안전한 통신 서비스를 제공하는 향상된 SSL(Secure Socket Layer) 기반 정보보호 시스템의 설계

준회원 유 성 진*, 김 성 열*, 정회원 정 일 용*

The Design Of Improved Information Security System based on SSL(Secure Socket Layer) For Providing Secure Communication Services

Seong-Jin You*, Seong-Yeol Kim* *Associate Members*, Il-Yong Chung* *Regular Member*

요 약

SSL 프로토콜은 최근 네트워크 시스템에서 광범위 하게 사용되는 메카니즘중의 하나이다. 본 논문에서는 SSL 프로토콜을 기반으로 향상된 정보보호 메커니즘을 설계하였다. 여기에서는 네가지 중요한 정보보호 서비스를 제공한다. 첫번째는 SSL에서 제공하는 인증서를 이용한 인증서비스이고, 두번째는 DES 암호 알고리즘을 이용한 메시지 암호화 서비스이다. 세번째는 해쉬 알고리즘을 이용한 메시지의 무결성 서비스이고 네번째는 부인봉쇄 정보 서비스이다.

그러므로, 정보는 본 논문에서 특별히 설계된 부인봉쇄 서비스를 이용한 정보보호 메커니즘을 통해 안전하게 전송된다.

ABSTRACT

The SSL(Secure Socket Layer) protocol is one of the mechanism widely used in the recent network system. The improved information security mechanism based on the SSL is designed in this paper. There are important four information security services. The first is the authentication service using the Certificate offered from the SSL(Secure Socket Layer), the second is the message confidentiality service using the DES encryption algorithm, the third is the message integrity service using Hash function, and the fourth is Non-repudiation service.

Therefore, information could be transferred securely under the information security mechanism including Non-repudiation service especially designed in this paper.

I. 서 론

정보통신 및 전자기술의 발전[1]에 힘입어 현대사회는 과거 산업사회로부터 정보사회로 급속하게 진전되고 있다. 그 결과 과거에는 상상조차 할 수 없었던 정보서비스[4]의 편리함과 효율성 그리고 신속성을 제공받고 있으며 이러한 결과는 컴퓨터 네트워크[6]를 통하여 모든 사람에게 전달 가능하게 됨으로서 누구나 문명의 이기를 공유할 수 있게 되었다.

그러나, 인터넷에 접속되는 기관의 네트워크와 서버는 인터넷 사용자가 접근이 가능하여야 함으로, 대부분이 열린상태로 제공되어야 하며 이러한 공개성은 인터넷 불법침입자[5](Intruder, Hacker, Cracker) 등에 의해 보안상의 문제를 야기시킬 잠재적 위험

* 조선대학교 컴퓨터공학부 (iyc@mina.chosun.ac.kr)
논문번호: 00095-0317 접수일자: 2000년 3월 17일
※ 본 논문은 정보통신부지원 "리눅스시스템 보안연구센터"의 연구비를 지원받아 수행되었음

성을 항상 내포하고 있다고 할 수 있다.

네트워크상의 이러한 문제점들을 해결하기 위해 여러 가지 정보보호 서비스들이 제공된다. 즉, 정보보호의 중요성 및 인증, 암호화, 무결성, 부인봉쇄 등 여러 가지 정보보호 서비스들을 이용하여 이러한 정보보호 서비스를 제공하기 위한 방법론으로 여러 가지 정보보호 메커니즘들을 살펴본다. 그리고 응용프로그램 개발 도구에서도 본 연구에 핵심이 되는 SSL[15,16]에서 제공하는 정보보호 메커니즘을 살펴보고 이것을 개선하여 향상된 SSL의 정보 보호 메커니즘을 설계한다.

Ⅱ 정보보호 메커니즘의 분석

정보보호 서비스[4]는 통상 물리적인 문서와 관련된 기능의 유형을 갖는 것으로 생각할 수 있다. 통신망에서 제공되는 기본적인 정보보호 서비스는 기밀성, 인증, 무결성, 접근제어, 부인봉쇄, 가용성이 있다.

