보안토큰을 이용한 웹 보안 시스템 개발

하 경 주† · 윤 재 우†† · 강 창 구††† · 장 승 주††††

요 약

본 논문에서는 인터넷에서의 중요 정보 유출에 따른 여러 가지 문제점을 해결하기 위하여 인터넷의 웹 페이지를 이용하여 교환되는 정보에 대한 안전한 전송을 보장하는 시스템을 개발하였다.

개발된 시스템은 기존의 웹 서버나 브라우저를 수정하지 않는다는 가정아래 CGI 기술과 Plug-in 기술, Socket Spy 기술을 이용하여 WWW 서비스에서 안전한 보안 기능을 제공한다. 이 때, 종전에는 소프트웨어적인 방법으로 사용자의 접근을 제한 한 것과는 달리 보안토큰이라는 하드웨어를 통하여 사용자의 접근 제어, 데이터의 암/복호화 등을 수행함으로써 보다 강력한 접근 통제가 가능한 시스템을 개발하였다.

Development of a Web Security System Using Cryptographic Token

Kyeoung-Ju Ha† · Jae-Woo Yoon†† · Chang-Goo Kang††† · Seung-Ju Jang††††

ABSTRACT

In this paper, we develop a security system which enhances the security of information during transmission over the World Wide Web for solving some problems related to outflow of the information on the internet.

Our system provides safe security function without modifying the existing Web server and browser by utilizing CGI, Plug-in, and Socket Spy techniques. Our system implements user access control and data encryption/decryption by using the hardware cryptographic token instead of using a software technique as in previous systems, and hence is a more robust security system.

1. 서 론

인터넷을 이용한 정보의 교환이 날이 갈수록 늘어남에 따라 이에 대한 정보 보호 문제 또한 그 중요성을 더해가고 있는 실정이다. 특히 WWW(World Wide Web)의 개발로 일반 사용자들의 참여 또한 폭증하고 있는 추세여서 SSL(Secure Socket Layer), S-HTTP (Secure HTTP), SEA(Security Extension Architecture) 등 WWW 보안 메커니즘들이 많이 개발되고 있는 실정이다. 하지만 기존의 보안 메커니즘들은 모두 소프트웨어 적인 해결 방식에 의존하고 있어 근본적인 접근 통제가 어렵다는 단점이 있다. 이에 본 논문에서는 하드웨어 보안토큰(cryptographic token)을 이용하여 웹 보안 문제를 해결하는데, 이 때 데이터의 암/복호는 보안토큰 내에 저장된 자체 알고리즘을 사용함으로써, 보안토큰을 지니고 있지 않을 경우에는 복호화가 불가능하도록 한다.

하드웨어 보안토큰을 이용한 웹에서의 보안 기술은 먼저 웹 서버에 PCMCIA(Personal Computer Memory Card International Association) 형태의 보안토큰을 장

† 정 회 원 : 한국전자통신연구원 선임연구원
†† 준 회 원 : 한국전자통신연구원 선임연구원
††† 정 회 원 : 한국전자통신연구원 책임연구원
†††† 정 회 원 : 동의대학교 컴퓨터공학과 교수
논문접수 : 1998년 10월 7일, 심사완료 : 1999년 1월 11일

착한 후 이 토큰에서 제공하는 보안 API(Application Program Interface)인 Crypto-API(이하 CAPI라 칭함)를 이용하여 보안 서비스를 수행한다. 클라이언트에서도 서버의 토큰과 동일한 기능을 수행하는 토큰을 장착 후 역시 이 토큰에서 제공하는 CAPI를 이용하여 보안 서비스를 수행하게 된다. 따라서, 클라이언트가 서버에 접근하기 위해서는 보안토큰을 장착하고 있어야 하고, 보안토큰이 없는 사람은 원천적으로 접근이 봉쇄된다.

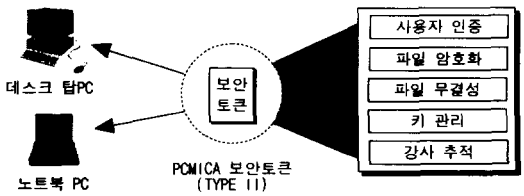
일반적으로 보안토큰은 신용카드 크기의 스마트(smart) 카드와 PCMCIA 카드가 많이 사용되고 있다. 이러한 카드는 내부에 사용자 키(key)와 보안 서비스 관련 정보를 내장하여 암호 알고리즘을 수행하며, 사용자가 필요로 하는 보안 서비스를 제공하게 된다. 일반 신용카드와 같은 종류인 스마트카드는 사용자의 암호 키 및 관련 정보를 카드 내에 보관하며, 소용량의 정보처리 기능을 갖고 있으며, 가격이 저렴하다는 장점을 가지고 있다. 그러나 스마트 카드는 외부 정합을 위한 전송 속도가 매우 낮으며, 8 비트 마이크로프로세서를 채택함으로 인해 계산 능력의 한계가 있다. 이에 비해 PCMCIA 카드는 초고속 시스템에 직접 접속할 수 있으며, 병렬 공유 메모리 정합 방식을 제공한다. 뿐만 아니라 PCMCIA 보안토큰 안에는 초고속 시스템의 데이터 처리 속도와 동일한 성능을 갖는 고속 연산 처리 프로세서를 내장하고 있다[1]. 따라서 PCMCIA 내부 프로세서는 보안토큰 자체만으로도 실시간 보안 서비스를 구현할 수 있다.

