情 報 保 護 學 會 誌
第 14 券 第 5 號, 2004. 10

유비쿼터스 보안 미들웨어의 접근통제 기능에 관한 고찰

박 희 만*, 이 영 록*, 이 형 효**, 노 봉 남*

요 약

컨텍스트 인식과 적응에 더불어 유비쿼터스 컴퓨팅 실현의 필수 요소는 보안이다. 유비쿼터스 환경에서 자원과 서비스는 응용이 실행되는 지역에 있을 수도 있지만 대게는 물리적으로 분산된 환경 안에 존재하게 된다. 유비쿼터스 환경은 응용이 실행되는 환경에서의 보안은 물론이고 분산 환경에서의 자원과 서비스에 대한 보안도 필수적으로 고려되어야 한다. 유비쿼터스 컴퓨팅의 새로운 취약점을 다루기 위해 유비쿼터스 컴퓨팅 환경의 보안은 미들웨어의 추가사항으로 고려되는 것보다는 미들웨어의 설계단계에서부터 고려되어야한다. 또한 유비쿼터스 환경에서 응용과 서비스는 미들웨어에 항상 고정되어 있는 것이 아니라 동적으로 결합되고 분리되므로 사용자가 실행한 응용이 신뢰할 수 있는지와 그 응용이 이용하려고 하는 서비스에 대해 접근 권한이 있는지는 중요하다.

기존의 유비쿼터스 미들웨어는 동적으로 변하는 컨텍스트에 대해 응용이 잘 적응 할 수 있는 구조로는 되어있지만 응용이 이용하려는 서비스에 접근 권한이 있는지에 대해서는 조사하지 않기 때문에 서비스는 여러 보안위협에 대해 안전할 수 없다. 본 논문은 유비쿼터스 환경에서 편재된 자원과 서비스를 사용하는 프로그램의 실행여부를 접근통제 관점에서 분석하고 유비쿼터스 보안 미들웨어의 접근통제 기능에 대해 고찰한다.

Ⅰ. 서 론

유비쿼터스 컴퓨팅[1, 2, 3] 환경은 컴퓨터나 서로 이질적인 기기들이 일상생활 속에 스며들어 있어서 사용자가 자신이 인식하지 못하는 사이에 여러 기기들의 도움을 받을 수 있으며, 기존의 기기 조작의 혼란으로부터 벗어날 수 있게 해준다. 유비쿼터스 환경에서, 응용은 사용자의 개입 없이도 사용자의 활동을 지원하기 위해 이용 가능한 자원과 서비스를 효과적으로 이용할 수 있어야 한다. 그런 환경에서의 자원과 서비스는 언제든지 시스템에 첨가되거나 제거될 수 있어야 하며, 사용자의 선호도나 개체를 둘러싼 컨텍스트들의 변화도 즉시 시스템에 반영될 수 있어야 한다. 따라서 그런 환경에서의 응용은 컨텍스트를 인식하고 그에 따라 자신을 적응시키는 것이 필수적이다. 이런 응용을 지원하기 위해서 Aura, Gaia, M3-RTE등의 미들웨어가 제안되어 연구되고 있다.

컨텍스트 인식과 적응에 더불어 유비쿼터스 컴퓨팅 실현의 필수 요소는 보안이다. 유비쿼터스 환경에서 자원과 서비스는 응용이 실행되는 지역에 있을 수도 있지만 대게는 물리적으로 분산된 환경 안에 존재하게 된다. 유비쿼터스 환경은 응용이 실행되는 환경에서의 보안 뿐만아니라 분산 환경에서의 자원과 서비스에 대한 보안이 필수적으로 고려되어야 한다. 하지만 이러한 연구들에서는 보안 요소를 찾기 어렵다.

기존의 미들웨어는 설계시에 보안요구사항을 고려하지 않았거나, 반영하였다하더라도 하나의 보안 담당 컴포넌트를 끼워넣는 수준의 보안을 반영하고 있다. 유비쿼터스 컴퓨팅의 새로운 취약점을 다루기 위해 유비쿼터스 컴퓨팅 환경의 보안은 미들웨어의 추가사항으로 고려되는 것보다는 미들웨어의 설계단계에서부터 고려되어야한다.

유비쿼터스 환경의 보안 미들웨어가 보안위협에 대처하기 위해서는 기밀성, 무결성, 가용성, 익명성, 부인방지의 보안서비스가 제공되어야 한다. 또한 유비쿼터스 환경에서 응용과 서비스는 미들웨어에 항상 고정

* 전남대학교 정보보호협동과정 ({hareup, yrlee, bongnam} @athena.jnu.ac.kr)
** 원광대학교 정보 전자상거래학부 (hlee@wonkwang.ac.kr)

되어 있는 것이 아니라 동적으로 결합되어지고 동적으로 분리되어지므로 사용자가 실행한 응용이 신뢰할 수 있는지의 인증과 그 응용이 이용하려고 하는 서비스에 대해 접근 권한이 있는지의 접근통제는 중요하다.

기존의 유비쿼터스 미들웨어는 동적으로 변하는 컨텍스트에 대해 응용이 잘 적응 할 수 있는 구조로는 되어있지만 응용이 이용하려는 서비스에 접근 권한이 있는지에 대해서는 조사하지 않기 때문에 서비스는 여러 위협에 대한 안전할 수 없다.

