정보보호 컴포넌트 인터페이스를 통한 SSL의 구현

신용녀° 이동근 신종회
한국정보보호진흥원
{ynshin°, dklee, jshin}@kisa.or.kr

Implementation of SSL Using Information Security Component Interface

Yongnyuo Shin° Dongkeun Lee Jongwhoi Shin
Korea Information Security Agency

요 약

정보보호 기능을 필요로 하는 각종 응용분야에서 다양한 보안 API가 사용되고 있다. 그러나 이러한 표준은 서로 호환성을 가지고 사용될 수 있는 것이 아니라 애플리케이션의 적용환경이나 프로그래밍 언어에 따라서 개발자가 선택적으로 이용해야 하는 단점을 가지고 있다. 이를 해결하기 위하여 정보보호 서비스 컴포넌트 표준화를 진행하고 있으며 표준안의 유효성을 검증하기 위하여 비밀성, 무결성 컴포넌트 인터페이스를 통한 SSL(Secure Sockets Layer)을 구현하였다. 구현된 SSL은 TCP/IP 위에 하나의 단계를 추가하여 보안 알고리즘을 구현하는 것처럼 실제 컴포넌트간에 RMI(Remote Method Invocation)로 통신을 할 때, 각각 하위 수준의 SSL 컴포넌트를 이용하는 구조이다.

1. 서 론

정보보호 기능을 필요로 하는 각종 응용분야에서 다양한 API(Application Program Interface)가 사용되고 있다. 각 표준은 특정 벤더의 응용분야에 적합하도록 개발되었고, 이렇게 개별적으로 추진되어 온 각각의 보안 API를 총괄적으로 표준화한다는 것은 불가능한 일이었다. 이러한 문제를 해결하기 위하여 기존의 정보보호 API보다 발전된 개념의 컴포넌트 설계기법을 도입하여 비밀성, 무결성 서비스 컴포넌트 인터페이스 표준안을 개발하였다. 표준안 개발의 유효성과 안전성을 검증하기 위하여 SSL(Secure Sockets Layer)를 구현하였다.

따라서 본 논문에서는 비밀성, 무결성 서비스 컴포넌트 인터페이스 표준안을 제시하고, 이 표준안의 유효성 검증을 위해 구현된 SSL v3.0을 J2EE 환경에서 EJB 컴포넌트의 구현방안을 보인다. 본 논문은 2장에서 정보보호 서비스 컴포넌트 인터페이스 표준을 기술하고, 3장에서는 SSL 컴포넌트의 설계와 구현을 설명하고, 4장에서 결론을 맺는다.

2. 정보보호 서비스 컴포넌트 인터페이스 표준

2.1 정보보호 컴포넌트

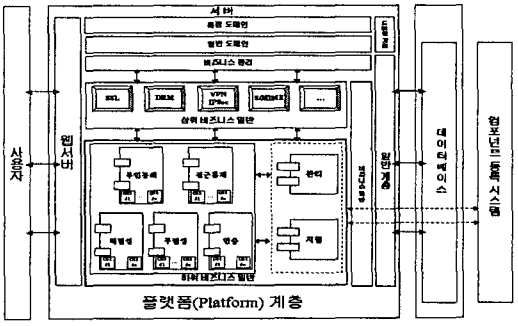
표준화가 개별적으로 추진된 IETF의 GCS-API와 GSS-API, RSA사의 Cryptoki, Microsoft의 CryptoAPI, Intel의 CDSA와 같은 보안 API는 호환성 부족으로 인해 분산환경에서 어플리케이션에 도입하기에 어려움이 있고, 여러 가지 요구사항에 대하여 적시에 이를 구현한 소프트웨어를 제공하기에도 부적합하였다. 컴포넌트는 요구사항에 대응하여 소프트웨어를 미리 제작함으로써 개발의 적시성 및 생산성을 향상시킬 수 있으며, 다양한 제품간 규격화 및 표준화를 유도할 수 있다. 정보보호 기능에 대한 표준화를 유도하기 위해서 정보보호 컴포넌트를 제안하게 되었으며, 이를 위해 정보보호 컴포넌트의 인터페이스를 표준화시키려는 노력을 기울이고 있다. 정보통신분야의 각 응용분야에서 정보보호 서비스를 지원하기 위하여 하나 이상의 기능을 갖는 독립적인 소프트웨어가 정보보호 컴포넌트이다. 정보보호 컴포넌트는 기존의 보안 API가 제공하던 핵심 정보보호 기능[1]인 비밀성, 무결성, 인증, 접근통제, 부인봉쇄 등의 서비스를 제공하는 컴포넌트이다.

