SecureJMoblet: 안전한 Jini 기반의 이동 에이전트 시스템

유양우° 문남두 이명준
울산과학대학 컴퓨터정보학부, 울산대학교 컴퓨터정보통신공학부
soft@mail.uc.ac.kr°, ndmoon@dreamwiz.com, mjlee@mail.ulsan.ac.kr

SecureJMoblet: Secure Jini-based Mobile Agent System

Yang-Woo Yu° Nam-Doo Moon Myung-Joon Lee
School of Computer Information, Ulsan College

요 약

Jini 구조의 네트웍 기능은 분산응용을 위하여 간단하면서도 유연한 네트웍 환경을 제공하고 있다. 이를 통하여 이동에이전트 시스템의 동적인 등록 및 위치 파악의 기능과 에이전트의 활동에 유용한 서비스들의 동적 제공이 용이하게 지원되어 이동에이전트 시스템 개발에 널리 사용되고 있다.

본 논문에서는 Jini 기반의 이동에이전트 시스템인 JMoblet 시스템을 썬 마이크로시스템에서 새롭게 제시한 Jini 2.0 보안모델을 적용하여 보안성이 강화된 안전한 JMoblet 시스템으로 확장하였다. 또한 에이전트 간 통신 시, 안전한 통신패러다임을 제시하였다.

1. 서 론

인터넷 사용이 보편화됨에 따라 인터넷을 통하여 정보를 제공하고 전자상거래 등의 편리한 서비스가 운영되고 있다. 또한 무선 인터넷 기반의 이동 단말기를 이용한 서비스가 점차적으로 증가하고 있다. 이에 따라 빠르게 변화하는 사용자의 요구를 만족하면서도 편리하고 유용한 서비스를 제공하기 위하여 효과적인 분산 기술 및 분산 구조가 제시되어 왔으며, 그러한 노력의 일환으로서 사용자를 대신하여 자발적으로 행동하는 에이전트를 다른 시스템으로 이동시켜 작업을 수행하는 이동에이전트 기술[1,2]과 네트워크 플러그 앤 워크(Network Plug and Play)라는 새로운 기능을 제공하는 Jini 기술[3]이 주목을 받고 있다. 더욱이 썬마이크로시스템사는 2003년 보안성이 강화된 새로운 버전의 Jini2.0을 발표하였다[1].

*이동에이전트*는 사용자가 정의한 이동 경로를 따라 다른 여러 이동에이전트 시스템들로 자발적으로 이동하여 자신의 작업을 수행하고 그 결과를 자신의 사용자에게 보고하는 프로그램이다[3]. 그리고 *이동에이전트 시스템*은 에이전트의 실행환경과 에이전트의 생성, 이동, 자원할당 그리고 자신이 생성한 이동에이전트의 상태를 제어하는 기본적인 기능들을 제공한다. 이동에이전트 기술은 많은 양의 데이터를 주고받는 서비스를 제공할 때 에이전트를 목적지 시스템으로 이동시켜 실행함으로써 네트워크 트래픽을 줄이고, 서버의 기본 기능을 변경하지 않고 사용자의 다양한 요구사항을 지원할 수 있다.

이전의 연구에서 Jini 기반의 이동에이전트 시스템인 *JMoblet 시스템*을 개발했었다[9]. JMoblet 시스템의 보안구조는 SSL(Secure Socket Layer) 패키지를 이용하여 상호인증과 객체무결성 등 네트웍 보안을 지원하고 있다. 하지만, JMoblet 시스템에서 중요한 기능을 담당하는 Jini 서비스 중 하나인 JavaSpace는 블랙보드(black board) 정책을 사용하기 때문에 누구든지 접근하여 객체를 저장하거나, 저장된 객체를 가져갈 수 있는 보안성이 취약한 구조로 설계되어 있다. JMoblet 시스템에서는 에이전트를 저장하고, 에이전트 간 통신 그리고, 에이전트 시스템의 상태를 임시 저장하는 공간으로 JavaSpace를 사용하고 있으며, 이러한 경우 JMoblet 시스템은 악의를 가진 에이전트 또는 시스템으로부터 공격을 당할 수 있는 결점을 갖고 있다[10].

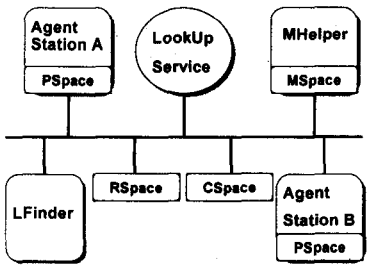
본 논문에서는 JMoblet 시스템의 이러한 단점을 개선하여 보안성이 강화된 Jini2.0 기반의 JMoblet 시스템을 확장하고자 한다. 먼저, JavaSpace를 접근하는 클라이언트는 상호인증과정을 거쳐 인증된 클라이언트만이 접근이 가능하도록 설계하였으며, JavaSpace 내의 객체 또한 송신자(sender)에서 규정한 수신자(receiver) 만이 접근권한을 갖도록 설계하였다.