본 논문에서는 유비쿼터스 환경에서 편재된 자원과 서비스를 사용하는 프로그램의 실행여부를 접근통제 관점에서 분석하고 유비쿼터스 보안 미들웨어의 접근통제 기능 및 구조에 대해 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 보안 프레임웍과 관련된 선행 연구에 대해 알아본다. 3장에서는 유비쿼터스 컴퓨팅 환경에서의 보안 요구사항에 대해 알아보고, 4장에서는 보안 요구사항을 반영한 유비쿼터스 보안 프레임웍에 대해 기술한다. 5장에서는 유비쿼터스 컴퓨팅 환경의 접근통제를 위한 보안 시나리오를 도출하고 RBAC을 이용해 어떻게 접근통제를 수행할 수 있는지 예를 들어 설명한다. 그리고 6장에서 결론을 짓는다.

Ⅱ. 관련 연구

유비쿼터스 환경은 수 많은 이질성-기기나 네트워크의 이질성-에도 불구하고, 사용자나 기기의 이동이나 변화에 의해서 생기는 환경의 변화에 응용이 자신을 적응시킬 수 있는 환경이다. 이 장에서는 환경의 변화에도 반응할 수 있는 컴퓨팅 환경을 제안한 연구들을 살펴본다.

1. PIMA 프로젝트

PIMA 프로젝트[4]는 기기, 응용, 환경에 대한 새로운 관점을 모바일 컴퓨팅에 적용시키고자 한다. 기기는 응용 공간이나 데이터 공간으로 들어가기 위한 출입구의 관점으로, 응용은 사용자가 작업을 수행한다는 관점으로, 컴퓨팅 환경은 사용자를 둘러싸고 있는 사용자의 향상된 정보, 물리적 환경의 관점으로 제시하고 있다.

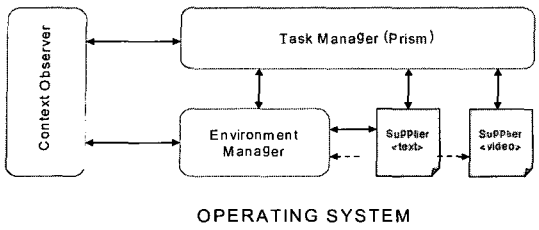
PIMA 프로젝트는 이러한 새로운 관점을 지원하기 위해 새로운 응용 모델을 제안한다. PIMA 프로젝트가 생각하는 응용을 설계하기 위해서, 응용의 생명주기(life-cycle)를 고려하고 있다. 이 생명주기는 설계시점(design-time), 적재시점(load-time), 수행시점(run-time)의 세부분으로 나뉜다. 개발자가 응용을 만들고, 유지하고, 향상시킬 때가 설계시점에 고려할 내용이다. 적재시점 때 시스템은 응용 컴포넌트를 하드웨어의 응용 인스턴스(instance)로 구성하고, 적응시키고, 적재한다. 수행시점 때, 최종 사용자(end-user)는 응용을 불러오고 그것의 기능을 이용한다. 시스템은 응용이 실행할 수 있는 환경을 제공하고, 그 환경에 따른 변화에 응용을 적응시킨다.

컴퓨팅 환경이 컨텍스트를 인식한다면, 응용 설계시점에 개발자는 이용 가능한 서비스에 대해 미리 전제할 필요가 없게된다. 따라서 응용이 실행하는데 필요한 서비스들은 명백히 이름 지어지는 것이 아니라, 추상적인 방법으로 기술되어야한다는 필요성을 역설하고 있다. 더군다나, 설계시점에 개발자에게 알려지지 않거나 이용가능하지 않은 서비스라도 수행시점에 응용에 이용 가능할 수 있다. 다시 말하면, 그 서비스들이 작업에 유용한 것들이라면 응용은 그런 서비스들을 사용할 수 있어야한다. 설계시점의 개발자는 응용을 향상시키는 선택적인 서비스들을 추상적으로 명기할 수 있다.

적재시점에 시스템은 응용이 이용 가능한 서비스를 동적으로 발견해야하고, 시스템은 응용을 이용 가능한 기기자원에 적응시켜야 한다. 응용은 이용할 서비스나 자원을 요구사항으로 기술하고 있고, 기기는 자신의 성능을 기술하며, 중재 알고리즘은 이들 제약조건들 사이에서 어울리는 것들을 교섭해서 연결시켜줘야 한다. 수행시점에 시스템은 밧데리의 소모나 서비스 충돌과 같은 그런 예상하지 못한 실패를 다루어야 한다.

2. Aura

Aura[5] 기반구조 설계의 가장 중요한 도전 중의 하나는 상대적으로 새로운 성질 속성인 사용자 이동성을 지원하는 것이다. Aura는 모바일 사용자들로 하여금 유비쿼터스 환경을 가장 잘 이용하도록 사용자들이 성능이나 자원의 차이와 동적인 변화성을 관리하지 않아도 되게 한다. 특히 Aura가 제안한 프레임워크는 많은 중요한 이점을 지니고 있다. 명시적으로 사용자 태스크를 나타냄으로써, 사용자의 의향(intent)을 캡처할 수 있는 도구를 제공해준다. 따라서 새로운 환경에 사용자가 진입했을 때 적절한 설정을 쉽게 찾을 수 있도록 이 지식을 사용할 수 있다.



〔그림 1〕 Aura 구조

그림 1은 Aura의 프레임워크를 위에서 내려다보는 그림을 보여준다. 4개의 컴포넌트 타입이 있는데, 첫째로 프리즘이라고 불리는 작업관리자가 개인비서용의 Aura (Personal Aura) 개념을 구체적으로 표현한다. 두 번째로 컨텍스트 관찰자(Context Observer)는 물리적인 컨텍스트에 대한 정보를 제공하고, 물리적 컨텍스트의 적절한 이벤트를 프리즘과 환경관리자(Environment Manager)에게 알려준다. 셋째, 환경관리자는 환경으로의 게이트웨이를 구현하고, 넷째로 공급자(Supplier)는 텍스트나 편집 그리고 비디오 재생과 같이 태스크가 이루고 있는 추상 서비스를 제공한다.

