LDAP을 이용한 보안 키 관리 시스템 구현

윤성중* · 김건웅*

An Implementation of Security Key Management System by LDAP

Sung-Jung Yoon* · Geonung Kim*

이 논문은 한국과학재단 목적기초연구(R05-2002-000-01055-0)의 지원을 받았음

요 약

보안 키 관리 기능은 망 보안을 위한 필수 요소 중 하나로서, IPSec, HIP 등 다양한 프로토콜에서 이러한 기능을 요구하고 있다. 망 계층에서 보안 키 관리 기능을 제공하기 위한 두가지 방안이 있는데, 그것들은 디렉토리 서비스를 이용하는 것과 DNS 서비스를 이용하는 것이다. 본 논문은 디렉토리 서비스를 이용하여 보안 키 관리 시스템을 구현한 예를 소개한다. 디렉토리 서비스(OpenLDAP)와 공개키 알고리즘(FLINT/C), IPSec(FreeS/WAN)의 기능들을 공개 소프트웨어들을 이용하여 구축하였으며, 보안 키 관리 시스템을 이용한 암호화된 메시지 교환, IKE 데몬과의 연동을 통해 구현된 관리시스템의 기능을 확인하였다.

ABSTRACT

The security key management function is a key element to secure network environment, and many protocols include IPSec, HIP, etc. demand this function. There are two solutions to provide the key management function in the network layer; one is a method for storing security key material in the directory, and the other is a method for storing security key material in DNS. In this paper we present an implementation of key management system by LDAP. We deployed the open source solutions for directory service(OpenLDAP), cryptographic algorithm (FLINT/C), IPSec(FreeS/WAN), and verified the key management system by the encrypted message exchange and the interoperability test by IKE daemon.

키워드

망 보안, IPSec, 키 관리 기능, 디렉토리 서비스

* 목포해양대학교

접수일자 : 2005. 12. 6

Ⅰ. 서 론

1990년대 초부터 기존 인터넷의 구조적 문제인 IP(Internet Protocol)주소 부족, 라우팅(routing)에 관련된 여러 문제점, 실시간 멀티미디어 서비스 제공의 어려움, 보안의 취약점을 해결하기 위한 IPng(IP next generation)연구가 시작되었다. IPv6 프로토콜은 많은 IETF(Internet Engineering Task Force) 제안서와 WG (Working Group)의 활동으로 개발되어 IPng를 위한 권고안이 1994년 7월에 만들어지고, 그 권고안에 따라 IPv6가 개발되었다[1][2]. IPv6는 기존의 IPv4의 기능 중 이용되지 않은 것은 제거하고, 새로 필요한 기능은 추가하는 방향으로 설계되었다. 그 결과, 128비트로 주소 능력이 확장되었으며, 헤더(header) 형식은 단순화하면서 향상된 확장 기능과 옵션을 제공하고, 흐름 레이블(Flow Labeling) 능력, 규모 조정이 가능한 라우팅 등이 제공되며, AH(Authentication Header)와 ESP(Encapsulating Security Payload)를 이용하여 인증(authentication), 무결성(integrity), 비밀성(confidentiality)등의 보안 서비스를 제공하는 IPSec(IP Security)기능이 IPv4에서는 선택 사양이었으나 IPv6에서는 필수적 기능으로 되었다[3][4][5].

IPv6의 구현 작업은 활발히 진행되어, 이미 대부분의 UNIX 계열 운영체제에서 IPv4와 IPv6 프로토콜이 듀얼 스택 형태로 탑재되고 있으며, Window 계열 운영체제인 Windows XP에서도 IPv6가 지원되고 있다[6][7][8][9][10]. 그러나 대부분의 구현이 기본적인 IPv6 프로토콜과 IPSec 프로토콜, 확장된 소켓(Socket) API 제공, IPv4에서 IPv6로의 전환 등에 초점이 맞추어져 있고, IPSec을 지원하기 위해서는 필수적인 키 관리(key management) 기능의 구현은 미루어지고 있다. IPSec을 지원하기 위해서는 수동적 SA(Security Association) 및 키 관리와 자동적 SA 및 키 관리를 모두 지원해야 한다. 수동적 방식은 가장 단순한 형태로, 개인이 SA 관리 데이터와 키 재료를 바탕으로 각 시스템을 수동으로 설정하는 것이다. 이 기술은 작고, 정적인 환경 하에서는 실용적이지만, 확장성이 없다. IPSec이 널리 보편화되고 사용되기 위해서는, 확장성이 있고 자동화된 SA 관리 프로토콜이 필요하다. 그런 프로토콜은 AH, ESP의 재사용(replay) 방지 특성을 쉽게 사용할 수 있도록 해야 하며, 사용자 기반과 세션(session) 기반의 키 방식의 SA를 생성하는 요구를 수용해야 한다. 또한 IPSec과는 독립적으로 구현되어 운영되면서, 앞으로 나올 여러 키 관리 프로토콜 중에서 선택 사용이 가능해야 하고, 키의 수명 주기(PFS: Perfect Forward Secrecy)를 제공해야 하며, UDP(User Datagram Protocol), TCP (Transmission Control Protocol)상에서 운반되는 상위계층의 함수이므로, 망 계층의 보안을 위해 사용되지만 그 자신의 데이터를 보호하기 위해서 응용 계층(application layer)의 보안을 필요로 한다. 이러한 성질을 만족시키기 위해서 IETF에서는 ISAKMP(Internet Security Association & Key Management Protocol)와 OAKLEY 프로토콜의 일부를 이용한 IKE (Internet Key Exchange)를 제안하고 있다[11][12][13][14].

