공개키를 이용한 SNMPv3 보안 모듈 설계 및 구현

한 지 훈† · 박 경 배†† · 곽 승 욱† · 김 정 일† · 정 근 원†
송 인 근††† · 이 광 배†††† · 김 현 욱††††

요 약

TCP/IP 기반하의 네트워크 사용으로 많은 사용자들은 서로 정보를 공유하고 자원을 효율적으로 이용할 수 있게 되었다. 그러나 복잡해진 망 구조를 효율적으로 관리할 수 있는 프로토콜이 필요하게 되었다. 1989년 분산된 네트워크의 망 관리를 위하여 SNMP(Simple Network Management Protocol)가 표준으로 채택되었고 그 후 보안 기능이 추가된 SNMPv2에서 제공하는 암호화 방식은 대칭형 암호화 방식인 DES(Data Encryption Standard)와 인증(Authentication)을 위한 MD5(Message Digest 5) 해쉬 함수이다. 그러나 DES는 키 길이의 취약성과 암호화 및 인증 알고리즘이 분리되어 수행되는 단점을 가진다. 이를 해결할 수 있는 방안으로 본 논문의 보안모듈에서는 RSA 공개키 방식을 사용한다.

본 논문은 SNMP와 관련된 사항들에 대해 고찰한 후, 표준 SNMPv3에서 제안하는 암호화 알고리즘 DES와 인증을 위한 MD5방식과 더불어 암호화와 인증을 동시에 수행할 수 있는 공개키 방식인 RSA를 적용함으로써 보안성을 강화하였다. 제안한 SNMPv3의 보안 모듈은 JAVA 언어로 구현하였으며 Windows NT 환경에서 실험되고 분석되었다.

SNMPv3 Security Module Design and Implementation Using Public Key

Ji-Hoon Han† · Gyong-Bae Park†† · Seung-Uk Kwak† · Jeong-Il Kim† · Keun-Won Jeong†
In-Keun Song††† · Kwang-Bae Lee†††† · Hyen-Uk Kim††††

ABSTRACT

Uses can share information and use resources effectively by using TCP/IP-based networks. So, a protocol to manage complex networks effectively is needed. For the management of the distributed networks, the SNMP(Simple Network Management Protocol) has been adopted as an international standard in 1989, and the SNMPv2 in which a security function was added was published in 1993. There are two encryption schemes in SNMPv2, the one is a DES using symmetric encryption scheme and the other is a MD5(Message Digest5) hash function for authentication. But the DES has demerits that a key length is a few short and the encryption and the authentication is executed respectively. In order to solve these problems, we use a RSA cryptography in this paper.

In this paper, we examine the items related with SNMP. In addition to DES and MD5 proposed in SNMPv3, we enhance security functionality by adopting RSA, a public key algorithm executing the encryption and the authentication simultaneously. The proposed SNMPv3 security module is written in JAVA under Windows NT enviroment.

※ 본 연구는 학술진흥재단의 연구비지원으로 수행된 것임.
† 준 회 원 : 명지대학교 대학원 전자공학과
†† 준 회 원 : 여주대학 전산정보처리학과 교수
††† 정 회 원 : 우송대학교 전자공학과 교수
†††† 정 회 원 : 명지대학교 전자공학과 교수
논문접수 : 1998년 8월 28일, 심사완료 : 1998년 11월 13일

1. 서 론

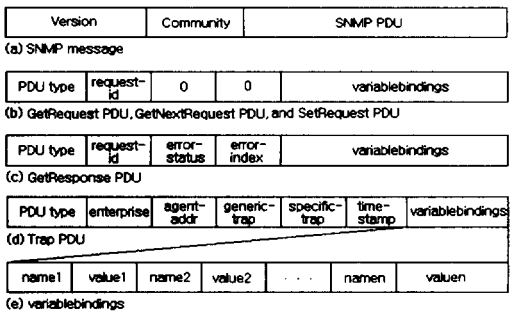
최근 네트워크, 컴퓨터 및 통신 구성 장비와 같은 정보 기술의 급격한 발달은 개인 컴퓨터의 한정된 사용에서 자원의 공유를 통한 원거리상의 유무선 통신 선로를 통해 서로에게 필요한 정보를 주고받고 자신의 컴퓨터를 원격 접근(remote access)할 수도 있으며, 전화가 아닌 컴퓨터를 사용하여 통신을 하거나 화상 회의를 할 수 있게 되었다. 1970년 국방용으로 시작된 ARPANET은 소수의 대학, 연구실, 공공기관에서만 사용되다가 TCP/IP기반하의 인터넷으로 개방되면서 점차 분산 시스템으로 발전하였고, 이로 인해 많은 사용자들이 네트워크를 통해 서로 정보를 공유하고 사용할 수 있게 되었다. 점차적인 사용자의 증가와 기술의 발달에 따라 네트워크의 복잡성은 유지와 관리면에서 기존의 프로토콜만으로는 많은 어려움을 갖는다. 네트워크 관리 문제를 해결하기 위하여 ISO와 인터넷 위원회에서는 국제 표준 네트워크 표준 프로토콜인 개방형 시스템 상호접속(OSI) 모델에서 CMIP(Common Management Information Services and Protocols)를 인터넷 표준인 TCP/IP는 SNMP(Simple Network Management Protocol)를 채택하였다[1]. CMIP는 모든 네트워크 환경에 공통으로 적용될 포괄적인 것을 목표로 하는 대신에 SNMP는 인터넷 관리의 직접적인 해결을 위해 신속하게 설계되었으며, CMIP가 가진 복잡성 대신 단순성을 통하여 분산된 네트워크에서의 효율적인 망 관리를 위하여 SNMP가 표준으로 자리잡고 발전하게 되었다. 이후 SNMP가 내포한 여러 문제점을 해결하고 트래픽 분석과 모니터링의 목적으로 RMON(Remote Network Monitoring), RMON 2가 개발되었다. 보안의 중요성이 대두되면서 단순한 커뮤니티 이름(Community name)을 통한 인증 방식에서 프라이버시, 인증, 접근 제어라는 세 개의 보안관련 서비스를 추가한 SNMPv2가 표준화되었다. RFC1446, RFC1447에서는 인증을 위하여 비대칭형 키 방식인 RSA와 대칭형 키를 이용하는 DES를 사용한 암호화, 복호화 모듈을 정의하고 있으며, 인증을 위한 MD5 해쉬함수의 사용에 대하여 정의하고 있다[2]. SNMPv2에서 제공된 보안 프로토콜은 파티를 통해 구현이 되었으며 SNMPv3는 USM(User-Based Security Module)를 통한 구현을 정의한다[3, 4].