논리적인 관점으로 보면, 환경은 환경관리자, 컨텍스트 관찰자, 그리고 작업관리자 타입의 인스턴스를 가지고 있다. 환경의 경계가 관리자에 의해 정의되었을지라도, 그들은 층(floor)이나 빌딩과 같은 어떤 물리적인 영역에 대응할 수 있다. 각 환경은 다수의 서비스 공급자(Supplier)를 가지고 있을 수 있는데, 그것이 더 많은 공급자를 가질수록 더 풍부한 환경이 된다. 환경관리자는 사용자의 작업을 위해 필요한 공급자를 프리즘이 요청할 때마다 원격 공급자를 찾아서 안내해준다.

3. Gaia

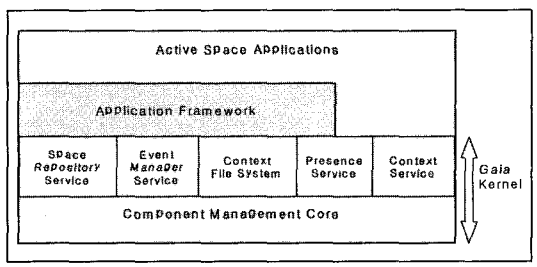
Gaia〔6, 7〕는 소프트웨어 개체들과 물리적 공간 안의 이기종 네트워크 장치들을 연결시켜주는 분산 미들웨어 기반구조이다. Gaia는 자원을 요청하고 이용하는 모바일 어플리케이션을 개발하기 위해 현재의 컨텍스트를 사용하여 자원인식과 컨텍스트 인식을 할 수 있도록 해주기 위한 프래임워크를 제공한다. Gaia는 유비쿼터스 컴퓨팅 공간 관리와 어플리케이션 개발을 단순화시키기 위해 기존의 운영체제의 개념을 확장하고 있다. Gaia의 주요한 특징은 작업을 수행하기 위해 단순히 독립적인 서비스를 이용하는 것에 그치지 않고 그들 서비스 간에 상호작용을 통해서 작업을 수행하도록 설계하였다는 것이다.

Gaia는 사용자 중심 활성공간(active space) 어플리케이션을 지원하기 위해 활성공간(active space)의 자원과 서비스를 관리한다. 또한 활성공간(active space)에 대한 장소, 컨텍스트, 이벤트와 정보의 저장소를 위한 서비스도 제공한다.

그림 3에서는 3개의 주요한 구성요소인 Gaia 커널, Gaia 응용 프래임웍, 응용을 보여주고 있다. Gaia 커널은 분산객체를 위한 관리와 배포 시스템과 모든 어플리케이션이 사용하는 기본 서비스들을 포함한다. 컴포넌트 관리 코어(Component Management Core)는 Gaia의 모든 컴포넌트나 어플리케이션들을 동적으로 적재, 이동, 생성, 폐기한다. Gaia 컴포넌트는 분산 객체이고, 떨어진 원거리 상호작용을 지원하기 위한 통신 미들웨어를 필요로 한다.

Gaia는 CORBA를 이용하여 구현되었지만, SOAP, RMI를 포함한 다른 통신 미들웨어 기반구조로 이식하는 것이 가능하다. CORBA는 분산 객체 상호작용을 위해 이식 가능한 기반구조를 지원한다. Gaia는 이벤트 관리자 서비스(Event Manager Service), 존재 서비스(Presence Service), 컨텍스트 서비스(Context Service), 공간 저장소 서비스(Space Repository Service), 컨텍스트 파일시스템(Context File System)의 5가지 기본 서비스를 제공한다. Gaia 소프트웨어 기반구조는 임의의 물리적 공간 안에서 Gaia 커널의 실행을 시작시키는 초기적재(bootstrap) 기법으로 구현되어 있다.

Gaia의 어플리케이션은 활성공간안에서 실행되는 어플리케이션을 지원하기 위해 Gaia 어플리케이션 프래임워크를 구성하는 컴포넌트의 집합을 이용한다. 이러한 프래임워크은 이동성, 적응성, 동적 바인딩을 지원한다. Active Space 어플리케이션 계층은 어플리케이션을 포함하고, Gaia 커널 서비스를 통해서 어플리케이션을 등록, 관리, 통제하는 기능을 제공한다.



〔그림 2〕 Gaia 구조

4. M3-RTE

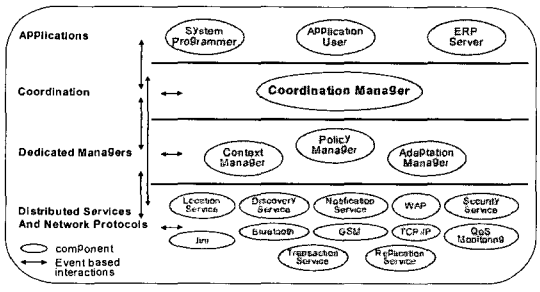
유비쿼터스 미들웨어인 M3-RTE〔8〕는 단일구조의 구성을 피하기 위해 일반성을 제공하려는 것을 목표로 하고

있으며, 조정관점과 계산관점을 명확히 구분하는 구조로 설계하는 것을 또한 목표로 하고 있다.