보안의 중요성이 강조됨에 따라 IPSec 외에 다른 프로토콜에서도 시스템의 인증이나 데이터그램의 인증을 고려하고 있는데, 예를 들면 DNS(Domain Name System)[15][16] 서비스의 보안을 위한 DNSSEC(Domain Name System SECurity)[17][18], 새로 종단 플랫폼의 이름 공간을 고려하고 있는 HIP(Host Identity Protocol)[19] 등에서도 인증을 고려하고 있다. 이러한 보안 기능을 제공하기 위해서는 공인된 제3자가 인증 관련 정보를 관리하고, 사용자는 이를 자유로이 접근하고 검색할 수 있어야 하는데, 망 계층이 아닌 응용 계층의 특정 응용, 예를 들면, 전자상거래 분야의 SET(Secure Electronic Transaction)[20]를 위한 사용자 인증기관은 존재하지만 망 계층을 위한 인증기관은 현재까지 존재하지 않는다.

IPSec을 비롯한 키 관련 정보를 유지하고 사용자에게 제공하는 대표적인 두 가지 방안이 하나는 기존의 DNS 기능을 확장하여 관련 키 정보를 유지하고 응답하도록 하는 방법이고, 다른 하나는 디렉토리(directory)[21] 시스템을 이용하여 별도의 키 관리 시스템을 구축하는 방안이다. 본 논문에서는 두 가지 방안 중 디렉토리 서비스를 이용하여 키관리 시스템을 구현한 예를 소개한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 인터넷 보안 구조에 대한 간략히 정리하고, 3장에서는 전체 통신과정에서 키관리 시스템의 역할을 정리한다. 다음 4장에서는 디렉토리 서비스를 이용하여 키 관리 시스템을 구현하기 위해 진행한 작업들을 소개하고, 5장에서는 구현된 시스템과 이를 확인하기 위해 작성한 응용을 소개하고 6장에서 결론을 맺는다.

Ⅱ. 인터넷 보안 구조

인터넷 보안 구조는 4가지 요소들로 구성되어 있는데,

그것들은 보안 프로토콜, 보안 연관, 키 관리 프로토콜, 그리고 인증이나 암호화에 쓰이는 다양한 알고리즘들이다[3].

보안 프로토콜에는 AH(Authentication Header)와 ESP (Encapsulating Security Payload)가 있다. 여기서 AH는 비연결형 무결성(connectionless integrity), 자료 출발지 인증(data origin authentication)을 제공하며, 부수적으로 재전송 방지(anti-replay) 서비스를 제공할 수 있다. ESP는 기밀성(confidentiality)과 제한된 트래픽 흐름 기밀성(traffic flow confidentiality)을 제공하며, 비연결형 무결성, 출발지 인증, 재전송 방지 서비스도 제공할 수 있다. 이러한 프로토콜들은 각각 적용될 수도 있고 동시에 적용될 수도 있다. 또한 각 프로토콜은 트랜스포트 모드(transport mode)와 터널 모드(tunnel mode)로 동작할 수 있다.

보안 연관(SA: Security Association)은 논리적 연결로서 AH나 ESP 서비스는 이를 이용하며, 뒤에서 언급할 키 관리 프로토콜들의 주 역할은 이러한 SA를 생성하고 유지하는 역할을 담당한다. 이러한 SA는 SPI(Security Parameter Index), IP 목적지 주소, 보안 프로토콜 식별자(AH 또는 ESP)로 구별되며, 서비스별로, 방향별로 하나씩 생성되어야 한다. 따라서 하나의 양방향 연결에서 AH와 ESP 서비스를 동시에 이용하고자 한다면 4개의 SA가 필요하다.

이러한 보안 연관에 관련된 데이터베이스로 SPD와 SAD가 있는데, SPD는 각 IP 데이터그램이 어떤 방식으로 다루어져야 하는지에 대한 정보를 담고 있다. IPSec 수신자측은 이를 바탕으로 데이터그램을 폐기, IPSec을 적용하지 않고 통과, 또는 IPSec을 적용한다. 각 SA는 SAD에 하나의 엔트리(entry)로 저장되는데, 여기에는 IPSec 처리에 관련된 순서 번호나 각 프로토콜에서 이용하는 알고리즘이나 키에 대한 정보를 담고 있다.

