

SIP 기반 멀티미디어 통신 시스템을 위한 프로토콜 분석기

(A Protocol Analyzer for SIP based Multimedia
Communication System)

정 인 환 [†]

(In-hwan Jung)

요 약 SIP(Session Initiation Protocol)은 VoIP(Voice over IP)와 같은 인터넷 멀티미디어 통신 시스템의 세션 제어 프로토콜로 제안되었으며 많은 표준안에서 채택하고 있다. SIP는 다양한 형태의 오디오와 비디오 형식과 품질이 보장된 실시간 데이터 전송을 지원하기 위해 복잡한 세션 제어 단계를 갖는다. 그러나 지금까지의 프로토콜 분석기는 SIP 기반 멀티미디어 통신 시스템에서 발생되는 트래픽을 상세하게 분석해주지 못하고 있다. 이에 본 논문에서는 SIP 기반의 멀티미디어 통신 시스템의 세션 설정 단계에서 데이터 교환 및 세션 변경에 따른 세부 정보를 분석하고 진단할 수 있는 도구로 새로운 프로토콜 분석기를 설계하고 구현한다. 제안된 프로토콜 분석기는 **STAT**(SIP based Traffic Analysis Tool)이라고 부르며 범용성과 확장성이 있도록 PC Windows에서 사용 가능한 소프트웨어로 구현되었다. STAT은 이더넷의 특징인 동보 기능을 이용하여 저수준 패킷을 수집하여 분석함으로써 SIP 기반 통신 시스템에 직접적인 영향 없이 세션 연결 상태와 실시간 트래픽 모니터링 정보를 제공한다. 따라서 본 논문에서 구현한 STAT은 SIP 기반 멀티미디어 통신 시스템의 개발 및 관리 도구로 활용될 수 있다.

키워드 : 프로토콜 분석기, SIP(Session Initiation Protocol), 멀티미디어 통신

Abstract SIP(Session Initiation Protocol) has been proposed for session control protocol of Internet multimedia communication system like VoIP(Voice over IP). SIP has complicated session control steps to support various kinds of audio and video formats and to assure service quality of real time data communication. Up until now, existing protocol analyzers can not provide such detailed information of SIP based communication system. In this paper, therefore, we propose a new protocol analyzer as a tool that can analyze and diagnose SIP based multimedia communication system throughout the session initiation, data exchange and session change steps. The propose traffic analyzer, which is called **STAT**(SIP based Traffic Analysis Tool), is implemented on Windows environment so that it is generally usable and extensible. Since STAT analyze low level packets captured via Ethernet broadcasting property, it is able to provide session status and real time traffic monitoring information without any affection to the communication system. The STAT which is implemented in this paper, therefore, is expected to be a useful tool for developing and managing of a SIP based multimedia communication system.

Key words : Protocol Analyzer, SIP(Session Initiation Protocol), Multimedia Communication System

1. 서 론

인터넷 멀티미디어 통신을 위한 표준 호 설정 프로토콜로 IETF(Internet Engineering Task Force)의

SIP(Session Initiation Protocol)[1]와 ITU-T(International Telecommunication Union-Telecommunication)의 H.323[2]을 들 수 있다. 현재 대부분의 응용들이 H.323을 기반으로 되어 있으나, 프로토콜의 복잡성으로 인해 구현이 어렵고, 대역폭이 증가하는 단점을 가지고 있다. 이에 비하여 SIP는 HTTP(HyperText Transfer Protocol)와 유사한 텍스트 기반의 메시지를 사용함으로서 이를 구성하는 헤더의 확장이 용이하고,

· 본 연구는 2005년 한성대학교 교내 연구비 지원 과제임

† 종신회원 : 한성대학교 컴퓨터공학부 교수

ihjung@hansung.ac.kr

논문접수 : 2004년 11월 17일

심사완료 : 2005년 4월 17일

간단하게 세션을 설정하고 수정 및 종료할 수 있기 때문에 인터넷 응용 분야로의 적용이 가능한 장점을 가지고 있다[3]. 이런 장점으로 인하여 다수의 표준화 단체들이 차세대 호 설정 프로토콜로서 SIP를 채택함에 따라 SIP를 기반으로 한 많은 응용 연구가 진행되고 있다. 즉, 초고속 무선 네트워크를 통해 멀티미디어를 제작, 전달 및 재생을 논의하는 표준화 단체인 3GPP(3rd Generation Partnership Project)[4]와 동기식 IMT—2000 이동통신 규격을 논의하는 표준화 단체로, 최근 멀티미디어 서비스를 위한 규격들을 제정하고 있는 3GPP2(3rd Generation Partnership Project 2)[5]의 경우, SIP를 멀티미디어 서비스를 위한 표준 프로토콜로 채택하면서, 멀티미디어 서비스(Multimedia Service), 팩스(Fax), 홈 네트워킹(Home Networking), 인스턴스 메세징(Instant Messaging), 망 관리(Network Management), 웹(World Wide Web) 및 SMTP(Simple Mail Transport Protocol) 등의 응용 서비스들이 SIP를 기반으로 통합되는 구조를 보이고 있다[6,7].

대표적인 인터넷 멀티미디어 통신 시스템으로 VoIP(Voice over IP)가 있다. VoIP 기술이란 지금까지 공중전화망을 통해 이루어졌던 음성 서비스를 Internet Protocol을 이용해 여러 가지 다양한 서비스를 제공하는 기술을 말한다[8]. 이렇게 IP 망을 이용함으로서 기존의 전화망에서 하지 못했던 많은 서비스들이 이루어지고 있으며, 대표적인 응용 프로그램들은 Internet Call Center(Web Call Center), Instance Message, CTI(Computer Telephony Integration), UMS(Unified Messaging System) 등을 들 수 있으며, 이에 대한 연구가 활발히 이루어지고 있다[9,10]. VoIP 기술이 시장성 있는 기술로서 각광을 받으면서 VoIP를 위한 호 설정 프로토콜인 SIP는 기존의 H.323을 대체하는 기술로서 주목을 받고 있다.

SIP를 이용한 인터넷 멀티미디어 통신 시스템은 그림 1과 같은 구조를 가진다. UAC(User Agent Client)와

UAS(User Agent Server)는 SIP 메시지를 생성하고 메시지를 처리할 수 있는 사용자 측면의 응용 프로그램이고, 프록시 서버(Proxy Server)와 재지정 서버(Redirect Server)는 SIP 메시지를 처리하기 위한 핵심 요소이다. 프록시 서버는 수신한 메시지를 직접 다음 노드로 전달하고 재지정 서버는 수신한 메시지가 전달되어야 할 새로운 노드의 주소를 알려준다. 위치 서버(Location Server)는 한 도메인 내에 속하는 사용자의 위치를 기록하고 등록 서버(Registrar Server)는 사용자에 대한 정보를 등록 및 관리한다.

일반적으로 SIP 기반 응용 프로그램은 상대편을 세션에 참석시키기 위하여 호출하는 형태로 전개되는데 간략히 설명하면 다음과 같다.

송신자는 호출시 자신에 대한 정보 즉, 자신의 SIP 주소와 호에 대한 정보를 전달한다. 이때 멀티미디어 서비스 통신을 위하여 세션에 표현되어야 할 세션 정보들은 **SDP(Session Description Protocol)**[11]를 이용하여 기술한다. 세션 정보에는 호출자가 통신에서 사용할 주소, 포트번호, 미디어 종류 등이 포함된다. 이후에 세션 협약 과정이 발생한다. 즉, 호출을 받은 수신자는 송신자의 세션 정보를 검토하여 수용 및 거부를 표시하여 송신자에게 전달한다.その後 **RTP(Real-time Transport Protocol)**[12] 세션이 열리고, 이 세션을 통하여 오디오와 비디오 정보는 UDP 상으로 전송된다.

SIP를 이용한 인터넷 멀티미디어 서비스는 오디오, 비디오, 화이트 보드 등과 같은 대용량의 멀티미디어 데이터를 전송하기 때문에 망에 많은 트래픽이 발생된다. 또한 멀티미디어 서비스를 사용하는 사용자들은 각자의 편의와 상황에 여러 종류의 단말들(데스크탑이나 노트북 또는 PDA와 같은 휴대 단말)들과 액세스 망을 사용할 수 있다. 이러한 경우 각 단말이 서로 다른 통신 능력과 미디어 코딩 방식을 가지고 있기 때문에 SIP 도메인 내에서 발생되는 트래픽 정도에 따라서 멀티미디어 서비스 품질이 결정된다. 따라서 망의 운영 상태를 파악하고 검사하는데 있어서 SIP 기반의 응용 서비스들을 구분하여 분석할 수 있는 망 진단 도구의 필요성을 절실히 느끼고 있다.

일반적으로 망 진단 도구로서 가장 많이 활용되는 것이 프로토콜 분석기[13]이다. 프로토콜 분석기는 이더넷(Ethernet)의 특성인 동보 기능(broadcasting)을 이용하여 LAN에 흘러 다니는 모든 패킷들을 실시간으로 수집하여, 패킷의 세부 내용을 보여주는 프로토콜 분석(protocol analyze) 기능과 망 사용 통계를 보여주는 트래픽 측정 기능을 가지고 있다. 그러나 아직까지 SIP 기반 응용 프로그램에서 발생되는 트래픽을 세부적으로

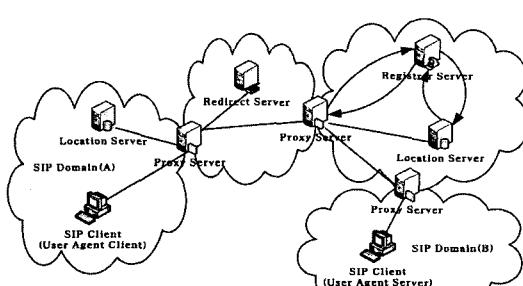


그림 1 SIP 기반의 인터넷 멀티미디어 통신 시스템의 구조

분석할 수 있는 프로토콜 분석기는 찾아보기 힘들다. 따라서 SIP를 기반으로 하는 응용 프로그램에서 발생되는 멀티미디어 통신 트래픽을 분석할 수 있는 도구를 설계하고 구현하는 일이 중요하다.

일반적인 망 진단 도구들은 패킷의 헤더 정보, 현재의 통신 속도와 프로토콜 별 사용량을 보여주는 구조로 되어있다. 이런 진단 도구들은 SIP 기반 멀티미디어 서비스를 이용하는 사용자가 어느 정도인지 즉, 세션의 수가 몇 개이고 세션에서 발생되는 멀티미디어 통신 트래픽이 어느 정도인지 분석할 수 없고, 또한 발생되는 트래픽이 어떤 종류의 미디어 타입인지와 같은 세부적인 정보를 알 수가 없다. 그러므로 SIP 기반 인터넷 멀티미디어 통신 시스템 환경에서 멀티미디어 통신 트래픽을 분석하기 위해서는 프로토콜 분석기가 다음과 같은 기능을 가져야 한다.

- SIP 프로토콜 분석(요청 또는 응답 메시지, 세션 설정, 세션 변경, 세션 종료 등) 기능
- SDP 프로토콜 분석(사용할 주소, 포트번호, 미디어 종료 등) 기능
- RTP 프로토콜 분석(미디어 타입, 전송된 데이터량 등) 기능
- 세션 인식 및 세션 관리(세션의 수, 세션에서 발생되는 트래픽의 량 등) 기능
- 세션별 또는 미디어 별 통계 표시 기능

본 논문에서는 위와 같은 기능을 갖는 새로운 프로토콜 분석기 **STAT**(SIP based Traffic Analysis Tool)를 설계하고 구현한다. STAT은 일반 PC에 기반을 둔 소프트웨어이므로 확장성과 범용성을 갖는다.

본 논문의 구성은 다음과 같다. 2장 관련연구에서는 인터넷 멀티미디어 통신 프로토콜과 일반적인 프로토콜 분석기에 대해 알아보고, 3장에서는 STAT 설계 및 구현에 대하여 설명한다. 4장에서는 실험 및 성능 평가에 대하여 설명한다. 마지막으로 5장에서는 결론 및 향후 연구에 대하여 기술한다.

2. 관련연구

본 장에서는 우선 대표적인 인터넷 멀티미디어 통신 프로토콜인 SIP 프로토콜, SDP 프로토콜 및 RTP 프로토콜에 대해 살펴보고 일반적인 프로토콜 분석기의 구조에 대하여 설명한다.

2.1 SIP(Session Initiation Protocol)

SIP는 ITU-T(International Telecommunication Union-Telecommunication)의 H.323에 대응되는 프로토콜로서 IETF(Internet Engineering Task Force) MMUSIC(Multiparty Multimedia Session Control) WG(Working Group)에서 개발되어 오다가 IETF SIP

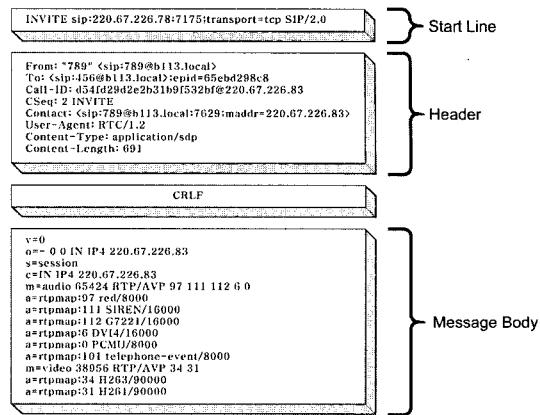


그림 2 SIP 메시지의 구조

WG를 신설하여 작업을 진행하고 있다. SIP는 클라이언트/서버 구조에 기반을 두고 있으며 사용이 간단한 텍스트 기반의 인터넷 프로토콜로서 사용자간의 상호 통신 세션을 초기화하는 용도로 설계되었다.

2.1.1 SIP 특성

SIP 메시지는 그림 2와 같이 시작 줄, 헤더, 헤더의 마지막을 가리키는 CRLF와 메시지 본문으로 구성되어 있으며 SIP는 시작 줄과 헤더의 값만을 기술하고, 메시지 본문의 값은 세션 정보를 기술하는데 사용하는 프로토콜인 SDP가 기술한다. SIP 메시지의 시작 줄에 따라 클라이언트에서 서버로 보내는 SIP 요청(Request) 메시지와 서버에서 클라이언트로 보내는 SIP 응답(Response) 메시지로 구분된다. SIP 요청 메시지는 신호 명령, 요청 URI와 SIP 버전으로 시작 줄이 구성되며, SIP 응답 메시지는 SIP 버전, 상태 코드와 응답 구조 시작 줄이 구성된다.

