SSL Handshake 프로토콜의 성능 개선

정상곤, 정전대, 신재호, 송유진*
동국대학교 전자공학과, 동국대학교 정보산업학과*
E-mail : hitop4@cakra.dongguk.ac.kr

Performance Improvement of SSL Handshake Protocol

Sang-Gon Jung, Jeondae Cheong, Jaeho Shin, Youjin Song
Dept. of Electronic Eng., Dongguk University

Abstract

In this paper, we analyzed SSL(Secure Socket Layer) protocol. symmentric algorithm is borken by choosen attack and known attack. This protocol prevents SSL from reusing a key and make new key without computational load of public-key algorithm.

Ⅰ. 서론

통신망 기술, 컴퓨터 기술 및 전송 기술의 발달로 인하여 정보통신 분야는 큰 발전을 이루었으며, 이러한 기술의 발달은 인터넷 분야의 눈부신 발전을 가져왔다. 현재 인터넷에 연결된 세계 여러 나라의 네트워크의 수는 수만 개를 넘어섰고, 이들 네트워크에 연결된 호스트의 수는 수백만을 넘어서며 사용자는 수천만 명에 이르렀다.

1990년 Tim Berners-Lee에 의해 처음으로 제안된 WWW은 사용자 인터페이스를 편리하게 설계하고 여러 가지 형태의 자료를 손쉽게 보여주는 브라우저(Browser)의 출현으로 인터넷 사용을 폭발적으로 증가시켰다. Mosaic, Netscape와 같이 편리한 WWW의 사용자 인터페이스는 일반인들의 인터넷 접속을 증가시켰고, 결과적으로 인터넷 전체의 크기를 증가시키고 있다.

본 연구는 한국과학재단지원 핵심전문연구 (981-0928-155-2)의 결과중 일부임

인터넷에는 많은 연구 자료들이 저장되어 있고 이들 중에는 기밀성을 요구하는 자료들도 상당수 있다. 이러한 자료들은 어느 특정한 집단 안에서만 공유하고 그 외의 사용자들에게는 열람을 허용하지 않는 정보의 접근제어가 필요하게 되었다.[6] 또한 편리한 사용자 인터페이스를 가진 WWW 의 등장은 이러한 선별적인 정보의 공유뿐만 아니라 전자 상거래의 등장을 초래하였다. 가상상점 에서는 실세계에서와 같이 상점을 경영하기 위한 건물이 필요없고, 광고비도 절감할 수 있을 뿐만 아니라 점원의 수도 줄일 수 있다. 또한 구매자의 입장에서도 쇼핑의 시간적 제약이나, 공간적 제약이 없다는 점에서 사용이 늘어날 것으로 보이고 있다. 이러한 이유들로 인해 앞으로 인터넷을 통한 전자상거래의 규모는 급속히 늘어날 것이다. 현재 전세계적으로 운영되고 있는 가상상점들은 대부분 구매자의 신용카드를 네트워크를 통한 지불수단으로 이용하고 있다.

인터넷의 근간인 TCP/IP는 본래 개방형 시스템을 기반으로 설계되었기 때문에 네트워크를 통한 정보의 기밀성을 제공하지 못하고 있다.[2] 또한 호스트(host)에 대한 인증도 제공하지 않고 있다. 그래서 만약 공격자가 상점으로 가장한다면 많은 문제를 야기할 수 있다. 그렇기 때문에 네트워크내에서 전송되는 데이터의 기밀성과 무결성을 제공하고 호스트를 인증할 수 있는 서비스가 필요하다.[5] 이러한 서비스를 제공하기 위하여 netscape 사에서는 SSL 프로토콜을 제안하였다.[1] 그러나 이 SSL 프로토콜은 키를 재사용하는 문제점이 있다. 본 논문에서는 SSL 프로토콜을 분석하고 키 재사용에 의한 문제점을 지적하고, 새로운 키를 생성하기 위해서 개선한 handshake protocol을 제안하였다.

본 논문의 구성은 Ⅱ장에서는 SSL을 분석하고 Ⅲ장에

서는 SSL의 한계점을 제시하고, 그것을 개선한 handshake 프로토콜을 제안하고 Ⅳ장에서 결론을 맺는다.

