에이전트 코드와 상태 추적을 통한 이동에이전트의 보호 기법

정회원 정 창 렬*, 고 진 광*

Mobile Agent Protection Scheme through Execution Tracing of Agent Code and Status

Chang-Ryul Jung*, Jin-Gwang Koh* *Regular Members*

요 약

컴퓨터 기술의 확산과 더불어 분산 컴퓨팅 환경에서 이동에이전트 코드의 이동성은 유연성이 있어 인터넷상에서 분배된 애플리케이션들을 설치하는데 사용되고 있다. 동시에, 이동에이전트의 이동코드를 전송하고 멀리 떨어진 곳의 호스트로부터 수신할 수 있는 능력을 가지고 있어서 에이전트가 실행되는 호스트 시스템의 실행 환경은 에이전트의 코드와 이를 실행할 수 있는 실행 환경에 접근을 해야 함으로 악의적인 행위들로부터 에이전트의 위조, 변조, 잘못된 실행을 방지하기가 매우 어렵다.

따라서 본 논문은 암호화된 에이전트의 코드와 상태 추적을 통하여 여러 악의적인 행위로부터 에이전트를 보호할 수 있도록 하는 메카니즘을 제안한다. 제안한 메카니즘을 통해 이동에이전트가 보호되는 과정을 통해서 안전성을 증명하였다

Key Words : Mobile Agent, Execution Tracing, Mobile Code, Information Protection.

ABSTRACT

With the expansion of computer technology the mobility of a mobile agent code having the flexibility in the dispersive computer situation is used to set up the applications distributed on the Internet. As it also has the ability to transmit the mobile code of a mobile agent and to receive it from a far-off host, the executive circumstances of the host system in which an agent is executed have to access to an agent code and the executive state capable of executing an agent code. Therefore, it is difficult to prevent the forgery, the alteration and the wrong execution of an agent from a malicious host.

This dissertation suggests the mechanism which can protect an agent from the malicious action through the executive pursuit of a code-named mobile agent. The security of this mechanism is verified through the protective process of a mobile agent in this mechanism.

Ⅰ. 서 론

인터넷과 정보기술의 발전으로 인터넷환경과 같은 분산 네트워크에서는 에이전트기술이 발전하게 되었다. 이동에이전트가 지니는 이동성으로 인하여, 에이전트는 많은 분야에서 응용되고 있다. 분산 네트워크 환경에서 이동에이전트기술과 모바일 코드 기술들은 분산 컴퓨터 환경에서 코드 이동성의 일부 형태를 사용하는 시스템이자 언어이다. 이는 몇 개의 결합된 컴퓨터의 환경 또는 사이트가 네트워크 상에 밀집되어 있는데 에이전트들의 실행을 위해 지원한다. 에이전트들은 컴퓨터의 실행에 대한 코드 세그먼트와 콜 스택, 명령어 포인터와 같은

* 순천대학교 컴퓨터과학과 (chari7@empal.com),
논문번호 : 030229-0527, 접수일자 : 2003년 5월 27일

컴퓨터의 상태와 연관된 제어 정보를 포함하는 실행 상태를 가지고 있어 연속적인 명령을 처리한다. 이동에이전트의 기술은 크게 두 가지로 나눌 수 있다. 하나는 실행상태를 포함하지 않는 경우이고, 또 하나는 실행코드나 실행상태를 원거리 사이트로 보내어 이용하는 기술이다.[1,4]

실행상태를 포함하지 않는 시스템은 Java[2]와 Agletes시스템[3]등이 있고, 실행코드나 실행상태를 이용하는 경우의 elescript[5]과 Agent-Tcl[6]등이 있다.

모바일 코드를 이용한 전형적인 방법은 클라이언트-서버 접근[7]에서 상당한 유연성과 관습화를 가지고 있으나 보안에 대한 심각한 문제를 가지고 있다. 에이전트는 사용자를 대신해서 네트워크 상을 돌아다니면서 충돌할 수 있는 다양한 오브젝티브[8]를 가지고 서로 다른 사이트를 방문한다. 그러므로 이동에이전트 시스템은 악의적인 호스트나 에이전트로부터 에이전트 자신을 보호해야 한다. 또한 에이전트들은 사이트에서 다른 사이트로 이동하는 동안 도청이나 위조로부터도 보호되어야 한다.

