서 첫번째 클라이언트와 서버를 선택했을 때 나타나는 상세 분석 결과이다.



Index	Source IP	Destination IP	Source Port	Destination Port	Trans Protocol	Session Info	Packets	Bytes	Percentage (bytes/total)
1	mail.ex.onkino.co.kr	danube.postech.ac.kr	2929	1733	UDP	data session	264	2,893,524	99.91 %
2	danube.postech.ac.kr	mail.ex-onkino.co.kr	1733	1755	TCP	control session	12	1,480	0.05 %
3	mail.ex.onkino.co.kr	danube.postech.ac.kr	1755	1731	TCP	control session	10	1,216	0.04 %

그림 23 개별적으로 이용된 서비스 분석 결과

이 분석 항목에서도 전체적인 화면은 앞 절에서와 유사하다. 상단에는 특정 호스트에서 이용한 멀티미디어 서비스 트래픽의 전체 정보를 보여주고, 오른쪽 그래프에서는 선택된 트래픽의 변화 상태를 보여준다. 그리고 원하는 분석 시간을 입력하는 부분이 있다.

화면 하단에는 선택된 서비스를 이루고 있는 여러 개의 플로우 정보를 보여준다. 이러한 플로우들은 기존에는 개별적으로 분류되고 분석되었다. 그러나 MST-MON은 특정 멀티미디어 서비스를 위해 발생한 여러 개의 플로우들을 연관시키고 종합해서 분석할 수 있다. 이 예에서는 danube.posech.ac.kr와 mail.ex.onkino.co.kr사이에 WMT 서비스 이용을 위해 발생한 플로우들을 종합 분석한 트래픽 정보가 나타나 있다. 특히 거의 대부분의 양을 차지하고 있는 데이터 세션은 잘 알려진 포트 번호를 사용하고 있지는 않음에도 불구하고, 멀티미디어 세션인지를 파악

하고 다른 제어 세션과 연관시켜서 정보를 보여주고 있다. 이로써 멀티미디어 서비스 트래픽을 파악하고 분석할 수 있게 되었다.

VI. 결론 및 향후 연구

본 논문에서는 플로우 기반 네트워크 모니터링 시스템에 적용될 수 있는 멀티미디어 서비스 트래픽 모니터링 및 분석 방법을 제시하였다. 이를 위해 멀티미디어 서비스를 위한 트래픽 발생 형태와 관련 프로토콜에 대해서 조사했다. 특히 멀티미디어 서비스를 위한 제어 프로토콜이 메시지를 교환하는 과정을 상세히 분석함으로써, 각 제어 프로토콜 별로 제어 세션 패킷의 페이로드로부터 다이나믹 세션 정보를 추출하는 방법을 찾아내었다. 이렇게 얻어진 정보를 바탕으로 멀티미디어 서비스 도중 생성되는 다이나믹 세션을 파악함으로써, 기존에 포트번호만을 가지고 판단했을 경우에는 어플리케이션을 알 수 없었던 멀티미디어 서비스 트래픽들을 분석해 낼 수 있게 되었다. 또한 동일한 멀티미디어 서비스를 위해 발생한 여러 개의 제어 및 데이터 세션들을 하나로 연관시킬 수 있게 됨으로써 어플리케이션 관점에서 분석을 용이하게 만들었다.

또한 본 논문은 멀티미디어 서비스 트래픽 분석 시 기존 연구에서 생겨나는 부정확성의 문제점들을 해결하였다. 하나는 멀티미디어 서비스 어플리케이션에 따라 발생할 수 있는 IP 단편화(IP fragmenation)를 고려했다는 점이다. 이로써 Windows Media Technology와 같이 서비스 전송에 있어서 IP 단편화가 생겨남으로 인해서 파악이 되지 않았던 서비스 트래픽에 대한 분석이 가능하게 되었다. 또 하나는 활성화된 다이나믹 세션 정보를 관리 및 유지한다는 것이다. 이를 통해 기존 연구에서 다이나믹 세션 정보가 사라지고 난 후에 도착함으로써 인식할 수 없었던 다이나믹 세션 트래픽을 잡아 낼 수 있게 되었다.