Ⅱ. SSL 분석

SSL은 netscape사에서 개발한 secure socket layer protocol이다. SSL의 기본적인 목적은 정보보호의 중요한 요소인 사생활 보호와 신뢰성을 제공하는 것이다. SSL은 양자간의 안전한 전속을 이루고 상대방의 비밀키 없이 암호요소를 전달하고 새로운 공개키나 데이터 암호방법을 적용시킬 수 있고, optional session을 가져서 효율성을 높이는데 주력하였다. 또한 SSL은 application 프로토콜과 독립적이라는 이점을 가지고 있다.

SSL은 2가지 부분으로 나눌 수 있다. 한 부분은 데이터를 암호화하고 압축을 하여 안전하게 전송하는 SSL Record 프로토콜이고, 다른 하나는 데이터를 전송하기 전에 서버(server)와 클라이언트(client)가 서로를 인증하고 암호 알고리즘과 암호화키 등을 분배하는 SSL handshake 프로토콜이다.

Record layer에서는 전송될 데이터를 보호하기 위해서 다음과 같은 과정을 거친다. 각각의 data를 알맞은 크기로 자르고, 선택적으로 압축하고, MAC를 적용하고, 암호화하여 그 결과를 전송한다. 전송된 데이터는 복호화되고 입증되고, 압축이 풀리고, 분할된 데이터가 다시 합쳐져서 상위 level로 전송된다.

위와 같이 Record layer에서 데이터를 보호하기 위해서 적용되는 암호알고리즘과 암호키, MAC 알고리즘은 cipher spec에 의해 결정된다. 이는 handshake protocol을 통해 전송되고, 이때 데이터 암호키 생성을 위해서 pre-master-key도 전송된다. 데이터 암호 알고리즘은 공개키 알고리즘보다 계산량이 적은 Block 암호 알고리즘을 사용한다.[4]

SSL 클라이언트와 서버가 처음 통신할 때 그들은 protocol version, 암호 알고리즘, 선택적인 각각의 인증, 공개키(public key) 암호기술을 협의하고, 공개키 암호기술을 이용하여 이 비밀정보를 분배한다.[3] 이러한 과정은 그림 1에서 보이는 SSL handshake protocol을 이용한다.

Client hello message :
클라이언트가 수행할 수 있는 cipher spec list와 Client random number, session ID를 포함한다.
Server hello message :
선택된 cipher spec과 Server random number, session ID를 포함한다.
Server key exchange massage :
서버의 key exchange parameters MD5-hash값, SHA_hash값, signature Algorithm, E_{Kpub}(sign된 임의의 공개키)를 포함한다.
Client key exchange message :
E_{Kpub}[premaster secret]를 포함한다.

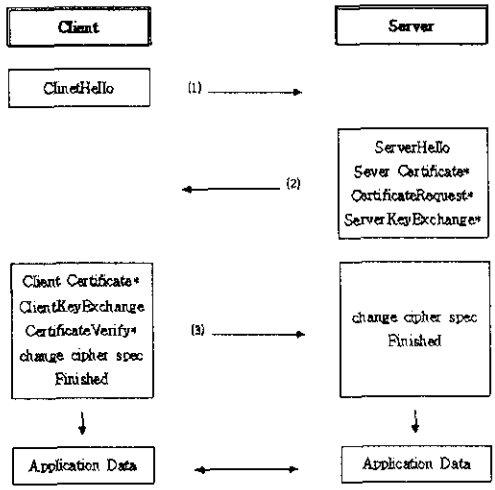

그림 1. SSL의 handshake protocol

(1) 클라이언트는 서버에게 Client hello message를 전송한다.

(2) Sever는 hello message에서 session ID가 어떠한 값을 갖는다면, 이 session ID와 session cache값과 비교하여 값이 있으면 기존의 cipher spec을 사용한다. 그렇지 않을 경우에 서버는 Client hello message로부터 얻은 cipher spec list 중에서 하나를 선택하여 Server hello message를 클라이언트로 전송한다. 곧바로 Server Certificate를 전송하고, 또한 Serverkey Exchange는 임의의 공개키를 사인하여서(RSA certificate DSS certificate) 전송한다(RSA 암호화를 사용할 경우).

(3) 클라이언트는 Server Certificate를 증명(verify)하여 서버를 인증한다.
만약 서버가 CertificateRequest를 요구하면 Client Certificate를 전송한다. random하게 pre-master-key를 생성하여 Serverkey Exchange에서 얻어낸 서버의 임의의 공개키로 암호화하여 서버에 전송한다.
클라이언트와 서버는 pre-master-key로부터 master-key를 생성한다.