이처럼 이동 에이전트는 신뢰할 수 없는 네트워크 상을 돌아다니는 동안 에이전트의 보호는 잘 알려져 있는 암호 프로토콜인 SSL (Secure Socket Layer)을 사용한다. 에이전트가 가지고 다니는 출처가 분명치 않는 소스에서 나온 코드로부터 호스트를 보호하기 위한 메카니즘이나 연구[9-13]가 최근에 많이 이루어지고 있다. 적절한 접근 제어와 sandboxing 메카니즘을 사용함으로써 실행 환경을 다양한 범위의 공격으로부터 보호가 가능해 졌다.

그러나 에이전트의 가장 힘든 보안 문제는 에이전트가 실행되는 호스트 시스템의 컴퓨터 환경으로부터 발생하는 공격에 대해서 에이전트를 보호하는 문제이다. 에이전트가 실행되는 호스트 시스템의 실행 환경은 에이전트의 코드와 이를 실행할 수 있는 실행 상태에 접근해야 함으로 악의적인 호스트로부터의 에이전트의 위조, 변조, 잘못된 실행을 방지하는 것은 매우 어렵다.

따라서 본 논문은 에이전트의 코드와 상태 추적을 통하여 여러 가지 요소들의 악의적인 행위로부터 에이전트를 보호하는 프로토콜과 메카니즘을 제안한다. 또한 프로토콜이 종료되고, 에이전트가 에이전트의 소유자에게 린턴(return)되면 수신된 메시지를 검토하여 현재까지 실행되었던 에이전트의 리소스(resource)를 확인하여 에이전트의 안전성 여부를 검증한다.

논문의 구성은 2장에서는 관련연구로서 이동에이전트와 모바일 코드보안에 대해 기술한다. 3장은 실행추적과 프로토콜에 대한 문제점을 지적하고 실행추적 프로토콜은 제안하고, 4장에서는 제안된 프로토콜에 의해 모바일코드의 암호화된 실행추적을 통해서 이동에이전트를 보호하는 메카니즘을 기술한다. 마지막 5장에서는 결론 및 향후연구에 대해 기술한다.

Ⅱ. 관련 연구

에이전트 보호에 관련한 많은 연구가 되고 있으나[8][11][13] 실제로 실행을 책임지는 해석기에서 발생하는 공격에 대해 방어하는 보호 프로그램은 의미를 가지지 못한다. 이런 경우 호스트가 에이전트를 올바르게 실행하여 완료할 수 있는 것과 에이전트가 가지고 있는 데이터가 변조나 유출되는 위협으로부터 완전하다는 보장은 불가능하다.[4][12-13] 왜냐하면 현실적으로 실행되어지는 호스트환경으로부터 발생가능한 공격에 대해서는 아직까지 이동 에이전트를 보호할 수 있는 해결책은 예방 메카니즘과 오류감지 메카니즘에서 찾고 있기 때문이다.

예방 메카니즘은 에이전트의 코드와 상태를 수정하거나 접근하는 것을 불가능하도록 위조-방지 장치를 채택하는 것이다. 이 장치들은 에이전트들을 물리적으로 봉인된 환경에서 실행하는 프로세서들이다. 때문에 시스템 내부에는 시스템 자체를 파괴하지 않고서는 소유자도 접근할 수 없다. 그러나 이러한 시스템들이 안전하게 보호할 수 있는 반면, 고가의 하드웨어를 필요하다. 그러므로 이 시스템들은 넓은 범위에서 사용되지 못하고 있다.

소프트웨어에 기초한 접근 방식으로는 code scrambling[14] 메카니즘이 있는데 이 기법은 모바일 코드는 원거리 사이트로 전송되기 이전에 "재배열"됨으로 코드를 수정하거나 다시 작업(re-engineer)하는 것을 어렵게 만들고 코드의 원래동작을 보호한다.

또 다른 해결방법으로는 실행 호스트가 에이전트의 컴포넌트를 암호화하여 사용하기 때문에 암호화와 복호화를 할 수 있어 안전하게 접

근을 허용할 수 있다. 그러나 미리 정해진 사이트에서만 사용할 수 있으며, 에이전트의 이동 루트를 어느 정도 알고 있어야 한다는 단점을 가지고 있어서 새롭고 유망한 접근 방식은 암호화된 기능들[4][12][15]을 이용한다. 이러한 경우에 모바일 코드는 몇 개의 외부 입력이 주어졌을 때 암호화된 값을 처리하는 알고리즘을 수행하는데 실행되는 사이트의 호스트는 실제로 컴퓨터에서 실행되는 기능에 관해 어떠한 단서도 가지고 있지 않기 때문에 알고리즘 실행과 실행 결과를 위조할 수 없다.

