효율성과 확장성을 고려한 SSL 인터페이스의 설계 및 구현

조은애°, 김영갑, 문창주
고려대학교 컴퓨터학과
e-mail:{eacho°, ygkim, cjm}@software.korea.ac.kr

Design and Implementation of SSL Component Considering Efficiency and Extension

Eun-Ae Cho°, Young-Gab Kim, Chang-Joo Moon
Dept of Computer Science and Engineering, Korea University

요 약

인터넷 시장의 괄목할만한 성장에 따라 정보 보안에 관한 중요성이 증대되고 있다. 현재 Netscape사의 SSL 프로토콜이 인터넷 상의 정보보호를 위해 사용되고 있지만 한번 연결되면 융통성이나 확장성이 없이 데이터를 처리해야한다는 단점을 가지고 있고, 최근 소프트웨어를 부품화 시켜 조립, 분해가 가능하도록 지원하는 컴포넌트 플랫폼에 대한 호환 API가 없어 이에 대한 도입의 필요성이 증가하였다. 따라서 본 논문에서는 SSL의 기능을 수행하면서 선택적으로 데이터를 암호화할 수 있고, 국내 표준 암호화 알고리즘의 제공이 가능하며, 컴포넌트의 재사용적 측면, 생산성 향상 측면, 비용 절감적 측면을 향상시켜주는 SSL 컴포넌트의 설계 및 구현을 제안하였다.

1. 서론

인터넷의 사용이 보편화되고 정보에 대한 보안이 증가함에 따라 보안의 중요성이 인식되어 이 분야에 대한 연구, 개발이 최근 활발하게 이루어지고 있다. 특히 정보 보호에 대한 개념을 도입한 개발이 크게 증가하고 있는데, 따라서 온라인상에 있는 데이터들을 암호화하기 위한 작업들을 필요로 하게 되었다.

현재 쓰이고 있는 온라인상의 보안 프로토콜로는 Netscape사의 SSL(secure socket layer)이 HTTPS(secure hypertext transfer protocol)의 형태로 가장 많이 쓰이고 있다. 이 외에도 OpenSSL, JSSE(java secure socket extension) 등이 개발되어 쓰이고 있다.

그러나 기존의 SSL 프로토콜은 보안의 기능은 제공하지만 애플리케이션 내부에서 함수 형태로 이용될 뿐, 컴포넌트 환경을 도입할만한 API 호환 기능은 가지고 있지 않다. SSL 프로토콜은 한번 설정되면 모든 데이터에 대한 암호화를 실행하기 때문에 암호화 트랜잭션의 성능과 서비스 속도를 현격히 저하시킨다. 또, 선택적인 암호화를 할 수 없을 뿐만 아니라 개발에 있어서 프로그래밍 언어에 따라 개발자가 직접 API를 습득하고 연구하여야 하므로 일반적인 개발자가 보안의 기능을 하는 소프트웨어를 개발하기에는 어려움이 따른다는 단점이 있다.

따라서, 본 논문에서는 위의 단점을 해결하고자 온라인상의 보안성을 제공해주는 SSL 프로토콜에 생산성 향상과 재사용성이 보장되는 컴포넌트의 개념을 결합하여, 보안 API 등에서 개발자의 편의를 제공하고 선택적 암호화가 가능하도록 CBD 기반의 SSL 컴포넌트의 설계와 구현을 제안한다.

2. 관련 연구

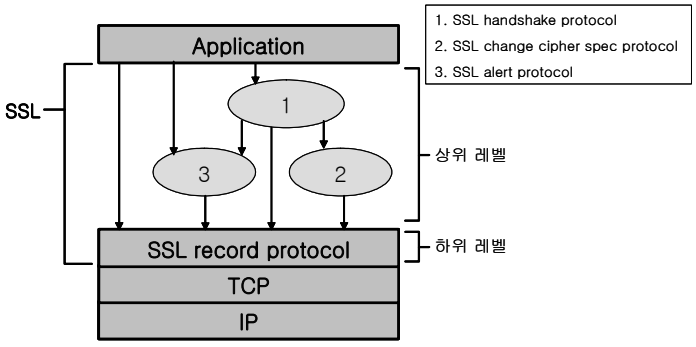
2.1 SSL의 목적 및 기능

SSL 프로토콜은 HTTP 프로토콜을 확장시켜 웹의 보안성을 향상시킨 S-HTTP(secure-HTTP)와는 달리, 웹 서버와 브라우저간의 종단 간(end-to-end) 보안 서비스를 제공하기 위한 프로토콜이다. TCP/IP 계층과 응용 계층(application layer) 사이에