네트워크 보안 기술에는 SET, IPsec, TLS, PCT, SSH, SSL등이 있다. 이를 간단히 설명하면, SET (Secure Electronic Transaction)은 인터넷과 같은 공개된 네트워크(Open Network)상에서의 신용카드 지불을 위한 Protocol이라 할 수 있다. IPsec은 데이터 인증, 패킷 무결성, 데이터 신뢰성, 응답보호, 암호화 키 자동 관리 및 SA(Security Association) 등을 규정하고 있다. TLS는 두 개의 통신응용프로그램 사이에서 개인의 정보보호와 데이터의 무결성을 제공하기 위해 만들어졌고, TLS Record 프로토콜과 TLS Hand _shake 프로토콜로 구성되어 있고, PCT(Private Communication Technology)는 두 통신 응용간에 비밀보장과 적어도 하나(주로 서버)에 대한 인증 서비스를 제공하기 위해 설계되었다. SSH (Secure Shell)는 rlogin, rsh, rcp, rdist와 telnet을 대치하기 위해 설계된 방식으로 안전하지 않은 네트워크 상에서 안전한 로그인 세션과 통신을 제공하는 응용이라고 할 수 있고, SSL(Secure Socket layer)은 Netscape사에서 개발된 프로토콜로 특정 응용을 위한 보안 프로토콜이 아닌 일반적인 인터넷 보안 프로토콜로 사용될 수 있다.

Ⅲ SSL(Secure Socket Layer) 분석

1. SSL Protocol의 소개

SSL은 Layer를 기반으로 하는 protocol이다. 각 layer의 message는 length, description, content field 들을 포함할 수 있다. SSL이 전송해야할 message를 가질 때, SSL은 그것을 제어 가능한 크기의 block으로 나누고, 선택적으로 data압축과정을 거친 후, 메시지 인증 코드(Message Authentication Code)를 추가하고 암호화를 한 후, 결과를 전송한다. data가 도착하게 되면 복호화를 하고, 메시지 인증 코드를 verify한 후, 압축된 data를 원상태로 만들고, block을 재배열하여 얻어진 message를 상위레벨로 넘겨주게 된다.

2. SSL의 구조

SSL은 크게 Record layer와 Handshake layer로 나누어 볼 수 있다. Record는 SSL에서 data 전송시 가장 기본이 되는 단위이다.

Handshake Layer는 암호화 방법이나 열쇠의 결정 및 협상을 담당한다. Handshake Layer는 크게 Change Cipher Spec Protocol, Alert protocol, Handshake protocol로 나누어진다. Record Layer는 change cipher spec, alert, handshake, application data의 네 가지 content type을 가질 수 있다. Record Layer에는 데이터에 대한 정보와 실제 데이터가 들어가게 된다. 구조를 간단히 살펴보면 SSL Header와 SSL Body로 나누어진다. SSL Header에는 Message Type, 버전, 데이터 길이를 포함하고 있고 [그림1]과 같이 구성된다. 버전정보에 관한 구조체에는 SSL이 지원할 수 있는 SSL 버전에 관한 정보를 클라이언트와 서버가 서로 교환을 하게 되고, 데이터의 Type에 관한 구조체는 SSL에서 보내는 메시지가 어떤 Type의 메시지인가를 판별할 수 있다.