오류감지 메카니즘은 모바일 에이전트의 코드, 상태, 실행의 흐름을 불법적으로 수정하는 행위를 감지하는 것으로 정적코드와 동적인 성분을 지니는 다른 코드를 이용한다. 이유는 정적 코드는 디지털 신호들을 사용함으로써 쉽게 보호되지만, 상태, 실행의 흐름은 동적 성분이기 때문이다. 예를 들어, 상태 평가 메카니즘[7]은 모바일 에이전트를 상태 평가 기능과 결합한다. 이동하는 에이전트가 새로운 실행 환경에 도달했을 때, 평가 기능은 매개변수로서 통과하는 에이전트의 현재 상태로 평가받는다[16]. 평가 기능은 에이전트의 상태를 체크하는 과정에서 형식이 변화하지 않고 유지하는지를 체크한다. 이러한 방법은 에이전트의 상태를 위조하고자 하는 악의적인 행위의 시도들이 감지할 수 있다.

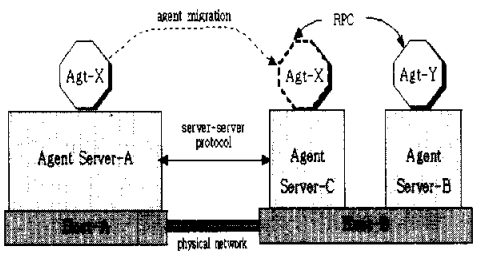


그림 1. 전통적인 이동에이전트 플랫폼

그림 1은 이동에이전트의 일반적인 물리적인 플랫폼을 나타낸 것으로 호스트와 호스트로 에이전트가 이동하는 과정과 각각의 호스트와 RPC에 의해 제어되는 과정이다. 이렇게 이루어져 있는 에이전트 플랫폼에서 에이전트를 보호하기 위해서 본 논문에서는 에이전트 코드, 상태, 실행 흐름을 불법적으로 수정하는 행위를 감지하여 에이전트의 안전한 활동을 보장하도록 에이전트를 보호하는 메카니즘으로 에이전트 실행동안 수집된 것들을 추적하였다. 이때의 추적은 사후분석이라 불려지는 post-mortem분석에 기반 하여 에이전트 데이터를 분석한다.

추적은 프로그램 실행의 증명을 위해 사용되어져 에이전트 프로그램의 실행 가능성을 비교하여 체크하기 위한 것이다. 만약 위조가 발생하였다면 이러한 방법을 사용해 에이전트 소유자는 에이전트에 대한 승인된 동작들이 올바르게 수행되지 않았음을 확인할 수 있다.

Ⅲ. 실행추적과 프로토콜

원격 실행추적 코드의 방법은 처음에 remote job submissions[17]으로 원격 실행 방법으로 사용되어 Obliq과 M0와 같은 몇 개의 기본적인 모바일 코드 시스템에 표현되다가 공식적으로 Stamos에 의해 정의되었다.

이러한 실행추적 기법은 이동에이전트의 이동성으로 더욱 많이 사용되고 있는데, 전통적인 에이전트 실행 추적은 호스트 플랫폼에서 실행되고 있는 이동에이전트의 실행 추적에 의해 생성된 코드의 정확한 라인에 추적된다. 추적되어지는 동안 모든 외부의 값은 이동에이전트에 의해 읽혀지고 실행되는데 호스트는 이러한 경로에서 실행되는 에이전트의 값을 저장한다. 이때 추적활동이 이루어지는데 모든 추적활동은 에이전트의 모든 경로에 의해 이루어지고, 에이전트 소유자에게 리턴(return) 되도록 하여 처음 호스트 플랫폼으로부터 에이전트 실행의 완벽한 추적을 한다. 추적에서 이동에이전트 기반의 정보를 실험하는 것을 포함한다. 모의실험은 이동에이전트의 다음 방문지 호스트의 상태와 확인을 중개하는 결과가 될 것이다. 에이전트 소유자는 에이전트 방문계획에 의해서 추적 플랫폼으로부터 실행되는 코드의 결과를 요구하고, 처리는 에이전트 방문계획에 의해서 모든 호스트에서 이루어진다.

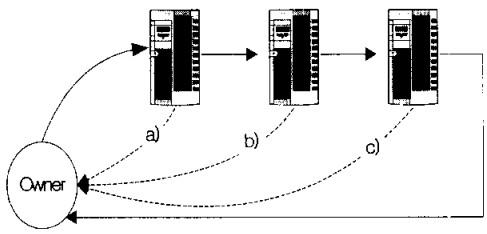


그림 2 전통적인 이동에이전트의 추적

그림 2는 전통적으로 실행되는 이동에이전트의 실행추적기법이 적용되고 있는 예 이다. 이러한 방법에는 추적의 상태가 체크되는 동안 한계가 있다. 만약 약간의 악의적인 호스트의 의해 이동에이전트의 코드나 상태에 문제가 발생하여 원래의 코드와 상태에 차이가 발생한다면 특별한 호스트 플랫폼에 의해 제공되어지는 추적의 검사 동안에 발견하여 악의적인 호스트를 찾을 수 있다.