이에 본 연구에서는 대칭키 방식을 표준으로 하는 SNMPv3에 공개키 방식을 응용한 개선된 암호화 방식을 설계한다. 본 논문의 구성은 2장에서 기본적인 SNMP의 구조와 암호 알고리즘에 대한 연구를 고찰하고, 3장에서는 SNMPv3의 구조와 보안 알고리즘을 4장은 공개키 방식을 기존 시스템 모델에 적용한 시스템 설계를, 5장에서는 시뮬레이션 및 분석을, 6장은 결론 및 향후 연구 과제를 기술한다.

2. SNMP의 구조와 암호화 및 인증

2.1 SNMP의 구조

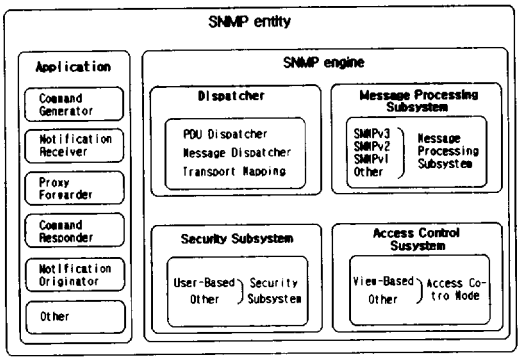
현재 많은 수의 연구소와 학교등의 비영리 기관에서는 SN MPv1을 기반으로 망 관리를 사용하고 있으며 H/W, S/W제조업체들은 보안성이 강한 SNMPv2를 기반으로 한 망 관리를 채택하고 있으며 현재 SNMPv3가 표준화 중이며 일부는 표준화가 되었다. SNMP의 기본 메시지 프로토콜은 스테이션(station)과 에이전트(agent)간의 MIB(Management Information Base)에 저장된 객체들을 액세스하고 값을 변경함으로써 망 관리를 한다. SNMPv1의 메시지 필드의 구성은 (그림 1)과 같이 SNMP 버전을 알리는 버전 number, 메시지 교환을 위해 사용되는 커뮤니티 네임, 5가지 PDU(Protocol Data Unit) 형식중의 한 형식으로서 구성되며 서로 다른 PDU들은 독립된 ASN.1 형식으로써 정의하며, 각각의 PDU들은 BER(Basic Encoding Rule)에 의해서 인코딩된다[5].



(그림 1) SNMP PDU 포맷
(Fig. 1) SNMP PDU format

기존 SNMPv1에 새로운 PDU형식과 MIB객체가 추가된 SNMPv2에서는 SNMP에 파티(party)모듈을 정의한다. SNMPv2 엔티티(entity)는 SNMPv2 파티를

포함하는 데이터 베이스를 가지고 있으며, SNMPv2에서의 정보 교환은 이러한 파티를 포함한 메시지 헤더와 데이터로 구성되며 보안과 관련된 정보와 프로토콜 데이터 단위의 많은 유형을 갖는다. USM기반의 SNMPv3의 전체적인 구조는 (그림 2)와 같다.



(그림 2) SNMPv3 구조
(Fig. 2) SNMPv3 structure

SNMPv3의 응용에 대한 메시지 처리는 고유의 기능을 갖는 여러 부시스템(subsystem)으로 구분되어지며 상호 작용한다. SNMP 엔티티내의 엔진은 다음과 같은 4개의 구성요소를 갖는다[6].

① 디스패처(Dispatcher)

모든 Snmp 엔진은 단 하나의 디스패처가 있으며, 다음과 같은 역할을 수행한다.

- 네트워크에 메시지를 송수신한다.
- SNMP메시지의 버전을 결정한다.
- 원격 SNMP 엔티티에 PDU를 전송하는 SNMP 응용에 abstract interface를 제공한다.

② 메시지 처리 부시스템(Message Processing Subsystem)

- 송수신 메시지에 대한 처리를 한다.

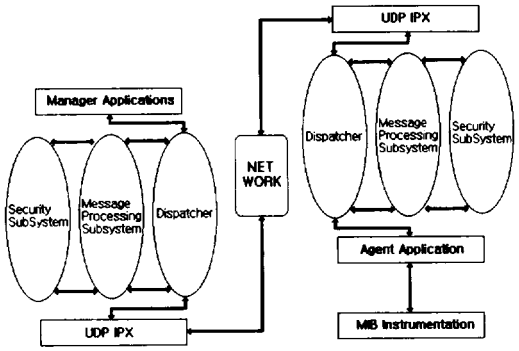
③ 보안 부시스템(Security Subsystem)

- 레벨에 따라 3개의 보안 기능을 제공한다.

④ 접근 제어 부시스템(Access Control Subsystem)

- 하나 또는 그 이상의 접근 제어 기능을 제공한다.

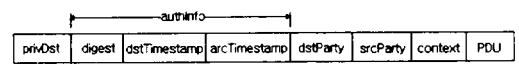
매니저와 에이전트간의 데이터 블록도는 (그림 3)과 같다.



(그림 3) SNMPv3 매니저와 에이전트 데이터 흐름도
(Fig. 3) The data flowchart between manager and agent in SNMPv3

2.2 SNMP의 암호화

각각의 에이전트들은 지역 MIB를 제어하고 많은 수의 관리 스테이션들에 의한 MIB 사용을 제어한다. 제어에는 MI B에 접근하는 것을 제한하는 인증 서비스와 서로 다른 관리 스테이션들에게 다른 접근 권한을 부여하는 접근 정책(policy), 상호 에이전트들이 프록시(proxy)로써 수행되는 프록시 서비스가 있다. 제어에 대한 3가지 측면은 보안과 관련이 있는데, SNMPv1은 보안에 대하여 단지 원시적이고 제한된 기능만을 제공한다. SNMP 커뮤니티란 인증, 접근 제어, 프록시 특징들을 정의하는 SNMP 에이전트와 SNMP 관리자들 사이의 관계를 나타내며 각 지역 시스템의 에이전트에 의해서 인증, 접근 제어, 그리고 프록시 특징 등을 조합한 하나의 커뮤니티를 생성한다. (그림 4)와 같이 SNM- Pv2는 헤더에 authInfo 필드를 갖는데 메시지의 보안이 없는 경우에 authInfo 필드는 길이 8비트의 스트링으로 구성되고 프라이버시가 제공될 때 privDst를 제외한 전체 PDU를 암호화한다. 이때 privDst필드는 수신자 SNMPv2 엔티티 수신자 파티를 결정하고 메시지 프라이버시 특성을 결정하도록 비암호화된 상태로 있어야 한다 .



