제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

SNMP 기반 EMS 설계 및 구현

강석민, 허재두
한국전자통신연구원
e-mail : {esemkang, jdhuh}@etri.re.kr

Design and Implement of SNMP-based EMS

Seok-Min Kang, Jae-Doo Huh
Electronics and Telecommunications Research Institute

요 약

네트워크의 규모가 급격히 커지면서 네트워크에 참여하는 장비들도 다양해졌다. SNMP 는 이러한 복잡한 네트워크와 장비들을 효율적으로 유지 및 관리하기 위한 대표적인 프로토콜이다.

본 논문에서는 리눅스에 SNMP 에이전트를 구현하고, 기존의 복잡한 명령어 체계를 개선한 CLI 를 이용하여 콘솔 기반의 관리 효율을 높이고, 플랫폼 독립적인 자바를 이용한 GUI 와 웹 브라우저 기반의 관리 내용에 대해 기술한다. 본 논문에서 구현한 에이전트는 매니저의 요청에 적절한 행동을 취하고 결과를 응답해주며 시스템의 오류와 같은 긴급 상황에 대해 매니저에게 TRAP 을 발생시키는 기능을 갖는다.

1. 서론

인터넷의 급속한 확산 및 팽창으로 인해 네트워크에 참여하는 장비들이 다양해졌다. 이처럼 복잡한 네트워크 내에 다양한 장비들이 설치 및 운용되는 상황에서 그것들을 효율적으로 유지, 관리 하기 위한 기능이 필요하게 되었다. SNMP (Simple Network Management Protocol)는 이러한 네트워크 관리 기능을 제공하기 위한 대표적인 프로토콜로써, 구조가 단순하고 구현이 쉽기 때문에 네트워크 관리 표준으로 자리 잡고 있다. SNMP 를 이용하여 관리자는 보다 효율적으로 네트워크를 관리하고자 한다. 이를 위해 콘솔을 통한 관리 또는 웹을 통한 관리의 필요성이 대두된다. 즉, 보다 효율적인 네트워크 관리를 위해 간단한 콘솔 기반의 원격 접속 관리와 플랫폼 독립적인 자바를 이용한 GUI (Graphic User Interface) 형태의 관리가 가능한 EMS (Element Management Station)의 개발이 필요하게 된다.

그러므로 본 연구에서는 SNMP 를 이용한 네트워크 관리 시스템을 구현하였는데, 개발의 편의를 위해 에이전트는 리눅스 기반으로 구현을 하고, SNMP 에 적합한 shell 을 구현하였다. 또한 매니저는 CLI (Command Line Interface)를 이용하여, 원격 접속을 통한 콘솔 관리가 가능하고, Java 를 이용한 GUI 형태의 관리도 가능한 EMS 시스템으로 구현하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 SNMP 의 기본 용어와 구조 및 동작을 소개한다. 3 장에서는 콘솔 및 GUI 관리가 가능한 SNMP 에이전트/매니저의 구현에 관한 상세한 사항을 기술한다. 마지막으로 4 장에서는 결론 및 향후 연구과제에 대하여 설명한다.

