산업기술연구(강원대학교 산업기술연구소 논문집), 제25권 A호, 2005.
Journal of Industrial Technology, Kangwon Natl. Univ., Korea, No. 25 A, 2005.

SNMP를 이용한 효율적인 시스템 모니터링

Low Overhead System Monitoring Based on SNMP

김 태 수* 정 창 영* 김 동 억* 김 용 석**
Kim, Tae-Su Jeong, Chang-Young Kim, Dong-Uk Kim, Yong-Seok

Abstract

SNMP is a standard protocol for management of networking devices. Nowadays, most computer systems have capability to act as SNMP agents. In this paper, we describe our system monitoring software based on SNMP. It consists of a monitoring server, SNMP agents, and client programs. The monitoring server collects status information from the SNMP agents running on the monitoring target graphical display. We developed two version of clients, Java based standalone program and Web based program. Since most known SNMP packages are too big and heavy, we developed an efficient version of SNMP library for out system monitoring

키워드 : *시스템 모니터링, SNMP, java, 웹*
Keywords : *system monitoring, SNMP, java, web*

1. 서론

네트워크로 연결된 컴퓨터 시스템 및 네트워킹 장비들을 관리하는 입장에서는 원활한 네트워크 상황을 유지하기위해 각각의 시스템들이 처한 상황을 알고 그에 대한 적절한 조치를 취할 수 있어야 한다. 이러한 네트워크 관리라는 목적을 달성하기 위해 전 세계적으로 여러 가지 방법이 연구 되었고, 이러한 연구의 결과 중 네트워크가 보급되기 시작한 초기에 네트워크 관리를 위해 사용되었던 방법으로는 ICMP(Internet Control Message Protocol)[1]가 있다. 하지만 이는 단순한 기능만을 제공하는 프로토콜로써 원하는 시스템이 현재 가동 중인지 아닌지를 판별하고 도달하는데 걸리는 시간이 어느 정도인지를 알려주는 ping서비스를 제공해주는 제한적인 프로토콜이다. 이러한 프로토콜로는 현재 구성이 점점 복잡해지는 네트워크를 관리하기에는 너무 부족하다. 또한 네트워크의 중요성이 강조되면서 관리에 있어 더 많은 정보를 원하는 현재에는 그 사용이 부적합했다. 이러한 이유로 더 많은 정보를 보고 네트워크를 관리할 수 있는 새로운 네트워크 관리 프로토콜이 생겨나기 시작하였다. 여러 가지 프로토콜이 생겨나자 국제기구인 IAB (Internet Architecture Board)에서 여러 프로토콜의 기능을 복합적으로 가지고 있는 SNMP(Simple Network Management Protocol)를 국제 표준으로 선택하였다.

* 강원대학교 전기전자정보통신공학부, 컴퓨터 전공
** 강원대학교 전기전자정보통신공학부 교수, 공학박사
(주) 본 논문은 BK21 사업의 지원에 의해 이루어졌음

SNMP의 등장과 함께 해당 프로토콜을 사용하여 네트워크를 관리하기 위한 여러 응용프로그램과 라이브러리들이 생겨났다. 이중 대중적으로 많이 알려진 것으로 공개 소프트웨어인 ucd-snmp 프로젝트가 있다. 하지만 이렇게 생긴 라이브러리들은 그 양이 방대하고, 복잡하여 응용 프로그래머가 사용하기에 많은 어려움이 있다. 너무 많은 기능을 담고 있으며, 필요 이상으로 많은 요소들을 첨부한 라이브러리들은 학습적인 역할로서나, 기능적인 역할을 하기위해서 너무 어렵게 구성되어 있다. simple 이란 이름에 걸맞지 않게 프로토콜의 사용을 위한 방법은 복잡한 것이다. 그래서 본 논문에서는 SNMP 기반의 응용프로그램을 위해 보다 사용이 간편하고 이해하기 쉬운 SNMP 라이브러리를 제작하였다.

SNMP는 MIB (Management Information Base)[2]라는 특수한 자료를 이용해 관리를 하게 되는데 네트워크뿐만 아니라 시스템 상황(cpu, 메모리, 디스크사용률 등)까지 정보를 수집해 올 수가 있다. 본 논문에서는 자체 제작한 SNMP 라이브러리를 이용하여 네트워크 관리뿐 아니라, 시스템 전체적인 상황을 관리하는 시스템 모니터링 소프트웨어를 개발하였다

본 논문은 1장의 서론에 이어, 2장에서 SNMP의 동작 원리 및 구조에 대한 기술을 하며, 3장에서는 본 논문에서 제시한 시스템 모니터링 서버에 관한 전반적인 내용을 기술한다. 그리고 4장에서는 클라이언트 프로그램에 관해 기술하고 5장에서 결론으로 끝맺는다.