그림 3에서 나타낸 바와 같이 m3-RTE는 세 계층으로 구성되어 있다. 조정계층은 MEADL(Mobile Enterprise Architecture Description Language)로 쓰여진 조정 스크립트를 수행하는 조정 관리자가 위치하는 부분이고, 전용관리자 계층은 유비쿼터스 환경의 세 가지 기초부분(fundamental part)을 이루는 것으로 컨텍스트 관리자(Context Manager: CM), 적응 관리자(Adaptation Manager: AM), 정책 관리자(Policy Manager:PM)가 존재한다. 분산 서비스 계층은 보안이나 통보와 같은 서비스들과 네트워크 프로토콜들을 포함한다.

구조기반 개발을 지원하기 위한 표기들을 모델링하는 수단으로써 구조명세언어를 제안하여 이용한다. 자율적인 콤포넌트들을 다시 기록하지 않고도 그러한 콤포넌트들의 이벤트들을 조정하기 위해 그들은 규칙과 이벤트기반의 조정 스크립트(MEADL)를 정의했다. 그들이 조정언어 MEADL을 정의한 주된 장점 중의 하나는 MEADL이 단순하고 역할기반이며, XML로 변환될 수 있다는 것을 강조하고 있다.

조정관리자는 응용으로부터의 요청(request)을 받아서 그 요청을 정책관리자에게 보내 권한체크를 하도록 한다. 만일 권한이 주어져 있으면, 그 요청은 적응관리자에게 보내진다. 그 응용의 컨텍스트에 관한 사항은 컨텍스트관리자에 의해 끊임없이(constantly) 갱신되어 적응관리자에게 보내진다. 적응관리자는 컨텍스트에 기반해 적응규칙을 선택하고, 적응행위가 수행된다. 응답(reply)이 조정관리자에게 즉각 보내지고, 조정관리자는 그 응답을 응용에 보낸다.



〔그림 3〕 M3-RTE 구조

Ⅲ. 유비쿼터스 미들웨어 보안 요구사항

유비쿼터스 컴퓨팅의 새로운 취약점을 다루기 위해 유비쿼터스 컴퓨팅 환경의 보안은 미들웨어의 추가사항으로 고려되는 것보다는 미들웨어의 설계단계에서부터 고려되어야한다. 하지만 기존의 미들웨어는 설계 시에 보안요구사항을 반영하지 않았거나, 반영하였다하더라도 하나의 보안 담당 컴포넌트를 끼워넣는 수준의 보안을 반영하고 있다. 이 장에서는 유비쿼터스 컴퓨팅환경을 위한 미들웨어의 보안 서브시스템을 위해 중요한 보안요구사항을 기술한다[9, 10, 11, 12].

첫째, 유비쿼터스 컴퓨팅 환경에서 사용자는 보안 시스템이 어떻게 동작하는지 신경 쓰지 않아도 되고, 보안 시스템에 의해 방해 받아서도 안된다. 유비쿼터스 컴퓨팅의 초점은 사용자가 더 이상 컴퓨터 기기에 신경을 쓰지 않아도 될 수 있도록 하는 것이다. 그 결과로 보안 시스템은 환경과 조화되어 사용자를 방해하지 않고 서비스를 제공해야 한다.

둘째, 보안 구조는 시스템 정책, 컨텍스트 정보, 환경 상황, 시간적 상황에 따라서 각각 다른 보안레벨을 제공할 수 있어야 한다. 이것이 가능할 수 있도록 다양한 인증도구마다 신뢰 값을 부여하고 필요에 따라 요구하는 인증의 강도를 달리하는 방법이 필요하다.

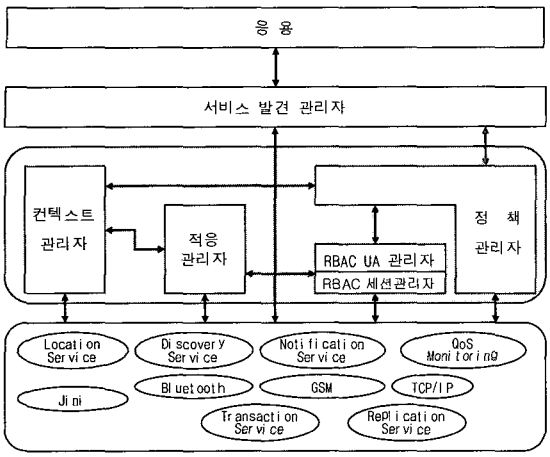
셋째, 컨텍스트와 결합하여 동적인 보안을 제공하여야 한다. 유비쿼터스 컴퓨팅은 컨텍스트와 상황정보를 통합하여, 컴퓨팅 환경으로 변환하게 된다. 보안도 예외는 아니다. 보안 서비스도 컨텍스트 정보의 광범위한 사용이 가능하게 만들어야 한다. 예를 들면, 접근통제 결정이 시간이나 특정한 환경에 의존하도록 하는 것이다. 컨텍스트 데이터는 침입 탐지 메커니즘을 위해 가치 있는 정보를 제공 가능해야 한다. 보안 정책들은 시간이나 상황에 따라 퍼미션을 제한하기 위해 동적으로 변화가 가능해야 한다. 어떤 보안 정책은 특정한 시간이나 특정한 상황 하에서는 가능하지 않도록 하는 방법도 요구된다.

넷째, 보안 시스템은 유연하고 적응가능하고, 특정 요구에 맞춤 가능해야 한다. 그것은 극단적인 상황과 부족한 자원이 있는 환경에도 적응 가능해야한다. 정책들을 정의하고 관리하기 위한 툴은 동적으로 이러한 사항을 적용하는 것이 가능해야 한다.