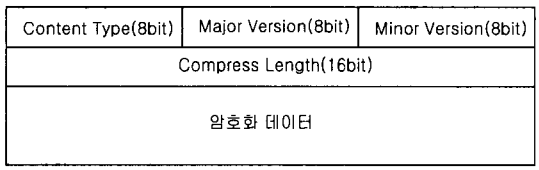


그림 1. SSL 데이터의 구조

SSL Body에는 [그림 2]와 같이 단편화, 압축, MAC 생성, 암호화과정을 거치게 된다. Fragmen _tation이란 Record layer는 214 또는 그이하의 byte의 정보블럭으로 단편화한는 것을 의미하고, Record compression and decompression에서는 레코드들은 현재 세션상태에서 정의된 압축 알고리즘

을 사용해 선택적으로 압축된다. Record pay _load protection and the CipherSpec은 모든 레코드들은 암호화와 현재 CipherSpec에서 정의한 MAC 알고리즘을 사용해 보호된다는 것을 의미한다. 버전정보에 관한 구조체에는 SSL이 지원할 수 있는 SSL 버전에 관한 정보를 클라이언트와 서버가 서로 교환을 하게 되고, 데이터의 Type에 관한 구조체는 SSL에서 보내는 메시지가 어떤 Type의 메시지인가를 판별할 수 있다. 레코드 layer에서 데이터를 스트림 암호기법을 이용한 암호화와 블록 암호기법을 이용한 암호화를 할때의 구조를 살펴보면 아래와 같다.

1.1 data를 스트림암호 기법을 이용하여 암호화하고자 할 때 사용

```
stream-ciphered struct{
opaque MAC[CipherSpec.hash_size];
opaque content[SSLCompressed.length];
                } GenericStreamCipher;
```

1.2 data를 블록암호 기법을 이용하여 암호화하고자 할 때 사용

```
block-ciphered struct{
opaque MAC[CipherSpec.hash_size];
opaque content[SSLCompressed.length];
uint8 padding
[GenericBlockCipher.paddi_nglength];
uint8 padding_length;
                } GenericBlockCipher;
```

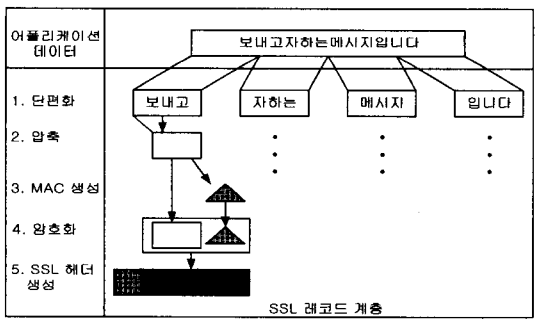


그림 2. SSL 레코드 계층

3. SSL의 정보보호 서비스

SSL은 크게 인증, 암호화, 무결성을 제공하고 있다. SSL에서 제공하는 인증에서는 클라이언트와 서버가 서로 자신의 인증서(Certificate)를 교환한다. 그리고 각각은 자신이 받은 인증서에 기록되어 있는 유효기간, 서명을 확인한다. 클라이언트는 이후에 수행될 암호화와 메시지 인증 코드 (MAC : Message Authentication Code)의 생성에 필요한 난수(비밀키)를 생성하여 이것을 서버의 공개키로 암호화하여 서버에게 전송한다. 서비스를 받는 동안 사용할 암호화 알고리즘과 해쉬함수의 종류를 결정한다. 이 과정에서는 클라이언트는 자신이 제공할 수 있는 암호화 알고리즘의 리스트를 서버에게 제시하고 이들 중에서 하나를 서버가 선택하여 클라이언트에게로 보낸다. 서버나 클라이언트는 자신이 수신한 메시지가 정말로 송신자가 보낼 때의 메시지와 똑같은 것인지를 확인하기 위해서는 키가 있는 암호학적 해쉬함수를 이용한다. 송신자는 메시지를 키를 이용하여 해쉬함수에 입력하여 그 결과인 일정한 길이의 데이터 (이것을 MAC이라고 부른다)를 보내려는 메시지와 함께 보낸다. 그러면 수신자는 수신된 메시지를 키를 이용하여 해쉬함수에 입력하여 그 결과를 자신이 수신한 MAC와 비교한다.