IPSec에서는 SA의 수동 설정과 자동 설정을 모두 규정하고 있는데, 보안 프로토콜과 독립적으로 운영되도록 되어있다. 수동 설정은 관리자가 직접 각 시스템의 키를 설정하는 방법으로 소규모의 고정된 망 환경에서 운영이 가능하다. 자동 설정은 IKE[11]를 비롯한 키 관리 프로토콜을 이용하여 사용자별, 세션별로 키를 생성하고, 유지하며, 폐기하는 것으로서 현재 IKEv2 [22]가 논의되고 있다. SA의 세밀성(granularity)은 이러한 자동 설정이 지원되는가에 대해 종속적인데, 미세한 SA를 지원하기 위해선 자동 설정이 필수적이며, 또한 규모가 큰 유동 망에서 보안 구조를 적용하고자 할 때에도 필수적이다.

마지막으로 인증이나 암호화에 쓰이는 다양한 알고리즘들인데, 인터넷 보안 구조에서는 이들에 대한 특별한 제약이 없이 모두를 수용할 수 있는 구조로 되어 있으며 상호 호환을 위해 최소한의 알고리즘을 반드시 포함하도록 규정하고 있다.

Ⅲ. 키 관리 시스템의 역할

비대칭형 알고리즘에서는 공개키 또는 개인키로 전달 정보를 암호화함으로써 보안이 가능한데, 문제는 교환하는 상대방이 자신이 주장하는 신분과 일치하는지를 확인하는 인증 문제이다. 이를 해결하기 위한 방안이 전자인증서(digital certificate)와 이를 발급하는 인증기관(CA: Certificate Authority)의 도입인데, 대표적인 표준이 X.509 PKI(Public Key Infrastructure)이다. X.509는 1988년 발표된 이후, 보완과 추가를 통하여 1996년 v3까지 발표되었는데, IETF의 pkix 그룹에서는 X.509를 기반으로 하는 전자인증서의 구성과 전자인증서의 발급/보관/폐기, 전자인증 기관의 구성 등을 정의하는 작업을 하고 있다[23].

IKE에서는 2단계에 걸쳐 SA를 설정하는데, 첫 단계에서는 IKE 자체의 SA를 설정하고, 두 번째 단계에서 IPSec SA를 설정한다. 첫 단계에서는 IPSec SA 설정시 필요한 암호화 매개변수를 합의하고 공유 비밀키를 생성한다. 이때 쌍방간의 인증 방법으로 ①전자서명, ②공개키 암호화, ③수정된 공개키 암호화, ④미리 공유된 비밀키 등을 활용할 수 있다. 이들 중 가장 보안이 강화된 방법이 공개키 암호화 방법을 이용하는 ②와 ③이다. 다음 그림 1은 ②의 1단계의 메인 모드(main mode)를 보여주고 있다. ③은 이러한 공개키 기반 암호화/복호화를 반으로 줄인 방법이다[11].

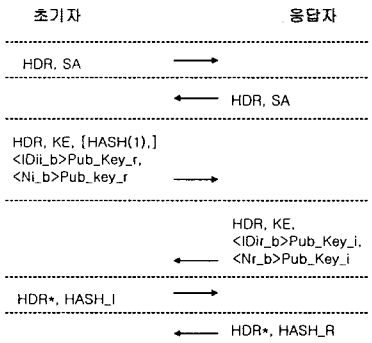

그림 1. IKE 메인 모드
Fig. 1 Main Mode of the IKE

그림 1에서 보이는 바와 같이 6개의 메시지를 통해 인증을 수행하고, IKE SA를 설정한다. 여기서 세 번째 메시지와 네 번째 메시지에서 각각 상대방의 공개키를 이용하여 신원(IDii, IDir)과 nonce(Ni, Nr)을 암호화하여 전송하고, 그것을 자신의 비밀키로 복호화 하여 다음 처리를 함으로써 인증이 이루어진다. 결국 이를 이용하기 위해서는 각자 신뢰할 수 있는 제3자를 통해 상대방의 공개키를 알아내야 한다.

일반적인 공개키 기반 구조의 구성 요소는 ①인증서(Certificates), ②인증서 상태 확인 방법(Certificate Status Mechanism), ③인증기관(CA: Certificate Authority), ④등록기관(RA: Registration Authority), ⑤데이터 복구 에이전트(Data Recovery Agent), ⑥인증서와 CRL(Certificate Revocation List) 저장소, ⑦인증 정책(CP: Certification Policy)과 인증 적용 지침(Certification Practice Statement)이다[9]. 공개키 기반 키 관리 시스템의 구성도 일반적인 공개키 기반 구조의 요소들을 그대로 이용할 수 있는데, 다만 실제로 인증되어야 하는 대상이, 일반적인 사용자가 아닌, 호스트인 관계로 호스트가 스스로 키를 생성하고, 공개키를 등록하는 방법이 제공되어야 한다[24].

일반적인 호스트의 망 구성 설정은 하드웨어 주소와 IP 주소간의 매핑, IP 주소와 호스트 이름간의 매핑으로 나누어 이루어진다. 하드웨어 주소와 IP주소간의 매핑은 호스트 내의 설정 파일을 이용할 수도 있고, DHCP[25]나 RARP[26] 등의 프로토콜을 이용할 수 있다. IP 주소와 호스트 이름간의 매핑은 DNS(Domain Name Service)를 이용한다. IPSec을 도입하고자 하면, 앞서 언급한 두 가지 매핑 이외에도 IP 주소와 그것의 인증 정보간 매핑이 필요하다. 이러한 매핑을 담당하는 것이 공개키 기반 키 관리 시스템이다.

