CORBA 환경에서 보안 서버와 그 인터페이스 설계

한승완, 이영록, 노봉남
전남대학교 전산학과

Security Server and Its Interface Design in CORBA

Seungwan Han, Youngrok Lee, Bongnam Noh
Dept. of Computer Science, Chonnam National University

요 약

정보 처리 기술의 발달과 컴퓨터 네트워크의 발달은 분산 컴퓨팅이라는 새로운 컴퓨팅 환경을 낳았고 분산 컴퓨팅과 객체지향을 통합하여 분산 객체 컴퓨팅 환경이 생겨났다. 분산 객체 컴퓨팅의 목표는 분산된 응용 프로그램을 이루는 각각의 구성 요소들을 객체로 간주하여 분산 객체들 사이에 상호운용성을 제공하는 것이다. OMG에서 제안한 분산 객체 컴퓨팅 시스템의 표준 구조는 CORBA이다. 그러나, 현재까지 CORBA의 보안에 관한 연구는 거의 되고 있지 않은 상태이므로 앞으로 분산 객체 컴퓨팅 환경에서 CORBA의 채택이 급증할 것을 감안할 때 이것은 중대한 문제로 등장한다. 따라서, 본 논문에서는 OMG에서 제안한 CORBA에 보안 서비스를 제공하기 위하여 보안 서버를 제안하고 보안 서버의 인터페이스를 설계하였다.

1. 서론

정보처리 기술의 발달과 사용자의 요구가 다양해지면서 분산 컴퓨팅에 대한 요구가 급증하고 있다. 또한 컴퓨터 네트워크의 발달은 컴퓨팅 환경을 더욱 분산시키고 있다. 분산 컴퓨팅 환경의 응용 프로그램은 한 시스템에서 동작하던 기존의 메인 프레임 중심의 응용 프로그램과는 달리 하나 이상의 시스템에 분산되어 동작한다. 이러한 분산 컴퓨팅 환경에서는 값 비싼 하나의 시스템 대신에 더 저렴하고 작은 여러 시스템을 사용할 수 있으므로 시스템의 성능대 가격비를 향상시킬 뿐아니라 네트워크로 연결된 시스템들의 다양한 처리 능력을 이용할 수 있는 장점을 제공한다. 또한, 메인 프레임을 사용했던 기존의 컴퓨팅 환경에 비해서 시스템의 확장이 쉽다.

최근 가장 널리 소개되고 있는 또 다른 기술은 객체지향이다. 객체지향은 응용 프로그램을 프로시져로 나누어 개발하던 전통적인 프로시져 기반과는 달리 자료와 자료를 다루는 메소드를 결합시킨 객체 단위로 개발하는 방법이다. 객체지향 방법론은 객체의 재사용성과 정보의 은닉화를 통하여 응용 프로그램 개발의 생산성을 크게 향상시켰다. 또한, 모든 대상을 객체로 고려하므로 실세계의 반영이 쉽다.

'90년대에 들어 가장 널리 사용되고 있는 두 기술인 분산 컴퓨팅과 객체지향을 통합하려는 많은 시도들이 있어 왔다. 이러한 시도들은 응용 프로그램을 하나 이상의 시스템에 분산하여 처리하는 분산 컴퓨팅에 객체지향 기술을 도입시켜 분산 객체 컴퓨팅(distributed object computing)이라는 새로운 패러다임을 만들어냈다. 분산 객체 컴퓨팅은 분산된 응용 프로그램을 이루는 각각의 구성 요소들을 객체로 간주하여 분산 객체들 사이에 상호운용성(interoperability)을 제공하는 것을 목표로 한다.

분산 객체 컴퓨팅에 관한 연구와 표준화 작업을 위한 많은 노력들이 있어 왔지만 관련 기술 업체들 사이의 견해 차이로 인하여 많은 어려움을 겪고 있다. 이러한 문제들을 해결하기 위하여 OMG(Object Management Group)라는 협의체를 구성하고 분산 객체 컴퓨팅의 표준 구조로 CORBA(Common Object Request Broker Architecture)를 제안하고 있다[1].

이기종 환경에서 응용 프로그램간의 상호운용성(interoperability)과 다중 객체 시스템의 상호연결성(interconnection)을 제공해주는 메커니즘인 CORBA는 객체지향 분산 컴퓨팅 환경의 실제적인 표준으로 자리 잡아 가고 있다. 그러나, 현재까지 CORBA의 보안에 관한 연구는 거의 되고 있지 않다. 앞으로 분산 컴퓨팅 환경에서 CORBA의 채택이 급증할 것을 감안할 때 이것은 중대한

문제를 야기한다. 본 논문에서는 OMG에서 제안한 CORBA에 보안 서비스를 제공하기 위하여 보안 서버를 제안하고 보안 서버의 인터페이스를 설계하고자 한다.