SIP 메시지의 시작 줄 다음에는 하나 이상의 헤더가 온다. SIP 헤더의 구조는 하나 이상의 일반 헤더, SIP 메시지 종류에 따라 요청(Request) 헤더 또는 응답(Response) 헤더와 본체(Entity) 헤더로 구성되며 SIP 헤더의 종류를 표 1에 나열하였다.

2.1.2 SIP 구성 요소

SIP UA(User Agent)는 SIP 메시지 전송을 요구하고 전송된 SIP 메시지를 수신하는 단말 시스템으로, UAC(User Agent Client)와 UAS(User Agent Server)로 나뉜다. 일반적으로 UAC는 요청 메시지를 만들어서 전송하는 단말을 말하며, UAS는 요청 메시지를 받으면 클라이언트에게 알리거나 내부적 판단에 의하여 대응되는 응답 메시지를 전송하는 단말을 말한다.

SIP 서버들은 프록시 서버(Proxy Server), 재지정 서버(Redirect Server), 등록 서버(Register Server), 위

표 1 SIP 헤더 종류

General Header	Request Header	Response Header	Entity Header
Accept	Authorization	Allow	Content-encoding
Accept-encoding	Hide	Proxy-authenticate	Content-length
Accept-language	Max-forwards	Retry-after	Content-type
Call-ID	Organization	Server	
Contact	Priority	Unsupported	
Cseq	Proxy-authorization	Warning	
Date	Proxy-require	WWW-authenticate	
Encryption	Route		
Expires	Require		
From	Response-key		
Record-route	Subject		
Time stamp	User-agent		
To			
Via			

치 서버(Location Server) 등이 있다. 프록시 서버는 UA로부터의 요청 메시지를 실제 메시지가 전달되어야 할 곳으로 전달(forwarding)하는 역할을 한다. 재지정 서버는 수신한 메시지가 전달되어야 할 새로운 위치를 응답 메시지를 통해 알려주어 사용자가 직접 전송하도록 하며, 등록 서버는 UA를 등록할 수 있게 하고 이를 관리하며 인증 과정을 통해 정당한 사용자인지를 구별하는 역할도 한다. 또한 위치 서버는 한 도메인 내에 속한 사용자의 현재의 위치를 기록하기 위한 서버이다. 이러한 위치 서버를 이용할 경우 현재 사용자가 접속한 시스템의 위치를 파악할 수 있기 때문에 사용자의 이동성(User Mobility)[14]을 지원해 줄 수 있다.

2.1.2 SIP UA의 동작원리

SIP UA가 통신을 하기 위해서는 INVITE 메시지를 전송하여 세션 설정을 요청을 해야한다. 이때 INVITE 메시지의 메시지 본문에 세션 정보가 기술된다. INVITE 메시지를 받은 착신자는 송신자의 세션 정보를 검토하여 각 미디어 스트림을 수용할지 거부할지 판단한다. 수용된 미디어 스트림 정보는 그대로 응답 메시지에 복사하고 거부된 스트림은 포트번호를 0으로 지정한다. 수정된 세션 정보를 응답 메시지 200 OK에 실어서 송신자에게 전송하여 세션 참여를 알린다. 송신자가 ACK 메시지를 전송하면 미디어 코덱이 결정되고 세션이 설정된다. 세션이 설정된 이후에 다른 INVITE 메시지에 의하여 세션이 변경될 수 있다. 세션 변경의 종류로는 미디어 스트림의 추가 및 삭제, 코덱 변경, 주소 정보 변경 등이 있다. 설정된 세션을 종료하기 위해서는 BYE 메시지를 이용하여 세션의 종료를 알린다. BYE 메시지를 받은 착신자는 자신의 세션을 종료하고, 200 OK 응답 메시지를 송신자에게 전송한다. 200 OK 응답 메시지를 받은 송신자는 자신의 세션을 종료한다. 그림

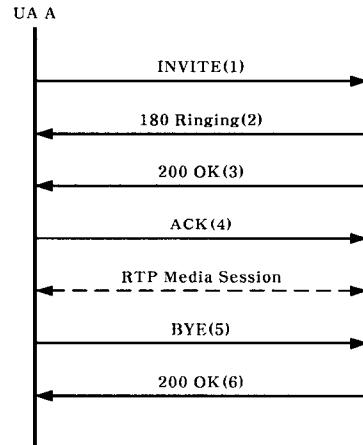


그림 3 SIP 호출 흐름

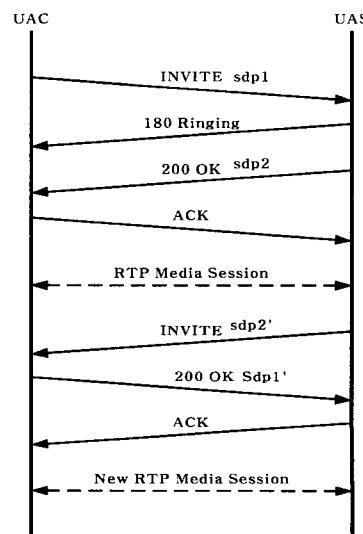


그림 4 SIP 세션 변경

3과 그림 4는 SIP UA의 호출 흐름과 세션의 변경 과정을 보여주고 있다.

위와 같이 SIP 특성, SIP 구성요소와 SIP UA의 동작원리에 대해 살펴보았다. SIP는 클라이언트/서버 구조에 기반을 두고 있으면 SIP 메시지는 요청 메시지와 응답 메시지로 구분된다. 각각의 SIP 메시지가 어떠한 구조를 가지고 있으며 SIP 메시지의 종류에 따라서 헤더의 종류가 결정되고 세션 정보가 기술되는 메시지 본문이 어떤 경우에 존재하는지 알 수 있었다. 또한 SIP 세션 설정 과정 중에 전달되는 SIP 메시지의 종류들을 확인하였다.

본 논문에서는 SIP 메시지들을 수집하고 분석하여,

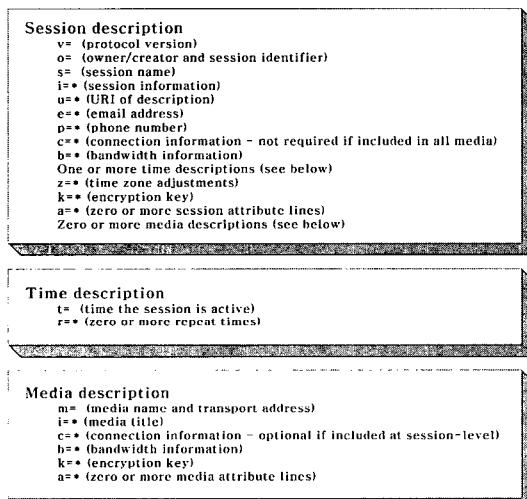


그림 5 SDP 프로토콜 구조

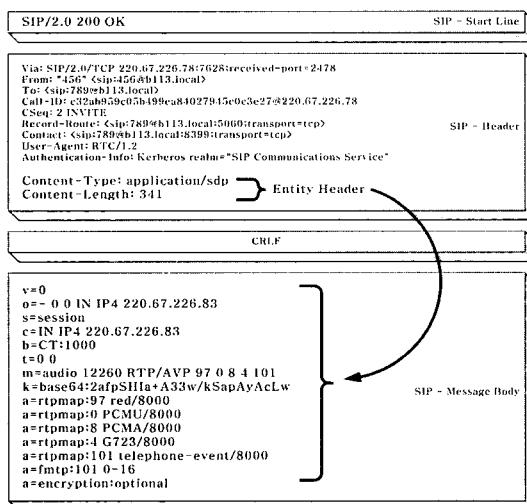


그림 6 SIP 응답 메시지에 기술된 SDP 메시지

세션에 대한 설정이나 변경 또는 세션 종료에 관련된 메시지인 경우에는 해당하는 SIP 메시지를 구조화하고, 구조화된 SIP 메시지를 통하여 세션을 인식하고 관리하여 트래픽을 분석할 수 있도록 한다. 구조화된 SIP 메시지에는 SIP 요청 메시지인 경우에는 신호 명령이 저장되고 SIP 응답 메시지인 경우에는 상태 코드가 저장되며 호를 식별하는데 필요한 SIP 일반 헤더의 Call-ID, From, To, CSeq와 SIP 본체 헤더의 Content-length, Content-type 등이 저장된다.

2.2 SDP(Session Description Protocol)

SDP는 멀티미디어 세션을 기술하기 위해 IETF에서 제안되었고, 멀티미디어 회의를 알리고 설명하기 위한 표준이다. SIP 메시지의 메시지 본문에는 SDP에 의해 정의된 바와 같이 값이 기술되어 있다. SDP 세션의 기술은 필드 이름과 속성 이름으로 이루어진 세션 기술(Session Description), 시간 기술(Time Description)과 미디어 기술(Media Description)의 3개 부분으로 기술된다. 세션의 기술은 하나의 세션 기술(Session Description), 0개 이상의 시간 기술(Time Description) 및 0개 이상의 미디어 기술(Media Description)의 세 부분으로 구성된다.

세션 기술(Session Description)은 전체 회의나 모든 미디어 스트림에 적용되는 전역 특성을 포함하고, 시간 기술(Time Description)은 회의 시작, 중지 및 반복 시간 정보를 포함하며, 미디어 기술(Media Description)은 특정 미디어 스트림에 대한 세부 정보를 포함한다. 그림 5는 SDP 프로토콜의 구조이고, 그림 6은 SIP 메시지와 SDP 메시지와의 관계를 보여주고 있다.

SIP 메시지에 세션 정보가 기술되었을 경우에는 그림 6에서 보듯이 SIP 본체 헤더가 존재된다. SIP 메시지에서 세션 정보가 기술되는 경우는 크게 두 가지로 볼 수 있다. 하나는 SIP INVITE 요청 메시지와 다른 하나는 SIP INVITE 요청 메시지에 대한 응답으로 상태 코드가 200인 SIP 응답 메시지이다.

본 논문에서는 SIP 메시지의 메시지 본문에 기술된 SDP 메시지를 구조화하고, 구조화된 SDP 메시지를 통하여 발생된 트래픽의 미디어 종류를 식별하고 미디어별 통계를 측정한다. 구조화된 SDP 메시지에는 세션 기술(Session Description)의 접속 정보(connection information)와 미디어 기술(Session Description)의 미디어 이름 및 전송 주소와 미디어 특성 등이 저장된다.

2.3 RTP(Realtime Transport Protocol)

RTP는 IETF에서 표준화한 실시간 전송 프로토콜로써, 패킷 기반 네트워크를 통한 실시간 통신 요구를 충족하도록 설계되었다. RTP는 실시간 응용 프로그램을 위한 실시간 데이터의 종단간 네트워크 전송 기능을 제

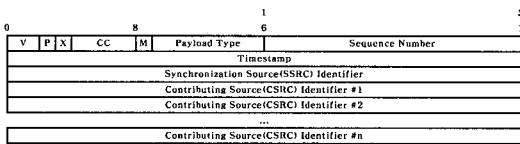
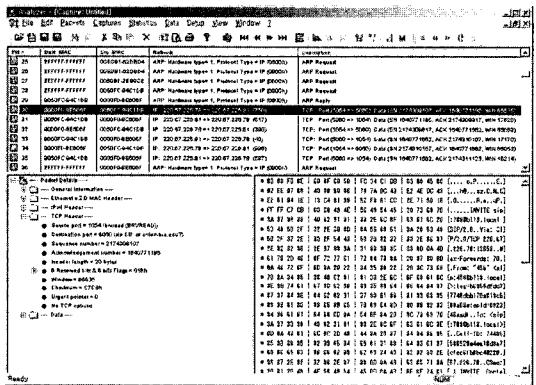
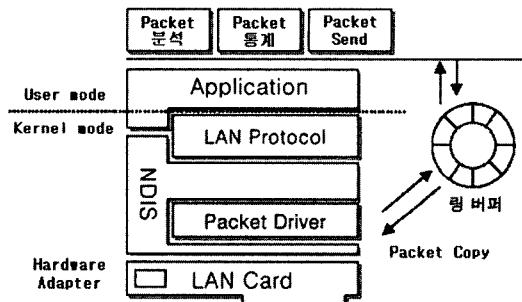


표 2 RTP 패킷의 헤더 내용

헤더	내용
Version	• RTP 프로토콜의 버전을 표시
Padding	• 암호화를 위해 padding된 octet이 있을 경우에 설정 • 가장 마지막 바이트는 padding 수를 표시
Extention	• 고정 RTP 헤더에 추가된 확장 헤더가 존재
CSRC Count	• 고정 RTP 헤더 뒤에 오는 CSRC 식별자의 수
Maker	• RTP 프로필에 따라 마커 비트의 정의 및 사용이 결정 - 비디오 페이로드 : 프레임의 끝을 표시 - 오디오 페이로드 : 셀 발생(Talk Spurt)의 시작을 표시
Payload Type	• 페이로드의 종류를 식별(JPEG 비디오 또는 GSM 오디오 등)
Sequence Number	• 순서한 패킷 발견에 사용 • 시간표(Timestamp)를 가진 패킷들의 순서를 정합 • 전송된 각 RTP 패킷에 대해 1씩 증가
Timestamp	• 데이터가 패킷에 저장되는 순간을 기록 • 깊은 Payload에 따라 다름
SSRC Identifier	• RTP 스트림의 소스와 동기를 맞추기 위한 식별자 • SSRC는 임의로 선택
CSRC Identifiers	• RTP 세션에 기여한 여러 스트림의 소스를 표시 • CSRC Count에서 CSRC의 수를 표시하며 16개까지 퍼시 가능

공하고, 오디오 및 비디오 등의 실시간 데이터는 부호화 및 압축을 통해 패킷 기반 네트워크를 통한 전송에 맞게 최적화 된 후에 RTP 내에 내장(encapsulation)된다. 일반적으로 RTP는 UDP와 함께 기본 전송 계층으로 사용되고, IP와 함께 기본 네트워크 계층으로 사용된다. RTP는 특정 미디어 스트림의 송신자와 수신자간에 협상된 동적 UDP 포트를 사용한다. RTP 패킷은 그림 7과 같이 12 바이트 고정 헤더와 페이로드(Payload)로 구성된다. 표 2는 RTP 패킷의 헤더 내용이다.

SIP는 SIP 메시지의 본문에 기술된 SDP 메시지의 접속 정보, 미디어 종류, 전송 주소(포트 번호), 미디어 특성 정보 등을 참조하여 세션에 참가하는 클라이언트에게 RTP를 이용하여 실시간 데이터를 전송한다. 본 논문에서는 SIP의 실시간 데이터 전송에 사용되는



RTP 패킷의 헤더를 분석하고 구조화된 SDP 메시지와 연계하여 실시간 데이터의 미디어 종류와 미디어 데이터의 량을 분석하고 미디어 별 통계를 측정할 수 있도록 한다.

