통합형 암호 API의 설계와 구현

김태성[0], 노종혁, 최대선, 원형석, 진승헌
한국전자통신연구원
(taesung[0], jhroh, sunchoi, moho, jinsh)@etri.re.kr

The Design of Integrated Cryptographic API

Taesung Kim, Jonghyuk Roh, Hyung, Suk Won, Seunghun Jin
Electronics and Telecommunications Research Institute

요 약

정보보호 서비스의 폭 넓은 사용을 위해 암호 알고리즘 접근의 표준이 중요한 역할을 한다. PKCS#11과 MSCSP는 암호 알고리즘, 전자 서명, 인증서 저장 등을 포함하는 표준 인터페이스이다. 보안 응용프로그램 개발에 있어 두 표준을 동시에 준용하는 설계와 개발은 쉽지 않은 일이다. 이에 따라 본 논문은 두 표준을 통합적으로 사용 할 수 있도록 하는 통합 암호 API를 제안하였다.

1. 서론

인터넷을 이용한 전자상거래, 인터넷 뱅킹등의 증가에 따라 안전한 사용을 보장하는 정보보호서비스의 요구도 꾸준한 증가추세에 있다. 이러한 정보보호 서비스의 폭 넓은 활용을 위해 암호 표준은 중요한 역할을 한다.

암호 표준으로는 PKCS#11과 MSCSP가 있다. 이들 표준은 암호 알고리즘과 암호연산을 수행하거나 키 또는 인증서를 저장하는 물리적인 장치에 대한 표준 인터페이스를 제공한다. 그러나 두 표준을 동시에 준용하는 보안응용프로그램을 개발하고자 할 때는 상이한 인터페이스 때문에 설계와 구현에 많은 노력을 기울여야 한다. 따라서 본 논문은 두 표준을 통합하여 사용 할 수 있는 통합형 암호 API를 제안한다.

제안된 통합형 암호 API는 두 표준을 간단하고 일관된 API로 통합하였을 뿐만 아니라 두 표준을 동시에 사용할 때 생기는 인증서 호환 사용의 문제를 해결하였다. 본 논문의 구성은 다음과 같다. 2장에서는 암호 표준에 대해 알아보고, 3장에서는 제안된 통합형 암호 API에 대해 기술한다. 그리고 4장에서는 결론을 맺는다.

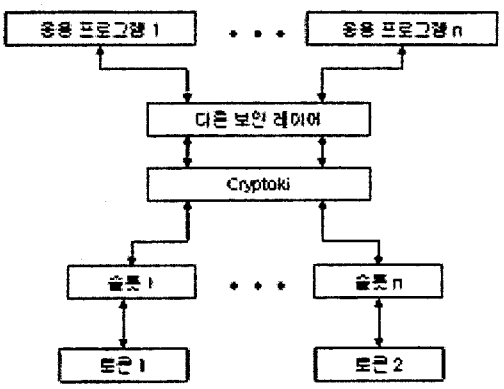
2. Cryptographic Service Provider

CSP(Cryptographic Service Provider)는 키 생성, 암복호, 전자서명, 서명 검증 등의 정보보호 서비스를 제공하는 인터페이스이다. CSP는 일반 목적 응용 프로그램 개발자들이 최소의 암호학 지식으로 암호 기능이나 보안 기능을 가진 응용 프로그램을 개발할 수 있도록 한다. CSP는 성능, 편리성, 키 보관의 보안성 등의 고려에 따라 여러 가지 형태로 구현될 수 있다. 예를 들어 개인키와 인증서를 안전하고 편리하게 보관하기 위해 스마트 카드, PCMCIA, USB 키 등의 형태로 구현되기도 하고 암복호나 전자서명의 성능을 향상 시키기 위해 전용 프로세서나 네트웍 소켓의 형태로 구현되기도 한다. CSP의 대표적인 표준으로는 PKCS#11과 MSCSP가 있다.