위의 handshake protocol에서 session ID와 캐쉬(cache)의 값이 맞게 되면 그림 2에서와 같은 과정을 거

치며 캐쉬에 저장되어있던 기존의 cipher spec과 암호키를 재사용한다.

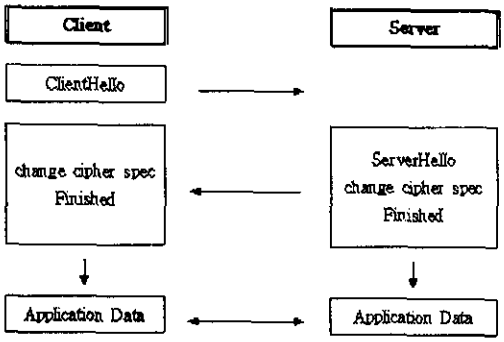

그림 2. Session ID를 이용한 키 재사용 protocol

이렇게 handshake protocol을 이용하여 서버와 클라이언트는 cipher spec과 pre-master-key를 가지게 된다. pre-master-key로부터 다음과 같이 master키를 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));
```

master secret는 다음과 같이 데이터를 암호화하기 위한 키와 MAC 키, IV(Initial Vector)를 생성하는데 사용된다.

```
key_block =
    MD5(master_secret + SHA('A' + master_secret
    + SeverHello.random + ClientHello.random))
  + MD5(master_secret + SHA('BB' + master_secret
    + SeverHello.random + ClientHello.random))
  + MD5(master_secret + SHA('CCC' + master_secret
    + SeverHello.random + ClientHello.random))
  + [...];
```

위와 같이 key_block을 생성하여 다음과 같이 차례로 키를 얻어낸다.

```
client_write_MAC_secret[CipherSpec.hash_size]
server_write_MAC_secret[CipherSpec.hash_size]
client_write_key[CipherSpec.key_material]
server_write_key[CipherSpec.key_material]
client_write_IV[CipherSpec.IV_size]
server_write_IV[CipherSpec.IV_size]
```

이렇게 얻은 암호키를 이용하여 cipher spec에서 정의한 알고리즘을 이용하여 데이터를 안전하게 암호화한다.

Ⅲ. SSL의 한계와 성능 개선 프로토콜 제안

SSL handshake protocol은 X.509를 이용하여 서버와 클라이언트를 서로 인증하고 암호 알고리즘, MAC 알고리즘 등의 cipher spec과 pre-master-key를 주고 받는다. 이러한 비밀 정보를 안전하게 전송하기 위해서 SSL에서는 RSA공개키 알고리즘계산이 필요하다. 이러한 공개키 알고리즘은 계산량이 크기 때문에 자주 사용하게 되면 계산 시간이 많아진다. 그렇기 때문에 SSL handshake protocol은 session ID를 이용하여 클라이언트와 서버가 기존에 사용했던 cipher spec과 데이터 암호용 암호키를 재사용하게 된다.

그러나 데이터를 암호화하는 대칭암호 알고리즘은 이러한 키의 재사용을 함으로서 취약성을 가진다. http와 같은 서비스에서 주고받는 데이터는 예측이 가능하다. 그러므로 known text attack과 선택 공격(choosen attack)에 대한 취약성을 가진다.[7] 그렇기 때문에 키의 재사용은 키의 안전성을 저하시킨다.

또한 키를 재사용하지 않고 기존 SSL의 handshake protocol을 거쳐서 새로운 cipher spec과 암호키를 생성하려면 공개키 알고리즘의 계산을 다시 해야 된다. 이는 시스템의 계산량을 증가시키게 된다. 이러한 SSL의 한계를 보완하기 위해서 개선된 handshake protocol을 제안한다.

기존의 SSL handshake protocol을 기반으로 하여, 키의 재사용을 막고, 새로운 데이터 암호키를 생성하기 위하여 기존의 session ID의 의미를 변경하였다. 기존의 handshake protocol에서는 session ID가 값을 가질 경우 캐쉬값과 비교하여 값이 맞을 경우에 기존에 사용한 cipher spec과 암호키를 사용하였다. 이는 키 재사용 문제를 가진다. SSL에서는 키를 재사용하지 않고 데이터 암호키를 변경하기 위해서는 다시 handshake protocol을 거쳐야 한다. 이렇게 키 변경을 할 때마다 handshake protocol을 거치게 되면 SSL의 한계에서 지적했듯이 공개키 알고리즘을 계산하기 위해 많은 계산량을 가지기 때문에 비효율적이다. 따라서 제안한 프로토콜에서는 계산량이 적은 키 변경 프로토콜을 제안한다.

기존의 handshake protocol과 같이 session ID가 값을 가질 경우에 이를 캐쉬값과 비교하여 값이 맞을 경우에 기존의 cipher spec을 유지한다. 그러나 여기서 기존의 데이터 암호키를 재사용하지 않고 기존의 master key와 새롭게 주고받은 Client random number와 Server random number를 이용하여 새로운 키를 생성한다. 다음 그림은 제안한 프로토콜을 설명한다.

(1) Client Hello를 서버에 전송한다. Client Hello는 새로운 Client random number를 포함한다.
(2) 서버는 session ID를 확인하고 기존의 cipher spec을 선택하여 Server Hello를 전송한다. Server Hello는 새로운 Server random number를 가진다.

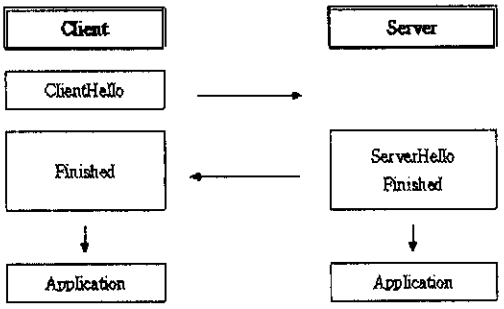

그림 3. 제안한 키변환 프로토콜

이렇게 교환한 새로운 random number와 기존의 master key를 이용하여 다음과 같이 새로운 암호 key를 생성해낸다.