본 논문에서는 데이터 파일 보호 및 네트워크 상의 각종 정보를 보호하는데 이용될 수 있는 PCMCIA 보안토큰과 이를 활용하기 위한 보안 API인 CAPI를 개발하였다. 그리고 보안토큰 및 CAPI를 이용하여 웹 상에서 정보를 안전하게 전송 할 수 있는 보안 시스템을 구현하였다[14,15,16,17,18]. 구현된 웹 보안 시스템은 기존의 웹 서버나 브라우저를 수정하지 않고, 웹 서버 측면에서 이용할 수 있는 CGI(Common Gateway Interface) 기술과 클라이언트 측에서 이용할 수 있는 Plug-in 기술, Socket Spy 기술을 이용하여 WWW 서비스에서의 보안 기능을 제공한다[13,21].

본 논문의 구성은 다음과 같다. 2장에서는 보안토큰에 대해 살펴보고, 3장에서는 보안토큰의 기능 및 보안 API에 대해 살펴본다. 4장에서는 개발된 보안토큰을 이용한 웹 보안 시스템 구조 및 동작 원리에 대해 설명하고, 5장에서 결론을 맺는다.

2. 보안토큰

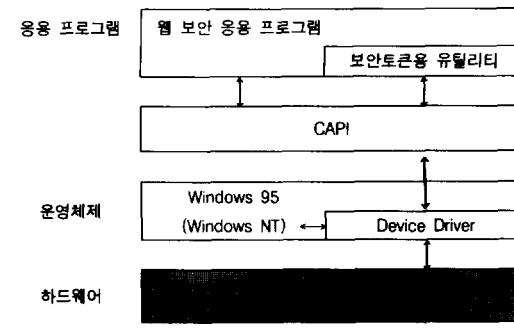
본 논문에서는 보안 시스템의 기본 서비스인 사용자 인증, 무결성, 기밀성 등의 서비스를 제공하기 위하여 데스크 탑 PC 또는 노트북 PC에 장착 할 수 있는 PCMCIA Type II[1]의 보안토큰을 개발하였다. 그리고 이러한 보안 토큰을 이용하여 각종 보안 서비스를 제공하기 위한 보안 API인 CAPI를 설계하였다. 설계된 전체 보안토큰의 개념도는 (그림 1)과 같다.



(그림 1) PCMCIA 보안토큰의 설계 개념도
(Fig. 1) Design Concept of PCMCIA Cryptographic Token

2.1 보안토큰 시스템 구조

보안토큰 시스템은 보안토큰과 이의 사용을 위한 공통 보안 인터페이스인 CAPI 및 보안 기능을 제공하는 각종 유틸리티로 구성된다[22]. 보안토큰은 DSP칩을 CPU로 사용하고 DPRAM을 사용하여 PC와 인터페이스 되며, 보안 API인 CAPI는 Windows 95(혹은 Windows NT)의 PnP 드라이버 기능을 사용하여 보안토큰과의 인터페이스를 담당한다. 이러한 보안토큰 시스템의 전체 구조는 (그림 2)와 같다.

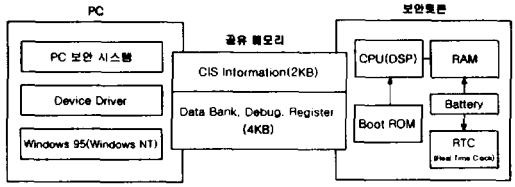


(그림 2) 보안토큰 시스템 구조
(Fig. 2) System Architecture of the Cryptographic Token

2.2 보안토큰 H/W 구조

PCMCIA Specification 2.1(Type II Card) 규격[1]을

준수하는 보안토큰의 개략적인 H/W 구조 및 PC의 보안토큰 인터페이스 구조는 (그림 3)과 같다.

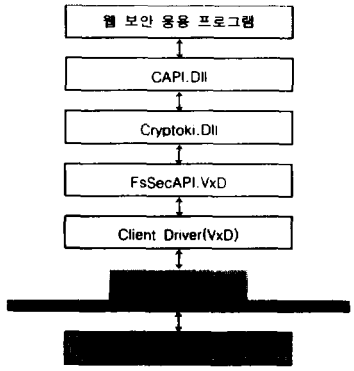


(그림 3) 보안토큰 하드웨어 구조
(Fig. 3) Hardware Architecture of the Cryptographic Token

보안토큰은 CPU, RAM, BOOT ROM, Real Time Clock, 공유 메모리 그리고 배터리로 구성된다. CPU는 TMS320C542(40MHz)가 사용되며, RAM은 보안토큰을 사용하지 않는 경우 보안토큰의 프로그램을 저장하기 위하여 사용된다. Boot ROM은 보안토큰을 부팅하기 위하여, Real Time Clock은 안전한 시간 서비스를 제공하기 위하여 사용된다. 그리고 공유 메모리는 PC와의 메모리 공유를 위하여 배터리는 RAM과 Real Time Clock에 전원을 공급하고 Tamperproof를 위하여 사용된다.

2.3 PC 보안 프로그램 구성

PC의 보안 프로그램 구성은 (그림 4)와 같다. 먼저 Windows 95(Windows NT)의 PnP 드라이버를 이용하여 보안토큰의 공유 메모리를 엑세스 하기위한 클라이언트 드라이버가 있고, 보안카드와의 인터페이스를 위한 FsSecAPI.VxD, Cryptoki.Dll과 CAPI.Dll이 있다. 여기서 FsSecAPI.VxD, Cryptoki.Dll과 CAPI.Dll이 표준 보안 API 기능을 수행한다[3,9,11,12].



(그림 4) PC 보안 프로그램 구성
(Fig. 4) Structure of the PC Security Program

3. 보안토큰 기능

3.1 보안 기능

보안토큰은 한 명의 사용자가 보안토큰을 사용하는 환경을 대상으로 하여, API의 표준인 RSA사의 Cryptoki (Cryptographic Token Interface, in PKCS#11)에 근거하여 개발하였다[10]. 그리고, 사용자를 보안 측면에서 관리 할 수 있도록 관리자의 관리 기능 또한 제공하도록 설계되었다.