2.4 프로토콜 분석기

일반적인 프로토콜 분석기는 이더넷(Ethernet)의 특성인 동보 기능(broadcasting)을 이용하여 LAN에 홀라다니는 모든 패킷들을 실시간으로 수집하는 기능과 수집된 패킷의 세부 내용을 보여주는 프로토콜 분석(protocol analyze) 기능과 망 사용 통계를 보여주는 트래픽 측정 기능 등을 가지고 있다. 그림 8은 일반적인 프로토콜 분석기의 구조이고, 그림 9는 프로그램 실행 모습 예이다.

구조와 기능에 대한 설명은 다음과 같다.

가. 실시간 패킷 수집 기능

이더넷은 이론적으로는 LAN 카드에서 LAN에 홀라다니는 모든 패킷을 볼 수 있다. 그러나 일반적인 TCP/IP 및 이더넷 응용 프로그램들이 수행되는 환경에서는 LAN 카드가 모든 패킷을 읽지만 해당 LAN 카드의 MAC(Medium Access Control) 주소에 맞는 패킷만 선택하여 상위 계층인 응용 프로그램으로 전달하고

다른 주소로 보내지는 패킷들은 버리게 된다. 하지만 망진단 도구는 모든 패킷들을 볼 수 있어야 한다. 따라서 LAN 카드에서 모든 패킷을 수집하여 응용 계층으로 전달해 주는 기능이 필요한데 이러한 기능을 담당하는 부분이 패킷 드라이버(Packet Driver)[15,16]이다. 패킷 드라이버에서 수집된 패킷들은 링 버퍼(ring buffer) 또는 원형 큐(circular queue)를 통해서 응용 프로그램으로 전달되게 된다. 원형 큐를 사용하는 이유는 응용 프로그램에서는 패킷 드라이버에서 수집한 패킷을 순서대로 분석해야 하며 패킷 드라이버는 실시간으로 패킷을 수집하여 상위 계층인 응용 프로그램으로 순서에 맞게 전달해야 하기 때문이다.

나. 프로토콜 분석 기능

프로토콜 분석은 저장된 패킷의 헤더를 분석하여 각각의 헤더 내용을 프로토콜의 필드별로 설명하는 기능이다. 그림 9에서와 같이 패킷 목록과 헤더 분석 부분, 그리고 패킷의 Dump 내용이 한 화면에 나온다.

다. 조건적 패킷 수집을 위한 필터(filter) 지정 기능

필터링은 프로토콜 분석기가 특정 조건에 맞는 패킷들만 수집하여 분석이 가능하도록 사용자에게 선택권을 주는 기능이다. 프로토콜 분석기에서는 사용자에게 패킷 필터를 지정하도록 메뉴를 제공하며 지정된 필터를 패킷 드라이버에 설정함으로써 패킷 드라이버는 조건에 맞는 패킷만을 수집하여 응용 프로그램에게 전달한다.

라. 트래픽 통계(속도, 패킷 수 등) 표시 기능

트래픽 통계 표시 기능은 프로토콜 분석 기능과 더불어 프로토콜 분석기의 가장 핵심적인 기능으로 필터링 조건에 맞는 패킷들의 사용 통계를 다음과 같은 세부 조건 및 기준으로 표시하는 것을 말한다.

- 패킷 길이 또는 트래픽(Bits, Bytes, 패킷 수) 기준
- 레이어(Layer) 별 필터링 조건(Mac 레이어, 응용 레이어 등)
- 통계를 표시하는 방법 설정(수평 또는 수직 그래프, 원 그래프 등)
- 통계를 표시하는 시간 간격 설정

위와 같이 일반적 프로토콜 분석기의 구조에 대해 알아보았다. 프로토콜 분석기가 SIP 기반 응용 프로그램에서 발생되는 트래픽을 분석하기 위해서는 SIP 프로토콜과 SIP 프로토콜과 관련된 인터넷 멀티미디어 프로토콜을 분석하는 기능을 가져야한다. 또한 프로토콜 분석기는 SIP 프로토콜 연동의 기본 단위인 트랜잭션(Transaction)을 처리하여 세션 수립을 인식하고 관리할 수 있는 구조가 필요하다.

2.4.1 SIP 패킷 분석을 지원하는 프로토콜 분석기

기존의 프로토콜 분석기 중에서 SIP 패킷의 분석을 지원하는 가장 대표적인 프로토콜 분석기로는 Ethe-

real[24]이 있다. Ethereal은 LAN을 통과하는 모든 SIP 패킷들을 실시간으로 수집하는 기능, 수집된 패킷의 세부 내용을 보여주는 프로토콜 분석 기능, 세션 통계 기능 그리고 RTP Data량에 대한 그래프 표시기능 등을 가지고 있다. 그림 10은 실행중인 Ethereal 프로토콜 분석기의 모습이다.

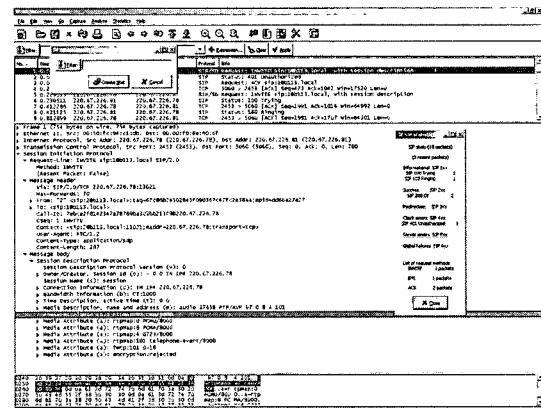


그림 10 Ethereal의 실행 화면

SIP 패킷 분석에 대한 Ethereal의 대표적인 기능은 다음과 같다.

가. 실시간 패킷 수집 기능

일반적인 패킷 분석기와 마찬가지로 실시간으로 SIP 패킷을 수집하는 기능을 가지고 있다. LAN을 통과하는 SIP 패킷을 수집하고, SIP 패킷 수집이 정지된 후에 일괄적으로 화면에 표시한다.

나. SIP 분석 기능

수집된 SIP, SDP, RTP 패킷들의 헤더를 분석하여 각 필드별로 설명하는 기능을 가지고 있다. 대표적인 필드로는 Request Method, State Code 등이 있다.

다. SIP 패킷을 위한 추가의 필터 지정 기능

SIP 패킷을 위한 필터링 기능을 추가하여 효과적인 필터링 기능을 강화하였다.

라. 통계 기능

SIP 패킷에 대한 상태 코드를 통계하는 기능이다. 세션에 사용된 패킷의 수와 상태 코드들을 나열하는 기능을 가지고 있다. 이는 맺어진 세션의 수를 표시함으로써 수집된 패킷에 대해서 몇 개의 세션이 맺어졌는지 파악 할 수 있게 한다. 또한, RTP 패킷들만 분류하여 보여주는 기능을 포함하고 있다.

마. 그래프 기능

SIP 패킷에 의해 세션이 맺어지게 되면 RTP 패킷에 의해서 음성 혹은 영상 데이터가 전송이 된다. 이렇게

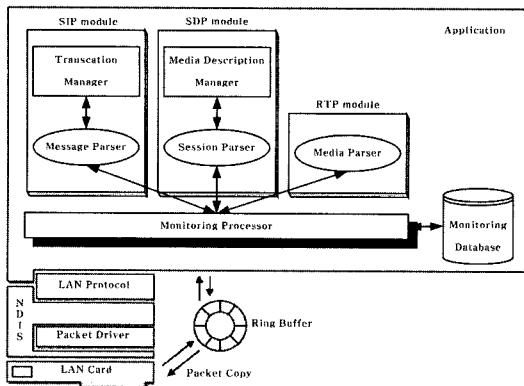


그림 11 STAT 구조

전송되는 RTP 패킷들의 트래픽을 세션이 맺어진 시점부터 종료시점까지 그래프로 보여준다.

Ethereal의 특징은 패킷을 수집하는 단계와 분석 단계로 이원화 되어 있어서 실시간으로 SIP 정보를 보여주지 못한다. 그러나 본 논문에서 제안하는 새로운 트래픽 분석기는 위와 같은 Ethereal의 기능들을 포함할 뿐만 아니라 실시간 SIP 패킷 분석을 위한 모니터링 기능을 추가하였다. 즉, 패킷이 수집되는 것과 동시에 SIP 수집된 패킷을 실시간으로 분석함으로써 수집과정 중에도 세션의 수와 각 세션에 대한 상세한 정보를 바로 보여줄 수 있다.

본 논문에서는 구현된 프로토콜 분석기의 기능과 성능 비교를 위해 Ethereal과 비교 실험을 하였다. 실험 결과는 4장에서 기술한다.

3. STAT 설계 및 구현

본 장에서는 2장에서 소개된 SIP 기반 응용 프로그램에서 발생되는 트래픽을 분석하기 위해서 일반적인 프로토콜 분석기가 가져야 할 구조와 기능을 반영하여 STAT 설계 및 구현에 대해 기술한다.

본 논문에서 제안된 프로토콜 분석기는 Windows를 운영체제로 사용하는 일반 PC에서 수행되는 응용 프로그램으로 패킷 드라이버에서 정의된 API(Application Programming Interface)와 Windows의 사용자 GUI(Graphical User Interface) 함수를 이용하여 개발되었다.

3.1 STAT 구조

본 논문에서 제안하는 새로운 프로토콜 분석기는 그림 11과 같이 패킷 드라이버와 응용 프로그램으로 구성되어 있으며 본 논문에서 구현한 부분은 응용 프로그램 부분이다. 응용 프로그램은 모니터링 처리기(Monitoring Processor)에서 SIP 모듈, SDP 모듈과 RTP 모듈을 호

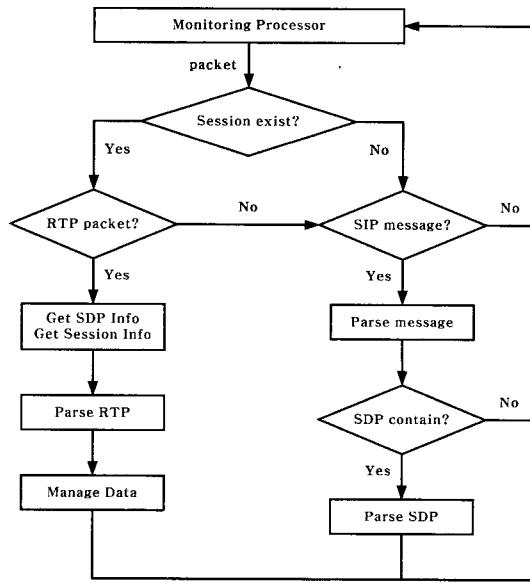


그림 12 STAT 흐름도

출하여 패킷 드라이버를 통하여 실시간으로 수집되는 패킷을 분석하며, 분석된 데이터는 모니터링 저장소(Monitoring Database)에 저장된다. 그림 12는 제안된 새로운 프로토콜 분석기의 흐름도이다.

3.2 SIP 모듈

SIP 모듈은 메시지 분석기(Message Parser)와 트랜잭션 관리자(Transaction Manager)로 구성되어 있다. 메시지 분석기는 모니터링 처리기에서 전달된 SIP 요청 메시지 및 SIP 응답 메시지의 시작 줄과 헤더를 분석하고 구조화하여 저장한다. SIP 메시지에서 구조화되는 정보들은 SIP 신호 명령(Method) 또는 SIP 상태 코드(Status Code)와 트랜잭션을 구분하는데 사용되는 호식별자(Call-ID), 호출자 주소, 호출 받은 사용자 주소, 순서 번호 등이 있다. 트랜잭션 관리자는 메시지 분석기에 의해 저장된 구조화된 SIP 메시지를 이용하여 세션 상태를 확인하고 세션 설정에 대한 트랙잭션이 완료되었을 경우에 모니터링 처리기에게 해당 세션에 대한 트래픽을 분석하도록 알려주며 세션별로 구조화된 SIP 메시지를 저장한다. 트랙잭션 관리자가 추가적으로 세션별로 관리하는 데이터들은 호출자의 세션 정보, 호출 받은 사용자의 세션 정보, 세션 설정 및 종료 상태를 나타내는 세션 상태 정보 등이 있다.

SIP 패킷은 이더넷 헤더, IP 헤더, TCP 헤더 또는 UDP 헤더와 SIP 메시지로 구성되어 있고, SIP 메시지는 이더넷 패킷의 응용 데이터 부분에 실려온다. 그림 13은 SIP 요청 메시지가 포함된 이더넷 패킷을 세부적으로 분할한 것이다.

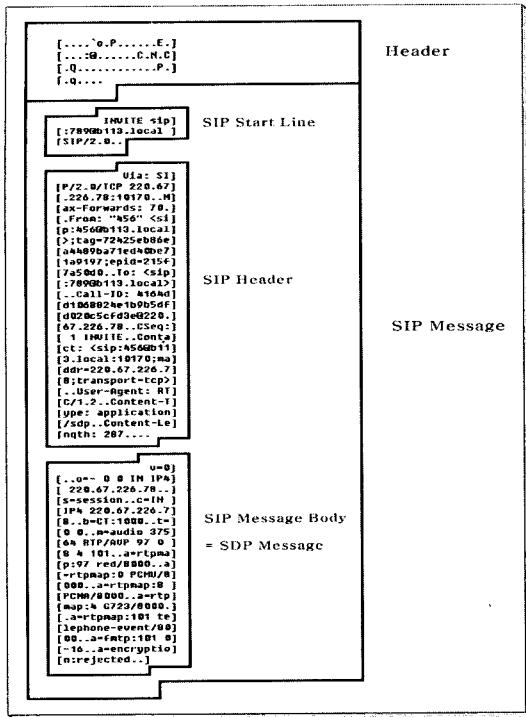


그림 13 SIP 메시지가 포함된 이더넷 패킷의 구조

이더넷 패킷이 SIP 패킷인지 판단하기 위해서는 패킷의 포트 정보를 분석하는 과정이 필요하다. 기본적으로 SIP 패킷은 포트 번호 5060을 사용하여 전송되지만 SIP 서버에 의해 전송된 패킷은 포트 번호 5060을 사용

```

SIP 메시지를 저장하기 위하여 SIP_INFO 구조체와 SIP_CALL_INFO 구조체를 생성한다.
SIP 메세지의 첫번째 줄(SIP 시작 줄)을 읽어 SIP 메세지의 종류를 구분한다.

switch( SIP 메세지의 종류 )
{
    SIP 메세지의 종류에 따라 각 신호 명령 또는 응답 코드를 저장한다.
    case SIP_TYPE_REQUEST :      SIP_INFO에 SIP 신호 명령을 저장한다.
    case SIP_TYPE_RESPONSE :     SIP_INFO에 SIP 응답 코드를 저장한다.
    case SIP_ETC :               아래
}

SIP 메세지에서 한줄씩 읽어와 SIP Header를 분석한다.

while( SIP 메세지 길이 > 분석된 길이 )
{
    SIP 메세지의 한줄을 읽는다.

    읽은 줄에서 ':' 문자를 찾고 SIP 헤더인지 확인하고 Index로 변환한다.

    switch( SIP 헤더 Index )
    {
        각 SIP 헤더 종류에 따라 각 SIP 헤더 내용을 저장한다.
        case SIP_HEADER_From :      SIP_CALL_INFO에 From 헤더 내용을 저장한다.
        case SIP_HEADER_To :        SIP_CALL_INFO에 To 헤더 내용을 저장한다.
        case SIP_HEADER_Call_ID :   SIP_CALL_INFO에 Call-ID 헤더 내용을 저장한다.
        case SIP_HEADER_CSeq :      SIP_CALL_INFO에 CSequence 헤더 내용을 저장한다.
        ...
    }
}

분석된 SIP 메세지를 저장한다.