분산 컴퓨팅 환경은 악의를 가진 사용자로부터 고의적인 공격을 받을 수 있다. 특히, 인증된 사용자로 가장(masquerade)한 사용자의 침입, 정보의 노출(information disclosure), 비인가된 사용자로부터의 무결성 훼손(integrity violation) 등은 분산 컴퓨팅 환경의 대표적인 위협으로 여겨지고 있다. 이러한 문제들로부터 CORBA의 안전성을 보장하기 위해서는 인증, 무결성, 비밀성 등의 보안 서비스들이 제공되어야한다. 그러나, 제공될 보안 서비스들은 CORBA의 구조에 최소한의 영향을 주도록 설계되어야 한다. 이러한 점들을 감안하여 본 연구에서는 CORBA에 보안성을 부여하는 방안으로 보안 서버의 개념을 도입한다.

2. 관련 연구

OMG는 분산 객체 컴퓨팅 환경에서 사용자의 실수나 고의적인 침입에 대하여 안전성을 부여하기 위하여 보안 구조와 제공될 보안 서비스들에 대한 작업을 진행하고 있다. 현재 OMG에서는 CORBA의 보안과 관련하여 White Paper On Security[4]와 Object Services RFP3(Request For Proposal)[3]를 발표해 놓고 있다. [4]에서는 분산 객체 컴퓨팅 환경에서 요구되는 보안 성질들을 일반적 요구와 기능적 요구로 나누어 기술하였고 CORBA의 구성 요소들이 보안 성질에 대한 요구를 만족하기 위하여 보안 서비스들을 이용하는 구조를 보였다. 그리고, [3]에서는 CORBA에 제공될 보안 서비스의 사양을 표준화하기 위한 제안을 나타내고 있다.

[9]에서 Chapin, Herndon 등은 전통적인 보안 요구 분석을 통하여 CORBA에 보안 기능을 부여하고자 했다. [9]에서 분석된 전통적인 보안 요구들에는 접근 제어, 식별 및 인증, 회계 감사, 통신 보안 등이 있다. 또한, 분산 객체 컴퓨팅 환경에서 고려되어져야 되는 보안의 부가적인 요구사항으로 보안 정책의 변환(Policy Translation), 보안 문맥의 위임(Security Context Delegation), 보안 보증(Assurance) 등을 기술하였다.

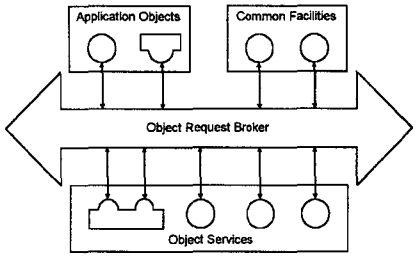
Deng, Bhonsle 등은 [8]에서 객체들 사이의 투명한 요청과 응답을 제공해 주는 객체 요청 중개자(Object Request Broker)에 보안성을 가미한 안전한 객체 요청 중개자(secure ORB)를 설계하는 방안을 제안하였다. 그들이 제안한 안전한 객체 요청 중개자는 SF(System Factory), SM(Security Manager), NS(Naming Service), ACL Base, OSC(Object Security Controller)로 이루어져 있다. 그들은 [8]에서 인증 메커니즘으로는 공개키 암호화 기법을 사용하였고, 접근 제어를 위해서는 접근 제어 목록을 사용하였다.

3. OMG CORBA의 개요

OMG는 분산 객체 컴퓨팅의 표준을 만드는 협의체로서 SunSoft, HP, DEC, IBM, HyperDesk 등 기업체 및 사용자들로 이루어져 있다. OMG에서는 분산 객체 컴퓨팅의 표준화 작업의 일환으로 이기종 환경에서 응용 프로그램간의 상호운용성과 다중 객체 시스템의 상호연결성을 제공해주는 메커니즘인 CORBA를 제안했다. OMG에서는 실제 구현에 관한 기술보다는 사양을 정의하는데 제안된 CORBA도 또한 구체적으로 구현된 것이 아니라 분산 객체 컴퓨팅 구조의 사양만을 기술하고 있다. 그러므로, 실제 CORBA의 구현 문제는 각각의 공급 업체들의 문제로 남게 되는데 이것은 CORBA 사양을 따르는 다른 업체들의 구현 사이에 상호운용성을 보장하지 못하는 문제를 낳게 되었다. 그래서, 최근 CORBA 2.0의 최대 쟁점은 다른 공급 업체들로부터 제공되는 구현 사이의 상호운용성 보장에 있다.

3.1 객체 관리 구조(Object Management Architecture)

OMG에서는 그림 3.1과 같이 응용 객체(Application Objects), 공통 지원 기능(Common Facilities), 객체 서비스(Object Service), 그리고 객체 요청 중개자(Object Request Broker)로 구성된 객체 관리 구조(OMA)를 제안했다[2].