(그림 4) SNMPv2 메시지 형식
(Fig. 4) Message format of SNMPv2

SNMPv3의 경우는 3가지 보안 레벨을 제공한다.

- 레벨 1 - 인증과 보안 모두 제공하지 않음 (without authentication and without privacy)
- 레벨 2 - 인증만 제공 (with authentication but without privacy)
- 레벨 3 - 인증과 보안 모두 제공 (with authentication and with privacy)

레벨 값이 높을수록 안정된 보안 기능을 제공하며 암호화가 되었을 경우에 인증은 반드시 선행되어야 한다. SNMPv2의 인증 메커니즘은 무결성의 검증, 메시지의 근원지에 대한 검증, 메시지 시기적절성을 위한 MD5요약 인증 프로토콜이다. 이러한 프로토콜과 함께 멀티 보안 프로토콜을 사용할 수 있도록 정의하고 있다[7].

SNMP의 보안 목적은 첫째 각 수신된 메시지가 네트워크를 통한 전송중에 변조되지 않아야 하며, 둘째 인증(사용자의 신원 확인)을 제공하며, 셋째 필요한 경우 수신된 메시지가 데이터 유출로부터 안전함을 제공해야 한다. 보안 서비스는 데이터 무결성과 데이터 인증, 데이터 기밀성, 시기 적절성 등이 있다. 이때 시기 적절성은 인증이 된 경우에만 수행되어야 하며 인증 모듈을 수행한 후, USM모듈의 고정 부분인 timeliness를 수행한다. SNMPv3에서의 암호화와 관련된 요소들은 아래와 같다.

- username - 사용자의 이름을 나타내는 octet string이다.
- privKey - DES 키와 초기화 벡터의 입력값으로 사용되는 사용자의 비밀키이다. privKey의 길이는 16octet이어야 한다.
- msgAuthoritativeEngineID - 특정 메시지에 대하여 인증된 SNMP엔진을 명시하는 인증된 메시지이다.

길이가 16octet인 privKey의 선행 8octet은 DES의 키로 사용이 되며, 나머지 8octet은 키로서 사용되지 않는다. 초기화 벡터값은 SNMP 엔진의 32-비트 snmpEngineBoots와 local 32-bit integer의 값과 XOR된다. 여기서 snmpEngine boots는 snmpEngineID가 마지막으로 구성된 이후로 SNMP engine이 재 초기화된 시간 숫자의 카운터이고 local 32-bit integer는 boot time에서 임의적으로 초기화되는 값이다. 따라서 같은 키를 사용하는 두 개의 데이터 packet은 초기화 벡터로 인하여 서로 다른 암호화 메시지를 생성한다. 암호화된 메시지 전송 서비스는 다음과 같은 요소들로 구성된다.

- statusInformation - 암호화 과정의 성립 여부를 알린다.
- encryptKey - 암호 알고리즘에 사용되는 비밀키이다.
- dataToEncrypt - 암호화될 데이터로써 scopedPDU가 해당된다.
- encryptedData - 암호화된 데이터이다.

위에서 명시된 scopedPDU는 contextEngineID, contextName, PDU로 이루어진 데이터 블록으로써 contextEngineID는 SNMP 엔티티에 의해 접근 가능한 관리 정보의 집합인 context에 대하여 engineID를 명시하며, context Name은 사용되는 물리적 장치의 이름으로 둘 다 고유성을 가진다.

2.3 SNMP에서의 인증

관리 스테이션으로부터 에이전트로 가는 모든 메시지는 커뮤니티 이름을 포함하는데 이때 커뮤니티 이름이 패스워드의 기능을 하며, 송신자가 패스워드를 알고 있다면 메시지는 인증된 것이라고 가정된다. 커뮤니티 이름은 단순히 초기 패스워드-스크린 장치로써의 역할과 함께 인증을 수행한다. 다수의 관리자가 있는 경우에 에이전트는 각각에 서로 다른 접근 권한을 부여할 수 있는데 이러한 접근 제어에는 <표 1>에서와 같이 두 가지 측면이 있다.

① SNMP MIB view : MIB내 객체(object)들의 집합이다.
② SNMP 접근 모드 : [READ-ONLY, READ-WRITE] 두가지 요소이다.

〈표 1〉 SNMPv2 MIB MAX-ACCESS Value와 Protocol Access Mode사이의 관계
〈Table 1〉 The relation of SNMPv2 MIB MAX-ACCESS Value and Protocol Access Mode

MIB-view / MAX-ACCESS VALUE	SNMPv2 Access Mode	
	READ-ONLY	READ-WRITE
read-only	Available for get and trap operations	
read-write	Available for get and trap operations	Available for get, set and trap operations
read-create	Available for get and trap operations	Available for get, set, create, and trap operations
accessible-for-notify	Available for trap operations	
not-accessible	Unavailable	

접근 제어에 사용되는 두 가지 모드는 각각의 커뮤니티 이름에 부여할 수 있고, 두 가지 모드를 합하여 SNMP 커뮤니티 프로파일이라 한다. SNMPv2에서는 MD5메시지 알고리즘을 사용하여 송신 메시지에 대해 비밀값으로 계산된 메시지 요약을 생성한 후 전송된다. 수신측에서는 수신된 메시지와 비밀값을 이용하여 메시지 요약을 생성한 후 송신된 메시지 요약과 수신측에서 계산된 메시지 요약이 같은지를 확인하며, 같은 경우 수신된 메시지는 믿을 만한 것이며, 메시지 요약의 사용은 무결성을 보장하고 비밀값의 사용은 근원지 인증을 보장한다. SNMPv3에서 제공되는 인증 프로토콜은 기존 SNMPv2에서와 같이 MD5를 사용하며, 사용되는 요소들은 아래와 같다.

- usrname - 사용자 이름을 명시하는 string
- authKey - digest를 계산할 때 사용되는 사용자 비밀키
- msgAuthoritativeEngineID - 특정 메시지에 대하여 인증된 SNMP 엔진을 명시하는 인증된 메시지

위와 같은 SNMPv3에서 제안된 알고리즘에 암호 및 인증 시스템을 구현하기 위해 다음의 표준 암호 알고리즘을 이용한다.