```

그림 14 SIP 메시지 분석 과정

하지 않는다. 따라서 SIP 패킷을 판단하는데 있어 포트 정보 외에 SIP 메시지의 시작 줄에 해당하는 부분을 분석하는 과정이 필요하다. 본 논문에서 구현된 SIP 모듈은 패킷의 포트 정보와 SIP 메시지의 시작 줄 부분을 이용하여 SIP 패킷을 판단하고 SIP_INFO 구조체를 사용하여 SIP 메시지의 시작 줄 부분을 대응되는 부분으로써 SIP 메시지가 요청 메시지이면 신호 명령 정보가 저장되고 응답 메시지이면 응답 코드 정보가 저장된다.

SIP 메시지의 시작 줄 다음에는 하나 이상의 헤더가 온다. SIP 헤더의 구조는 헤더 이름, ':' 문자, 공백 문자, 헤더 내용, 헤더의 끝을 나타내는 CRLF로 구성된

표 3 SIP 모듈의 중요 함수

함수	설명
SIP_DecodeLine()	• SIP 메시지의 종류(요청, 응답)를 판단한다.
SIP_Request_Line()	• SIP 메시지의 신호 명령과 상태 코드를 가져온다.
SIP_Response_Line()	• SIP 메시지의 헤더 내용을 분석하기 위하여 헤더를 식별한다.
SIP_KnownHeader()	• SIP 메시지가 특정 트랜잭션에 속하는지 판단한다.
SIP_TransactionBelong()	• 트랜잭션이 세션 설정 또는 세션 종료인지 판단한다.
SIP_TransactionInvite()	• SIP 메시지가 동일한지 판단한다.
SIP_TransactionBye()	• 세션을 관리하기 위한 구조를 생성하고 삭제한다.
SIP_EqualMessage()	• 세션을 관리하기 위한 구조를 생성하고 삭제한다.
SIP_CreateElement()	• 세션을 관리하기 위한 구조를 생성하고 삭제한다.
SIP_DeleteElement()	• 세션을 관리하기 위한 구조를 생성하고 삭제한다.
SIP_CreateTable()	• SIP 메시지들을 저장하기 위한 구조를 생성하고 삭제한다.
SIP_DeleteTable()	• SIP 메시지들을 저장하기 위한 구조를 생성하고 삭제한다.
SIP_ElementAdd()	• 세션을 추가, 변경 및 삭제를 수행한다.
SIP_ElementUpdate()	• 세션을 추가, 변경 및 삭제를 수행한다.
SIP_ElementDiscard()	• 세션을 추가, 변경 및 삭제를 수행한다.
SIP_SdpIsContained()	• SIP 메시지 내에 포함된 세션 정보의 크기를 반환한다.
SIP_SdpCallerUpdate()	• 해당 세션에 대한 호출자의 세션 정보를 변경한다.
SIP_SdpCalleeUpdate()	• 해당 세션에 대한 호출 받은 사용자의 세션 정보를 변경한다.
SIP_SdpInfoDiscard()	• 해당 세션의 세션 정보를 삭제한다.

```

SIP Element Information

SessionOpen = TRUE
SessionClose = TRUE
CallerSdp = 0
CalleeSdp = 1

Transaction : INVITE
Call_ID : 84529ddb9d6a44218c232320ba55e11c@220.67.226.83
From : sip:789@b113.local
To : sip:456@b113.local
CSeq : 2

Transaction : 180
Call_ID : 84529ddb9d6a44218c232320ba55e11c@220.67.226.83
From : sip:789@b113.local
To : sip:456@b113.local
CSeq : 2

Transaction : 200
Call_ID : 84529ddb9d6a44218c232320ba55e11c@220.67.226.83
From : sip:789@b113.local
To : sip:456@b113.local
CSeq : 2

Transaction : ACK
Call_ID : 84529ddb9d6a44218c232320ba55e11c@220.67.226.83
From : sip:789@b113.local
To : sip:456@b113.local
CSeq : 2

```

그림 15 SIP 세션별 정보

```

BOOL SPP_DecodeLine(..., TRANSPORT_INFO_POINTER pti, SDP_FIELD_TYPE SFT)
{
    if( SFT == DIME_CREATION_AND_SESSION_IDENTIFIER ) return SDP_DecodeLine_D(..., pti);
    if( SFT == CONNECTION_INFORMATION ) return SDP_DecodeLine_C(..., pti);
    if( SFT == RAMPDOWN_INFORMATION ) return SDP_DecodeLine_R(..., pti);
    if( SFT == TIME_THAT_SESSION_IS_ACTIVE ) return SDP_DecodeLine_T(..., pti);
    if( SFT == REPEAT_TIMES ) return SDP_DecodeLine_R(..., pti);
    if( SFT == TIME_ZONES_ASSIGNMENTS ) return SDP_DecodeLine_Z(..., pti);
    if( SFT == FORWARDING_KEY ) return SDP_DecodeLine_F(..., pti);
    if( SFT == SESSION_ATTRIBUTE_LINES ) return SDP_DecodeLine_SA(..., pti);
    if( SFT == MEDIA_NAME_AND_TRANSPORT_ADDRESS ) return SDP_DecodeLine_MR(..., pti);
    if( SFT == MEDIA_ATTRIBUTE_LINES ) return SDP_DecodeLine_MN(..., pti);

    return TRUE;
}

```

그림 16 SDP 필드 분석 함수

```

SDP Information
220.67.226.83
audio 5558/0 RTP/AVP 97 111 112 6 0 8 4 5 3 101
    red/8000
    SIREN/16000
    G7221/16000
    DV14/16000
    PCMU/8000
    PCMA/8000
    G723/8000
    DV14/8000
    GSM/8000
    telephone-event/8000
video 19540/0 RTP/AVP 34 31
    H263/90000
    H261/90000

SDP Information
220.67.226.78
audio 37554/0 RTP/AVP 97 0 8 4 101
    red/8000
    PCMU/8000
    PCMA/8000
    G723/8000
    telephone-event/8000
video 0/0 RTP/AVP 34
    (null)

```

그림 17 분석된 세션 정보

다. 구현된 SIP 모듈은 SIP 헤더 구조를 참조하여 각각의 헤더와 헤더 내용을 분석하고 SIP_CALL_INFO 구

표 4 SDP 모듈의 중요 함수

함수	설명
SDP_DecodeSdp()	◦ SDP 메시지를 분석한다.
SDP_FindFieldType()	◦ SDP 메시지의 필드 종류를 알려준다.
SDP_DecodeLine()	◦ SDP 메시지의 필드 종류 정보를 이용하여 필드 분석 함수를 호출한다.
SDP_DecodeLine_O() SDP_DecodeLine_C() SDP_DecodeLine_B() SDP_DecodeLine_T() SDP_DecodeLine_R() SDP_DecodeLine_Z() SDP_DecodeLine_K() SDP_DecodeLine_SA() SDP_DecodeLine_M() SDP_DecodeLine_MA()	◦ SDP 메시지의 각 필드를 분석한다.
SDP_CreateElement() SDP_DeleteElement()	◦ 세션 정보를 저장하기 위한 구조를 생성하고 삭제한다.
SDP_ElementAdd() SDP_ElementDiscard()	◦ 세션 정보를 추가하고 삭제한다.

조체를 사용하여 정보를 저장한다. SIP_CALL_INFO 구조체는 SIP 메시지의 헤더에 대응되는 부분으로써 SIP 트랜잭션을 구분하는데 필요한 정보를 저장한다. 그림 14는 SIP 모듈의 핵심 함수인 메시지 분석기의 SIP 메시지를 분석 과정을 기술한 것이다.

구현된 SIP 모듈의 메시지 분석기는 SIP 메시지 분석 중 Content-Type, Content-Length 등과 같은 SIP 본체 헤더를 발견하면 SIP 메시지에 포함된 세션 정보를 SDP 모듈이 분석할 수 있도록 내용을 저장한다. 표 3은 SIP 메시지 분석기가 이용하는 중요 함수들이다.

그림 15는 구현된 SIP 모듈에 의해 세션별로 관리되는 정보를 보여주고 있으며, 세션별 정보는 세션을 구성하는 SIP 메시지들, 세션 설정 정보와 호출자 및 호출 받은 사용자의 SDP 색인 정보로 구성되어 있다.

3.3 SDP 모듈

SDP 모듈은 세션 분석기(Session Parser)와 미디어 기술 관리자(Media Description Manager)로 구성되어 있다. 세션 분석기는 SIP 모듈의 메시지 분석기에서 전달된 SIP 본체(Entity) 헤더의 값을 이용하여 SDP 메시지의 필드 및 필드 내용을 분석하고 주요 필드들을 구조화하여 저장한다. SDP 메시지에서 구조화되는 필드들은 접속 정보, 미디어 안내 정보(미디어 이름, 포트 번호, 사용할 전송계층 프로토콜, 미디어 형식), 미디어 속성(페이지로드 종류, 코덱 이름, 표본화 속도) 등이 있다. 미디어 기술 관리자는 세션 분석기에 의해 저장된 세션 정보를 색인 목록을 만들어 관리하고 세션 정보를

SIP 모듈의 트랜잭션 관리자에게 전달하여 트랜잭션 관리자가 호출자 및 호출 받은 사용자의 세션 정보를 관리할 수 있도록 한다.

SDP 메시지의 구조는 필드 이름, '=' 문자와 속성 이름으로 구성된다. 본 논문에서 구현된 SDP 모듈은 SDP의 이런 특성을 이용하여 필드와 속성을 분석하고 TRANSPORT_INFO 구조체를 사용하여 저장한다. TRANSPORT_INFO 구조체에는 접속 정보, 미디어 안내 정보(미디어 이름, 포트 번호, 사용할 전송계층 프로토콜, 미디어 형식)와 미디어 속성(페이로드 종류, 코덱 이름, 표본화 속도)을 저장한다. SDP 메시지를 구성하는 필드들은 필드 종류에 따라 속성이 기술되는 방식이 다르기 때문에 각 필드를 분석할 수 있는 함수를 구현하였다. 그림 16은 SDP 메시지의 필드 종류 정보를 이용하여 각 필드를 분석하는 함수를 호출하는 SDP_DecodeLine() 함수이고, 그림 17은 세션 설정 과정에서 분석된 세션 정보이고, 표 4는 SDP 모듈에서 사용하는 함수들이다.

3.4 RTP 모듈

RTP 모듈은 모니터링 처리기에 의해 전달된 RTP 패킷의 헤더를 분석하고 저장한다. RTP 모듈의 미디어 파서(Media Parser)는 저장된 RTP 패킷의 페이로드의 종류를 식별하고 모니터링 처리기에게 페이로드의 종류와 미디어 데이터 량을 전달하여 트래픽을 분석할 수 있도록 한다.

RTP 패킷은 이더넷 헤더, IP 헤더, UDP 헤더, RTP 헤더와 RTP 데이터로 구성되어 있다. 구현된 RTP 모듈은 RTP 패킷의 헤더를 분석하고 RTP_INFO 구조체를 사용하여 저장한다. RTP_INFO 구조체는 RTP 패킷의 고정 헤더 정보와 RTP 데이터의 크기 정보를 가지고 있다. 그림 18은 SIP 패킷을 이용하여 세션이 설정된 후에 호출자와 호출 받은 사용자간에 미디어 데이터 전달에 사용되는 RTP 패킷의 예이고, 그림 19는 RTP 모듈에 의해 분석되어 저장된 RTP 패킷의 헤더 정보와 미디어 데이터의 크기를 보여주고 있다.