이러한 보안 요구사항은 미들웨어 설계시부터 반영되어야만 미들웨어의 다른 부분과 조화를 이루며 기능할 수 있고, 보안 기능을 미들웨어 전체에 미치도록 할 수 있다.

Ⅳ. 유비쿼터스 보안 미들웨어 구조

유비쿼터스 환경에서 응용과 서비스는 미들웨어에 항



〔그림 4〕 유비쿼터스 보안 미들웨어 구조

상 고정되어 있는 것이 아니라 동적으로 결합되어지고 동적으로 분리되어지므로 사용자가 실행한 응용이 신뢰할 수 있는지와 그 응용이 이용하려고 하는 서비스에 대해 접근 권한이 있는지는 중요하다. 기존의 유비쿼터스 미들웨어는 동적으로 변하는 컨텍스트에 대해 응용이 잘 적응할 수 있는 구조로는 되어있지만 응용이 이용하려는 서비스에 접근 권한이 있는지에 대해서는 조사하지 않기 때문에 서비스는 여러 위협에 대해 안전할 수 없다.

우리는 역할기반접근통제(Role-based Access Control: RBAC)를 적용해 서비스의 접근권한을 통제하고자 한다. 이 장은 현재 연구진행중인 RBAC 접근통제기능을 지닌 유비쿼터스 보안 미들웨어 구조를 간략하게 설명한다.

1. 응 용

유비쿼터스 응용은 사용자가 수행하는 작업(task)으로써의 의미를 갖는다. 그 작업을 수행하기 위해서는 응용에 요구되는 자원과 서비스가 필요하다. 하지만 응용 개발자는 요구되는 자원과 서비스들을 특정지어서는 안된다. 유비쿼터스 응용은 같은 기능을 하는 다양한 자원과 서비스를 이용할 수 있어야 하며, 응용의 개발 당시에는 존재하지 않았지만 응용이 실행될 때에 새롭게 나타나는 서비스도 이용할 수 있어야 한다. 그러기 위해서는 응용이 요구하는 자원과 서비스는 추상적으로 기술되어야 하고, 그 응용이 실행될 때 요구되는 자원과 서비스를 발견하여 이용할 수 있어야 한다.

예를 들어, 위치 서비스에는 GPS기반 위치 서비스, 웹 기반 위치 서비스 등의 다양한 실제 서비스의 구현들이 존재한다. 하지만 유비쿼터스 응용을 개발할 때에는 이렇게 특정적으로 서비스를 명시하면 동적으로 변하는 컨텍스트를 반영하지 못하고 결국 응용은 서비스를 지속적으로 이용할 수 없게 된다. 따라서 유비쿼터스 응용은 GPS기반 위치 서비스, 웹 기반 위치 서비스 처럼 필요한 서비스를 특정적으로 기술하는 대신 "위치 서비스"라고 추상적으로 기술하여야 한다.

2. 컨텍스트 관리자

컨텍스트는 "개체의 환경을 특징짓는데 사용될 수 있는 모든 정보, 여기서 개체는 사람이나 장소, 물리적 또는 계산 가능한 객체들일 수 있다"로 정의〔13〕되어진다.

컨텍스트 관리자는 컨텍스트를 인지하는데 그 목적이 있다. 컨텍스트 관리자가 컨텍스트를 인지한다는 것은 컨텍스트를 명세하고, 컨텍스트를 감지하고, 감지된 컨텍스트를 해석하고, 의미있는 컨텍스트를 결정하는 것을 포함한다. 여기에서 컨텍스트 결정에는 앞으로 발생할 수도 있는 변화를 미리 추론하거나, 서로 반하는 컨텍스트들 사이의 올바른 컨텍스트를 결정하는 것이 포함된다.

m3-RTE〔8〕에서 정의한 것처럼 컨텍스트 관리자는 자원서술구조(Resource Description Framework: RDF) 그래프를 사용하여 컨텍스트를 값/속성 쌍들로써 표현한다. RDF 인스턴스들은 값들을 인스턴스들에게 배정한다. 이러한 인스턴스들이 컨텍스트를 서술한다. RDF는 스키마들의 정의를 가능하게 해줄 뿐만 아니라 속성들간의 연관성 연산자들(relationship operators)의 사용(예를들면 카디널리티)을 허용해주며, 제약사항 특성(constraint properties)을 서술하도록 허용해준다.

RBAC 관리자와 적응관리자는 컨텍스트 관리자를 참조하므로써 컨텍스트 인식 접근통제와 컨텍스트 인식 적응을 할 수 있다.

3. 정책관리자

정책관리자는 정책을 일관성 있게 추가, 삭제, 수정 할 수 있다. 여기에서 정책은 커뮤니티의 목적과 관련된 특별한 목적을 지닌 규칙들의 집합이다. 규칙의 예로는 "커뮤니티에 속해 있는 역할이 정해진 시간 동안 행위 수행이 금지된다"의 금지, 또는 역할과 관련된 의무나 권한 등을 정할 수 있다. .

정책관리자는 이런 규칙들을 일관성 있게 유지할 책임을 지고 있고, 목적에 부합하는 중요도에 따라 적응 규칙에 우선순위를 부여할 수 있다. 정책관리자는 서비스 발견 관리자로부터 온 요청(request)을 처리하기 위해 먼

저 응용의 사용자나 응용의 역할배정의 유무를 검사하기 위해 RBAC UA 관리자에게 질의를 한다.

4. 서비스 발견 관리자

유비쿼터스 환경에서 자원과 서비스는 응용이 실행되는 지역에 있거나 아니면 물리적으로 분산된 환경 안에 존재하게 된다. 이런 환경에서 응용이 실행되기 위해서는 필요한 서비스를 찾는 메커니즘이 필요하다.