```
key_block =
  MD5(master_secret + SHA('A' + master_secret
    + new.SeverHello.random
    + new.ClientHello.random))
+ MD5(master_secret + SHA('BB' + master_secret
    + new.SeverHello.random
    + new.ClientHello.random))
+ MD5(master_secret + SHA('CCC' + master_secret
    + new.SeverHello.random
    + new.ClientHello.random))
+ [...];
```

위와 같이 key_block을 생성하여 다음과 같이 차례로 키를 얻어낸다.

client_write_MAC_secret[CipherSpec.hash_size]
server_write_MAC_secret[CipherSpec.hash_size]
client_write_key[CipherSpec.key_material]
server_write_key[CipherSpec.key_material]
client_write_IV[CipherSpec.IV_size]
server_write_IV[CipherSpec.IV_size]

제안한 프로토콜은 기존의 프로토콜과는 달리 키 생성을 하기위해서 pre-master-key의 교환이 필요하지 않다. 따라서 pre-master-key를 암호화하고 복호화하는 공개키 알고리즘계산 과정이 필요치 않고, 서로의 certification의 확인이 필요치 않다.

Ⅳ. 결론

본 논문에서는 SSL의 데이터 보호 방법과 인증과 비밀정보의 전달방식을 분석하였다. 대칭암호키 재사용 때문에 생길 수 있는 공격에 의한 문제점을 제시하고, SSL에서 새로운 키를 생성하기 위해서 필요한 handshake protocol반복에의해서 계산량이 증가되는 문제점을 제시하였다. 효율적인 키변경을 위해서, 공개키 알고리즘 계산과정이 필요치않고 서로의 인증과정없이 안전하게 암호키를 변경할수 있는 개선된 handshake protocol을 제안하였다.

참고문헌

[1] Alan O. Rfeier, Philip Karlton, Paul C. Kocher, "The SSL Protocol", Internet draft, 1996.3.
[2] S.M. Bellovin, "Security problem in the TCP/IP Protocol Suite", Computer Communication Review, vol.19, No. 2, pp. 32-48, 1989.4.
[3] Douglas R. Stinson, "CRYPTOGRAPY Theory and Practice", CRC, 1995.
[4] National Bureau of Standards, "The Data Encryption Standards", FIPS publication 46, 1977.
[5] B.Kaliski, "Privacy Enhancement for Internet Electronic Mail", Internet draft, 1993.2.
[6] G. Bossert, S. Cooper, W. Drummond, "Considerations for Web Transaction Security", Internet draft, 1997.1.
[7] Mitsuru matsui, "Linear Cryptalysis Method for DES Cipher", Eurocrypt '93, pp. 386-397, 1993.5.