4. SSL에서 제공하지 못하는 서비스

SSL에서는 인증, 암호화, 무결성을 제공하고 있지만 부인봉쇄를 제공하고 있지 않다. 그래서 SSL에서 제공하지 못하는 부인봉쇄 서비스를 설계함으로써 향상된 SSL기반 정보보안 시스템을 설계한다.

Ⅳ SSL기반 향상된 정보보호 메커니즘 설계

1. SSL기반 향상된 정보보호 메커니즘 설계

1.1 Secrets 생성 및 전송

(1). pre_master_secret생성 방법

pre_master_secret는 Client의 SSL버전과 랜덤값으로 구성되어 있다. Client의 버전은 2바이트로 구성되어 있고, 랜덤값은 안전하게 생성된 값으로써 46바이트로 구성되어 있다.

```
struct {
        ProtocolVersion client_version;
        opaque random[46];
        }PreMasterSecret;
```

(2). pre_master_secret 전송

pre_master_secret(48byte)를 서버의 공개키로 암호화 시켜 서버에게로 전달한다.

```
struct {
    public-key-encrypted        PreMasterSecret
pre_master_secret;
```

}EncryptedPreMasterSecret;

(3). master_secret(48byte) 생성.

Client로부터 받은 pre_master_secret를 이용하여 master_secret생성한다. master_secret는 pre_master_secret, ClientHello.random, Se _rverHello.random로 구성되어 있다.

1) ClientHello.random

Random은 타임스탬프와 안전하게 생성된 랜덤값으로 구성되어 있다.

```
struct {
        uint32 gmt_unix_time;
        opaque random_bytes[28];
            } Random;
```

ClientHello는 client_version, random, sess _ion_id, cipher_suites, compression_metho _ds로 구성되어 있다. client_version는 클라이언트의 SSL 버전을 나타내고, session_id는 sess _ion이 계속 진행되는 동안 사용될 식별자이고, cipher_suites는 암호화 방법, 암호화 알고리즘, 암호화 스펙에 관한 정보를 가지고 있고, Compre _ssion_Method는 압축방법에 관한 정보를 가지고 있다. Client_Hello는 다음과 같이 구성되어 있다.

```
struct {
          ProtocolVersion client_version;
           Random random;
           SessionID session_id;
           CipherSuite cipher_suites<2..2^16-1>;
           CompressionMethod
           compression_methods<1..2^8-1>;
            } ClientHello;
```

ClientHello의 구조중 CipherSuite에 정의된 값중 하나를 선택하여 서버가 클라이언트에게 보내 줌으로써 암호화 알고리즘과 암호화 방법을 교환하게 된다.

2) ServerHello.random

Random은 클라이언트와 같은 구조의 타임스탬프와 안전하게 생성된 랜덤값으로 구성되어 있다.

```
struct {
          ProtocolVersion server_version;
           Random random;
           SessionID session_id;
           CipherSuite cipher_suites<2..2^16-1>;
           CompressionMethod
           compression_methods<1..2^8-1>;
            } ServerHello;
```

ServerHello는 server_version, random, se _ssion_id, cipher_suites, compression_meth _ods로 구성되어 있다. ServerHello는 client_ version, random, session_id, cipher_suites, compression_ methods로 구성되어 있다. client _version는 클라이언트의 SSL 버전을 나타내고, session_id는 session이 계속 진행되는 동안 사용될 식별자이고, cipher_suites는 암호화 방법, 암호화 알고리즘, 암호화 스펙에 관한 정보를 가지고 있고, Compression_ Method 는 압축방법에 관한 정보를 가지고 있다. Cipher_Suite는 Client에서 보낸 ClientHello를 참조하여 서버가 하나를 선택하여 다시 Client에게 보내준다.

