이동 에이전트의 안전한 전송을 위한 무결성 매커니즘

정은주° 김영갑 정동원 백두권
고려대학교 컴퓨터학과
{violetto°, ygkim, withimp, baik}@software.korea.ac.kr

An Integrity Mechanism for Secure Transmission of Mobile Agents

Eunju Jeong° Younggab Kim Dongwon Jeong Dookwon Baik
Software System Laboratory, Dept. of Computer Science, Korea University

요 약

이동 에이전트가 갖는 많은 장점에도 불구하고 실생활에서 이동 에이전트를 사용하는데 보안 문제는 큰 장애물로 작용하고 있다. 이동 에이전트 보안문제는 크게 이동 에이전트에 대한 보안과 이동 에이전트 시스템에 대한 보안으로 나눌 수 있는데 실질적인 이동 에이전트 보호에 대한 연구는 미비하다. 이동 에이전트나 이동 에이전트 시스템을 불법적인 공격으로부터 보호하기 위해 요구되는 가장 기본적인 서비스는 기밀성, 무결성, 인증, 접근 통제, 부인 봉쇄 및 가용성 등이며, 이 중 무결성은 보안 서비스가 제대로 작동하기 위한 가장 기본 서비스라고 할 수 있다. 일반적으로 무결성은 해쉬함수를 사용해서 무결성 검사값을 생성한 후, 데이터나 코드의 불법적인 변경을 감시한다. 따라서 이 논문에서는 에이전트를 전송 또는 수신하는 호스트의 상호인증과 해쉬함수(HMAC)를 사용해 이동 에이전트의 안전한 전송을 위한 무결성 매커니즘을 제안하고자 한다.

1. 서론

기존의 클라이언트/서버 패러다임에서 제시된 문제점들을 해결하고자 이동 에이전트 패러다임이 제시되었다. 이동 에이전트란 자율적으로 네트워크 상의 여러 호스트로 이동할 수 있는 에이전트를 말한다[1]. 이동 에이전트는 이동성, 지능성, 자율성, 적응성, 통신 및 협력성 등의 많은 장점을 지니고 있다[2]. 그러나 이동 에이전트의 많은 장점에도 불구하고 이동 에이전트의 보안 문제는 이동 에이전트를 실생활에 활용하는데 큰 장애물로 작용하고 있다.

이동 에이전트의 보안 문제는 크게 2가지로 구분할 수 있는데, 이동 에이전트 시스템에 대한 보안과 이동 에이전트에 대한 보안이다.

이동 에이전트 시스템에 대한 위협은 불법적인 이동 에이전트 시스템 또는 에이전트가 정당한 사용자로 가장하여 인증을 얻거나, 시스템 내의 자료나 서비스에 불법적으로 접근해서 사용함으로써 이동 에이전트 시스템의 자원 소모, 정보 노출 또는 훼손을 목적으로 한다. 이러한 위협으로부터 이동 에이전트 시스템을 보호하기 위한 대응책은 소프트웨어 기반 결점 고립, 안전한 코드 번역, 서명 코드, 상태 평가, 경로 기록 등이 있다[3].

이동 에이전트에 대한 위협은 코드 및 데이터가 불법적인 공격자에 의해서 도청, 내용 변조, 또는 삭제 등이다. 이런 위협으로부터 이동 에이전트를 보호하기 위한 방법은 부분 결과 캡슐화, 상호여정 기록, 실행 추적, 환경 키 생성 등이 있다[3]. 그러나 현재 이러한 이동 에이전트 보호 방법을 제공하는 시스템은 미비한 상태이다.

