CryptoAPI 지원 암호 모듈(CSP) 구현에 관한 연구

홍순좌, 박중길, 김영진
국가보안기술연구소
e-mail : hongsj,{jgpark,yjkim67}@etri.re.kr

A Study for Implementation of Cryptographic Service Provider(CSP)

Soonjwa Hong, Joonggil Park, Youngjin Kim
National Security Research Institute(NSRI)

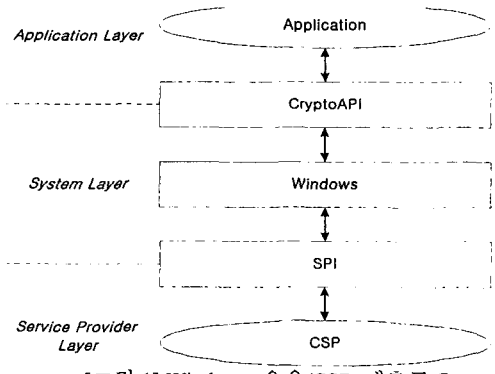
요 약

최근 정보 보안에 대한 연구 및 개발이 활발하게 이루어지고 있으며, 그 중 보안 API 는 보안 서비스를 제공하는 인터페이스 규격으로서 중요성이 증대되고 있다. 대표적인 보안 API 로는 MS 의 CryptoAPI, Intel 보안 구조인 CDSA 의 CSSM API, IETF 의 GSS-API/IDUP-GSS- API, X/Open 그룹의 GCS-API 등이 있다. 보안 API 는 응용 개발자와 보안 장비 개발자의 편리성 및 독립성을 최대한 보장할 수 있어야 하지만, 실제 구현 환경에서 부딪치는 문제는 OS 플랫폼이 기반이 되지 못한 경우 시스템 보안 구조의 계층화가 어렵고, 실제 구현 환경에서 호환성을 보장할 수 없다는 것이다. 이러한 관점에서 MS 의 CryptoAPI 는 응용 및 보안 장비의 개발 규격 및 절차를 제안하고 있으며, 두 분야의 개발자 사이의 연동은 시스템 OS 인 Windows 가 담당하고 있다

1. 개요

Microsoft 사의 OS 는 현재 Win9x, WinNT, Win2000 이 제공되고 있으며 서버 및 개인용 PC 에서 광범위하게 채택되어 사용되고 있다. 대부분의개발자들이 제공하는 응용 S/W 들은 MS Windows 를 기반으로 운영되고 있다. 응용 S/W 는 Windows 가 지원하는 시스템 호출 함수인 Win32 API 를 동적으로 링크하여 사용하게 된다. 즉, MS 의 지원 API 를 인터페이스 규격으로 사용하고 있다, 현재 MS 사는 보안 관련 API 를 따로 구분하여 응용 개발자에게 CryptoAPI V2.0 규격으로 제공하고 있으며, 보안 API 규격 외에도 개발 절차를 규정하고 있다.

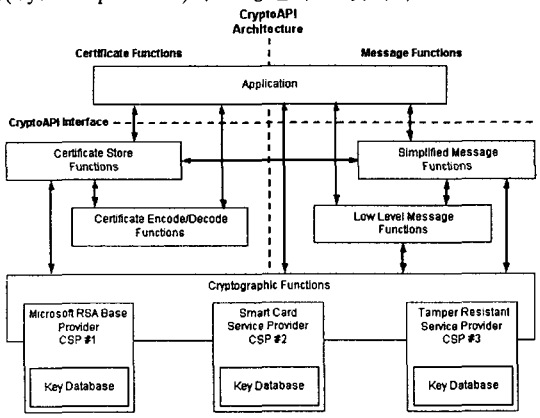
Cryptographic Service Provider(이하 CSP)는 Microsoft Windows 에서 채택하는 보안 토큰 개발 방식으로서 상위의 보안 API 인 CryptoAPI 와 연동되는 하위 레벨의 보안 계층 구조의 구성요소이다. MS Windows 는 CryptoAPI 와 CSP 의 간접 연동을 지원하여 응용 개발자와 보안 토큰 개발자의 개발 편리성 및 독립성을 제공하고 있다. 즉, 응용 개발자는 Windows 의 인터페이스 규격인 CryptoAPI 를 링크 시키는 개발 방식과 각 CryptoAPI 를 적절하게 사용하면 하위의 보안 토큰 형식에 관계없이 응용 S/W 를 개발할 수 있다. 보안 토큰 개발자도 응용의 변화에 관계없이 CSP 엔트리 함수만 규격에 맞추고 MS 에서 제안하는 개발 방식과



[그림 1] Windows 응용/CSP 계층구조

절차에 따른다면 독립적인 보안 토큰의 개발이 가능하다.

