Java Code를 중심으로한 PKI기반 전자상거래 보안시스템 설계★

방기천* · 노시춘**

요 약

공개키 방식으로 가장 널리 사용되는 알고리즘은 RSA이다. 본 연구는 SSL 통신 및 RSA를 이용한 사용자 인증을 통해 결제 등의 과정에서 사용자의 소중한 정보를 안전하게 보호할 수 있는 안전한 사이버거래 구축방안을 제시한다. SSL은 전자서명 기술을 기반으로 한 암호화 프로토콜이며 이 기술이 적용된 전자문서는 별도의 암호화 과정을 거쳐 상대방에게 전달되므로 정보 송신자(웹브라우저에 정보를 입력하는 사용자)와 정보 수신자(해당 사이트의 웹서버 관리자) 외에는 그 내용을 해독할 수 없다. 따라서 전자문서가 전송되는 도중에 해커가 Sniffing을 시도한다고 해도 정보가 암호화 되어 있기 때문에 그 내용을 절대로 파악할 수 없다. 본 연구는 인터넷 쇼핑몰에서의 사용자 인증'과 'SSL 통신'을 이용한 안전한 쇼핑몰 구축을 목표로 방법론을 제시한다.

A Study of PKI-Based E-commerce Security System Design under Java Code Environment

Kee-Chun Bang* · Si Choon Noh**

ABSTRACT

RSA is the most widely used public key algorithms. Payment via the SSL communications, and user authentication using RSA secure shopping mall that can protect the user's valuable information in the process of building. SSL-based electronic signature technology and encryption protocols for this technology are electronic documents are delivered to the other party through a separate encryption process, the information sender to enter information on a web browser (user) and the recipient (the Web server of the site Manager), except you will not be able to decrypt the contents. Therefore, the information is encrypted during the transfer of electronic documents even if hackers trying to Sniffing because its contents can never understand. Of internet shopping mall in the user authentication 'and' Communications' SSL secure shopping mall built with the goal of the methodology are presented.

Key words : SQL Injection, Attacks Code, Countermeasures, Model, OWASP

접수일(2012년 11월 30일), 수정일(1차: 2012년 12월 10일), 게재확정일(2012년 12월 11일)

★ 본 논문은 2012년도 남서울대학교 교내연구지원에 의해 연구되었음.

* 남서울대학교 멀티미디어학과

** 남서울대학교 컴퓨터학과

1. 서 론

사이버거래는 자신이 원하는 상품과 서비스를 얻기 위해 따로 시간을 내지 않아도 언제나 어디서나 원하는 거래를 수행할 수 있는 24시간, 365일 서비스를 제공한다. 그러나 시스템 운영자의 소홀한 보안의식으로 자신의 소중한 개인 정보가 잘못 사용된다면 큰 문제가 될 것 이다. SSL은 전자서명 기술을 기반으로 한 암호화 프로토콜이며 이 기술이 적용된 전자 문서는 별도의 암호화 과정을 거쳐 상대에게 전달되므로 정보 송신자(웹 브라우저에 정보 입력 사용자)와 정보 수신자(해당 사이트의 웹서버 관리자) 외 에는 그 내용을 해독할 수 없다. 따라서 전자문서 전송도중 해커가 Sniffing을 시도해도 정보가 암호화 되어있기 때문에 그 내용을 파악 할 수 없다. 사이버거래에 대한 새로운 공개키 방식이 제시되고 있는데, 바로 타원 곡선의 원리를 이용한 ECC (Elliptic Curve Cryptosystem) 알고리즘 이다. 인터넷 쇼핑몰 에서의 사용자 인증'과 'SSL 통신'을 이용한 안전한 전자상거래 구축을 목표로 방법론을 제시한다. 연구순서는 관련연구, PKI 기반 SSL 통신시스템 개발, 결론의 순서이다.

2. 관련연구

2.1 SSL(Secure Socket Layer)

SSL은 전자서명 기술을 기반으로 한 암호화 프로토콜이며 이 기술이 적용된 전자문서는 별도의 암호화 과정을 거쳐 상대방에게 전달되므로 정보 송신자(웹브라우저에 정보를 입력하는 사용자와 정보 수신자(해당 사이트의 웹서버 관리자 외에는 그 내용을 해독할 수 없다. 따라서 전자문서가 전송되는 도중에 해커가 Sniffing을 시도한다고 해도 정보가 암호화되어 있기 때문에 그 내용을 절대로 파악할 수 없다 [1][2].