그러나 이런 접근은 호스트에 의해서 유지되고 필요에 따라 추적을 실행함으로 해당 호스트의 로그 사이즈와 수에 커지는 문제가 발생한다. 그리고 검출처리기는 에이전트를 조작하므로 알아채기를 자극하는 요인이 된다. 또 다른 문제는 멀티에이전트를 추적하는 실행결과에서 차이가 발생할 수 있다는 것이다.

때문에 본 논문에서는 이러한 점들을 개선하기 위해서 검증 서버를 두어서 에이전트가 실행되는 동안 안전성을 보장 할 수 있도록 하는 실행프로토콜을 제안한다.

1. 검증 서버

검증서버는 호스트를 통해 전달되어지는 에이전트의 안전성을 보장하기 위한 것으로 호스트 시스템에서 에이전트의 추적이 이루어지는 것을 관리하는 서버이다.

그림 2에서와 같이 전통적인 추적방법에서는 에이전트가 이동하는 호스트에서 실행되는 상황을 소유자에게 알려서 현재 에이전트의 이동경로에 따라 소유자는 실행 추적을 할 수 있었다.

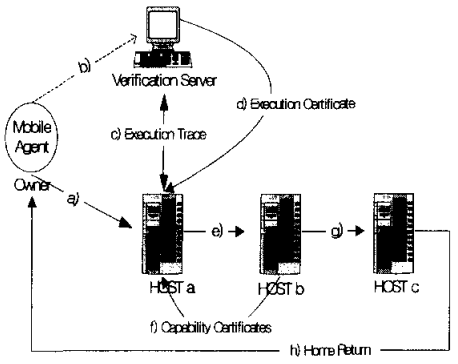


그림 3. 이동에이전트 실행 추적 -
검증서버 운영 프레임워크

그러나 소유자와의 토의나 협의 없이 검증서버를 이용하여 이동에이전트를 완벽하게 분석하여, Moreau의 제안을 바탕으로 하여 검증서버를 두어서 에이전트 소유자와의 커뮤니케이션 되는 등의 상호작용에 관계없이 실행추적이 이루어질 수 있도록 한다.

그림 3은 이동에이전트의 실행추적을 위한 검증서버를 두어서 운영되는 프레임 워크를 나타내고 있다. 그림 3의 프레임워크는 PKI에 기반을 두고 있는 권한 인증 환경이다. 에이전트 소유자는 이동에이전트를 네트워크를 통해서 보내는 역할을 한다. 호스트 플랫폼은 이동에이전트가 실행되는 환경으로서 에이전트가 다양한 일을 안전하게 수행하도록 하는 곳이다. Capability Certificates는 호스트 간 안전하고 정확하게 악의적이지 않는 방법에 의해서 실행할 수 있도록 확인하는 과정이다. 이처럼 확인은 각각의 서버플랫폼에서 가지고 있는 공개키와 비밀키를 이용하여 서명, 인증한다. 검증서버는 각 기능들로 구현되어 있는 사이즈나 구현되어 있는 복잡도 정도의 차이가 있는 것과 사이즈의 차이가 있는 것을 인식하고, 할당된 명세를 책임 있게 허가한다. Execution Certificates는 호스트 플랫폼이 검증서버에 의해서 비교하여 허락이 되면 성공적으로 수행할 수 있도록 검증결과를 호스트 플랫폼에 제공한다. 이때 Execution Certificates에서는 검증결과를 호스트 플랫폼에 알려주기 위해서 에이전트 코드와 상태를 단방향(one-way) 해쉬를 이용하고 그리고 timestamp등을 제공한다.

이동에이전트는 이동 하고자 하는 목적지의 호스트와 현재의 호스트간의 이동 플랫폼간의 상호작용을 한다(e). 서로 상호작용을 통해 현재의 탬플릿을 인증하거나 증명한다.

이동에이전트의 소유자로부터 서명된 에이전트의 코드 등을 호스트b 가 이동에이전트에 의한 이동되는 것을 검증한다(f). 이들 호스트들은 모든 명세를 인증하는 능력을 가지고 있어 이동할 것인가 그렇지 않을 것인가를 협의한다(f). 때문에 안전성 체크하고 나서 이동에이전트를 받아들일 것인가 아니면 취소 할 것인가를 진단한다. 만약 이동에이전트가 안전하지 않다면 호스트의 이동이 허락되지 않으므로 다음 동적인 원격지로 이동하여 일정계획을 수행한다.