[그림 1]은 Windows 의 응용과 CSP 의 계층 구조를 보여 주고 있다. 여기서 가장 큰 특징은 상위 계층인 응용과 하위의 CSP 간에 연동되는 처리 과정을 Windows 가 조정해 줌으로써 상/하위 간의 계층 독립성(layer independence)이 보장된다는 것이다.



[그림 2] CryptoAPI 구조

2. CryptoAPI 와 CSP 함수

2.1. CryptoAPI

CryptoAPI 는 Windows 기반의 응용 S/W 가 사용하는 보안 API 규격 및 처리부로 구성되며, CryptoAPI 는 다음과 같은 범주로 구분된다([그림 2]).

- Base Cryptography Functions
- Certificate and Certificate Store Functions
- Certificate Verification Functions)
- Message Functions
- Auxiliary Functions

CryptoAPI 형식을 보면 각 함수 API 의 처음은 "Crypt" 및 "Cert"로 식별할 수 있다. 즉 해당 API 이름은 "Crypt" 및 "Cert"에 연접된다. 예를 들면 "CryptAcquireContext"를 들 수 있다. "Cert"는 인증서에 관련된 함수에서 볼 수 있다.

CryptoAPI 관련 Header 는 "Wincrypt.h"에서 선언되며, 위치는 대개 "DevStudio60\VC98\include"에 있게 된다. 응용을 실행하는데 요구되는 라이브러리는 "Advapi32.lib", "Crypt32.lib"가 있으며, 해당 함수의 import 정보를 갖는 라이브러리는 " DevStudio60\VC98 \Lib 에 위치한다. 실제 실행 코드인 DLL 은 "Windows \system"에 위치한다.

2.2. CSP 함수

CSP 는 암호 기능(cryptographic functions)을 지원하는 소프트웨어 및 하드웨어로 구현되어 Windows 에 사용될 수 있는 독립적인 보안 모듈로서 CryptoSPI (System Program Interface)와 Crypto Module 로 구성된다.

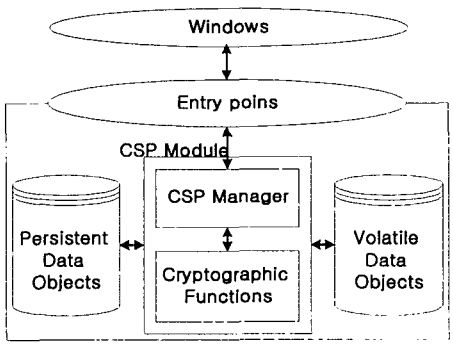
CSP 의 인터페이스는 Windows 와 CryptoSPI 를 통해 상호 대화를 수행하며 응용 S/W 에 직접 연결되는 것은 통제된다. 즉, 응용 S/W 는 CryptoAPI 를 통해서 CSP 에 접근하는 것이 가능하다.

CSP 함수는 다음과 같이 구분한다.

- CSP connection functions(4)
- Key Generation and Exchange Functions(10)
- Data Encryption Functions(2)
- Hashing and Digital Signature Functions(9)

CSP 함수는 "CP"로 시작되는 함수명을 가지며 CryptoAPI 의 Base Cryptography Functions 의 일부를 제외하고는 일대일로 대응된다. 예를 들어 "CPAcquire Context"의 CSP 함수는 CryptoAPI 의 "CryptAcquire Context"와 대응된다.

CSP 함수는 총 25 종류로 MS 에서 제안하는 Entry Point 이다. 즉 상위 계층인 시스템 계층과의 인터페이스 역할을 하게 된다. 이 함수명은 CSP 개발지침에 따라 동일하게 사용하여야 한다. Windows 에서 등록된 CSP DLL 의 export 되는 함수를 동적으로 링크하게 되어 있다. 각 CSP 함수는 CryptoAPI 의 Base Cryptography Functions 에 있는 API 와 일대일로 대응되나 CryptoAPI 의 Base Cryptography Functions 이 전부 CSP 함수로 대응되지는 못한다.