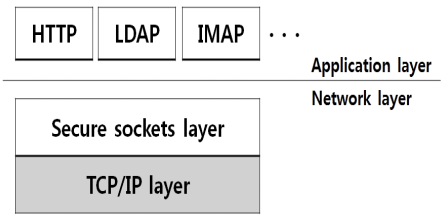
(그림1) SSL 전자서명 기술 기반 흐름

TCP/IP는 인터넷 상에서의 데이터 전송 프로토콜과 이다. HTTP나 Lightweight Directory Access Protocol(LDAP), Internet Messaging Access Protocol(IMAP) 등의 다른 프로토콜은 TCP/IP를 이용하기 때문에 TCP/IP 기반위에 있다.SSL은 TCP/IP 기반 위에서 작동 하며, HTTP나 IMAP과 같은 높은 레벨 프로토콜 아래에서 작동한다. 인터넷과 TCP/IP 네트워크의 통신에서 아래와 같은 특징을 가진다 [2][3].

○ SSL server authentication과정을 통해 사용자가 서버의 신원을 확인할 수 있게 한다. SSL 기능을 가지고 있는 클라이언트가 공개키 기술을 이용하여 서버의 인증서(Certificate)와 public ID가 유효하고 Certificate Authority에서 발행된 것인지를 확인한다.

○ SSL client authentication 과정을 통해 서버가 사용자의 신원을 확인할 수 있게 합니다. SSL 기능을 가지고 있는 서버가 사용자의 인증서와 public ID, 그리고 공인된 인증기관에서 인증을 받았는지 확인한다.

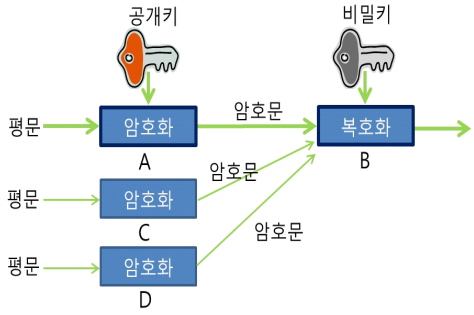
○ An encrypted SSL connection 과정을 통해 SSL로 이루어진 연결은 모두 암호화되어 전송됩니다. 그리고 암호화된 SSL 연결은 위조방지를 위한 메커니즘으로 보호된다.



(그림2) TCP/IP 환경의 SSL과 하위 프로토콜

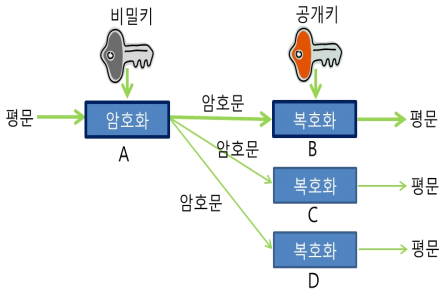
2.2 공개키 암호 기술

대칭키 암호기술의 가장 큰 문제점은 키분배의 어려움에 있다. 이에 대한 방안으로 스탠포드 대학 Whitfield Diffie와 Martin Hellman은 공개 키 방식이라는 새로운 암호모델을 1976년에 제안했다. 이 기술은 비밀키(private key)와 공개키 (public key)라는 2개의 키의 쌍을 구성하여 각 키가 암호화와 복호화에 사용되는 것을 의미한다. 비밀키는 소유한 사람 이외에는 절대 알 수 없는 키이고, 공개키는 원하는 누구나 공개 된다. 만일 공개키 로 암호해서 메시지를 보내고나서 이를 비밀키로 해독한다면, 메시지의 비밀성은 유지 됨을 알 수 있다. 이 경우 갑순이는 하나의 키쌍 (key pair)으로 N명의 송신자로부터 비밀 메시지를 받을 수 있다[4][5].