PC 시스템을 사용 할 때 보안토큰이 CAPI를 통하여 시스템에 제공하는 보안 기능들은 다음과 같다.

- 사용자 로그인 기능
- 사용자 접근 통제 기능
- 데이터 암/복호화 기능
- 데이터 무결성 확인 기능
- 접근제어 기능
- 키 관리 기능
- 시간 서비스 기능
- 소프트웨어적인 난수 발생 기능
- 보안토큰 관리 기능
- 배터리 백업 회로를 사용한 Tamperproof 기능

3.2 보안 API

CAPI는 모든 종류의 보안 장비(cryptographic devices : PCMCIA 카드나 스마트 카드 등의 휴대 가능한 장비를 포함)가 제공하는 보안기능을 사용하여 응용 프로그램들이 자신의 보안기능을 사용할 수 있도록 공통 보안 인터페이스를 정의한다.

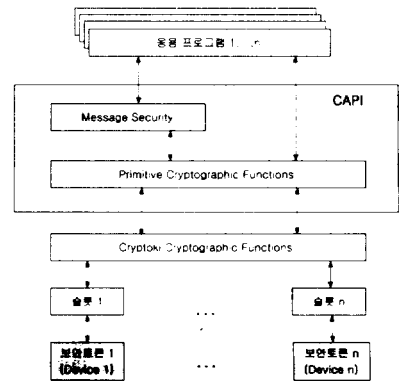
이러한 CAPI는 다음 사항을 고려하여 설계되었다.

- 모든 종류의 보안 장비를 고려한다. 특히 PCMCIA 카드나 스마트 카드 같은 이동성을 가진 휴대 가능한 장비(portable device)를 우선적으로 고려한다.
- CAPI는 보안토큰에 대한 자원 공유(resource sharing)를 고려한다. 다중 오퍼레이팅 시스템(multi-tasking operation systems)에서 여러 응용 프로그램들이 하나의 토큰을 공유하여 사용할 수 있도록 하며, 또한 응용 프로그램들이 동시에 하나 이상의 토큰을 사용할 수 있도록 한다.
- CAPI는 암호학에 전문적인 지식이 없는 프로그래머도 CAPI를 사용하여 보안 서비스를 제공할 수 있도록 generic CAPI를 제공하도록 한다.

- 키나 인증서(certificate) 등 토큰이 관리하여야 할 모든 종류의 데이터 구조를 일반적으로 관리할 수 있는 API 기능을 제공하기위해 단순화된 객체지향 방법론(simple object-based approach)을 사용한다.

CAPI는 크게 두 가지 계층(layer)으로 나누어지며, 하위 계층은 토큰과 직접적인 인터페이스를 제공하는 하위 레벨의 기본 CAPI이고, 상위 계층은 통신환경에서 메시지 보안 서비스를 제공하기 위한 일반적인 CAPI이다. 이러한 점들을 고려하여 하위 계층의 CAPI는 RSA사의 Cryptoki에 근거하여 개발 하였으며[10], 상위계층 CAPI는 Microsoft사의 CryptoAPI의 Certificate Store Functions과 Simplified Cryptographic Function에 근거를 두고 구현하였다[5].

위와 같은 사항을 고려하여 설계된 CAPI의 일반적인 모델은 (그림 5)와 같다. 특정한 암호 연산(cryptographic operations)을 필요로 하는 하나 혹은 복수개의 응용 프로그램이 CAPI를 동시에 사용하며, 이러한 암호 연산들을 하나 혹은 복수개의 토큰이 수행하게 된다. 이 때 CAPI는 시스템에서 사용할 수 있는 하나 혹은 복수개의 슬롯(slot)을 통하여 실제로 암호 연산을 수행하는 토큰과 인터페이스하게 된다. 이 때 각각의 슬롯은 토큰에 대한 인터페이스를 제공하는 물리적인 읽기 장치(physical reader)나 그 밖의 장치 인터페이스이다. 각 토큰에 대한 물리적인 인터페이스는 CAPI가 수행하므로, 응용 프로그램에서는 토큰과 슬롯에 대해 논리적인 관점만을 가지는 것으로 충분하다.



(그림 5) 일반적인 CAPI 모델
(Fig. 5) General CAPI Model

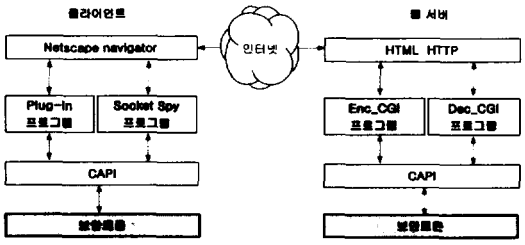
개발된 CAPI의 대표적인 함수는 <표 1>과 같다.

