논문 01-26-1A-13

한국통신학회논문지 '01-1 Vol.26 No.1A

Okamoto 다중서명을 이용한 이동에이전트 보안시스템의 설계

준회원 김 경*, 정회원 배 용 근*, 정 일 용*

The Design of Secure Mobile Agent System Employing Okamoto Digital Multi-Signature Scheme

Kyung Kim* *Associate Member*, Yong-Geun Bae*, Il-Yong Chung* *Regular Members*

요 약

이동 에이전트 시스템은 최근 각광받기 시작한 새로운 컴퓨팅 기법으로 서버의 과부하를 줄이고, 네트워크 이벤트의 발생을 줄이면서 필요한 서비스를 사용자에게 지원하기 위한 해결책으로 연구되고 있다. 이동에이전트 시스템은 해결해야할 많은 과제들이 있으나 그 중에서 주요한 문제로 인식되고 있는 것이 보안 문제이다. 본 논문은 이동에이전트 시스템의 보안고려사항을 분석하고 공개키 기반의 암호시스템과 일방향함수를 사용하는 디지털 다중서명 기법을 적용하여 이동에이전트 시스템의 보안구조를 제안한다.

ABSTRACT

Mobile agent system is a new computing paradigm in the spotlight recently, it is considered as a solution which reduces server overload and network event, supports the services to the user. Security issue is a matter of prime importance of mobile agent system which has many problems to be solved. This paper analyzes security elements to be considered of moblie agent system and proposes a secure mechanism based on Okamoto Digital multi-signature scheme which employs public-key cryptosystems and a one-way hash function.

I. 서 론

최근 컴퓨터시스템과 네트워크시스템 분야에서는 '대리인'의 개념을 이용하여 사용자의 시스템 사용을 보다 쉽도록 하는 연구가 각 분야에서 활발히 연구되고 있다. 다중 에이전트 (Multi Agent), 이동에이전트(Mobile Agent), 보조에이전트(Assistant Agent), 사용자인터페이스 에이전트(User Interface Agent), 지능형 에이전트(Intelligent Agent) 등의 연구분야로 나누어 볼 수 있는 에이전트(Agent) 시스템[1]의 등장은 당연한 기술적 요구로 받아들여지고 있는 듯하다. 그 중에서 서버의 과부하를 줄이고, 네트워크 이벤트의 발생을 줄이면서 필요한 서비스를 사용자에게 지원하기 위한 통신 응용 프로그램들은 데이터와 코드를 결합한 객체의 이동 즉, 이동에이전트를 사용하고 있다. 이동 에이전트 시스템[2]은 최근 각광받기 시작한 새로운 컴퓨팅 기법으로서 원격시스템의 코드를 호출하는 기존의 RPC (Remote Procedure Call)와는 다르게 실행코드가 네트워크를 통해 해당시스템으로 전송되어 실행되는 방식이다. 소프트웨어 에이전트 기술은 인공지능 분야에서 인간을 대행하는 컴퓨터 프로그램에 대한 연구분야인데 이 기술에 이동코드 개념이 결합된 것이 이동 에이전트 기술이다. 이동 에이전트는 현재 객체의 상태와 실행 가능한 코드를 포함하는 객체로서 분산환경에서 특정서비스를 제공하기 위하여 서로 통신할

* 조선대학교 컴퓨터공학부(iyc@mina.chosun.ac.kr)