또한, 이동에이전트의 소유자로부터 이동되어지면 호스트a 로 출발할 때(a) 이동에이전트의 코

드와 상태를 검증서버로 보내어 복사한다. 복사한 이동에이전트는 에이전트 실행추적을 위한 검증할 때 비교하여 안전성 검증을 한다. Execution Tracing은 검증서버를 통하게 됨으로 이동에이전트의 코드를 확인할 수 있다(c). 만약 이동 에이전트의 추적 검증결과가 정당하다면 실행인증(d)을 호스트 플랫폼에 알려 호스트가 정상적으로 실행을 준비하거나 실행할 수 있도록 한다.

그리고 여기서 복사되어진 이동에이전트의 코드나 상태는 인증서버에 보관되고 원래의 이동에이전트의 코드는 다음 호스트로 이동하여 다시 인증서버에 복사되어 보관되어 있는 이동에이전트 자료와 비교하면서 안전성 여부를 검증서버에서 검증한다.(g) 만약 추적하는 동안 악의적으로 tampering하는 경우에는 호스트 간 인증을 하지 않는다. 이는 인증 취소 리스트(CRL:certificate revocation list)에 의해서 이루어지는 데 나중에는 이러한 것들이 인증서버의 인증 취소 리스트 엔트리에 들어 있어 다른 서버 플랫폼에서도 인증이 무효화됨으로 실행되거나 인증이 이루어질 수 없다.

이러 과정을 반복적으로 거치면서 에이전트의 수행계획에 따라 이동을 하게 된다. 이렇게 이동을 하게 되면 처음 이동에이전트의 소유자에게로 돌아오게 된다.

이러한 방식으로 실행추적을 할 때 인증 서버를 중앙 두어 중앙 집중식 관리를 함으로써 서버간의 커뮤니케이션이 안전하고 관리적인 측면에서는 매우 효과적으로 관리를 할 수 있도록 한다.

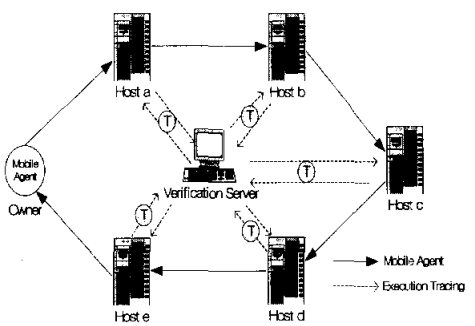


그림 4. 이동에이전트의 실행추적을 위한 검증서버 중앙집중 방식

그림 4는 이동에이전트의 실행추적을 위한 검증서버를 중앙에 두어서 이동에이전트의 플랫폼간의 이동경로에 따라 이동에이전트 실행을 추적하는 메카니즘으로 Ⓣ는 실행추적(execution tracing)을 의미한다.

본 논문에서는 이동에이전트 이동 경로에 검증서버를 두어서 이동에이전트의 플랫폼간의 이동경로에서 이동에이전트 실행을 추적한다.

만약 이동에이전트의 문제가 발생 할 경우 이동에이전트의 이동과정에서 이전의 검증서버나 다음 이동경로에 있는 검증서버를 통해서 실행추적을 하여 이동에이전트의 안전성을 보장받는다.

또한 이동에이전트의 이동과정을 검증하는 검증서버는 암호키를 사용하여 이동에이전트 호스트 플랫폼과의 상호작용이 이루어지도록 함으로써 이동에이전트는 안전하고 효율적으로 주어진 임무를 수행한다.

Ⅳ. 이동에이전트의 보호

본 논문에서 제안된 프로토콜은 안전하고 효율적으로 이동에이전트에게 주어진 임무를 수행 할 수 있도록 하고 있다. 이동에이전트 보호를 위해서 이동에이전트 이동과정을 통해 안전성을 증명한다.

