Plug-in 기법을 이용한 PGP 기반의 Web 보안 시스템 개발

김 태갑*, 조 은경**, 박 정수**, 류 재철*

* 충남 대학교 컴퓨터과학과
** 한국전자통신연구소 정보기술개발단

WWW Security Mechanism Using Plug-in and PGP

Tae-Gap Kim*, Eun-Kyung Cho, Jung-Su Park**, Jae-Cheoul Ryou***

* Department of Computer Science, Chungnam National Univ.
** Information Technology Division, ETRI

요약 : 인터넷 기반의 World Wide Web(WWW)은 여러 가지 측면에서 유용한 특성을 가지고 있다. 사용의 편리함, 멀티미디어 데이터 지원, Interactive한 사용자 인터페이스 등이 그 대표적인 예이다. 이러한 이유로 WWW은 급속한 성장을 이루었고 거의 모든 분야에 적용될 정도로 확산되어 있는 실정이다. 그러나 WWW은 위와 같은 장점 들에도 불구하고 인터넷이 지니고 있는 보안상의 취약점을 그대로 포함하고 있다는 단점을 지니고 있다. 이는 WWW 상에서 전송되는 데이터가 불법적인 접근에 쉽게 노출될 수 있는 형태 그대로 전송되기 때문이다. 이러한 문제점을 근본적으로 해결 할 수 있는 방법은 전송되는 데이터의 암호화로써 불법적인 데이터 접근을 시도하려는 사용자로부터 데이터를 보호 할 수 있다. 본 논문에서는 공개 키 암호 방식에 기초하고 있는 PGP(Pretty Good Privacy) 툴을 이용하여 WWW 상으로 전송되는 데이터의 암호화 및 인증 메커니즘을 제공하고자 한다. PGP 보안 툴의 삽입을 위해 Plug-in 이라는 Web 브라우저 확장 기법을 사용한다.

Abstract : World Wide Web(WWW) has a lot of useful charaters. Easiness of use, multi-media data supporting and interactive communication capability are typical reasons why people want to use WWW. But because WWW is based on Internet, it has some security problems which originate in plain format data transmission on physical transmission line. The unique solution for this problems is data encryption. Since theoritically proved encryption algorithms ensure data confidentiality, a unauthorized user can not know what is transmitted on network. In this paper, we propose a cryptography system which uses public key system. In detail, our public key based web security mechanism is using PGP module. PGP is a e-mail security system implemented by Phil Zimmermann. The basic idea of our propose is data encryption and integrity checking for all data which is transmitted on Web. To implement these facilities, we use netscape browser extension technology, plug-in. Through these technology, security mechanisms are added on netscape browser.

1. 서론

인터넷은 현재 지구상에 존재하는 최대의 통신 네트워크로서 지금도 그 사용이 급격히 증가하고 있으며 정보의 바다라고 불릴 만큼 방대한 양의 데이터를 포함하고 있다. 따라서 사용자들이 인터넷을 통해서 자신이 원하는 데이터를 찾아낸다는 것은 그리 쉽지 않은 작업이다. 이러한 문제점으로 인하여 인터넷은 점차 그 사용의 한계를 드러내고 있었다. 그러나 1989년 유럽의 CERN에서 제안된 World Wide Web(WWW)의 도입은 이러한 문제점을 해결하기 위한 방안으로 고려되기 시작하였고 이후 이에 대한 연구가 진행 되었다. 결과적으로 GUI 기반의 Web 브라우저인 Mosaic이 발표되었고 이후 Netscape, MicroSoft Explorer등이 공개 되면서 인터넷은 새로운 전기를 맞이하게 되었다. 이러한 Web 브라우저들은 WWW의 접근 메커니즘으로 Hyper-link를 제공함으로써 사용자는 단지 마우스를 누르는 단순한 작업을 통하여 자신의 원하는 데이터를 손 쉽게 얻을 수 있게 되었다. 사용이 간단하다는 점 이외에도 WWW은 다양한 형태의 멀티미디어 데이터(예를 들면 텍스트, 이미지, 음성, 오디오 데이터, 동 화상)를 처리 할 수 있는 기능을 제공 함으로써 그 사용자들에게는 네트워크를 사용하기 위한 일종의 Total-Solution으로 인식되고 있다

이러한 WWW의 사용 증가 추세에서 반드시 고려되어야 할 사항은 WWW이 가지고 있는 보안상의 문제점 들이다. WWW은 인터넷이 가지고 있던 보안상의 문제점들을 그대로 물려 받았기 때문에 데이터 전송 시의 기밀성이나 무결성을 보장 받을 수 없다. 따라서 전송되는 데이터에 대한 추가적인 보안 메커니즘이 고려되지 않는 한 데이터 보호가 요구되는 분야나 상업적인 용도로의 이용은 불가능 할 것이다. 예를 들면 군사 기밀이나 기업간의 정보 교환, 전자 상거래, 홈 뱅킹 등의 분야에서 불법적인 사용자에 의한 정보 유출이나 변조를 막을 수 없다면 상당한 혼란만 초래 할 것이다. 위와 같은 보안상의 문제점을 해결하기 위한 방안으로 1994년 EIT 사에서는 Secure HyperText Transfer Protocol(S-HTTP) 라는 새로운 형식의 HTTP 프로토콜을 제안하였다[1]. 이와 비슷한 시기에 Netscape 사에서는 Secure Socket Layer(SSL)[2] 이라는 프로토콜을 제안함으로써 WWW 보안 문제 해결에 새로운 전환점을 맞이하게 되었다. S-HTTP는 WWW의 기반 프로토콜인 HTTP 상에 암호화 모듈을 첨가 함으로써 데이터의 기밀성과 무결성을 보장 하였으며 SSL의 경우 HTTP의 기반이 되는 TCP/IP 레벨의 암호화 모듈을 제공함으로써 데이터의 보호를 시도 하였다. S-HTTP나 SSL의 경우 모두 기본적인 아이디어는 데이터의 암호화 송수신 이라는 면에서 서로 일맥상통 하지만 암호화 모듈이 적용되는 위치에서 서로 다르다는 차이점이 있다.