3) master_secret

master_secret는 pre_master_secret, Client _Hello.random, ServerHello.random으로 구성되어 있다. master_secret생성 방법은 MD5 해쉬 함수를 이용하여 아래와 같이 생성한다.

```
master_secret =
MD5(pre_master_secret + SHA('A' +
pre_master_secret + ClientHello.random +
ServerHello.random)) +
MD5(pre_master_secret +
SHA('BB' + pre_master_secret +
ClientHello.random +
ServerHello.random)) +
MD5(pre_master_secret +
SHA('CCC' + pre_master_secret +
ClientHello.random +
ServerHello.random));
```

(4). 키 생성

클라이언트와 서버 각각 client-Mac-secret, server-Mac-secret, client-write-key, server -write-key, client-write-IV, server-write-IV 값을 생성한다.

1) key_block 생성방법

```
key_block =
MD5(master_secret +
SHA('A' + master_secret +
ServerHello.random +
ClientHello.random)) +
MD5(master_secret +
SHA('BB' + master_secret +
```

ServerHello.random +
ClientHello.random)) +
MD5(master_secret +
SHA('CCC' + master_secret +
ServerHello.random +
ClientHello.random)) + [...];

2) MAC에 사용할 키 생성

client-Mac-secret은 client의 Mac 값을 계산하기 위한 key값이 되고, server-Mac-secret는 server의 Mac값을 계산하기 위한 key값이 된다.

client_MAC_secret = key_block[0..15]
server_MAC_secret = key_block[16..31]

3) 공통키 생성(통신키 생성)

client-write-key는 client가 data를 암호화할 때 사용하는 공통키가 된다. server-write-key는 server가 data를 암호화할 때 사용하는 공통키가 된다.

client_write_key = key_block[32..36]
server_write_key = key_block[37..41]
final_client_write_key =
MD5(client_write_key +
ClientHello.random +
ServerHello.random)[0..15];
final_server_write_key =
MD5(server_write_key +
ServerHello.random +
ClientHello.random)[0..15];

4) 초기 벡터값

client-write-IV는 client의 초기 벡터 값이 되고, server-write-IV는 server의 초기 벡터 값이 된다.

client_write_IV =
MD5(ClientHello.random +
ServerHello.random)[0..7];
server_write_IV =
MD5(ServerHello.random +
ClientHello.random)[0..7];

클라이언트는 pre_master_secret생성 하고 이것을 서버의 공개키로 암호화하여 서버에게 보내준다. pre_master_secret는 Client의 SSL버전(2 바이트)과 랜덤값(46바이트)으로 구성되고, 랜덤값은 안전하게 생성된 값이다. 그다음은 Client로부터 받은 pre_master_secret를 이용하여 master _secret생성한다. 그리고 마지막으로 실제 사용할 키를 생성한다. 여기에는 client-Mac-secret, ser_ver-Mac-secret, client-write-key, server -write-key, client-write-IV, server-write- IV 값이 생성된다.

client-Mac-secret은 client의 Mac 값을 계산하기 위한 key값이 되고, server-Mac-secret는 server의 Mac값을 계산하기 위한 key값이 된다.

client-write-key는 client가 data를 암호화할 때 사용하는 공통키가 된다. server-write-key는 server가 data를 암호화할 때 사용하는 공통키가 된다.

client-write-IV는 client의 초기 벡터 값이 되고, server-write-IV는 server의 초기 벡터 값이 된다.

1.2 향상된 SSL정보보호 메커니즘

본 논문에서 제안한 향상된 SSL정보보호 메커니즘을 설계한다. 여기에서는 클라이언트와 서버간의 상호인증, 메시지의 무결성, 비밀성, 부인봉쇄 서비스의 실현과정을 설명 하고자 한다.