호스트가 부팅되면 자신의 IP 주소를 설정한 후, 그림2에서 보이는 기능을 수행하여야 한다. 이때 CA 관련 정보는 구성 파일을 이용하거나, DHCP를 통해 전달받을 수 있다. 또한 호스트가 종료하는 경우, 자신이 등록했던 공개키를 삭제하는 기능이 필요하다. 반대로, 망 계층 인증 서버는 등록된 공개키의 생명주기가 끝나면 자동 삭제하는 기능이 필요하다.

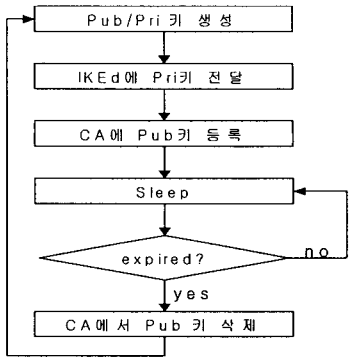

그림 2. 인증 정보 등록/폐기 과정
Fig. 2 Procedure of Registration/Deletion of Authentication Information

그림 3은 LDAP[27]을 기반으로 하는 키관리시스템과 IKEd와의 동작을 보이고 있다. 그림에서는 IKEd의 일부에서 CA와 동작하는 것으로 표현하였지만, IKEd와 이런 인증을 담당하는 기능간의 인터페이스가 정의되면 분리되어 수행할 수도 있다. 또한 망 관리자가 Web 서비스를 기반으로 구성원들의 인증 정보를 검색하고 관리할 수 있는 일련의 작업을 수행할 수 있는 인터페이스를 제공할 수 있다.

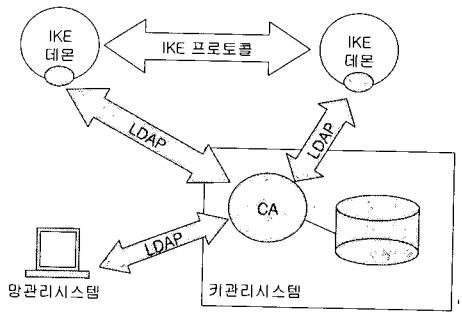

그림 3. 디렉토리 서비스를 이용한 키 관리 시스템
Fig. 3 Key Management System by Directory Service

Ⅳ. 키 관리를 위한 LDAP 서버 구현

4.1 디렉토리 서비스와 LDAP

디렉토리란 정보를 가지고 있는 객체를 일련의 순서로 나열한 정보의 리스트이다. 디렉토리의 대표적인 예가 전화부나 도서관의 도서색인카드 목록들이다. 디렉토리는 특별한 형태의 데이터베이스라고 할 수 있는데, 1980년대 말에 디렉토리 서비스 도입에 대한 요구가 높아감에 따라 CCITT(International Telegraph and Telephone Consultative Committee)와 ISO (Inter national Organization for Standardization)가 공동으로 작업하여 X.500 시리즈를 만들기 시작해서, 1993년, 1997년 몇 번의 수정을 거쳐 현재에 이르렀다. X.500은 최초의 일반적인 목적의 디렉토리 시스템이었고 다양한 쿼리를 사용하는 강력한 검색기능을 제공하였을 뿐만 아니라 서버와 데이터의 분산이 용이했고 그리고 무엇보다도 특정 운영체제나 특정 네트워크, 특정 응용프로그램에 구애받지 않고 사용될 수 있는 표준이라는 점이 특징이다[28].

LDAP(Lightweight Directory Access Protocol)은 TCP/IP 상에서 디렉토리 서비스로 데이터를 저장, 삭제, 수정, 검색을 할 수 있도록 서버와 연결할 수 있도록 한 프로토콜이다[27]. LDAP은 X.500의 DAP(directory access protocol)가 너무 방대하고, 복잡하여 구현이 어려운 문제를 해결하고자 연구되기 시작한 것으로, DAP의 주요 기능을 대부분 지원하면서, 복잡하거나 효용이 떨어진다고 판단되는 기능은 단순화하거나 제거했다. 특히 대부분의 데이터 형식에서 단순한 문자열을 사용함으로써 구현이 용이하도록 하였다. 현재 미시건 대학의 LDAP(UM LDAP)을 비롯하여 많은 상용 또는 오픈 소스의 LDAP 구현물이 나와 있다.

디렉토리에서는 정보를 트리 형태로 구성하고 관리하는데, 이를 DIT(Directory Information Tree)라고 부른다. DIT 구조에서 각 노드들을 엔트리(entry)라고 부르며, 엔트리는 디렉토리에서 하나의 데이터를 나타낸다. 이러한 모든 엔트리는 그 자신의 위치와 고유성을 나타내는 DN(distinguished name)으로 구분되며, 속성(attribute)들을 갖는다. 또한 각 객체들은 그들을 추상화하여 성질을 정의해 놓은 객체 클래스를 통해 관리된다. 모든 엔트리는 한 가지 이상의 obejctclass에 속하게 되며 objectclass의 정의에 열거된 속성들을 가지게 된다. 다음 그림은 DIT의 한 예를 보여준다.