논문번호 : 00355-0906, 접수일자 : 2000년 9월 6일

※ 본 논문은 2000년도 조선대학교 학술연구비의 지원을 받아 연구되었음

필요가 있는 호스트간 직접 이동을 통하여 실행되는 객체로서 자신의 홈 플랫폼에서 출발하여 주어진 일을 완수할 때까지 네트워크상의 플랫폼들을 이동한다. 이동에이전트는 독립적이고 자의적으로 이벤트가 발생했는지 등의 상태를 볼 수 있고(watch), 인터넷에서 원하는 정보를 어디에 있는지 이동하면서 찾을 수 있고(search), 여러 가지 서비스를 조화롭게 처리(orchestrate)[3]할 수 있다. 이와 같은 이동에이전트는 전자상거래, 그룹공동작업, 이벤트 모니터링, 작업흐름 자동화, 정보검색, 망관리 및 이동컴퓨팅 등에 이용될 수 있어 그 활용 범위가 매우 넓다. 그러나 분산 에플리케이션의 구성에 유연한 환경을 제공하는 이동에이전트 시스템은 심각한 보안 문제를 안고 있다. 첫째는 불완전한 통신채널을 이용하는 것에 대한 기본적인 보안 문제가 있으며, 둘째는 에이전트가 행하는 불법적인 행위나 공격에 대한 호스트 컴퓨터와 에이전트 서버의 보호문제이고, 셋째는 에이전트 서버가 행하는 에이전트공격에 대한 보안 문제이다. 에이전트가 호스트 컴퓨터를 공격하는 경우 디지털서명을 이용한 에이전트의 인증과 접근레벨에 따른 접근제어[4]로서 해결될 수 있다. 또한 JAVA 보안모델이 지원하는 보안 메카니즘을 사용하면 서버 보안 문제를 해결할 수 있다. 하지만 에이전트는 이동한 서버에게 모든 것이 노출되고 서버는 에이전트의 코드나 상태를 변경함으로서 에이전트의 행동을 변경/방해할 수 있다. 에이전트의 실행상태는 항상 변화하기 때문에 에이전트 서버의 불법적인 변경행위의 공격에 매우 취약하게 된다. 때문에 에이전트 서버의 공격에 대한 에이전트의 보호는 해결하기 어려운 문제이다. 이러한 공격을 탐지하기 위해서는 에이전트의 모든 실행상태를 생성자에게 전달하고 생성자가 이것을 확인할 수 있는 방법이 필요하다. 그러므로 여러 응용에 안전하게 이동에이전트 시스템을 이용하기 위해서는 보다 효과적으로 에이전트 시스템을 보호하는 방법이 요구되고 있다.

본 논문에서는 이동에이전트 시스템의 보안문제 중 에이전트가 에이전트 서버를 이동시 서버로부터 받을 수 있는 공격에 대응할 수 있도록 순차 다중서명 방식의 하나인 'Okamoto 다중 서명 방식'[5]을 이용하여 에이전트 보안시스템을 제안하고자 한다.

Ⅱ. 이동에이전트 시스템의 보안 고려요소

1. 이동에이전트

에이전트는 "특정 목적을 수행하기 위하여 사용자를 대신하여 작업을 수행하는 자율적인 프로세스"라고 정의된다. 소프트웨어 에이전트 기술은 인공지능 분야에서 인간을 대행하는 컴퓨터 프로그램에 대한 연구분야인데, 여기서 파생된 Mobile Agent라는 것은 기하급수적으로 증가하는 분산처리 환경과 이동 컴퓨팅의 변화 때문에 주목받고 있는 기술이다. 에이전트는 수동적으로 주어진 작업을 수행하는 것이 아니고 자신의 목적을 가지고 그 목적 달성을 추구하는 능동적인 자세를 가진다. 또한 에이전트는 사람 또는 조직의 권한을 대신하여 오랜 동안 혼자 독립적으로 수행될 수 있고 다른 에이전트와 만나서 상호교류할 수 있는 프로그램이다.

이동 에이전트란 이러한 에이전트 기술에 이동코드 개념이 결합된 기술이다. 이동 에이전트는 현재 객체의 상태와 실행 가능한 코드를 포함하는 객체로서 분산환경에서 특정서비스를 제공하기 위하여 서로 통신할 필요가 있는 호스트간 직접 이동을 통하여 실행되는 객체로서 자신의 홈 플랫폼에서 출발하여 주어진 일을 완수할 때까지 네트워크상의 플랫폼들을 이동한다.

고정 에이전트(Stationary Agent)는 에이전트가 실행을 시작한 시스템에서만 실행되며 다른 시스템에 있는 에이전트나 에이전트 시스템의 정보가 필요할 때는 기본적으로 RPC(Remote Procedure Call)와 같은 통신 메커니즘을 이용한다. 반면에 이동 에이전트(Mobile Agent)란 에이전트가 수행을 시작한 시스템에 묶여있지 않은 에이전트[2]이다. 고정 에이전트(Stationary Agent)와 비교해 보면 서로 다른 시간에 서로 다른 시스템에서 수행될 수 있는 차이를 갖는다. 이동에이전트는 이질적인 네트워크에서 자신의 제어로 사이트를 이주하고 각 호스트의 다른 에이전트와 상호 동작하거나 자원을 이용하여 맡겨진 임무를 수행하고 수행이 끝날 경우 [그림 1]과 같이 홈(Home; 생성자)으로 돌아온다. 에이전트가 네트워크를 돌아다니며 수행되기 위해서는 코드와 상태정보가 이동되어야 한다. 코드는 이동될 모든 호스트에 동일한 형식으로 동작되어야 하고 직접 해석되거나 다시 컴파일 되지 않고 수행될 수 있는 이식성이 있는 중간언어가 되어야한다. 상태는