[그림 3] CSP 모델

3. CSP 구조

3.1 CSP 구성요소

CSP 는 Entry Points, Data Objects, CSP Module 같은 구성요소를 갖는다([그림 3]). Entry points 는 MS Windows 에서 규정한 CSP 인터페이스 규격으로 DLL 로 구현되어 MS 의 서명 및 등록을 마쳐야 사용될 수 있다. Entry points 에 해당되는 함수는 CPAcquireContext 를 포함한 25 종이 정의되어 있다.

CSP DLL 내에서 Entry point 함수를 선언할 때, 키워드 WINAPI 를 사용하여야 한다. 예를 들면 CPAcquireContext 의 경우 다음과 같이 선언되어야 한다.

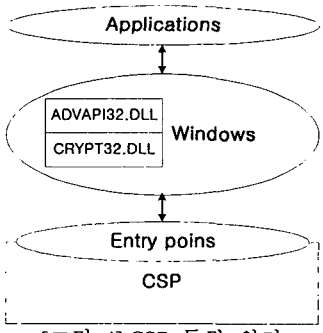
BOOL WINAPI CPAcquireContext(… …)

Data Objects 는 Persistent Data Objects 와 Volatile Data Objects 로 구분된다. Persistent Data Objects 는 지속적으로 유지되는 공개키/비밀키 쌍을 저장 관리하며 Volatile Data Objects 는 임시적인 데이터인 세션 키 및

해쉬 데이터를 저장 관리한다.

CSP 의 구현 방식은 소프트웨어 및 하드웨어로 구현될 수 있는데, 하드웨어로 구현되더라도 Entry point 에 대한 처리를 S/W 적인 DLL 로 구현하여야 한다. S/W CSP 인 경우 Data object 저장소로서 Registry 나 File 을 이용할 수 있고, H/W CSP 인 경우 H/W 토큰에 저장한다.

CSP entry-point 에 응답하는 모듈은 CSP 를 관리하는 CSP Manager 와 암복호, 디지털 서명, 키생성 등의 암호 기능을 제공하는 Cryptographic functions 으로 구성된다.



[그림 4] CSP 동작 원리

3.2 CryptoAPI/CSP 동작 원리

CryptoAPI 와 CSP 가 연동되는 과정은 [그림 4]에서 볼 수 있다. 응용에서 요구하는 CryptoAPI 함수는 Windows 에서 규정된 API 로서, Windows 시스템이 "ADVAPI32.DLL"과 "CRYPT32.DLL"로 동적 링크를 실행하게 해준다. 응용은 일반적인 DLL 링크 과정과는 다르다. Windows 의 링크는 Run-time 링크로서 응용의 LoadLibrary 혹은 LoadLibraryEx 와 같은 Win32 함수의 사용을 요구하지 않고, 응용 및 CSP 개발자가 개발규격에 맞게 선언된 함수를 자동으로 연동한다. 이러한 이유로서 응용 및 토큰 개발자는 독립적인 개발이 가능하다.

Windows 에서 허용된 CryptoAPI 중에서 CSP 의 접근을 요청하는 CryptoAPI 함수는 Windows 에 등록된 CSP 를 찾게 되며, 해당 CSP Entry point 의 함수를 호출하게 된다. 이때 Windows 는 CSP 가 할당한 핸들 값과 다른 핸들 값을 포괄하여 관리하게 된다. 예를 들면, 각 CSP 별로 동일하게 "1"이라는 핸들 값을 할당하더라도 사용자에게 보여지는 값은 유일한 식별 값으로 Windows 에서 재할당된 핸들 값이 리턴된다. 즉, 핸들 값의 중복성을 Windows 의 핸들 할당 정책에 의해 해결된다.

3.3 CryptoAPI/CSP 함수의 비교

CryptoAPI 의 CSP 관련 함수와 CSP Entry point 함수를 일대일로 대응시켜 보면 CSP 함수를 CryptoAPI 에 대응 시킬 수 있다. 대부분은 매개변수가 일치하나 키 컨테이너 핸들에 대한 처리가 차이가 난다.

4. CSP 개발 절차

CSP 를 개발하기 위한 절차는 다음과 같다.