본 논문에서는 기존 NG-MON에 위에서 분석 방법을 적용하여 MST-MON을 개발하였다. 기존 NG-MON 프로그램을 수정하지 않고, NG-MON에서 수행하는 기존 트래픽 모니터링에 오버헤드를 주지 않도록 설계했다. 구현된 이 시스템은 100Mbps 네트워크 환경에서 멀티미

디어 서비스 트래픽을 모니터링했다. 그 결과 기존에는 알 수 없는 어플리케이션으로 분류되었던 멀티미디어 서비스 관련 트래픽을 파악하고, 어플리케이션에 근간한 분석을 함으로써 멀티미디어 사용 실태를 파악할 수 있는 기반을 제공하였다.

이러한 방법은 네트워크 트래픽 분석에 있어서 기존에 TCP나 UDP의 포트 번호만을 보고 어플리케이션을 추정하던 분석에서 벗어났다는 점에서 의의를 찾을 수 있다. 실제로 어플리케이션의 세션 레벨에서 메시지를 분석하고 세션 정보를 유지함으로써 기존 트랜스포트 계층의 포트 번호만을 보고서는 파악할 수 없었던 트래픽을 분석할 수 있게 되었다. 따라서 트랜스포트 계층에서 어플리케이션 계층으로 분석 수준을 높였다고 볼 수 있다.

향후 연구로 다이나믹 세션 정보 추출의 정확성을 보완하는 노력을 계속하고자 한다. 현재 본 논문에서는 MMS 패킷의 페이로드에서도 다이나믹 세션 정보를 추출해서 이용하고 있다. 그러나, MMS 프로토콜의 스펙이 비공개되어 있기 때문에 이에 대한 정확성을 입증하기 위해서는 프로토콜 스펙을 구해서 구현하고 이를 검증하는 연구가 필요할 것이다.

그리고 MST-MON을 캠퍼스 네트워크에 적용하고자 한다. 이러한 시도를 위해서는 우선 1Gbps 상에서 동작하는 NG-MON과의 연동 및 성능에 대한 검증이 필요하게 될 것이다. 이렇게 시스템을 적용한 결과를 바탕으로 네트워크 상에서의 멀티미디어 서비스별 트래픽 사용량과 이용 실태를 파악하고자 한다.

또한, 인스턴트 메시징에서의 파일 전송 등과 같이 파일 전송이나 기타 다른 서비스를 위해서 다이나믹 세션을 생성하고 이용하는 어플리케이션에 본 논문에 제시된 방법을 더욱 확장시키는 연구가 필요하다. 이러한 적용 범위의 확장을 통해 기존에 포트번호만으로는 판별이 불가능한 트래픽을 파악하고 분석할 수 있게 될 것이다.

참고문헌

[1] J. Won-Ki Hong, S. U. Park, Y. M. Kang and J. T. Park, “Enterprise Network Traffic Monitoring, Analysis and Reporting Using Web Technology”, Journal of Network and Systems Management, Plenum Press, March 2001, pp. 89-111.

[2] J. Won-Ki Hong, Soon-Sun Kwon and Jae-Young Kim, “WebTrafMon: Web-based Internet/Intranet Network Traffic Monitoring and Analysis System”, Computer Communications, Elsevier Science, Vol. 22, No. 14, September 1999, pp. 1333-1342.

[3] Soon-Hwa Hong, Jae-Young Kim, Bum-Rae Cho, J. Won-Ki Hong, “Distributed Network Traffic Monitoring and Analysis using Load Balancing Technology”, 2001 Asia-Pacific Network Operations and Management Symposium, Sydney, Australia, September 2001, pp.172-183.

[4] Jacobus van der Merwe, R amon Caceres, Y ang-hua Chu, and Cormac Sreenan “mmdump- A Tool for Monitoring Internet Multimedia Traffic,” ACM Computer Communication Review, 30(4), October 2000.