위에서 언급한 불법적인 공격으로부터 이동 에이전트를 보호하기 위해 요구되는 가장 기본적인 보안 서비스는 기밀성, 무결성, 인증, 접근 통제, 부인 봉쇄, 가용성 등이 있다. 이 중 무결성(Integrity)이란 시스템 내부의 보안 데이터에 대한 인가되지 않은 변경을 감지함과 동시에 네트워크를 통해 전송되는 데이터에 대해 전송 도중 고의적인 변경과 하드웨어 문제로 인해 발생될 수 있는 변경을 감지하는 보안 정책이다. 만약 무결성이 제공되지 않아 데이터의 불법적 변경 및 삭제가 발생하게 되면, 보안 서비스 전체의 기능이 제 역할을 수행할 수 없게 된다.

따라서 이 논문에서는 네트워크를 통해 전송되는 에이전트에 대한 무결성을 제공하는 매커니즘을 제안하고자 한다. 2장에서는 기존 이동 에이전트 시스템의 보안 정책 및 무결성 서비스에 대해 살펴보고, 3장에서는 이동 에이전트의 안전한 전송을 위한 무결성 매커니즘을 제안한 후, 4장에서 결론 및 향후 연구 과제에 대해 서술하도록 한다.

2. 관련 연구

2.1 무결성

무결성을 제공하기 위한 방법으로는 원래의 데이터와 동일한 복사본을 유지하여 비교하는 방법과 원래 데이터의 정보를 유지하여 현재의 데이터 정보와 비교하는 방법, 해쉬함수(hash function)와 같은 단방향(one-way) 함수를 이용하여 만든 값을 유지하는 방법 등이 사용될 수 있다[13],[14].

데이터의 복사본을 유지하는 방법은 정확성을 제공할 수는 있으나, 공간 및 시간상의 오버헤드가 발생할 수 있다. 원래 데이터의 정보를 유지하여 비교하는 방법은 에이전트에 대한 정보를 유지하기 위한 정보의 변경이 가능하기 때문에 무결성 제공을 위한 정확한 검사값으로 사용하는데 문제가 있다[13]. 이런 이유 때문에, 일반적으로 무결성 검사값 생성을 위해서 해쉬함수가 많이 사용된다. 해쉬함수 크게 두 가지로 구분할 수 있는데, 키를 사용하는 해쉬함수인 MAC(Message Authentication Code)과 키를 사용하지 않고 정수론 등

수학적 분야에 기초한 단방향 함수를 이용하는 해쉬함수가 있다.

2.2 이동 에이전트 시스템의 보안 정책

보안 정책을 제공하는 이동 에이전트 시스템 중 Aglet[6], Voyager[7], Concordia[8], Agent Tcl[9], Ajanta[10], SOMA[11]의 보안 정책을 요약하면 표1과 같다.

<표 1> 이동 에이전트 시스템의 보안 정책

이름	이동 에이전트 시스템 보호	이동 에이전트 보호
Aglet	- proxy object를 사용해서 접근 제어 - trusted와 untrusted 정책	X
Voyager	Java Security Manager 확장	X
Concordia	사용자의 identity에 따라 Security Manager를 사용해서 접근 제어	X
Agent Tcl	- 보안 정책에 따라 접근 제어 - anonymous와 authenticated 정책	X
Ajanta	- proxy based 매커니즘을 사용해서 접근 제어 - 에이전트 소유자 기반	에이전트 소유자가 상태를 보호할 수 있는 암호 매커니즘
SOMA	룰 기반으로 계층화된 보안 정책에 따라 접근 제어	수집한 데이터에 대한 무결성 제공

현재, SOMA, Ajanta 등의 시스템에서 이동 에이전트 보호를 제공하고 있지만 아직 연구가 미비한 상태이다. SOMA는 MH(Multiple-Hops) 프로토콜을 사용하여 수집한 데이터에 대한 무결성만을 제공하고 있다. Ajanta는 전자서명을 사용해서 에이전트 소유자가 이동 에이전트를 보호하는 방법을 제공하고 있는데, 에이전트의 정보, 공개 및 인증기관의 인증서 등을 포함해야 하기 때문에, 인증기관(Certificate Authority, CA)과의 연동과 관련된 연산 작업으로 인하여 인증 속도가 느리다는 단점이 있다.