2.4 암호 알고리즘

현대 암호화 기술의 특징은 알고리즘의 공개와 키의 보호이다. 암호화 방식은 크게 데이터를 교환하는 사용자들끼리 하나의 키를 공유하며 암호화 키와 복호화 키가 서로 같은 대칭형 암호화 방식과 수학적 알고리즘을 통하여 유일한 하나의 쌍을 갖는 공개키/비밀키로 암호화하고 복호화하는 비대칭형 암호화 방식이 있다.

2.4.1 비밀키 암호화 방식

비밀키 암호 알고리즘으로 대표적인 DES는 64비트의 평문 입력을 56비트의 키를 사용하여 64비트 단위의 블록 암호문으로 만드는 블록 암호시스템이다. 64비트의 입력 후에 전치와 대치, 비트를 삽입하는 확장과 비트의 길이를 고정하기 위한 압축의 단계를 주어진 알고리즘에 따라 수행한다. 그러나 DES는 키 길이의 취약성으로 인하여 크랙 메커니즘에 대하여 DES를 확장하고 변조한 Triple-DES, LOKI, IDEA, MMB와 같은 많은 비밀키 암호화 알고리즘이 발표되었다. 비밀키 암호화 방식은 단순히 암호화만을 제공하므로 인증 기능을 위하여 MD5와 같은 해쉬함수를 함께 사용한다[8].

2.4.2 공개키 암호화 방식

관용 암호화 방식이 갖는 키에 대한 분배와 관리의 어려움 그리고 디지털 서명을 확인할 수 없는 문제점들을 해결하기 위하여 제안된 것이 공개키 암호화 방식이다. 공개키 방식에서는 두 개의 키를 사용하는데 공개키는 모든 사람이 볼 수 있도록 디렉토리 파일 등에 공개하고 공개키와 수학적인 관계를 갖는 비밀키만을 관리하면 된다. 공개키와 비밀키는 수학적인 관계를 통하여 공개키를 통한 비밀키의 유추가 거의 불가능하며 관용키 방식과는 달리 디지털 서명과 인증이 가능하다. 널리 알려진 공개키 방식은 단지 키 교환만을 제공하는 이산 대수 문제를 통한 Diffie-Hellman 방식과 Merkle-Hellman, RSA, RSA를 변형한 Lucifer 등이 있다[9, 10].

다음은 공개키 암호화 방식인 RSA알고리즘의 예이다.

* RSA 알고리즘

1. 두 개의 큰 소수 A, B를 선택한다.
2. n = A*B 인 n을 공개한다.
3. $\varphi(n)=(A-1)*(B-1)$과 서로 소인 임의의 정수 e를 선택한다 (e= 공개키)
4. $e*d \equiv 1(\mathrm{mod}\,\varphi(n))$을 만족하는 d값을 구한다.

* 암호화 단계

평문 M을 공개키 e를 사용하여 다음과 같이 암호화한다.

$$C \equiv M^e(\mathrm{mod}\ n)$$

* 복호화 단계

암호문 C를 비밀키를 사용하여 다음과 같이 복호화한다.

$$C^d \equiv (M^e)^d = M^{t\varphi(n)+1} = M^{\varphi(n)t} \cdot M \equiv M(\mathrm{mod}\ n)$$

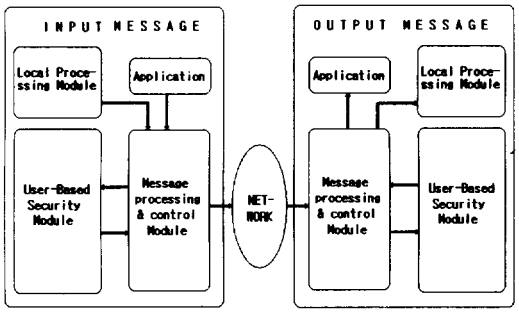
공개키 방식은 원격 로그 인 프로토콜, 전자 화폐, EFT(Electronic Funds Transfer), DB의 분산 운용등에 쓰이고 있다[11, 12].

3. SNMPv3 구조와 보안 알고리즘

기존 환경의 프로그램에 공개키 방식의 알고리즘를 적용하는 것은 DES의 키 취약성과 각각 독립된 암호 및 인증 방식을 통합하는 장점을 가지며 이러한 방식은 사용자에게 탄력적인 인터페이스를 제공할 수 있다. 본 연구에서 제안한 방식은 기존의 인터페이스에 사용가능하며, 공개키 생성자를 통한 사용자 관리를 제공한다.

3.1 SNMPv3구조

본 시스템은 Twente 대학의 SNMP Workgroup에서 제공한 알고리즘을 토대로 설계되었다. 시스템의 구성은 크게 다음과 같이 세 모듈로 구성되며 보안 모듈에서의 데이터 처리에 대해서만 언급하기로 한다. (그림 5)는 전체적인 구조를 보여준다[13, 14].



(그림 5) 시스템 모듈 구성도
(Fig. 5) Configuration of system module

3.1.1 메시지 처리 제어 모듈

송수신되는 메시지에 관하여 수행할 작업을 결정하는 모듈로써 응용으로부터의 입력된 메시지를 보안 모듈로 전달하고 보안 모듈로부터 재수신되는 메시지를 전송하기 위한 작업을 실행한다.

3.1.2 사용자 기반 보안 모듈

메시지 처리 모듈을 통해 수신된 데이터는 보안 레벨에 따라 암호화와 인증, timestamp를 사용하는 timeliness를 수행한다. 가장 먼저 암호 모드를 결정하게 되며 암호모드가 선택되면 그에 따른 알고리즘을 수행한다. 구성 요소는 아래와 같이 4가지로 구성되며 (그림 6)은 보안 모듈의 구성도이다.

① CBC-DES - 사용자의 비밀키를 사용한다.
② MD5 - 메시지 요약을 통한 인증을 수행하는 모듈이다.
③ RSA - 키 생성자를 통하여 자신이 사용할 키의 쌍을 생성하며 데이터 송·수신을 위하여 각각의 사용자에 할당된 공개키 테이블을 사용한다.
④ timeliness - 메시지의 반복이나 지연을 막기 위한 모듈이다.

3.1.3 지역 처리 모듈

항상 에이전트에 위치하며 정보를 찾기 위하여 MIB를 사용하는 모듈로써 관리 정보를 처리하거나 제공한다.