본 논문의 구성은 다음과 같다. 2장에서는 Jini 기반의 이동 에이전트 시스템에 대하여 자세하게 설명하고 3장에서는 안전한 이동에이전트 시스템을 개발하기 위한 설계 방법을 제시하였으며, 마지막으로 4장에서는 결론 및 추후 연구 방향에 대하여 살펴볼 것이다.

2. Jini 기반의 이동 에이전트 시스템

JMoblet 시스템은 AgentStation, LFinder(Location Finder), MHelper(Moving Helper), RSpace(Resource Object Space), 그리고 CSpace(Communication Space)로 구성된다. (그림 1)은 JMoblet 시스템의 각 구성요소를 포함한

구조를 나타내고 있다.



(그림 1) JMoblet 시스템의 구조

JMoblet 시스템에서 이동에이전트의 속성은 에이전트 이름, 자신이 실행되는 AgentPlace, 에이전트 생성자, 에이전트 소유자, 에이전트 이동경로, 에이전트 자원등급, 현재 에이전트 위치정보이다. 이동에이전트는 Java의 직렬화(Serialization)를 통하여 네트워크를 통하여 전송될 수 있는 객체로 선언된다. 그리고 이동에이전트이름은 이동에이전트의 식별자로써 생성시간, 생성한 에이전트 시스템 이름, 작성자로 구성된다.

3. 안전한 이동에이전트 시스템의 설계

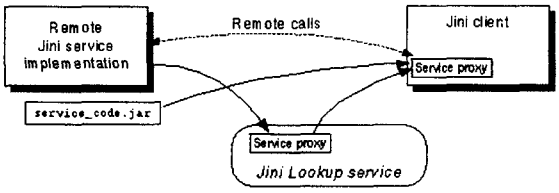
JMoblet 시스템은 보안기능을 제공하는 JavaSpace를 제공하지 못하여 보안성이 결여된 시스템으로 동작하도록 구현되었다.

본 논문에서는 이전 버전의 단점을 개선하기 위하여 보안성이 강화된 Jini2.0 기반의 JMoblet 시스템으로 확장할 것이다. 그 결과 안전한 JavaSpace 서비스를 제공할 수 있으며, 또한 JavaSpace를 이용한 에이전트 간 통신 시, 올바른 수신자를 판별하기 위한 독자적인 알고리즘을 제시할 것이다.

3.1 안전한 Jini 2.0 구조

Jini2.0의 보안모델은 3단계로 구성되어 있다. 첫째, 서비스프록시 객체가 신뢰할 만한지 아닌지를 결정해야 한다. 둘째, 다운로드된 프록시가 믿을만하다면, 프록시에 대한 제약(constraint)을 둬야한다. 마지막으로, 다운로드된 서비스프록시에게 동적으로 허가(permission)를 제공할 수 있어야 한다. 아래의 그림은 Jini 시스템에서 객체와 데이터를 이동하는 방법을 설명하고 있다. 먼저 Jini 서비스는 서비스에 대한 프록시를 룩업서비스에 등록시키고, 클라이언트는 등록된 서비스프록시를 검색하여 올바른 서비스프록시를 찾아 다운로딩한다. 클라이언트는 다운로드된 서비스프록시 정보를 이용하여 서비스를 제공받는다.

(그림 2)와 같이 동작을 수행 시, 서비스프록시는 클라이언트를 인증할 수 있고, 역으로 클라이언트가 서비스프록시에 제약을 두어 서버를 인증할 수 있다.



(그림 2) Jini2.0의 보안정책

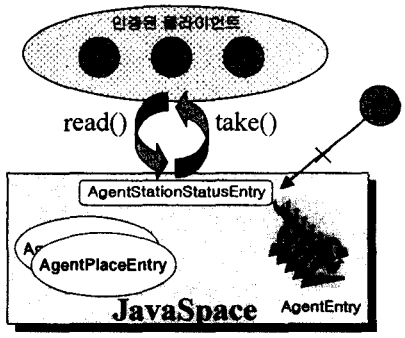
Jini 보안모델에서, Jini 서비스가 클라이언트를 인증하길 원한다면 서버프록시 내에 `ClientAuthentication.YES`와 같이 설정한다. 클라이언트가 서비스프록시를 제약하기 위하여 `net.jini.core.const.RemoteMethodControl` *인터페이스*를 구현하여야 하며, 제약은 `net.jini.core.constraint.InvocationConstraint` 인터페이스로 표현될 수 있다.

3.2 안전한 JMoblet 시스템

본 논문에서는 클라이언트와 Jini 서비스를 상호인증함으로서 인증된 클라이언트만이 JavaSpace를 사용할 수 있도록 새로운 설계방법을 제안할 것이다. 두 번째 제안은 JavaSpace를 이용한 에이전트간 통신 시, sender가 보낸 메시지를 수신자(receiver)만이 볼 수 있도록 보안성을 강화하였다.