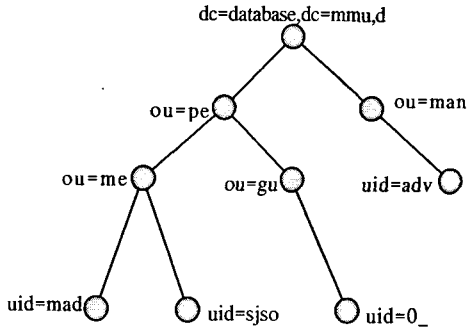

그림 4. 디렉토리 정보 구조 예
Fig. 4 Example of the Directory Information Tree

상용 또는 공개된 LDAP 제품들은 디렉토리와 프로토콜이 모두 포함하고 있다. 우선 클라이언트와 서버 사이의 상호작용을 위하여 정의된 프로토콜이 제공되어 클라이언트/서버, 서버/서버 사이의 연동이 가능하며, 디렉토리 정보 저장소의 기능과 다양한 프로그래밍 언어로 관련 응용서비스를 개발할 수 있는 API들이 제공된다.

4.2 키관리 디렉토리 서버 구현

본 논문에서는 OpenLDAP 2.3을 이용하여 키관리 서버를 구현하였다. OpenLDAP은 공개된 LDAP 구현으로 LDAP 데몬 서버와 서버간 복사를 담당하는 데몬, 그리고 LDAP 프로토콜을 구현한 라이브러리로 구성되어 있다. 현재의 최신 버전은 2.3.12이다[29].

구현을 위해 먼저 키관리 서버에 적합한 스키마를 정의하였는데, LDAP에서의 스키마 작성은 새로운 속성 정의와 이를 바탕으로 하는 새로운 객체 클래스 정의로 이루어진다. 먼저 공개키를 디렉토리에 저장하기 위하여 다음과 같은 속성들을 새로 정의하였다.

```
attributetype ( 1.1.2.3.1 NAME 'IPaddress'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
)
attributetype ( 1.1.2.3.3 NAME 'PubKey'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
)
attributetype ( 1.1.2.3.4 NAME 'InputDate'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
)
attributetype ( 1.1.2.3.5 NAME 'LifeTime'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
)
```

그림 5. 새로 정의한 속성들
Fig. 5 Defined Attributes

예를 들면"IPaddress"는 정수로 이루어져 있으므로 속성 타입의 신택스를 정수형인 "1.3.6.1.4.1.1466.115.121.1.27"로 설정하였고, 매칭 룰(matching rule)은 "EQUALITY" 타입을 사용하고, "intergerMatch"을 사용한다. "Pubkey"는 RSA알고리즘으로 생성된 공개키 값이 바이너리로 생성 되어서 저장되므로 신택스를 바이너리 타입인 "1.3.6.1.4.1.1466.115.121. 1.5"로 설정하였다. 그리고 공개키 값을 찾기 위해서 키의 일부분을 사용하여 검색을 하여 전체 공개키 와 일부분만 일치 한다고 하여 같다고 할 수 없으므로 매칭 룰을 적용하지 않았다.

다음은 공개키를 저장하기 위한 objectclass 이다.

```
objectclass ( 1.1.2.3.6 NAME 'PKImanagement'
        SUP organizationalUnit
        MUST ( IPaddress $ PubKey )
        MAY ( InputDate $LifeTime)
)
```

그림 6. 새로 정의한 객체클래스
Fig. 6. Defined Object Class

objectcass 의 OID는 대외적으로 사용하는 경우 임의로 지정할 수 있으며, 대외적인 사용을 위해서는 IANA (Internet Assigned Numbers Authority)에 등록하는 절차가 필요하다. 새로 정의한 "PKImanagement"클래스는 기존의 "organizationalUnit"클래스를 상속 받았고, 원래 "organizationalUnit"의 필수 속성은 "ou" 이다. 또한 "PKImanagement"클래스의 필수 속성은 "IPaddress"와 "PubKey"이고 선택 가능한 속성으로 "InputDate"와 "LifeTime"을 두었다.

LDAP으로 구현한 키 관리 서버에 데이터를 입력하는 방법은 명령문 형태로 입력할 수도 있고, LDIF(LDAP Data Interchange Format)으로 표현한 파일 형식을 이용할 수도 있다. 기본적인 데이터의 입력은 LDIF를 이용했으며, 키를 생성하는 프로그램에서 LDAP을 통해 등록하도록 구현하였다.