먼저 상호인증은 인증기관에서 인증서를 발급 받아서 클라이언트와 서버가 서로 교환을 하기 때문에 상호인증이 이루어진다. 그 다음 클라이언트는 서버로부터 받은 공개키를 이용하여 MAC Secret, 공통키와 초기벡터 값에 사용될 Secret 값을 서버의 공개키로 암호화 시켜서 서버에게 전달한다. 이 과정으로 서로 메시지를 암호화시킬 수 있는 공통키를 교환한다. 그리고 해쉬함수를 이용하여 MAC을 생성한다. 이것은 메시지의 무결성을 검사하기 위하여 사용된다. 마지막으로 부인봉쇄 정보를 생성하는데 이것은 사용자의 uid, 호스트 이름, IP 주소로 이루어져 있다. 이렇게 생성된 MAC과 부인봉쇄 정보를 메시지와 함께 통신키로 암호화 시켜서 클라이언트는 서버에게 전달하고, 서버는 클라이언트에게 같은 방법으로 메시지를 전달함으로써 향상된 SSL정보보호 메커니즘을 실현 할 수 있다.

(1) Notation

C : client S : server
SK_i : 노드 i의 비밀키 PK_i : 노드 i의 공개키
Ver_i : 노드 i의 버전 CK_i : 노드 i의 통신키
$Suite_i$: 노드 i에서 지원할 수 있는 암호화 알고리즘 리스트
Message : Message Header와 Message Content로 구성됨
N_i : Random number와 Time stamp로 구성된 구조체
Message Header : 사용자에 대한 정보 저장
Message Content : 실제 데이터 부분
MAC : Message Authenticate Code
P : 사용자 P가 어떤 검증을 하기 위한 계산단계

P → Q:M : 사용자 P가 사용자 Q에게 메시지 M을 보내는 교환단계
$(p, PK_p)Cert_i$: 인증기관 i에서 발행한 p의 인증서
pre_master_secret : SSL Version과 Random number로 구성된 구조체
Pad : 길이를 맞추기 위한 Pad
Msgs : ClientHello의 메시지부터 자기자신 메시지 전단계의 모든 메시지
M : A || B : M은 A와 B로 구성되어 있다.
{M}A : A를 암호화키로 하여 메시지 M을 암호화
Master : pre_master_secret를 계산하여 생성된 Master키
(M)Hash : 메시지 M을 Hash알고리즘으로 수행

(2) 알고리즘
1) Connect
C -> S : C, Ver_c, $Suite_c$, N_c
/* $Suite_c$는 클라이언트에서 지원할 수 있는 암호화 알고리즘 리스트/*
S -> C : Ver_s, $Suite_s$, N_s
/* 서버의 버전, 암호화리스트, 랜덤값과 타임 스탬프로된 구조체 전달 */

2) Certificate or KeyExchange
S -> C : {S, PK_s}$Cert_{CA}$
/* 인증기관으로부터 서명된 인증서를 PEM방 식으로 클라이언트에게 전달 */
C -> S : {C, PK_c}$Cert_{CA}$, {premaster_$secret_c$}PK_s
/* pre_master_secret(48byte)를 서버의 공 개키로 암호화시켜 서버에게로 전달 */

3) ClientVerify
C -> S : {(Master + Pad2 + (Msgs + Master)Hash + Pad1)Hash}SK_c
Master : N_c || N_s || $pre\_master\_secret_c$
Msgs : Connect || KeyExchange

4) Finished
S -> C : {(Master + Pad2 + (Msgs + S + Master + Pad1)Hash) Hash}Master
Master : N_c || N_s || $pre\_master\_secret_c$
Msgs : Connect || KeyExchange || ClientVerify
C -> S : {(Master + Pad2 +(Msgs + C + Master + Pad1)Hash) Hash}Master
Master : N_c || N_s || $pre\_master\_secret_c$
Msgs : Connect

5) Data Exchange
C -> S : {$Message_c$}CK_c
/* 통신키로 메시지를 암호화하여 전달 */
$Message_c$: Message $Header_c$ || Message $content_c$
Message $Header_c$: $Type_c$ || $MajorVer_c$ || $MinorVer_c$ || $CompressLen_c$
Message $content_c$: {UID_c, UID_s, IP_c, IP_s, MAC_c}SK_c || $Data_c$
S -> C : {$Message_s$}CK_s
/* MAC 검증 : MAC_c == MAC_s */