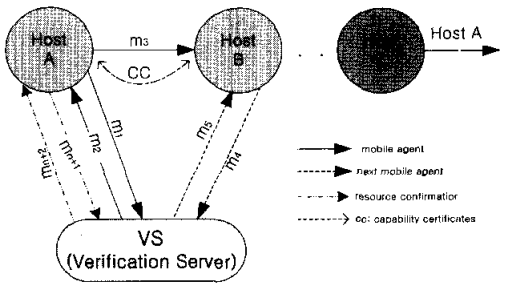


그림 5 에이전트 보호를 위한 검증 프로토콜

전달되어지는 기본적인 이동에이전트는 출발하는 에이전트 호스트 X에서 메시지 m_i를 이동에이전트 Y로 전송하고 메시지의 내용(cont1, cont2, . . . ,contn) 과 함께 연속적으로 전달된다.

$$X \xrightarrow{m_i} Y: cont1, cont2, \ldots, contn \quad (1)$$

위의 식(1)처럼 이동에이전트가 메시지를 포함하여 이동하면 이동에이전트 보호를 위해 검증서버 간 이동에이전트를 보호하기 위한 검증 프로토콜은 그림 5와 같다. 검증 프로토콜을 통해해 이동에이전트의 안정성을 보장하기 위한

검증하는 절차를 통해 안정성을 입증한다.

처음 에이전트 소유자로부터 출발하는 이동에이전트는 이동에이전트 코드 c와 현재 실행 상태 S_A를 검증서버에 전송한다. 처음 출발하는 S_A는 초기 상태가 된다.

$$A \xrightarrow{m_1} Vs: \ A_S(A, Vs, (K_A(c, S_A)), agent_A) \quad (2)$$

식(2)는 에이전트가 전송될 때 A에 의해 선택된 암호 키 K_A와 함께 에이전트 토큰(agent token)을 포함하여 검증서버에 전송한다. 에이전트의 토큰은 이동에 사용될 에이전트의 기본적인 데이터이다. 여기에서 토큰의 구조는 $A_S(A, i_A, t_A)$로 A_S는 에이전트 A로부터 전송되어지는 것을 의미한다. 에이전트 전송자 A, 에이전트 확인자 i_A, 에이전트가 발송된 시간 t_A등이다.

Vs(verification server)에 이동에이전트 m_1이 도착하였을 때 메시지와 에이전트 토큰을 체크하여 이동에이전트를 검증한다. 검증이 되면 execution certificate를 통해 다시 에이전트 서버 A에게 전송한다. 이때 검증과정에서 문제가 있다면 리젝트(reject) 할 수도 있다.

$$Vs \xrightarrow{m_2} A: \ Vs_S(Vs_p(Vs, A, i_A, H(p), M)) \quad (3)$$

이때 에이전트가 안전하다면 안전하다는 증명을 Vs는 자신의 비밀키와 단 방향(one-way) 해쉬 키를 이용하여 전자서명 한다. 서명된 증명을 사용하여 에이전트서버 A에게 전송한다. 에이전트 서버A는 다시 전송 되어온 에이전트 토큰의 메시지M을 확인하여 M이 리젝트(reject)를 의미한다면 A는 이동에이전트를 실행하지 않는다. 그렇지 않다면 M은 에이전트 실행에 안전하다는 의미임으로 이동 에이전트를 실행한다.

A는 에이전트가 다음 목적지로 이동하고 할 때까지 실행하고, 실행을 끝내고 이동에이전트를 B에게 Vs로부터 전달받은 (Vs_p)로 함께 전송한다.

$$A \xrightarrow{m_3} B:$$
$$A_S(A, B, agent_A, (K_A(c, S_A), Vs_p), H(m_3), t_A, M) \quad (4)$$

이동에이전트 A는 B에게 A의 처음과 같이 $agent_A$를 전송한다. 뿐만 아니라 에이전트가 발송된 시간 t_A와 Vs로부터 받은 (Vs_p)을 포함하여 전송한다. 그러면 에이전트 B는 A로부터 받은 에이전트 코드와 A에 의해서 선택된 랜덤키 K_A와 함께 암호화된 에이전트의 현재 상태 S_A를 포함하는 메시지 M을 수신한다. 수신하면 B는 이에 대응하는 A를 체크하여 B의 비밀키로 (Vs_p)를 해독하여 $agent_A$와 t_A시간에 에이전트가 A에 의해 전송되었는지를 확인하고 에이전트를 받아들인다. 전송받은 B는 식(5)와 같이 검증서버로부터 안전성을 검증받기 위해 A로부터 받은 (Vs_p)와 함께 Vs에게 전송한다. 검증서버로부터 안정성을 검증 받은 후 이동에이전트를 실행하게 되는데 만약 안전하다고 전송받은 에이전트가 검증서버에 의해서 안전하지 않다는 검증결과와 메시지를 execution certificate을 통해서 받으면 에이전트 실행을 중지하고 리젝트(reject) 한다.

