LDAP를 이용한 보안키 관리 시스템 구현

윤성중, 김건웅

목포해양대학교 해양전자·통신공학부

Implementation of Security Key Management System by LDAP

Yoon Sung-Jung, Geonung Kim

Division of Electronic & Communication Engineering, Mokpo National Maritime University

e-mail: {zeroyoon, kgu}@mmu.ac.kr

요 약

대칭형 암호 알고리즘의 단점을 보안하기 생겨난 비대칭 암호 알고리즘의 대표적인 알고리즘이 RSA 알고리즘이다. 본 논문에서는 RSA 알고리즘의 공개키와 비밀키를 사용자가 생성하고, Web을 통해 등록할 수 있는 키 관리 시스템을 구현하고, 이를 이용하여 사용자 인증, 암호화, 복호화 하는 과정을 보인다.

ABSTRACT

키 워 드

IPsec, IKE, 공개키, 키관리

Ⅰ. 서 론

인터넷 사용의 급증으로 인터넷 관련 시스템과 기술이 비약적인 발전을 해왔으나, 정보보호와 통신망 관리가 소홀한 점으로 지적되고 있다. 또한 통신망을 관리하는 측면에서 통신 사업자의 주요 정책과 동적으로 변화되는 통신망 상태에 따라 효율적으로 통신망을 관리하는 방안이 연구되고 있다. 그중 통신망의 안전한 통신을 지원하기 위하여 IP 계층에서는 IPsec(IP Security)이 제시되고 있다. IPsec은 두 호스트들이나 보안 게이트웨어들 사이, 또는 호스트와 보안 게이트웨어 사이의 경로를 보호하기 위하여 AH(Authentication Header)와 ESP(Encapsulation Security Payload) 두 프로토콜을 이용한다. 또한 양단간의 신뢰성 있는 암호화 키 교환을 위해 IKE(Internet Key Exchange)가 사용된다.

이러한 IPSec 보안 구조가 구현되기 위해서는 IPSec 키를 저장하고 검색, 수정, 삭제할 수 있는 키 관리 시스템이 반드시 필요하다. 본 논문에서는 디렉토리(directory) 서비스를 이용하여 공개키(public key)를 저장, 탐색하여 메시지를 암호화/복호화 기능을 구현한 시스템의 결과를 소개한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 IPsec과 보안키 관리 기능과의 관계에 대해 정리하고, 다음 3장에서는 LDAP(light- weight directory access protocol)을 이용한 보안키 관리 시스템의 구현을 설명하고, 4장에서 결론을 맺는다.

Ⅱ. IPsec과 보안키 관리 기능과의 관계

인터넷 보안 프로토콜은 AH(Authentication Header)와 ESP(Encapsulating Security Payload) 확장헤더를 기반으로 한다. AH 프로토콜은 IP 데이터그램에 대해 무결성(Integrity), 인증(Data Origin Authentication), 재전송 공격(Replay Attack) 방지 등과 같은 보안 서비스를 제공하기 위해 사용되며 MD5, SHA-1 등의 알고리즘을 사용한다. ESP 프로토콜은 IP 데이터그램에 3DES, AES 등의 알고리즘을 적용하여 기밀성(Confidentiality), 무결성(Integrity), 인증(Data Origin Authentication), 재전송 공격(Replay

Attack) 방지 등과 같은 보안 서비스를 제공하기 위해 사용된다.

IPsec을 통한 보안 기능에 있어서 양단간의 신뢰성 있는 암호화 키의 교환이 중요하며, 이를 위한 키 교환 시스템으로서 IKE가 사용된다. IKE는 두 종단간의 상호 인증을 제공하며 AH와 ESP를 위한 키를 생성하고 관리한다. IKE에서는 2단계에 걸쳐 SA를 설정하는데, 첫 단계에서는 IKE 자체의 SA를 설정하고, 두 번째 단계에서는 IPsec SA를 설정한다. 첫 단계에서는 IPsec SA 설정에 필요한 암호화 매개변수를 합의하고 공유 비밀키를 생성한다. 이때 쌍방간의 인증방법으로 전자서명, 공개키 암호화, 수정된 공개키 암호화 등을 활용할 수 있다.