또한, 이동 에이전트의 안전한 전송을 위한 무결성 매커니즘을 제공하는 이동 에이전트 시스템들은 자바 보안을 기반으로 해서 무결성을 제공하고 있지만 이에 대한 자세한 언급하고 있지 않다.

3. 이동 에이전트의 안전한 전송을 위한 무결성 매커니즘

이 장에서는 에이전트를 전송 및 수신하는 호스트 간에 상호인증을 통해 공유 비밀키(shared secret)이라는 값을 생성하고, 생성된 공유 비밀키를 키로 사용하는 해쉬함수(HMAC)을 통해 MAC 값을 생성하여 에이전트의 무결성을 인증하는 매커니즘을 제시한다.

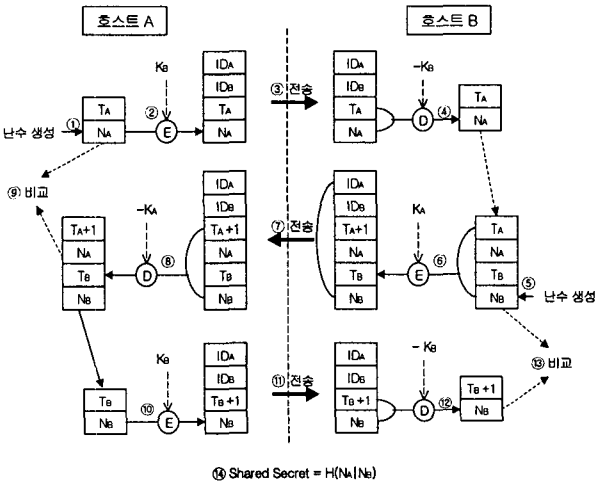
3.1 상호인증 방법

호스트 A가 특정 상품의 가격 정보를 수집하고자 에이전트 A를 호스트 B에 보내려고 한다면, 호스트 A는 호스트 B에 에이전트를 전송하기 전에, 호스트 A와 B는 신뢰성 있는 호스트인가를 검사하고자 할 것이다.

호스트 A는 호스트 B의 신뢰성을 검사하고자 할 것이고, 호스트 B도 호스트 A에서 전송한 에이전트의 신뢰성을 검사하고자 할 것이다. 그러므로 그림 1의 인증 방법을 사용하여 호스트 A와 B를 상호인증 한다.

호스트 A와 B를 상호인증 할 때, 공유 비밀키를 생성하는데, 이 값은 호스트 A와 B에서 난수를 암호화하여 전송한 값이므로 악의적인 다른 호스트가 이 값을 알 수 있는 방법은 없다고 할 수 있다.

그러므로 호스트 A에서 호스트 B로 에이전트 A'를 보내고자 한다면(또는 그 반대의 경우), 최초의 인증 과정을 수행했기 때문에, 인증을 하지 않고 에이전트를 전송할 수 있다.