(그림 3) 공개키 방식 : 암호모드

공개키 방식은 대칭키에 비해 키관리는 편리하나 알고리즘이 더 복잡하기 때문에 처리 속도가 더 걸린다. 메시지가 커질 경우 부담은 더욱 크게 증가한다. 공개키가 많이 사용되는 응용 분야는 인증 모드(authentication mode이다. 이 경우 메시지 비밀성은 보장되지 않는다. 공개키는 누구든지 가질 수가 있으며 이는 누구든지 이 메시지를 읽을 수 있다는 것을 의미한다. 그러나 갑돌이는 자신의 비밀키를 통해서 암호화했고, 갑순이는 비밀키에 해당하는 공개키로 메시지를 해독할 수 있기 때문에 갑순이는 받은 메시지가 갑돌이로 부터 온 것 임을 확신할 수가 있다. 결국 송신자 인증 이라는 보안 서비스가 구현되며, 바로 이 특성이 디지털서명 기본원리 이다 [5][6].



(그림 4) 공개키 방식 : 인증모드

3. PKI기반 SSL통신시스템 개발

3.1 개발목표

SSL통신 및 RSA를 이용한 사용자 인증을 통해 결제 등의 과정에서 사용자의 소중한 정보를 안전하게 보호할 수 있는 안전한 쇼핑몰을 구축한다. 공개키 방식으로 가장 널리 사용되는 알고리즘은 RSA이다. RSA는 MIT의 Rivest, Shamir, Adleman라는 3사람 암호학자의 이름을 따서 만든 공개키 기반 암호기술로서 그 자체가 알고리즘의 이름이자 회사의 이름이 되었다. RSA는 민간 기업의 알고리즘이지만 실질적인 표준으로서 업계에서 자리잡고 있다. 그러나 RSA는 키 길이가 상당히 길어서 많은 제약이 따른다. 특히 스마트카드 같은 소용량 하드웨어 에서 사용할 경우 RSA 알고리즘은 적합하지 않다. 아직 RSA 알고리즘을 내장한 스마트 카드가 많이 있지만, 키 길이가 길어지면 길어질 수록 그 한계가 명확 하게 드러날 것이다. 이에 대한 새로운 공개키 방식이 제시되고 있

는데 키 길이를 줄일 수 있는 타원곡선 원리를 이용한 ECC(Elliptic Curve Cryptosystem)알고리즘이다.

3.2 SSL 연결과정 설계

① 클라이언트는 자신의 SSL 버전과 암호화 알고리즘, 임의로 생성된 데이터, 그리고 서버가 필요로 하는 또다른 정보들을 서버로 보낸다.
② 서버는 서버의 SSL 버전, 암호화 알고리즘, 임의로 생성된 데이터 그리고 클라이언트가 필요 로하는 정보를 클라이언트로 보낸다. 서버는 자신의 인증서(certificate)를 보낸다. 이때 클라이언트의 인증서를 요구할 수도 있다.
③ 클라이언트는 서버가 보낸 정보를 이용하여 서버를 인증한다. 만약 인증이 실패할 경우, 유저는 경고를 받게 된다.
④ handshake 동안 생성된 모든 데이터를 이용하여 *premaster secret*를 만든다. 이를 서버의 공개키(서버의 인증서로 부터 얻음)로 암호화 시켜서 서버 보낸다.
⑤ 만약 서버가 클라이언트를 인증하기를 원한다면 클라이언트는 handshake에 대해 유일한 데이터를 전사서명의 형식으로 보낸다. 이때 클라이언트는 서명된 데이터와 클라이언트 의 인증서, premaster secret 를 보낸다.
⑥ 서버가 클라이언트를 인증하기를 원해서, 그것이 실패한다면 연결이 끊어진다. 성공한 경우, 서버는 서버의 비밀키를 이용해서 premaster secret를 복호화하고 master secret를 생성한다.
⑦ 서버는 서버부분의 handshake가 끝났고, 이후의 데이터는 암호화 된다는 메시지를 암호화시켜 보낸다.
⑧ SSL handshake가 끝났고, SSL session이 시작된다. 서버와 클라이언트는 세션키를 이용하여 데이터를 암호화/복호화 시켜며, 데이터의 integrity를 확인한다.