- 비밀성 서비스 컴포넌트
  온라인 및 오프라인 환경에서 저장 및 전송되는 중요 정보가 확인되지 않고 인가되지 않은 상대방에게 노출되지 않도록 하는 특성으로서, 관용암호알고리즘 및 공개키 암호 알고리즘에 의한 암·복호화 기능을 제공하는 컴포넌트이다.
- 무결성 서비스 컴포넌트
  네트워크를 통하여 송수신되는 정보의 내용이 불법적으로 생성 또는 변경되거나 삭제되지 않도록 보호되어야 하는 성질로서, 해쉬 알고리즘, 메시지 인증코드 생성 등의 기능을 제공하는 컴포넌트이다.
- 인증 서비스 컴포넌트
  정보교환에 의해 실체의 식별을 확실하게 하는 방법으로서 임의의 정보에 접근할 수 있는 객체의 자격이나 객체의 내용을 검증하는데 사용되며, 전자서명 알고리즘과 무결성 서비스 기능을 제공하는 컴포넌트이다.
- 접근통제 서비스 컴포넌트
  보안 정책에 따라 접근주체(사용자, 프로세스 등)의 접근에 대해서 접근 권한을 확인하여 접근을 제어함으로써 자원의 비 인가된 사용을 방지하며, 인증 서비스 기능을 제공하는 컴포넌트이다.
- 부인봉쇄 서비스 컴포넌트
  송수신자간의 전송사실 및 내용에 대하여 부인을 방지하기 위해 사용되며, 인증 서비스 컴포넌트의 기능을 제공하는 컴포넌트이다.

또한 컴포넌트를 통해 정보보호 기능을 사용하고자 할 때 정보보호 컴포넌트 자체에 안전성을 확보하여 신뢰성을 기반으로 컴포넌트가 사용되어 지도록 해야 한다. 컴포넌트가 본질적으로 가지는 확장성이나 범용성의 다른 단면에는 컴포넌트 자체가 보안적 결함을 가질 수도 있고 컴포넌트 조립(Composition) 단계에서 보안적 결함을 가질 수 있는 가능성이 있다. 컴포넌트의 안전성을 확보하고 정보보호 컴포넌트로서 사용되기 위해서는 이러한 가능성을 배제 시켜야 한다. 이를 위해서 많은 연구들이 진행되고 있는 것이 현실이나, 본 고에서 제안하는 표준안에서는 각각의 인터페이스 입출력 단계에서 사전조건(Pre-Condition)과 사후조건(Post-Condition)을 명확하게 제시함으로써 인터페이스를 사용하여 컴포넌트를 제작할 때 이러한 문제를 최소화시키도록 노력했다.

2.2 정보보호 컴포넌트 프레임워크

기존의 보안 API가 난립해 있는 상태에서 일반 사용자는 정보보호 컴포넌트가 어떠한 영역 계층에 위치하고 개발된 컴포넌트를 어느 계층에서 연동하여 사용할 수 있는지에 대해 혼란에 처해 있는 것이 현실이다. 이를 위해서 분산 환경 기본 구조인 3-Tier계층과 컴포넌트 영역별 계층 구조를 이용하여 설계한 정보보호 컴포넌트 프레임워크는 (그림 1)과 같다. 이러한 계층 구조에서 모든 컴포넌트 영역에 정보보호 서비스를 제공하기 위하여 정보보호 컴포넌트를 비즈니스 일반 계층에 포함되도록 설정하였다. 상위 계층에 속하는 각종 컴포넌트 기반 정보보호 제품들은 정보보호 컴포넌트 프레임워크가 제공하는 기본 정보보호 컴포넌트인 비밀성, 무결성, 인증, 부인봉쇄, 접근통제 컴포넌트를 이용하여 구현되어 진다. 정보보호 컴포넌트 프레임워크에서 비즈니스 일반 계층에 위치한 정보보호 컴포넌트는 다른 계층의 컴포넌트들을 위하여 정보보호 서비스 기능을 제공한다.



