情報保護學會論文誌
第11券 第3號, 2001. 6

VPN을 이용한 이동 에이전트의 보호

박 재 경*, 원 유 헌**

Protecting Mobile Agent with VPN

Jae-Kyoung Park*, Yoo-Hun Won**

요 약

인터넷 환경 속에서 사용자 또는 터미널의 이동성은 분산환경에서 지속적인 관심과 연구의 대상이 되어왔다. 사용자의 동기적인(Synchronous) 접근은 시간적으로나 네트워크의 효율적인 측면에서 많은 제약이 발견되었고 이를 해결하고자 하는 대안으로 연구된 분야가 이동 에이전트(Mobile Agent) 분야이다. 이동 에이전트는 사용자가 정한 여행리스트에 따라 자율적으로 이동을 하며 사용자의 요구사항을 대신하는 대리인의 역할을 한다. 이러한 이동 에이전트를 통해 분산환경 속에서 여러 가지 이득을 얻을 수 있게 되었다. 하지만, 이동 에이전트는 보안상 많은 취약점을 가지고 있다. 특히 이동 에이전트에 대한 불법적인 호스트의 공격은 이동 에이전트가 상업적으로 널리 이용될 수 있는 가장 큰 걸림돌이 되고 있다. 이에 따른 이동 에이전트 자체에 대한 보안으로 많은 연구가 진행되고 있다. 이 논문에서는 이동 에이전트가 가지고 있는 이동성에 따른 위험성을 제거하고 코드의 중요도에 따라 코드를 분할 및 재생성하여 이동 에이전트 자체의 보안을 유지하고자 한다. 또한 이러한 이동 에이전트의 분할 및 재생성을 수행하는 역할을 가상 사설망(VPN-Virtual Private Network)을 이용하고 이를 이동 에이전트 게이트웨이(MAG-Mobile Agent Gateway)로 설계하였다.

ABSTRACT

In the course of Internet proliferation, many network-related technologies are examined for possible growth and evolution. The use of Internet-based technologies in private networks has further fuelled the demand for network-based applications. The most promising among the new paradigms is the use of mobile agents. The mobile agent is capable of migrating autonomously from node to node in the network, to perform some computations on behalf of the user. The mobile agent paradigm is attractive alternative to traditional client-server programming for a significant class of network-centric applications. It does however, suffer. from a major drawback, namely, the potential for malicious attacks, abuse of resources, pilfering of information, and other security issues. These issues are significantly hampering the acceptance of the mobile-agent paradigm. This paper describes the design of a secure mobile agent gateway that can split and merge the agent code with security policy database on the VPN. This mechanism will promote security in the mobile agent systems.

keyword : *Mobile Agent, VPN, IPSEC, IKE*

Ⅰ. 서 론

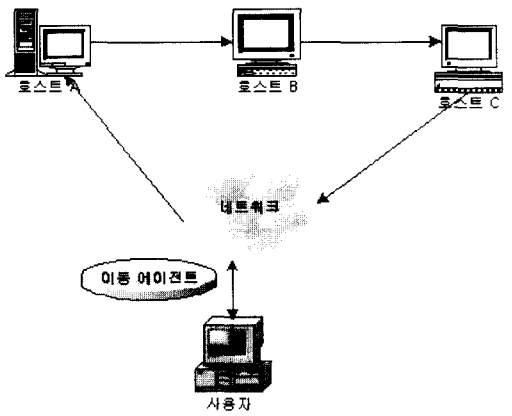
1.1 이동 에이전트

인터넷 환경 속에서 네트워크와 관련된 많은 기술들이 연구되어 왔다. 또한 중앙 집중적인 환경은 점차 분산 환경으로 대체되었고 이러한 분산환경은 많은 응용 프로그램들에 대한 발판이 되었다. 분산 환경 하에서 세 가지 주요한 기술로 다음과 같은 것들을 들 수 있다[1-3]. 첫 번째는 메시지 전달 시스템

* (주)퓨처시스템 정보통신 연구소(jkpark@future.co.kr)
** 홍익대학교 컴퓨터 공학과 교수(won@cs.hongik.ac.kr

(Message Passing System)이고 두 번째는 원격 프로시저 호출(Remote Procedure Call)이고 마지막은 분산 객체 시스템(Distributed Object System)이다. 메시지 전달 시스템은 모든 네트워크의 핵심적인 기술이다. 네트워크의 클라이언트들은 간단한 메시지를 주고 받으며 통신을 수행한다. 대표적으로 FTP나 Web과 같은 응용프로그램을 들 수 있다. 하지만, 이러한 메시지 전달 기법은 클라이언트들에게 프로토콜상 제약점을 가지고 있다. 즉, 미리 약속된 명령어들에 대해서만 적용할 수 있기 때문에 일반적이고 포괄적인 내용들을 적용하기가 어렵다는 단점을 지닌다. 이러한 메시지 전달 시스템의 단점을 보완하기 위하여 RPC가 연구되었다. RPC는 원격지에 프로그래밍 인터페이스를 통해 접근을 시도한다. RPC는 해당 컴퓨터에서 제공하는 함수를 호출하는 방식을 통해 해당 컴퓨터의 프로그램과 통신을 수행한다. 분산객체도 RPC와 비슷한 형태로 동작한다. 대표적인 경우로 OMG CORBA를 들 수 있다. 위의 패러다임을 이용하여 클라이언트는 원격지의 함수나 객체를 참조할 수 있다. 하지만 여전히 클라이언트가 적용할 주문형태의 기능을 이용할 수는 없었다. 따라서 관심의 대상은 위와 같은 기능들을 클라이언트들이 요구하는 형태로 적용할 수 있는 방법을 연구하게 되었고 이것이 이동 에이전트 출현의 계기가 되었다[1].