$$B \xrightarrow{m_4} Vs: \ B_S(B, Vs, (K_B(c, S_B)), agent_B, Vs_p) \quad (5)$$

이들의 프로토콜은 에이전트의 실행이 종료될 때까지 계속해서 반복된다. 만약 마지막 에이전트 서버 플랫폼이 Z 라면 처음 실행한 에이전트 서버를 검색한다. 에이전트 소유자인 처음 실행한 서버를 찾아 접속을 한다. 이때 식(6)과 같이 마지막 에이전트 코드와 상태를 서명되어진 (Vs_p)와 함께 $agent_Z$를 에이전트 소유자인 A에게 전송한다.

$$Z \xrightarrow{m_n} A: \ Z_S(Z, A, (K_Z(S_Z), agent_Z, Vs_p, M) \quad (6)$$

A는 메시지의 안전성과 유효성을 체크하여 K_Z와 함께 에이전트의 현재 상태 S_Z를 포함하는 메시지 M을 전송 받는다.

$$A \xrightarrow{m_{n+1}} Vs: \ A_S(A, Vs, i_A, M, Vs_p) \quad (7)$$

A는 Vs에게 Z로부터 받은 에이전트와 메시지에 대한 안전성을 검증 받는다.

$Vs \overset{m}{\rightarrow}{}^{n+2} A$:

$Vs_S(Vs, A, i_A, A_p(K_Z), H(agent_Z), agent_Z, M)$ (8)

만약 안전하다면 Vs는 A에게 암호화 되어있던 키 $A_p(K_Z)$와 프로토콜을 종료시키는 메시지를 A에게 보낸다. 이때 $H(agent_Z)$와 수신된 메시지를 검토하여 현재까지 실행되었던 에이전트의 리소스를 확인한다.

V. 결 론

이동에이전트를 모든 응용분야에서 이용하는데 가장 큰 어려운 점은 이동에이전트가 지니고 있는 안전성에 대한 보안문제이다. 이런 안전성에 대한 보안문제를 해결하는 것이 매우 중요하다. 본 논문에서는 이러한 이동에이전트의 보안문제를 실행추적을 통해 이동에이전트를 안전하게 보호할 수 있도록 하는 메카니즘을 제안하였다. 실행추적은 에이전트 실행과 관련한 악의적인 코드의 변질을 통한 위조를 감지하기 위해 만든 메카니즘으로 에이전트의 소유자로 하여금 에이전트가 안전하게 실행되는 것을 보장받을 수 있도록 하는 검증서버를 두어서 확인한다. 실행 중에 이동에이전트의 코드와 상태가 악의적인 방법에 의해 수정되어 불안전한 이동에이전트가 실행함으로서 발생할 수 있는 에이전트 소유자의 오버헤드를 줄일 수 있다. 또한 실행 추적기법은 에이전트가 실행 후에 위·변조를 감지할 수 있음으로 위변조의 시기를 감지하기 위해 timestamp를 주어서 감지 할 수 있도록 하였다.

뿐만 아니라 에이전트의 안전한 실행을 보장받을 수 있도록 하기 위해 검증서버를 중앙에 두어서 이동에이전트와 에이전트 서버는 항상 안전성을 검증 받도록 하였다. 검증을 받는 것은 이동에이전트가 목적한 바의 일을 충실히 수행하고, 목적하는 바를 안전하게 실행하고 목적지에 귀환할 수 있도록 하였다. 본 논문에서 제안된 메카니즘의 안정성을 체크하기 위해서 이동에이전트의 이동과정을 통해 통해서 안정성을 검증하였다.

향후연구로는 검증서버를 분산시켜서 이동에이전트를 검증할 수 있도록 하는 지역 검증서버와 같이 분산할 수 있는 메카니즘과 각각의 이동에이전트 서버에서 검증서버의 기능을 수행할 수 있는 이동에이전트 시스템 모델에 대한 연구가 계속적으로 이루어져야 할 것이다.

참 고 문 헌

[1] G. Cugola, C. Ghezzi, G. P. Picco, and G. Vigna, "Analyzing Mobile Code Languages," *In Mobile Object Systems : Towards the Programmable Internet, Vol. 1222 of LNCS*, pp.93-111, Springer, April, 1997.

[2] Sun Microsystems. 「The Java Language: An Overview」. Technical Report, Sun Microsystems, 1994.

[3] D. B. Lange and D. T. Chang. IBM Aglets Workbanch-Programming Mobile Agents in Java, IBM Corp. white paper, September 1996.

[4] G. Vigna, "Protection Mobile Agent through Traceing ", *In Proceedings of the Third Workshop on Mobile Object Systems(ECOOP Workshop'97)*, Finland, June 1997.
http://cui.www.unige.ch~ecoopws/ws97/paper/vigna.ps.