<그림 3.1> 객체 관리 구조

객체 관리 구조를 이루는 네가지 구성 요소들의 기능은 다음과 같다.

첫째, 응용 객체는 OMG에서 정의한 표준을 따르고 사용자에 의해서 작성된 응용 프로그램이나 클래스를 말한다.

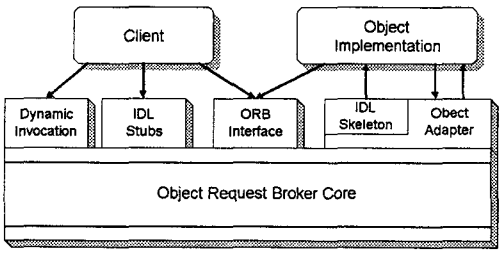
둘째, 공통 지원 기능은 다양한 응용 프로그램에게 공통으로 제공되는 서비스들에 대한 인터페이스를 제공한다. 예를 들면, 복합 문서, 객체 연결(Object Linking)과 삽입(Object Embedding) 등이다. 현재 OMG는 공통 지원 기능에 대한 표준을 개발하고 있다.

셋째, 객체 서비스는 모든 객체들에게 유용한 서비스들에 대한 인터페이스를 제공한다. 예를 들어, 객체들의 위치를 찾는데 도움을 주는 이름 서비스(naming service), 객체 사이의 사건을 전달해 주는 사건 서비스(event service), 객체 저장을 위한 서비스(persistence service) 등이 있는데 이들을 포함한 몇몇 서비스들은 COSS(Common Object Service Specification) Vol. 1[3]과 Vol. 2에 기술되어 있고 보안 서비스를 포함한 몇몇 서비스들에 대해서는 표준화 작업이 진행 중에 있다. 공통 지원 기능과 객체 서비스 사이의 구분은 대개 제공하는 서비스의 수준에 따라 구분할 수 있다. 공통 지원 기능은 보통 고수준의 서비스 집합으로 구성되어 있고 객체 서비스는 저수준의 서비스 집합으로 이루어져 있다.

넷째, 객체 요청 중개자는 객체 사이에서 발생하는 요청과 응답을 투명하게 전달해 주는 메커니즘이다. 이것의 기능은 객체 사이에서 발생하는 정보들의 통로를 제공한다는 점에서 하드웨어의 버스와 같다.

3.2 CORBA

OMG의 CORBA는 분산된 객체들 사이의 투명한 요청과 응답을 제공하는 하나 이상의 객체 요청 중개자를 이용하여 분산 객체 컴퓨팅을 지원해주는 메커니즘이다. 현재 OMG로부터 정의된 CORBA 1.1의 전체적인 구조는 그림 3.2와 같다.



<그림 3.2> CORBA의 구조

그림 3.2에서 나타난 것처럼 CORBA는 객체 요청 중개자 코어(ORB Core), 인터페이스 정의 언어(Interface Definition Language, IDL), 동적 호출 인터페이스(Dynamic Invocation Interface,

DII), 객체 어댑터(Object Adaptor), 인터페이스 저장소(Interface Repository), 구현 저장소(Implementation Repository) 등으로 이루어져 있다.

(1) 객체 요청 중개자 코어

객체 요청 중개자 코어는 객체의 기본적인 표현과 객체의 서비스 요청/응답에 투명성을 제공하는 객체 요청 중개자의 핵심 부분이다. CORBA는 다양한 객체 메커니즘을 지원하기 위해서 객체 요청 중개자 코어와 객체 요청 중개자 코어의 구현상의 차이점을 감추기 위해 사용되는 인터페이스를 구성요소로 객체 요청 중개자를 구성한다.

(2) 인터페이스 정의 언어

분산 객체 컴퓨팅에서는 분산된 응용 프로그램들을 객체로 간주하는데 객체들은 캡슐화 되어 있으므로 외부에서 응용 프로그램의 서비스를 이용하기 위해서는 응용 프로그램의 인터페이스를 이용한다.

CORBA에서는 객체의 인터페이스를 정의하기 위해서 IDL이라는 인터페이스 정의 언어를 사용하는데 그것은 C++과 매우 유사하다. IDL로 기술된 객체의 인터페이스는 객체가 다른 객체에게 제공하는 서비스(메소드)와 서비스를 호출할 때 요구되는 매개 변수 등을 나타낸다. 그리고, IDL로 정의된 객체의 인터페이스는 IDL 컴파일러에 의해서 클라이언트의 스터브(stub)와 서버의 골격(skeleton)을 생성한다. 이 정보들은 객체의 서비스에 대한 요청을 서비스를 구현한 실제 메소드로 연결하기 위하여 사용된다.