본 논문에서는 전자우편 보안 시스템인 PGP[3][4]를 이용한 WWW 보안 시스템을 제안하였다. S-HTTP나 SSL의 경우 TCP/IP나 HTTP와 암호화 모듈과의 연동으로 인하여 프로토콜 계층에서의 수정이 불가피한 반면 본 논문에서 제시된 보안 시스템의 경우는 HTTP 프로토콜의 수정이 필요 없는 응용 프로그램 계층에서의 암호화 방안이라는 데에서 그 특징을 찾을 수 있다. 이는 데이터의 암호화가 특정 그룹이나 회원 위주의 관리 시스템 내에서 수행되는 데이터 전송이나 서비스에 쉽게 적용된다는 것을 의미하는 것으로 WWW 통신의 오버헤드를 줄일 수 있다는 특징을 가진다. 인터넷이 정보의 공유를 주장한다는 점이나 사용 증가로 인한 전송률의 저하를 고려한다면 이러한 방식의 암호화 전송 방식은 여러 가지 장점을 가질 수 있다.

한편 본 논문에서 제시된 WWW 보안 시스템의 기능은 크게 HTML 문서 내에 입력된 사용자 데이터의 암호화 전송이나 사용자 소유의 화일 전송 혹은 서버로 부터의 화일 수신 기능으로 구분될 수 있다. 사용자 데이터 전송의 경우는 전자 상거래 분야의 기본 메커니즘을 제공 할 수 있으며 화일 송수신 기능의 경우 특정 폐쇄 그룹 내의 정보 공유를 위한 방안으로 사용될 수 있다. 특히 서버로부터의 화일 수신 메커니즘에서는 데이터의 암호화 전송 이외에도 간단한 형태의 사용자별 접근 제어 기능도 제공한다는 특징을 가지고 있다.

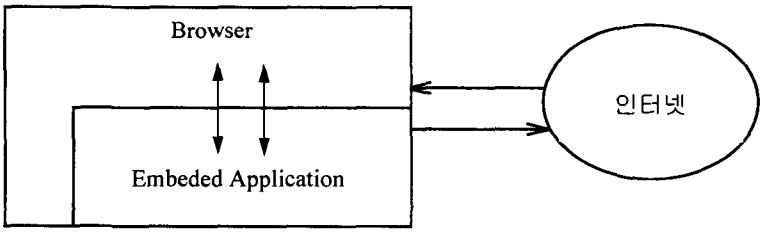
2장에서는 Web 브라우저에 보안 메커니즘을 첨가하기 위해서 사용될 Plug-in 기법에 대하여 살펴본다. 3 장에서는 PGP 기반의 Web 보안 시스템의 구현에 대하여 기술하고 4 장에서 결론을 맺는다.

2. Plug-in을 이용한 브라우저 확장 기법

Netscape 브라우저를 비롯한 대부분의 Web 브라우저들은 멀티미디어 정보를 효과적으로 처리할 수 있는 기능을 제공하고 있지만 브라우저 내에서 이러한 기능들을 모두 포함하고 있지는 않다. Netscape의 경우 지정된 URL의 방문(navigation), 전송되어 오는 데이터의 번역(Parsing) 및 디스플레이(Display), 방문한 URL의 로그정보 기록(History), 화일 열람(File open) 등의 기본적인 기능들만을 브라우저 자체 내에 포함하고 있을 뿐이다. 그렇다면 어떠한 방법으로 셀 수없이 다양한 형식의 멀티미디어 데이터들을 처리 할 수 있을까 하는 의문이 생길 것이다. 이러한 멀티미디어 데이터들은 종류도 다양할 뿐 아니라 점점 그 수가 늘어나고 있는 실정이라 이러한 수많은 형태의 데이터를 전부 다 고려한다는 것은 불가능한 일이다. 이러한 문제점을 해결하기 위한 방법은 브라우저와 데이터 교환이 가능한 외부 프로그램을 이용하는 것이다. 이러한 브라우저 확장 기법은 특정 멀티미디어 데이터의 MIME(Multi-purpose Internet Mail Extension) 타입과 외부 프로그램의 결합으로 표현될 수 있으며 브라우저 개발 초기부터 제공되고 있다. 본 장에서는 Plug-in을 이용한 브라우저 확장 기법에 대하여 기술한다.

2.1 Embeded Application

Embedded Application이란 <그림 1>에서 나타난 바와 같이 브라우저의 일부로 포함된 Embedded Application을 이용해서 사용자가 정의 할 수 있는 프로그램 모듈을 브라우저의 일부 기능으로 삽입하는 것을 말한다. 우선 전체적인 데이터 흐름과 제어를 살펴보면 <그림 1>과 같이 표현될 수 있다. 그림에서 표현된 바와 같이 Embedded Application은 브라우저가 구동 될 때 브라우저에 의해 인식되며 브라우저와 연결되기 때문에 Embedded Application 역시 독립적으로 수행이 불가능한 프로그램이다.



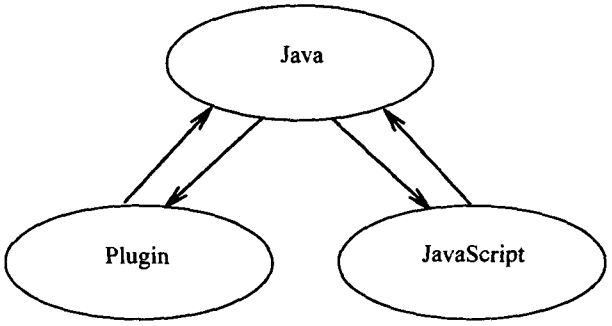
<그림 1> Embeded Application

Embedded Application 하나 이상의 MIME 타입과 연계 되며 Netscape에서 지원하는 Helper Application 프로그램과 달리 사용자가 외부 프로그램으로 등록하여야 하는 형태가 아니다. 즉 Embedded Application 자체 내에 MIME 타입을 지정하고 있기 때문에 브라우저에 의해 인식될 당시에 자동적으로 등록되는 형태이다. Embedded Application은 몇 가지의 프로그래밍 인터페이스를 지원하고 있으며 그 내용은 다음과 같다.