1.3 제안된 알고리즘 분석

클라이언트와 서버는 각각 자신이 가지고 있는 통신키와 DES 알고리즘을 이용하여 Message를 암호화하여 Message의 비밀성을 보장하고, 신뢰된 인증기관에서 인증서를 발급 받아서 클라이언트와 서버가 안전하게 인증서를 서로 교환함으로써, 클라이언트와 서버는 서로 신뢰 할수 있게 된다. 그리고 클라이언트는 해쉬함수를 이용하여 MAC 생성한후에 서버에게게 전달하고, 서버는 클라이언트로 부터 받은 MAC과 자신이 생성한 MAC을 비교하여 같은 값이 나오는지 확인한다. 같은 값일 경우 메시지의 무결성이 검증 되는 것이고 그러지 않을 경우는 위조 또는 변조되었음을 알 수 있다.

본 논문에서 특별히 제안된 알고리즘은 부인봉쇄를 추가함으로써 분쟁을 해결 할 수가 있게 되었다. 즉, 클라이언트와 서버는 부인 봉쇄 정보를 생성하여 각각의 개인키로 암호화를 하고 부인 봉쇄정보를 메시지와 함께 다시 통신키로 암호화를 시켜서 서로 교환을 하게된다. 부인봉쇄 정보로는 사용자의 uid, 호스트 이름, IP 주소가 있다. 이 정보를 서로의 개인키로 암호화를 하기때문에 디지탈서명을 하는것과 다름없다. 기존의 SSL 3.0 에서는 부인봉쇄를 지원하지 못하기 때문에 클라이언트와 서버사이에 분쟁이 일어나도 해결을 하지 못한다. 그러나, 제안된 알고리즘에서는 클라이언트와 서버는 각각 부인봉쇄 정보를 생성하여 서로 교환함으로써 향후 분쟁이 일어나도 이를 해결 할 수 있게 된다.

V. 결 론

보통신의 급속한 발전과 더불어 대부분의 컴퓨터

들은 Internet등 각종 통신망으로 연결되어 있으며, 이러한 네트워크의 상호 연결을 통해 사람들은 생활에 편리함을 누리고 있다. 통신망에서 보내는 자료는 중요하지 않은 자료도 있지만 전자 상거래나, 인터넷 쇼핑, 홈뱅킹에서 사용되는 비밀이 보장되어야만 하는 중요한 자료가 오고갈 수 있다. 따라서 정보보호시스템의 이용확대와 함께 정보보호 사업이 크게 부각되었다.

정보보호의 중요성이 부각되고, 정보보호 산업이 정보통신산업에서 차지하는 비중이 높아짐에 따라, 본 논문에서는 정보보호의 중요성과 인증, 비밀성, 무결성, 부인봉쇄 등 여러 가지 정보보호 서비스들에 대하여 기술 하였다. 또한 정보보호 서비스를 제공하기 위한 방법론으로 여러 가지 정보보호 메커니즘들을 살펴보고, 이 중에서 본 논문의 핵심이 되는 SSL의 구조 및 정보보호 서비스를 분석하여 지금의 SSL3.0은 제한적인 정보보호 서비스를 제공하고 있다는 것을 밝혀 내었다.

따라서 본 논문은 SSL을 기반으로 한 클라이언트와 서버의 인증 및 클라이언트와 서버사이의 주고받는 메시지의 비밀성 및 무결성 그리고 부인봉쇄를 보장할 수 있는 SSL기반의 향상된 정보보호 메커니즘을 설계하였다. 제시된 정보보호 메커니즘의 장점은 다음과 같다.

첫째, 클라이언트와 서버는 각각 인증서를 교환함으로써 상호 인증을 할 수 있다.

둘째, 해쉬함수를 이용하여 메시지의 Digest를 생성함으로서 전송되는 데이터의 무결성을 보장할 수 있다.

셋째, 클라이언트와 서버간에 전송되는 메시지를 관용알고리즘인 DES를 이용하여 암호화함으로서 메시지의 비밀성을 보장한다.