서비스발견 관리자는 응용에게 필요한 서비스를 동적으로 찾고 식별하기 위해 필요한 서비스 목록을 유지 관리하는 일과 서비스 발견 요청이 들어오면 이용가능한 서비스 리스트를 리턴해준다.

5. RBAC 관리자

RBAC 관리자는 보안미들웨어서 핵심적인 컴포넌트로 자원과 서비스에 대한 접근통제와 적응관리자와 함께 작업의 충돌을 해결하는 역할을 한다. 역할기반 접근 통제에서 가장 큰 특징은 서비스에 대한 연산을 수행할 수 있는 권한들은 사용자나 응용에 직접 할당되지 않고, 주어진 환경에서 정의된 역할에 대해서만 배정한다는 점이다. 따라서 사용자나 응용이 원하는 서비스에 대한 연산을 수행하기 위해서는 먼저 해당 서비스에 대한 연산을 실행할 수 있는 권한을 가진 역할의 소속원이 되어야 한다.

RBAC 관리자는 발견된 서비스들에 사용자가 접근할 수 있는 권한이 있는지를 결정하는 역할을 한다. 접근 권한 결정에는 현재의 컨텍스트와 정책을 고려하여 결정한다. 또한 RBAC 관리자는 RBAC의 임무분리를 이용하여 작업의 충돌도 해결할 수 있다. 임무분리는 본 연구의 범주를 넘는 향후 계획으로 복잡성을 지니고 있어 본 논문에서는 생략한다.

5.1. RBAC UA 관리자

사용자 배정, 권한 배정은 다대다 관계이며 역할 기반 접근통제 모델에서 매우 중요한 구성요소이다. 사용자가 서비스에 대해서 실행할 수 있는 연산들을 직접 사용자에게 부여하는 대신 작업 수행에 필요한 역할에 권한을 배정하고 사용자는 해당 역할에 구성원이 됨으로써 서비스에 대하여 원하는 연산을 수행할 수 있도록 하는 것이다.

RBAC UA 관리자는 서비스 발견 관리자가 제시한 특정적인 서비스에 대해서 응용을 실행 시킨 사용자가 그 서비스에 필요한 연산을 할 수 있는 역할에 배정되어 있는지를 검사한다.

5.2. RBAC 세션 관리자

우리가 제안한 유비쿼터스 보안 미들웨어에서 사용자와 응용은 다대다 관계이며, 응용과 세션은 일대다 관계를 이룬다. 전통적인 역할 기반 접근통제와 마찬가지로 각 세션은 반드시 한 사용자와만 관련을 맺지만 각 사용자는 여러 세션과 관련을 맺을 수 있다.

역할 기반 접근통제에서 세션은 한 사용자와 여러 개의 역할들로 구성된 집합으로 표현 될 수 있으며, 사용자는 세션을 통해서 자신에게 배정된 역할들 중의 일부 또는 전체를 수행할 수 있다.

전통적인 RBAC에서는 응용이 시작될 때 사용자의 선택에 의해 세션이 정해지고, 그 세션을 통해 사용자의 역할이 활성화된다. 그러나 유비쿼터스 환경에서는 적응관리자가 컨텍스트 관리자가 보내온 사용자의 환경 값을 이벤트로 받아 그 사용자의 상황에 적합한 세션이 선정되어야만 한다. 따라서 사용자의 직접적인 개입 없이 세션이 결정된다.

6. 적응 관리자

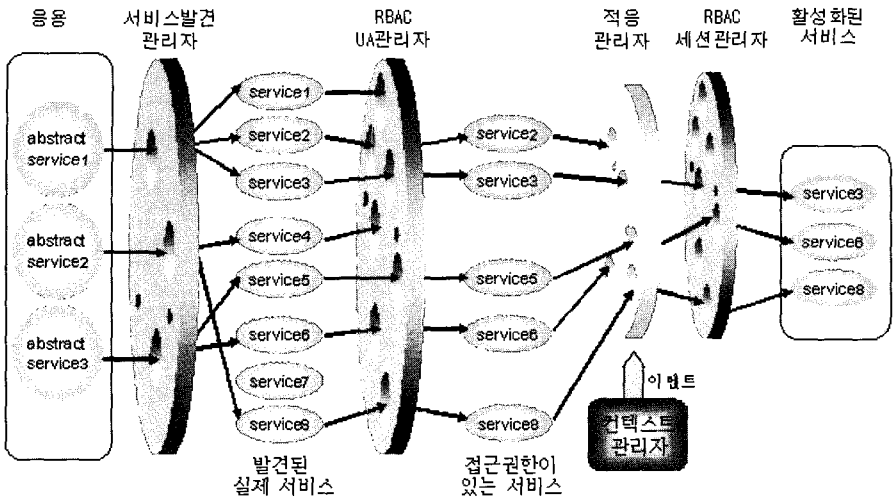
적응관리자는 컨텍스트 관리자를 참조하여 응용이나 응용의 사용자 컨텍스트의 변화된 상황에 따라 적절한 적응을 결정하고, 적응규칙을 수행할 책임을 지고 있다. 지금까지의 대부분의 연구는 컨텍스트 충돌에 대한 적응 대해서는 다룬 바가 없다. 우리는 적응 관리자가 RBAC 관리자와 협력하여 이런 충돌을 임무분리를 적용하여 해결하는 방법을 연구중에 있다.

V. RBAC을 이용한 접근통제 활용 예

유비쿼터스 컴퓨팅은 일상생활에서 실현될 수 있다. 본 장에서는 역할기반 접근통제 모델(RBAC)을 이용하여 어떻게 유비쿼터스 보안 미들웨어가 적응을 하는지를 설명한다.