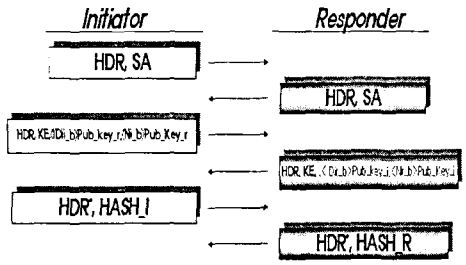

그림 1. 공개키 암호화를 이용한 메인 모드

그림 1에서 보는 것과 같이 1단계에서 6개의 메시지를 통해 인증을 수행하고 IKE SA를 설정한다. 여기서 세 번째 메시지와 네 번째 메시지에서 각 각 상대방의 공개키를 이용하여 신원(IDii, IDir)과 nonce(Ni, Nr)를 암호화하여 전송하고, 그것을 자신의 비밀키(private key)로 복호화 하여 다음 처리를 함으로써 인증이 이루어진다. 결국 이를 이용하기 위해서 각자 신뢰할 수 있는 제 3자를 통해 상대방의 공개키를 알아내야 한다.

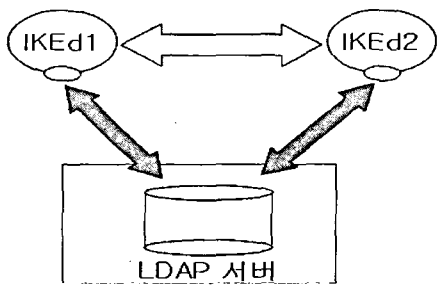

그림 2. 공개키 관리 시스템

본 논문에서는 그림 2에서 보는 것과 같이 제3자 인증 기관을 LDAP 서버를 사용하여 IKEd1에서 public 키를 생성하여 LDAP 서버에 등록을 하고, IKEd2에서 LDAP 서버에서 IKEd1의 public 키를 가져온 후 메시지를 암호화하여 IKEd1에게 보내고, 이를 IKEd1이 비밀키를 이용하여 복호화하는 기능을 구현한 예를 보여준다[1][2][3].

III. LDAP을 이용한 보안키 관리 시스템 구현

보안키 관리 시스템 프로그램은 공개키와 비밀키를 생성하고, 공개키를 LDAP 서버에 저장한 후, 사용자의 요청이 있을 때 암호문을 비밀키를 사용하여 복호화 하여 반환해주는 create_key 부분과 사용자에게 입력받은 메시지를 공개키를 LDAP 서버에서 가져와서 메시지를 암호화를 하여 반환해 주는 crypto부분으로 구성하였다[6][7][8][9][10].

그림 3은 create_key부분의 알고리즘이다.