넷째, 기존의 SSL3.0에서 지원하지 않는 부인봉쇄를 본 논문에서는 지원 가능하다.

본 연구에서는 정보보호 서비스 중에서 비밀성, 무결성, 인증 및 부인봉쇄 서비스가 가능 하도록 설계하였으며, 향후의 연구방향으로는 제시한 정보보호 메커니즘을 Web 환경에서 지원할 수 있는 시스템을 구현하고자 한다.

참 고 문 헌

[1] 정태성, "안전한 데이터 전송을 위한 개선된 RPC기반 정보보호 메커니즘의 구현", *석사학위 논문*, 1997.

[2] 김미선 · 고영한 · 김도윤 · 서재현, "권한 관련성을 지원하는 망기반 접근제어", *정보과학회지* 제 26권 1호, pp.494-496, 1999.

[3] 이임영 · 박춘식, "암호기법", *정보과학회지* 제 15권 4호, pp.14-20, 1997.

[4] William Stallings, *통신망 정보보호*, 그린출판사, 1996.

[5] William Stallings, *인터넷 보안*, 도서출판비앤씨, 1996.

[6] 김화종, *컴퓨터 네트웍 프로그래밍*, 홍릉과학출판사,1997.

[7] 김건우 · 장희진 · 박보석 · 김상욱, "대규모 망에서의 계층적 보안 유지" *정보과학회지* 제 26권 1호, pp.506~508, 1999.

[8] 조은경 · 박정수 · 강신각 · 박성열, "SecWeb : S-HTTP 기반의 안전한 웹 시스템 개발" *정보처리논문지* 제 5권 12호, pp.3165~3175, 1998.

[9] 강창구 · 박진호 · 최용락, "통합 정보 모델을 이용한 접근제어 메커니즘 설계 및 구현" *정보처리논문지* 제 4권 9호, pp.2354~2365, 1997.

[10] 나선희 · 장승주, "Network Packet 관리 및 보안 도구 개발" *정보과학회지* 제 26권 1호, pp.485~487, 1999.

[11] 김윤기 · 이계영 · 김규식, "확장된 네트웍 보안 모니터" *정보과학회지* 제 26권 1호, pp.497~499, 1999.

[12] BRUCE SCHNEIER, *APPLIED CRYP _TOGRAPHY* : Published by Jone Wiley & Sons, Inc., pp.234~560, 1996.

[13] JENNIFER SEBERRY · JOSEF, *PIEP _RZYK CRYPTOGRAPH* : Prentice Hall of Australia Pty Ltd, pp.1~60, 1989.

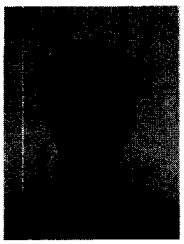
[14] 김철, *암호학의 이해*, (주)영풍문고, pp.55~204, 1996.10.

[15] David Wagner · Bruce Schneier "Analy _sis of the SSL 3.0 Protocol" The Second USENIX Workshop on Electro _nic Commerce Proceedings USENIX0 Press : pp.29~40, 1996.

[16] http://home.netscape.com/eng/ssl3/3-SPEC. HTM, SSL 3.0 SPECIFICATION, 1996.

[17] http://home.netscape.com/eng/ssl3/4- appn.ps, SSL 3.0 APPENDIX, 1996.

유 성 진(Seong-Jin You) 준회원



1998년 2월 : 조선대학교
전자계산학과(이학사)
2000년 2월 : 조선대학교
전자계산학과 석사
2000년 8월 : 조선대학교
전자계산학과 박사2 학기

<주관심 분야> 컴퓨터 네트워크, 분산 시스템, 네트워크 보안

김 성 열(Seong-Yeol Kim) 준회원
한국통신학회 논문지 제 24권 제3호 참조

정 일 용(Il-Yong Chung) 정회원
한국통신학회 논문지 제 24권 제3호 참조