이동 에이전트는 다음 두 가지 요소의 소개로 인해 더욱 발전을 하게 되었다. 첫 번째로 클라이언트의 요구를 수용할 수 있는 기능이고 다른 하나는 프로그램 내부에 소프트웨어 모듈들을 집합화하여 구성한 것이다. 이것은 위에서 언급한 기법과는 여러 가지 측면에서 다르다. 이동 에이전트는 스크립트



〔그림 1〕 이동 에이전트

언어로 쓰여진 프로그램이다. 이는 서로 다른 분산 환경에서 사용자의 역할을 대신하여 특정한 즉, 사용자가 요구하는 작업을 수행 할 수 있다는 것이다. 이동 에이전트는 우선 사용자의 기계에 위치한 후 수행을 위해 원격지로 보내어지게 된다. 호의적인 호스트들은 이동 에이전트가 수행하기 위한 적절한 환경을 제공한다. 이동 에이전트는 이동 후 수행되어 정보를 수집하고 다음 여행지로 이동하기 위하여 자체적으로 상태와 변수들을 실시간적으로 수정한다. 이러한 여행을 마지막 호스트까지 수행한 후 자신의 홈으로 귀환한다. 그림 1은 이러한 이동에이전트의 동작을 설명하고 있다[4~7].

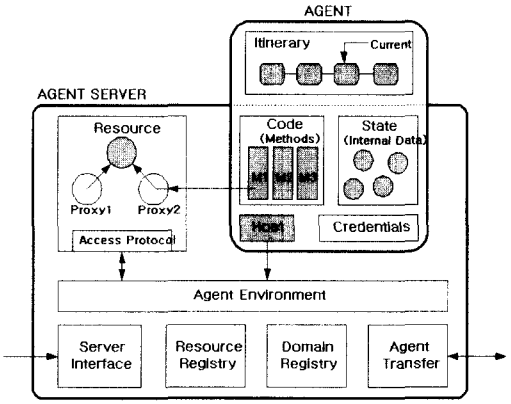
이동 에이전트의 특징으로는 다음과 같은 특성들을 들 수 있다.

- 자율성 : 독립적으로 사용자를 대신하여 기능을 수행- 적응성 : 현재 상황과 과거의 경험을 토대로 지식을 습득
- 이동성 : 여러 호스트들을 이동 가능

첫 번째로 이동 에이전트의 특징중의 하나는 상호적이 통신이 아닌 독립적으로 수행할 수 있다는 능력이다. 지금까지 분산환경에서의 작업 수행은 두 개체 사이의 상호작용을 통하여 이루어졌다. 하지만 이와는 달리 이동 에이전트는 한 번의 파견과 한 번의 귀환만을 통해 다수의 네트워크 작용을 감소시켰다. 이러한 점은 대역폭이 낮고 높은 지체를 일으키는 고비용 환경에서 특히 유용한 기능이다. 두 번째로 이동 에이전트는 서버 소프트웨어에 의해 제공되는 기능을 확장할 수 있으므로 인해 클라이언트들에게 보다 폭넓은 기능 및 환경을 제공할 수 있다. 기존의 환경에서는 클라이언트가 새로운 서비스를 원할 경우 이에 해당하는 서비스가 서버에 먼저 설치되어야만 했다. 하지만 이동 에이전트를 이용할 경우 필요한 서비스를 이동시에 서버에 가상적으로 설치하여 해당 서비스를 이용할 수 있다. 예를 들어 클라이언트가 암호연산(RSA)을 필요로 하는 모듈을 서버에서 수행하고 싶을 경우 서버는 클라이언트가 필요로 하는 암호 모듈이 설치되어 있어야만 서비스를 할 수 있다. 하지만 이동 에이전트는 이러한 모듈을 코드내부에 함께 적재하여 서버로 이동하고 서버는 이를 수행할 수 있는 환경만을 제공하면 된다. 이는 사용자의 요구를 충족시키는 중요한 장점이 될 수 있다. 그림 2는 이동 에이전트 시스템의 일반적인 구조를 나타내고 있다.

1.2 이동 에이전트 시스템의 보안

연구 초기에 이동 에이전트 패러다임은 획기적인 것처럼 생각되어 졌으나 완벽하지는 못했고 고려해야할 여러 가지 문제가 있었다[3~7~8]. 예를 들어 네트워크를 이동하는 이동 에이전트에 대한 신뢰성과 보안성에 대한 관심이 일어났다. 즉, 이동 에이전트가 어떠한 유해한 방해나 공격없이 수행할 수 있을 것인가 하는 의문이 제기되었다. 이러한 문제는 아직 풀리지 않은 여러 가지 문제들이 있고 많은 연구가 진행되고 있다. 다음절에서 이러한 문제점을 좀 더 상세히 살펴보겠다.