3.3 JSP 수행 과정 설계

브라우저는 웹 서버가 클라이언트의 요청에 대해 만들어 낸 응답(컨텐츠)만을 본다는 것과 이 응답은 원래 소스의 도큐먼트였을 수도 있고 아니었을 수도 있다는 것이다. 웹 서버가 JSP 페이지에 대한 요청을 받았을 때 이 요청이 JSP컨테이너로 건네진다는 사실은 웹 브라우저가 알 필요 없고 알 수 없다. 해당 파일에 있는 코드를 읽고 해석 하는 동적 컨텐츠를 만들고 이것을 정적 컨텐츠에 끼워넣은 다음 최종 페이지를 HTTP 서버에게 돌려주는 일을 JSP 컨테이너가 맡고있다.

① HTML과 Java Code가 결합된 JSP Web Page를 작성한다.
② JSP를 최초로 호출하면 해당 JSP File을 Servlet으로 변환하고, Servlet을 Class Code로 Compile한다.
③ Class Code가 생성되면 Class Loader에 의해 Class를 Loading한다.
④ Class가 Loading되면 다음 요청이 있을 때 마다 재사용된다.
⑤ JSP의 변경이 일어나면 위의 과정을 반복한다.

자바빈즈(JAVABeans)는 광범위한 개념으로 사용되고 있으며, 간단히 설명하면 네 가지 요소를 갖추고 있는 자바 클래스 이다. 인자 없는 생성자 함수(default constructor)를 갖는다. 영속성을 위해 Serializable 또는 Extenalizable 인터페이스를 구현한다. 네이밍 컨벤션(Naming convention)을 지킨다. 캡슐화 (Encapsulation)를 지킨다. 열거한 규칙을 잘 지켜서 만들어진 빈즈는 어떠한 자바 환경 - 애플릿, Servlet, JSP, 또는 Stand -alone에서도 사용될 수 있다.

빈의 속성으로서 대부분의 빈들은 속성(properties)을 갖는다. 이는 그 빈의 읽기와 또는 쓰기 메소드(get/set)를 제공 한 빈의 속성이다. 빈의 속성에 대한 모든 접근은 이 메소드를 통해 수행되어야 하고, 데이터 필드 정의 시 접근 지정자는 private가 된다. 이는 캡슐화 개념을 코드에 적용한 경우가 된다. 메소드들의 이름을 지정하는 규칙으로 읽기 동작 을 위한 메소드는 get<PropertyName>으로, 쓰기 동작을 위한 메소드는 set<PropertyName>으로 정의된다.

자바빈즈와 JSP 액션으로서 JSP 페이지 개발자가 자바빈즈를 JSP 페이지에서 좀 더 쉽게 사용할 수 있도록 하기위해 다음과 같이 특별한 액션 태그를 제공하고 있다.

UseBean 액션태그

<jsp:useBean> 태그는 빈을 생성하거나 직렬화하지 않고 사용된 변수를 연동하기 위해서 사용된다. 형식은 다음과 같다.

<jsp:useBean id="name" scope="scope" typespec/>

setProperty 액션태그

<jsp:setProperty> 액션은 jsp 페이지에서 빈 속성들에 값을 할당하는 태그로 다음과 같은 형식으로 사용할 수 있다.

<jsp:setProperty name="name" property="property" value="value"/>

getProperty 액션태그

빈 속성 값들은 <jsp:getProperty> 액션을 사용하여 값을 얻을 수 있다. 형식은 다음과 같다.