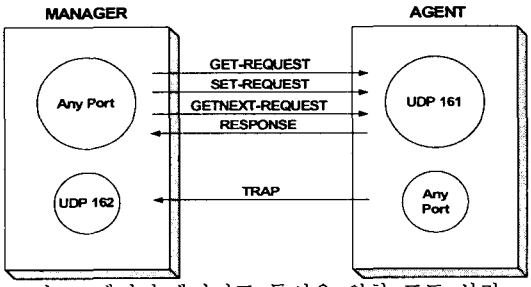
2. SNMP 의 기본 구조 및 동작

SNMP 는 관리자가 위치하는 매니저인 관리국 (Management Station)과 매니저의 요청에 반응하여 응답하는 에이전트(Managed Agent), 네트워크 관리 대상에 대한 개념적 구조체인 MIB (Management Information Base), 그리고 매니저와 에이전트 사이의 통신을 위한 프로토콜인 SNMP 의 네 가지 요소를 가지고 있다. SNMP 는 크게 버전 1, 2, 3 로 구분되는데 버전 1 은 GET, SET, TRAP 의 세가지 동작으로 이루어지며 보안에 대한 문제를 다루지 않고, 대역폭을 낭비한다는 단점이 있다. 이를 개선한 버전 2 는 송신 시 DES (Data Encryption Standard) 알고리즘을 사용하고, 사용자 인증 시 MD5 (Message Digest 5) 알고리즘을 사용하여 보안에 대해 강화를 하였다. 또한 대역폭 낭비를 막기 위해 GetBulkRequest 동작을 추가했다. 버전 3 는 버전 2 에 사용자 인증, 보안, 접근 제어 등의 기능을 추가하였다.

2.1 SNMP 의 동작 방식

SNMP 의 주된 동작은 GET-REQUEST, SET-REQUEST, TRAP 의 세가지로 나눌 수 있다.

매니저는 에이전트에게 관리에 필요한 대상에 대한 요청을 하게 되고, 이에 에이전트는 모듈에 정의된 절차를 걸친 후 그 결과를 매니저에게 응답하게 된다. 이때 사용되는 동작이 GET-REQUEST 이고 에이전트는 이 요구에 대해 응답을 보낸다. SET-REQUEST 는 매니저가 에이전트의 설정을 바꾸기 위해 보내는 요청이다. 마지막으로 TRAP 은 위의 두 가지와는 달리 에이전트가 매니저에게 보내는 메시지로, 관리자가 미리 설정해 놓은 정보에 따라 시스템 등에 문제가 발생하는 경우 관리자에게 보고하는 메시지이다. 이러한 매니저와 에이전트간의 요청/응답은 특정한 UDP (User Datagram Protocol) 포트를 통해서 이루어지는데, 이에 대한 내용은 (그림 1)과 같다.



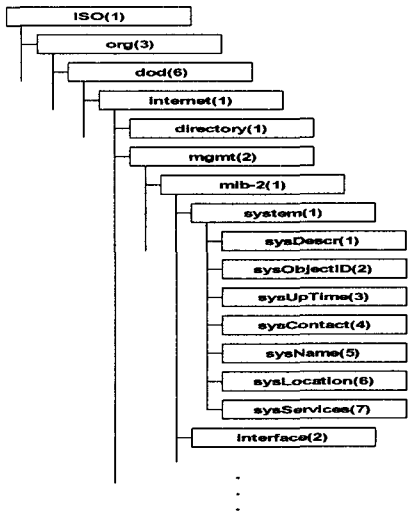
(그림 1) 매니저-에이전트 통신을 위한 포트 설정

2.2 MIB (Management Information Base)

매니저와 에이전트 사이에 특정한 정보를 주고 받음으로써 네트워크 관리가 이루어진다. 이때 관리되어야 할 특정한 정보 또는 자원을 객체라 한다. 이러한 객체들을 트리 구조로 모아놓은 집합체를 MIB 라 한다. MIB 를 정의하고 구성하는 프레임워크인 SMI(Structure of Management Information)는 다음과 같은 사항을 정의하고 있다.

- ✓ **MIB 의 각 객체들은 ISO 와 ITU-T 에 의해 표준화되고 개발된 언어인 ASN.1 을 사용해서 정의한다.**
- ✓ **정의할 모든 객체는 반드시 name, syntax, encoding 을 갖는다.**
- ✓ **ASN.1 에서 정의된 데이터 종류 중 SNMP 에서 사용할 수 있는 것과 허용되는 구조가 명시되어 있다.**