V. 키관리 시스템 구현과 기능 확인

5.1 FLINT/C 라이브러리

본 시스템에서 암호화 알고리즘에 관련된 부분은 FLINT/C(Functions for Large Integers in Number theory and Cryptography) 라이브러리를 이용하였다[30]. 이 라이브러리에서는 암호화 알고리즘에서 필수적으로 필요한 큰 정수 연산 함수들과 그것들을 이용하여 구현한 RSA[31], Rijndael[32] 알고리즘들이 제공된다. 본 연구에서는 암호화 키 생성을 위하여 RSA 암호화 알고리즘을 사용하였는데, 다음과 같은 RSA의 공개키와 비밀키를 위한 클래스를 일부 수정하여 이용했다.

```
class RSAkey
{
  public:
    inline RSAkey (void) {};
    RSAkey (const int);
    RSAkey (const int, const LINT&, const LINT& = 1);
    PKEYSTRUCT export_public (void) const;
    UCHAR* decrypt (const LINT&, int*);
    LINT sign (const UCHAR* const, const int);
    void purge (void);
    RSAkey& operator= (const RSAkey&);

    friend int operator== (const RSAkey&, const RSAkey&);
    friend int operator!= (const RSAkey&, const RSAkey&);
    friend fstream& operator<< (fstream&, const RSAkey&);
    friend fstream& operator>> (fstream&, RSAkey&);

  private:
    KEYSTRUCT key;

    int makekey (const int, const LINT& = 1);
    int testkey (void);
    LINT fastdecrypt (const LINT&);
};

class RSApub
{
  public:
    inline RSApub (void) {};
    RSApub (const RSAkey&);
    LINT crypt (const UCHAR* const, const int);
    int verify (const UCHAR* const, const int, const LINT&);
    void purge (void);
    RSApub& operator= (const RSApub&);
    friend int operator== (const RSApub&, const RSApub&);
    friend int operator!= (const RSApub&, const RSApub&);
    friend fstream& operator<< (fstream&, const RSApub&);
    friend fstream& operator>> (fstream&, RSApub&);

  private:
    PKEYSTRUCT pkey;
};
```

그림 7. RSA 알고리즘 객체 클래스
Fig. 7 Object Classes of the RSA Algorithm

RSAkey 클래스는 공개키와 비밀키를 생성하며, 비밀키를 이용하여 암호문을 복호화하는 함수를 가지고 있고, RSApub 클래스는 RSAkey 클래스에서 생성한 공개키를 이용하여 평문을 암호화 하는 함수를 가지고 있다.

5.2 키 관리 시스템 기능 확인을 위한 암호화된 메시지 교환 구현

구현된 키 관리 시스템의 기능 확인을 위해 다음과 같은 2개의 데몬과 사용자 프로그램을 구현하여 키관리 시스템의 동작을 확인하였다.

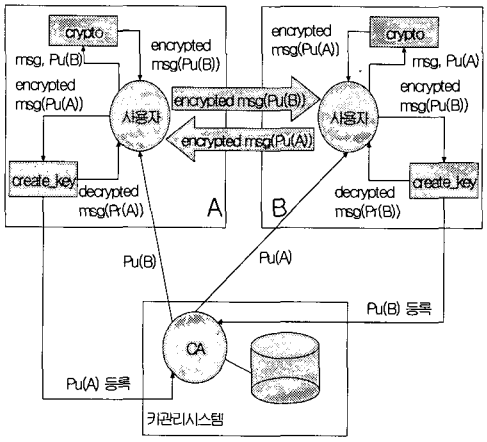

그림 8. 키 관리 시스템을 이용한 응용 예
Fig. 8 An Application of Key Management System

creat_key 데몬은 RSA알고리즘을 사용하여 공개키와 비밀키를 생성하고, 공개키를 LDAP 서버에 저장하는 역할과, 사용자의 요청에 의해 암호문은 복호화 시켜 반환하는 역할을 담당한다. 또한 공개키의 유효시간이 만료되었을 경우 LDAP 서버로부터 이를 통보받고, 사용자의 의견에 따라 공개키를 갱신하거나 공개키를 삭제한다. 사용자가 공개키를 삭제할 경우 LDAP에 저장한 공개키를 삭제하고 create_key 데몬을 종료 시키게 되며, create_key 데몬을 종료 시키는 경우 LDAP 서버에 저장한 공개키를 삭제하고 종료를 하도록 설계하였다.

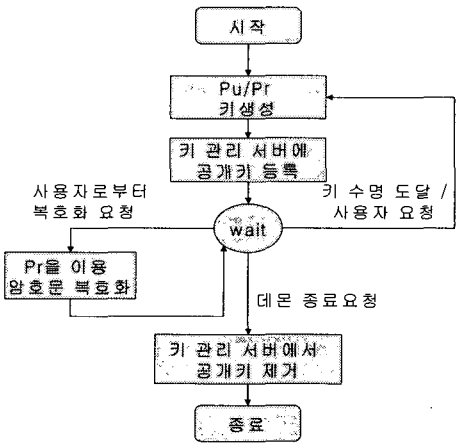

그림 9. create_key 데몬 동작
Fig. 9. Flow chart of the create_key daemon

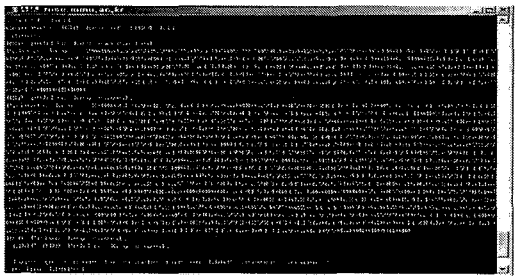

그림 10. 공개키/비밀키 생성 및 공개키 등록 화면
Fig. 10 Creation of Pu/Pr Key and Registration of Pu Key

사용자는 LDAP 서버에 접속하여 공개키 리스트를 검색하여 통신하고자 하는 상대방의 공개키를 선택하여 메시지와 함께 crypto 데몬에게 전송하여 암호문을 만들고, 반환된 암호문을 상대방에게 전송한다. 다음은 사용자가 상대방의 공개키를 검색하고 선택하는 과정을 보여주는 화면이다.