3.5 모니터링 처리기

모니터링 처리기는 패킷 드라이버로부터 패킷이 전달되면 동작을 시작하며 이더넷 패킷의 중요 헤더 정보와

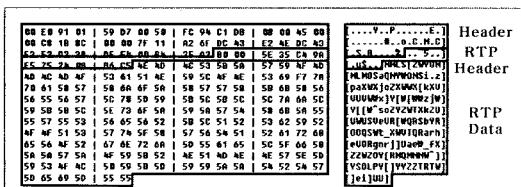


그림 18 미디어 데이터 전달에 사용되는 RTP 패킷의 예

패킷 정보를 분석하여 PACKET_INFO 구조체를 사용하여 저장하고, 저장된 정보를 통하여 SIP, SDP와 RTP 모듈이 해당 패킷을 분석할 수 있도록 도와주며, 각 모듈에 의해 분석된 데이터는 모니터링 저장소에

RTP Information	
bPaddingFlag	= 0
bExtensionFlag	= 0
nNumberOfMixedField	= 0
bMarkerBit	= 1
nPayloadType	= 0
nSequenceNumber	= 31813
nTimestamp	= 3302184443
nSSRC	= 1027423292
nDataLength	= 160

RTP Information	
bPaddingFlag	= 0
bExtensionFlag	= 0
nNumberOfMixedField	= 0
bMarkerBit	= 0
nPayloadType	= 0
nSequenceNumber	= 31814
nTimestamp	= 3302184603
nSSRC	= 1061043516
nDataLength	= 160

RTP Information	
bPaddingFlag	= 0
bExtensionFlag	= 0
nNumberOfMixedField	= 0
bMarkerBit	= 0
nPayloadType	= 0
nSequenceNumber	= 31815
nTimestamp	= 3302184763
nSSRC	= 993671739
nDataLength	= 160

그림 19 분석된 RTP 패킷 정보

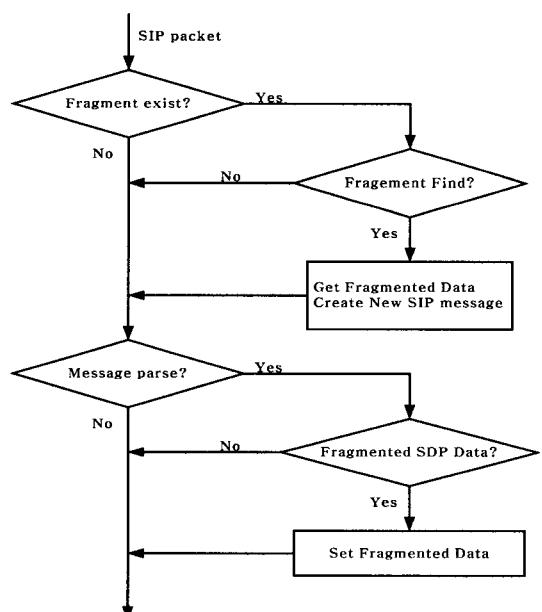


그림 20 모니터링 처리기의 동작 흐름 - SIP 패킷

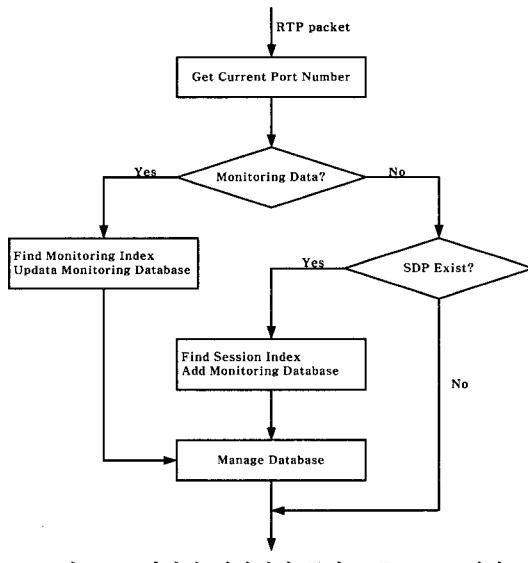


그림 21 모니터링 처리기의 동작 흐름-RTP 패킷

표 5 모니터링 처리기 모듈의 중요 함수

함 수	설 명
Monitoring_ManageData()	• 세션 별 미디어 통계 정보를 관리한다.
Monitoring_CreateData() Monitoring_DeleteData()	• 세션 별 미디어 통계정보를 저장할 수 있는 구조를 생성하고 삭제한다.
Monitoring_DataAdd() Monitoring_DataUpdate() Monitoring_DataDiscard()	• 해당 세션에 대한 미디어 통계 정보를 추가, 변경 및 삭제한다.
Monitoring_CreateFragmentData() Monitoring_DeleteFragmentData()	• 분할된 SIP 메시지들을 저장하고 처리할 수 있는 구조를 생성하고 삭제한다.
Monitoring_SetFragmentedData() Monitoring_GetFragmentedData()	• 분할된 SIP 메시지들과 분할된 정보를 저장하고 분할된 정보를 이용하여 결합시켜 전달한다.
Monitoring_FragmentAdd() Monitoring_FragmentDiscard() Monitoring_FragmentFind()	• 분할된 SIP 메시지들을 저장, 삭제 및 검색한다.

MONITORING_DATA 구조체를 사용하여 저장하여 트래픽을 분석하는데 사용한다. MONITORING_DATA 구조체에는 SIP 모듈에 의해 관리되는 SIP 세션 색인 번호와 호출자와 호출 받은 사용자의 트래픽 정보(미디어별 데이터량과 패킷 수 등)가 저장된다. 또한 모니터링 처리기는 SIP 메시지 또는 SIP 메시지 본문에 기술되는 세션 정보의 크기로 인하여 패킷이 분할되는 현상을 관리하고 처리한다. 분할된 SIP 메시지는 FRAGMENT 구조체와 FRAGMENT_DATA 구조체를 사용하여 저장한다. FRAGMENT 구조체와 FRAGMENT_DATA 구조체에는 분할 색인 번호, PAC-

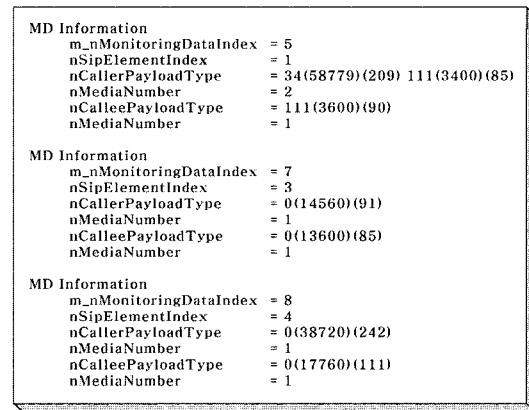


그림 22 분석된 미디어 트래픽 정보

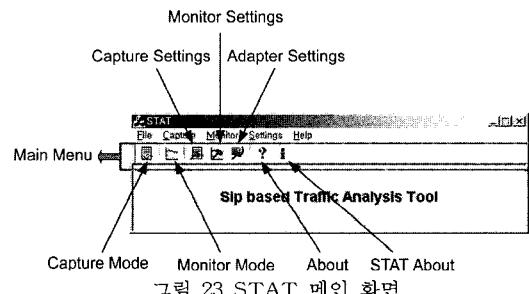


그림 23 STAT 메인 화면

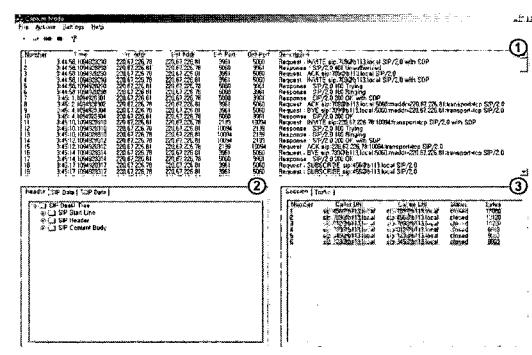


그림 24 STAT Capture Mode 화면

KET_INFO 정보와 패킷 내용 등이 저장된다.

구현된 모니터링 처리기의 동작 흐름은 패킷 드라이버로부터 전달된 패킷이 SIP 패킷인 경우와 RTP 패킷인 경우의 두 가지로 나눠진다. 전달된 패킷이 SIP 패킷인 경우에 모니터링 처리기의 동작은 그림 20와 같이 분할된 SIP 메시지를 저장하거나 분할되어 저장된 SIP 메시지를 찾아서 결합시켜 SIP 메시지를 분석할 수 있도록 하며, 전달된 패킷이 RTP 패킷인 경우에는 그림 21과 같이 모니터링 저장소에 관리되고 있는 미디어 네

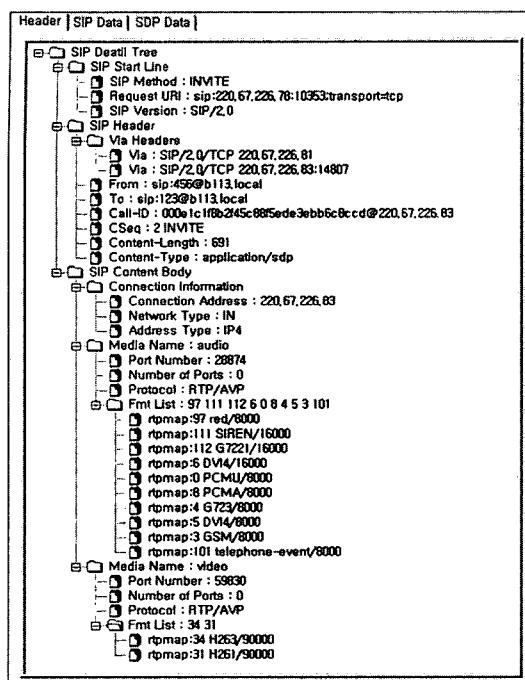


그림 25 STAT Capture Mode - Header 화면

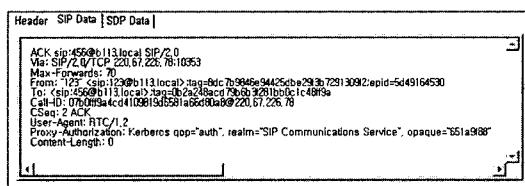


그림 26 STAT Capture Mode - SIP Data 화면

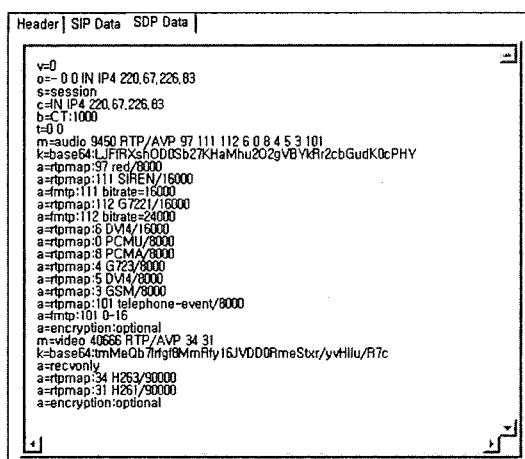


그림 27 STAT Capture Mode - SDP Data 화면

이타인지를 판단하여 미디어 데이터를 쟁신하거나 모니

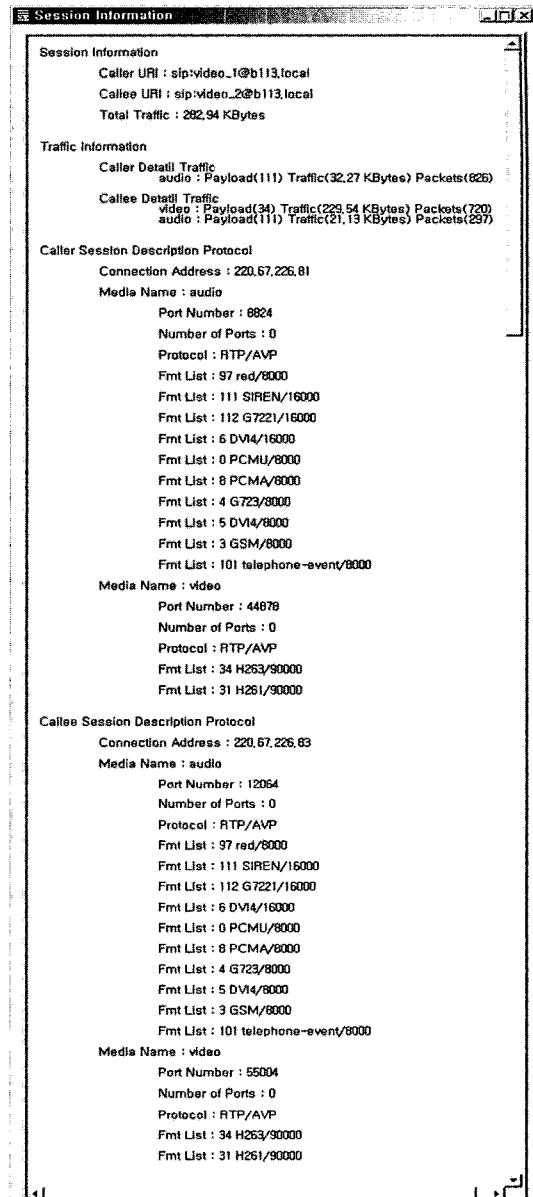


그림 28 STAT Capture Mode - Session 정보 화면

터링 저장소에 새로 추가한다.

모니터링 처리기는 표 5와 같이 모니터링 처리기 모듈 내에 구현된 함수들을 사용하며, 그림 22과 같이 모니터링 저장소에서 미디어 트래픽 정보를 관리한다.

3.6 STAT 화면 설계 및 구현

AT는 “Capture Mode”와 “Monitor Mode”로 나누어 동작할 수 있는 인터페이스를 가지고 있으며 동시에 다수의 모드가 실행될 수 있다. 그림 23은 STAT의 메인

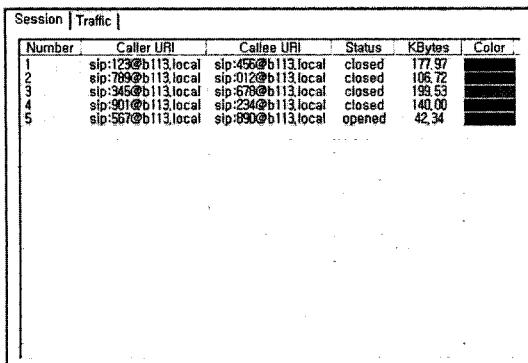


그림 29 STAT Capture Mode - Session 화면

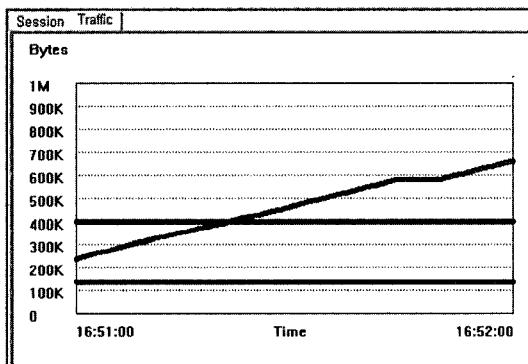


그림 30 STAT Capture Mode - Traffic 화면

화면으로 각각의 기능이 설명되어 있다.

“Capture Mode”는 실시간으로 들어오는 SIP 패킷을 수집하여 패킷 목록, 패킷 세부 내용과 트래픽 정보를 보여주는 기능으로써 수집된 SIP 패킷들의 목록을 보여주는 리스트 컨트롤(①), 세부 내용을 보여주는 템 컨트롤(②)과 트래픽 정보를 보여주는 템 컨트롤(③)로 구성되어 있고, “Monitor Mode”는 SIP 세션이 설정된 이후에 발생되는 미디어 트래픽 통계를 표시하는 기능으로써 선 그래프, 원 그래프와 막대 그래프로 구성되어 있다. 그림 24는 “Capture Mode”로 실행되는 STAT의 실행 예이다.

“Capture Mode”的 세부 내용을 보여주는 템 컨트롤(②)은 분석된 패킷을 헤더별로 보여주는 “Header”와 패킷 내용을 보여주는 “SIP Data”와 “SDP Data”로 구분된다. 그림 25, 그림 26와 그림 27은 “Capture Mode”에 의해 수집된 패킷의 세부 내용을 보여주고 있다.

“Capture Mode”的 트래픽 정보를 보여주는 템 컨트롤(③)은 설정된 SIP 세션 정보를 보여주는 “Session”과 SIP 세션별 발생된 트래픽 량을 보여주는 “Traffic”으로 구분된다. 그림 28과 그림 29는 “Capture Mode”

에 의해 수집된 SIP 세션과 SIP 세션 정보를 보여주고 있으며 그림 30은 SIP 세션별 트래픽 정보를 보여주고 있다.