〔그림 2〕 이동 에이전트 시스템 구성

1.2.1 호스트에 대한 보안

대부분의 호스트는 발원지를 알 수 없는 소프트웨어를 실행할 만큼 충분이 안전하지 못하다[2~5]. 마찬가지로 근원지를 알 수 없는 이동 에이전트에도 동일한 문제가 적용된다. 어떤 컴퓨터 프로그램은 다른 사람의 컴퓨터를 침입하여 데이터를 파괴하려고 시도한다. 트로이 목마나 바이러스 등등의 위협을 초래하는 유해한 프로그램들이 많이 퍼져있다. 이동 에이전트 시스템에서 이동 에이전트들은 네트워크의 호스트들을 옮겨다니며 사용자를 대신하여 작업을 수행한다. 이러한 작업들이 적절하게 수행될 경우 피해를 주지 않지만 일반적인 경우는 그렇지 않다. 예를들어 해커들은 이동 에이전트를 이용하여 호스트를 침입할 수 있고 또한 보안상 취약점을 공격할 것이다. 이러한 문제를 해결하기 위하여 여러 연구들이 이루어졌고 많은 문제가 해결되고 있다.

1.2.2 이동 에이전트에 대한 보안

이동 에이전트를 수행하는 이동 에이전트 시스템 뿐만아니라 이동 에이전트 자체도 위험에 노출되어 있다[3~7~9]. 호스트에 의해 제공되는 수행환경을 통해 호스트는 이동 에이전트를 완벽하게 제어할 수 있다. 즉, 코드를 분석하거나 또는 상태 등의 동적 코드나 정적코드를 모두 변경할 수도 있다. 예를 들어 호스트는 이동 에이전트의 수행을 지연할 수 있고 변경된 내용으로 수행할 수도 있다. 따라서, 이동 에이전트의 정적, 동적 코드는 모두 호스트에게 드러나게 된다.

1.3 개선 방안

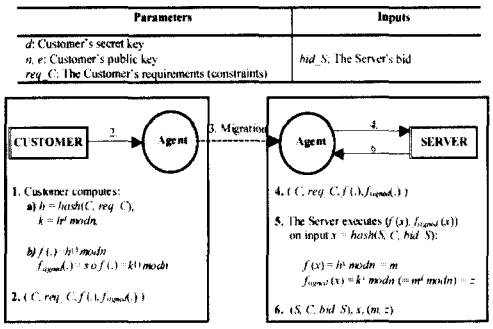
이동 에이전트 시스템 및 이동 에이전트에서 발생할 수 있는 여러 가지 보안상 문제점이 여러 연구를 통해 개선 또는 개발되고 있다. 그러나, 이동 에이전트 시스템에 대한 연구는 활발하게 진행되었으나 유해한 호스트로부터 이동 에이전트를 보호하는 연구는 그렇지 못한 실정이다. 이 논문에서는 이동 에이전트의 보안적 취약점을 점검하고 그에 대응되는 방안을 제시하고자 한다. 무엇보다도 이동 에이전트의 코드 변경에 따른 사용자의 피해를 최소화하고 나아가 민감한 데이터를 안전하게 처리할 수 있는 방안을 제시하고자 한다.

II. 관련연구

지금까지 이동 에이전트 보안에 대하여 많은 연구가 이루어 졌으나 본 논문에서는 이동 에이전트의 코드를 암호화하여 처리하는 연구에 대해 살펴보겠다.

2.1 분리되지 않는 서명(Undetachable Signatures)

분리되지 않는 서명 방법은 Sander와 Tschudin에 의해 제안되었다[5~8]. 그리고, 이는 CEF (Computing with Encrypted Function)라 불리는 방법에 기초를 두고 있다. 여기서 이동 에이전트는 s를 자신의 서명 함수로 대동하며 호스트는 이 함수를 통해 호스트의 결정을 서명하게 된다. 그러나 이 서명 함수는 호스트에게 드러나게 되므로 이를 암호화된 함수 f를 통해 암호화하여 s를 참조하는 대신 $f \circ s$를 이용하여 일을 처리하게 한다. 예를 들어 소비자는 인터넷을 통하여 물건을 구매하기 위하여 이동 에이전트를 보내고자 한다고 가정하자. 이때



〔그림 3〕 RSA를 이용한 안전한 서명 스키마

소비자의 서명 함수 s를 사용할 수 있을 경우만 에이전트는 트랜젝션을 인증한다. 그러나, 에이전트는 잠정적으로 유해한 호스트에서 수행될 수 있다. 따라서 이 s를 보호하기 위하여 다음과 같은 암호화 함수를 이용하여 호스트는 s를 직접적으로 알 수 없게 한다.

$$f_{signed} = s \circ f \qquad (1)$$

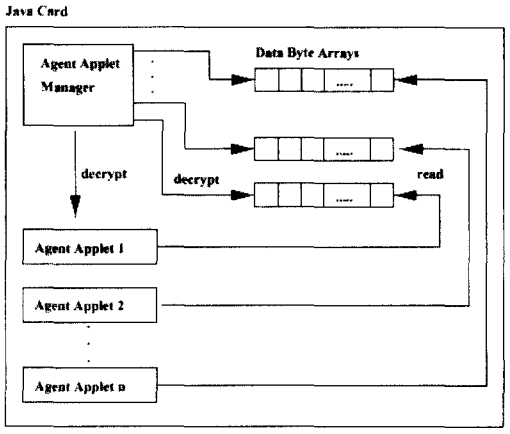
이러한 암호학적인 방법을 통해 에이전트의 보안성을 개선시켰다. 그림 3은 RSA 알고리즘을 이용한 전체적인 구조를 나타내고 있다.