서 동작하며, 쇼핑몰과 같은 상거래에 목적을 둔 사이트에서 신용카드나 개인정보와 같은 중요 정보를 인터넷을 통하여 보낼 때, 기밀성(privacy), 신뢰성(authenticity), 무결성(integrity) 등을 보장해주는 기능을 해준다.[1]

2.2 SSL 프로토콜(SSL protocol)의 구성

SSL 프로토콜은 그림 1과 같이 하나의 계층으로 이루어진 것이 아니라 크게 SSL 핸드쉐이크 프로토콜(SSL handshake protocol)과 SSL 레코드 프로토콜(SSL record protocol)로 이루어진다.[2]



(그림 1) SSL 프로토콜 스택

SSL 프로토콜의 상위 계층인 SSL 핸드쉐이크 프로토콜은 SSL 암호기 스펙 교환 프로토콜(SSL change cipher spec protocol)과 SSL 경고 프로토콜(SSL alert protocol)과 함께 SSL 프로토콜의 동작에 대한 관리를 위한 서비스를 제공하는 부분이고, SSL 프로토콜의 하위 계층인 SSL 레코드 프로토콜은 전송 계층(transport layer) 바로 위에 위치하여 네트워크 상에서 실질적인 보안 서비스를 제공한다.

SSL 핸드쉐이크 프로토콜은 SSL 클라이언트가 서버와 처음 통신할 때, 각 프로토콜의 버전이 일치하는지를 확인하고, 데이터를 암호화할 때 사용할 알고리즘을 선택한다. 그런 다음 서버와 클라이언트는 서로를 인증하고, 공개키 암호 기법을 이용하여 공유 비밀키를 생성하여 서버와 클라이언트의 상태를 통합하는 일을 한다. 이 때 SSL 암호기 스펙 교환 프로토콜은 암호화 방법에서 현재 상태에 따라 신호의 변화를 주는 기능을 하며, SSL 경고 프로토콜은 경고와 치명적인 결과에 대한 상세한 정보를 전달하는 기능을 한다.[3]

SSL 레코드 프로토콜은 상위 계층에서 받은 데이터들을 수신하여 분할, 압축하거나 암호화하여 실질적으로 네트워크 상을 오가는 메시지들을 안전하게 보호하는 기능을 한다. 본 논문에서는 이 프로토콜 중 주요 역할을 담당하는 SSL 핸드쉐이크 프로토콜과 SSL 레코드 프로토콜을 중점적으로 설계, 구현하였다.

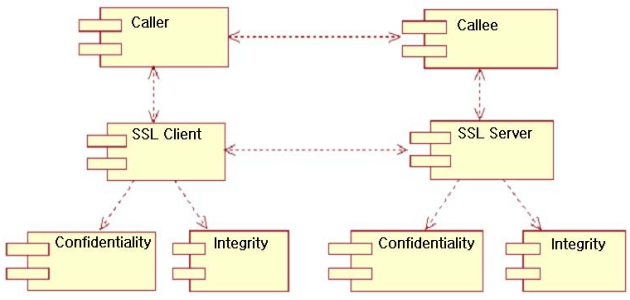
3. SSL 컴포넌트의 설계

본 논문에서 제안하는 CBD에 기반한 SSL 컴포넌트는 개발자에게 플러그앤플레이(plug-and-play) 방식으로 보안 서비스에 대한 메커니즘을 제공한다. 또한 컴포넌트의 개념 도입을 통해 기존의 SSL 프로토콜의 단점을 보안하는데 그 목적을 둔다.