2. SNMP의 동작 원리 및 구조

2.1 SNMP의 동작 원리

SNMP는 관리 대상이 되는 에이전트와 관리를 하는 주체인 매니저로 구성이 되어있다. SNMP의 동작 원리는 매니저의 요청과 에이전트의 응답이라는 방식을 기본으로 동작하게 된다. 매니저가 포트번호 161번을 이용하여 에이전트에게 UDP로 요청 패킷을 보내면 에이전트가 응답하는 방식이다. 이러한 기본적인 요청 방식이 아닌 에이전트가 능동적으로 매니저에게 자신의 특수한 상황을 알리는 수단으로 트랩메시지를 사용한다. 트랩 메시지는 162번 포트를 이용하여 UDP로 전달된다.[3]

여기서 에이전트란 호스트 컴퓨터나 브리지, 라우터, 허브 같은 장비로 독자적으로 장비 자신의 트랙픽에 관련된 수치를 모니터링하고 특정 정보를 자신의 MIB의 해당하는 객체에 저장해 두었다가 매니저로 부터의 트래픽 정보요구나, 특정 동작 요청에 응답하고 특정 상황 발생 시 매니저에게 중요 정보를 제공하는 역할을 하는 장비를 말한다. 이때 사용되는 MIB는 특정 정보를 하나의 오브젝트로 하여 여러 오브젝트들을 계층적구조로 관리하는 것을 말하는데 트래픽 정보나 특정 정보를 나타내는 오브젝트들의 모임이라고 할 수 있다. MIB에 대한 자세한 구조는 뒤에 설명한다.

SNMP가 전달하는 메시지 중 get과 set 메시지는 요청, 응답이라는 방법을 사용한다. get메시지는 agent에게 MIB의 특정 오브젝트 정보를 단순히 요청하는 GETRequest와 요청 MIB의 오브젝트의 다음 정보를 요청하는 GETNext가 존재하여 이 두 가지 중 하나의 요청 메시지를 이용하여 원하는 정보의 응답을 받는다. set메시지는 에이전트의 MIB중 특정 오브젝트의 값을 원하는 값으로 변경할 때 사용된다. 해당 set 메시지를 전달(요청)하면 에이전트로 지정된 오브젝트의 값을 변경한 후 응답을 해준다. 트랩 메시지는 에이전트가 매니저에게 특수한 상황이나 정보를 알려 줄때 사용하는 방법이다. 트랩 메시지만 매니저로부터 에이전트로 가는 요청 과정이 없다.

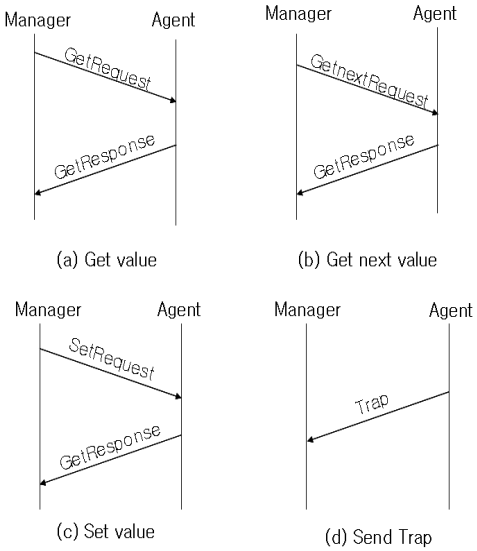

그림 1 SNMP의 작동 도식

매니저와 에이전트와의 패킷 송수신을 간단히 도식화하면 그림 1과 같이 된다. 그림 1에서와 같이 대부분의 작동이 요청과 수신의 쌍으로 이루어져 있으며, 트랩 메시지의 경우만 에이전트가 능동적으로 메시지를 보내는 것을 알 수 있다.

2.2 MIB의 구조

MIB는 SNMP에 의해 관리되는 객체들을 말하는 것으로 트리구조로 되어있다. 각각의 객체에는 OID (Object IDentifier)가 있는데 이를 이용해 특정 객체의 정보를 요청하고 객체에 해당하는 값으

로 응답하게 된다. 이 OID는 유일하며, 정수의 나열로 표현을 한다. 각 정수들은 부모의 숫자를 가지고 있어 관리에 용이성과 유일한 번호 부여에 도움을 준다.

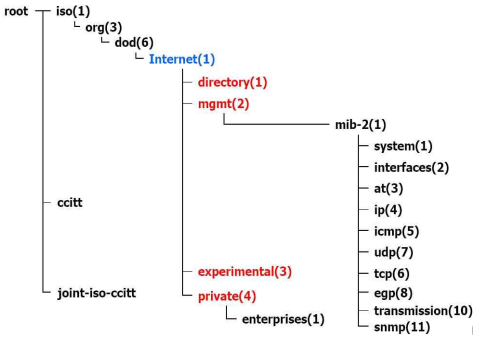

그림 2 MIB의 계층 구조

MIB는 트리형식으로 각각의 OID를 부여 하게 되는데 그림 2와 같은 트리 형식을 사용한다. 괄호 안의 숫자들을 나열하는 방법을 이용하여 리프 노드의 객체에 대한 OID를 설정한다. 이 중 가장 많이 사용 되는 MIB의 객체는 일반적인 트래픽 관리를 위한 mgmt와 각 에이전트의 시스템 제조업체나 운영체제 제공업체에서 제공하는 private 필드이다. 이 private 필드는 해당 시스템의 특수한 정보를 담고 있는 객체를 나타내는 것으로서 에이전트의 업체들이 설정한 여러 가지 시스템 관련 정보를 담고 있다. private 객체를 이용하여 시스템 관리를 위한 정보들을 수집할 수 있다. 즉 에이전트의 특수한 OID를 이용하여 네트워크 관리뿐만이 아니라 시스템 관리도 가능해지는 것이다.