4. 실험 및 성능 평가

4.1 실험 방법

본 논문에서 구현된 STAT의 실험 환경과 실험 방법은 다음과 같으며, 이를 통하여 성능을 평가한다.

가. 실험 환경

실험 환경은 그림 31과 같으며 상세히 설명하면 다음과 같다.

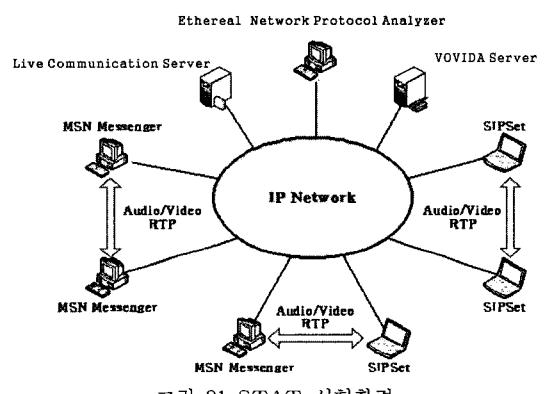


그림 31 STAT 실험환경

SIP 서버는 Microsoft Windows Server 2003 운영 체제에서 동작하고 SIP 기반 인스턴트 메시징 서버인 실시간 통신 서버(Live Communications Server 2003)[17]와 Linux 기반인 VOVIDA Open Communication Application Library(VOCAL)-1.5.0[18]을 사용하였다. SIP User Agent는 SIP 프로토콜을 지원하고 Microsoft Windows XP 또는 Windows 2000 운영 체제에서 동작하는 Microsoft 실시간 통신 서버용 Windows Messenger 5.0[19]과 VOVIDA SIPSet-1.5.0[20]을 사용하였고 음성 채팅 기능 또는 PC 카메라를 이용한 화상 채팅 기능을 갖도록 설정하였다. 비교 분석을 위하여 기본의 프로토콜 분석기 중에서 SIP 패킷의 분석을 지원하는 대표적인 프로토콜 분석기 Ethereal[24]을 선택하였다. 표 6과 표 7은 SIP 서버와 SIP User Agent의 시스템 사양을 나열한 것이다.

나. 실험 방법

실시간 통신 서버와 VOVIDA 서버에 생성된 계정을 이용하는 MSN Messenger와 SIPSet를 사용하여 4가지 실험과 트래픽 발생기(Traffic Generator)[21]를 이용하여 1가지 실험을 하였다.

- 실험 1 - SIP User Agent가 해당 SIP 서버에 로

표 6 SIP 서버의 시스템 사양

구 분	실시간 통신 서버	VOVIDA 서버
OS	Windows Server 2003	Redhat 9.0
CPU	Pentium 4 - 2.8GHz	Pentium M - 1.4GHz
RAM	512MB	256MB
NIC	Intel(R) PRO /100 Mbps VE	3Com 3C90x 10 /100 Mbps PCI

표 7 SIP User Agent의 시스템 사양

구 분	SIP User Agent	
OS	Windows Server 2003	Windows 2000, XP
CPU	Pentium 4 - 2.8GHz	Pentium 4 - 2GHz
RAM	512MB	512MB
NIC	Intel(R) PRO /100 Mbps VE	Intel(R) PRO /100 Mbps VE

그인, SIP 세션 요청 및 로그아웃 과정에서 발생되는 SIP 패킷을 수집하여 정상적으로 분석할 수 있는지 SIP User Agent의 수를 늘려가며 실험을 하였다.

• 실험 2 - SIP 세션 수립 과정에서 발생되는 세션 정보(SDP 패킷)를 분석하는 실험과 SIP 세션이 수립되었을 경우에 SIP 세션의 호출자 정보, 호출 받은 사용

자 정보 및 사용되는 미디어 정보 등의 SIP 세션별 세션 정보를 관리하는 능력을 실험하였다.

• 실험 3 - 미디어 트래픽을 분석하는 실험을 하였다. SIP 세션 수립 후에 RTP 패킷을 이용하여 발생되는 멀티미디어 통신 트래픽을 분석하여 SIP 세션별 발생된 트래픽 통계와 미디어 종류별 트래픽 통계를 잘 처리할 수 있는지 실험하였다.

• 실험 4 - 그래프 인터페이스의 성능 실험이다. SIP 세션별 실시간으로 발생되는 트래픽 정보를 이용하여 그래프가 정상적으로 표현되는지 실험하였다.

• 실험 5 - STAT의 부하 테스트이다. 트래픽 발생기를 이용하여 전송속도를 늘려가면서 미디어 트래픽(RTP 패킷)을 발생시켜 트래픽 분석 성능을 측정하였다.

실험 3과 4에서 사용된 SIP 세션의 생성 시나리오는 표 8와 같으며 호출자와 호출 받을 사용자가 사용할 계정, SIP 서버, SIP User Agent와 트래픽 발생을 위해 사용될 미디어 종류가 나열되어 있다. 생성된 SIP 세션의 수는 15개로 실시간 통신 서버와 계정을 이용하여 5 개의 SIP 세션이 생성되었고 VOVIDA 서버와 계정을

표 8 SIP 세션의 생성 시나리오

Caller			Callee			Media Type
ID	Server	SIP UA	ID	Server	SIP UA	
123	Live Comm. Server	Messenger	456	Live Comm. Server	Messenger	음성
789			012			
345			678			
901			234			
567			890			
0000			1111			
2222	VOVIDA Server	SIPSet	3333	VOVIDA Server	SIPSet	음성
4444			5555			
6666			7777			
8888			9999			
11111		Messenger	22222		SIPSet	화상
33333	VOVIDA Server	Messenger	44444	VOVIDA Server	SIPSet	
55555		SIPSet	66666		Messenger	
77777		SIPSet	88888		Messenger	
99999		Messenger	00000		SIPSet	

표 9 하나의 SIP User Agent에서 발생된 SIP 패킷 수

SIP 서버	SIP User Agent	로그인/SIP 세션 요청/로그아웃 각 과정에서 발생된 SIP 패킷 수(총 SIP 패킷 수)			
		STAT		Ethereal	
		등록된 사용자 유 ④	등록된 사용자 무 ⑤	등록된 사용자 유 ④	등록된 사용자 무 ⑤
실시간 통신 서버	MSN Messenger ①	65/6/20(91)	19/6/10(35)	65/6/20(91)	19/6/10(35)
	SIPSet
VOVIDA 서버	MSN Messenger ②	27/3/10(40)	3/3/6(12)	3/3/6(12)	3/3/6(12)
	SIPSet ③	3/3/6(12)		3/3/6(12)	

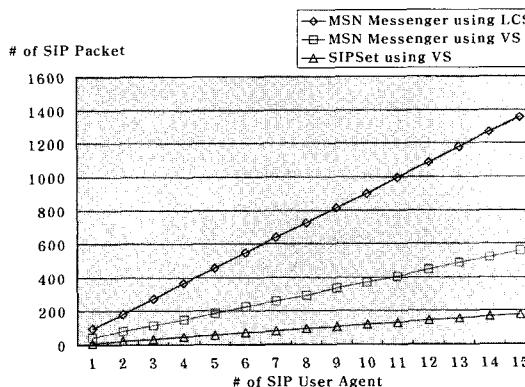


그림 32 온라인 상태 인식 기능 유무에 따라 발생된 패킷량

이용하여 10개의 SIP 세션이 생성되었다. VOVIDA 서버를 이용하여 생성된 10개 SIP 세션 중에 5개의 세션은 MSN Messenger와 SIPSet를 함께 사용하였다. SIPSet은 다른 SIP 서버와 호환성을 갖지 않고 VOVIDA 서버만 동작되도록 설계되어 실시간 통신 서버를 이용한 실험은 하지 못하였다.

4.2 실험 결과 및 분석

4.2.1 실험 1 - SIP 패킷 수집 및 분석 실험

SIP 서버에 생성된 계정을 이용하는 SIP User Agent가 해당 서버에 로그인, SIP 세션 요청 및 로그아웃 과정에서 발생되는 SIP 패킷을 수집하여 정상적으로 분석할 수 있는지 SIP User Agent의 수를 늘려가며 실험하였다. 표 9는 하나의 SIP User Agent가 해당 서버에 로그인, SIP 세션 요청 및 로그아웃의 각 과정에서 발생된 SIP 패킷들을 수집한 결과를 나열한 것이고 등록된 사용자 수는 14명이다.

표 9의 SIP 패킷 수집 실험을 통하여 등록된 사용자가 있을 경우와 없을 경우에 동일한 SIP 서버를 사용하여 발생된 SIP 패킷 수의 비율은 결과 ①에서는 로그인 시 약 3.4배와 로그아웃시 약 2배, 결과 ②에서는 약 9배와 약 1.6배로 등록된 사용자가 있을 경우에 보다 많은 패킷이 발생되었고, 서로 다른 SIP 서버를 사용하여 발생된 SIP 패킷 수의 비율은 결과 ④에서는 로그인 시 약 2.4배와 로그아웃시 약 2배, 결과 ⑤에서는 약 6.3배와 약 1.6배로 결과 ①, ②와 같이 등록된 사용자가 있을 경우가 그렇지 않은 경우보다 많은 SIP 패킷이 발생되었다는 것을 알 수 있다. 이와 같은 결과는 사용되는 SIP 서버와 SIP User Agent에서 제공하는 기능의 차이로 인하여 발생한 것이다. 즉 다른 사용자의 로그인 상태를 확인할 수 있는 온라인 상태 인식(Presence awareness) 서비스를 제공하는 실시간 통신 서버와 같

No.	Time	Source	Destination	Protocol	Info
1	2007-07-22T09:25:50	220.67.226.78	SIP/2.0	status	403 Unauthorized
2	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
3	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
4	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
5	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
6	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
7	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
8	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
9	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
10	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
11	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
12	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
13	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
14	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
15	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
16	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
17	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
18	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
19	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
20	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
21	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
22	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
23	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
24	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
25	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
26	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
27	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
28	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
29	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
30	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
31	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
32	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
33	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
34	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
35	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
36	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
37	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
38	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
39	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
40	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
41	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
42	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
43	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
44	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
45	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
46	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
47	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
48	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
49	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
50	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
51	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
52	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
53	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
54	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
55	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
56	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
57	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
58	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
59	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
60	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
61	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
62	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
63	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
64	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
65	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
66	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
67	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
68	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
69	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
70	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
71	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
72	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
73	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
74	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
75	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
76	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
77	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
78	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
79	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
80	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
81	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
82	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
83	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
84	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
85	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
86	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
87	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
88	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
89	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
90	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
91	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
92	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
93	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
94	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
95	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
96	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
97	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
98	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
99	2007-07-22T09:25:50	220.67.226.78	TCP	status	1000 > 1287 [ACK] Seq=173 Ack=107 Wnn=7320 Len=0
100	2007-07-22T09:25:50	220.67.226.78	TCP	status	

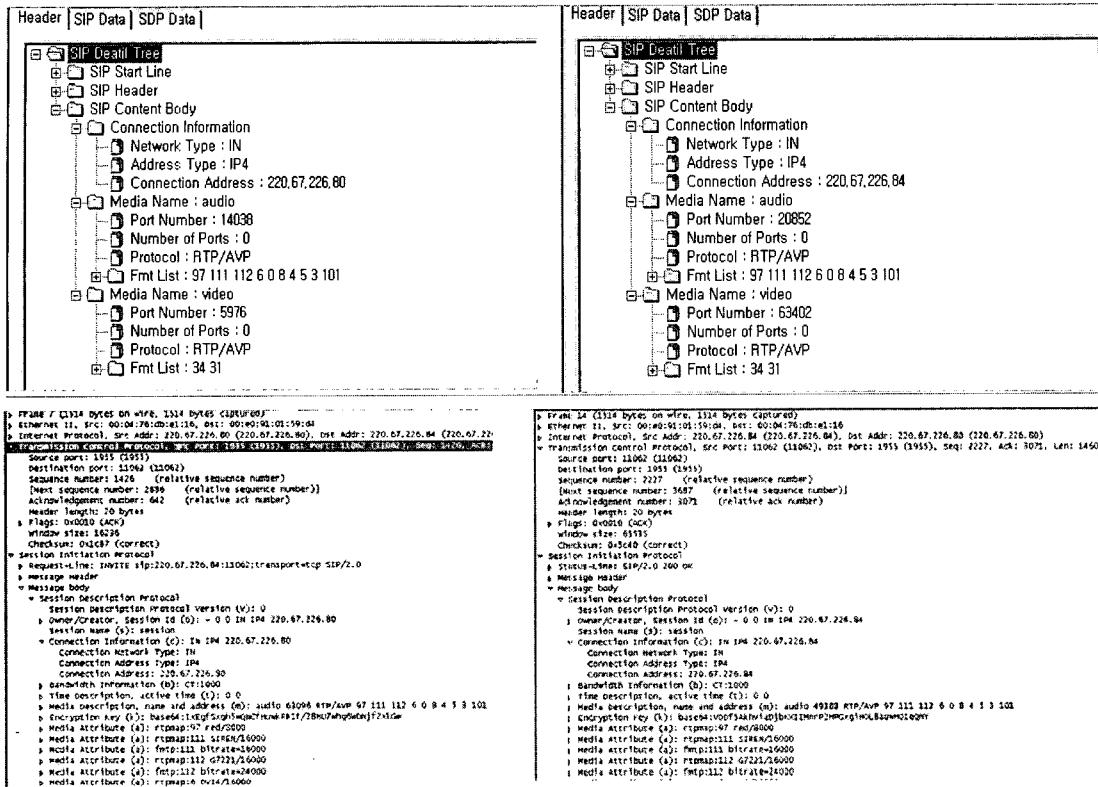


그림 35 세션 정보 분석 결과 - 화상 대 음성

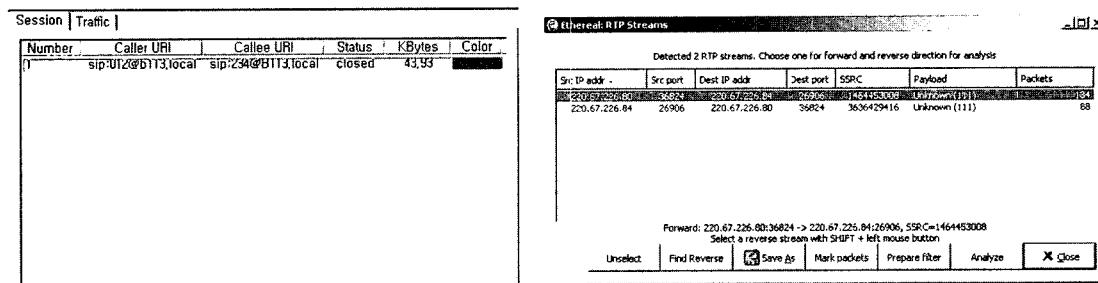


그림 36 수립된 SIP 세션 - 화상 대 음성

사용자의 미디어 정보를 이용하여 SIP 세션별 세션 정보를 관리하는 능력을 실험하였다. 그림 34는 호출자는 화상 채팅 기능을 갖고 호출 받은 사용자는 음성 채팅 기능을 갖는 SIP User Agent를 사용하여 SIP 세션 수립 시 발생된 SIP 패킷을 수집한 화면이다.