3.1 SSL 컴포넌트의 구성

SSL 컴포넌트는 시스템의 하부 암호화 API의 종류에 상관없이 SSL 컴포넌트 플랫폼에서 독립적으로 지원하여야 한다. 또한 보안 통신 프로토콜인 SSL의 기능이 필요한 기존의 애플리케이션 컴포넌트 또는 해로 개발될 애플리케이션 컴포넌트 구현 코드 내에서 가능한 최소한의 보안 프로그래밍이 요구되도록 해야 한다. 이를 만족하기 위해서 컴포넌트 플랫폼을 기반으로 하여 비밀성, 무결성 SSL 컴포넌트를 각각 구현하였으며, 핸드쉐이크 프로토콜 중 서버에서 익명의 Diffie-Hellman(Anonymous Diffie-Hellman) 키 교환법을 선택하여 비밀성과 무결성 기능의 구현에 초점을 맞춘다.

그림 2는 SSL 컴포넌트가 실행되어 클라이언트와 서버로 각각 동작하고 비밀성, 무결성 컴포넌트와 연계되어 동작하는 구성을 컴포넌트 다이어그램으로 나타낸 것이다.[4]

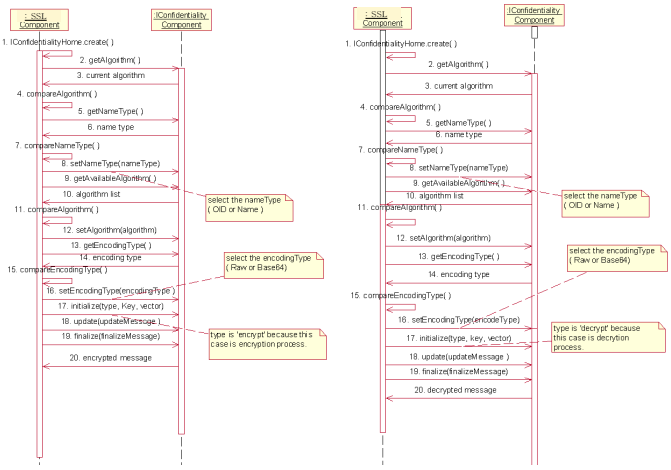


(그림 2) 컴포넌트 다이어그램

Caller가 Callee를 호출하면, SSL Component가 실행되고, SSL Client와 SSL Server는 서로 연결을 형성하여 SSL 프로토콜이 교환하는 메시지의 순서에 맞게 메시지를 교환하게 된다. 데이터의 암복호화 시 Confidentiality와 Integrity 컴포넌트의 메소드를 호출하여 비밀성과 무결성의 기능을 하는데 사용한다.

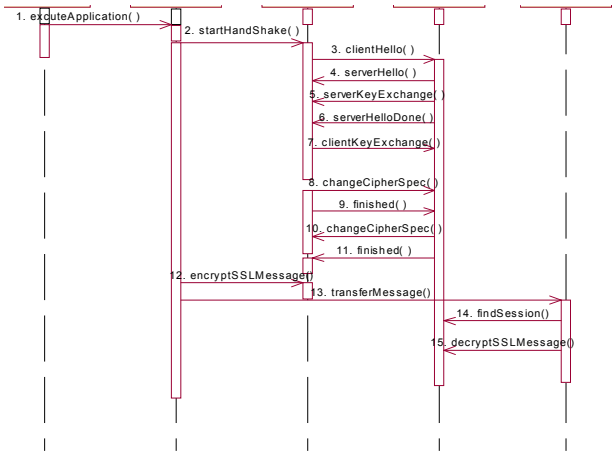
3.2 비밀성, 무결성 및 SSL 컴포넌트의 설계

비밀성 서비스는 메시지의 내용을 알아볼 수 없도록 암호화하는 서비스이고, 무결성 서비스는 정보의 내용이 불법적으로 변경되거나 삭제되지 않도록 하는 정보 보호 서비스이다. 그림 3은 비밀성 컴포넌트의 시퀀스 다이어그램이다.[5]



(그림 3) 비밀성 컴포넌트의 암복호화 시퀀스 다이어그램

먼저 비밀성 컴포넌트의 인터페이스를 생성하고 사용 가능한 알고리즘을 알아본 뒤, 알고리즘을 교환하고 선택한다. 이때 알고리즘은 개발자가 선택할 수 있으므로, 타입을 이름으로 할 것인지 그 알고리즘의 OID(Object Identifier)로 할 것인지, 또 인코딩 타입을 “raw”로 할 것인지 ”base64”로 할 것인지를 설정하는 것이 가능하다. 알고리즘과 인코딩 타입의 설정이 끝나면 initialize, update, finalize의 과정을 거쳐 암복호화된 메시지를 전송한다. 무결성 컴포넌트는 메시지의 변경 여부에 대한 검증을 하는 기능을 하므로 비밀성 컴포넌트와 같은 순서와 메소드를 가지고 동작한다.