MIB 는 루트로부터 시작하는 트리 형식의 구조체인데 각각의 관리 대상은 하나 이상의 대상 객체로 구성되고 특정 객체 식별자(OID; Object IDentifier)에 의해 확인된다. 트리 형식의 구조에서 SNMP 는 최종 노드인 리프 만을 읽고 쓸 수 있다. 이때 트리의 깊이가 증가함에 따라 '.' 로 구분하게 된다. 예를 들어, 시스템의 설명을 나타내는 sysDescr 는 'iso. org. dod. internet. mgmt. mib-2.system. sysDescr' 으로 식별하는데, 실제 sysDescr 의 OID 는 '1.3.6.1.2.1.1.1' 로 표현되는 정수이다. 여기서 system 은 리프 노드가 아니기 때문에, 읽기/쓰기가 허용되지 않는다. sysDescr 가 리프이기 때문에 읽기/쓰기가 가능하다. (그림 2)는 IANA(Internet Assigned Number Authority)에서 관리하는 MIB 구조의 일부분이다.



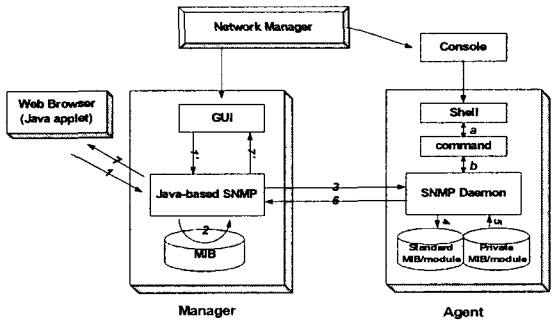
(그림 2) MIB 구조

3. SNMP 기반 EMS 설계 및 구현

SNMP 기반 EMS 의 구현 환경은 다음과 같다. 에이전트는 리눅스 커널 2.4.0 기반의 펜티엄 3 에서 구현이 되었고 ucd-snmp 2.4.1 과 CISCO 라우터를 위한 사설 MIB (Private MIB)를 참고하였다. 콘솔 기반의 관리를 위한 명령어 처리기인 쉘은 에이전트와 동일한 플랫폼에서 구현되었다. 또한 윈도우즈 환경에서의 망관리를 위해 자바 기반의 GUI 매니저를 구현하였는데, 이때 운영체제는 윈도우 2000 이 설치된 펜티엄 4 PC 이다. GUI 는 FOUNDARY Networks 사의 라우터를 위한 사용자 매뉴얼을 참고하였다.

3.1 매니저-에이전트 구성

본 연구에서 구현한 매니저와 에이전트간의 EMS 동작 과정은 (그림 3)과 같다.

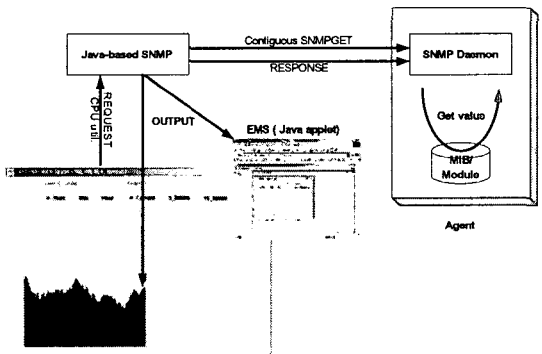


(그림 3) 매니저-에이전트간 EMS 동작 과정

(그림 3)에서 보는 바와 같이 매니저는 GUI 또는 원격 접속 가능한 콘솔을 통하여 명령어를 에이전트

에게 전송해 필요한 정보를 요청한다. 콘솔은 에이전트로 원격 접속이 가능하여야 한다. MIB 의 복잡한 구조 때문에 콘솔상에서 입력할 명령어도 매우 복잡할 수 있는데 본 논문에서는 CISCO ISO 12.0 의 표준에 의한 CLI 를 통해 복잡한 명령어들을 간단한 형태로 표현 가능하도록 구현하였다. 에이전트의 쉘은 CLI 를 통한 명령어를 해석하여 SNMP 데몬에게 적합한 형태로 전달(*a, b*)하는 역할을 한다.