<jsp:getProperty name="name" property="property-Name"/>

3.4 JDBC 인터페이스 설계

○ JDBC API

JDBC API는 자바 개발자들이 데이터베이스의 데이터를 액세스할 수 있도록 제공되는 표준 자바 API로서 클래스와 인터페이스로 구성되어 있다. 이들의 구현 내용들은 오라클, MS, IBM 등 DBMS을 만드는 회사에서 드라이버 형태로 제공되고 있다. 즉 JDBC는 이들 드라이버들의 공통된 인터페이스가 된다. 각 DBMS를 만드는 회사들은 많고, 이들이 만든 데이터베이스의 구조 및 조작 방법 등이 모두 다르기 때문에 드라이버 구현내용 도 다르다. 이로 인해 회사 마다 사용방법이 다른 DBMS 사용을 위해 프로그램 개발자들은 그때마다 새로 공부를 하고 정보를 얻어 개발을 해야 하는 어려움이 생긴다. 이 문제를 표준이 되는 인터페이스를 제공하여 해결할 수 있는데 DBMS를 위해 프로그래머에게 제공되는 인터페이스가 JDBC가 된다. 인터페이스는 추상 메소드들로만, 즉 구현되지 않은 메소드로만 구성된다. 데이터 베이스 제조 회사에서 JDBC 인터페이스를 자신의 데이터베이스에 맞게 기능 을 구현하고, 사용자들은 이들 메소드들이 어떻게 구현되어 있는지에 대한 자세한 정보를 몰라도 인터페이스 만 알면 데이터 베이스를 조작할 수 있다. JDBC API는 두개 패키지를 정의 하고 있다. java.sql 패키지는 JDBC의 기본적인 기능을 정의 하며 javax.sql 패키지는 서버 측의 기능을 강화하기 위한 기능을 정의한다.

○ 데이터베이스와 연결하기

데이터베이스와 작업을 하기 위해서 우선 사용 하고자 하는 데이터 소스와 연결부터 해야 한다. 데이터 소스와 연결하기 위해 JDBC API 에서는 DriverManager와 DataSource를 사용할 수 있다. 드라이버 로딩을 위해서는 코드 한 라인만 추가 하면 된다. 예를들어, JDBC- ODBC브리지 드라이버를 사용하고자 할 경우 드라이버를 로딩 하기 위해 코드를 다음과 같이 작성할 수 있다.

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

각 벤더별로 사용할 수 드라이버 클래스 이름이 다르다. 이 이름은 사용하고 있는 드라이버의 문서를 참조하면 알 수 있다. 예를들어, 드라이버 클래스 이름이 jdbc.DriverXYZ일 경우 드라이버를 로딩하기 위해 코드를 다음과 같이 작성할 수 있다.

```
Class.forName("jdbc.DriverXYZ");
```

여기에서 드라이버 인스턴스를 생성해서 Driverma- nager에 등록할 필요가 없다. 왜냐 하면, Class.for- Name() 메소드를 호출함으로써 이 작업이 자동으로 수행되기 때문 이다. 따라서 드라이 버 인스턴스를 직접 생성하는 코드를 작성하는 것은 불필요하게 작업을 중복해서 수행하는 것 이다. 또한, Class.forname() 메소드를 사용 함 으로써 런타임에

로딩할 클래스를 정할 수 있다. forName() 메소드로 클래스를 로딩하면 java. lang.Class를 리턴되지만 이미 클래스가 로딩 되었다는 것이 중요하므로 리턴되는 java .lang. Class 객체는 여기에서는 무시해도된다.

○ 데이터 소스와 연결

드라이버 로딩 후 데이터 소스와 연결하기 위해 다음과 같이 코드를 작성할 수 있다.

```
Connection con = Drivermanager.getConnection(url,"
myLogin", "myPassword");
```

JDBC-ODBD 브리지 드라이버를 사용할 경우, JDBC URL은 jdb:odbc:로 시작한다. 나머지 URL 부분은 데이터 소스 이름 또는 데이터베이스 시스템이 된다. 예를들어, “Fred”라는 ODBC 데이터 소스에 액세스하기 위해 ODBC를 사용할 경우, JDBC URL은 jdb:odbd: Fred가 된다. “myLogin”과 “myPassword” 부분은 DBMS에 로그인하기위해 사용할 이름과 패스워드를 입력 하면 된다. DBMS에 로그인 하기위해 사용할 로그인 이름 “SL”이고 패스워드가 “JAVA”일 경우, 데이터 소스와 연결하기 위해 코드를 다음과 같이 작성 할 수 있다.

```
String url = "jdbc:odbc:Fred";
Connection con = DriverManager.getconnection(url,
"SL", "JAVA");
```

JDBC 드라이버를 사용할 때, 문서를 살펴보면 데이터베이스에 연결하기 위해 사용해야할 서브 프로토콜이 무엇인지 명시하고 있고, 이 서브 프로토콜은 JDBC URL을 지정할 때 jdbc: <subprotocol> 다음에 놓는다. 예를들어, 드라이버 개발자가 서브 프로토콜을 acme로 등록 했을 경우, JDBC URL은 jdbc:acme 이 된다. JDBC URL에서 나머지 부분은 데이터 소스를 식별하기 위한 정보를 제공하게 된다. 즉, JDBC URL은 jdbc:<subprotocol>:<datasource_ identifier> 이다.