그림 5는 유비쿼터스 응용이 요구하는 서비스에 대한 접근통제와 컨텍스트의 변화에 응용의 적응이 어떻게 이루어지는지 본 논문에서 제안한 유비쿼터스 보안 미들웨어 내의 관리자들 사이의 상호작용을 통해 보여주고 있다.

유비쿼터스 응용 개발자는 응용에 요구되는 서비스를 특정짓지 않고, 추상적으로 기술하여 그 응용이 실행되는 환경에 적응할 수 있도록 한다. 예를 들어, 응용에 GPS 기반 위치 서비스나 웹 기반 위치 서비스 등과 같이 특정적으로 서비스를 기술하는 대신 "위치 서비스"라고 추상적



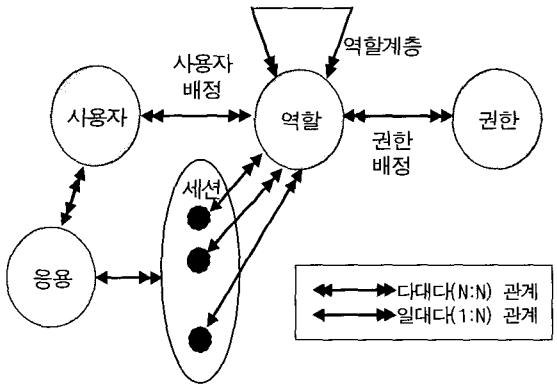
〔그림 5〕 미들웨어 내부 관리자 사이의 상호작용

으로 기술한다.

이렇게 응용에서 요구되어진 추상적으로 기술된 서비스들의 목록은 서비스 발견 관리자에게 보내진다. 위치 서비스라는 추상적인 서비스를 요구받게 된 서비스 발견 관리자는 GPS 기반 위치서비스나 웹 기반 위치 서비스가 로컬에서 이용할 수 있는지, 아니면 원격지 몇 홉의 거리에 있는지 등의 정보를 알아낸다. 다시 말하면, 추상적으로 기술된 서비스들에 대해 실제로 사용 가능한 서비스들을 찾아내고, 그에 대한 정보(로컬에 있는지 원격지에 있는지, 구체적으로 어떤 서비스인지 등)를 목록화 한다.

서비스 발견 관리자는 응용을 구동한 사용자가 이들 서비스들에 접근권한이 있는지를 검사하기 위해 RBAC UA 관리자에게 질의한다. 서비스에 대한 접근 권한은 모두 역할에 배정되어 있으므로 RBAC UA 관리자는 그 역할에 사용자가 배정되어 있는지를 조사하여 접근 권한을 결정하게 된다.

발견된 서비스가 접근 권한이 있으면 응용은 비로소 서비스를 이용할 수 있게 된다. 하지만 응용이 이용가능한 서비스가 두 개 이상일 수도 있고, 없을 수도 있다. 만약 이용 가능한 서비스가 없다면 적응 관리자는 컨텍스트 관리자를 참조하여 대체 가능한 서비스를 찾는 요구를 서비스 발견 관리자에게 하게 되고 위 과정을 다시 반복하게 된다. 만약 두 개 이상의 서비스를 이용할 수 있다면 적응 관리자는 컨텍스트 관리자를 참조하여 현재 응용에 적합한 최적의 서비스를 결정한다. 이렇게 사용자의 컨텍스트를 고려하여 결정된 이용 가능한 최적의 서비스가 결정되면 적응관리자는 RBAC 세션 관리자 통해 해당 세션을 열고 이들 이용가능한 서비스를 수행하는 데 필요한 역할을 활성화시켜 응용을 적응시킨다. 이후 시스템은 컨텍스트 관리자를 통하여 자원과 서비스의 변화를 감시하고, 자원과 서비스 변화에 적응을 지속적으로 수행한다.



〔그림 6〕 역할기반 접근통제 모델

이 과정을 RBAC 모델로 간략화하면 그림 6과 같다. 사용자가 응용을 실행하면 미들웨어는 응용 실행에 적당하고, 실행 시간에 이용할 수 있는 서비스들을 찾아낸다. RBAC UA 관리자는 서비스 발견 관리자가 제시한 특정적인 서비스에 대해서 응용을 실행 시킨 사용자가 연산을 할 수 있는 역할에 배정되어 있는지를 검사한다. 적응관리자는 컨텍스트 관리자를 참조하여 컨텍스트가 변한다면 적절한 적응을 결정하고, 적응 규칙을 수행하는데, 이때 RBAC 세션 관리자를 통해서 적응 관리자는 사용자 개입없이 사용자에 적합한 세션을 열게 된다

Ⅵ. 결 론

유비쿼터스 컴퓨팅의 응용들은 그 응용이 적재되고 실행되는 동안 많은 변화에 대한 요구사항을 만족해야 한다. 그래서 유비쿼터스 컴퓨팅 환경을 만드는 데 가장 중요한 목표 중의 하나는 동적으로 변화하는 유비쿼터스 컴퓨팅 자원과 서비스들을 수용하고 적응시키는 소프트웨어 시스템을 디자인하는 능력이다. 유비쿼터스 환경의 실현 핵심 기술과 더불어 유비쿼터스 환경에서의 보안에 관한 연구도 매우 중요한 분야로써 사용자 인증, 암호화, 사생활 보호와 같은 보안 분야에서 연구가 진행되고 있다. 이러한 연구들은 기존 컴퓨팅 환경에서의 연구가 무선 이동통신 환경 구조로 기반 구조만 바뀌었을 뿐이다. 유비쿼터스 환경에서 시시각각으로 변하는 상황 정보, 즉 컨텍스트 정보에 대해서는 고려하지 않고 있다.