3.2.1 JavaSpace의 클라이언트 인증

기존의 JMoblet 시스템에서, 이동에이전트 시스템은 시스템 다운이라는 최악의 상황에서도 이전의 상태로 복구하여 재 시작할 수 있는 기능을 PSpace를 통하여 제공하고 있으며, 에이전트간 통신은 CSpace에서 제공하고 있다. 두 서비스는 모두 JavaSpace를 이용하여 구현하였다. 이러한 경우 기존의 JMoblet 시스템에서는 JavaSpace내에 모든 정보를 네트웍상의 누구든지 가져가거나 읽어갈 수 있었다. 이를 해결하기 위하여 클라이언트 상호인증 하에 인증된 클라이언트만이 서비스 프록시를 가져갈 수 있도록 설계하였다.



(그림 3) 클라이언트 인증서비스

첫 번째, password를 이용한 key를 생성하여 keystore에 저장한다.

두 번째, 사용자와 서비스에 따른 principal을 생성하여 접근가능한 클라이언트를 생성한다.

세 번째, 위의 정보를 이용하여 .config 파일을 작성한다.

ssl-outrigger.config 파일의 작성은 다음과 같다.

```
com.sun.jini.outrigger {
    /* JAAS login */
    loginContext                    =                    new
LoginContext("com.sun.jini.Outrigger");

    /* 사용자 정보 */
    private static users =
        KeyStores.getKeyStore("file:lib/truststore",
                              null);

    private static clientUser = Collections.singleton(
        KeyStores.getX500Principal("client", users));
    private static outriggerUser =
        KeyStores.getX500Principal("outrigger", users));

    /* Exporters */
    private              serviceEndpoint              =
            SslServerEndpoint.getInstance(0);
    private serviceConstraints =
        new BasicMethodConstraints(
            new InvocationConstraints(
                new            InvocationConstraint[]{
                Integrity.YES }, null
            )
    );
        :
        :
}
```

3.2.2 올바른 수신자 에이전트 판별 방법

송신자 에이전트는 JavaSpace 내에 암호화된 entry를 저장할 때 다음과 같이 저장한다.

write(α, encryption(entry));

α: receiver 에이전트의 id를 receiver 에이전트의 공용키로 암호화시킨 값.

다음은 메시지를 수신하는 수신자 에이전트가 올바른 수신자인지를 확인하는 방법이다.

read(β, encryption(entry));

β: 자신의 id를 receiver 에이전트의 개인키로 암호화시킨 값.

수신자는 β를 공용키로 해독하여 에이전트의 id를 구하고, 이를 다시 자신의 공용키로 암호화 시켜서 α와 값이 같음을 증명한다. 이로서 값이 같으면 올바른 수신자임을 증명하는 것이고, 값이 다르면 올바른 수신자가 아님이 판명되는 것이다.

이러한 설계방법을 이용하여 더욱더 안정된 JMoblet을 구현할 수 있도록 설계 구현하였다.

4. 결론 및 추후연구

JavaSpace의 프로그래밍 모델은 누구든지 객체를 저장하고, 원하는 객체를 검색하여 그 객체를 읽거나 가져갈 수 있도록 설계되어있다. 이러한 블랙보드(blackboard) 패러다임은 객체를 공유하는 차원에서는 유용한 서비스로 사용되지만, 보안을 요구하는 서비스에서는 매우 적절하지 않다.

본 논문에서는 보안기능이 취약한 JavaSpace를 상호인증 하에 정보를 저장하고 저장된 정보를 검색하여 가져갈 수 있는 안전한 JavaSpace를 설계하여 안전한 JMoblet을 구현하였다. 또한 에이전트간 통신 시 올바른 수신자가 메시지를 접근할 수 있는 새로운 정책을 제시하였다.

추후 연구과제는 안전한 JMoblet 이동 에이전트시스템을 이용한 애플리케이션으로 경매시스템을 설계하고 있으며 이를 이용하여 JMoblet 시스템의 유용성을 검증할 것이다.

[참고문헌]

[1] Frank Sommers, "Jini Starter Kit 2.0 tightens Jini's security framework", 2003.5.

[2] Jan Newmarch, "Jan Newmarch's Guide to Jini Technologies", 2003.7.

[3] Sun Microsystems Inc., "Jini Technology Core Platform Specification", 2000.10.

[4] Jan Newmarch, "Jan Newmarch's Guide to JINI Technologies," http : //jan.netcomp.monash.edu.au /java/jini/tutorial/Jini.xml.

[5] Forge Information Technology, "Protekt Encryption 3.0 Programming Guide," 1999

[6] Scott Oaks and Henry Wong, "Jini in a Nutshell", Oreilly Press, March 2000.

[7] Alfonso Fuggetta, Gian Pietro Picco, Giovanni Vigna, "Understanding Code Mobility," IEEE Transaction On S/W Engineering, Vol.24, No.5, May, 1998.

[8] 구형서, 윤형석, 김진홍, 유양우, 문남두, 이명준, "Jini 기반의 이동 에이전트 시스템" 한국정보과학회 가을 학술발표논문집 Vol.28, No.1, 2001.

[9] 김진홍, 구형서, 윤형석, 안건태, 유양우, 이명준, "JMoblet: Jini 기반의 이동에이전트 시스템.", 한국정보처리학회논문지 B 제8-B권 제6호, 2001.12.

[10] Sun Microsystems, Inc. JavaSpacesTM Service Specification, Version 1.1, October 2000.