- Plug-in : Netscape
- Java : Sun
- ActiveX Control : Microsoft

Plug-in, Java, ActiveX Control 등은 Embedded Application과 브라우저를 연결 시켜주는 역할을 수행하며 프로그래머는 이러한 기능들을 이용해서 프로그램 모듈을 브라우저 내부에 삽입한다. 브라우저에 의해서 인식된 Embedded Application은 이와 연결된 MIME 타입의 데이터가 브라우저에 의해 수신될 경우 메모리로 로드되어 수행된다. Plug-in 개념은 Netscape 이외의 다른 응용

프로그램에서도 기능 확장을 위해서 일반적으로 사용되는 기법으로 개념상 뛰어난 장점을 가지고 있다. 한편 Java[5]의 경우 수행 기종에 거의 제약을 받지 않는 뛰어난 이식성을 비롯하여 이미지나 오디오, 동화상 등의 처리에 뛰어난 장점을 가지고 있다. 더구나 사용자와의 Interaction 적인 측면에서 뛰어난 장점을 가지고 있기 때문에 최근 각광을 받고 있는 기법이다. Active X Control은 MicroSoft Explorer를 위하여 개발된 확장 기법으로 Netscape Plug-in과 앞으로 치열한 경쟁을 벌일 것으로 전망된다. 한편 Plug-in과 Java/JavaScript 와의 상호 연결이 가능한 LiveConnect라는 기법이 Netscape 사에 의해서 개발됨에 따라 Plug-in과 Java의 특성을 결합 할 수 있는 방안이 마련되었다. LiveConnect 기법을 그림으로 표현 하면 <그림 2>와 같다[6].



<그림 2> LiveConnect

<그림 2>에서 나타나는 바와 같이 LiveConnect 기법이란 Java와 JavaScript, Plug-in 사이의 Inter-Communication 메커니즘을 의미한다. Netscape 3.0 Beta 4 이후의 버전에서만 제공되며 Java/JavaScript를 이용하여 Plug-in을 제어하거나 반대로 Plug-in을 이용하여 Java/JavaScript를 제어 할 수 있는 기능을 제공한다. 본 논문에서 제시된 PGP 기반의 Web 보안 시스템은 위에서 기술된 LiveConnect 기법을 이용하여 구현 되었다.

2.2 Netscape Plug-in

Netscape Plug-in이란 Netscape에 의해서 수행되는 C/C++등의 프로그래밍 언어로 제작된 동적 프로그램 모듈을 말한다. 이러한 Plug-in 모듈을 개발하기 위해서는 Plug-in API라는 일종의 프로그래밍 인터페이스를 사용하게 된다. Plug-in API의 목적은 C/C++등의 프로그래밍 언어로 제작된 Application 프로그램과 브라우저의 핵심 기능(예를 들면, Navigation)을 연결 시켜주는 역할을 수행함으로써 또 하나의 브라우저 확장 기법을 제공한다. Java 애플릿 역시 브라우저가 제공하지 못하는 여러 가지 추가적인 기능을 제공함으로써 일종의 브라우저 확장 기법을 제공하지만 프로그래머 입장에서는 Java Language라는 새로운 형태의 프로그래밍 언어의 습득과 이와 관련된 개발 환경의 구축이라는 부담을 안게 된다. 이에 비해 Plug-in의 경우 기존에 C/C++로 작성된 프로그램을 이용할 수 있다는 점과 이미 많은 프로그래머들에게 친숙한 언어라는 장점을 가지고 있다. 그러나 Java의 멀티미디어 데이터 처리 능력이나 Interactive한 수행 환경 및 수행 기종에 영향을 받지 않는 특성들은 나름대로 장점을 가지고 있다.

본 논문에서 구현된 PGP 기반의 Web 보안 시스템은 C/C++로 이미 제작되어 있는 PGP를 이용한다는 점과 Java/JavaScript가 제공하지 않는 Navigation 능력을 제공한다는 측면에서 Plug-in을 사용하여 구현 하였다. 현재까지 발표된 Plug-in의 기능을 살펴보면,

- MIME 타입의 등록
- 브라우저 화면 내에 Plug-in 윈도우 삽입
- 키보드/마우스 이벤트 감지

- 특정 URL의 데이터 요구
- 특정 URL로의 데이터 Posting
- Hyperlink 지원과 URL Book Marking

로 요약 될 수 있다. Plug-in 모듈은 HTML 문서 내에 Embed 되어서 브라우저에 의해 방문 될 때 메모리 상으로 로드되어 수행된다. 일단 Plug-in이 로드되면 Plug-in 윈도우가 브라우저 화면에 나타나게 되고 그 HTML문서를 떠날 때까지 활성화된 상태로 존재한다.

1. ***Embedding in HTML***

일반적으로 Plug-in은 Dynamic Linking Library(DLL)의 형태로 작성되며 브라우저가 구동 되면서 지정된 디렉토리(일반적으로 Netscape/Plug-ins)에 존재하는 Plug-in 모듈들이 인식 된다. 따라서 사용자는 자신이 원하는 기능을 제공하는 Plug-in 모듈을 브라우저가 지정하는 디렉토리에 복사해 두어야 한다. Plug-in이 성공적으로 등록 되었는지의 여부는 Netscape Help 메뉴의 "About Plug-ins" 항목을 보면 알 수 있다. 예를 들어 본 논문에서 제시된 PGP 기반의 Web 보안 시스템의 경우,

NPNETPGP Dynamic Link Library
File name: C:\NAVIGOLD\PROGRAM\Plug-ins\NpNetPGP.dll
NPNETPGP DLL

Mime Type	*Description*	*Suffixes*	*Enabled*
application/netpgp	*Netscape PGP as Plug-in*	*pgp*	*Yes*

의 형태로 등록되어 있다. 이러한 정보들은 Plug-in 모듈 안에 지정이 되어 있으며 브라우저가 구동 될 때 각 Plug-in 모듈들에 관한 정보를 읽어 들이게 된다. 위와 같은 형태로 브라우저에 등록된 Plug-in 모듈은 사용자가 특정 URL을 방문 할 경우 활성화 된다. Plug-in이 활성화되기 위해서는 사용자가 방문한 URL(HTML문서)에 <EMBED>라는 HTML Tag가 존재하여야 한다. <EMBED> Tag의 형식을 살펴보면,

<EMBED attributes> ... </EMBED>

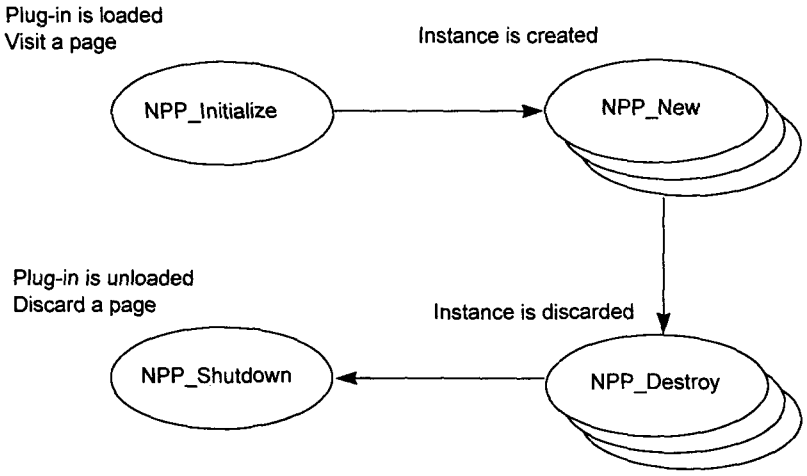
와 같다. 여기서 Attributes란 활성화될 Plug-in의 특성을 규정하는 것으로 다음과 같은 것들이 정의 되어 있으며 사용자가 추가적인 Attribute들을 정의 하여 사용 할 수도 있다.