그러나, 이러한 암호화된 서명을 통한 방법은 이동 에이전트의 여정리스트를 변경하거나 또는 암호 모듈을 전체적으로 다른 암호 모듈로 대체하였을 경우의 단점이 드러나게 된다. 또한 모든 데이터를 암호화할 수 없으므로 인해 보안상의 문제가 여전히 남아 있게 된다. 본 논문에서는 이러한 문제를 해결하기 위해 호스트의 결정을 에이전트 자체에서 결정하지 않고 보다 안전한 이동 에이전트 게이트웨이를 통해 수행하여 해결하였다.

2.2 스마트 카드를 이용한 에이전트 보호

에이전트의 보호를 위하여 스마트카드를 이용하는 방법이 제안되었다[10]. 이 방법은 에이전트의 보호를 위해 에이전트에 대한 모든 참조 및 연산 등을 컴퓨터상에서 수행하지 않고 스마트카드 내부에서 수행한다. 예를 들어 자바카드와 같은 신뢰할 수 있는 카드를 이용하여 다음과 같은 기능에 따라 수행된다.

- 암호화된 코드 부분을 입력 받는다.
- 카드 내에서 데이터를 복호화하고 수행되는 경로상에 데이터를 저장한 후 수행한다.
- 다음 이동될 카드의 키를 이용하여 결과를 암호화한 후 에이전트에게 전송한다.



〔그림 4〕 스마트카드의 구조

그림 4는 이러한 스마트 카드 내부의 구조를 나타내고 있다. 스마트카드를 이용할 경우 외부로 복호화된 데이터가 들어나지 않으므로 신뢰성있는 에이전트 수행을 기대할 수 있다. 또한 다른 하드웨어 장비에 비해 저렴한 가격과 보편화되어 있다는 장점을 가진다.

그러나, 이러한 스마트 카드를 이용하는 방법은 다른 스마트카드간의 상호적인 키를 알아야 하므로 중앙집중적인 방법을 통해 관리를 해야하고 또한 모든 관계된 키를 추가 변경이 생길때마다 갱신해야 하는 번거로움이 있다. 또한 하드웨어의 컴퓨팅 능력이 다소 떨어지므로 고속연산이나 큰 데이터를 처리하는데는 다소 문제가 있다고 할 수 있다. 이 논문에서는 이러한 한계로 인해 웹상에서 설계를 하여 일반적인 자바카드 형태로 이동 에이전트를 처리하는 실험을 하였다.

위의 두 관련연구에서 드러난 것처럼 암호화 모듈을 함께 전송하는 방법이나 또는 스마트 카드와 같은 하드웨어를 사용하는 방법에서 단점으로 들어난 사항들이 있다. 즉, 암호화 모듈의 강제적인 교체나 여정리스트의 변경 등의 대책이 없었고 하드웨어를 사용하더라도 다른 호스틀과의 연계성이라든지 관리적인 문제가 해결되지 않았다. 따라서 이 논문에서는 이러한 문제점을 VPN을 통해서 해결하고자 한다. VPN은 자체적으로 방화벽과 같은 기능을 가지므로 암호화 모듈을 변경한다든지의 외부 공격으로

부터 다소 안전하고 또한 에이전트를 분석하여 여정 리스트를 수정하여 일대일의 암호통신을 수행하므로 위의 단점을 보완할 수 있다. 다음 장에서는 이동 에이전트 게이트웨이에 대해 논하겠다.

Ⅲ. 이동 에이전트 게이트웨이(MAG-Mobile Agent Gateway)

이동 에이전트 보안에 대해 제시된 이론들은 이동 에이전트에 대한 공격을 줄일 수는 있으나 완벽하게 보안을 달성할 수 있는 방안은 아니다. 또한 코드상으로 많은 오버헤드를 감수해야만 한다. 현실 세계에서 다수의 사용자가 호스트를 이용할 경우 호스트에 과중한 부하가 생길 수도 있다. 이 논문에서는 이러한 단점들을 보완하기 위하여 이동 에이전트와 이동 에이전트 시스템 이외에 추가적인 개체로 이동 에이전트 게이트웨이를 제안하고자 한다.

3.1 이동 에이전트 게이트웨이

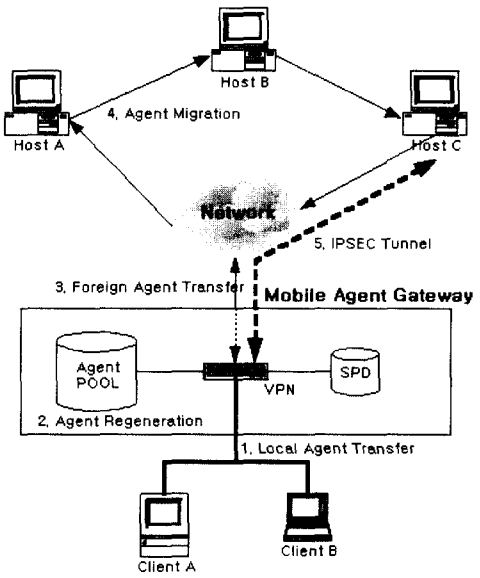
이동 에이전트 게이트웨이는 현재 실제 네트워크 상에서 많이 적용되는 VPN(Virtual Private Network) 장비를 이용하여 구축하고자 한다. 그림 4는 MAG의 전체적인 시스템의 구성도를 나타내고 있다.