MIB의 각 object들에 대한 상세는 SMI문서로 기술하게 되어있다. SMI문서는 RFC1155[4]에 기술되어있는 형태로 적혀져서 시스템에 제공된다. 관리자는 이 문서를 봄으로서 각 객체들에 대한 구조, 데이터 타입, 객체의 정의, 접근 허가권 등을 알 수가 있다. 보통 mgmt 필드같이 공통적인 MIB의 object들은 각 에이전트마다 동일하기 때문에 특별히 객체에 대한 조사를 하지 않아도 해당 데이터를 적절히 가져와 사용할 수 있다. 하지만 private같은 vender 독립적인 객체들은 SMI문서를 파싱하여 제공하는 객체의 OID라든지 데이터 타입 사용용도 등을 알아내야 한다. 결론적으로 특정 시스템에서 자신이 원하는 정보(객체)에 대한 OID를 알기 위해서는 SMI 문서를 살펴야한다는 것이다. 하지만 SMI문서를 본다는 것이 생각처럼 쉬운 일이 아니다. 이를 위해 문서적으로 복잡하게 구성되어 있는 SMI에서 간략하게 OID, 데이터 타입, 역할에 대한 기술만을 뽑아내는 유틸리티를 만들었다. 해당 유틸리티의 입력으로 SMI문서를 넣어주면 SMI문서를 파싱하여 적절한 형식으로 출력해 준다.

2.3 SNMP의 구조

SNMP는 pdu의 형태로 데이터를 교환한다. 요청과 응답 시 교환되는 메시지는 버전 정보, 커뮤니티 정보 및 SNMP pdu이다. 버전과 커뮤니티 정보는 버전 1일 경우 0, 2일 경우 1 이다. 커뮤니티는 agent의 특정 정보의 접근 및 수정을 제한하기위한 보안의 수단으로 사용 되는 것으로써 에이전트에 해당 커뮤니티에 대한 접근 권한을 설정하게 된다. 특정 커뮤니티를 사용하는 매니저에게 데이터를 쓸 수 있는 권한을 주거나, 읽기만 가능하게 하는 방법을 사용한다. 대부분의 시스템에서는 public 커뮤니티를 기본적으로 제공하는데 해당 커뮤니티는 set에 대한 권한(쓰기 권한)이 대부분 제한되어 있으며, get요청(읽기)에는 제한이 거의 없다.

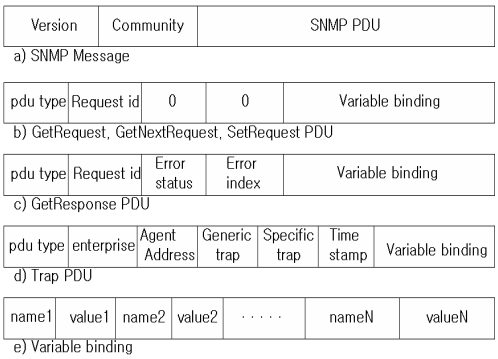

그림 3 SNMP 프로토콜 포맷

SNMP pdu에 대한 자세한 포맷은 그림 3과 같다[5]. 일반적인 get, set에 대한 메시지 포맷은 pdu type에 get, set, getnext등의 pdu의 종류가 들어가며 request id에는 송신자가 임의로 생성한 숫자가 들어간다. 이 숫자는 메시지 응답 시 받은 pdu의 request id와 비교하여 자신이 요청한 메시지에 대한 응답인지를 확인하기 위하여 사용하는 것이다. 그 뒤로 0이 두 개 들어가는데 이는 회신용 메시지에서 만 사용하는 필드이다. (c)의 응답 pdu 포맷을 보면 error-state와 error-index 필드가 있는 것을 확인 할 수 있다. 이 두 개의 필드는 요청한 메시지에 대해 에러가 있는지 없는지를 판단하여 수신자가 송신자에게 알려주는 역할을 한다. 즉, agent가 manager에게 받은 메시지에 에러가

포함 되어있으면 이 필드를 이용하여 manager에게 해당 메시지기 잘 못되었음을 알려주는 것이다. error-state의 경우 에러가 발생했을 경우 어떠한 에러인지를 알려주는 역할을 하며 error-index는 송수신 메시지의 마지막 필드인 variable-binding에서 몇 번째 OID가 에러가 났는지를 알려준다. variable-binding은 그림에서 (e)항을 보면 그 형태가 나오는데 송신자가 수신자에게 특정 Object의 정보를 요청하는 경우, 여러 개의 OID에 대해 한 번에 요청할 수 있게 해준다. 즉, 메시지 전송은 횟수를 줄이고 얻어오는 정보의 수를 늘리기 위한 방법이라고 할 수 있다. 정보의 요청과 수신은 name 필드에 OID를 써넣으면 회신 시 해당 name에 대응하는 value 필드에 그 값이 저장되어 돌아오는 방법을 사용하는데 name필드에 여러 OID를 채워 넣어 여러 OID에 대응하는 value를 한 번에 얻을 수 있다.