[5] R. S. Gray, "Agent Tcl: A Transportable Agent System", *In Proc. of the CIKM Workshop on Intelligent Information Agents*, Baltimore, Md., Dec. 1995.

[6] J. E. White, Telescript Techology: Mobile Agents, In the Software Agents, AAAI Press/MIT Press, 1996.

[7] A. Carzaniga, G. P. Pico, and G. Vigna, "Designing Distributed Applications with Mobile Code Paradigms, *In Proc. of the 19th Conference on Software Engineering (ICSE'97)*, pp.22-32, ACM Press, 1997.

[8] J. Ordille, "When agents Roam, Who can you trust?", *In proc. of the First Conference on Emerging Technologies and Applications in Communications*, May 1996.

[9] A. Villazon, W.Binder, "Portable Resource Reification in Java-based Mobile Agent

Systems", *In Mobile Agents : Proc. of the 5th International Conference, Number 2240 in LNCS*, Springer-Verlag, Altanta, USA, 2001.

[10] Lue Moreau, Christian Queinnec, "Distributed Computations Driven by Resource Consumption", *In Proc. of IEEE International Conference on Computer Languages(ICCL'98)*, Feb., 1998.

[11] H. Reiser, G. Vogt, "Security Requirments for Management Systems using Mobile Agents", *In the Proc. of the 5th IEEE Symposium on Computers and Communications: ISCC 2000*, Antibes, France, July 2000,

[12] H. K. Tan, L. Moreau, "Extending Execution Tracing for Mobile Code Security", *In Proc. of the 2nd International Workshop on Security in Mobile Miti-Agent Systems*, associated to AAMAS-2002, Bologna, Italy, July, 2002.

[13] W. M. Farmer, J. D. Guttman and V. Swarup, "Security for Mobile Agents: Issues and Requirements", *In Proc. of the 19th National Information Systems Security Conference*, pp.591-597, Baltimore, MD, USA, Oct. 1996.

[14] M. Strasser, K. Rothermel, "System Mechanisms for Partial Rollback of Mobile Agent Execution", *In Proc. 20th International Conference on Distributed Computing Systems(ICDCS2000), IEEE Computer Society*, L.A, California, pp.20-28, 2000.

[15] T. Sander, C. Tschudin, "Towards Mobile Cryptography", *In Proc. of the 1998 IEEE Symposium on Security and Privacy*, Oakland, CA, May 1998.

[16] G. Vigna, "Cryptographic Traces for Mobile Agents", *Mobile Agents and Security*, pp.137-153, LNCS1419, Springer-verlag, 1998.

[17] J. K. Boggs, "IBM Remote Job Entry Facility:Generalize Subsystem Remote Job Entry Facility, IBM Technical Disclosure Bulletin 752, IBM, August 1973.

[18] J. W. Stamos et. al, "Implementing Remote Evaluation, IEEE Transaction on Software Engineering, Vol.16 No.7 pp.710-722, July, 1990.

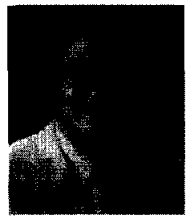
[19] C. R. Jung et. al. "An Agent System Protection Mechanism for Secure Action of Mobile Agent in Open Network System", in the Journal of KIMICS, Vol.6. No.7, pp.371-378, Korea, April, 2002.

정 창 렬(Chang-Ryul Jung) 정회원

1995년 2월 : 광주대학교 컴퓨터공학과 졸업
1999년 8월 : 순천대학교 컴퓨터교육과 석사
2000년 3월~현재 : 순천대학교 컴퓨터과학과 박사수료
2003년 6월 : Dept. of Computing Science. Alberta State University. Visiting Researcher.

<주관심분야> Information Security, Mobile Agent, Watermarking, Image processing

고 진 광(Jin-Gwang Koh) 정회원



1982년 2월 : 홍익대학교 컴퓨터공학과 졸업
1984년 2월 : 홍익대학교 컴퓨터공학과 석사
1997년 8월 : 홍익대학교 컴퓨터공학과 박사
1997년 8월~1998년 8월 : Oregon State University. 컴퓨터공학과 방문교수
2000년 12월~2001년 2월 : 일본 류큐대학 정보공학과 방문교수
2001년 3월~2002년 8월 :순천대학교 정보전산원장
1988년 3월~현재 : 순천대학교 공과대학 정보통신공학부 정교수

<주관심분야> Database, Information Security, Electronic Commerce, Watermarking