기존의 콘솔 기반 관리는 관리자가 일일이 명령어를 입력해야 하는 불편함이 있는데 이를 개선하여 Java 를 이용한 GUI 를 구현하여 보다 효율적인 망 관리가 가능하게 된다. 관리자는 GUI(1')나 웹 브라우저(1)를 통해 매니저의 자바 기반 SNMP 에게 정보를 요청한다. 자바 기반 SNMP 는 MIB 를 참조하여(2) 에이전트에게 요청을 보내고(3), 에이전트는 MIB 나 MIB 모듈에 구현된 절차에 따라(4) 시스템 내에서 결과를 가져와(5) 다시 매니저의 자바 기반 SNMP 에 전달한다(6). 결과를 받은 자바 기반 SNMP 는 GUI(7')나 웹 브라우저(7)에게 결과를 보내주고, 최종적으로 이 결과를 관리자의 화면에 출력해 준다. 실제로, 매니저가 에이전트의 CPU 사용률을 요청하여 GUI 또는 웹 브라우저로 보게 되는 절차가 (그림 4)에 나타나 있다.

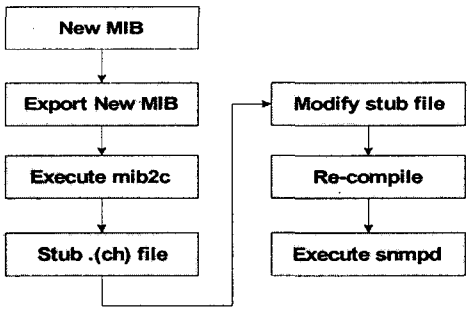


(그림 4) EMS 동작 과정 예 (CPU 사용률 출력)

3.2 에이전트

매니저의 요청에 대해 적절한 행동을 하고 결과를 취하여 보내주기 위해서 에이전트가 필요하다. 이런 일련의 과정에서 에이전트의 행동을 명확히 규정하기 위해 MIB 와 그 모듈이 필요한데, 이러한 MIB 는 표준 MIB 와 각 기업에서만 제공하는 사설 MIB 로 구분할 수 있다. 본 연구에서는 표준 MIB 를 포함하고, 라우터 환경에 적용하기 위한 사설 MIB 와 그 모듈을 리눅스 환경에서 정의하였다. 에이전트에서 사설 MIB 와 그 모듈을 정의하는 절차는 (그림 5)와 같다.

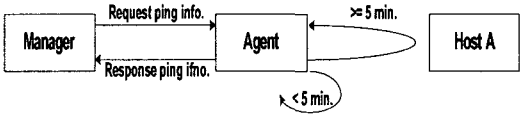
새로운 MIB 를 정의하고 문법을 검사한 후, 기존의 MIB 에 추가한다. 그리고 MIB 에 맞는 스터브 파일을 생성해 주는 mib2c 펄스크립트를 실행하여 스터브 파일을 얻는다. 매니저의 요청에 적합한 행동을 취하는 모듈을 스터브 파일에 추가 구현한 후, 재 컴파일 하여 SNMP 데몬을 실행해 주면 새로 추가한 MIB 과 모듈이 적용되게 된다.



(그림 5) 새로운 MIB 및 모듈 추가 절차

3.2.1 ping-MIB

본 연구에서 구현한 사설 MIB 중 ping-MIB 는 다음과 같다. 기본적으로 에이전트는 어떤 호스트와의 ICMP (Internet Control Message Protocol) 정보를 미리 파일로 저장하고 있다. 이때, 매니저는 에이전트에게 ICMP 메시지를 보내 네트워크의 상태에 대한 정보를 요청하게 되는데 (GET-REQUEST) 에이전트가 파일로 유지하는 ICMP 관련 정보가 최근 5 분 이내의 정보이면, 직접 호스트에 ICMP 메시지를 보내지 않고 파일의 정보를 매니저에게 보내준다. 만일 파일의 정보가 보관된 지 5 분을 초과하면 이 정보가 유효하지 않다고 판단하여 호스트에 ICMP 메시지를 보내 정보를 수집한 후 파일에 저장하고 그 결과를 매니저에게 전송한다. 이렇게 함으로써 네트워크의 부하를 줄이고 가능한 최신의 ICMP 정보를 유지할 수 있다. 이러한 동작을 (그림 6)에서 볼 수 있다.