set의 경우는 수정하고자 하는 OID를 name 필드에 그리고 수정하고자하는 값을 value필드에 써서 전송한다. 수신된 pdu의 error-state 나 error-index 필드에 0 값이 설정되어 돌아오면 해당 요청은 유효한 요청이며, 에러가 없다는 뜻이 되고, 그렇지 않으면 error-index에 해당하는 OID가 error-state에 해당하는 상황에 처해 있다는 것을 알려준다. 일반적인 에러 상황으로는 존재하지 않는 OID를 요청한 경우, 접근 권한이 없는 경우 등이다. 이외에도 패킷 자체의 에러를 검출하기도 하는데 이런 경우는 error-state에만 값이 세팅되어 돌아온다. 패킷의 에러는 패킷의 크기가 너무 큰 경우이다. 프로토콜의 형식을 지키지 못했을 경우에는 아무런 메시지도 회신되지 않는다. 이는 에이전트가 SNMP 메시지라고 판단하지 않기 때문에 전혀 반응하지 않는 것이다. 이를 위해 메시지 송, 수신시 대기시간 설정이 필요하다.

SNMP에서 에이전트가 능동적으로 메시지를 보내는 트랩 메시지의 경우는 pdu의 포맷이 틀리다. 그림의 (d)를 보면 우선 트랩의 타입이 들어가고, 그 후에 enterprise 값이 들어가게 된다. 이 값은 에이전트가 가지고 있는 특수한 값으로 에이전트의 제조회사나 os 제공업체가 가지고 있는 특수한 값이다. 이는 MIB의 private필드 아래에 있는 것으로 MIB의 한 객체이며 OID의 일부이다. 그 다음 항목에는 트랩 메시지를 보낸 agent의 IP가 들어가고, 그 후에는 일반적인 트랩 상황을 나타내는 코드가 들어간다. 이 코드는 agent의 강제 종료, 네트워크정지 등 일반적인 트랩 상황을 말한다. 하지만 이 필드에 6이 들어가면 각 시스템 벤더나 OS 제공업체가 지정한 특수한 트랩 상황으로 앞에 온 enterprise 값과 뒤에 있는 specific 필드의 값을 이용하여 agent의 상황을 알 수 있다. 그리고 agent가 가동된 시간이 나오며, 다시 variable-binding이 나온다. 이는 트랩 OID가 들어가며, 해당 OID와 값을 확인하여 트랩 상황에 대한 더욱 자세한 정보를 알 수 있다. 결국 트랩 메시지는 agent가 manager에게 보내는 상황 보고서이다. 일반적으로 정해진 상황뿐만 아니라 시스템마다 특수하게 일어 날 수 있는 상황 까지도 고려하고 있으며, 이는 private 필드를 참조하여 알 수 있다.

이렇게 구성된 패킷을 네트워크를 통해 전달하기 위해서는 적절한 방법으로 인코딩되어야 한다. 에이전트와 매니저는 pdu를 담은 패킷을 소켓을 이용하여 UDP로 통신하게 되는데, 이 때 패킷은 모든 플랫폼에 독립적인 언어인 ASN.1(Abstract Syntax Notation 1)의 형식에 맞게 인코딩 되어서 전송된다. ASN.1은 플랫폼 마다 다른 데이터 타입을 극복하기 위해 정의된 데이터 정의이다[6]. pdu의 각 데이터들은 ASN.1의 정의에 의한 데이터 타입을 가지고 있으며, 이들은 해당 데이터 타입을 가지고 BER(Basic Encoding Rule)이라는 인코딩 방식에 의해 인코딩되어 네트워크상에서 송수신 될 수 있는 패킷을 구성하게 된다. BER은 각 데이터를 type, length, value로 표현하는 인코딩 방법으로[7], pdu에 있는 각각의 필드들이 데이터 타입, 데이터의 길이, 데이터의 값의 형식으로 인코딩 되는 것이다.

3. 시스템 모니터링 서버

3.1 시스템 모니터링 체계

본 논문에서 제시한 모니터링 시스템은 기본적으로 3가지 요소로 구성되어 있다. 모니터링의 대상이 되는 에이전트 시스템과 해당 에이전트들의 정보를 수집하고 가공하는 서버프로그램, 서버로부터 가공된 정보를 받아 관리자에게 보여주는 클라이언트 프로그램으로 구성되어 있다.

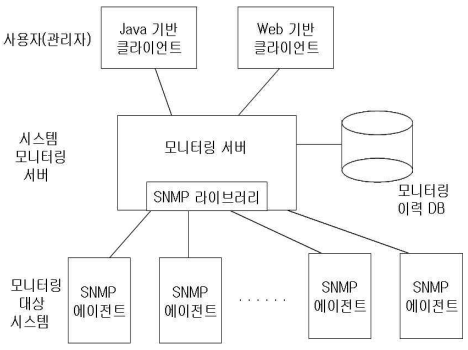

그림 4 모니터링 시스템 구성도

그림 4는 시스템의 전체적인 구성도를 보여준다. 모니터링의 대상이 되는 에이전트의 상태 정보를 서버가 라이브러리를 이용하여 직접 수집하며 해당 데이터는 적절히 가공되어 DB와 클라이언트에게 전달된다. 클라이언트는 서버로부터 받은 정보를 이용하여 사용자에게 에이전트의 상태를 보여주는 역할을 한다.