(3) 동적 호출 인터페이스

클라이언트가 서버의 서비스를 호출하는 방법은 크게 정적 호출과 동적 호출이 있다. 정적 호출은 컴파일 시간에 인터페이스가 알려진 서버의 서비스를 호출하기 위하여 IDL 컴파일러에 의해 생성된 스터브를 이용하는 방법이다. 한편, 동적 호출은 컴파일 시간에 인터페이스가 알려지지 않은 서버의 서비스를 인터페이스 저장소에 저장된 정보를 이용하여 실행 시간에 호출하는 방법이다. 동적 호출 인터페이스는 실행 시간에 동적 호출을 할 수 있도록 하는 기능을 제공한다.

(4) 객체 어댑터

객체 어댑터는 객체 구현이 객체 요청 중개자의 서비스를 이용할 수 있는 방법을 제공한다. 객체 구현이 객체 어댑터를 통하여 이용할 수 있는 객체 요청 중개자의 서비스는 다음과 같다.

- 객체 참조의 생성과 해석
- 메소드 호출
- 객체 사이의 보안
- 객체와 구현의 활성화
- 객체 참조와 구현 사이의 사상
- 객체 구현의 등록

객체 구현들은 본질적으로 구현마다 객체 요청 중개자의 다른 서비스나 다른 인터페이스를 요구한다. 이런 다양한 요구를 만족시키기 위하여 여러 종류의 객체 어댑터들이 있을 수 있다. 자주 사용되는 객체 어댑터로는 기본 객체 어댑터(Basic Object Adapter), 라이브러리 객체 어댑터(Library Object Adapter), 객체지향 데이타베이스 어댑터(Object-Oriented Database Adapter) 등이 있다.

(5) 인터페이스 저장소

인터페이스 저장소는 객체 인터페이스의 정보가 실행 시간에 사용될 수 있도록 저장하는 장소이다. 인터페이스 저장소에 저장된 정보는 클라이언트가 서버의 서비스를 동적으로 호출할 때 서버의 서비스에 대한 인터페이스를 구성하는데 사용된다.

(6) 구현 저장소

구현 저장소는 객체 요청 중개자가 클라이언트에 의해서 요청된 서비스를 제공하기 위해서 객체 구현을 찾고 활성화시키는데 필요한 정보를 저장하고 또한, 객체 요청 중개자와 운영 환경에 관련된 종속적인 정보를 저장한다. 그리고, 구현 저장소는 객체 구현의 설치를 위한 연산과 객체 구현의 활성화와 실행에 관련된 정책의 제어를 위한 연산을 제공한다.

4. 분산 객체 컴퓨팅 시스템 보안

분산 객체 컴퓨팅 시스템의 본질적인 특징 중 하나는 개방성이다. 개방성은 분산된 시스템을 쉽게 연결할 수 있게 해주는 반면 악의를 가진 사용자들의 공격에 시스템을 노출시키는 단점이 있다. 특히, 분산 객체 컴퓨팅 시스템은 보안상의 침해를 당할 경우 하나의 독립된 시스템보다 더욱 큰 영향을 받게 되므로 분산 시스템의 보안은 반드시 고려되어야할 문제이다.

분산 객체 컴퓨팅 시스템에 보안성을 부여하기 위해서 먼저 분산된 시스템에서 나타날 수 있는 위협적인 요소를 살펴보고 제공되어야할 보안 서비스를 알아보고자 한다.

4.1 분산 객체 컴퓨팅 시스템에서의 위협

분산 객체 컴퓨팅 시스템에서 나타날 수 있는 위협은 크게 다음 세가지로 나눌 수 있다 [8,9,12].

- ▨ 인증된 사용자로 가장(masquerade)
  인증되지 않은 사용자가 인증된 사용자로 가장하여 그 사용자의 권한을 부여받게 되면 인증된 사용자가 할 수 있는 모든 일을 할 수 있게 된다.
- ▨ 정보의 노출(information disclosure)
  객체 사이에 교환되는 메시지의 도청이나 정보의 비인가된 접근 등에 의해서 정보가 노출될 위험이 있다.
- ▨ 비인가된 사용자로부터의 무결성 훼손(integrity violation)
  비인가된 정보의 생성이나 변경 그리고 삭제에 의해서 정보의 무결성은 훼손될 수 있다.

4.2 보안 서비스

앞 절에서 살펴본 위협적인 요소에 대해서 분산 객체 컴퓨팅 시스템의 안전성을 보장하기 위해서는 다음과 같은 보안 서비스가 제공되어야한다[8,9,12].