그림 34의 수집된 패킷을 보면 STAT(상단그림) 경우 SIP 세션이 맺어지기까지 수집한 패킷은 총 7개이다. SIP 세션 수립을 요청하는 INVITE 메시지와 SIP 세션 수립에 응답하는 200 OK 메시지가 수집되었다는 것을 알 수 있다. Ethereal(하단그림) 경우 STAT와 동

일한 7개의 세션 패킷이 수집되었음을 알 수 있다. 독립적 SIP 수집이 불가능하여 SIP 이외의 다른 패킷들이 수집됨을 알 수 있다.

그림 35는 초기 SIP 세션 수립 시 발생된 세션 정보를 분석한 결과이다. 초기 Request 메시지에 대한 응답 메시지로 Reponse 메시지가 오게 된다. STAT(상단그림) 경우 Request 메시지와 Response 메시지에 개별 분석한 값을 보여주고 있다. Media Name이 화상과 음성 채팅을 요청함을 알 수 있다. Ethereal(하단그림) 경우 세션 정보에 대한 분석 결과가 STAT와 비슷하게

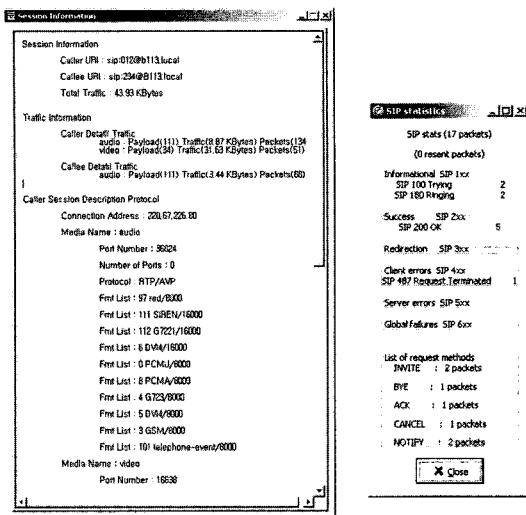


그림 37 SIP 세션 정보 - 화상 대 음성

보이고 있다. Media Name은 음성 채팅은 인식하나 화상 채팅에 대한 분석은 하지 못함을 알 수 있다.

그림 36은 화상채팅 기능을 갖는 SIP User Agent와 음성 채팅 기능을 갖는 SIP User Agent간에 SIP 세션이 맺어짐을 보여주고 있다. STAT(좌측그림)의 경우 하나의 세션이 성립됨을 보여주고 있다. Ethereal(우측그림)의 경우 SIP를 통한 세션의 성립함을 보여주지 않고, RTP 데이터를 분석하여 현재 2개의 Agent가 존재함을 보여주고 있다. RTP 데이터가 수집되지 않으면 Agent의 존재를 볼 수 없음을 알 수 있었다.

그림 37은 세션에 대한 정보를 보여주고 있다. STAT(좌측그림)의 경우 각 세션에 대한 정보를 따로 저장함을 볼 수 있다. Ethereal(우측그림)의 경우 각 세션에 정보는 저장하지 않고 있었으며, 수집된 SIP 패킷에 대한 상태코드를 분류하였다.

4.2.3 실험 3 - 미디어 트래픽(RTP 패킷) 분석 실험

SIP 세션 생성 시나리오를 이용하여 15개의 SIP 세션을 수립한 후에 RTP 패킷을 이용하여 발생되는 멀티미디어 통신 트래픽을 분석하여 SIP 세션별 발생된 트래픽 통계와 미디어 종류별 트래픽 통계를 처리하는 능력을 실험하였다.

그림 38은 STAT의 미디어 트래픽 분석 및 미디어 종류별 트래픽 통계를 보여주고 있다. 총 15개의 세션이 생성되어 정상적으로 인식함을 볼 수 있다. 세션 중 14 번째 세션에서 가장 많은 트래픽이 발생되었고, 12번째 세션에서 가장 적은 트래픽이 발생되었다. 하단좌측 그림에 의해 SIP 세션의 세부 정보를 통하여 화상 채팅이 이루어졌으며, 페이로드 종류가 34인 H.263 비디오 코덱을 이용한 상 래피이 SIP 세션에서 발생된 총 트래픽

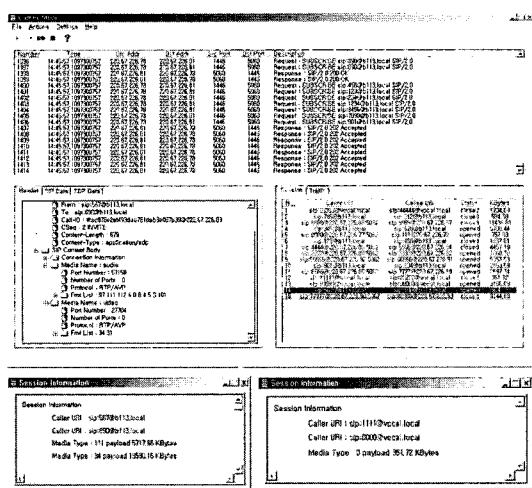


그림 38 STAT 미디어 트래픽 분석 결과 및 미디어 종류별 트래픽 통계

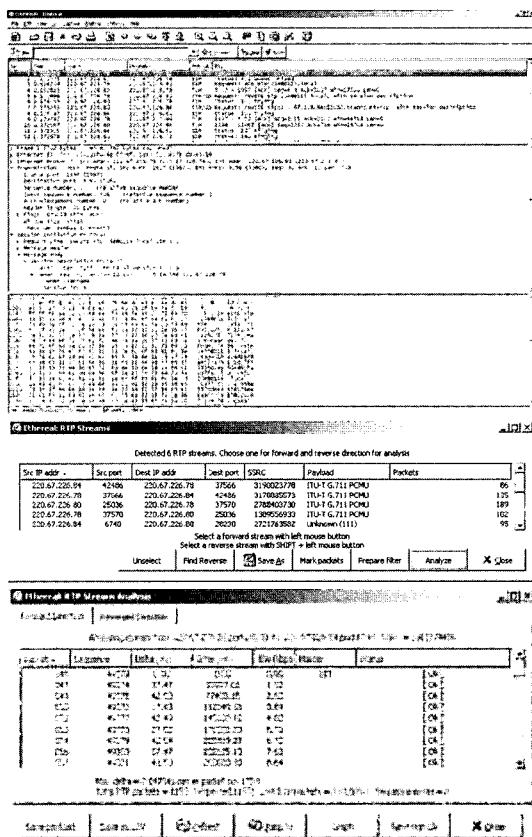


그림 39 Ethereal 미디어 트래픽 분석 결과 및 미디어 종류별 트래픽 통계

의 약 70% 차지하였다는 것을 알 수 있고, 하단우측

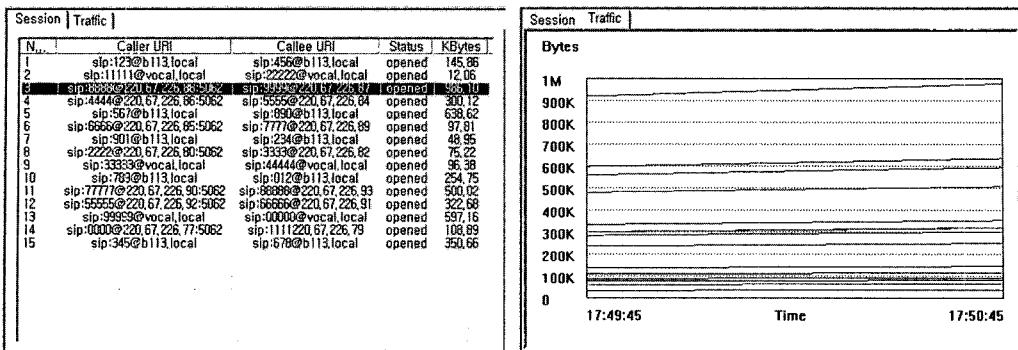


그림 40 STAT의 SIP 세션 생성 시나리오에 의해 생성된 세션 및 그래프 인터페이스 기능의 실험 결과

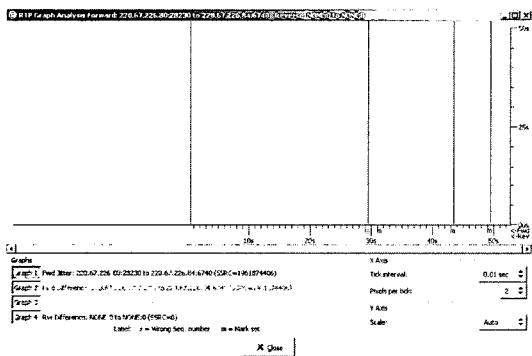
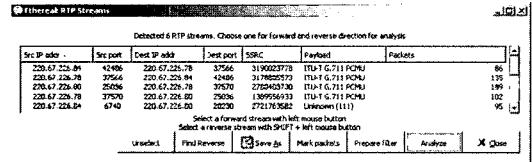


그림 41 Ehtereal의 SIP 세션 생성 시나리오에 의해 생성된 RTP Agent 개수 및 Agent에 해당하는 그래프 인터페이스 기능의 실험 결과

그림에 의해 페이로드 종류가 0인 PCMU 오디오 코덱을 사용한 음성 채팅이 이루어졌으며 SIP 세션에서 발생된 총 트래픽이 음성 트래픽임을 알 수 있었다. 그림 39는 Ehtereal의 미디어 트래픽 분석 및 미디어 종류별 트래픽 통계를 보여주고 있다. 따로 세션별로 분류하지 않고 Agent의 개수와 통신정보를 보여주고 있다. 5번쨰 정보에 의하면 Unknown(111)이라 하여 화상통신은 분석하지 못함을 알 수 있다. Agent 정보의 분석정보는 RTP 패킷에 대한 데이터량과 전송된 시간을 보여주고 있다.

4.2.4 실험 4 - 그래프 인터페이스 실험

실험 3과 같이 SIP 세션 생성 시나리오를 이용하여 15개의 SIP 세션을 수립한 후에 SIP 세션별 실시간으로 발생되는 트래픽 정보를 이용하여 그래프 인터페이

표 10 부하 테스트시 사용된 시스템의 사양

구 분	내 용
OS	Windows 2000 Server
CPU	Pentium 4 - 1.8GHz
RAM	768MB
NIC	Realtek RTL8139(A) PCI Fast Ethernet Adapter

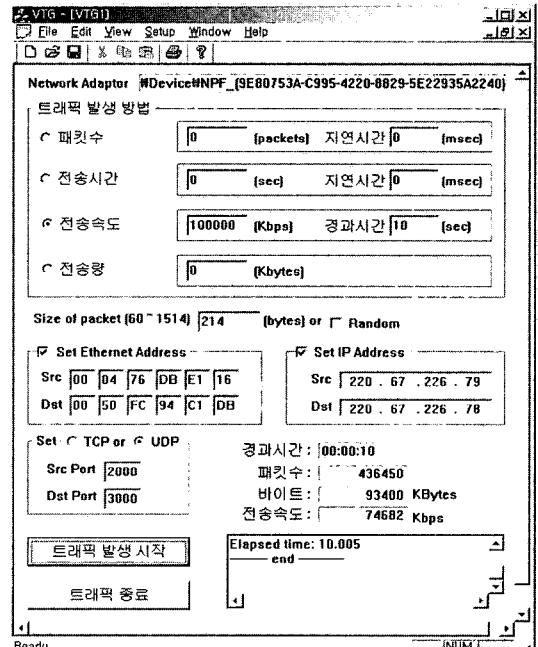


그림 42 트래픽 발생기의 실행 화면

스 기능이 정상적으로 동작하는지 실험하였다.

그림 40은 STAT의 SIP 세션 생성 시나리오에 의해 수립된 SIP 세션과 세션의 상태 정보를 보여주고 있으며, 수립된 SIP 세션의 트래픽 량을 선 그래프를 이용하여 SIP 세션별 트래픽 정보를 표현하고 있다. 그림 40 결과를 통하여 SIP 세션이 수립되었을 경우 SIP 세

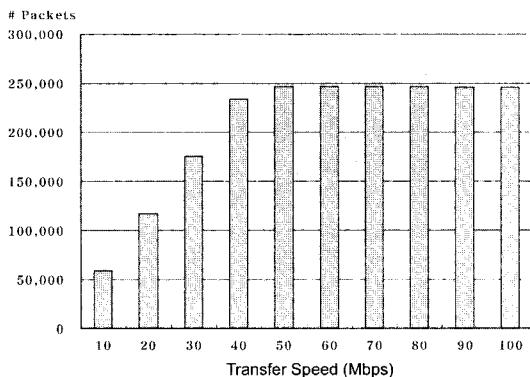


그림 43 트래픽 발생기에 의해 발생된 음성 트래픽 량

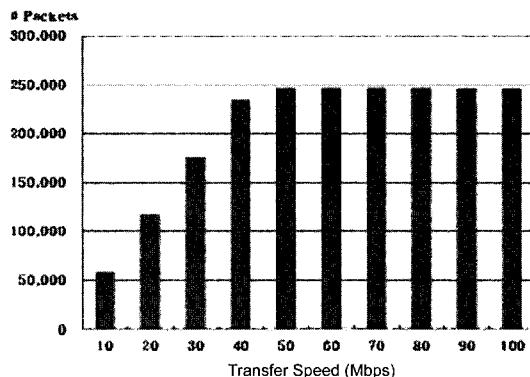


그림 45 Ethereal 부하 테스트 결과 - 음성 트래픽

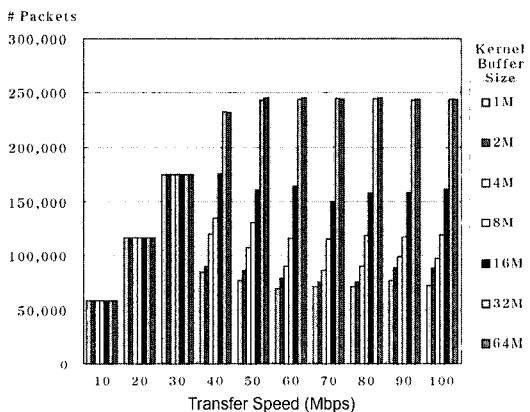


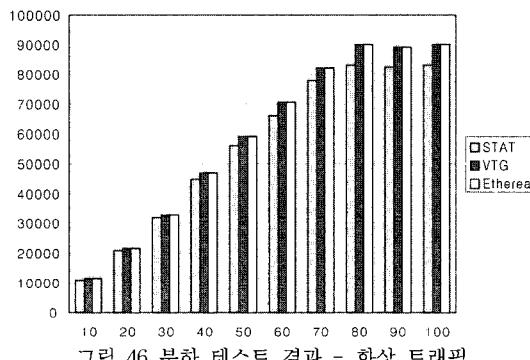
그림 44 STAT 부하 테스트 결과 - 음성 트래픽

선에서 발생되는 트래픽 정보를 그래프 인터페이스가 실시간으로 화면에 정상정으로 표시하는 것을 확인할 수 있다. 그림 41은 Ethereal의 RTP Agent 개수와 Payload 값을 보여주고 있으며, 해당 Agent의 트래픽량을 막대 그래프로 이용하여 RTP 데이터 정보를 표현하고 있다. Agent에 해당하는 RTP 데이터량은 시간별로 분류하여 RTP 패킷이 전송된 시점을 막대그래프로 표현하였다.