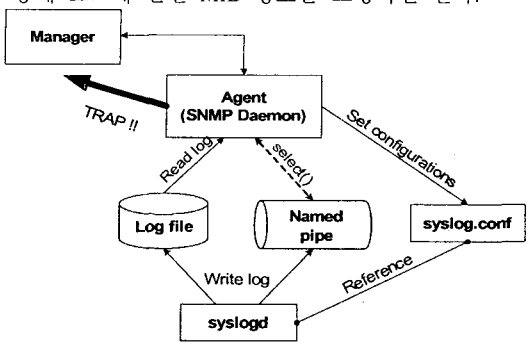


(그림 6) ping-MIB 의 동작 과정

3.2.2 syslog MIB

에이전트 시스템이 문제가 발생했을 때, 이를 매니저에게 알려주어야 한다. 이때 사용되는 SNMP 동작이 바로 TRAP 이다. Syslog MIB 는 이 TRAP 동작을 이용하여 에이전트에서 발생한 각종 이벤트를 매니저에게 전달할 수 있다. (그림 7)에 이러한 syslog MIB 의 동작 과정이 있다. 매니저는 에이전트의 시스템 로그의 enable/disable, 로그 파일의 위치, 로그를 저장할 위험 수준 등을 설정 할 수 있다. 에이전트의 syslog 데몬은 이러한 설정에 따라 로그 파일에 시스템의 로그를 기록/관리하게 되는데 동시에 네임드 파이프(named pipe)에도 로그를 기록한다. 에이전트의 SNMP 데몬은 이 네임드 파이프를 select() 함수를 이용하여 검사하고, 만약 설정한 수준 이상의 로그가 발생 했을 경우 매니저에게 TRAP 을 이용하여 이벤트 발생을 알린다. 매니저는 TRAP 메시지를 확인하여, 보낸 에이전트와 시스템 로그라는 정보를 알아내고 에이전트에게 EMS

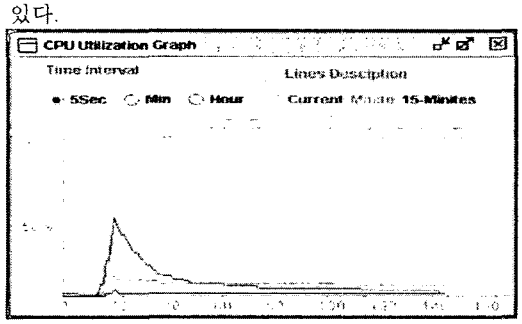
를 통해 로그에 관한 MIB 정보를 요청하면 된다.