(1) 인증 서비스(Authentication Service)

인증은 서비스를 요청하는 사용자가 바로 그 사용자라는 신용을 얻는 수단이다. 그러므로, 인증 서비스가 제공되는 시스템은 인증된 사용자로 가장하는 위협에 대처할 수 있다. 흔히 쓰이는 인증 서비스로는 패스워드를 사용하는 기법이 있으나 다양한 공격에 취약하다. 그런 취약성을 극복한 방법으로는 대칭키 암호화 기법을 사용하는 Kerberos가 있다. 또한, 디지탈 서명에 사용되는 공개키 암호화 기법은 공유하는 비밀 정보 없이 인증 서비스를 제공한다.

(2) 접근 제어 서비스(Access Control Service)

접근 제어는 어떤 사용자가 어떤 서버 또는 서버의 어떤 서비스를 접근할 수 있는지 관리하는 수단이다. 접근 제어 서비스는 정보의 노출이나 무결성 훼손 같은 위협으로부터 시스템을 보호한다. 접근 제어 정책은 크게 자율적 접근 제어(Discretionary Access Control)와 강제적 접근 제어(Mandatory Access Control)로 나눌 수 있다. 자율적 접근 제어는 사용자 스스로의 판단으로 그들의 자료를 누가 접근할 수 있도록 할 것인가를 결정하여 시스템에게 알려주므로써 접근 제어를 하는 반면에 강제적 접근 제어는 보안 관리자나 시스템에 의해서 부여된 사용자와 데이타들의 고정된 보안 속성을 사용하여 접근 제어를 한다. 또, 다른 접근 제어 정책으로는 자율적 접근 제어 정책을 확장한 역할기반 접근 제어(RAC: Role-based Access Control)가 있다.

(3) 정보의 무결성 서비스(Information Integrity Service)

정보의 무결성을 제공하기 위해서 전송될 정보의 체크섬(check sum)을 구하고 얻어진 체크섬을 암호화하여 전송한다. 만일 제삼자로부터 정보가 변경된다면 암호화되어 전송된 체크섬의 값은 변경된 정보의 체크섬 값과 다르게 되어 정보의 변경에 대한 여부를 쉽게 알 수 있다.

(4) 정보의 비밀성 서비스(Information Privacy Service)

정보의 비밀성을 제공하기 위해서는 전송될 정보를 통신하는 관련 객체들만이 알고 있는 키를 사용하여 암호화시켜 전송한다. 이것은 제삼자가 전송되는 정보를 도청하더라도 암호화된 정보를 제삼자는 풀 수 없으므로 도청된 정보를 쓸모 없게 만든다. 이런 점에서 정보의 비밀성 서비스는 정보의 무결성 서비스보다 더 높은 수준의 보안 서비스로 고려되어진다. 자주 쓰이는 암호화 기법으로는 대칭키 암호화 기법과 공개키 암호화 기법이 있다.

(5) 회계 감사 서비스(Audit Service)

회계 감사는 시스템에서 발생하는 보안 관련 사건들을 추적하는 수단이다. 회계 감사 기능을 사용하여 시스템의 보안 정책을 위반한 사용자를 찾아낼 수 있다. 회계 감사 기능을 발휘할 수 있도록 하기 위해서 제공되어야 할 최소의 정보는 클라이언트와 서버의 식별자, 클라이언트를 생성한 사용자의 식별자, 교환된 정보 등이다.

5. CORBA 환경에서 보안 서버

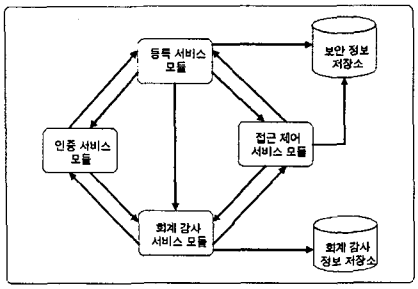
분산 객체 컴퓨팅 시스템의 사실상 표준인 CORBA 사양 1.1은 보안에 관한 자세한 언급이 없다. 다만, OMG에서 객체 서비스에 대한 표준화 작업의 일환으로 보안 서비스에 대한 논의를 하고 있을 뿐이다. 그러나, 지금까지 OMG에서 발표된 표준안들을 참고해 보면 발표될 보안 서비스의 표준안도 역시 사양을 정의하는 수준에 머무를 것으로 예상된다.

본 논문에서는 CORBA의 구조에 제4절에서 언급된 보안 서비스를 제공하기 위하여 보안 서버를 제안하고 보안 서버를 이루는 서비스들에 대한 인터페이스를 정의하고자 한다. 본 논문에서는 CORBA에 요구되는 보안 서비스를 제공하기 위하여 안전한 객체 요청 중개자를 설계하기보다는 현재 구현된 객체 요청 중개자를 이용하여 CORBA에 보안성을 부여하기 위한 하나의 방법으로 보안 서버를 도입한다. 보안 서버의 경우는 안전한 객체 요청 중개자보다 실제로 적용하기에 적합한 면이 있으나 완전한 보안 기능을 부여하지 못하는 단점이 있다.

