자바 애플릿을 사용한 사용자 인증 및 안전한 통신 시스템 설계 및 구현

서정우*, 손태식*, 구원본*, 문종섭*

*고려대학교 정보보호대학원/정보보호기술연구센터

Implementation and Design a User Authentication and Secure Communication System using Java Applets

Jungwoo Seo, Taeshik Sohn, Won bon Koo, Jongsub Moon*

*CIST/GSIS, Korea University.

요 약

최근 인터넷 사용이 보편화 되면서 웹을 통하여 다양한 서비스가 제공되면서 웹 보안에 대한 중요성이 증가하고 있으며 실제로 여러 방법들이 제안되고 구현되었다. 하지만 현재의 웹 보안 방식들은 벤더의 웹 브라우저나 프로토콜을 수정해야 하는 문제점이 존재한다. 본 논문에서는 사용자 인증과 안전한 통신을 수행하기 위하여 자바 애플릿을 사용함으로써 벤더의 웹 브라우저나 프로토콜을 수정하지 않으면서 사용자 인증 및 안전한 통신을 수행할 수 있도록 한다.

I. 서론

최근 인터넷 상용이 보편화되면서 웹을 이용한 광고, 전자상거래, 인터넷 뱅킹 등 다양한 서비스가 네트워크를 통해 제공되면서 웹 보안에 대한 필요성이 증가하고 있다. 특히 인터넷 뱅킹이나 인터넷 쇼핑몰을 이용하는 경우 인터넷을 통하여 사용자의 신용카드 번호나 개인 정보가 불법적으로 유출되거나 위.변조될 가능성이 있다. 이러한 문제들은 프로토콜 레벨(e.g., HTTP/TCP/IP/)에서 데이터에 대한 보안 서비스를 제공하지 않기 때문이다. 그러므로 WWW(World Wide Web)환경에서 안전한 데이터 전송을 위해서는 인증, 기밀성, 무결성, 부인방지 등의 보안 서비스를 제공해야 한다. 일부 벤더나 표준화 그룹들에서는 이러한 문제들에 대한 해결책들을 제시하고 있지만 아직까지 명확한 만족을 주지는 못하고 있다.

본 논문에서는 기존의 HTTP(Hypertext Transfer Protocol)에서 자바 애플릿을

실행 시킴으로써 HTTP 프로토콜을 수정하거나 기존의 웹 브라우저를 새롭게 개발하지 않으면서 암호화 통신을 수행할 수 있도록 하였다. 클라이언트에 설치되어 있는 자바 애플릿과 어플리케이션을 이용하여 웹 브라우저 밖에서 사용자 인증 및 암호화를 통한 데이터 전송을 수행하는 시스템을 구현하였다[1][2][3].

II. 관련 기술

현재 안전한 HTTP 통신을 위한 많은 방법들이 제안되었으며 실제로 웹 환경에서 사용되고 있다. 위에서 제안된 여러 방법들을 분류하면 다음과 같다.

표 1. 안전한 HTTP 통신 방법 분류

적용 계층	적용 방법	적용 예제
어플리케이션 프로토콜 계층	HTTP 프로토콜이 키 관리, 암호화, 서명 등의 작업을 할 수 있도록 프로토콜의 헤더를 수정한다.	S-HTTP
세션 계층	수정된 트랜스포트 API의 상위에서 구현되며, 양단간 인증 및 기밀성을 보장한다.	Secure Socket Layer(SSL)
전송 계층	IPv6, PPTP, SSH와 같은 터널링 프로토콜을 사용한 TCP 접속에 대한 안전성을 제공한다.	PPTP SSH VPN
어플리케이션 계층	TCP 레벨의 소켓 API나 HTTP 프로토콜을 변경하지 않으면서 다른 어플리케이션 프로그램을 사용하여 안전한 TCP/IP 통신을 수행한다.	PGP

III. 시스템 구현

1. 시스템 동작 시나리오

클라이언트와 서버 사이에 세션 키를 공유함으로써 안전한 정보 전송을 수행함과 동시에 두 시스템에 대한 상호 인증을 통하여 신뢰성을 향상을 목적으로 한다. 클라이언트의 로컬 애플릿과 어플리케이션은 사전에 웹 서버로부터 다운로드 받아 설치하도록 한다.

1) 클라이언트와 서버 연결 설정

서버와 클라이언트 사이에 상호인증 및 키 교환을 위하여 소켓연결을 수행해야 한다. 이를 위하여 클라이언트는 (그림 1)과 같은 과정을 수행한다.