(그림 7) syslog MIB 의 동작 과정

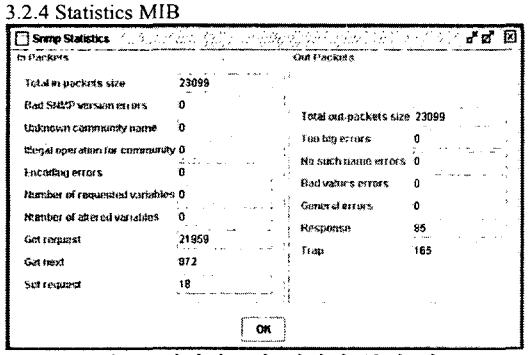
3.2.3 CPU MIB

(그림 8)은 에이전트 시스템의 CPU 의 사용률을 GUI 를 통해 그래프 형식으로 보여주고 있다. 복잡한 구조를 갖는 MIB 내의 CPU 정보를 갖는 OID 를, GUI 를 통하여 식별하기 쉽게 구현하였다. CPU MIB 가 가진 정보를, 연속적인 OID 의 요청을 통해 그래프로 나타내고 시스템 모니터링을 할 수 있으며 이미 설정된 값 이상의 사용률을 기록하면 TRAP 을 발생하도록 할 수 있다. 또한 요청 시간 간격을 임의로 조절하여 네트워크의 부하를 줄일 수 있다. 이와 비슷한 동작을 하는 예로 메모리 사용률, 메시지 통계 정보 등을 들 수 있다.



(그림 8) CPU 사용률 모니터링

3.2.4 Statistics MIB



(그림 9) 에이전트의 메시지 통계 정보

에이전트는 SNMP 를 통해 메시지에 대한 통계 정보를 수집하고 관리한다. 이러한 통계 정보를 이용하면 향후 개발이나 발생 가능성이 있는 문제점을 유추/해결 하는데 도움이 된다. 들어오고 나가는 패킷에 대한 통계 정보는 (그림 9)와 같은 결과를 보여준다.

3.3 Command Line Interface

앞에서 살펴본 GUI 와 웹 브라우저 기반의 관리 방법 외에도 콘솔을 통한 에이전트 관리가 가능하다. 그러나 SNMP 의 복잡한 명령어 체계와 트리 형식의 MIB 전반에 대한 이해를 가져야 하는 불편함이 있다. 이에 본 연구에서는 CISCO ISO 12.0 을 기준으로 단순화한 명령어 SET 을 만들고, 이를 파싱-해석-수행 할 수 있는 shell 을 구현하고 에이전트에 탑재하여 콘솔 기반 관리의 편의성을 도모하였다. 구현한 주요 명령어의 SET 은 <표 1> 과 같다.

<표 1> 명령어 SET

	명령어		명령어
1	show snmp	8	snmp-server engineID
2	show snmp engineID	9	snmp-server group
3	show snmp group	10	snmp-server view
4	show snmp user	11	snmp-server location
5	snmp-server chassis-id	12	snmp-server packetsize
6	snmp-server community	13	snmp-server user
7	Snmp-server contact	14	no snmp-server

4. 결론 및 향후 연구 과제

본 논문에서는 SNMP 를 이용하여 네트워크 관리가 가능한 매니저와 에이전트를 설계 및 구현하였다. 이 EMS 는 플랫폼 독립적인 자바를 이용한 GUI 와 웹 브라우저를 이용하므로 관리자에게 관리의 효율성을 높여주며 다른 OS 로의 쉬운 확장을 제공한다. 또한, 콘솔 기반의 관리도 지원하여 간단한 접근 및 관리의 용이함도 제공한다.

향후 연구 과제로는, GUI 와 웹 브라우저를 통하여 보다 효율적인 관리가 가능하도록 에이전트의 환경에 맞는 사설 MIB 와 모듈의 추가 구현, 그리고 리눅스 외의 다른 OS 로의 포팅에 대한 연구가 필요하다.

참고문헌

[1] W.Stallings, "SNMP and SNMPv2: the infrastructure for network management", IEEE Communications Magazine, Vol:36 Issue:3, pp.37-43, March 1998.

[2] M. J. Wren and J. A. Gutierrez, "Agent and Web-based Technologies in Network Management", GLOBCOM '99 Global Telecommunications Conference, Vol:3, pp.1877-1881, December 1999.

[3] RFC1157, "Simple Network Management Protocol (SNMP)", May 1990.

[4] RFC1213, "Management Information Base(MIB-II)", March 1991.

[5] AdventNet, http://www.adventnet.com/products/index.html#nms, August 2002.

[6] CISCO Systems, http://www.cisco.com/warp/public/477/SNMP/index.html, August 2002.