3.2 가정

이동에이전트 게이트에 대한 다음과 같은 가정을 둔다.

a. 모든 클라이언트들의 통신은 VPN 장비를 통해서 외부와 연결된다.
b. MAG는 이동 에이전트를 모듈 또는 코드 형태로 인식할 수 있는 인터프리터 기능을 내장하고 있다.
c. MAG는 이동 에이전트들을 임시적으로 저장할 수 있는 이동 에이전트 풀을 운영하고 이는 VPN의 기능으로 보호된다.
d. MAG는 새로운 이동 에이전트를 생성할 수 있으며 보안 정책에 따라 적용 받는다.

보안 정책은 다음과 같은 구성 요소로 구성되어진다.
- 각 클라이언트에서 적용할 이동 에이전트의 보안 클래스



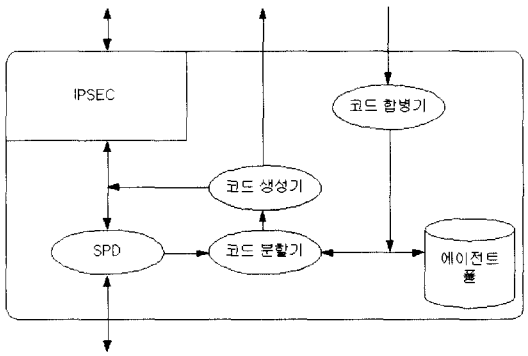
〔그림 5〕 이동 에이전트 게이트웨이

- 각 클라이언트이 방문한 호스트들의 리스트 및 위험성 여부
- MAG에서 테스트 이동 에이전트를 통해서 등록한 호스트들의 리스트 및 위험성 여부
- 각 보안 클래스에 따른 보안설정 데이터

위의 데이터는 클라이언트들의 설치 시점에 준비되고 운영 시점에 점진적으로 추가 및 갱신된다.

3.3 이동 에이전트 구조상의 문제점 해결

이동 에이전트는 이동 에이전트 시스템에서 수행하는 동안 코드를 통해 호스트의 자원을 참조한다. 이동 에이전트는 목표한 결과를 얻었을 경우 내부적으로 상태를 변경시킨 후 여정리스트에 따라 다음 목적지로 이동한다. 이때 ATP(Agent Transfer Protocol)를 통해 다음 호스트로 전송된다. 이러한 환경 속에서 만약 호스트가 불법적으로 상태를 변경하거나 여행 목적지를 강제로 변경할 경우가 발생할 수 있다. 또한 내부 데이터 중에서 민감한 데이터가 포함되어 있을 경우 문제는 더욱 심각해진다. 예를 들어 항공기의 예약을 위한 이동 에이전트가 각 항공사를 항해한다고 생각해보자. 이때 이동 에이전트는 내부적인 코드에 예약일자와 적정가격 및 신용카드 등의 중요한 정보를 포함하고 있을 것이다. 또한 여정리스트에는 각 항공사의 주소가 리스트로 연결



〔그림 6〕 VPN의 내부구조

되어 있을 것이다.

각 항공사 A, B, C에 예약 가능한 잔여 좌석이 있고 운임료는 각각 50,000원, 47,000원, 43,000원으로 책정되어 있다고 가정하자. 이때 이동 에이전트는 여행 리스트의 순서에 따라 A, B, C 각 항공사의 호스트를 방문할 것이고 정상적인 경우 C항공사의 비행기를 예약하여 홈으로 귀환할 것이다. 그러나, 과연 그러할 것인가? A항공사의 호스트는 이동 에이전트의 코드를 분석하여 내부 데이터 즉, 상태값을 변경하여 A항공사로 예약을 체결하고 여정 리스트의 내용과 무관하게 홈으로 이동 에이전트를 강제적으로 귀환시킬 수 있다. 이러한 공격을 방어하기 위해 제안된 많은 연구가 있지만 이 논문에서는 이동 에이전트 게이트를 이용하여 결정의 시점을 최종 여정이 끝난 뒤로 미루고 또한 여정리스트를 분할하여 다중 이동에이전트로 분산하여 목적을 달성할 것이다. 그림 5는 VPN의 내부구조를 나타내고 있다.

3.4 에이전트 코드 분할 및 재생성

이동 에이전트는 사용자의 요구를 대신하여 작업을 수행하는 자율적인 프로그램으로 요구에 부합되는 결과를 얻었을 경우 특정한 행동을 취할 수 있다. 예를 들어 예약과 관련되어 신용카드의 번호를 호스트에게 제공한다든지 등의 작업을 수행할 것이다. 이 데이터는 호스트의 공개키로 암호화되어 해당 비밀키를 소유한 호스트만 복호화 할 수 있다. 그러나, 이 과정에서 호스트는 이러한 민감한 정보를 악용할 우려가 있다. 실제 예약을 수행할 수 없는 조건을 가진 호스트도 이 정보를 알 수 있다는 것이다. 따라서 얼마든지 이러한 정보는 유출가능하고 악용될 우려가 있다. 이러한 문제를 해결하기 위해서 이 논문에서는 민감한 정보가 포함되었을 경우 MAG에서 이를 보관한 후 최종적으로 결정된 경우에만 이를 코드에 적용하여 실제 작업을 수행하게 한다. 그리고, 이동 에이전트를 생성하는 과정에서 코드에 대한 민감성 여부를 판단하기 위해 코드를 다음과 같이 분류한다. 이와 같은 분류는 관리자 및 사용자의 설정에 따라 조절 가능하도록 이동 에이전트 생성 프로그램 및 이동 에이전트 게이트웨이에서 사용자 인터페이스로 제공한다.