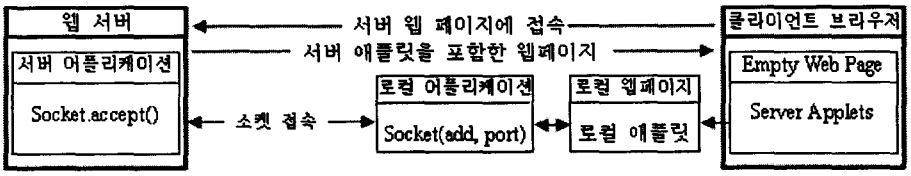


그림 1. 클라이언트와 서버 연결 설정

(그림 1)의 과정을 단계적으로 살펴보면 다음과 같다.

① 서버의 정보를 요청하기 위하여 브라우저에 서버 URL 주소를 입력한다.

② 서버 애플릿이 클라이언트에 다운로드와 동시에 클라이언트 시스템의 로컬 웹 페이지와 로컬 어플리케이션을 실행한다.

③ 로컬 어플리케이션이 실행 되면서 서버 어플리케이션과 소켓 접속을 한다.

④ 로컬 애플릿은 사용자와의 인터페이스 역할을 담당하며, 로컬 어플리케이션은 서버와 어플리케이션과 통신을 실행한다.

실제적으로 네트워크상의 통신은 클라이언트의 로컬 어플리케이션과 서버의 원격 어플리케이션이 수행하게 된다.

2) Diffie-Hellman 방식의 키 교환 및 사용자 인증

클라이언트가 서버에 접속하면 Diffie-Hellman 방식의 키 교환 알고리즘을 수행한다. (그림 2)는 키 교환 알고리즘에 의한 세션키 생성 과정을 보여준다.

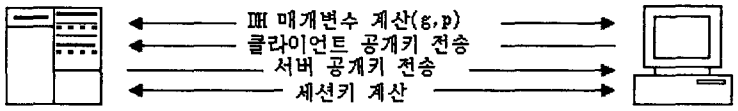


그림 2. Diffie-Hellman 키 교환 방식에 의한 세션키 생성

클라이언트와 서버가 세션키를 공유하게 되면 상호인증을 수행한다(그림 3).

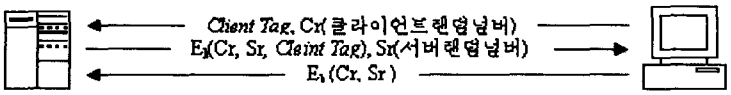


그림 3. 클라이언트와 서버 상호인증

위 과정이 완료되면 송신 측에서 암호화된 데이터를 클라이언트에 전송하고 수신 측은 데이터를 복호화하여 사용자에게 제공 한다.

3) 데이터 암호화에 의한 안전한 통신 채널 구성

안전한 데이터 전송을 위하여 Diffie-Hellman 키 교환프로토콜에 의하여 얻어진 키를 세션키로 사용하였다. (그림 4)는 클라이언트 시스템의 사용자로부터 입력된 메시지가 어떻게 암호화되어 전송되고, 서버에서 복호화 되는지를 보여준다.

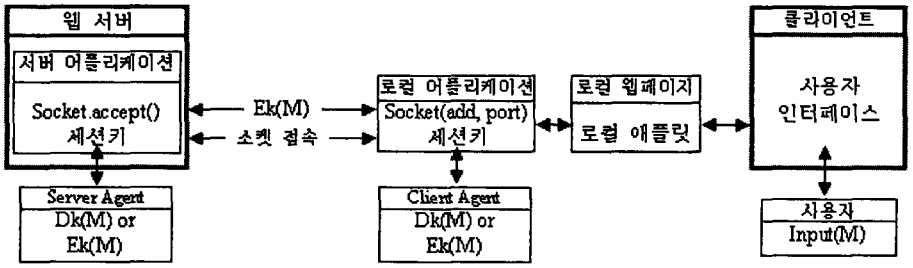
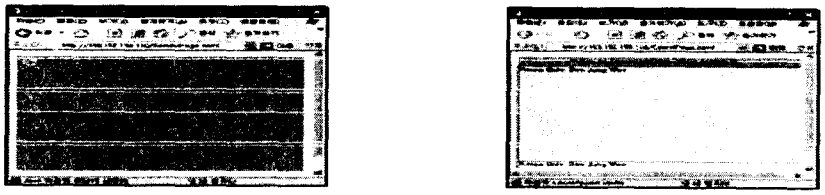


그림 4. 안전한 통신 채널 구성

3. 시스템 구현 및 실행 화면

(그림 5)는 사용자가 서버의 하이퍼링크나 URL 주소를 입력하면 (그림 5(a))가 다운로드 되면서 클라이언트의 로컬 애플릿(그림 5(b))을 호출한다. 동시에 클라이언트의 로컬 어플리케이션이 실행되면서 서버의 어플리케이션과 소켓 연결을 시도한다. (그림 6)과 같이 성공적으로 접속이 이루어 지면 DH 방식에 의한 키 교환과 함께 세션키를 생성한다. 생성된 세션키는 데이터의 암.복호화에 사용된다.