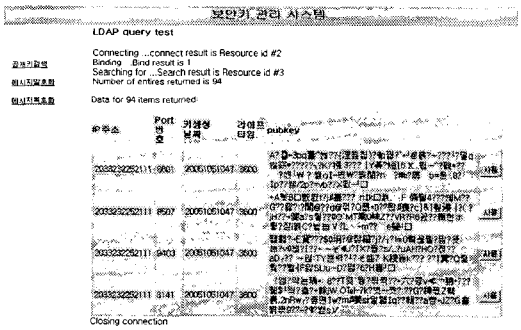

그림 11. 공개키 탐색 결과 화면
Fig. 11 Search of Registered Public Keys

crypto 데몬은 사용자에게 메시지와 공개키를 받아 RSA 알고리즘을 사용하여 메시지를 암호화시켜 암호문을 반환하는 역할을 수행한다. 암호화된 메시지 교환 응용에서는 이렇게 반환된 메시지를 상대방에게 전송하며, 수신자는 자신의 비밀키로 메시지를 복호화하여 통신을 계속 진행한다.

5.3 IKE 데몬과의 연동

본 논문에서 구현된 키 관리 시스템은 IPSec 요소 중 하나인 IKE 데몬과의 연동을 목표로 설계되고 구현되었다. 구현된 시스템과 기존 IKE 데몬과의 동작을 확인하기 위

해 LINUX 환경에서 구현된 FreeS/WAN 2.01[33][34]을 일부 수정하여 자동으로 공개키/비밀키 쌍을 생성하고, 공개키를 키 관리 시스템에 등록하며, 키 관리 시스템에서 상대방의 공개키를 찾아 IKE의 SA 설정 과정을 수행하는 것을 확인하였다.

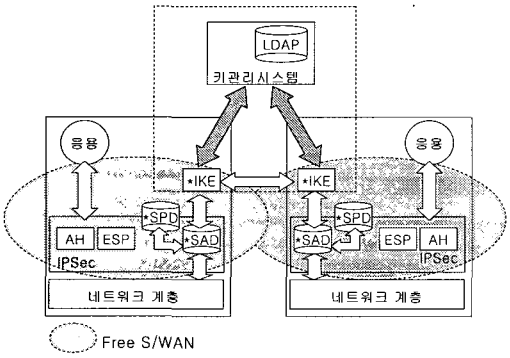

그림 12. 키 관리 시스템과 Free S/WAN과의 연동
Fig. 12 Interoperation between the Key Management System and the FreeS/WAN

FreeS/WAN은 IPSec을 LINUX에서 구현한 것으로서 망 프로토콜 계층에서의 암호화와 인증 서비스를 지원한다. IPSec의 대표적인 세 프로토콜인 AH, ESP, IKE를 지원하는데, KLIPS(kernel IPSec)에서는 AH, ESP와 커널에서의 패킷 처리를 담당하고, Pluto에서는 IKE를 이용한 연결 협상을 담당한다. 키 설정은 수동 설정방법과 자동 설정 방법 모두 지원되는데, 본 논문과 관련된 자동 설정의 경우, 공유된 보안 정보를 이용하는 방법과 공개키 방법이 같이 고려되고 있다.

FreeS/WAN의 Pluto에서 제공하는 기능은 IKE 프로토콜 자체와 키를 생성하는 알고리즘 일부이다. 이때 서버들의 인증 기능은 제공하지 않고 있기 때문에, 이러한 기능 확장을 위해 전체적인 프로그램 분석과 기능 확장 작업을 수행하여 LDAP으로 구현한 키 관리 시스템을 통해 상대 서버를 인증하고 나머지 절차를 수행하도록 하였다. Pluto는 IKE를 구현한 데몬으로 LINUX 상에서 IPSec을 구현한 KLIPS와 맞물려 동작한다. 원래 IKE와 IPSec의 다른 프로토콜은 독립적으로 구현될 수 있어야 하는데, Pluto에 이러한 제한이 있는 것은 보안 정책 데이터베이스(SPD)가 따로 존재하지 않고, 직접 KLIPS 커널에 코딩하여서 야기된 문제이다(spdb.c). 이러한 SPD에 관련된 사항은 현재 논의 중이다[35].

FreeS/WAN은 2004년 4월 22일 마지막 자유 배포판 2.06을 발표했으며, 추후 다른 프로젝트로 계속한다고 선언하였다. 이러한 결과는 FreeS/WAN 구현 팀에서 제안한 optimistic 키 설정 방식이 인정받지 못하였고, ipv6으로의 전환 과정이 예상보다 늦어지는 것에 주된 이유가 있다.