3.2 SNMPv3 보안 알고리즘

3.2.1 송신 메시지의 처리 단계

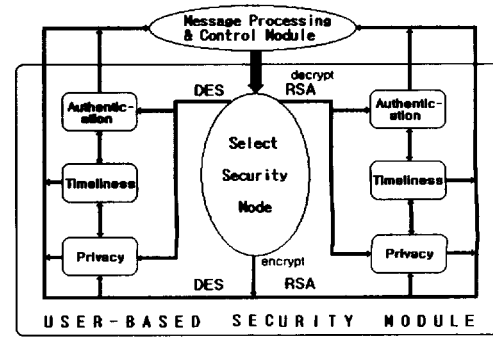
* (CBC-DES)를 이용한 경우

과정 1 : 메시지를 구성하는 Ber-encoded global data, Berencoded scopedPDU와 보안 레벨(LOS), 보안 쿠키(security cookie)의 정보를 선택한 보안 모듈로 전송한다. (보안 쿠키란 고유한 보안 엔티티와 사용자를 정의한다.)

과정 2 : 보안 모듈은 사용자의 이름과 사용자의 snmpEngineID, 사용자가 속해 있는 그룹, 사용자가 사용할 인증 및 보안 프로토콜을 찾는다.

과정 3 : 보안 레벨 체크 후에 선택한 암호 및 인증 프로토콜이 설치되어있는지를 확인한다. 적절하지 않은 경우, error 코드를 발생한다.

과정 4 : 과정1, 2, 3의 단계가 모두 성립이 되면 레벨



(그림 6) Security 모듈 구조
(Fig. 6) The structure of security

에 따른 보안 및 인증 모듈이 선택된다.
과정 5 : 모듈은 scopedPDU를 암호화하고 인증이 필요한 경우, username과 engineID를 timeliness 모듈로 보낸다.
과정 6 : 인증 모듈은 전체 메시지에 대하여 메시지 요약을 계산한 후 재구성된 메시지를 반환한다.

* RSA를 이용한 경우

RSA의 경우는 메시지 처리 단계에 있어서 키 생성기를 통하여 각 사용자와 사용자가 속해 있는 그룹, snmpEngineID에 대하여 고유의 키 쌍을 생성한다. 생성된 키는 메시지를 주고받는 매니저와 매니저 또는 에이전트에 각각의 공개키를 테이블화한다. RSA를 통한 암호 및 인증 알고리즘은 사용자의 공개키 테이블과 자신이 생성한 키 쌍을 통하여 이루어지며 기본적으로 과정 1~과정 4까지의 프로토콜은 DES의 인터페이스와 같다, 수행되는 프로토콜은 다음의 두 모듈에서 정의한 것과 같다.

- 보안 모듈 - 상대방의 공개키로 암호화하여 기밀성을 제공한다.
- 인증 모듈 - 자신의 비밀키로 암호화하여 인증 기능을 제공한다.

과정 1~과정 4 : DES 인터페이스와 동일.
과정 5 : 인증이 필요한 경우 자신의 비밀키로 scoped-PDU를 암호화한다.
과정 6 : 보안이 필요한 경우 과정 5를 수행한 메시지 필드를 상대방의 공개키로써 암호화한다.

3.2.2 수신 메시지 처리 단계

* CBC-DES를 이용한 경우

과정 1 : 수신된 메시지는 암호 및 인증 파라미터들이 아닌 사용자 이름과 snmpEngineID가 있는 필드를 BERdecode한다.
과정 2 : 사용자 이름의 유무성을 확인한 후, userdata를 찾는다.
과정 3 : 보안 레벨 확인 후에 암호 및 인증 프로토콜이 설치되었는지 확인한다.
과정 4 : 과정 1, 2, 3의 단계가 모두 성립이 되면 메시지는 인증 모듈을 수행한다.
과정 5 : 표준인 MD5의 복호 알고리즘을 통하여 메시지를 인증한다.
과정 6 : timestamp 시기적절성을 통하여 두 번째 인증을 수행한다. (timestamp는 timeliness 모듈에 포함된다.)
과정 7 : 인증이 끝나면 scopePDU를 복호함으로써 보안 모듈에서의 수행을 끝마친다.

* RSA를 이용한 경우

과정 1~과정 3까지는 DES에서와 같은 인터페이스를 제공하며 인증과 보안에 사용되는 privKey의 사용 대신에 키 생성기를 통하여 구축된 테이블내의 키를 사용한다.

과정 1~과정 3 : DES 인터페이스와 동일.
과정 4 : 암호화 및 인증이 모두 되었을 경우 자신의 비밀키로써 scopedPDU를 복호화한다.
과정 5 : 과정 4를 수행한 메시지 필드를 송신측 공개키로 복호화함으로써 수행을 끝마친다.

4. Java를 이용한 공개키 방식 SNMPv3 시스템 구현

RSA 공개키 방식이 첨가된 SNMPv3시스템 설계는 객체 지향 프로그래밍 언어인 Java를 사용하였다. 시스템 구축에 있어서 3장에서 정의된 모듈들은 사용자들에 따라 다른 모듈로 대체하거나 새로운 모듈을 생성할 수 있다. 에이전트와 매니저 엔티티들간의 데이터 통신 구현에서 가장 기본이 되는 모듈은 메시지 처리 및 제어 모듈이고 다른 모듈과의 상호 데이터 교환은 아래의 인터페이스를 통하여 이루어진다.

4.1 응용 인터페이스

메시지 처리 모듈과의 인터페이스로써 단순히 수신된 메시지와, response PDU 반환을 요구하는 Inform 메시지에 대한 두 가지로 나누어 볼 수 있다. 다음은 이러한 두 가지를 만족시키기 위해 사용된 메서드와 파라미터이다.

```
public void received( LoS los,
                      int scopedPDUmms,
                      int securityModelNR,
                      int mms,
                      SecurityCookie securityCookie,
                      ScopedPDU scopedPDU);
```

```
public ScopedPDU receivedInform(
                        LoS los,
                        int scopedPDUmms,
                        int securityModelNR,
                        int mms,
                        SecurityCookie securityCookie,
                        ScopedPDU scopedPDU);
```

응용 인터페이스에서의 작용은 첫째, 변수로 정의된 scopedPDU, securityModelNR, mms와 같은 데이터를 선택된 응용 프로그램에 전송하고 둘째, inform 메시지의 도착에 대하여 응용 프로그램에 대하여 response-PDU를 요구하는 기능을 한다.