CORBA에 보안성을 부여하기 위하여 보안 서버를 도입한 방법은 다음과 같은 장점들을 갖는다. 첫째, 제안된 보안 서버는 CORBA의 사양을 따른다. CORBA 사양 1.1은 인증 서비스에 대해서는 객체 요청 중개자가 제공하도록 하고 접근 제어는 각각의 서버에서 제공하도록 하고 있다. 둘째, 객체지향의 관점에서도 객체의 접근 제어는 각각의 객체 내에서 행해지는 것이 바람직하다. 셋째, 접근 제어를 서버가 처리하도록 하므로써 객체마다 접근 제어 단위를 다르게 적용할 수 있다.

먼저 이 절에서는 제안된 보안 서버를 이루는 모듈 사이의 관련성을 설명하고 보안 서버와 기존의 CORBA의 구조, 클라이언트, 그리고 다른 서버들과의 관계를 설명한다. 그리고, 보안 서버를 이루는 서비스들의 인터페이스는 제6절에서 정의한다.

제안된 보안 서버는 인증 서비스 모듈, 등록 서비스 모듈, 접근 제어 모듈, 회계 감사 서비스 모듈로 구성된다. 서비스 모듈 각각의 기능은 다음과 같고 그들 사이의 관계는 그림 5.1에 표현되어 있다.



<그림 5.1> 보안 서비스 모듈간의 관계

(1) 인증 서비스 모듈

인증 서비스 모듈은 사용자가 시스템에 로그인할 때 사용자의 인증 허용 여부를 판단하기 위한 서비스를 제공한다. 이러한 인증 서비스를 제공하기 위해서는 시스템 운영 환경에 종속적인 요소를 포함하게 된다.

그러나, 클라이언트와 서버 사이의 인증을 비롯하여 두 객체 사이에서 교환되는 정보의 무결성이나 비밀성을 제공하는 서비스는 제공하지 않는다. CORBA 구조에서 제안된 보안 서버에서 고려되지 않은 서비스를 포함하는 완전한 인증 서비스 기능을 제공하기 위해서는 객체 사이의 투명한 요청과 응답을 제공하는 객체 요청 중개자가 그러한 기능을 제공하도록 구현되어야 한다. [8]에서는 완전한 인증 서비스를 제공하기 위하여 안전한 객체 요청 중개자를 제안하였다. 그러나, 본 논문에서는 객체 요청 중개자의 구조에 대한 고려보다는 현재 구현된 객체 요청 중개자에 보안 서비스를 첨가할 수 있는데 중점을 두었다.

인증된 사용자로부터 생성되는 모든 클라이언트는 따로 인증의 절차를 거치지 않고 인증된 것으로 간주하고 시스템으로부터 인증을 획득한 사용자의 보안 정보를 상속받는다.

(2) 등록 서비스 모듈

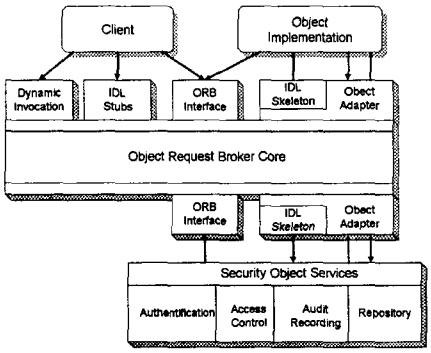
서버는 시스템에 설치될 때 보안과 관련된 정보를 등록 서비스 모듈을 사용하여 보안 서버에 등록해야 한다. 등록된 정보는 클라이언트가 서버의 서비스를 요구할 때 요청된 서비스에 대한 접근 허용 여부를 판단하기 위하여 사용된다. 서버의 보안 관련 정보들은 보안 서버의 보안 정보 저장소(Security Information Repository)에 저장된다. 서버는 필요에 따라서 등록된 접근 제어에 관한 정보를 변경할 수도 있다. 서버가 등록해야 할 정보로는 서버의 식별자, 접근 제어와 관련된 정보, 그리고 서버가 시스템에 등록될 때 사용된 서버의 비밀키 등이다.

(3) 접근 제어 서비스 모듈

서버는 클라이언트로부터 서비스의 요청을 받으면 클라이언트가 서비스를 이용할 수 있는 정당한 사용자인가를 판별하기 위하여 접근 제어 서비스 모듈을 활용한다. 이때, 접근 제어 서비스 모듈은 서버가 보안 정보 저장소에 등록한 보안 관련 정보를 사용한다.