<그림 1> 호스트 A와 B의 상호인증 방법

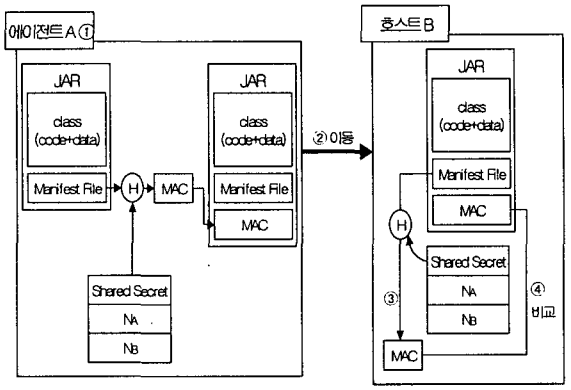
① 호스트 A는 타임 스탬프(timestamp) 값(T_A) 및 난수(N_A)를 생성한다.
② 호스트 B의 공개키(public key)를 이용해서 암호화한다.
③ 암호화한 값과 호스트 A와 B의 ID를 함께 묶어 호스트 B에 보낸다.
④ 호스트 B는 암호화 한 T_A와 N_A를 호스트 B의 비밀키를 사용해서 복호화한다.
⑤ 호스트 B는 타임 스탬프 값(T_B) 및 난수(N_B)를 생성한다.
⑥ 호스트 A의 공개키를 이용해서 암호화한다.
⑦ 암호화한 값과 호스트 A와 B의 ID를 함께 묶어 호스트 A에 보낸다.
⑧ 호스트 A는 암호화 한 T_A+1, N_A, T_B, N_B를 호스트 A의 비밀키를 이용해서 복호화한다.
⑨ 호스트 A는 호스트 B에 보냈던 N_A와 자신이 생성한 N_A를 비교한다. 두 값이 같다면 호스트 A가 호스트 B를 인증한 것이다.
⑩ 복호화한 T_B와 N_B를 호스트 B의 공개키를 이용해서 암호화한다.
⑪ 호스트 A는 호스트 B로부터 받은 T_B와 N_B를 호스트 A와 B의 ID를 함께 묶어 호스트 B에 보낸다.
⑫ 호스트 B는 암호화 한 T_B, N_B를 호스트 B의 비밀키를 이용해서 복호화한다.
⑬ 호스트 B는 호스트 A에 보냈던 N_B와 자신이 생성한 N_B

를 비교한다. 두 값이 같다면 호스트 B가 호스트 A를 인증한 것이다.

⑭ ⑨와 ⑬의 비교 결과가 같다면 호스트 A와 호스트 B는 상호인증이 된 것이다. 인증된 N_A와 N_B를 사용해서 (어떤 방법으로) 공유 비밀키를 생성한다.

3.2 무결성 매커니즘

호스트 A와 호스트 B가 상호인증 됐다면, 그림 2와 같은 매커니즘을 사용해서 에이전트 A는 호스트 B로 이동한다. 이 때, 에이전트 A가 수행될 호스트 B는 호스트 A와 상호인증이 이루어졌으므로, 호스트 A와 B는 신뢰할만한 실행 환경(호스트)이라고 가정한다.



<그림 2> 안전한 전송을 위한 무결성 매커니즘

① 에이전트 A는 코드 및 데이타를 JAR로 압축한다. 생성된 매니패스트(manifest) 파일을 공유 비밀키(그림 1 참조)를 키로 하여 MAC 값을 생성한다. 생성된 MAC 값을 전송할 JAR에 추가한다.

② 에이전트 A(JAR 파일)는 호스트 B로 이동한다.

③ 호스트 B는 JAR를 읽어, 전송된 매니패스트 파일을 자신의 공유 비밀키를 키로 하여 MAC 값을 생성한다.

④ 생성된 MAC 값을 에이전트 A의 MAC 값과 비교한다. 두 값이 일치한다면 에이전트 A의 내용은 수정되지 않았다.

에이전트 A가 호스트 B에서 정보 수집을 끝내고, 호스트 C로 이동하고자 한다면, 호스트 B와 C는 그림 1처럼 상호인증 과정을 수행한다. 그 후, 에이전트 A는 호스트 B에서 수집한 정보와 수행 코드를 JAR로 묶어 그림 2의 방법으로 호스트 C로 이동한다.

이동 에이전트 A는 정보 수집을 위해 자율적으로 여러 호스트들을 방문하고 정보 수집이 끝나면, 홈 호스트(호스트 A)로 돌아오게 된다. 홈 호스트로 돌아온 에이전트는 무결성을 검사한 후, 수집한 가격 정보를 비교분석한다. 그러나 이동 중 무결성 검사를 통해 이동 에이전트의 정보가 변경되었다는 것이 감지되면 홈 호스트로 소환되게 된다.