- HEIGHT="value" related to UNIT
- HIDDEN='True" or "False"
- PALETTE="Foreground" or "Background"
- PLUG-INSPAGE="URL" indicates the location of instructions on install
- **SRC="URL" plug-in data file location**
- **TYPE="MIME Type"**
- WIDTH="value" related to UNIT
- UNITS="pixels" or "en" defines measurement unit

위에서 지정된 Attribute들 중 SRC, TYPE 두 Attribute 중 하나는 반드시 명시가 되어 있어야 한다. 이는 브라우저가 어떤 Plug-in 모듈을 로드 할 것인가를 결정 하는데 사용되기 때문이다. 한편 위에 설명된 NetPGP의 경우 다음과 같은 <EMBED> Tag에 의해서 활성화 된다.

<EMBED TYPE=application/netpgp NAME=NetPGPlug
ALIGN=CENTERWIDTH=0 HEIGHT=0>

일단 하나의 Plug-in이 활성화 되면 여러 개의 Instance들이 생성될 수 있어 사용자가 원하는 다양한 기능들을 수행하며 해당 URL을 떠날 때 까지 존재한다. <그림 3>은 이러한 Plug-in의 수행 과정을 설명하고 있다.



<그림 3> Plug-in Lifecycle

<그림 3>에 나타난 바와 같이 하나의 Plug-in이 활성화 되려면 Netscape에 의해서 Plug-in Method중의 하나인 NPP_Initialize이 호출되어야 한다. NPP_Initialize와 같이 NPP_, NPN_으로 시작하는 이름을 가진 Method들은 Netscape 사에 의해서 제공되는 일종의 API 함수 이다. 일단 Plug-in이 활성화 되고 나면 NPP_New를 이용하여 여러 개의 Plug-in 인스턴스를 생성할 수 있게 된다. 예를 들어 하나의 HTML 문서에 여러 개의 동일한 Plug-in을 지정하는 객체가 정의되어 있거나 동일한 MIME 타입의 데이터를 디스플레이 하는 Plug-in 윈도우가 여러 개 존재할 경우이다. Plug-in 인스턴스는 사용자가 그 URL을 떠나거나 Plug-in 윈도우를 닫을 때 소멸되며 활성화 되어 있는 마지막 인스턴스가 사라지면 Plug-in 모듈은 메모리로부터 제거되고 NPP_Shutdown이 호출된다.

Stream이란 URL과 그 URL이 가지고 있는 데이터들을 상징하는 것으로 Plug-in을 이용하여 특정 URL에 있는 데이터를 가져오거나 전송할 경우에 발생되는 데이터들은 Stream이라는 형태로 간주되어 처리된다. 본 논문에서 구현된 PGP 기반의 Web 보안 시스템의 경우 PGP를 이용하여 암호화된 데이터를 전송 하거나 수신 할 때 Stream이 생성되게 된다.

Plug-in API는 HTTP의 GET이나 POST 기능을 수행하는 함수들을 제공 하고 있는데 NPN_GetURL이나 NPN_PostURL 등이 그 대표적인 예이다. PGP 모듈을 이용해서 암호화 한 데이터들은 NPN_PostURL을 통하여 서버로 전송하며 서버가 생성해서 전송하는 암호화된 응답은 NPP_Write API 함수에서 처리하게 된다. 한편 NPN_GetURL이나 NPN_PostURL의 경우 서버로부터 전송되어 오는 응답을 직접 브라우저 화면으로 디스플레이 할 수 있는 기능을 제공하고 있다. 그러나 전송되어 오는 데이터에 대한 추가적인 처리 루틴이 필요한 경우에는 이러한 기능은 고려될 수 없으며 일단 임시 화일에 저장한 후 필요한 데이터 처리를 수행하고 나서 브라우저 화면에 디스플레이 하여야 한다.

2. *Java Runtime Interface (JRI)*

Java Runtime Interface(JRI)[7]란 Java와 C/C++ 형태의 Plug-in 사이의 상호 인터페이스를 정의한 것으로 Java에서 사용되는 데이터 타입과 C/C++ 에서 사용되는 데이터 타입의 연결 및 변환 등의 기능을 수행한다. Java 프로그램의 입장에서 보면 Plug-in은 자신과 형식이 다른 일종의 Stub 화일로 간주되므로 JRI를 사용하지 않는 한 직접적으로 Java 프로그램과 링크 될 수 없다. JRI에서 제공하는 기능을 살펴보면 다음과 같다.

- Native method와 JRI를 지원하는 Java platform 사이의 연동
- Native method에 영향을 끼치지 않는 Java 클래스의 수정

Java 클래스를 정의하는데 있어서 위와 같은 JRI 기능을 제공받기 위해서는 JRI 버전의 Java 컴파일러를 사용하여야 하는데 Java 클래스의 컴파일이나 Stub 화일의 생성에 있어서도 -jri, -stub 와 같은 특별한 옵션을 지정 함으로써 JRI 기능을 제공 받을 수 있다. Java는 사용자 Event의 처리나 Interactivity 측면에서 C/C++ 형태의 Plug-in 프로그램 보다 손 쉽게 처리할 수 있는 장점을 가지고 있다. 예를 들어 HTML 문서 안에 존재하는 FORM Tag에서 지원하는 텍스트 입력이나 아이템 리스트, 버튼 클릭 등의 기능은 Java 프로그램과 손쉽게 연결될 수 있으나 Navigation 기능의 지원이나 기존의 C/C++로 작성된 프로그램과의 연동 등의 측면에서 약점을 가지고 있다. 따라서 서버에 데이터를 전송하거나 전송받는 기능, 혹은 PGP 모듈과의 연동 등은 Plug-in 모듈을 이용하는 것이 바람직하기 때문에 JRI를 이용한 Java와 Plug-in 사이의 연동은 필수적인 요소 이다.