○ JDBC Statement 생성 및 실행

Statement 객체는 SQL문을 DBMS에 전송 하는 동작을 수행한다. JDBC API에는 Statement, Prepared-Staement, CallableStatement 이렇게 3종류의 State-ment 인터페이스가 정의되어 있다. State-ment를 이용해 데이터베이스로 전송할 SQL 문이 SELECT문의 경우에는 execute Query() 메소드를 사용하고, 테이블 생성 및 수정을 위한 SQL문의 경우에는 execute-Update() 메소드를 사용 한다. State-ment 객체는 다음과 같이 Connection 객체를 이용해 생성할 수 있다.

```
Connection con = .....;
Statement stmt = con.createStatement();
```

위 문장에 의해 stmt를 생성했지만, 아직 SQL 문을 DBMS에 전송한 것은 아니다. SQL문을 DBMS에 전송하기 위해서는 Statement가 제공 하는 메소드를 사용할 수 있다.

4. 결 론

본 연구를 통해 PKI기반 SSL통신 전자상거래 시스템 설계방법론을 제시했다. SSL은 전자서명 기술을 기반으로한 암호화 프로토콜이며 이 기술이 적용된 전자문서는 별도 암호화 과정을 거쳐 상대방에게 전달되므로 정보 송신자, 웹브라우저에 정보를 입력하는 사용자와 정보 수신자, 해당 사이트의 웹 서버 관리자 외에는 그 내용을 해독할 수 없다. 아직 RSA 알고리즘을 내장한 스마트카드가 많이 있지만 새로운 공개키 방식이 제시되고 있는데, 바로 타원곡선 원리를 이용한 ECC(Elliptic Curve Cryptosystem) 알고리즘 이다. 전자문서가 전송되는 도중에 해커가 Sniffing을 시도한다 해도 정보가 암호화되어 있기 때문에 그 내용을 절대로 파악할 수 없다. 본 연구에서 제안한 방법을 활용하여 인터넷환경에서 전자상거래시스템 운영에 참고 되기를 기대 한다.

참고문헌

[1] Biham, Eli and Alex Biryukov: An Improvement of Davies' Attack on DES. J. Cryptology 10(3): 195 - 206 (1997)

[2] Biham, Eli, Orr Dunkelman, Nathan Keller: Enhancing Differential-Linear Cryptanalysis. ASIACRYPT 2002

[3] Campbell, Keith W., Michael J. Wiener: DES is not a Group. CRYPTO 1992: pp512 - 520

[4] Diffie, Whitfield and Martin Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard" IEEE Computer 10(6), June 1977

[5] Ehrsam and others., Product Block Cipher System for Data Security, U.S. Patent 3,962,539, Filed February 24, 1975

[6] Gilmore, John, "Cracking DES: Secrets of Encryption Research, Wiretap Politics and Chip Design", 1998, O'Reilly, ISBN 1-56592-520-3.

[7] Kaliski, Burton S., Matt Robshaw: Linear Cryptanalysis Using Multiple Approximations. CRYPTO 1994

[8] Knudsen, Lars, John Erik Mathiassen: A Chosen-Plaintext Linear Attack on DES. Fast Software Encryption - FSE 2000

[저 자 소 개]

방 기 천(Kee-Chun Bang)

1981년 : 서울대학교 전자공학과(학사)
1988년 : 성균관대학교 정보처리학과(석사)
1996년 : 성균관대학교 전산통계학전공(박사)
1984년~1995년 : MBC 기술연구소
1995년~현 재 : 남서울대학교 멀티미디어학과 교수

email : bangkc@nsu.ac.kr

노 시 춘(Si Choon Noh)

1987년2월 : 고려대학교 경영정보학 석사
2005년2월 : 경기대학교 정보보호기술 박사
2002년11월 : KT 시스템보안부장
2004년 12월 : KT 충청전산국장
2005년3월 ~ 현재 :남서울대학교 컴퓨터학과 교수
IT융합연구소연구위원

email : nsc321@nsu.ac.kr