<표 1> CAPI의 대표적인 함수들
<Table 1> Representative Functions of the CAPI

		함 수 명	내 용
Common Functions	General-Purpose	Initialize()	응용 프로그램이 CAPI를 이용한 작업을 시작 할 때 호출하는 함수
		Finalize()	응용 프로그램이 CAPI를 이용한 작업을 종료 할 때 호출하는 함수
	Token Management	GetTokenInfo()	토큰의 정보를 읽는 함수
	Session Management	OpenSession()	토큰에서 정의된 객체나 함수를 사용하기 위한 응용 프로그램과 토큰과의 논리적 연결을 위한 함수
		CloseSession()	토큰에서 정의된 객체나 함수를 사용하기 위한 응용 프로그램과 토큰과의 논리적 연결 해제를 위한 함수
Primitive Cryptographic Function	Cryptographic Function	EncryptMessage()	메시지 암호화 함수
		DecryptMessage()	메시지 복호화 함수
		Digest()	해쉬 연산을 수행하는 함수
		Sign()	메시지에 대한 전자 서명을 수행하는 함수
		Verify()	메시지에 대한 서명 값을 검증하는 함수
	Key management Function	GetMasterKeySet()	마스터 키들에 대한 ID를 검색하여 ID를 되돌려 주는 함수
		GetMasterKeyHandle()	마스터 키ID에 해당되는 마스터키 핸들을 되돌려 주는 함수
		GenerateKey()	비밀키를 새로운 객체로 생성하는 함수
		GenerateKeyPair()	공개키/개인키 쌍을 새로운 객체로 생성하는 함수
		WrapKey()	개인키 혹은 비밀키를 암호화하는 함수
		UnWrapKey()	암호화된 키를 복호화하여 새로운 객체로 생성하는 함수
	Random number generation Function	GenerateRandom()	토큰 내에서 난수를 생성하는 함수

4. 보안토큰을 이용한 웹 보안 시스템

보안토큰을 이용한 웹 보안 시스템은 서버와 클라이언트로 구성되는 분산 시스템 구조로써, 서버에서는 Windows NT를, 클라이언트에서는 Windows 95를 운영체제로 채택하고 있다[12,17,18,19,20]. 전체적인 시스템 동작 원리는 (그림 6)과 같다.



(그림 6) 웹 보안 시스템 구성도
(Fig. 6) Configuration of the Web Security System

4.1 서버와 클라이언트의 동작

먼저 일반적인 웹에서의 자료의 전송은 누군가에 의해 가로막기, 가로채기, 수정, 위조 될 위험성이 있으므로 안심하고 전송 할 수가 없다.

이에 본 시스템에서는 데이터를 안전하게 전송하기 위해, 송신 측에서는 데이터를 암호화 시켜 전송하고, 수신측에서 이를 복호화 시키는 메커니즘을 채택하였다. 이를 위해 서버에서 클라이언트로 데이터 전송 시에는 이를 암호화 시키는 CGI 프로그램을, 서버로부터 암호화된 데이터를 받은 클라이언트에서는 이를 복호화 시키는 Plug-in 프로그램을 각각 개발하여 설치하였다. 또한 클라이언트에서 서버로 데이터 전송 시에는 이를 암호화 시키는 Socket Spy 프로그램을, 클라이언트로부터 암호화된 자료를 전송 받은 서버 측에서는 이를 복호화 시키는 CGI 프로그램을 각각 개발하여 설치하였다.

개발된 시스템에서 서버와 클라이언트간의 데이터 전송은 다음과 같다.

서버에서는 클라이언트에서 데이터 요청이 있을 시, 데이터를 암호화하는 CGI 프로그램을 이용하여 데이터를 암호화하여 클라이언트로 전송한다. 이 때 전송되는 파일은 확장자가 tex인 암호화된 파일이다. 그리고, 암호화된 파일을 받은 클라이언트 측에서는 이를 복호하는 Plug-in 프로그램을 동작시켜 데이터를 원래의 상태로 복호화 시킨 다음 웹 브라우저를 통해 나타낸다.

그리고 실제 웹 환경에서는 클라이언트에서 서버로 데이터를 보내야 하는 경우도 발생하게 되는데, 이는 회원 가입 혹은 물건 구매 등의 경우이다. 이 때 회원의 신상 정보 등과 물건 구입 시 구매자의 카드 정보 등은 보호되어야 할 정보들이다. 이러한 정보들은 클라이언트에서 사용자의 입력 후 서버로 전송되기 직전 Socket Spy 프로그램에서 현재 접속한 사이트가 보안 사이트(웹 서버에서 PCMCIA 보안 토큰을 이용한 보안 기능 지원 가능한 사이트)인 경우 데이터를 가로채기하여 암호화 시킨 후 서버로 전송한다. 즉 Socket Spy 프로그램은 기존의 윈속 프로그램의 기능을 그대로 이용하고, 단지 클라이언트에서 서버에 데이터를 보내는 순간 암호화 기능을 수행한다. 만약 현재 접속한 사이트가 보안 사이트가 아니면 Socket Spy 프로그램은 데이터를 암호화하지 않고 기존의 윈속 기능을 이용하여 서버로 데이터를 전송한다.

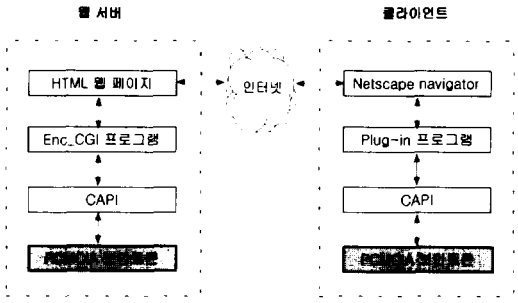
암호화된 자료를 받은 서버에서는 이를 복호화 시킬 수 있는 CGI 프로그램을 동작 시켜 복호화 시킨 후 이를 적당한 형태로 저장하게 된다. 이 때 서버와 클라이언트간에 이동되는 데이터는 누군가에 의해 가로채여도 복호가 불가능하므로 안전한 데이터 전송이 보장되게 된다.

위와 같이 서버와 클라이언트간의 안전한 자료 전송을 위해 데이터를 암/복호화 시켜 주고받게 되는데, 이 때 실제 데이터의 암/복호는 개발된 PCMCIA 보안 토큰의 CAPI와 토큰이 제공하는 자체 암/복호화 알고리즘을 사용하여 하드웨어 보안토큰 내에서 수행된다.