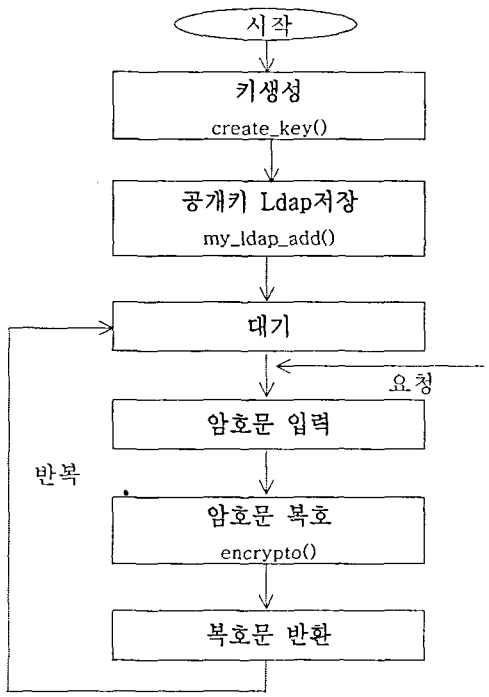

그림 3. create_key 서버 알고리즘

다음은 LDAP 서버에 공개키 값을 저장하는 소스이다.

```
int my_ldap_add(char *buf,int size)
{
    .....
    struct berval *test[2];
    struct berval tttt;

    test[0] = &tttt;
    test[0]->bv_len = size;
```

```
        test[0]->bv_val = buf;
        test[1] = NULL;
    .....

        mods = ( LDAPMod ** ) malloc(( NMODS + 1 ) *
sizeof( LDAPMod * ));
        if ( mods == NULL ) {
                    }
    for ( i = 0; i < NMODS; i++ ) {
        if (( mods[ i ] = ( LDAPMod * ) malloc( sizeof(
LDAPMod ))) == NULL ) {
                        }
        }
.....
         mods[ 8 ]->mod_op = LDAP_MOD_ADD |
LDAP_MOD_BVALUES;
        mods[ 8 ]->mod_type = "RSAKey";
        mods[8]->mod_bvalues = test;

        mods[9]=NULL;

    }
```

LDAP 서버에 공개키 값을 저장할 때 공개키 값은 바이너리 값이므로 mods 구조체의 mod_op에 바이너리 값을 넣기 위하여 LDAP_MOD_BVALUES 추가하고, struct berval 의 bv_val에 키 값을 저장하여 LDAP 서버에 저장하였다.

그림 4는 crypto 부분의 알고리즘이다.

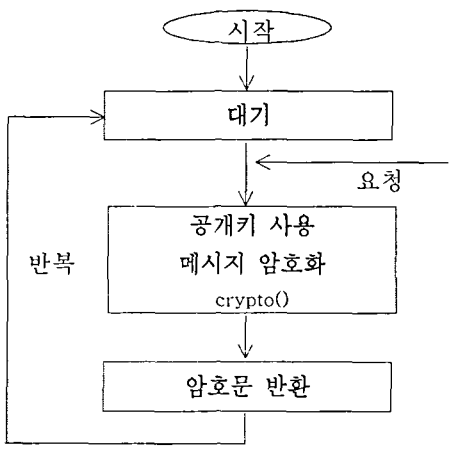

그림 4. crypto서버 알고리즘

LDAP 서버로부터 공개키 값을 검색을 할 때 키의 값이 바이너리 값이므로 다음과 같이 ldap_get_values_len을 사용하여 키 값을 검색해서 저장한다.

```
ber_valu=ldap_get_values_len(ld,entry,attr);
write_file(ber_valu[i-1]->bv_val,ber_valu[i-1]->bv_len);
```

LDAP 서버에 공캐키를 사용하기 위해 스키마를 작성을 할 때 OID 값을 바이너리 값으로 설정하여서 다음과 같이 스키마를 작성했다.

```
attributetype ( 1.1.2.1.6 NAME 'PubKey'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.5 )
```

다음은 앞에서 설명한 함수를 구현한, 결과 화면들이다. 그림 5는 create_key부분을 실행한 화면으로 공개키와 비밀키를 생성하여 파일에 저장하고 공개키 값을 LDAP 서버에 저장한 복호화 요청이 있을 때까지 대기하는 과정을 보여준다.

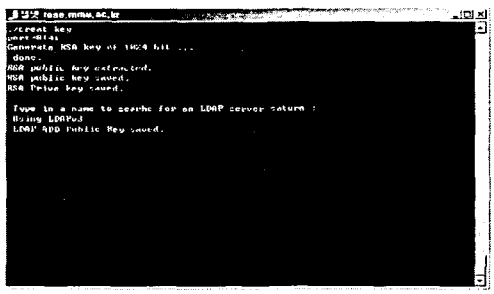

그림 5. create_key 서버 실행화면

다음 그림 6은 create_key부분에서 대기 중 복호 요청으로 복호화를 실행한 결과 화면이다.