모니터링 시스템은 전체적으로 3계층으로 구성되어 있으며 모니터링 대상 시스템은 개인 컴퓨터나 네트워크 장비 등이 그 것에 속하고, SNMP 에이전트 프로그램을 실행 시켜야 한다. 시스템 모니터링 서버는 네트워크상에 한 대만 존재하며, 클라이언트 시스템과 에이전트 시스템사이에서 데이터를 수집 및 가공하고 전달한다. 또한 DB를 이용하여 데이터를 저장하는 역할도 수행하게 된다. 관리자가 직접 사용하게 되는 프로그램인 클라이언트 프로그램은 보다 명확하고, 자세하게 에이전트의 상황을 사용자에게 보여주는 역할을 하며, 경보 및 통신 수단을 제공하기도 한다. 또한 DB의 내용을 사용자가 사용할 수 있게 하는 수단도 제공한다.

서버 프로그램의 구현은 linux 환경에서 C언어로 제작하였으며, linux에서 구동된다. DB는 mysql을 사용한다. java 기반 클라이언트 프로그램은 java2의 swing을 이용하여 작성하였고, 웹 기반 클라이언트 프로그램은 JSP를 이용하여 작성되었으며, 클라이언트를 이용하게 해주는 웹 서버로는 httpd-2.0.50 tomcat 5.0을 설치하여 사용하였다.

3.2 SNMP 라이브러리의 구현

모니터링 시스템을 위하여 제작한 라이브러리는 기본적인 SNMP의 기능인 get메시지 구현을 위해 snmpGet(), snmpGetMulti(), snmpGetNext(), snmpGetInstance() 함수를 제공한다. 각각의 함수는 순서대로 하나의 OID에 대한 정보 요청, 여러 개의 OID에 대한 정보 요청, 요청 OID의 다음 OID의 정보를 가져오는 요청, MIB 트리의 자손에 대한 모든 OID에 대한 정보 요청을 하는 기능을 제공한다. 그리고 set 메시지의 기능을 구현하기위해 snmpSet()함수가 제공된다. 각 함수들은 매개변수로 에이전트의 ip주소, 요청 OID 그리고 반환되는 값을 위한 변수가 기본적으로 들어가며, 그 외에 추가 기능을 위해 몇몇 매개변수가 같이 들어간다. 각 함수는 에이전트의 가동 시간을 알아오는 요청을 동시해 함으로 시간 데이터가 에이전트의 OID값의 상태에 대한 시계역할을 해준다. 이러한 시계는 에이전트의 특정 OID 값에 대한 델타 값을 계산하게 하여준다.

get과 set이외에 트랩 메시지를 송신하기 위한 snmpTrapSend()와 agent로부터 트랩 메시지를 수신하기 위한 snmpTrapPoll() 함수가 제공 된다. 라이브러리는 매니저의 역할을 하면서도 다른 매니저에게 트랩 메시지를 전달할 수 있는 기능을 제공하여 준다.

SNMP의 기본적인 기능을 위한 함수 이외에도 응용 프로그래머의 편리를 위한 몇 몇 함수가 추가로 제공되는데 이는 라이브러리에서 에이전트에게 메시지를 보낼 때 Timeout을 설정하기 위한 snmpConfigTimeout(), 재송신 횟수를 설정하기 위한 snmpConfigRetry(), SNMP를 지원하는 agent를 찾기 위한 snmpAgentSearch(), 숫자로 구성된 OID를 문자열로 반환하는 snmpOid2Name(), 이름으로부터 OID를 반환해주는 snmpName2OID()를 제공한다.

3.3 모니터링 서버의 구현

서버 프로그램은 자체 제작한 SNMP 라이브러리를 이용하여 등록된 에이전트에게 주기적으로 SNMP 메시지를 전송하여 원하는 정보를 수집한다. 그리고 수집된 정보를 적절히 가공하여 각 클라이언트에게 전달한다. 이때 에이전트는 관리가 되는 대상이고, 클라이언트는 관리자가 사용하는 프로그램으로 서버가 수집한 에이전트의 상태를 보여주는 역할을 하는 프로그램이다. 서버 프로그램이 시스템 관리를 위하여 에이전트로부터 수집하는 정보는 네트워크 트래픽 사용률, 트래픽중의 에러율, cpu 사용률, 메모리 사용률이다. 이외에도 에이전트의 기본적인 정보(ip, 이름, 에이전트 사용자 연락처, 시스템 위치)를 얻어온다.

에이전트로부터 수집된 정보는 적절히 가공되어 클라이언트 프로그램에게 실시간으로 전송하게 된다. 만일 이러한 정보가 적정 수준을 넘어선 경우에는 자체 모니터링 이력 DB에 저장 된다. 이 DB에는 에이전트의 기본정보도 저장을 하고 있다. DB의 사용은 에이전트의 위기 상황에 대한 이력 정보를 제공해 주며 후에 통계자료로도 사용할 수 있다.