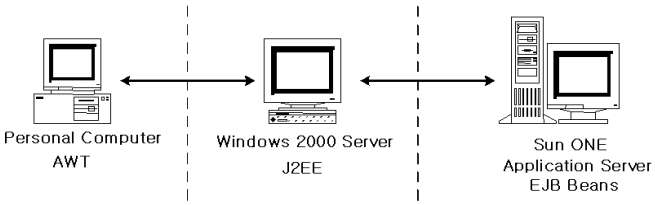
(그림 4) SSL 컴포넌트의 시퀀스 다이어그램

SSL 컴포넌트가 동작하는 시퀀스 다이어그램은 그림 4와 같다. 그림 4는 SSL 프로토콜이 동작하는 메시지 종류와 기능을 모두 만족한다. 그림에서 2~11은 SSLClient와 SSLServer가 논리적 접속을 초기화하여 핸드쉐이크 프로토콜을 실행하는 단계이고, 그중 5~7은 SSLClient와 SSLServer 간에 비대칭 키들을 교환하는 과정이다. 키 교환 방법으로 익명의 Diffie-Hellman을 사용하므로 인증은 생략한다. 12~15는 SSL 레코드 프로토콜을 실행하는 과정이다.

4. SSL 컴포넌트의 구현 결과 및 평가

4.1 구현 환경

SSL 컴포넌트는 그림 5와 같은 환경에서 구현하였다.



(그림 5) 구현 환경 구성도

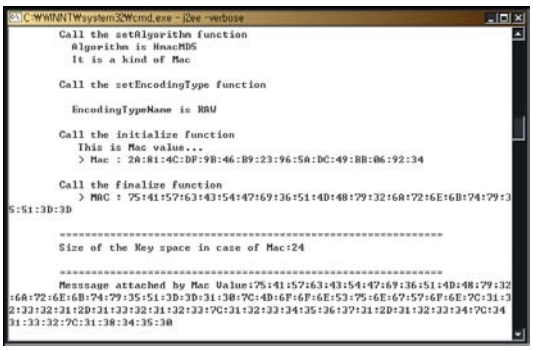
구현을 위한 서버, 클라이언트의 운영체제는 모두 Windows2000 서버이고 개발 도구는 J2SDK1.4를 사용하였다. 애플리케이션 서버로는 Sun ONE Application Server를 사용하고, 자바 기반의 암호화 라이브러리는 (주)STI의 J/LOCK을 사용하였다.[6]

4.2 구현 결과

CBD 기반의 SSL 컴포넌트는 위에서 언급한 바와 같이 무결성 컴포넌트, 비밀성 컴포넌트, SSL 컴포넌트로 나누어 구현하였다. 그림 6은 핸드쉐이크 과정이 시작된 화면이다. 사용자가 보안이 필요한 메시지를 보내기 위해서 SSL 컴포넌트를 실행시켜 핸드쉐이크 과정이 시작된 화면이다.

(그림 6) 핸드쉐이크의 시작

핸드쉐이크가 실행된 후 클라이언트와 서버가 알고리즘을 교환한 다음 암호화를 진행한다. 이때 그림에서 볼 수 있듯이 개발자의 의도에 따라 알고리즘과 타입의 선택이 가능하고 SEED, HAS-160과 같은 국내 암호화 알고리즘의 선택도 가능하다.