다음 절에서 웹 보안을 위해 서버와 클라이언트에서 각각 개발된 웹 보안 모듈에 대해 살펴본다.

4.2 서버에서 클라이언트로의 데이터 전송

일반적인 웹 환경에서는 서버에서 클라이언트로 데이터를 전송하는 경우가 대부분이다. 이 때 본 시스템에서는 서버에서 클라이언트로 전송되는 데이터를 보호하기 위하여 서버의 보안토큰 내에서 데이터를 암호화하는 기능을 가진 CGI(이하 Enc_CGI라 칭함) 프로그램을 개발하였다. 그리고 클라이언트 측에서는 (그림 7)과 같이 Plug-in 모듈을 통해서 서버로부터 암호화된 데이터를 받은 다음 보안토큰에서 복호시킨 후 브라우저를 통해 화면에 나타내게 된다.



(그림 7) 서버에서 클라이언트로의 데이터 전송모델
(Fig. 7) Data Transfer Model from Server to Client

4.2.1 Enc_CGI

일반적으로 웹 브라우저에서는 HTML로 여러 가지 정보를 처리하지만, 우리가 원하는 모든 기능을 처리하기에는 HTML에 여러 가지 한계가 있다. 이에 외부 프로그램과 웹 서버간의 연결이 필요하게 되는데, 이러한 연결을 하기위한 규약을 CGI라고 한다. 또는 넓은 의미로 CGI를 수행하는 외부 프로그램을 포함하여 CGI라고 한다[4].

본 논문에서는 웹 서버에서 클라이언트로 데이터 전송 시 이를 암호화하기 위해 Enc_CGI 프로그램을 이용한다. 이 때 개발된 Enc_CGI 프로그램은 서버의 적정 디렉토리에 등록되어야 한다. 본 시스템에서는 Windows NT를 사용하고 있으므로, "C:\InetPub\wwwroot\cgi-bin" 디렉토리에 Enc_CGI.exe를 복사함으로써 이를 등록한다. 물론 Enc_CGI에서 보안토큰과의 인터페이스를 위해 개발된 CAPI들이 호출하는 CAPI.dll과 CAPI.lib 또한 적정 디렉토리(C:\WinNT\System)에 복사한다.

등록된 Enc_CGI.exe는 서버에서 클라이언트로 보낼 데이터를 암호화하여 "output.tex"를 생성한다. 이때 Enc_CGI에서는 개발된 CAPI(OpenSession(), GetMasterKeySet(), GetMasterKeyHandle(), EncryptMessage(), CloseSession() 등)를 이용하여 보안토큰 내에서 실제 암호화를 수행시킨 후 그 결과로 "output.tex"를 생성하게 된다.

이를 전달받은 클라이언트에서는 데이터를 복호 할 수 있는 Plug-in을 호출하여야 한다. 이를 위해 서버에서 웹 페이지 구축 시 HTML 언어에 <EMBED> 필드를 추가하여야 하는데 <EMBED>의 간단한 형식을 예제를 통하여 살펴보면 다음과 같다.

<EMBED "src = output.tex" WIDTH = 100 HEIGHT = 100>

위에서 src는 클라이언트의 웹 브라우저에게 보내기 위한 암호화된 문서 파일이다. 이때 클라이언트의 넷스케이프 웹 브라우저에게 Plug-in이 실행되도록 하기 위해 파일의 확장명(tex)을 정확히 명기해야 한다. 물론 클라이언트에는 넷스케이프 웹 브라우저에 tex Plug-in 타입이 등록되어 있어야 한다. 즉 위의 문장은 Tex 파일 형식으로 폭과 높이가 100인 Plug-in 모듈을 넷스케이프 웹 브라우저로 하여금 호출하도록 하는 역할을 한다.

4.2.2 Plug-in

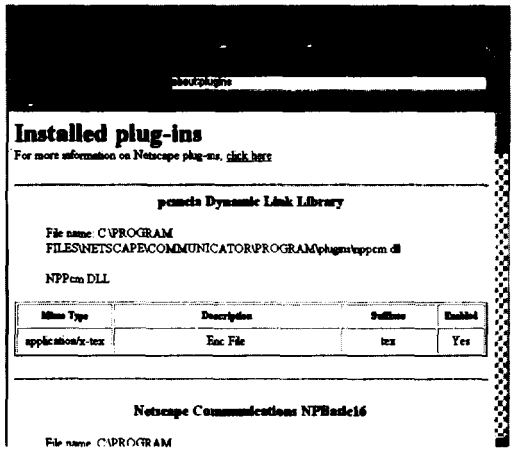
Plug-in 프로그램 기능은 인터넷을 사용하는 사용자들에게 너무나도 편리한 기능으로써, Plug-in 프로그램 기능이 나타나면서 사용자들은 특정 프로그램의 원천 코드(source code)가 없이도 자신이 추가시키고자 하는 기능에 대해서 쉽게 프로그램 가능하게 되었다[13].

본 논문에서는 이러한 Plug-in 프로그램 기능을 이용하여 서버에서 암호화된 데이터를 복호 시켜, 브라우저를 통해 화면에 나타낸다. 이 때 클라이언트에서 Plug-in이 수행되는 과정은 다음과 같다.

넷스케이프로 접속 시 클라이언트에서는 서버의 HTML에서 제공하는 CGI 파일로부터 읽어 들이는 문서 정보의 형식에 따라 그에 맞는 Plug-in을 찾게 된다. 실행된 Plug-in은 넷스케이프 웹 브라우저 속에 포함되거나 분리되어 윈도우에 나타난다. 본 논문에서는 서버에서 암호되어 온 tex 타입의 문서를 클라이언트에서 복호 시키는데 Plug-in 프로그램을 사용한다. 실제로 암호 데이터를 받은 Plug-in 프로그램에서는 개발된 CAPI(OpenSession(), GetMasterKeySet(), GetMasterKeyHandle(), DecryptMessage(), CloseSession() 등)를 이용하여 보안토큰 내에서 복호시킨다.