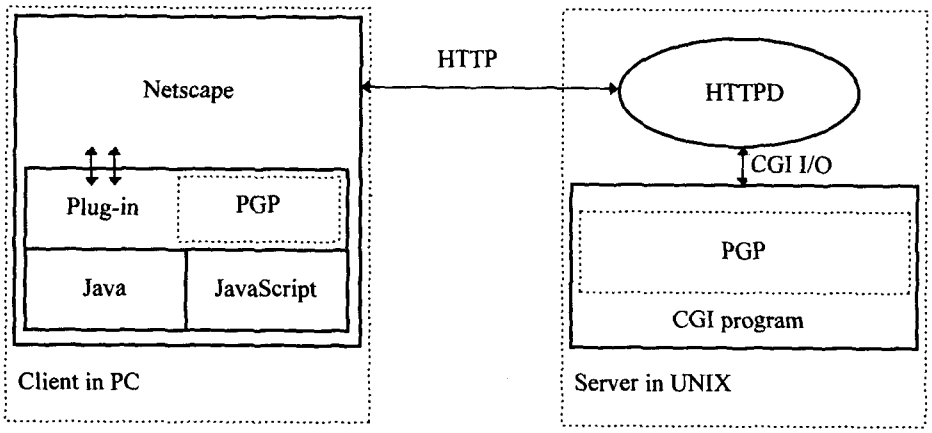
3. PGP 기반의 WWW 보안 메커니즘

공개 키 시스템은 관용 암호 방식의 몇 가지 문제점들을 해결하기 위한 시도로써 발전된 개념이다. 여기서 말하는 문제점이란 크게 두 가지로 대표될 수 있는데 그 중 첫번째 문제는 키 분배의 문제이고 두 번째는 전자 서명에 관한 문제이다. 이러한 문제점이 발생하는 근본 원인은 암호화 당시에 사용되는 키를 가지고 복호화를 수행 하여야 한다는 점이다. 한 사람의 사용자가 자신의 시스템에서 화일을 암호화 해두는 경우 이러한 형식의 관용 암호 방식은 가장 효율적인 방법이다. 그러나 암호화된 데이터가 다른 누군가에게 전송되어 복호화 되어야 하는 경우에는 관용 암호 암호 방식의 최대 문제점이라 할 수 있는 키 분배 문제가 발생 한다[8][9]. 즉 관용 암호 방식을 이용할 경우 송신자와 수신자가 알고 있는 키가 동일해야 한다는 데에서 발생하는 문제점 이다. 특히 송신자와 수신자가 서로를 모르는 상황에서의 데이터 암호화 전송의 경우 송신자는 자신의 키를 수신자에게 인지 시켜야 한다는 부담을 안게 된다. 이러한 키 공유에 관한 문제점은 공개 키 암호 방식의 사용으로 쉽게 해결 될 수 있다. 공개 키 암호 방식이란 암호화에 사용되는 키를 2 가지를 정의하는데 그 중 키 소유주 만이 알고 있는 키를 비밀 키라 하고 다른 사람들에게 공개 하는 키를 공개 키라 한다. 공개 키로 암호화된 데이터는 비밀 키로 복호화 될 수 있으며 반대로 비밀 키로 암호화된 데이터는 공개 키로 복호화 될 수 있다. 따라서 데이터 암호화 전송에 있어서 공개되어 있는 수신자의 공개 키를 이용해서 암호화 한 후 전송 할 경우 비밀 키를 알고 있는 수신자만이 암호화된 내용을 복호화 해 낼 수 있다.

두 번째로 문제가 되는 관용 암호 방식의 단점은 전자 서명 기능[8][9]을 지원하지 않는다는 점이다. 전자 서명이란 송신자의 전송 부인이나 수신자의 수신 부인을 방지하기 위한 방안으로 말 그대로 자신임을 입증하는 서명을 데이터 전송에 추가하는 것이다. 관용 암호 방식을 이용할 경우 송신자와 수신자가 하나의 키를 공유하기 때문에 송신자와 수신자를 분명히 구분해 낼 수 있는 방법이 존재하지 않는다. 그러나 공개 키 시스템의 경우 암호화 할 데이터와 더불어 그 데이터에 대한 해쉬 값을 송신자의 비밀 키로 암호화하여 전송할 경우 위와 같은 부인 방지에 관한 문제는 해결 될 수 있다. 즉 송신자가 자신이 보낸 데이터가 아님을 주장 할 경우 송신자만이 알고 있는 비밀

키로 데이터를 암호화 한 결과가 수신자 측에서 수신한 송신자의 서명과의 일치 여부를 확인하면 된다. 따라서 본 논문에서는 공개 키 암호 방식에 근거한 Web 보안 시스템을 구현 함으로써 위와 같은 문제점들을 해결 하고자 한다.

<그림 4>는 공개 키 암호 방식을 이용한 Web 보안 시스템의 전체적인 구성도 이다.



<그림 4> PGP 기반의 Web 보안 시스템의 구성도

공개 키 암호 방식의 도입을 위하여 PGP 툴을 사용하며 2 장에서 기술된 LiveConnect 기법을 이용하여 Netscape과 암호화 모듈과의 데이터 교환을 구현 하였다. 전체적인 구조를 살펴보면 클라이언트의 경우 Plug-in과 Java/JavaScript로 구성된 암호화 Application 모듈이 Netscape에 Embedding 되는 형태이다. Plug-in과 Java/JavaScript 사이의 데이터 교환을 위하여 2 장에서 기술된 LiveConnect 기법이 사용된다. 암호화 Application 모듈 중 Plug-in 부분은 데이터 암호화를 수행하기 위하여 PGP 모듈을 포함하며 Netscape와의 실질적인 데이터 교환과 제어 기능을 수행한다. 서버는 HTTPD와 PGP 모듈을 포함하는 CGI 프로그램으로 구성되며 UNIX 시스템 상에서 동작한다. 클라이언트와 서버 사이의 데이터 전송은 HTTP 프로토콜을 그대로 이용하며, 물론 암호화를 기본으로 한다. 이를 위하여 클라이언트와 서버 시스템에는 각각 PGP 툴이 설치되어 있어야 하는데 이는 본 Web 보안 시스템에 포함된 암호화 모듈이 시스템에 설치된 PGP 툴을 그대로 이용하기 때문이다.