[5] Streaming, http://www.terms.co.kr/streaming.htm.

[6] Micorsoft, Windows Media Technology, http://www.microsoft.com/windows/windowsmedia/default.asp.

[7] Real Networks, Real Media Technology, http://www.realnetworks.com/.

[8] Apple, QuickTime, http://www.apple.com/quicktime/.

[9] H. Schulzrinne, A. Rao, and R. Lanphier, “Real Time Streaming Protocol (RTSP),” RFC 2336, April 1998.

[10] H. Schulzrinne, S. Casner, R. Frederick, V, and Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC1889, January 1996.

[11] ITU-T, “Recommendation H.323: Visual Telephone Systems and Equipment for Local Area Networks Which Provide a Non-guaranteed Quality of Service,” 1996.

[12] M. Handley, H. Schulzrinne, E.Schooler, J. Rosenberg, “SIP: Session Initiation Protocol,”RFC 2543, March 1999.

[13] T. Berners-Lee, R.Fielding, H. Frystyk, “Hypertext Transfer Protocol--HTTP,” RFC 1945, May 1996.

[14] SIP, http://gce.sejong.ac.kr/~leemaster/leemaster.files/project/SIP/sip.htm.

[15] Dave Plonka, “FlowScan: A Network Traffic Flow Reporting and Visualization Tool,” Proc. of 2000 LISA XIV, New Orleans, USA, December 2000, pp. 305-317.

[16] Siegfried Loefler, “Using Flows for Analysis and Measurement of Internet Traffic,” Diploma Thesis, Institute of Communication Networks and Computer Engineering, University of Stuttgart, 1997.

[17] K.C. Claffy, H.W. Braun, and G.C. Polyzos, “A Parameterizable Methodology for Internet Traffic Flow Profiling,” IEEE Journal on Selected Areas in Communications, vol. 13, no. 8, pp. 1481-1494, October 1995.

[18] Se-Hee Han, Myung-Sup Kim, Hong-Taek Ju, and James W. Hong, “The Architecture of NG-MON: a Passive Network Monitoring System for High-Speed IP Networks,” DSOM, October 2002.

[19] Cisco, White Papers, “NetFlow Services and Applications,” http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm.

[20] Luca Deri and Stefano Suin, “Effective Traffic Measurement using ntop,” IEEE Communications Magazine, 38(5), pp. 138-145, May 2000.

[21] C. Fraleigh and C. Diot and B. Lyles and S. Moon and P. Owezarski and K. Papagiannaki and F. Tobagi, “Design and Deployment of a Passive Monitoring Infrastructure,” PAM Workshop, Amsterdam, April 2001.

[22] CAIDA, http://www.caida.org/.

[23] Daniel W. McRobb, “cflowd design,” http://www.caida.org/tools/measurement/cflowd/configuration/design/design.html, 1998.

[24] Tobi Oetiker, “RRDtool – Round Robin Database Tool,”

http://ee-staff.ethz.ch/~oetiker/webtools/rrdtool/.

[25] S. Leinen, Fluxoscope, http://www.switch.ch/lan/stat/fluxoscope/.

[26] Jacobus van der Merwe, R amon Caceres, Y ang-hua Chu, and Cormac Sreenan “mmdump- A Tool for Monitoring Internet Multimedia Traffic,” ACM Computer Communication Review, 30(4), October 2000.

[27] M. Handley, V. Jacobson, “SDP: Session Description Protocol,” RFC 2327, April,1998.

[28] NTP, http://www.ntp.org/.

[29] MySQL 3.23.51, http://www.mysql.com/.

[30] lipcap, http://tcpdump.org/.

[31] Apache 1.3.26, http://httpd.apache.org/.

[32] PHP, http://www.php.net/.

[33] JpGraph, http://www.aditus.nu/jpgraph/.

감사의 글

그동안 너무 너무 감사했습니다. 진짜루요~^^