이러한 기능을 가진 Plug-in 모듈을 실행 할 수 있도록 하기 위해서는 Plug-in 프로그램 모듈을 넷스케이프 웹 브라우저가 설치된 디렉토리 내에 설치해야 하는데, 본 논문에서는 개발된 Plug-in 모듈인 nppcm.dll을 적정 디렉토리(C:\ProgramFiles\Netscape\Communicator\Program\Plugins)에 복사한다. 그리고 Plug-in에서 데이터 복호 역시 CAPI를 통하여 보안토큰 내에서 이루어지므로, CAPI가 호출하는 CAPI.dll과 CAPI.lib 파일을 같은 디렉토리에 복사한다. 그러면 Plug-in 파일인 nppcm.dll이 동작하면서 CAPI.dll 파일과 CAPI.lib를 이용하게 된다.

Plug-in 모듈을 넷스케이프 웹 브라우저에 복사한 후 Plug-in 모듈이 제대로 등록되었는지 확인하는 방법은 다음과 같다. 먼저 넷스케이프 웹 브라우저를 실행시킨 상태에서 help 메뉴에서 About Plug-ins를 실행하여 (그림 8)과 같은 화면이 나타나면 제대로 등록이 된 상태이다.



(그림 8) 넷스케이프 웹 브라우저 상에서 Plug-in의 정보 보기 화면

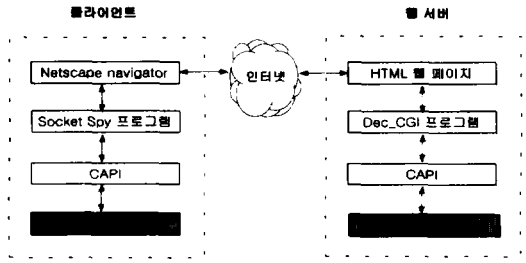
(Fig. 8) Window of the "About Plug-ins" on the Netscape

4.3 클라이언트에서 서버로 데이터 전송

우리가 네트웍을 통해 웹을 검색하다 보면 서버에서 데이터를 받는 경우가 대부분이지만 클라이언트에서 서버로 데이터를 전송하는 경우도 있다. 예를 들어 웹 페이지 상에 사용자 개인의 정보를 입력하거나 게시판에 글을 올릴 경우가 이에 해당된다. 게시판은 누구나 다 보아도 아무런 상관이 없지만 개인의 신상 혹은 금융 정보(신용카드번호) 같은 중요한 정보는 어느 누구에게도 유출되어서는 안된다. 이러한 문제를 해결하기 위해서 보안 모듈을 이용해서 웹 서버로 전송하게 되면 개인의 중요한 정보가 유출되는 것을 막을 수 있다.

이와 같이 클라이언트의 중요한 데이터를 보호해야 하는 경우는 클라이언트의 넷스케이프 웹 브라우저에서 서버로 전송되는 메시지를 암호화하여 서버로 전송해야 한다. 이를 위해 클라이언트는 Socket Spy 프로그램을 사용하여 데이터를 암호화한다. Socket Spy 프로그램은 웹 브라우저가 접속하고 있는 사이트가 보안 사이트인 경우, 브라우저에서 발생되는 메시지를 안전하게 전송하기 위하여 (그림 9)에서와 같이 CAPI를 이용하여 보안토큰에서 암호 한 후 서버로 전송한다.

서버는 클라이언트에서 암호화된 문서가 전송되면 HTML로부터 이 메시지를 받아서 이를 복호 시키는 프로그램인 Dec_CGI를 수행한다. Dec_CGI 에 전달된 암호 메시지는 (그림 9)에서와 같이 CAPI를 이용하여 보안토큰 내에서 복호된다.



(그림 9) 클라이언트에서 서버로의 데이터 전송모델

(Fig. 9) Data Transfer Model from Client to Server

4.3.1 FORM 태그와 Socket Spy

클라이언트에서 서버로 데이터를 넘겨주는 경우 안전한 메시지 전송 과정에 대한 동작을 살펴보자.

먼저 클라이언트에서 사용자가 어떠한 데이터를 서버에게 보낼 수 있는 기능은 <FORM>태그를 통해 이루어지는데, <FORM>태그의 간단한 예제를 살펴보면 다음과 같다.

```
<form method=post  action="http://129.254.27.165/
    cgi-bin/ Dec_CGI.exe">
<p> 이          름: <input name="name"size="12">
<p> 주민등록번호: <input name="personalID" size
    ="14">
<p> 전 화 번 호: <input name="tel size="12">
<p> 삐삐/휴대폰 : <input name="hp"size="13">
<p> 주          소: <input name="addr"size="50">
<p><input type=submit value="확  인"> <input
    type=reset value="재입력">
</form>
```

<FORM> 태그 문의 첫 번째 라인의 "method"는 입력되는 데이터에 대한 속성을 결정하는데, 여기에는 POST와 GET 두 가지 방식이 있다. GET 방식은 클라이언트 사용자의 입력 값이 환경변수에 저장되어 넘겨지는 반면 POST 방식은 stdin(표준입력)을 통해서 전달된다. 따라서 GET 방식은 인수를 통해 전달되므로

명령라인의 길이에 제한을 받지만 POST 방식은 stdin (standard input)을 이용함으로 데이터 양에 제한이 없다는 장점이 있다. 그 다음의 action문은 서버측에서, 클라이언트로부터 입력되어 온 데이터를 처리하기 위한 CGI를 링크 시켜주는 역할을 한다. <input> 태그는 입력받을 데이터의 이름과 길이를 정해주고, <input> 태그 안의 submit은 데이터를 전송하고, reset은 입력한 데이터를 깨끗하게 화면에서 지우는 역할을 한다.