네트워크 트래픽의 사용률과 에러율은 일반 MIB의 객체가 아니므로 서버 자체적으로 해당 사용률을 계산하게 된다. 식 (1)은 네트워크 사용률을 계산하는 식을 나타내는데 우선, 각 에이전트가 네트워크상에서 자신이 받는 패킷의 개수를 라이브러리의 snmpGet 함수를 이용하여 얻어온다. 그리고 해당 패킷의 개수에 MTU (Maximum Transmission Unit)를 곱한다. 결국 각 패킷이 최대의 패킷 크기로 들어온다고 간주하고, 에이전트에 들어온 총 패킷의 양에 대한 네트워크 점유정도를 계산하는 것이다. 이는 최악의 경우를 예상하는 것으로 네트워크의 사용률에 대한 과부하를 감지한다.

$$\text{사용률} = \frac{packet\text{의개수} * MTU}{BandWidth} * 100 \quad (1)$$

네트워크에 대한 에러율 역시 식 (1)의 수식을 이용하여 계산하는데 이는 네트워크 사용률과 같은 방법으로 계산하지만 packet의 개수에 에러 패킷만을 걸러내어 대입해서 계산한다. 에러 패킷만 뽑아내는 방법은 식(2)와 같으며 수신된 패킷 중에서 TCP나 UDP, ICMP 계층까지 올라온 패킷을 정상적인 패킷으로 간주하고, 전체 수신패킷에서 해당 패킷(정상)의 개수를 뺀다. 그리고 남은 패킷의 수를 네트워크 사용률식의 packet의 개수에 대입하여 에러율을 계산한다.

$$packet\text{의개수} = \text{총}packet\text{수} - (TCP + UDP + ICMP) \quad (2)$$

서버는 에이전트와 통신을 할 뿐만 아니라, DB 관리도 하며, 클라이언트에게 실시간으로 정보를 넘겨주는 역할을 하게 된다. 즉 서버 프로그램은 모든 에이전트에 대해 SNMP로 관리를 하는 주체가 되며, 관리를 위해서는 반드시 필요한 존재이며 그 중심에 있는 프로그램이다. 또한 DB를 이용함으로써 보다 많은 정보를 관리자에게 제공해줄 수 있다. 서버 프로그램의 주된 역할은 에이전트와 클라이언트 사이에서 라이브러리를 이용하여 정보를 수집하고, 가공하여 클라이언트로 넘겨주는 것이다.

3.4 모니터링 대상 시스템

모니터링 대상 시스템은 관리의 대상이 되는 시스템으로서 개인 컴퓨터, 네트워크 장비 등이 포함된다. 각 대상 시스템은 SNMP 에이전트 프로세스가 필요하게 되는데 이는 관리자 시스템이 SNMP 요청 메시지를 보낼 때 이에 적절히 응답하기 위한 프로세스이다. 해당 에이전트 프로세스는 시스템이 만들어진 제조회사나, OS의 종류에 따라 다르게 된다. 현재 구현된 관리 프로그램은 windows와 linux 컴퓨터를 모니터링 대상으로 하고 있다. windows의 경우 SNMP 서비스를 통해 에이전트 프로세스가 실행되고 linux의 경우 net-snmp를 가동함으로서 에이전트 시스템의 역할을 수행하게 된다. 대부분의 네트워크 장비들은 기본적으로 SNMP 에이전트 수행 기능을 갖추고 있다.

기본적인 정보이외에 제조회사나 운영체제의 종류에 따라 특수한 MIB가 제공되기 때문에 에이전트의 시스템이나 운영체제를 아는 것은 중요하다. 하지만 공통적으로 제공되는 MIB의 객체를 요청할 경우에는 에이전트의 플랫폼에 상관없이 정보를 요구할 수 있다. 단지 에이전트에서는 SNMP 서비스를 제공해주기만 하면 된다. 결국 공통적으로 제공되는 MIB의 정보를 요청하는 서버프로그램의 경우는 특별한 플랫폼에 영향을 받지 않고 네트워크에 연결된 컴퓨터나 네트워크 장비들에게 정보를 요청할 수 가 있으며, 그들을 관리할 수 가 있는 것이다.

네트워크 관리뿐만 아니라 시스템 관리까지 하는 본 프로그램에서는 운영체제에 연관된 MIB를 사용하게 된다. 결국 관리의 대상이 되는 에이전트 시스템은 SNMP 서비스를 가동한 windows 컴퓨터와 net-snmp가 가동되는 linux 시스템이 에이전트 시스템으로서 사용 된다. 제한적으로 에이전트를 선정하는 것 같지만 서버 프로그램의 관리 대상(에이전트의 종류)에 따라 적절한 OID을 프로그램에게 제공한다면 위의 두 가지 시스템만을 관리하는 프로그램에서 다른 종류의 시스템 관리 프로그램으로의 변경이 쉽게 가능 할 것이다.

네트워크 장비를 위주로 기본적인 MIB만을 사용하여 네트워크 관리 전용 시스템을 구축할 경우, 보다 쉽게 에이전트의 확장이 가능하다. 기본 MIB만을 사용하는 서버를 만들고 SNMP를 지원하는 네트워크에 존재하는 여러 장비들(라우터, 스위치 등)에게 SNMP 정보를 요청만하면 각 장비들이 관리의 대상 시스템으로서 요청에 응답할 것이다.