4.2 지역 처리 인터페이스

이 모듈은 에이전트에만 존재하며 매니저인 고유 사용자가 요구한 MIB객체 request 메시지인 scoped-PDU에 대하여 메시지 처리 모듈 사이에 하나의 인터페이스를 갖는다. 사용된 메서드와 파라미터는 다음과 같다.

```
public interface LocalProcessing {

    public ScopedPDU processPDU(String group,
                    LoS los, ScopedPDU scopedPDU,
                                int scPDUmms)
}
```

request 메시지가 도착한 후 각각의 데이터들은 메시지 처리 모듈에서 지역 처리 모듈로 전송되며, PDU를 처리하고 응답 메시지를 반환한다.

4.3 데이터 전송 계층 인터페이스

TCP/IP외의 다른 전송 프로토콜을 사용할 경우에 많은 수의 전송 계층을 다루며, 전송 계층과 메시지 처리 모듈 사이의 인터페이스를 정의한다. 사용되는 메서드와 파라미터는 다음과 같다.

```
public interface TransportLayer{
  public void send(snmp3Packet packet) ;
  public snmp3Packet receive() ;
}
```

매니저와 에이전트의 어드레스를 통한 메시지 송수신은 자바 라이브러리에서 제공되는 데이터그램을 사용한다. Java는 두 개의 클래스를 사용하여 UDP프로토콜의 상위에서 데이터그램을 사용한다. DatagramPacket 객체는 데이터 컨테이너이고, DatagramSocket은 DatagramPackets을 보내거나 받기 위해 사용되는 메커니즘이다. 다음은 이러한 메커니즘을 구현한 메서드이다[15].

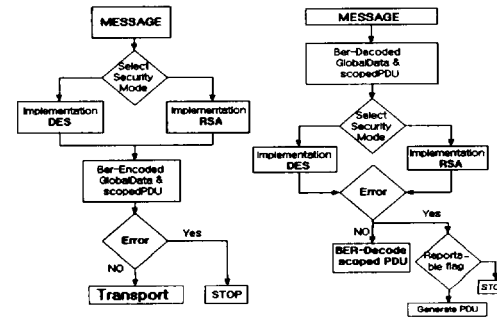
```
public class IPTransportLayer implements TransportLayer
        {
          DatagramSocket DSocket;
        }
```

4.4 보안 모듈 인터페이스

암호화와 인증의 프로토콜 모듈에 대한 인터페이스를 제공한다. 사용되는 프로토콜은 DES와 MD5 그리고 RSA가 사용되며 사용자가 선택한 레벨에 맞는 보안이 수행된다. 모듈내에서 각각의 사용자는 adduser, removeuser의 메서드로 사용자를 추가하거나 변경 갱신할 수 있다. 또한 시큐리티 모듈은 수신된 메지지를 체크함으로써 인증 작업과 scopedPDU를 복호화한다. 다음은 Security Module에 사용되는 메서드와 파라미터이다.

```
public interface SecurityModule {
     public byte[] securityMessage();
     public DecodeMessage checkMessage();
     public abstract int getSecModekNR();
    }
```

각각의 메서드는 globalData, scopedPDU, security-Cookie를 매개변수로 사용하며 메시지를 송수신 할 경우 시큐리티 모듈의 흐름도는 (그림 7)과 같다



(그림 7) 메시지 보안 흐름도
(Fig. 7) The security flowchart for message

① DES 클래스

보안 모드중 DES에 의해 암호화를 수행할 때 사용되는 메서드와 파라미터는 다음과 같다.

```
public class USMDES implements PrivProtocols {
    Hashtable userKeys ;
    public void addUser();
    public void removeUser();
    public privacyResult encrypt();
    public byte[] decrypt();
    public OID getOID();
}
```

snmpEngineID와 username을 합한 스트림이 user-Key이다.

선행되는 8octet이 비밀키로 사용되며 자바에서 제공되는 라이브러리 함수인 해쉬함수 테이블로 값들이 할당된다. addUser와 removeuser함수를 이용하여 매니저와 에이전트 사이의 사용자 추가 및 변경이 가능하다. 사용자와 쌍을 이루는 키들은 테이블에 저장되며, getOID를 사용하여 객체의 실체를 확인한다.

② Timeliness 클래스

Timeliness는 메시지가 특정 시간안에 전송되어질 때 체크되며, 이러한 작업을 수행하기 위해서는 boots와 snmpEngine의 시간이 저장되며 메시지에 첨가될 timestamp를 제공한다.

```
public class Timeliness {
                        Hashtable engineTimes;
                        Timeliness();
                        getTimestamp();
                        setTimestamp();
                        checkTimestamp();
                       }
```

③ MD5 클래스

MD5는 512비트 입력 메시지를 16개의 32비트 블록을 입력으로 하여 128비트 길이의 해쉬코드 값을 출력하는 함수로 송신 메시지에 첨가되고 수신자는 수신된 메시지와 같은 키를 통하여 생성된 메시지 요약을 비교하여 인증 유무를 판단한다. 사용된 메서드는 DES에서 사용된 메서드와 유사하다.

```
public class USMMD5 implements AuthProtocols {
        public USMMD5();
}
```

④ RSA 클래스

DES의 사용은 수행속도에서 RSA보다 성능이 좋지만 키 길이의 취약성으로 인하여 이를 보완한 많은 알고리즘이 개발되고 있다. 공개키의 키 생성은 임의의 큰 소수 두 개를 이용하여 오일러의 법칙을 적용한 후, 생성된 수와 서로 소인 정수를 선택한다. RSA 프로토콜은 3장에서 언급한 과정을 통해 수행되며 메서드와 파라미터는 다음과 같다[16].

```
public class USMRSA implements Auth_PrivProtocols{
      public RSA_Encrypt(P_e);
      public RSA_Auth();
      public RSA_Decrypt(P_d, P_a );
      public OID getOID();
}
```

키 생성 클래스는 다음과 같다.

```
public class Key_Generate {
        long first_p;
        long second_q;
        public Key_Generator(long p, long q);
}
```

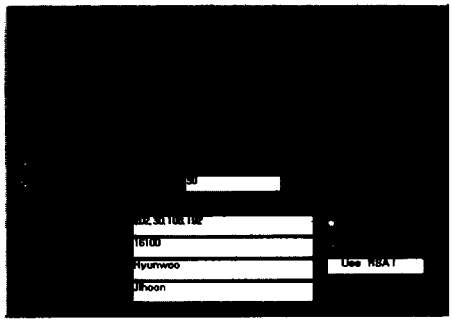
키 생성자를 통하여 생성된 키는 자바에서 제공되는 라이브러리 함수인 해쉬함수 테이블에 값을 할당할 수 있다. DES 모드와 같이 각 사용자에 대하여 쌍을 이루는 키들은 테이블에 저장되며, getOID를 사용하여 객체의 실체를 확인한다.