PKI 매커니즘과 비교할 때, 이 매커니즘의 장점은 MAC 값을 사용하기 때문에 인증 정보 작고, 인증이 빠르다는 장점이 있다. 그리고 인증 정보가 작기 때문에 빠른 전송이 가능하다. 또한, 에이전트가 전송될 때마다 MAC 값이 새로 생성되기 때문에 키의 신선도가 높다는 장점이 있다.

4. 결론 및 향후 연구 과제

이동 에이전트의 보호를 위한 보안 서비스 중 가장 기본이 되는 것은 무결성이다. 무결성이란 시스템 내부의 데이터에 대한 변경 및 시스템을 통하여 전송된 데이터의 불법적 변경을 감지하여 적절한 조취를 취하는 보안 정책인데, 이 논문에서는 전송된 이동 에이전트가 불법적으로 변경되었는지를 검사해 주는 무결성 매커니즘을 제안하였다.

이동 에이전트의 안전한 전송을 위해서 이동 에이전트의 전송과 수신이 이루어지는 호스트를 상호인증 하였고, 이동하는 에이전트에 대해서 HMAC 함수를 사용하여 변경 여부를 검사하였다. 이 매커니즘의 장점은 빠른 인증 및 전송과 키의 신선도가 높다는 것이다.

이 논문에서는 네트워크를 통하여 전송된 에이전트가 불법적으로 변경된 경우에 대한 무결성 매커니즘만을 제시하였는데, 향후 연구에서는 제시한 매커니즘에 대한 구현 및 검증이 필요하다. 또한, 완전한 이동 에이전트의 보호를 위한 무결성 매커니즘은 이동한 호스트 내부에서 에이전트가 변경되었는가를 감지할 수 있는 무결성 매커니즘이므로 호스트 내부의 변경을 감지할 수 있는 무결성 매커니즘에 대한 연구가 더 필요하다.

참고문헌

[1] Danny B. Lange and Mitsuru Oshima, "Programming and deploying Java mobile agents with aglets", Addison-wesley, 1998

[2] Danny B. Lange and Mitsuru Oshima, "Seven Good Reasons for Mobile Agents", Vol. 42, No. 3, Communications of the ACM, March 1999

[3] Wayne Jansen and Tom Karygiannis, "Mobile Agent Security", NIST, Special Publication 800-19, August 1999

[4] http://java.sun.com/security/index.html

[5] Roshan Thomas, "A survey of Mobile Code Security Techniques", the 22nd NISSC Proceedings, October 1999

[6] Gunter Karjoth, Danny B. Lange, and Mitsuru Oshima. A Security Model for Aglets. In Giovanni Vigna, editor, Mobile Agents and Security, LNCS 1419, Springer-Verlag, 188-205, 1998

[7] http://www.recursionsw.com/products/voyager

[8] T. Walsh, N. Paciorek and D. Wong, "Security and Reliability in Concordia", the 31st Annual Hawaii International Conference on System Sciences, (HICSS31)" in Kona, Hawaii, 1998

[9] Robert S. Gray et al., "D'Agent: Security in a multiple-language, mobile-agent system", Mobile Agents and Security, LNCS 1419, Springer Berlin Heidelberg, 1998

[10] Neeran Karnik, Anand Tripathi, "Security in the Ajanta Mobile Agent System", Software - Practice and Experience, January 2001

[11] A. Corradi, R. Montanari and C. Stefanelli, "Mobile Agents Integrity in Ecommerce Applications", Proceedings of the 19th IEEE ICDCS'99, Austin, Texas, May 1999

[12] Paolo Maggi, Riccardo Sisto, "Experiments on Formal Verification of Mobile Agent Data Integrity Properties", WOA workshop, november 2002

[13] William Stallings, "Cryptography and Network Security : Principles and Practice(3rd Edition)", Prentice Hall

[14] Shari Lawrence Pfleeger , Charles P. Pfleeger, "Security in Computing (3rd Edition)", Prentice Hall