4. 클라이언트 프로그램

4.1 클라이언트 프로그램의 역할

클라이언트 프로그램은 서버와 함께 관리자에게 사용되는 역할을 한다. 서버 프로그램은 에이전트의 정보를 수집하고, 가공하며, 저장하는 역할을 하며 클라이언트 프로그램은 서버와 다른 위치에서 서버와 소켓 통신을 하여, 서버로부터 받은 정보를 관리자에게 보여주는 역할을 하는 것이다. 관리자는 클라이언트가 화면에 보여주는 정보를 살펴봄으로써 에이전트 시스템의 상태를 파악하고, 에이전트 시스템이 처한 상황에 대해 적절한 조치를 취할 수가 있는 것이다.

4.2 java 기반 클라이언트

java 언어로 작성된 클라이언트 프로그램은 서버로부터 에이전트의 상황을 전달 받으며, 해당 상황을 프로그램이 GUI(Graphic User Interface)환경으로 관리자에게 알려준다.

java 기반 클라이언트 프로그램은 모든 에이전트의 cpu, 메모리, 네트워크의 상황을 한 눈에 제공하는 오버뷰 기능을 막대그래프로서 제공 한다. 각 에이전트 마다 3개의 막대그래프가 존재하게 된다. 세부 메뉴(네트워크, cpu, 메모리)로 들어가면, 모든 에이전트의 각 항목에 대한 구체적 상황을 꺽은 선 그래프로 표현하여 준다. 꺽은 선 그래

프는 시간이 지남에 따라 해당 수치를 움직이며 보여준다. 그래프의 눈금은 백분율로 제공하게 되는데 네트워크의 사용률의 경우 해당 백분율의 격차가 커 네트워크 상황에 따라 최대값의 값을 조절한다. 각 상황에 맞게 최대값은 0.1, 1, 10, 100으로 구분되어 있다. 그래프는 네트워크의 사용률의 수치에 맞게 자동적으로 그래프의 최상위 값을 선정하고 그 사이에 알맞은 크기로 그래프를 그려나간다.

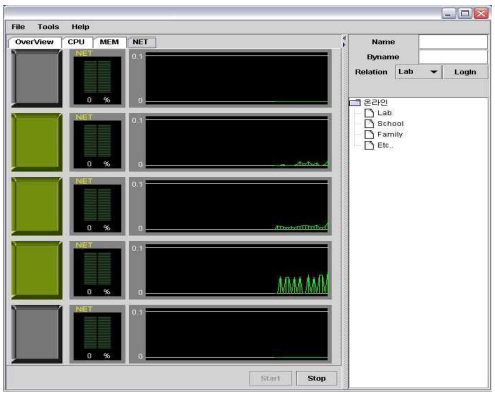

그림 5 java 기반 클라이언트의 실행화면

그림 5는 각 에이전트의 네트워크 상황을 보여주는 화면이다. 각줄은 하나의 에이전트를 나타내며, 맨 앞의 사각형에는 에이전트의 상태를 색으로 표현해준다. 또한 이 사각형은 각각의 에이전트를 나타내며 클릭 시 에이전트의 상세 정보를 화면에 보여준다.

이 상세 정보에는 에이전트의 이름, 위치, 설명, 소유자 연락처 등이 적혀있으며, 이러한 정보를 이용해 에이전트 소유자에게 시스템 관리자가 e-mail 및 간단한 메시지를 전송할 수 있다. 가운데 프레임에는 네트워크 상황을 막대그래프와 꺾은 선 그래프로 보다 상세한 수치 값을 표현해준다. 이 수치 값은 서버로부터 받은 값이며, 에이전트의 실시간 네트워크 정보이다. 만일 서버로 부터 받은 실시간 정보가 특정 한계치를 넘으면 맨 앞의 블록의 색을 변경함으로서 현 상황을 쉽게 알 수 있게 해준다.

에이전트 블록의 윗부분에는 오버뷰, cpu, 메모리 등 다른 화면으로의 전환 탭이 있다. 각 탭을 이용하여 쉽게 다른 상태를 나타내는 창으로 전환이 가능하며 에이전트의 상태를 알 수 있다. 또한 오른쪽에는 다른 관리자들의 로그인 여부를 살펴보고, 서로 메지시를 주고받을 수 있는 창이 있다. 이는 간단한 메신저를 구현한 것으로 보다 빠르게 관리자들끼리 연락을 취하고 적절히 대처하기 위함이다.

4.3 웹 기반 클라이언트 프로그램

웹 기반 클라이언트 프로그램은 시스템 관리자가 어느 곳에 있어도 웹을 통해 시스템의 상황을 살펴 볼 수 있게 한다. java 기반 클라이언트보다 많은 기능을 제공하지는 않지만 간단하게 cpu, 메모리, 네트워크 사용률을 관찰 할 수 있다.