(4) 회계 감사 서비스 모듈

회계 감사 서비스 모듈의 기능은 일련의 보안과 관련된 사건이 발생될 때마다 사건과 사건에 참여된 객체들을 기록하므로써 시스템의 보안 정책을 위반한 사용자를 발견하는데 있다. 제안된 보안 서버의 다른 서비스 모듈들은 외부로부터 서비스 요청을 받을 때마다 회계 감사 서비스 모듈에 서비스를 요청한 객체의 식별자와 객체의 사용자 식별자, 요구된 서비스의 정보 등을 기록한다. 예를 들면, 접근 제어 서비스 모듈은 서버로부터 서비스를 요청 받을 때 서버의 서비스를 호출한 클라이언트의 식별자, 클라이언트의 사용자 식별자, 그리고 서버의 식별자 등을 회계 감사 서비스 모듈에 저장해야 한다. 회계 감사 정보는 회계 정보 저장소(Audit Information Repository)에 저장된다.



<그림 5.2> CORBA 환경에서 보안 서버

그림 5.2는 제안된 보안 서버가 CORBA 구조 내에서 동작하는 것을 나타낸다. 그림 5.2에서 사용자는 인증 서비스 모듈을 이용하여 시스템에 로그인하므로써 시스템에 대한 인증을 획득하게 되고 인증된 사용자로부터 생성된 클라이언트는 또 다른 인증 과정을 거치지 않은 채 인증된 것으로 간주한다. 클라이언트는 필요에 따라서 서버의 서비스를 호출하게 되는데 요청된 서비스는 객체 요청 중개자에 의해서 투명하게 서버의 서비스로 연결된다. 이때, 서비스를 요청 받은 서버는 클라이언트의 요청이 타당한 접근인가를 판단하기 위하여 접근 제어 서비스 모듈을 이용한다. 그 결과에 따라서 요청을 받은 서버는 클라이언트에게 서비스를 제공하거나 하지 않을 수 있다. 물론 서비스를 제공하는 서버는 시스템에 설치될 때에 자신의 접근 제어 정보를 포함한 보안 정보를 등록 서비스 모듈을 이용하여 보안 정보 저장소에 등록하게 된다. 회계 감사 서비스는 사용자의 시스템 로그인 사건, 서버의 보안 정보 등록 사건, 그리고 서버가 접근 제어 서비스를 호출하는 사건 등을 회계 감사 정보 저장소에 기록한다.

6. 보안 서버의 인터페이스

분산 객체 컴퓨팅에서 서버는 다른 객체가 서비스를 이용할 수 있도록 인터페이스를 제공해야 한다. 이 절에서는 서버를 구성하는 각각의 모듈에서 제공해야할 기능들을 정의하고 그에 따른 인터페이스를 정의한다.

6.1 보안 서버의 모듈 기능 정의

보안 서버를 이루는 각각의 모듈들이 외부 객체에게 제공해야할 서비스들은 다음과 같다.

▨ 인증 서비스 모듈
- 로그인 검사 기능

▨ 등록 서비스 모듈
- 보안 관련 정보 등록 기능
- 보안 관련 정보 수정 기능
- 보안 관련 정보 판독 기능

▨ 접근 제어 서비스 모듈
- 접근 제어 기능
- 접근 제어 단위 설정 기능

▨ 회계 감사 서비스 모듈
- 회계 감사 정보 판독 기능

6.2 보안 서버의 인터페이스 정의

CORBA의 사양을 따르는 보안 서버의 인터페이스는 그림 6.1과 같다. 그림 6.1에서 사용되고 있는 사용자 정의 자료형은 보안 서버 구현 시에 기술될 내용으로 남겨두었다. 또한, 인터페이스의 연산들이 발생시키는 예외 사항을 처리하기 위하여 연산의 요구에 맞는 예외 사항 처리 루틴을 정의하였다.

```
interface Object;                                   // 객체 참조
interface Principal;                                // 서버를 호출한 객체의 정보
typedef struct _AccessControlInfo {
            // 접근 제어 정보에 관한 구조체
            // 보안 서버 구현 시에 정의
        } AccessControlInfo;
typedef union _AccessControlGranular {
            // 접근 제어 단위에 대한 유니온
            // 보안 서버 구현 시에 정의
        } AccesssControlGranular;
typedef struct _AuditInfo {
            // 회계 감사 정보를 위한 구조체
            // 보안 서버 구현 시에 정의
        } AuditInfo;

module SecuityServer {
        exception LoginFail { };
        exception RegUpdateFail { };
        exception GranularSetFail { };
        exception AuditReadFail { };
        typedef string Key;
        typedef string Operation;
        interface AuthenticationModule {
           boolean system_login(in Key password) raises(LoginFail);
        };
        interface RegistryModule {
           void sec_info_reg(in Object server, in Key serverkey, in AccessControlInfo info);
           Boolean sec_info_update(in Object server, in AccessControlInfo newinfo)
                   raises(RegUpdateFail);
           Boolean sec_info_read(in Object server, out AccessControlInfo info);
        };
        interface AccessControlModule {
           Boolean access_control(in Object server, in Principal client, in Operation op);
           Boolean access_control_granular(in Object server, in AccessControlGranular gran)
                   raises(GranularSetFail);
        };
        interface AuditModule {
           Boolean audit_info_read(in Object server, out AuditInfo auditinfo)
                   raises(AuditReadFail);
        };
};
```