- 신상 정보 코드(Personal Data)
- 보호 코드(Sensitive Data)
- 일반 코드(General Data)

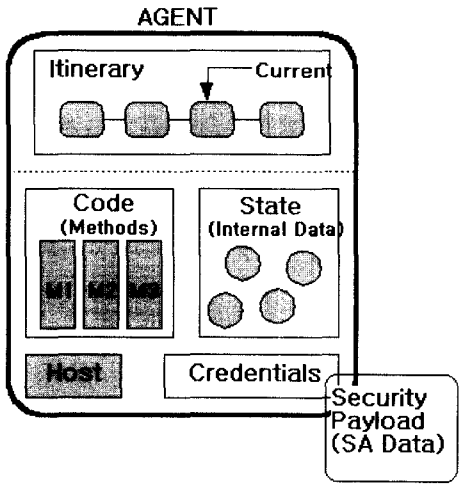
MAG는 클라이언트로부터 이동 에이전트를 수신한 후 분석기를 통해 원본 코드를 분류한다. 구분된 코드는 보안정책상에 클라이언트의 정책에 따라 재구성된다. 이러한 정책은 관리자에 의해 설정 가능한 정보로 유지한다. 이 과정을 통해 보안정책상에 외부로 유출을 금지한 코드를 포함하였을 경우 코드상에서 제외되고 안전한 코드만을 이용하여 이동 에이전트는 재구성된다. 원본 코드는 마지막 결정을 위해 에이전트 풀에 위치시킨다. 또한 코드를 재구성할 때 여정리스트를 보안정책 리스트에서 검사하여 위험한 호스트로 등록이 되어있을 경우 이를 분할한다. 즉, 위험 가능성이 있는 호스트일 경우는 다른 호스트와의 연계성을 배제함으로 인해 보다 안전한 통신을 하기 위함이다. 이러한 불법적인 호스트의 등록은 테스트 이동 에이전트를 통해 점진적으로 구축해 나간다. 이렇게 재구성되고 분할된 이동 에이전트는 외부 호스트로 이동되어 작업을 수행한다. 다음은 에이전트 코드 분할 및 재생성 알고리즘이다. 이러한 코드는 JAVA를 이용하여 설계하였다.

```
SplitAgent()
{
  Recieve_Agent_Data(pData); // VPN에서
        데이터를 // 처리하여 에이전트 코드를 전송
  pList = Get_Itinerary_List(pData);
  Get_SPD_Data(pSpdData); // SPD 데이터 적재
  for (int I = 0; I < pList->nCount; I++) {
    if (IsValidSplit_Agent(pList[i], pSpdData))
      pNewAgent[] = Split_Agent(pList[i]);
```

```
    }
    Send_Agent(pNewAgent); // 에이전트 재구성
                &전송
}
MergeAgent()
{
    Recieve_Agent_Data(pData); // 외부로부터의 전송
    if (IsMerge_Agent(pData, pSpdData))
        AddingAgent(pOrgAgent, pData);
    if (pOrgData->IsEnd)
        Send_Agent_To_Entity(pOrgData);
}
```

3.5 에이전트 이동 및 수행

이동 에이전트의 코드에는 민감한 데이터가 제외된 상태이므로 에이전트가 결정을 내려야할 사항 즉, 사용자 요구에 대한 결론에 도달하였거나 또는 마지막 호스트에서 홈으로 귀환할 경우 이를 다시 MAG가 수신한다. 이때 이동 에이전트는 VPN과의 안전한 통신을 위한 SA(Security Association) 데이터를 호스트에게 할당하며 이를 통하여 호스트는 MAG과 보안 통신을 수행할 수 있다. 이때 SA 데이터는 VPN 장비에서 사용되는 프로토콜 및 암호화에 대한 정보를 포함하는 것으로 이동 에이전트의 고유 기능인 사용자 주문형태의 데이터를 수행할 수 있는 방안이다. 즉, 호스트의 이동 에이전트의 내부에 그림 6과 같은 내용을 포함하고 SA 데이터는 VPN에서 운영되는 데이터의 모듈을 탑재한다. 이 논문에서는 IPSEC을 위한 IKE(Internet Key Exchange)를 탑재하였다. 이 정보를 이용하여 호스트는 자신의 서명값을 전송하고 함께 IPSEC을 위한 난수값을 함께 보낸다. 보안협상을 통한 보안 채널(Tunnel)이 생성된 후 해당 에이전트를 암호화하여 에이전트를 MAG에게 전송하고 호스트는 필요한 데이터를 안전하게 호스트로부터 받는다. 이는 상호 인증을 거친 후 SPD상에 정해진 규칙에 따라 즉, AH(Authentication Header) 또는 ESP (Encapsulating Security Payload)로 IPSEC을 수행한다. 이때 MAG는 호스트가 필요로 하는 데이터 즉, 이동 에이전트 풀에 저장된 원본 데이터 중 민감한 데이터를 안전하게 전송한다. 전송된 데이터를 통해 호스트는 사용자의 원하는 작업을 종료할 수 있다. 호스트와 MAG 사이의 프로토콜은 그