(그림 1) 정보보호 컴포넌트 프레임워크

2.3 비밀성 서비스 컴포넌트 인터페이스 명세

비밀성 서비스 컴포넌트 인터페이스는 비밀성 기능을 제공하도록 메시지를 암호화하거나 복호화 해주는 서비스를 제공한다. 상이한 플랫폼에서 범용성을 확보하기 위해서 IDL(Interface Definition Language)로 인터페이스를 디자인하였다. 인터페이스를 정의하는 방법은 여러 가지이다. Visibroker의 경우 Caffein이라는 것을 이용하면 IDL을 사용하지 않아도 되며, Java의 RMI나 RMI나 RMI-IIOP를 이용하면 IDL을 몰라도 인터페이스를 정의할 수 있다. 그러나 정보보호 컴포넌트가 가져야할 요구사항 중에 하나인 가용성을 확보하기 위해서 OMG(Object Management Group)에서 규정하고 있는 인터페이스 정의 언어의 표준[2]이며 사용자가 익히기에 용이한 IDL을 사용하여 (그림 2)과 같이 비밀성 서비스 컴포넌트 인터페이스인 IConfidentiality를 명세 하였다.

```
interface IConfidentiality {
        exception UnSupport {string why};
        exception IllegalState {string why};
        exception InvalidState {string why};
        exception InternalState {string why};

        void setAlgorithm(in string cipherAlgorithm);
        out string[] getAvailableAlgorithms();
        out string getAlgorithm();
        void setNameType(in string IDType);
        out string getNameType();
        void initialize(in string opMode, in byte[] key, in byte[] iv);
        out byte[] update(in byte[] message);
        out byte[] finalize(in byte[] message);
        out int getLength(in int inputLength);
        void setEncodingType(in string codingType);
        out string getEncodingType();
        }
```

(그림 2) 비밀성 서비스 컴포넌트 인터페이스(IConfidentiality)

SetAlgorithm 오퍼레이션은 비밀성 기능으로 사용되는 암·복호화 알고리즘의 명칭 또는 알고리즘 식별자를 설정한다. 사전조건으로는 cipherAlgorithm의 값은 OID, 일반명 또는 NULL이 되어야하고 operation 호출 이전에 컴포넌트는 초기화되어 있어야하는 것을 가질 수 있다. 사후조건으로는 컴포넌트 초기화가 안된 경우 IllegalState, 지원 알고리즘이 없는 경우 UnSupport 그리고 컴포넌트 내부 문제가 발생할 경우 InternalState 예외처리가 발생해야 한다.

3. SSL 컴포넌트의 설계와 구현

3.1 기존 컴포넌트에서 제공하는 SSL

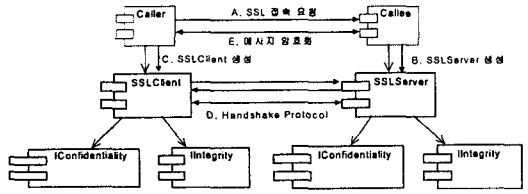
컴포넌트 사용이 활성화되면서 컴포넌트 사용에 있어서도 보안(Security)의 요구가 점차 있었고, 그 요구를 수용하기 위하여 각 플랫폼 개발 회사들은 각각의 보안 해결책을 서버 차원에서 지원하게 되었다. 인증과 권한설정, SSL(Secure Sockets Layer) 사용 등이 컴포넌트 플랫폼에서 보안 이슈들이다. 컴포넌트 플랫폼에서 보안설정을 하기 위해서는 다음과 같은 방법이 있다[3].

- 선언적(Declarative) 방법 : 배치 툴(Deploytool)에서 보안 요구사항을 설정하면, 코드에 영향을 미치지 않고 컨테이너가 트랜잭션 처리와 마찬가지로 보안 요구 사항에 대해 적절한 처리를 하게 된다.
- 프로그래밍적(Programmatic) 방법 : 프로그래밍적으로 보안 요구사항을 설정하는 것은 코드 차원에서 보안

요구사항을 반영하는 것을 의미하는데, 코드 수정이 있을 때마다 컴파일과 실행을 다시 수행해야 한다. 그러나 보다 세밀한(fine-grained) 보안 설정이 가능하다.

두 가지 방법론을 선택하는 기준은 어떻게 편리함과 컨트롤의 균형을 유지하느냐의 선택의 문제이다. SSL은 TCP/IP 상에서 접속간 인증과 암호화를 통한 비밀성 및 무결성을 제공해주기 위하여 Netscape사에서 표준으로 제정한 보안 통신 프로토콜이다. 일반적으로 브라우저와 웹 서버 등에 포함되어 HTTPS 형식으로 다양한 웹 통신 보안에 이용되고 있다. 새로운 SSL 컴포넌트 구현은, 보다 세밀한 보안 설정을 가능하게 해 줄 수 있다. 즉, 비즈니스 컴포넌트에서 발견과 연동이 쉽고, 플랫폼의 보안 정책에 따라 국산 암호화 알고리즘을 포함한 다양한 암호화 메커니즘의 설정이 가능하다.