<그림 6.1> 보안 서버의 인터페이스 정의

7. 결론 및 추후연구

정보 처리 기술의 발달과 컴퓨터 네트워크의 발달은 분산 컴퓨팅이라는 새로운 컴퓨팅 환경을 낳았다. 그리고, 분산 컴퓨팅을 지향하는 사람들은 분산 컴퓨팅과 객체지향을 통합하여 분산 객체 컴퓨팅 환경을 생성시켰다. 분산 객체 컴퓨팅은 분산된 응용 프로그램을 이루는 각각의 구성 요소들을 객체로 간주하여 분산 객체들 사이에 상호운용성을 제공하는 것을 목표로 한다.

분산 객체 컴퓨팅 환경에 관한 표준화를 위하여 업체와 사용자로 구성된 협의체인 OMG에서는 분산 객체 컴퓨팅의 표준 구조로 CORBA를 제안하고 있다. CORBA는 이기종 환경에서 응용 프로그램간의 상호운용성과 다중 객체 시스템의 상호연결성을 제공해주는 메커니즘이다. CORBA는 객체지향 분산 컴퓨팅 환경의 실제적인 표준으로 자리잡아 가고 있다. 그러나, 현재까지 CORBA의 보안에 관한 연구는 거의 되고있지 않다. 앞으로 분산 컴퓨팅 환경에서 CORBA의 채택이 급증할 것을 감안할 때 이것은 중대한 문제를 야기한다.

본 논문에서는 OMG에서 제안한 CORBA에 보안 서비스를 제공하기 위하여 보안 서버를 제안하고 보안 서버의 인터페이스를 설계했다. 그러나, 제안된 보안 서버는 현재 구현된 객체 요청 중개자를 기반으로 하고 있기 때문에 완전한 보안 서비스를 제공하지 못하고 있다. 특히, 클라이언트와 서버 사이의 인증과 인증된 객체 사이에 교환되는 정보의 무결성이나 비밀성이 제공되지 못한 점은 개선되어야 한다.

추후 연구에서는 본 논문에서 고려되지 못한 보안 서비스들을 제공하는 완전한 보안 서버를 구축하기 위한 노력이 필요하다. 그러한 노력의 일환으로 안전한 객체 요청 중개자 구현에 관한 연구를 계속 추진할 계획이다.

<참 고 문 헌>

[1] Object Management Group, "Framingham, Common Object Request Broker : Architecture and Specification", Document number 91.12.1, 1991.

[2] Object Management Group, Framingham, "Object Management Architecture Guide", Document number 92.11.1, 1992

[3] Object Management Group, Framingham, "Common Object Services Specification, Volume 1", 1992

[4] Object Management Group, Framingham, "White Paper on Security", Document number 94.4.16, May 1994.

[5] Object Management Group, Framingham, "Object Services Request For Proposal 3". June 1994.

[6] Thomas J. Mowbray, Ron Zahavi, "The Essential CORBA - Systems Integration Using Distributed Objects", John Wiley & Sons, Inc, 1995

[7] Randy Otte, Paul Patrick, Mark Roy, "Understanding CORBA - The Common Object Request Broker Architecture", Prentice Hall PTR, 1996

[8] Robert H. Deng, Shailendra K. Bhonsle, Weiguo Wang, Aurel A. Lazar, "Integrating Security in CORBA Based Object Architectures", Proceedings 1995 IEEE Symposium on Security and Privacy, May 1995.

[9] Susan L. Chapin, William R. Herndon, LouAnna Notargiacomo, Melony L. Katz, Thomas J. Mowbray, "Security For The Common Object Request Broker Architecture (CORBA)", Tenth Annual COMPUTER SECURITY APPLICATIONS Conference, December, 1994

[10] Ravi S. Sandhu, Pierangela Samarati, "Access Control: Principles and Practice", IEEE Communications Magazine, Vol. 32, No. 9, pp.40-49, September 1994.

[11] B. Clifford Neuman, Theodore Ts'o, "Kerberos: An Authentication Service for Computer Networks", IEEE Communications Magazine, Vol. 32, No. 9, pp.33-38, September 1994.

[12] W. Ford, "Computer Communications Security: Principles, Standard Protocols, and Techniques", Prentice Hall PTR, 1994.