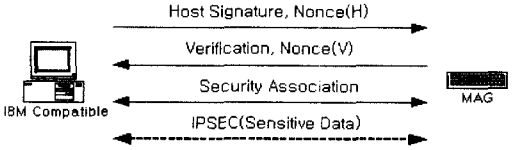


〔그림 7〕 재구성된 에이전트 구조

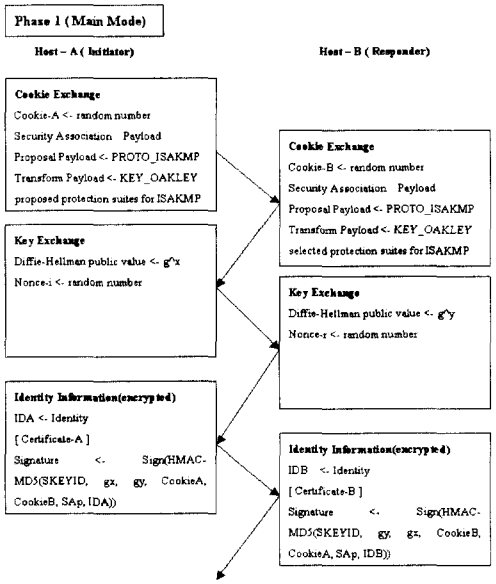
림 7과 같다. 상호 인증을 위해 서명값을 사용하고 암호화 프로토콜을 위해 난수를 생성하여 추가적으로 송/수신한다. IPSEC의 범위는 이 논문에서 벗어나므로 자세한 사항은 언급하지 않겠다.

그림 8에서 호스트의 Signature는 SSL과 같은 현태의 프로토콜에서 사용하는 전자서명 방법을 사용하기 위하여 인증서 기반을 사용한다. 인증서는 X.509에서 정의한 표준에 따르는 인증서를 사용한다. 난수는 각각 호스트 및 VPN에서 둘 사이의 세tus키를 만들기위한 매체로 사용된다. 다음으로 보안 협상과정은 Security Payload를 주고 받는 과정으로 즉, 사용할 알고리즘과 키의 크기 및 ESP(Encapsulation Security Payload - RFC2406) 또는 AH(Authentication Header - RFC2402)와 같은 통신 프로토콜을 정의한다.

이 과정은 ISAKMP(Internet Security Association and Key Management Protocol - RFC2408)에 따라 서례하였다. 여기서 키교환에 대한 인증으로 인증서를 사용하였고 이는 IKE(Internet Key Exchange - RFC2409) 프로토콜을 이용하여 사용하였다. 다음은 두 단계중 처음 단계를 나타내고 있다.



〔그림 8〕 호스트와 MAG간의 보안통신



〔그림 9〕 ISAKMP의 첫 번째 절

3.6 에이전트 작업 종료

위의 과정을 통해 호스트와 MAG간의 안전한 통신을 통해 이동 에이전트 고유의 목표를 달성하게 된다. 이때 호스트들에 대한 위험성 여부를 판단한다. 이는 원본 이동 에이전트의 리스트를 파악한 후 귀환되는 에이전트를 분석하여 얻을 수 있다. 즉, 정상적인 여정을 마친 이동 에이전트에는 해당 호스트의 서명이 포함되어 있고 만약 리스트를 다 여행하지 못했을 경우에는 마지막 호스트의 행위를 위심할 수 있다.

이를 통해 SPD안의 호스트 리스트를 갱신하고 또한 이동 에이전트 풀에서 원본 데이터를 적재하여 이에 대한 응답을 클라이언트에게 안전하게 전송한다. 따라서 클라이언트는 이동 에이전트 본래의 목적에 부합되는 한 번의 전송을 취하여 안전하게 결과를 받을 수 있고 민감한 데이터를 포함하였을 경우에도 이동 에이전트에 대한 보안성을 유지할 수 있다.

3.7 분석 및 결과

기존의 이동에이전트에 대한 보안으로 마련된 분리되지 않은 서명의 방법과 이 논문에서 제안한 VPN상에서의 IPSEC과의 보안성은 대동소이 하다고 볼 수 있다. 하지만 기존의 방법은 이동 에이전트 내부에 항상 암호화 모듈을 추가하여 움직여야하는 단점을 가지고 있다. 하지만 VPN을 이용할 경우 한번 방문한 호스트에 대해서 인증티켓을 가지고 접근하므로 다음 방문때는 ISAKMP에서 제안하는 2번째 절만 수행하면 되고 따라서 부가적인 암호코드에 대한 부담을 줄일 수 있다. 또한 이동에이전트가 단순한 데이터를 처리하였을 경우에는 상관이 없지만 많은 양의 데이터 즉, 데이터베이스와 연관된 작업을 수행할 경우 에이전트 자체에 모든 데이터를 추가하여 전송하기란 거의 불가능하다. 따라서 이러한 관점에서 볼 때 제3의 매체인 VPN을 통하여 안전한 통로를 확보하고 이를 통해 에이전트 고유의 작업을 수행하는 방안을 제시하였다. 그리고 하드웨어를 사용하여 이동 에이전트의 보안을 수행하는 방법은 호스트들이 연계되어 하드웨어에 대한 정보를 공유해야하는 관리적 차원에서도 문제점이 발견되었다. 다음 표는 이 논문에서 제시한 세가지 유형의 보안에 대해서 자체적인 평가를 제시하고자 한다.