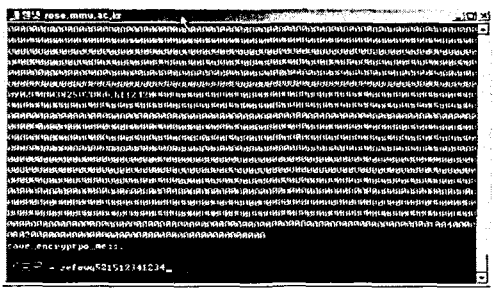

그림 6. create_key서버에서 복호화 결과

그림 7은 crypto 부분을 실행한 화면으로 대기 중 암호화 요청을 받아 메시지를 받아서 암호화하여 반환하였다.

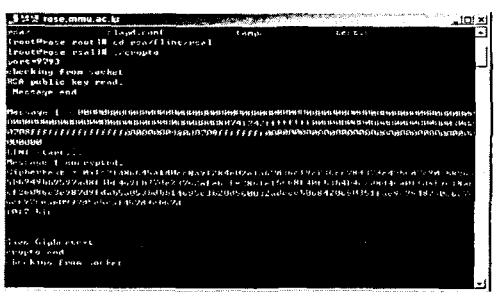

그림 7. crypto 서버에서 암호화

그림 8은 Web에서 사용자가 LDAP에 저장되어 있는 공개키를 선택하여 사용하는 화면이다.

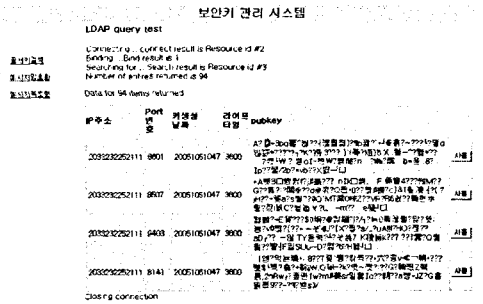

그림 8. 공개키 탐색 결과

그림 9는 Web에서 평문인 메시지를 서버로부터 암호화하여 결과인 암호문을 보여주는 화면이다.

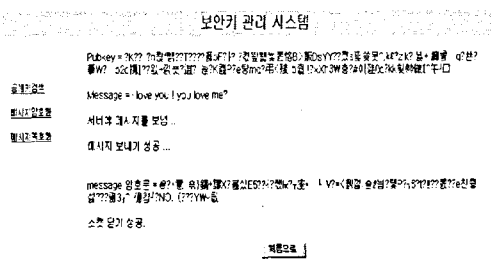

그림 9. 메시지 암호화 결과 화면

그림 10은 Web에서 암호문을 복호화하여 결과 메시지를 보여주는 화면이다.

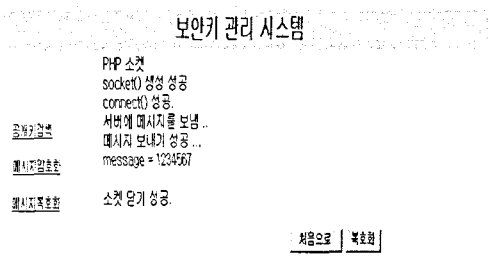

그림 10. 복호화 결과 화면

V. 결론

본 논문은 IPsec의 보안키를 RSA 알고리즘을 사용하여 생성하고 공개키를 LDAP 서버에 저장하여 관리하고 Web에서 사용자에게 메시지를 입력 받아서 암호화와 복호화를 구현한 샘플 프로그램을 작성한 결과를 소개하였다. 앞으로 키가 생성된 후 유효기간을 설정하여, 키의 유효 기간이 만료 되면 키를 폐기하거나 갱신을 할 수 있는 더욱 발전된 보안키 관리 시스템을 설계하고 구현할 예정이다.

참고문헌

[1] "The Internet Key Exchange(IKE)", RFC 2409, November 1998.
[2] "IP Authentication Header", RFC 2402, November 1998.
[3] "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.
[4] "Security Architecture for the Internet Protocol", RFC 2401, Novermber 1998.
[5] "Lightweight Directory Access Protocol (v3):Attribute Syntax Definitions", RFC 2252, December 1997
[6] "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997
[7] Michael Welschenbach ,"Cryptography in C and C++", Apress, 2001
[8] http://www.openldap.org/
[9] http://www.php.net/
[10] W.Richard stevens "UNIX NETWRK PROGRAMMING" 1999