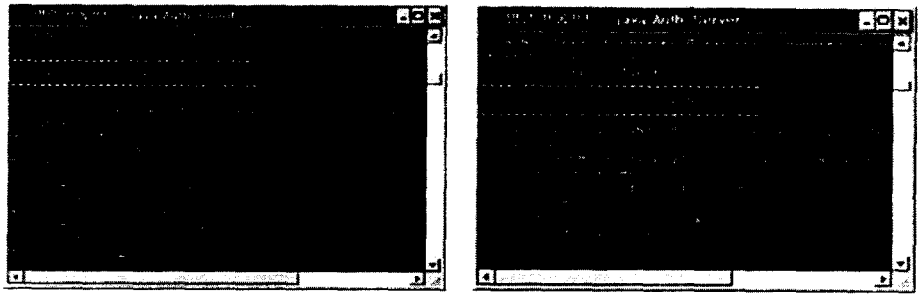
① 서버 애플릿에 의한 로컬 웹 페이지 호출



(a) 서버 애플릿 (b) 로컬 애플릿

그림 5. 서버 애플릿 및 로컬 애플릿 실행화면

② 클라이언트와 서버 세션키 생성 및 암호화된 메시지 전송 화면



(a) 클라이언트 어플리케이션 실행화면 (b) 서버 어플리케이션 실행화면

그림 6. 클라이언트와 서버 어플리케이션 실행 화면

IV. 결론

인터넷의 보편화는 오프라인에서 이루어지던 많은 일들을 온라인 상에서 가능하도록 하였다. 하지만 온라인을 통한 전자상거래나 인터넷 뱅킹 또는 기업의 기밀 정보들의 전송을 위해서는 송.수신자 사이에 인증 및 기밀성, 무결성, 부인방지 등을 제공해야 한다.

본 논문에서는 위와 같은 취약점을 보안하기 위하여 클라이언트의 웹 브라우저에 포함된 자바 애플릿과 자바 어플리케이션 프로그램을 사용함으로써 서버와 안전한 통신을 위한 채널을 생성하도록 하였다. 결국 클라이언트와 서버 사이에

상호인증을 수행함과 동시에 암호화된 메시지를 전송함으로써 인터넷 통신에서 발생할 수 있는 취약점을 해결 하도록 하였다.

참고문헌

[1] F.Bergadano, B. Crispo, M. Eccettuato "Secure WWW Transactions Using Standard HTTP and Java Applets", 3rd USENIX Workshop on Electronic Commerce, 1998, pp. 109-119.

[2] Whitfield Diffie, Paul C. van Oorschot and Michael J. Wiener "Authentication and Authenticated Key Exchanges", Designs Codes and Cryptography, 1992, pp. 107-125.

[3] Charlie Lai Li Gong, Larry Koved, Anthony Nadalin, and Roland Schemers "User Authentication and Authorization in the Java Platform", 15rh ANNUAL Computer Security Applications Conference, 1999.

[4] Mark Crosbie, Ivan Krsul, Steve Lodin, Eugene H. Spafford "A Secrue Message Broadcast System(SMBS)", CSD-TR-96-019, 1997.

[5] A. Freier, P. Karlton, and P. Kocher, "The SSL Protocol Version3", 1995.

[6] T. Dierks and C. Allen., "*The TLS Protocol, version 1.0*", Internet Engineering Task Force Internet Draft, 1997.

[7] Paul Ashley, Gary Gaskel, Joris Claessens and Mark Vandenwauver., "Intranet Security technologies – Sesame or SSL?", Proceedings of the AUUG'98 Conference Sydney, 1998, pp. 133-142.

[8] Joris Claessens, Bart Preneel and Joos Vandewalle., "Secure communication for secure agent-based electronic commerce applications", LNAI 2033, 2001, pp. 180-190.

[9] http://java.sun.com/j2se/1.4.1/docs/guide/security/