3.1 PGP (Pretty Good Privacy)

전자 우편은 모든 분산 환경에서 사용하는 네트워크 기반 응용 프로그램 중 가장 보편적으로 사용되고 있는 서비스 이다. 전자 우편이라고 하는 것이 개인의 사생활에 관한 내용이 대부분이기 때문에 이를 허가 받지 않은 사용자로부터 보호해야 하는 것은 당연한 일이다. PGP는 Phil Zimmermann에 의해서 개발된 전자 우편 보안 도구로써 RSA라는 공개 키 암호 방식에 근거 한다. 따라서 공개 키 시스템의 특성을 그대로 제공하고 있을 뿐 아니라 소스 코드가 공개되어 있기 때문에 다른 응용 분야에 효과적으로 이용될 수 있다. 더구나 현재 암호학 분야의 초 강대국이라 할 수 있는 미국에서 암호학에 관련된 제품의 수출을 규제하는 점을 감안한다면 이러한 소스 코드의 공개는 상당한 매력을 갖는다. 한편 PGP는 암호학적인 측면에 있어서도 상당히 잘 정립된 구조를 가지고 있는데 이러한 PGP의 구성 요소 및 기능을 살펴보면 <표 1>과 같이 정리될 수 있다.

기능	사용 알고리즘	기능 설명
메시지 암호화	IDEA, RSA	EIKs[D] || EUb[Ks]
전자 서명	RSA, MD5	ERa[D || H[D]]
메시지 압축	ZIP	ZIP[D]
전자 우편 호환성	Radix 64 Conversion	Radix[D]
단편화		Segmentation and Reassembly

<표 1> PGP의 구성 요소 및 기능

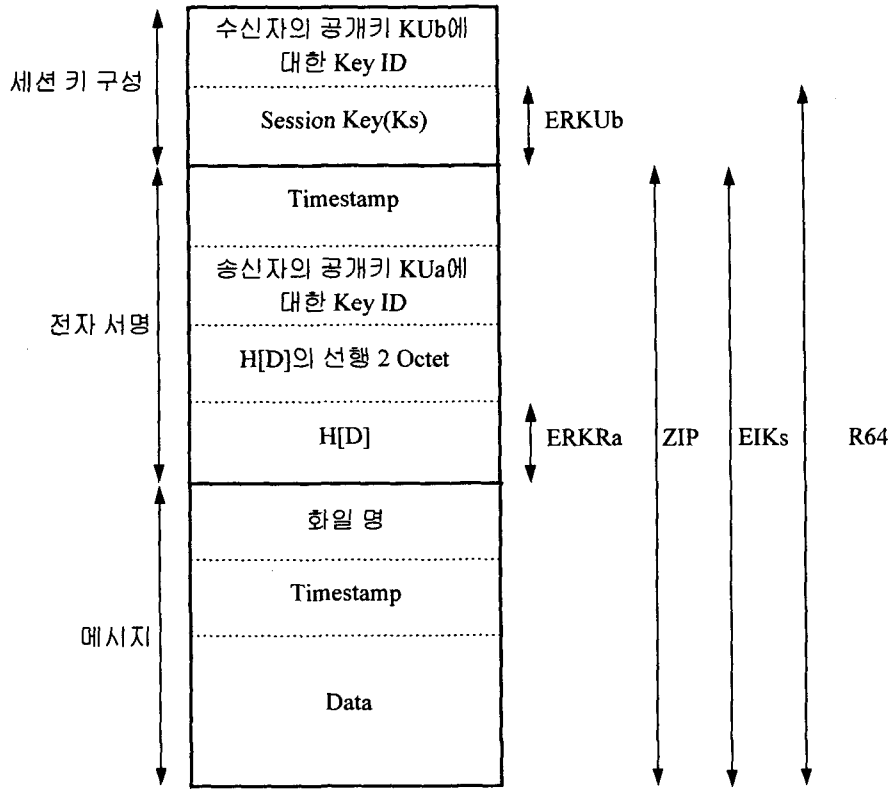
<표 1>에서 사용된 수식과 앞으로 사용될 수식들의 의미에 대하여 살펴보면,

- Ks = 관용 암호 방식(IDEA)의 암호화에 사용되는 128 비트 세션 키
- KRa = 사용자 A의 비밀 키
- KUa = 사용자 A의 공개 키
- ER = 공개 키 방식 (RSA)을 이용한 암호화
- DR = 공개 키 방식 (RSA)을 이용한 복호화
- EI = 관용 암호 방식(IDEA)를 이용한 암호화
- DI = 관용 암호 방식(IDEA)를 이용한 복호화
- H = 해쉬 함수, MD5
- ZIP = PKZIP 압축 알고리즘
- R64 = Radix 64 변환 알고리즘
- || = 메시지의 연결

와 같이 표현 될 수 있다. 여기서 메시지 암호화를 위해서 IDEA 알고리즘을 사용하는데 그 이유는 공개 키를 이용한 암호화는 수행 시간이 오래 걸린다는 단점을 보완하기 위해서 이다. 위에 표현된 기능들은 대개의 경우 단독적으로 사용되지 않고 서로 조합된 형태로 사용된다. 예를 들어 전자 서명을 붙인 데이터의 암호화 및 메시지 인증의 경우 다음과 같이 표현 될 수 있다.

*EIKs[**ERa[D || H[D]]**] || EUb[Ks]*

여기서 굵은 선으로 표현된 부분은 <표 1>의 메시지 암호화 기능에 사용된 데이터(D)에 해당한다. 한편 데이터의 암호학적인 처리 이외에도 E-mail 시스템이 ASCII 코드만을 전송 한다는 점을 감안하여 Radix 변환을 수행함으로써 이진 형태의 데이터를 ASCII 형태로 변환하는 기능을 제공하며 ZIP 형태의 압축 기법을 사용 함으로써 데이터 전송 시의 효율을 극대화 시킬 수 있다. 한편 PGP를 이용해서 전송되는 메시지의 구성 및 적용 기능을 살펴보면 <그림 5>과 같다. PGP 메시지 전송의 메시지 구조를 살펴보면 IDEA를 이용한 화일의 암호화와 RSA를 이용한 세션 키 전달과 전자 서명, MD5를 통한 메시지 인증 및 기타 압축과 데이터 포맷 변환 등의 기능들을 적절히 조합하여 사용해서 데이터를 허가 받지 않은 사용자로부터 보호한다는 것을 알 수 있다. 결과적으로 공개 키의 장점과 관용 암호 방식의 장점을 결합시킨 형태이다.