본 연구에서는 유비쿼터스 보안 미들웨어의 구조와 그 기능을 설명하였다. 그리고 편재되어 있는 자원과 서비스를 사용하는 프로그램의 실행여부를 접근통제 관점에서 분석하고 현재 연구진행되고 있는 유비쿼터스 보안 미들웨어의 접근통제 기능에 대해 고찰하였다. 사용자 작업 충돌 문제 해결에서 임무분리와 트랜잭션 처리는 앞으로 해결해야 할 과제이다.

참 고 문 헌

[1] Mark Weiser, "Some Computer Science Problems in Ubiquitous Computing," Communications of the ACM, July 1993

[2] Mark Weiser, "Ubiquitous Computing," Nikkei Electronics, pp.137-143 , December 1993

[3] Mark Weiser , "The World is Not a Desktop," Interaction, pp.7-8 , January 1994

[4] Guruduth Banavar외 5, Challenges: An Application Model for Pervasive Computing, Mobile computing and networking: MobiCom 2000

[5] Joao Pedro Sousa, and David Garlan, "Aura: an Architectural Framework for User Mobility in Ubiquitous Computing Environments", Software Architecture: System Design, Development, and Maintenance (Proceedings of the 3rd Working IEEE/IFIP Conference on Software Architecture) Jan Bosch, Morven Gentleman, Christine Hofmeister, Juha Kuusela (Eds), Kluwer Academic Publishers, August 25-31, 2002. pp. 29-43.

[6] F. Kon, A. Singhai, R. H. Campbell, D. Carvalho, R. Moore, and F. J. Ballesteros, "2K: A Reflective, Component-Based Operating System for Rapidly Changing Environments," presented at ECOOP'98 Workshop on Reflective Object-Oriented Programming and Systems, Brussels, Belgium, 1998.

[7] M. Roman and R. H. Campbell, "GAIA: Enabling Active spaces," presented at 9th SIGOPS European Workshop, Kolding, Denmark, 2000.

[8] Andry Rakotonirainy, Jaga Indulska, Seng Wai Loke, and Arkady Zaslavsky, Middleware for Reactive Components An Integrated Use of Context Roles and Event Based Coordination. 2001.

[9] M. Langheinrich, "Privacy by Design - Principles of Privacy-Aware Ubiquitous Systems," presented at ACM UbiComp 2001, Atlanta, GA, 2001.

[10] M. Langheinrich, "A Privacy Awareness System for Ubiquitous Computing Environments," presented at 4th International Conference on Ubiquitous Computing, 2002.

[11] F. Stajano, "Security for Ubiquitous Computing" Halsted Press, 2002.

[12] Roy Campbell, Jalal Al-Muhtadi, Prasad Naldurg, Geetanjali Sampemane, M. Dennis Mickunas, "Towards Security and Privacy for Pervasive Computing," Mext-NSF-JSPS International Sympsoium, ISSS 2002, pp. 1-15, November 2002.

[13] Dey, A. and Abowd, G., "Towards a Better Understanding of Context and Context-Awareness", Workshop on the what, who, where, when and how of context-awareness at CHI 2000,April 2000.

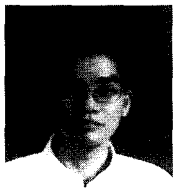
〈著 者 紹 介〉

박 희 만 (Hee-Man Park)
학생회원

2003년 2월 : 전남대학교 공과대학 전기공학과 졸업(공학사)
2004년 3월~현재 : 전남대학교 대학원 정보보호 협동과정(석사과정)
〈관심분야〉 컴퓨터와 네트워크 보안, 유비쿼터스 보안

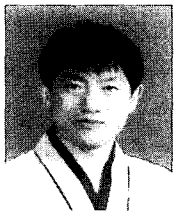
이 영 록 (Young-Lok Lee)
정회원

1986년 2월 : 전남대학교 계산통계학과 졸업(학사)
1990년 2월 : 전남대학교 전산통계학과 졸업(석사)
2003년 2월 : 전남대학교 전산학과 졸업(박사)
2003년 3월~현재 : 전남대학교 리눅스시스템연구센터 연구교수
〈관심분야〉 유비쿼터스 보안, 전자상거래 보안, 보안모델, 정보보호 시스템



이 형 효 (Hyung-Hyo Lee)
정회원

1987년 2월 : 전남대학교 계산통계학과 졸업(학사)
1989년 2월 : 한국과학기술원 전산학과 졸업(석사)
2000년 2월 : 전남대학교 대학원 전산학과 졸업(박사)
1990년 ~ 1997년 : 삼보컴퓨터 기술연구소, 한국통신 연구개발원
2001년 3월 ~ 현재 : 원광대학교 정보·전자상거래학부 조교수
〈관심분야〉 보안모델, 네트워크보안, 전자상거래보안



노 봉 남 (Bong-Nam Noh)
정회원

1978년 2월 : 전남대학교 수학교육과 졸업(학사)
1982년 2월 : KAIST 대학원 전산학과 졸업(석사)
1994년 2월 : 전북대학교 대학원 전산과 졸업(박사)
1983년 ~ 현재 전남대학교 컴퓨터정보학부 교수
2000년 ~ 리눅스 보안 연구센터 소장
〈관심분야〉컴퓨터와 네트워크 보안, 정보보호시스템, 전자상거래 보안, 사이버사회와 윤리