Ⅳ. 결 론

기존의 분산환경에서 대두된 이동 에이전트는 불법적인 호스트들에 대한 위험성 때문에 실제 응용에 적용하는데 많은 문제점이 있었다. 본 논문에서는 안전한 제3의 개체인 VPN을 통해 이러한 이동 에이전트의 보안상의 문제점을 해결하였고 그에 따른 부담을 클라이언트에게는 투명하게 유지하였다. 따라서, 불법적인 호스트로부터 클라이언트의 이동 에이전트를 안전하게 유지할 뿐만 아니라 코드를 재생성함으로 인해 보다 민감한 데이터를 유출하는 것으로부터 막을 수 있다. 이는 현재 가장 많이 네트워크 상에서 사용되는 VPN 장비를 이용하여 구축함으로 인해 시설 추가에 따른 부담을 줄일 수 있다. 하지만, 제3의 개체를 더 경유해야하는 추가 부담으로 인한 오버헤드는 여전히 단점으로 남고 있다. 이를 좀 더 개선하기 위한 방안이 필요하겠다. 또한 SPD에 대한 점진적이고 적응적인 구축이 더 연구

〔표 1〕 비교 평가

유형 / 항목	분리되지 않는 서명	스마트 카드	에이전트 게이트웨이
보안성	좋음	좋음	좋음
융통성	보통	낮음	좋음
효율성	보통	낮음	좋음
경제성	좋음	보통	낮음

되어야하며 이동 에이전트 분할을 스크립트 단위에서 좀 더 효율적으로 해야하는 문제가 남아있다. 그러나 이러한 분야의 연구가 아직은 미흡한 상태여서 비교나 테스트할 환경이 부족한 상황이다.

참 고 문 헌

[1] D.C. Chess, C. Harrison, A. Kershenbaum, "Mobile Agents:Are They a Good Idea?", IBM Research Report, 1995.

[2] R. S. Gray, "Agent Tcl: A flexible and secure mobile-agent system", Proceedings of Fourth Annual Usenix Tcl", Workshop, pp.9-23, 1996

[3] Antonio Corradi, Rebecca Montanari, Cesare Stefanelli, "Security Issues In Mobile Agent Technology", http://www-lia.deis.unibo.it/Software/SOMA

[4] Crystaliz Inc., General Magic Inc., GMD Fokus, IBM Corp., "Mobile Agent Facility Specipication", Joint Submission Supported by the Open Group, OMG TC Document, November 1997.

[5] Hartmut Vogler, Thomas Kunkelmann, Marie-Louise Moschgath, "An Approach for Mobile Agent Security and Fault Tolerance using Distributed Transactions", Proceedings of the 1997 International Conference on Parallel and Distributed Systems, pp.268-274,1997 IEEE.

[6] Jian Tang, Jari Veijalainen, "Using Agents to Improve Security and Convenience in mobile E-Commerce", Proceedings of the 34th Hawaii International Conference on System Science-2001, 2001 IEEE.

[7] F. Hohl, "A model of Attacks of Malicious Hosts Against Mobile Agents", presented at 4th Workshop on Mobile Object Systems" Secure Internet Mobile Computations, France, 1998.

[8] Panayiotis Kotzanikolaou, "Secure Stansactions with Mobile Agents in Hostile Environments", pp.193-202, 1999 IEEE.

[9] Niranjan Suri, "NOMADS:Toward a Strong and Safe Mobile Agent System", Agent 2000 Barcelona Spain, pp.163-164, ACM 2000.

[10] Stefan Funfrocken, "Protecting Mobile Web-Commerce Agents with Smartcard", Mobile Security Agent, LNCS 1419, 1999, springer, pp.44-64, Giovanni Vigna (Ed.)

[11] S. Kent, R. Atkinson, "IP Authentication Header", November 1998, RFC 2402, available at http://www.cis.ohiostate.edu/Service/rfc/rfc-text/rfc2402.txt

[12] S. Kent, R. Atkinson, "IP Encapsulating Security Payload (ESP)", November 1998, RFC 2406, available at http://www.cis.ohio-state.edu/Service/rfc/rfc-text/rfc2406.txt

[13] D. Maughan, M. Schertler, M. Schneider, J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", November 1998, RFC 2408, available at http://www.cis.ohio-state.edu/Service/rfc/rfc-text/rfc2408.txt

[14] D. Harkins, D. Carrel, "The Internet Key Exchange (IKE)", November 1998, RFC 2409, available at http://www.cis.ohio-state.edu/Service/rfc/rfc-text/rfc2409.txt

[15] Casey Wilson, Peter Doak, "Creating and Implementing Virtual Private Network", pp.379-418, 2000, CORIOLIS (Ed.)

〈著者紹介〉

박 재 경 (Jae-kyoung Park) 준회원
1994년 2월 : 동국대학교 컴퓨터공학과 졸업
1996년 2월 : 홍익대학교 전자계산학과 석사
1996년 3월~현재 : 홍익대학교 전자계산학과 박사과정
1997년 1월~현재 : (주)퓨쳐시스템 정보통신 연구소 선임연구원
〈관심분야〉 네트워크 보안, Mobile Agent Security, VPN

원 유 헌(Yoo-Hun Won) 준회원
1972년 2월 : 성균관대학교 수학과(B.S)
1975년 2월 : 한국과학기술원(KAIST) 전자계산학과(M.S)
1975년 3월 : 한국과학기술 연구소 연구원
1985년 2월 : 고려대학교 대학원 컴퓨터과학과(Ph.D)
1986년 3월 : 미국 RPI대학 교환 교수
1976년 ~ 현재 : 홍익대학교 컴퓨터공학과 교수
〈관심분야〉 프로그래밍 언어, 객체지향 언어, 실시간 언어, 실시간 시스템, 멀티미디어 시스템, 정보보호