위에서 선언된 클래스 함수들은 각각의 고유 기능을 실행하게 되며 전체적인 인터페이스는 앞에서 본 (그림 5)와 같다.

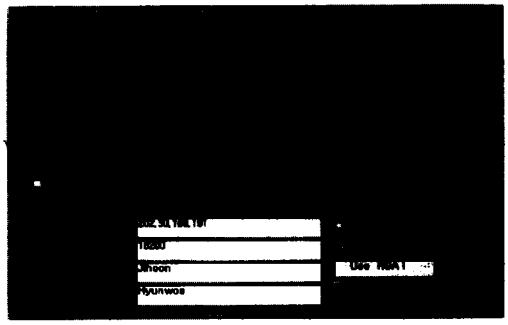
5. SNMPv3 시뮬레이션 및 분석

5.1 매니저 실행 모델

매니저는 Request, Inform, SetRequest 명령을 실행할 수 있는데 Request와 SetRequest는 에이전트의 MIB 정보를 액세스할 때에 사용되며 Inform 명령은 매니저 엔진들 사이의 정보를 주고 받을 때에 사용된다. 매니저 프레임 실행 화면은 (그림 8)과 같다.



(그림 8) 매니저 프레임
(Fig. 8) Manager frame



(그림 9) 에이전트 프레임
(Fig. 9) Agent frame

프레임 구성은 메시지 송신과 관련된 수신자의 IP 주소와 engineID, 사용자의 이름을 입력하는 필드와 수행할 수 있는 명령인 authors, 시간, Request, Inform, SetRequest 그리고 암호 방식 선택 모드인 DES와 RSA로 구성이 된다. RSA 공개키 방식을 사용할 경우에 키 생성 함수를 실행하여 고유의 키 쌍을 생성하며 자바 라이브러리 메서드인 Datagram방식을 사용하여 키를 전달한다. authors를 체크할 경우, 인증 과정을 수행하며 시간을 체크할 경우에 메시지 지연과 재발송을 막는 timeliness모듈을 수행한다.

매니저에서 에이전트로 Request, Inform, SetRequest명령을 실행하면, 메시지는 앞에서 언급한 모듈내에서 주어진 연산을 수행하며 인증되지 않은 snmpEngine의 경우나 사용자의 이름, 또는 IP주소가 맞지 않을 경우에 모니터 창에 각각에 해당하는 에러 메시지를 출력한다. 모든 설정이 올바른 경우에 모니터 창에 실행 명령 메시지가 출력되고 자세한 정보는 (그림 11)과 같은 활성창을 통해서 출력된다.

5.2 에이전트 실행

에이전트는 매니저에 의한 응답만을 실행하는데 에이전트 자체의 문제와 같은 정보를 전송할 때에 Trap을 통하여 메시지를 전송한다. 에이전트 프레임 실행 화면은 (그림 9)와 같다.

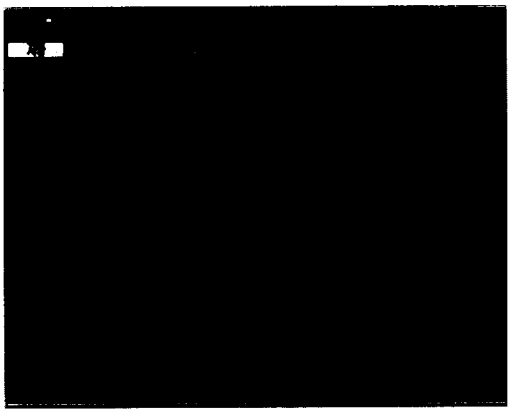
프레임 구성은 송신측의 IP주소와 정보를 기입하는 필드와 engineID와 사용자 이름 필드로 구성이 되고 수행할 수 있는 명령은 트랩과 시간이며, 매니저와 같은 암호화 선택 모드, 키 생성 함수로 구성이 된다. Trap 명령 실행 후에 테이블에 정의되지 않은 사용자 또는 IP주소가 맞지 않을 경우, 모니터 창에 각각에 해당하는 에러 메시지를 출력한다.

매니저의 Request 또는 SetRequest 명령을 받은 경우, 에이전트는 메시지 수신을 보여주며 선택한 암호화 모드와 인증 프로토콜과 같은 정보를 (그림 12)처럼 활성창에 출력한다. 매니저와 에이전트에서 보안 모드를 RSA로 선택할 경우에 (그림 10)과 같은 키 생성 프레임이 나타난다.

(그림 10) 키 생성 프레임
(Fig. 10) Key generate frame

매니저와 에이전트 프레임에 대한 활성창의 메시지는 각각(그림 11), (그림 12)와 같으며 시뮬레이션은 Windows NT 서버 환경에 JDK 1.1.6을 사용하였다.

(그림 11) 매니저 활성창
(Fig. 11) Active window of manager



(그림 12) 에이전트 활성창
(Fig. 12) Active window of agent

6. 결 론

본 연구에서는 네트워크에서 매니저와 에이전트들 간의 정보 교환에 있어서 표준으로 지정된 DES와 MD5에 보안성을 향상시키기 위하여 RSA 방식을 Java로 구현 설계하였다. RSA방식을 이용할 경우 각각의 이용자의 쌍에 보안에 대한 인식이 점차 부각되고 있는 현실에 맞게 안전한 망 관리를 위하여 키의 안정성이 증명된 공개키 방식을 적용한 시스템을 제안 설계하였으며, 모델링과 분석을 통하여 매니저와 에이전트에서 수행되는 작업을 각각 검증하였다.

본 논문에서 구현된 RSA 공개키 방식을 사용하여 향상된 보안을 제공하고 인증 및 암호화 작업을 동시에 수행하는 장점을 갖는다. 또한 현재 많이 쓰이는 SNMPv2에 대하여 표준화 단계가 진행중인 SNMPv3를 자바로 구현하였다. 시뮬레이션 결과, 공개키를 사용함으로써 기존 방식에서 사용되던 DES보다는 수행속도가 느려지고 IP어드레스를 통한 포트 입력에서 포트를 인식하지 못하는 단점을 발견했지만 하드웨어의 구현으로 충분히 보완할 수 있다고 사료된다.