4.2.5 실험 5 - STAT의 부하 테스트

트래픽 발생기를 이용하여 RTP 패킷을 생성하고 전송속도를 10Mbps부터 100Mbps까지 늘려가면서 트래픽을 발생시켜 미디어 트래픽 분석 성능을 측정하였다. 표 10은 부하 테스트 시 사용된 시스템의 사양이고, 그림 42는 실험에 사용한 트래픽 발생기의 실행 화면이다.

멀티미디어 트래픽 중 페이로드 타입이 0인 PCMU 코덱을 사용한 음성 트래픽인 경우에 160byte의 음성 데이터가 RTP 패킷에 내장된다. 그림 43은 이와 동일한 음성 트래픽을 트래픽 발생기를 이용하여 10초 동안 지연시간 없이 발생시킨 결과로 트래픽을 전송속도별로



보여주고 있고, 그림 44는 트래픽 발생기에 의해 발생된 음성 트래픽을 STAT의 커널 버퍼 크기 및 패킷 버퍼 크기를 늘려가면서 실험한 결과이다. Ethereal은 따로 커널 버퍼크기를 조절하는 부분이 없다.

그림 44의 실험 결과에서 보듯이 커널 버퍼 크기가 16Mbyte 미만인 경우에는 최대 70% 이상 손실이 발생하였다는 것을 알 수 있었으며 커널 버퍼 크기를 32Mbyte 이상으로 설정하였을 경우에는 99% 이상의 미디어 트래픽 분석 성능을 보였다. 그림 45은 STAT의 커널 버퍼 크기가 32Mbyte 일 경우와 동일한 결과를 가져왔다.

멀티미디어 트래픽 중 화상 트래픽은 음성 트래픽과 데이터 크기면에서 차이가 있다. 즉 화상 데이터는 프레임 단위로 전송되는데 움직임이 없을 경우에는 하나의 RTP 패킷에 내장되고 움직임이 많을 경우에는 여러 개의 패킷에 내장되어 전송된다. 실험에서는 화상 데이터의 크기를 1Kbyte로 고정시키고 10초 동안 지연시간 없이 트래픽 발생기를 이용하여 화상 트래픽을 발생시켰다. 그림 46은 화상 트래픽을 STAT의 커널 버퍼 크기를 64Mbyte로 설정하여 분석 성능을 측정 한 결과로 전송속도가 100Mbps인 경우 약 7% 미만의 패킷 손실

표 11 STAT와 Ethereal의 비교 분석

	Ethereal	STAT
실험 1 SIP 패킷 수집 및 분석	- 로그인과 로그아웃에서 발생되는 "NOTIFY", "SUBSCRIBE" 메시지에 대해 분석하지 못함.	- 기존의 프로토콜 분석기와 동일한 패킷을 수집, 분석하며, SIP 패킷 로그인과 로그아웃에서 발생되는 "NOTIFY", "SUBSCRIBE" 메시지도 추가적으로 분석함.
실험 2 세션 정보 분석 및 SIP 세션 정보 관리	- 화상 채팅에 대한 미디어 정보를 보여주지 못함. - SIP 패킷을 통한 세션의 성립함을 보여주지 않고 Agent별 RTP 데이터에 의해 분류하였다. - 세션에 대한 정보가 전혀 없음	- 화상 및 비디오 정보를 보여줌 - SIP 패킷을 통한 세션의 성립함을 보여주며, 세션에 대한 정보를 보여준다.
실험 3 미디어 트래픽 분석	- 세션별 통계정보가 아닌 각 Agent의 정보를 보여줌	- 세션별 발생된 트래픽 통계와 미디어 종류별 트래픽 통계를 처리하는 능력을 가짐
실험 4 그래프 인터페이스	- 해당 Agent의 트래픽량을 막대그래프로 표현하여 전체 트래픽량을 알 수 없음	- 세션별 실시간으로 발생되는 트래픽 정보를 선 그래프를 이용하여 보여줌

이 발생하였다. Ethereal 경우 트래픽 발생기에 의해 발생된 화상 트래픽과 거의 차이가 없음을 발견하였다.

실시간 통신 서버와 VOVIDA 서버에 생성된 계정을 이용하는 MSN Messenger와 SIPSet를 사용하여 4가지 실험과 트래픽 발생기(Traffic Generator)를 이용하여 1가지 실험을 하였다. 표 11은 총 5가지의 실험을 통해 얻은 두 프로토콜 분석기의 비교 분석한 것이다. 본 논문에서 구현한 STAT는 일반적인 프로토콜 분석기의 기능을 포함한 실시간 분석을 통하여 모니터링 기능까지 추가하였다. 실시간 맷어지는 세션을 보여줌으로써 세션에 대한 정보 및 트래픽 정보를 알 수 있다.

5. 결론 및 향후 연구

인터넷 텔레포니 기술은 기존 공중전화망 중심의 음성 및 데이터망 서비스에서 패킷망 중심의 음성 및 데이터망 서비스로의 전환적인 측면에서 매우 중요한 기술로써 인식되고 있지만 인터넷 텔레포니 기술의 역사 는 그리 길지 않고 인터넷 기술의 급속한 발전에 따라 아직도 표준으로 정리되지 않은 분야들이 매우 많다. 기본적인 세션 제어에 대한 기술만이 어느 정도 완성단계에 들어가 있으며 관리를 위한 요소와 각 시스템간의 상호 운용성 등에 대해서는 계속적으로 표준화가 진행되고 있다.

본 논문에서는 이러한 상황을 고려하여 SIP를 이용한 인터넷 멀티미디어 통신 시스템에서 발생되는 트래픽을 분석하여 망을 진단할 수 있는 프로토콜 분석기 STAT(SIP based Traffic Analysis Tool)를 설계하고 구현하였다.

4장 실험 및 성능 평가에서는 실제 프로젝트에서 많이 사용되는 SIP 서버인 실시간 통신 서버와 VOVIDA 서버, SIP User Agent인 Microsoft Messenger와 SIPSet를 이용하여 실험과 성능을 평가하였다. 실험 결과에서 보듯이 SIP를 기반으로 하는 SIP User Agent에서 발생된 SIP 패킷이 정상적으로 분석되었고 분석된

결과를 통하여 수립된 SIP 세션이 인식되고 처리되었다. 또한 수립된 SIP 세션에서 발생된 멀티미디어 통신 트래픽을 분석하여 미디어 종류와 미디어 종류별 트래픽 정보를 알아낼 수 있다는 것을 보여주었다.

실험 1에서는 사용되는 SIP 서버와 SIP User Agent의 종류에 상관없이 발생되는 SIP 패킷을 모두 수집하여 분석하는 것을 보았다. 또한 SIP 서버와 SIP User Agent의 기능에 따라 로그인, SIP 세션 요청 및 로그아웃의 각 과정에서 발생되는 SIP 패킷 런에 많은 차이가 있음을 발견하였다. 즉 온라인 상태 인식 기능의 유무에 따라 많은 수의 "NOTIFY"와 "SUBSCRIBE" SIP 패킷이 추가적으로 발생되었다. 멀티미디어 통신 트래픽은 대부분이 미디어 트래픽이 차지한다고 볼 수 있지만 SIP 서버에 로그인, SIP 세션 요청 및 로그아웃 과정에서 추가적으로 발생되는 트래픽을 무시할 수 없음을 확인하였다.

실험 2에서는 SIP 세션 수립 과정에서 발생되고 SDP에 의해 기술되는 세션 정보를 정상적으로 분석하여 트리 컨트롤을 이용한 사용자 인터페이스에 나열되는 것을 볼 수 있었고, SIP 세션이 수립될 경우 SIP 세션의 상태 정보와 호출자와 호출 받은 사용자의 세션 정보가 지속적으로 유지되고 관리되는 것을 호출자는 화상 채팅 기능을 갖고 호출 받은 사용자는 음성 채팅 기능을 갖는 경우의 SIP 세션 수립 실험에서 발생된 SIP 세션 변경 과정을 통하여 확인할 수 있었다.

실험 3에서는 호출자와 호출 받은 사용자간의 실시간 미디어 데이터 전달에 사용되는 RTP 패킷을 분석하여 SIP 세션별 발생된 트래픽 정보를 얻을 수 있었고 SIP 세션별 발생된 트래픽을 음성 트래픽과 화상 트래픽으로 나누어 세부적으로 표시하여 미디어 종류별 트래픽 통계 정보를 확인하였다.

실험 4에서는 미디어 통계를 표시하는 그래프 인터페이스의 성능을 평가하였다. SIP 세션 생성 시나리오를 이용하여 15개의 SIP 세션을 생성한 후에 SIP 세션별

로 발생되는 트래픽 정보를 이용하여 선 그래프, 원 그래프와 막대 그래프가 실시간으로 화면에 트래픽량을 표시하는 것을 알 수 있었다.

실험 5에서는 트래픽 발생기를 이용하여 실제 환경에서 발생되는 트래픽과 유사한 트래픽을 발생시켜 부하 테스트를 수행한 결과 실험 환경에서 STAT가 정상적으로 동작하기 위해서 요구되는 커널 메모리 크기가 32Mbyte 이상이 되어야 한다는 것을 알 수 있었다.

실험 및 성능 평가를 통하여 하나 이상의 참여자로 구성되는 세션을 초기화하고, 변경 및 종료하기 위해 사용하는 SIP 패킷, SIP 패킷의 메시지 본문에 멀티미디어 세션을 기술하기 위하여 사용되는 SDP와 실시간 미디어 데이터를 세션 참여자간 전송에 사용되는 RTP 패킷을 모두 수집하고 분석하여 멀티미디어 통신 트래픽을 측정하고 세부적으로 분석할 수 있다는 것을 보였다.

본 논문에 이은 향후 연구로는 다양한 조건으로 트래픽을 분석할 수 있도록 그래프 인터페이스의 기능 보완과 사용자 인터페이스의 개선이다. 그리고 현재 많이 사용되고 있는 이동성이 좋고 휴대하기 편리한 WinCE 기반 PDA용 SIP 기반 멀티미디어 통신 시스템을 위한 프로토콜 분석기를 개발하는 것이다.

참 고 문 헌

- [1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, SIP: Session Initiation Protocol, RFC 3261, Jung 2002.
- [2] ITU-T H.323, Packet-based Multimedia Communications Systems, September 1999.
- [3] IETF SIP Working Group, <http://www.ietf.org/html.charters/sip-charter.html>
- [4] <http://www.3gpp.org>, 3rd Generation Partnership Project Homepage.
- [5] <http://www.3gpp2.org>, 3rd Generation Partnership Project 2 Homepage.
- [6] 3rd Generation Partnership Project, "IP Multimedia(IM) Subsystem-Stage 2," 3GPP TS23.228 V5.1.0, 2001.
- [7] International Softswitch Consortium, "ISC Reference Architecture V 1.2," 2002.
- [8] <http://www.voip-forum.or.kr>, VoIP Forum Home-page.
- [9] 황세진, 박성순, "차세대 VoIP 통신개요 및 기술동향", 한국멀티미디어학회지, Vol.7, No.5, pp.139-147, 2003.
- [10] 이강석, 염창선, 황기현, "CTI/VoIP 기반 인터넷 콜 시스템의 설계에 관한 연구", IE Interface, Vol.15, No.4, 2002.
- [11] M. Handley, V. Jacobson, SDP: Session Description Protocol, RFC 2327, April 1998.
- [12] H. Schulzrinne, S. Casner, R. Frederick and Y. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 1889, January 1996.
- [13] F. Risso and L. Degioanni, Analyzer: a public domain protocol analyzer, <http://analyzer.polito.it>
- [14] E. Wedlund, H. Schulzrinne, "Mobility Support using SIP," VON Europe Spring 2000, June 2000.
- [15] <http://winpcap.polito.it/docs/default.htm>, WinPcap Homepage.
- [16] F. Risso and L. Degioanni, "An Architecture for High Performance Network Analysis," Proceedings of the Sixth IEEE Symposium on Computers and Communications, pp. 686-693, 2001.
- [17] <http://www.microsoft.com/office/livecomm/prodinfo/default.mspx>, MS Live Communication Server 2003 Homepage.
- [18] <http://www.vovida.org>, VOVIDA Homepage.
- [19] <http://www.microsoft.com/windows/messenger>, Microsoft Messenger Homepage.
- [20] <http://www.vovida.org/applications/downloads/sipset>, SIPSet Homepage.
- [21] 정인환, 김진환, 비쥬얼 이더넷 트래픽 발생기의 설계 및 구현, 제18회 정보처리학회 추계학술대회, Vol.9, No. 5, 2002.
- [22] IETF, "SIP: Session Initiation Protocol," RFC 3261, June 2002.
- [23] <http://www.iana.org/assignments/rtp-parameters>, RTP Parameters.
- [24] Ethereal, Public Domain Protocol Analyzer, <http://www.ethereal.com/>
- [25] M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, 최초 Version SIP: Session Initiation Protocol, RFC 2543, RFC3261에 의해 무효화 됨, March 1999.
- [26] <http://www.rfc-editor.org/>, The Publisher of the RFCs.
- [27] <http://www.rfc-editor.org/rfcsearch.html>, RFC 문서 검색.



정인환

1984년 한양대학교 원자력공학과(학사)
2000년 한국과학기술원 정보및통신공학
과(박사). 1985년~1998년 삼성전자(주)
부장. 2000년~2001년 (주)두루넷 팀장
2001년~현재 한성대학교 컴퓨터공학과
조교수. 관심분야는 분산시스템, 프로토
콜분석기, 멀티미디어통신