3.2 정보보호 서비스 컴포넌트 기반의 SSL

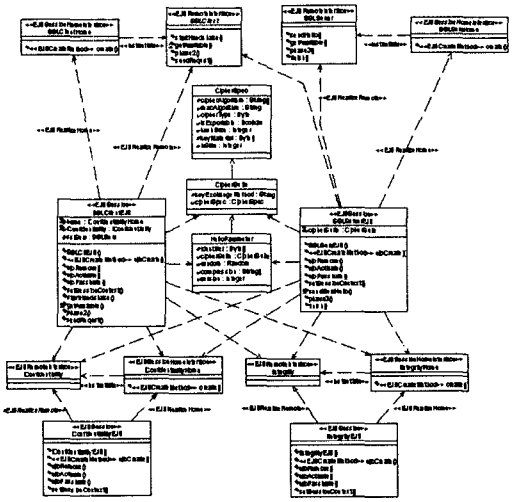


(그림 3) RMI 기반의 SSL 메시지 전송

SSL에서 TCP/IP 위에 하나의 단계를 추가하여 보안 알고리즘을 구현하는 것처럼 실제 컴포넌트간에 RMI로 통신을 할 때, 각각 하위 수준의 SSL 컴포넌트를 이용하는 구조이다. 호출자인 Caller 컴포넌트가 피호출자인 Callee 컴포넌트에 보안 접속을 요청하고 SSL을 사용하고자 할 경우, (그림 3)과 같이 Callee에서는 SSLServer 컴포넌트의 생성을 요청하고 Caller에서는 SSLClient 컴포넌트를 생성하여 SSL handshake 프로토콜을 거쳐서 클라이언트와 서버에서 사용하게 될 암호 알고리즘과 세션 키를 교환한다. Handshake 과정을 거치고 난 후 RMI 매개 변수와 결과 값과 같은 실제 메시지는 서로 교환된 암호 알고리즘들을 이용하여 암·복호화하고 MAC을 생성/검증하여 교환되는 메시지의 보안을 제공하게 된다. 실행방법은 다음과 같다.

가. 최초 Caller에서 Callee로 SSL 접속(보안 접속)요청
나. Callee에서 SSLServer 컴포넌트 객체 생성
다. Caller에서 SSLClient 컴포넌트 객체 생성
라. SSLClient와 SSLServer간에 Handshake Protocol 실행
마. Handshake의 결과를 이용하여 RMI 전송 메시지 암호화

SSL handshake 프로토콜이 완료된 후 Callee 컴포넌트와 Caller 컴포넌트의 메소드를 호출할 때 이미 결정된 SSLClient 컴포넌트의 알고리즘에 따라서 IConfidentiality와 Integrity 컴포넌트를 사용하여 메시지를 암호화하거나 MAC을 구하여 매개변수로 전송하게 된다. 마찬가지로 Caller에서는 정해진 알고리즘을 이용하여 MAC을 검증하여 Integrity를 보장하게 되며 정해진 암호화 알고리즘을 사용하여 전송 받은 메시지를 복호화한다. Callee에서 처리한 결과를 Caller로 보낼 때도 동일한 방법을 사용하게 된다. 이렇게 구현하는 것은 하부 TCP/IP 소켓을 직접 이용하지 않으므로 특정 컴포넌트 플랫폼 환경의 제약을 받지 않는다. SSL 컴포넌트 내부에서 작동하는 클래스간의 관계는 (그림 4)와 같다.



(그림 4) SSL 컴포넌트의 클래스 다이어그램

5. 결론

본 논문에서는 비밀성, 무결성 서비스 컴포넌트 인터페이스 표준안을 제시하고, 이 표준안의 유효성 검증을 위해 SSL v3.0을 J2EE 환경에서 EJB 컴포넌트의 구현방안을 보였다. 기존의 보안 API보다 발전된 개념의 컴포넌트 설계기법을 도입하여 보안 서비스가 필요한 소프트웨어의 품질을 보증하고, 시험 검증을 통하여 표준부품으로 등록된 컴포넌트의 재사용성을 극대화함으로써 시스템의 개발 생산성과 호환성을 획기적으로 향상시키기 위한 보안 서비스 컴포넌트에 대한 표준 개발이 요구된다. 현재 국내외 적으로 정보보호 컴포넌트에 대한 표준화가 전혀 이루어져 있지 않고, 국내의 경우, 컴포넌트 관련 용어 표준, 지침 등이 개발되는 등 초보적인 단계이다. 따라서 표준화 연구를 통하여, 정보보호 컴포넌트에 대한 초석을 마련하여 정보보호 산업의 국제 경쟁력을 강화시키는 효과를 기대할 수 있다.

6. 참고문헌

[1] ISO 7498-2,"Information processing systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture", 1989
[2] OMG Laboratories, "OMG IDL Syntax and Semantics", OMG, July 2002.
[3] 선 홈페이지, http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/Security.html
[4] 최용락, 박수진, "암호 컴포넌트 프레임워크 인터페이스 표준화 연구", KISA, 2002. 12.
[5] 한국전자통신연구원, "CORBA를 위한 개방형 GIS 인터페이스 표준", TTA, 2002. 2.