향후 연구 과제를 살펴보면, 표준안에 따라 멀티 인증 및 암호화 모듈을 사용할 수 있으므로 앞으로 많은 보안 알고리즘의 적용과 매니저와 에이전트간의 서로 소유하고 있는 사용자 정보 테이블 및 키 테이블 대신 신뢰할 수 있는 제3의 인증 기관을 통한 키 교환을 통하여 사용자 정보와 키 소유의 문제점을 해결할 수 있는 연구가 필요하다.

참 고 문 헌

[1] Gilbert Held, *LAN Management with SNMP and RMON*, Wiley-Computer, pp.2-25, 1996.

[2] http://www.standard.nca.or.kr/kis/kis-std/k081s96c.htm.

[3] http://www.snmp.com/v3hotspot/RFCs/rfc2274.txt

[4] ftp://ietf.org/internet-drafts/draft-ietf-snmpv3-usm-04.txt

[5] William Stallings, *SNMP SNMPv2 and RMON*, Addison-Wesley, pp.157-184, 1996.

[6] ftp://ftp.isi.edu/in-notes/rfc2272.txt

[7] http://www.snmp.com/v3hotspot/RFCs/rfc2274.txt

[8] Bruce Schneier, *Applied Cryptography*, John Wesley& Sons, pp.265-330, 1996.

[9] Bruce Schneier, *Applied Cryptography*, John Wesley & Sons, pp.461-525, 1996.

[10] M.bellare, P.Rogaway, "Optimal Asymmetric Encryption-How to encrypt with RSA," *Advance in Cryptology-EUROCRYPT'94 Processing*, Computer Science Vol.950, A De Santised, 1995.

[11] William Stallings, *Network and Internetwork Security Principles and Practice*, IEEE, pp. 107-156, 1995.

[12] Douglas R. Stinson, *Cryptography : Theory and Practice*, CRC Press, pp.114-161. 1995.

[13] http://wwwsnmp.cs.utwente.nl/software/

[14] William Stallings, "SNMP and SNMPv2 : The Infrastructure for Network Management," IEEE, pp.37-43, Mar., 1998.

[15] Mike Cohn, Bryan Morgan, Michael Morrison, Michae T. Nygard, Dan Joshi, Tom Trinko, *Developer's Reference JAVA* Sams.net, 1997.

[16] Jonathan Knudsen, *JAVA Cryptography*, O'Reilly, pp.120-192, 1998.

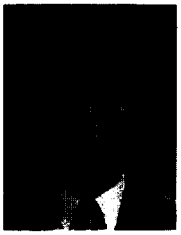
한 지 훈

e-mail : hanji@wh.myongji.ac.kr

1997년 명지대학교 전자공학과 졸업(공학사)

1997년~현재 명지대학교 대학원 전자공학과 석사과정

관심분야 : 네트워크 보안, 자바 프로그래밍, 암호학



박 경 배

e-mail : pgb@wh.myongji.ac.kr

1994년 명지대학교 전자공학과 졸업(공학사)

1996년 명지대학교 대학원 전자공학과 졸업(공학석사)

1996년~현재 명지대학교 대학원 전자공학과 박사과정

1998년~현재 여주대학 전산정보처리학과 전임강사

관심분야 : 멀티미디어 통신, 네트워크 보안, 자바 프로그래밍, 영상압축

곽 승 욱

e-mail : acalab@wh.myongji.ac.kr

1996년 명지대학교 전자공학과 졸업(공학사)

1998년 명지대학교 대학원 전자공학과 졸업(공학석사)

1998년~현재 명지대학교 대학원 전자공학과 박사과정

관심분야 : 멀티미디어 통신, 고장 감내, 컴퓨터 구조

정 근 원

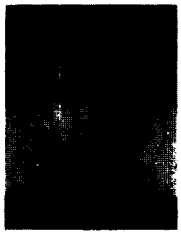
e-mail : acalab@wh.myongji.ac.kr

1991년 명지대학교 전자공학과 졸업(공학사)

1993년 명지대학교 대학원 전자공학과 졸업(공학석사)

1993년~현재 명지대학교 대학원 전자공학과 박사과정

관심분야 : ATM, 컴퓨터 구조 시스템, 멀티미디어 통신



김 정 일

e-mail : acalab@wh.myongji.ac.kr

1989년 명지대학교 전자공학과 졸업(공학사)

1994년 8월 명지대학교 대학원 전자공학과 졸업(공학석사)

1998년 8월 명지대학교 대학원 전자공학과 졸업(공학박사)

관심분야 : 영상통신, 음성통신, 멀티미디어 시스템

송 인 근

1978년 고려대학교 전자공학과 졸업(공학사)

1983년 고려대학교 대학원 전자공학과 졸업(공학석사)

1983년~1995년 한국전자통신연구소 교환연구단 선임연구원

1991년~현재 명지대학교 대학원 전자공학과 박사 과정

1995년~현재 우송대학교 전자공학과 교수

관심분야 : 멀티미디어 통신, 이동통신 시스템

이 광 배

e-mail : kblee@wh.myongji.ac.kr

1979년 고려대학교 전자공학과 졸업(공학사)

1981년 고려대학교 대학원 전자공학과 졸업(공학석사)

1981년 3월~1982년 3월 삼성반도체연구소

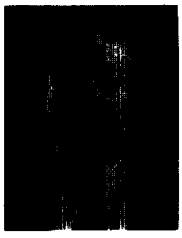
1982년 3월~1983년 4월 금성반도체연구소

1984년~1986년 Univ. of Southern California, Computer Engineering 전공(공학석사)

1986년~1991년 Arizona state Univ., Electronic Engineering 전공(공학박사)

1992년~현재 명지대학교 전자공학과 부교수

관심분야 : 멀티미디어(영상 및 음성 신호처리), 병렬처리 및 고속 컴퓨터(prolog 방식), Communication System(고장 감내형)



김 현 욱

e-mail : acalab@wh.myongji.ac.kr

1978년 고려대학교 전자공학과 졸업(공학사)

1980년 고려대학교 대학원 전자공학과 졸업(공학석사)

1987년 고려대학교 대학원 전자공학과 졸업(공학박사)

1980년~1981년 동양공업 전문대학 전자과 전임강사

1981년~1988년 명지대학교 전자공학과 교수

1988년~1990년 Dept. of Computer science of Arizona State University Adjunct Faculty

1990년~현재 명지대학교 전자공학과 교수

관심분야 : 병렬처리 컴퓨터 시스템, 고장 감내 시스템, 멀티미디어 시스템(영상통신, 음성통신)