1) CSP 개발 툴킷(CSPDK) 획득
2) CSP 구현
3) CSP 서명 및 등록

4.1 CSP 개발 툴킷

CSP 개발 툴킷인 CSPDK 를 획득하기 위해서는 MicroSoft 홈페이지의 절차에 따라 요청을 하여야 한다. 관련 홈페이지 URL 은 다음과 같다.

"http://www.microsoft.com/security/tech/cryptoapi/cspdkintrocontent.asp "

CSPDK 는 실행 파일(SIGN.EXE, CSP.DLL, TESTCSP.EXE, CSPINSTL.EXE), 소스파일(CSP.C, CSP.DEF, TEST-CSP.C, CSPINSTL.C), 개발용 Windows Dll(ADVAPI32.Dll)로 구성된다. SIGN.EXE 는 CSP DLL 인 CSP.DLL 의 서명 값을 생성하는데 사용되고 CSPINSTL.EXE 는 CSP.DLL 을 레지스트리에 등록하는 프로그램이고, TESTCSP.EXE 는 등록된 CSP 를 시험하는데 사용되는 응용 프로그램이다. ADVAPI32.Dll 는 개발용 CSP 의 entry-point 함수를 처리하는 DLL 이다.

4.2 CSP 개발환경 구성

Windows\system 에 있는 ADVAPI32.Dll 을 CSPDK 에 있는 ADVAPI32.Dll 로 대체한다. 원래의 ADVAPI.DLL 은 많은 시스템 호출 함수의 처리부로서 그 중에서 CryptoAPI 에 관련된 함수를 Windows98, Windows2000, CSPDK 의 ADVAPI 를 비교했을 때, Windows98 과 Windows2000 의 CryptoAPI 는 일치하고 있으나, CSPDK 의 ADVAPI32.DLL 의 함수를 살펴보게 되면 다음과 같은 함수가 누락되어 있다.

- CryptDuplicateHash
- CryptDuplicateKey
- CryptEnumProviderTypesA
- CryptEnumProviderTypesW
- CryptEnumProvidersA
- CryptEnumProvidersW
- CryptGetDefaultProviderA
- CryptGetDefaultProviderW
- CryptSetProviderExA
- CryptSetProviderExW

CSPDK 는 CSP 를 개발하는데 요구되는 최소한의 함수만을 export 하고 있으므로 개발된 CSP 를 가지고 모의 시험을 하게 되는 것이 어렵다.

4.3 CSP 구현

먼저 CSP Entry Points 를 처리할 수 있는 DLL 을 구현해야 한다. CSP 함수의 호출에 따른 입출력 값을 정의하고, CSP 함수 호출에 관련된 Process 관리 기능 구현, 내부 암호 기능의 구현을 하여 CSP 가 완료되면, 시스템에서 시험을 해야 한다. 이때, CSP 의 동작을 확인하려면 Windows 시스템에 등록해야 한다.

시스템 등록 절차를 보게 되면 다음과 같다.

- CSPDK 의 Sign.exe 를 사용하여 Csp.DLL 을 서명

하여 서명 값을 생성해야 한다.

```
> Sign -s Csp.Dll Csp.sig
```

- CSP 의 서명 값을 갖고 있는 Csp.sig 를 통한 서명 파일을 Windows 에 제시하여야 등록이 가능하다.
- CSP 명칭을 "CSP Provider"로 한다면 Windows Registry 에 다음의 키를 생성한다.
  "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\Defaults\Provider\CSP Provider"
- 서브 키 CSP Provider 가 생성되면 필드 값을 다음과 같이 생성한다.

Image Path	"Csp.Dll"
Signature	00 01 4a bf 3a c8 24 … … …
Type	0x00000385(901)

Image Path 필드는 Csp 의 이름으로서 해당 DLL 은 Window\System 에 존재하여야 한다. Signature 값은 Csp.Sig 의 문자열의 16 진수 코드 값으로 크기는 64 바이트이다. Type 은 CSP 식별 번호로서 임의의 CSP 은 10 진수 900 에서 999 사이의 값을 사용할 것을 권고한다.

- 이때, CspDk 의 ADVAPI32.Dll 이 Windows 에 있지 않을 경우 CSP.DLL 을 Windows Registry 에 등록을 할 수 없다.
- 이러한 절차는 임의의 개발자가 자체 CSP 를 통한 전체 CryptoAPI 의 구현을 원천적으로 봉쇄하는 데 있다.
- 이와 같이 Registry 에 등록이 완료되면 CSP Entry 함수를 시험해 볼 수 있다.
- CSP 시험 과정 중 CSP 를 변경하게 되면 서명 값 생성부터 다시 반복 시작해야 한다.