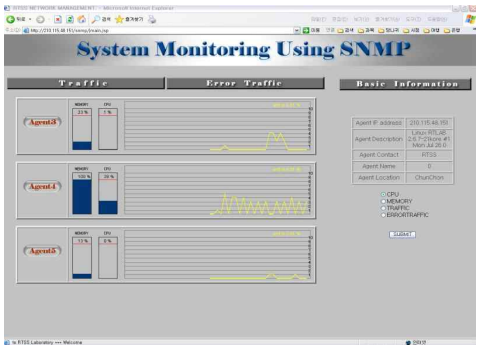

그림 6 웹 기반 클라이언트의 실행 화면

그림 6에서 볼 수 있듯이 웹 기반 클라이언트 프로그램의 에이전트 상태 표시 방법은 두 개의 막대그래프와 하나의 꺾은 선 그래프로 구성되어 있다. 앞의 두 개의 막대그래프는 cpu, 메모리 사용률을 보여주는 것이다. 꺾은 선 그래프는 네트워크 사용률을 나타내지만 상단 부분의 버튼으로 네트워크 에러율도 볼 수 있다. 즉, cpu와 메모리사용률은 기본적으로 항상 화면에 나타나며 상단의 버튼으로 에러율과 사용률에 대한 꺾은 선 그래프를 다르게 보는 것이다. 네트워크에 대한 정보는 자바 기반 클라이언트와 마찬가지로, 그 격차가 커 최대값을 변경하면서 해당 데이터의 양을 보여주게 되는데, 웹 버전에서는 그래프 상단에 0.01, 0.1, 1, 10, 100의 숫자를 표시하여 현재 그래프 최대값을 보여주게 된다. 만일 특정 데이터가 정해놓은 한계를 넘어설 경우 브라우저 창 위로 에이전트의 번호와 어떠한 데이터 값이 넘었는지에 대한 경고창이 뜬다.

좌측 프레임에 있는 각 에이전트마다 주어진 버튼을 누르면 해당 에이전트의 상세 정보를 오른쪽 프레임에서 볼 수 있다. 이는 java 기반 프로그램과 같은 내용을 담고 있다. 이러한 상세 정보 밑에는 각 에이전트마다의 에러 리포트를 볼 수 있는 버튼이 있다. 이 버튼은 관리하는 4개 항목에 대한 (cpu, 메모리, 네트워크, 에러) 리포트를 볼 수 있는 페이지를 보여준다. 해당 리포트는 서버 프로그램이 저장해놓은 이력 DB에서 추출해 오는 것으로서 각각의 항목에 대해 일별, 월별로 해당 항목이 기준치를 정확이 얼마를 넘었는지, 언제 넘었는

지를 볼 수 있는 페이지이다. 이러한 DB데이터를 적절히 이용하여 통계자료로 사용하거나, 관리자가 자리를 비운 시간에 각 에이전트에 대한 상태 조사 등을 위해 이용된다.

6. 결론

시스템 모니터링을 위한 본 프로그램은 소규모 네트워크나 PC방, 학교, 사무실 등에서 여러 대의 컴퓨터 시스템을 관리하기에 적합하다. 각각의 사용률로서 관리대상이 되는 에이전트 시스템의 문제를 인식할 수 있으며, 그에 대한 대처를 할 수 있다. 네트워크 내에 존재하는 패킷의 양을 파악함으로써 각 에이전트가 얼마나 많은 데이터를 네트워크로부터 받는지 또는 전송하는지 알 수 있으며, 에러 패킷의 양을 살펴봄으로써 바이러스에 의한 공격이나 감염여부도 분간해 낼 수 있다. 이러한 관리용 데이터를 이용하여 해당 에이전트에 대한 빠른 상황판단 및 조치, 복구가 가능하게 된다.

시스템 리소스에 대한 부하가 적은 라이브러리와 서버 프로그램으로서 가볍게 매니저 서버를 구축 할 수 있으며, 모니터링 대상이 되는 에이전트 시스템에는 해당 벤더가 제공하는 SNMP 에이전트를 설치함으로서 모니터링을 위한 시스템 설정이 끝난다. 관리자는 클라이언트를 통해 한 네트워크 내의 시스템들을 모니터링 할 수 있으며, 에러 리포트 등을 참고하여 잠정적으로 문제가 있는 시스템도 식별 할 수가 있다. 이는 개인 컴퓨터뿐만 아니라 서버 급 컴퓨터에도 쉽게 적용할 수 있어 일반 네트워크 어디서나 모니터링 시스템 구축이 용이하다.

SNMP 라이브러리를 이용하여 관리자가 원하는 정보만을 수집할 수 있고, 손쉽게 다른 종류의 관리 프로그램으로도 사용이 가능하다. 자신의 구미에 맞는 관리 프로그램을 쉽게 구현 할 수 있는 것이다. 또한 MIB의 조정으로 컴퓨터 시스템뿐만 아니라 네트워크 장비에 대한 관리도 가능하기 때문에 서버 프로그램의 약간의 수정만으로 라우터, 스위치 등도 관리할 수 있는 네트워크 관리 프로그램으로 쉽게 변환이 가능하다.

참 고 문 헌

[1] William Stallings, "Data and Computer Communications," *Prentice Hall*, 1999.

[2] S. Waldbusser. "Remote Network Monitoring Management Information Base", *IETF*, RFC1757, February. 1995.

[3] Douglas Mauro, Kevin Schmidt "Essential SNMP", O'REILLY, 2001

[4] M, Rose, K McCloghrie, "Structure and Identification of Management Information forTCP/IP-based Internets", *IETF*, RFC1155, May 1990.

[5] William Staling, "SNMP, SNMPv2, SNMPv3, and RMON1 and 2, 3rd ed", Addison-wesley, 1999

[6] Ostell,J. GenInfo "ASN.1 Syntax: Sequences. NCBI Technical Report Series, National Library of Medicine, NIH.", *Technical Report Number 1*, p. 37, 1990

[7] Mark A. Miller, "Managing Internet works with SNMP", *IDG*, 3ed, 1997