위와 같은 <FORM> 태그 문을 통해, 클라이언트에서는 데이터를 입력 할 수 있는 화면이 나타난다. 이 때 사용자는 자신의 신상 정보를 모두 입력한 후 확인 버튼을 클릭한다. 일반적으로 클라이언트의 웹 브라우저에 입력된 메시지는 확인 버튼을 클릭하기 전까지 웹 브라우저의 임시 버퍼에 남아 있다. 그리고 확인 버튼을 클릭하는 순간 윈도우에서 제공하는 winsock 기법을 통해서(wsock32.dll파일) 웹 서버로 전송된다. 이 때 Socket Spy 프로그램에서는 현재 웹 브라우저가 접속한 사이트가 보안 사이트이면, 웹 브라우저에서 발생되는 데이터를 가로채기하여 암호화한 후, 윈속 기법을 이용하여 서버로 전송하고, 현재 접속 사이트가 일반 웹 사이트인 경우는 데이터를 암호화하지 않고 기존의 윈속 기법을 이용하여 서버로 전송한다. 데이터를 암호화 한 후 전송 시, 실제 데이터의 암호화는 개발된 CAPI(OpenSession(), GetMasterKeySet(), GetMasterKeyHandle(), EncryptMessage(), CloseSession() 등)를 이용하여 보안토큰 내에서 이루어진다.

이와 같은 기능을 가진 Socket Spy 프로그램을 이용하기위해서는 넷스케이프 웹 브라우저에서 wsock32.dll파일을 불러들이는 부분을 수정해서 Socket Spy 프로그램이 생성한 wsock00.dll 파일을 이용하도록 수정해 주어야 한다. 넷스케이프 웹 브라우저의 프로그램을 수정하기 위해서는 Visual C++의 edit에서 Netscape.exe 파일을 이진(binary)형식으로 연다. 이진 형식으로 데이터가 보여지면, 여기서 Wsock32.dll 파일을 wsock00.dll로 변경 한 후 파일을 저장하고 Visual C++ 에디터 프로그램을 종료한다.

수정된 netscape.exe를 넷스케이프가 설치된 디렉토리에 복사하여 등록하고, 개발된 wsock00.dll은 C:\Windows\System\ 디렉토리에 복사하여 설치한다. 그리고, 현재 접속 사이트가 보안 사이트인지 일반 사이트인지를 체크하기 위하여, 넷스케이프가 설치된 디렉토리 내에 보안 사이트의 주소를 포함하고 있는 Security.txt를 복사하여 등록한다.

위와 같은 방법으로 등록된 Socket Spy 프로그램(wsock00.dll)은 앞서 언급한 바와 같이 넷스케이프 웹 브라우저에서 입력한 데이터를 받은 후, 현재 접속 사이트가 security.txt내에 있는 보안 사이트이면 입력 데이터를 보안토큰 내에서 암호화한다. 그 다음 wsock32.dll을 이용하여 웹 서버로 데이터를 전송한다.

4.3.2 Dec_CGI

클라이언트에서 암호화되어 전송된 데이터를 받아들인 서버에서는 이를 복호화하는 과정이 필요하게 되는데, 이는 앞서 언급한 바와 같이 FORM 태그 문장의 action에서 지정한 Dec_CGI 모듈에서 이루어진다. 이 때 실제 데이터의 복호화는 개발된 CAPI(OpenSession(), GetMasterKeySet(), GetMasterKeyHandle(), DecryptMessage(), CloseSession() 등)를 이용하여 보안토큰 내에서 이루어진다.

위와 같은 기능을 가진 Dec_CGI 모듈을 이용하기 위해서는 Dec_CGI.exe를 적정 디렉토리(c:\InetPub\wwwroot\cgi-bin)에 복사하여 등록하여야 한다. 물론 Dec_CGI에서 보안토큰과의 인터페이스를 위해 개발된 CAPI가 호출하는 CAPI.dll과 CAPI.lib 또한 적정 디렉토리(C:\WinNT\System)에 복사한다. 그러면 Dec_CGI에서는 클라이언트에서 온 암호화 된 데이터를 CAPI를 통하여 보안토큰 내에서 복호시키고, 복호된 데이터를 파일로 저장한다.

5. 결 론

PCMCIA 카드 형태의 다양한 기능을 갖는 보안토큰은 차세대 보안의 핵심 기술이다. 본 연구에서는 이러한 보안토큰을 이용하여 웹에서 안전한 자료 전송이 가능한 시스템을 개발하였다. 개발된 시스템은 기존의 소프트웨어 방식을 이용한 웹 보안 해결 방식과는 달리 보안토큰을 가진 사람들에게만 접근을 허용하며, 보안토큰 내에서 데이터의 암/복호 시 보안토큰이 제공하는 자체 암/복호 알고리즘을 사용한다. 따라서 보안토큰을 이용한 암/복호 방식은 보안토큰이 없으면 어떤 경우라도 복호가 불가능한 장점이 있다.

이와 같은 특징을 지닌 시스템은 정부 기관이나 회사 등 조직 내의 중요 정보를 웹 페이지로 구축 할 경우 아주 유용하게 적용될 수 있으며, 나아가서는 웹

페이지를 이용한 정보 교환의 전 분야(인터넷 EDI)에 활용 될 수 있을 것으로 예상된다.