<그림 5> PGP 메시지 구성

3.2 Netscape and Plug-in Methods

Netscape Plug-in은 Plug-in Method와 Netscape Method로 구분되는 총 28 가지의 API 함수를 제공하고 있다. Plug-in Method의 경우 총 14 가지의 API 함수로 구성되는데 "NPP_*APIName*" 형식의 함수 명을 가지며 주로 Plug-in 프로그램 내에서 호출된다. 반면에 Netscape Method의 경우 총 14 가지의 API 함수로 구성되는데 "NPN_*APIName*" 형식의 함수 명을 가지며 주로 Netscape에 의해서 호출된다. <표 2>는 Netscape과 Plug-in Method API 함수들을 나열한 것이다.

	Plug-in method	Netscape method
1	*NPP_Destroy*	*NPN_DestroyStream*
2	*NPP_DestroyStream*	*NPN_GetJavaEnv*
3	*NPP_GetJavaClass*	*NPN_GetJavaPeer*
4	*NPP_HandleEvent*	*NPN_GetURL*
5	*NPP_Initialize*	*NPN_MemAlloc*
6	*NPP_New*	*NPN_MemFlush*
7	*NPP_NewStream*	*NPN_MemFree*

8	*NPP_Print*	*NPN_NewStream*
9	*NPP_SetWindow*	*NPN_PostURL*
10	*NPP_ShutDown*	*NPN_ RequestRead*
11	*NPP_StreamAsFile*	*NPN_ Status*
12	*NPP_URLNotify*	*NPN_ UserAgent*
13	*NPP_Write*	*NPN_Version*
14	*NPP_WriteReady*	*NPN_Write*

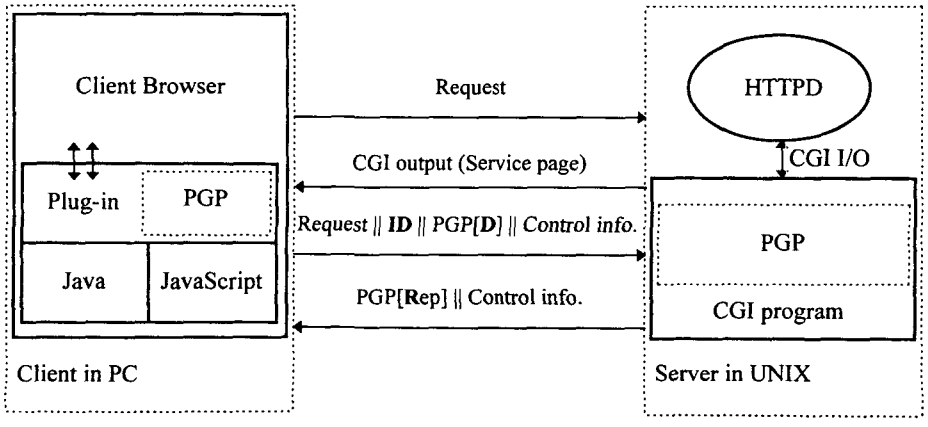
<표 2> Netscape Plug-in Methods

3.3 Implementation

공개 키 암호 방식에 기초한 Web 보안 시스템은 PGP 툴을 암호화 모듈로 사용하는 보안 시스템으로 대표적인 클라이언트 & 서버 모델이다. 클라이언트 는 PC(Personal Computer)를 대상으로, 서버는 UNIX 시스템을 대상으로 구현되었으며 Web 보안 시스템의 구현과 수행을 위한 시스템 환경을 살펴보면 다음과 같다.

- Windows 95, for client's operating system
- Visual C++ 4.0, for plug-in DLL
- PGP version of 2.6.3i
- Solaris 2.5, for server's operating system
- NCSA HTTPD version 1.5.1, for server HTTP daemon
- GCC compiler, for CGI compiling

본 논문에서 제시된 PGP 기반의 보안 시스템은 클라이언트의 화일을 서버로 전송하는 기능과 서버의 화일을 클라이언트로 전송 받아오는 기능을 제공할 뿐 아니라 사용자가 브라우저 상에서 입력한 데이터를 서버로 전송하는 기능도 제공한다. 사용되는 MIME 타입은 "application/netpgp" 이며 DLL 프로그램 내부에 지정되어 있어 사용자가 따로 지정할 필요가 없으며 브라우저가 구동 될 때 자동적으로 등록된다. <그림 6>은 클라이언트가 서버로 암호화된 데이터를 전송하는 과정을 나타내고 있다.



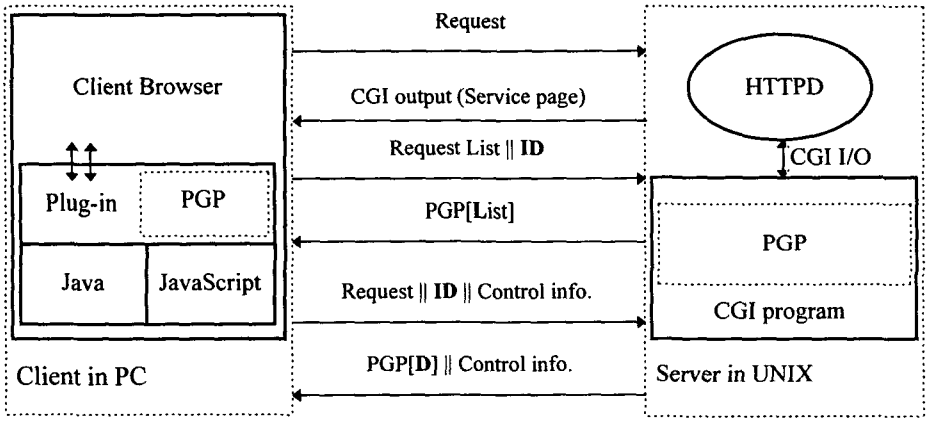
<그림 6> 클라이언트에서 서버로의 데이터 전송

관용 암호 시스템과의 가장 큰 차이점을 든다면 데이터 교류 이전에 세션 키의 공유 과정이 없다는 것이다. 클라이언트가 서비스를 요구하면 서버는 서비스를 제공하는 HTML 문서를 전송한다. 이 HTML 문서에는 사용자가 입력하는 데이터를 서버에 전송하는 기능이나 클라이언트의 화일을 서버로 전송하는 기능, 서버의 화일을 클라이언트로 받아오는 기능 등을 제공한다. 만일 사용자 입력 데이터를 전송하는 경우에는 HTML 문서에 입력된 데이터들을 PGP로 암호화 한 후 제어 정보와 함께 전송을 한 후 서버의 응답을 기다리게 된다. 서버는 전송되어온 데이터를 복호화한 결과가 성공적일 경우 인증된 사용자임을 알리는 메시지와 함께 보안이 요구되는 내용을 역시 PGP로 암호화해서 클라이언트로 전송한다. 제어 정보로는 사용자 ID가 포함될 수 있는데 여기서 사용되는 사용자 ID는 PGP에서 사용되는 사용자 ID와 동일하다. 한편 이러한 메커니즘은 클라이언트와 서버가 서로 상대방의 공개 키를 알고 있어야 한다는 것을 전제로 한다. 클라이언트가 자신의 화일을 암호화해서 전송하는 과정은 위에서 설명한 과정과 유사하지만 데이터 암호화와 전송 과정이 반복되며 전송 제어 정보에 화일 명이나 전체 화일의 크기, 현재까지 전송된 화일의 양, 전송 상태 등의 정보를 포함한다. 또한 Plug-in 모듈에서의 화일의 분할 과정과 CGI 프로그램에서의 화일의 병합 과정이 추가된다. 한편 전송되는 데이터, PGP(D)의 구조를 살펴보면 다음과 같이 표현될 수 있다.