(그림 7) 무결성 컴포넌트의 적용 과정

그림 7은 SSL 컴포넌트가 메시지의 변조 및 삭제의 여부를 검사하기 위해 무결성 컴포넌트를 적용한 화면이다. initialize와 update, finalize 과정을 거쳐 메시지에 대한 메시지 인증 코드(MAC) 값을 생성한다. 이와 같은 과정을 통해 보내고자 하는 데이터 중 선택적으로 암복호화를 수행하게 되므로 암복호화를 필요로 하는 데이터의 량이 줄어들게 되어 효율적 측면에서도 향상된 결과를 가져올 수 있다.

4.3 비교 및 평가

표 1은 본 논문에서 제안한 SSL 컴포넌트 프로토콜과 기존의 SSL 프로토콜을 비교한 것이다.

<표 1> SSL 컴포넌트와 기존의 SSL 프로토콜의 비교

	SSL 컴포넌트 프로토콜	기존의 SSL 프로토콜
재사용성	재사용이 가능함	재사용이 불가능
유연성	좋음	나쁨
다양성	다양한 보안 메커니즘의 선택, 적용이 가능	메커니즘을 선택하는 것이 불가능
이식성	국내 표준 알고리즘 사용 가능(SEED, HAS-160)	국내 알고리즘 사용 불가
확장성	선택적으로 SSL 서비스를 수행. 확장성 있음	한번 설정한 연결은 변경 불가. 확장성 없음
일반성	암호 API를 모두 알지 못해도 개발이 가능	API를 모두 숙지하고 있어야만 개발이 가능
효율성	CPU와 시간의 오버헤드를 줄일 수 있음	모든 데이터를 처리, 암호화하므로 성능이 저하

기존 SSL 프로토콜의 가장 큰 단점은 한번 SSL 채널이 설정되면 그 연결을 지나는 모든 데이터들은 암호화과정을 거치게 되므로 비효율적이라는 것이다. 이로 인해 선택적인 서비스를 할 수 없으며, 많은 데이터의 처리를 해야만 했다. 또한 컴포넌트 플랫폼 기반의 API를 제공하지 않아 재사용성, 확장성 등의 편의를 제공하지 못한다. 그러나 SSL 컴포넌트는 선택적인 데이터의 암복호화 뿐만 아니라 재사용성, 효율성 등으로 인해 생산성 향상과 비용 절감의 측면에서 많은 효과 줄 수 있다.

5. 결론 및 향후 연구

본 논문은 SSL 프로토콜과 컴포넌트 개념을 결합하여 SSL 프로토콜과 동일한 기능을 제공해주는 동시에 SSL 프로토콜의 단점은 보안하고 컴포넌트의 장점을 충분히 수용한 CBD 기반의 SSL 컴포넌트를 구현하고 제안하였다. CBD 기반의 SSL 컴포넌트는 기존의 SSL 프로토콜이 원자적인 성질 때문에 선택적인 서비스를 하지 못한 단점을 보안하기 위하여 컴포넌트 개념의 도입을 통해 선택적인 서비스가 가능하도록 개선시켰으며, 그로 인해 확장성이 향상과 재사용성 향상, 효율성 향상의 효과를 얻을 수 있도록 하였다. 또한 국내 표준 암호화 알고리즘의 적용을 통하여 국내 정보보호 시장의 활성화 계기를 제공하였다.

그러나 인증 부분에 있어 본 논문에서는 익명의 Diffie-Hellman을 사용하여 간소화하였으므로, 명확하고 복잡한 인증 부분에 대해서는 또 다른 제안과 설계가 필요하다. 또한 비밀성, 무결성 이외의 다른 보안 서비스를 제공하는 컴포넌트의 표준안이 제시되어야 할 것이며, 그에 따른 연구 개발이 필요하다.

참고문헌

[1] 윤재호, "인증서 기반의 SSL 프로토콜", KISA, 2001.12

[2] Netscape사, http://www.netscape.com/, 2003

[3] stallings, "Cryptography and Network Security : Principles and Practices", Pearson Education

[4] John Cheesman, John Danielss, "UML Components : A Simple Process for Specifying Component-Based Software(The Component Software Series)", Addison-Wesley, 2001

[5] KISA 기술팀, “The Standardization of Confidentiality and Integrity Service Component Interface", KISA, 2003

[6] Java Cryptography Library, J/LOCK, http://www.stitec.com/product/ejlock.html