PKCS#11[1]은 PKCS(Public Key Cryptography Standards)라고 하는 RSA Security가 제안한 표준 시리즈 중의 하나이며, cryptoki 라고 불리기도 한다. PKCS#11의 목적은 암호토큰에 대한 기술 독립된 접근의 보장과 자원의 공유이다. 기술 독립된 접근이란 물리적인 장치의 특성에 관계없이 공통된 인터페이스로 정보보호 서비스를 제공하는 것이고, 자원의 공유는 여러 응용 프로그램이 여러 개의 디바이스를 공유할 수 있도록 함을 의미한다. [그림 1]은 PKCS#11의 모델이다. 응용프로그램은 cryptoki를 통해 암복호, 전자서명등의 연산과 디바이스 접근을 한다. Cryptoki는 논리적인 관점으로 슬롯과 토큰을 관리하는데, 이는 실

제 디바이스의 리더기와 디바이스 미디어의 관계와 같다. 예를 들어 , 스마트카드의 경우 슬롯은 스마트카드 리더기이고 토큰은 스마트 카드를 지칭하게 된다. Cryptoki는 · 사용자를 일반사용자와 보안감독자(security officer)으로 구분하여 관리한다. 보안감독자는 일반사용자의 디바이스 접근 암호의 설정과 디바이스의 초기화를 수행하는 역할이다. 응용프로그램이 토큰의 함수를 호출하기 위해서는 하나 이상의 세션을 열어야만 한다. 세션은 로그인 여부에 따라 read-only 상태와 read-write 상태로 나뉘고, 세션이 닫히면 소멸되는 객체를 세션 객체, 그렇지 않은 객체를 토큰 객체라 한다.

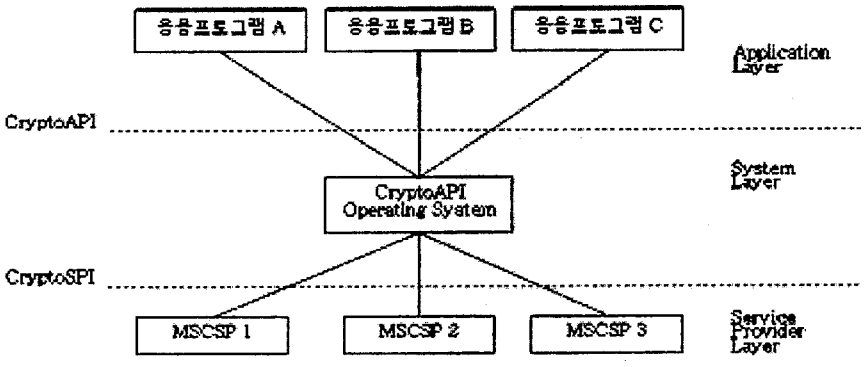
MSCSP[2]는 마이크로소프트가 제안한 암호화 알고리즘의 구현을 위한 표준이다. MSCSP는 CryptoSPI(system programming interface)라는 인터페이스를 가진 동적 라이브러리(dynamic library)로 구현된다. [그림 2]와 같이 응용 프로그램은 직접적으로 MSCSP에 접근하지 않고 CryptoAPI라는 운영체제의 advapi32.dll과 crypt32.dll에 구현되어 있는 함수들을 통해서 접근할 수 있으며, CryptoAPI는 내부적으로 CSP에 구현되어 있는 CryptoSPI를 호출하게 된다. MSCSP가 PKCS#11과 비교하여 다른 점은 일반 사용자에게 배포하기 위해서는 마이크로소프트로부터 서명을 받아야 설치가 가능하며, PKCS#11의 경우에는 인증서가 토큰에 직접 저장되는 반면 MSCSP는 인증서가 레지스트리에 저장된다는 것이다.



[그림 1] PKCS#11의 모델

3. Generic Crypto API

앞서 살펴본 바와 같이 PKCS#11과 MSCSP는 같은 기능을 하는 다른 표준이다. 따라서, 두 CSP 모두를 따르는 보안응용프로그램을 개발해야 하는 개발자는 시간과 노력을 기울여 두 CSP를 이해해야 하고 두 표준이 상충되지 않도록 설계와 개발에 주의해야 한다. 또, 두 CSP의 인증서 저장 위치가 서로 다르므로 하나의 CSP를 통해 발급 받은 인증서는 다른 CSP를 따르는 응용프로그램에서는 유효한 인증서일지라도 사용을 못하는 문제가 있다. 예를 들어, 어떤 토큰이 두 CSP를 모두 지원할 때 MSCSP를 따르는 인터넷익스플로러를 사용해 인증서를 발급 받으면, 비록 PKCS#11를 따르는 네스케이프가 그 토큰에 접근이 가능하더라도 발급된 인증서는 사용할 수 없다.