EIKs[D || ERKRc[H[D]]] || ERKUs[Ks]

위의 수식이 의미하는 바를 살펴보면, ERKRc[H[D]]]를 통하여 메시지 인증과 전자 서명을 지원하며 EIKs[D || ERKRc[H[D]]]를 이용해서 기밀성을 보장한다. IDEA 암호화에 사용된 세션 키를 알리기 위하여 ERKUs[Ks]를 사용 함으로써 서버만이 세션 키를 복호화 해낼 수 있도록 한다.

<그림 7>은 서버로부터 화일을 전송 받는 과정을 설명하고 있다. <그림 6>에서 설명한 화일 전송의 경우와 거의 유사한 형태로 전송이 진행되지만 암호화/복호화의 주체가 다르다는 차이점이 있다.



<그림 7> 서버에서 클라이언트로의 데이터 전송

화일의 수신 메커니즘에 있어서는 클라이언트 ID(PGP ID)에 따른 접근 제어 메커니즘이 첨가된다. 즉 각각의 클라이언트에게 서버에 등록된 모든 화일에 대한 접근을 허용 하는 것이 아니라 사용자별 권한에 따라 접근을 허용한다. 따라서 클라이언트의 화일 수신 요구를 받은 서버는 접근 제어 모듈을 참조하여 해당 클라이언트가 수신 할 수 있는 화일의 리스트(List)를 전송하며

클라이언트는 이 중의 하나의 화일을 선택해서 수신하게 된다. 기본적으로 클라이언트는 자신이 전송한 화일이나 공유 디렉토리에 저장되어 있는 화일에 대해서만 접근 권한을 갖지만 서버 시스템 관리자에 의해 추가적인 권한을 할당 받을 수 있다. 이후의 전송 메커니즘은 화일 송신의 경우와 유사하게 진행된다.

4. 결론

데이터 암호화 기법은 크게 관용 암호 방식과 공개 키 암호 방식으로 구분 될 수 있다. 관용 암호 방식을 이용하는 경우에는 암호화 속도상의 장점과 키 생성이 쉽다라는 장점이 있지만 송신자와 수신자가 키를 공유해야 한다는 단점이 있다. 더구나 송신자나 수신자의 데이터 송수신 부인에 관한 부인 봉쇄 기능을 제공하고 있지 않는다는 단점이 있다. 이에 비해 공개 키 기반의 암호 시스템의 경우 암호화 속도가 느리며 키 생성이 어렵다는 단점이 있지만 송신자와 수신자가 키를 공유할 필요가 없다는 장점이 있고 부인 봉쇄 기능을 제고한다. 본 논문에서 제시된 WWW 보안 시스템은 공개 키 알고리즘을 기반으로 하는 전자우편 보안 툴인 PGP를 사용하여 구현하였다. 따라서 WWW 보안 요구 사항에서 정의하는 데이터의 기밀성, 무결성, 사용자 인증, 부인 봉쇄 기능들을 모두 제공하고 있다.

이러한 보안 모듈을 WWW에 적용 시키기 위해서는 암호화 Application 프로그램과 브라우저 사이의 데이터 교환을 위한 메커니즘이 요구된다. 이를 위하여 본 논문에서는 Netscape에서 개발한 LiveConnect 기법을 사용 함으로써 WWW 보안 모듈을 브라우저 내에 Embedding 하여 사용한다. 즉 데이터 암호화가 요구되는 MIME 타입이 브라우저에 전달되면 암호화/복호화를 담당할 Application 프로그램이 수행되어 서버 측의 CGI 프로그램과 대응되어 암호화/복호화를 수행한다. 이러한 형태의 암호화는 Mosaic 브라우저에서 제공되는 CCI 프로그램을 이용한 암호화 방안과 유사하다. 한편, 본 논문에서 제안된 WWW 보안 시스템의 특징을 정리 하자면,

- HTTP나 그 하위 프로토콜의 수정없이 수행
- 데이터 전송 이전에 세션 키 공유 메커니즘이 필요 없다는 점
- 데이터 암호화/복호화 기능 이외에 메시지 인증, 부인 봉쇄 기능 제공
- Interactive한 형태의 사용자 데이터 암호화에 쉽게 적용 가능
- 특정 폐쇄 그룹내의 정보 공유 시스템 (화일 송수신)
- 정보 공유에 있어서 사용자별 접근제어 메커니즘 제공
- 가장 대중적인 Web 브라우저인 Netscape을 대상

등으로 요약될 수 있다. 위와 같은 특성은 전자 상거래나 EDI 시스템, 인트라넷 보안 모듈과 같은 특정 서비스 및 그룹 관리에 적합한 보안 시스템으로 의의를 가질 수 있다.

References

1. E. Rescorla, "Secure HTTP", http://www.eit.com/creations/s-http
2. Kipp E.B. Hickman, "The SSL protocol", http://www.netscape.com/newsref/std/SSL.html
3. Phillip Zimmermann, "PGP:Pretty Good privacy", O'Reilly & Associates, Inc.
4. Phillip Zimmermann, "PGP:Source Code and Internals", The MIT Press.
5. Gray Cornel, "Core Java", The SunSoft Press.
6. Netscape Corp., "The LiveConnect/Plug-in Developer's Guide", http://www.netscape.com/eng/mozilla/3.0/handbook/plugins/index.html
7. Netscape Corp., "The Java Runtime Interface", http://www.netscape.com/eng/jri/

8. William Stallings, "Network and Internetwork Security", Prentice Hall International Edition.
9. Bruce Schneier, "Applied Cryptography", John Wiley & Sons, Inc.