VI. 결 론

본 논문에서는 IKE 등 보안 관련 서비스들에서 이용할 수 있는 키 관리 시스템을 디렉토리 서비스를 이용하여 구현한 결과를 소개하였다. 구현된 키 관리 시스템은 OpenLDAP을 이용하여 구현되었으며, 공개키/비밀키 생성 부분은 FLINT/C 라이브러리, IKE 데몬 부분은 FreeS/WAN을 기반으로 하여 구현되었다. IPSec을 기본으로 지원하는 IPv6, DNS의 보안 기능을 강화하는 DNSSEC, 새로운 융합형 식별자 중 하나인 HIP 등 다양한 분야에서 키 관리 시스템을 요구하고 있으며, 디렉토리 서비스는 이러한 기능을 제공할 수 있을 것으로 판단되는 두가지 방안 중 하나이다.

또 다른 방안인 키 관리 시스템 기능을 DNS에 포함시키는 방안도 연구 중인데, EDNS0[36]를 지원하는 BIND 9.x[37]에 키 관련 확장 자원레코드(resource record)를 포함하는 방법으로 구현 작업이 진행 중이다. 이러한 구현이 완료되면 디렉토리 서비스를 이용하는 방안과 DNS를 이용하는 방안에 대한 비교 분석 작업이 계속 진행될 예정이다.

참고문헌

[1] C. Partridge, F. Kastenholz, "Technical Criteria for Choosing IP The Next Generation (IPng)", IETF RFC 1726, Dec. 1994

[2] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", IETF RFC 2460, Dec. 1998

[3] S. Kent, R. Atkinson, "Security Architecture for the Internet Protocol", IETF RFC 2401, Nov. 1998

[4] S. Kent, R. Atkinson, "IP Authentication Header", IETF RFC 2402, Nov. 1998

[5] S. Kent, R. Atkinson, "IP Encapsulating Security

Payload (ESP)", IETF RFC 2406, Nov. 1998
[6] http://www.ipv6.org
[7] http://www.tahi.org
[8] http://www.sun.com
[9] http://www.software.hp.com
[10] http://www.microsoft.com/windowsxp/pro/techinfo/administration/ ipv6
[11] D.Harkins, D.carrel, "The Internet Key Exchange (IKE)", IETF RFC 2409, Nov. 1998
[12] D. Maughan, M. Schertler, M. Schneider, J. Turner, "Internet Security Association and Key Management Protocol(ISAKMP), IETF RFC 2408, Nov. 1998
[13] H. Orman, "The OAKLEY Key Determination Protocol", IETF RFC 2412, Nov. 1998
[14] Radia Perlman, Charlie Kaufman, "Key Exchanges in IPSec: Analysis of IKE", IEEE Internet Computing Vol. 4, No. 6, Nov. 2000
[15] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987
[16] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987
[17] Eastlake 3rd, D., "Domain Name System Security Extensions", RFC 2535, March 1999
[18] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, "DNS Security Introduction and Requirements", IETF RFC 4033, March 2005
[19] http://www.ietf.org/html.charters/hip- charter.html
[20] Secure Electronic Transaction(SET) Specification Book 1: "Business Description", 1997.5
[21] ISO 9594-1, X.500, "The Directory Part1: Overview of Concepts, Models, and Services", 1993
[22] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", draft-ietf- ipsec-ikev2-17 (work in progress), October 2004
[23] http://www.ietf.org/html.charters/ pkix-charter
[24] http://www.pki-page.org/
[25] R. Droms, "Dynamic Host Configuration Protocol", IETF RFC 1541, Oct. 1993
[26] Finlayson, Mann, Mogul, Theimer, "A Reverse Address Resolution Protocol", IETF RFC 903, June, 1984
[27] Hodges, J. and R. Morgan, "Lightweight Directory Access Protocol (v3): Technical Specification", RFC 3377, September 2002
[28] http://archive.dante.net/np/ds/osi.html
[29] http://www.openldap.org
[30] Welchenbach, Michael, "Cryptography in C and C++", Springer-Verlag New York Inc, 2001
[31] ANSI X9.31-1998, Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry, 1998
[32] Joan Daemen and Vincent Rijmen, "AES submission document on Rijndael", June 1998.
[33] http://www.freeswan.org
[34] http://www.strongsec.com/freeswan
[35] S. Kent, K. Seo, "Security Architecture for the Internet Protocol", draft-ietf-ipsec-rfc2401bis-06.txt, March, 2005
[36] P. Vixie, "Extension Mechanisms for DNS (EDNS0)", IETF RFC 2671, Aug. 1999
[37] http://www.isc.org

저자소개

윤 성 중(Sung-Jung Yoon)

2003년 목포해양대학교 해양전자·통신공학부 (공학사)
2006년 목포해양대학교 해양전자통신과 (공학석사)

※관심분야 : 정보통신시스템, IPv6

김 건 웅(Geonung Kim)

1990년 고려대학교 전자전산공학과 (공학사)
1994년 고려대학교 대학원 전자공학과 (공학석사)
1998년 고려대학교 대학원 전자공학과 (공학박사)
1999년 ~ 목포해양대학교 해양전자·통신공학부 부교수
※관심분야 : 네트워크 프로토콜, 망관리시스템, 지능망, 정보망, IPv6