또한 응용 프로그램과 보안토큰 사이의 공통적인 보안 인터페이스로서 개발된 CAPI는 암호학에 지식이 없는 프로그래머들도 이를 사용하여 보안토큰이 제공하는 보안 서비스를 제공받을 수 있도록 함으로써 보안 기능을 가진 응용 프로그램의 개발을 쉽고 간편하게 할 것이다.

참 고 문 헌

[1] Anderson, Don, "PCMCIA System Architecture," Addison-Wesley, 1995.

[2] Bryan Waters, "MASTERING OLE 2 프로그래밍," 삼각형, 1995.

[3] Charles Petzold, "Programming Windows 95," 교학사, 1996.

[4] Ed Tittel, Mark Gaither, Sebastian Hassinger & Mike Erwin, "CGI 바이블," 영진출판사, 1997.

[5] MicroSoft, "Microsoft CryptoAPI Application Programmers Guide Version 1.0," 1996.

[6] Mori, M. T. and Welder, W. D., "The PCMCIA Developers Guide," 2nd ed., Sycard Technology, 1994.

[7] National Semiconductor, "Designing Multiple Function PC Cards," Application Note 975, January, 1995.

[8] National Semiconductor, "PCM16C02 Configurable Multiple PC Card Interface Chip," July 1995.

[9] Peter D. Hipson, "윈도우 NT서버," 사이버출판사, 1997.

[10] RSA Laboratories, "PKCS#11 : Cryptographic Token Interface Standard," Technical notes Ver.1.0, 1995.

[11] Vireo Software, "Driver : Works Users Guide," 1997.

[12] Walter Oney, "Systems Programming for Windows 95," Microsoft Press., 1996.

[13] Zan Oliphant, "넷스케이프 플러그-인 프로그래밍," 인포북, 1997.

[14] Bill McCarty, Steven Gilbert, "Visual C++6 Programming Bule Book," Coriolis Technology Press, 1998.

[15] Viktor Toth, "Visual C++5 Unleashed," IDG Books Wordwide, 1997.

[16] David Kruglinski, "Inside Visual C++," 4th Edition, Microsoft Press, 1997.

[17] John Mueller, Tom Sheldon, "Microsoft Internet Information Server 4 : The Complete Reference," Osborne/McGraw-Hill, 1997.

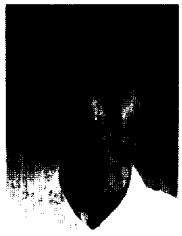
[18] Peter Dyson, "Mastering Internet Information Server 4," SYBEX Inc. 1997.

[19] 강창구, 윤재우, 하경주, 장승주, "인터넷 웹 환경에서 보안 데이터 전송을 위한 분산 시스템 설계 및 개발," '98 한국정보과학회 가을 학술발표 논문집, 제25권 제2호, pp.618-620, 1998.

[20] 김경만, "IIS로 웹 서버를 구축하자," 정보시대, 1997.

[21] 윤재우, 강창구, 하경주, 장승주, "플러그인 프로그램을 이용한 보안 데이터 전송 모듈 설계 및 개발," '98 한국정보과학회 가을 학술발표 논문집, 제25권 제2호, pp.550-552, 1998.

[22] 하경주, 윤재우, 강창구, 장승주, "보안토큰을 이용한 웹 보안 시스템 개발," '98 한국정보처리학회 가을 학술발표 논문집, 제5권 제2호, pp.797-800, 1998.



하 경 주

e-mail : kjha@etri.re.kr

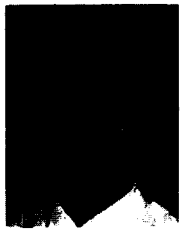
1991년 경북대학교 컴퓨터공학과 졸업(공학사)

1993년 경북대학교 대학원 컴퓨터공학과 졸업(공학석사)

1996년 경북대학교 대학원 컴퓨터공학과 졸업(공학박사)

1996년~현재 한국전자통신연구원 선임연구원

관심분야 : parallel algorithm, network security, web security



윤 재 우

e-mail : jyoon@etri.re.kr

1983년 전북대학교 전자공학과 졸업(공학사)

1985년 전북대학교 대학원 전자공학과 졸업(공학석사)

1997년~현재 전북대학교 대학원 전자공학과 박사과정 재학 중

1989년~현재 한국전자통신연구원 선임연구원

관심분야 : 정보보호 시스템, 전자상거래, 분산처리시스템

강 창 구

e-mail : cgkang@etri.re.kr

1979년 한국항공대학교 전자공학과(학사)

1986년 충남대학교 전자공학과(공학석사)

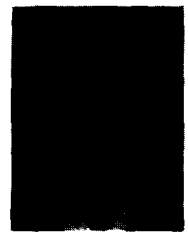
1993년 충남대학교 전자공학과(공학박사)

1979년~1982년 대한민국 공군장교

1987년~현재 한국전자통신연구원 책임연구원 부호3팀장

1997년~현재 한국통신정보보호학회 충청지부 부지부장

관심분야 : 부호 및 통신이론, 정보보호 이론, 디지털 서명, 전산 및 통신 정보보호 기술



장 승 주

e-mail : sjjang@hyomin.dongeui.ac.kr

1985년 부산대학교 계산통계학과(학사)

1991년 부산대학교 계산통계학과(석사)

1996년 부산대학교 컴퓨터공학과(공학박사)

1987년~1996년 한국전자통신연구원

1996년~1997년 한국전자통신연구원(위촉연구원)

1996년~현재 동의대학교 컴퓨터공학과 조교수

관심분야 : 운영체제, 분산 시스템, 컴퓨터 보안