[그림 2] MSCSP의 모델

본 논문에서 제안하고 설계한 통합 암호 API는 두 CSP를 간단하고 일관된 적은 숫자의 API로 통합하였다. 이로 인해 보안응용 프로그램 개발자는 PKCS#11 또는 MSCSP를 따르는 어떤 토큰도 접근 가능한 응용프로그램을 쉽게 개발 할 수 있다. 또 통합 암호 API를 통해 인증서를 발급 받으면 토큰뿐만 아니라 레지스트리에도 인증서를 저장하여 PKCS#11 또는 MSCSP 지원하는 모든 응용프로그램은 인증서를 사용할 수 있다.

다음은 각 함수의 설명이다.

✓ loadCSP – 이 함수는 PKCS#11과 MSCSP를 구현한 모듈중에서 사용할 CSP를 고르는 기능을 한다. 이 함수를 이용해 다른 CSP를 로드하기 전까지는 이후 요구되는 암복호등의 연산은 현재 선택한 CSP를 사용한다.

✓ registerTokenProvider – MSCSP의 경우는 보통 디바이스 공급자가 제공하는 인스톨프로그램을 통해 운영체제에 설치되지만, PKCS#11은 지원하는 각 응용프로그램에 설치되어야 한다. registerTokenProvider는 PKCS#11 모듈을 통합암호 API에 등록하는 함수이다.

✓ readCertificate – 현재 로드된 CSP가 PKCS#11일 경우에는 토큰에서 인증서를 가져오고, MSCSP일 경우에는 레지스트리에서 인증서를 가져온다. 인증서의 subject를 알고 있으면 해당 인증서를 그렇지 않으면 모든 인증서를 반납한다.

✓ storeCertificate – 현재 로드된 CSP가 PKCS#11일 경우에는 토큰에, MSCSP일 경우에는 레지스트리에 인증서를 저장한다.

✓ generateKey – 대칭키를 생성한다.

✓ generateKeyPair – 공개키 키쌍을 생성한다.

✓ Hash – 해쉬를 수행한다.

✓ Mac – MAC(message authentication code)를 수행한다.

✓ MacVerifiy – Mac 값을 검증한다.

✓ Encrypt – 대칭키 또는 공개키로 암호한다.

✓ Decrypt – 대칭키 또는 공개키로 복호한다.

✓ Sign – 전자서명한다.

✓ Verify – 전자서명을 검증한다.

✓ linkCSP – 구현된 PKCS#11과 MSCSP 모듈이 같은 토큰을 대상으로 하는 경우에 두 CSP를 연결 짓는다. 인증서를 발급 받는 경우 어떤 CSP가 로드 되었는가에 관계없이 토큰과 레지스트리에 인증서를 저장한다.

4. 결론.

본 논문에서는 PKCS#11과 MSCSP를 통합 사용할 수 있는 통합 암호 API를 제안하고 설계하였다. 통합 암호 API는 간단하고 일관된 인터페이스로 동시에 PKCS#11과 MSCSP를 사용할 수 있게 한다. 또한 같은 토큰을 지원하는 PKCS#11과 MSCSP 모듈은 인증서 발급 시 상호적으로 사용할 수 있게 한다.

참고문헌

[1]. PKCS#11 , "http://www.rsasecurity.com/rsalabs /pkcs/pkcs-11/index.html"

[2]. MSCSP, http://msdn.microsoft.com/library/

[3]. Housley, R., W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure: Certificate and CRL Profile", RFC 2459, January 1999.

[4]. Housley, R., W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile draft-ietf-pkix-new-part1-08", July 2001