CSP 시험 과정 중 CSP 를 변경하게 되면 서명 값 생성부터 다시 반복 시작해야 한다. 이와 같은 문제점은 개발된 CSP 의 Upgrade 및 변경을 할 경우 처음부터 등록 과정을 반복해야 한다는 것이다.

4.4. CSP 서명 및 등록

개발된 CSP 를 MicroSoft 의 등록 절차에 따라 Export Compliance Certificate (ECC)를 작성하여 CSP 관련 서류를 작성한 후 심사 통과를 기다려야 한다. 이와 같은 절차에 따라 발생할 수 있는 문제점은 다음과 같다.

- CSP 의 변경시 마다 매번 이 절차를 따라야 한다.
- CSP Dll 에 다른 Dll 을 연동하려면 그 Dll 도 등록 및 서명을 받아야 한다.
- CPAcquireContext 요구시 SubDll 의 CallBack 이 필요하다.

5. 결 론

CSP 는 Windows 기반의 시스템에서 가장 효율적인 개발 방식으로 제안되어 있으나 다음과 같은 문제점이 있다.

- 국내에는 이 분야의 개발 경험이 없다.
- CSP 개발에 요구되는 시스템 개발 능력과 암호학적 지식을 동시에 요구한다.
- 개발 과정 중 발생하는 문제점 해결을 위한 조언 및 자문을 구하기 어렵다
- MS 에 의존적인 개발이 될 수 있다.

Windows2000 에는 Gemplus 의 스마트카드 관련 CSP 인 GemSafe 가 Windows 의 기본 Service Provider 로 등록되어 있다.

이와 같은 특성으로 볼 때 다른 보안 솔루션의 개발 방식과는 많이 다르다는 것을 제대로 인식해야 할 것이다. 특히 CSP 개발에 대한 막연한 지식을 가진 프로젝트 관리자는 매우 위험하다. 경험이 없는 설계자의 논리적인 설계가 상세 설계 및 실제 구현에서 발생하는 문제점을 제대로 해결하지 못할 가능성이 크다.

또한 개발에 사용할 참고 자료 부족하다는 것은 커다란 문제점이다. MSDN Library 외에는 가용한 자료가 없으므로 이 자료를 100% 신뢰할 수 밖에 없다. 그러므로 구현시 발생하는 사소한 문제도 반복적인 실험을 통하여 해결할 수 밖에 없다. 암호 제품에 관련된 미국의 정책에 의존적이며, 이러한 부분은 순수한 개발 외에도 정책적인 해결이 관건이 될 수 있다.

CSP 개발을 하기 위해서 준비 및 연구해야 할 분야로 다음과 같이 정리할 수 있다.

- MS 정책에 대한 CSP 개발 계획 및 등록절차 수립
- CryptoAPI 에 대한 연구
- CSP Entry Function 에 대한 연구 및 구현
- CSP Manager 의 설계 및 구현
- CSP Cryptographic Function 의 설계 및 구현
- CSP Integration in Computer System
- CSP Upgrade 계획

마지막으로 MS 의 CSP 로 등록 및 인증된 보안 제품은 Windows 기반의 모든 응용과의 호환성을 보장받을 수 있고, 기술적인 노하우를 바탕으로 상당한 기술 경쟁력을 가질 수 있을 것으로 기대된다.

참고문헌

[1] MicroSoft MSDN Library-April2000, "CryptoAPI V2.0", 2000.4.

[2] MicroSoft MSDN Library-April2000, "Cryptographic Service providers", 2000.4.

[3] RSA Lab. "PKCS#11: Cryptographic Token Interface Standards", April, 1994

[4] 김점구외, "효율적인 정보보호를 위한 상용암호 서비스 기술(CryptoAPI 중심)", 통신정보보호학회지 제 8 권 제 4 호, pp113-130, 1998.12

[5] 김락현 외, "보안 서비스 API 를 위한 보안 응용 라이브러리 CSP 설계 모델", 충청지부정보보호학술발표논문집, 제 2 권 1 호, pp187-203, 1998.11