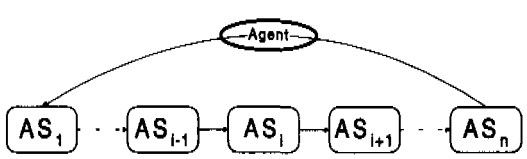

그림 1. 이동 에이전트

지속성(persistancy)이 있어야 한다. 이동 코드의 수행 위치, 중간 결과 등을 계속 유지해야하기 때문이다. 이동 에이전트 시스템의 예로서 Java를 사용한 odyssey, Aglet 시스템들이 있고 Tcl을 사용한 Agnet-Tcl 시스템이 있다.

2. 이동에이전트의 보호문제

이동에이전트 시스템은 분산 어플리케이션의 구성에 유연한 환경을 제공해 주지만 심각한 보안 문제를 야기한다. 이동 에이전트는 안전하지 못한 네트워크 채널을 통해 호스트 컴퓨터에서 다른 호스트 컴퓨터로 전송된다. 이로 인해 에이전트 코드와 데이터의 누설, 변경, 호스트 컴퓨터의 사칭, 에이전트의 전송 거부 등 보안 위협이 발생한다. 이동 에이전트는 네트워크를 통해 호스트 컴퓨터에 이동된 후 호스트 컴퓨터의 에이전트 서버에 의해 실행된다. 이것은 에이전트로부터 호스트 컴퓨터를 보호해야하는 보안 문제를 야기한다. 또한 이동 에이전트는 여러 호스트 컴퓨터를 이동하여 실행된다. 이것은 호스트 컴퓨터로부터 에이전트를 보호해야하는 보안 문제를 야기한다. [그림 2]는 이동에이전트 시스템에서 야기되는 보안문제 발생 부분을 나타낸 것이다. 이것을 나누어 살펴보면 첫째는 불완전한 통신채널을 이용하는 것에 대한 기본적인 보안 문제가 있으며, 둘째는 에이전트가 행하는 불법적인 행위나 공격에 대한 호스트 컴퓨터와 에이전트 서버의 보호문제이고, 셋째는 에이전트 서버가 행하는 에이전트공격에 대한 보안 문제이다. 이들의 세부사항은 다음과 같다.

① 안전하지 못한 네트워크채널의 보안
- 호스트 컴퓨터의 인증
- 에이전트 코드와 데이터의 기밀성
- 에이전트 코드와 데이터의 무결성
- 에이전트 재전송 방지
- 에이전트 전송의 부인방지

② 호스트 컴퓨터 보안
- 에이전트 인증
- 에이전트 호스트에 대한 접근제어

③ 이동에이전트 보안
- 에이전트 데이터의 기밀성
- 에이전트 실행상태 변화에 대한 감사

에이전트는 이동한 서버에게 모든 것이 노출되고 서버는 에이전트의 코드나 상태를 변경함으로서 에이전트의 행동을 변경/방해할 수 있다. 에이전트의 실행상태는 항상 변화하기 때문에 에이전트 서버의 불법적인 변경행위의 공격에 매우 취약하게 된다. 때문에 에이전트 서버의 공격에 대한 에이전트의 보호는 해결하기 어려운 문제이다. 이러한 공격을 탐지하기 위해서는 에이전트의 모든 실행상태를 생성자에게 전달하고 생성자가 이것을 확인할 수 있는 방법이 필요하다. 이동 에이전트의 보호를 위한 연구[6,7]가 이루어져 왔으나 이러한 방법들은 네트워크 자원을 남용하거나 실행시간이 많이 걸리고 비용이 높다는 지적을 받고 있는 실정이다. 또한 디지털서명과 감사도구를 이용한 보안구조[8]를 제안하였으나 에이전트의 실행 도중에 발생하는 변경행위에 대하여 즉각 생성자에 보고하지 않음으로서 불필요한 overhead를 갖을 수 있다. 그러므로 여러 응용에 안전하게 이동에이전트 시스템을 이용하기 위해서는 보다 효과적으로 에이전트를 보호하는 방법이 요구된다.

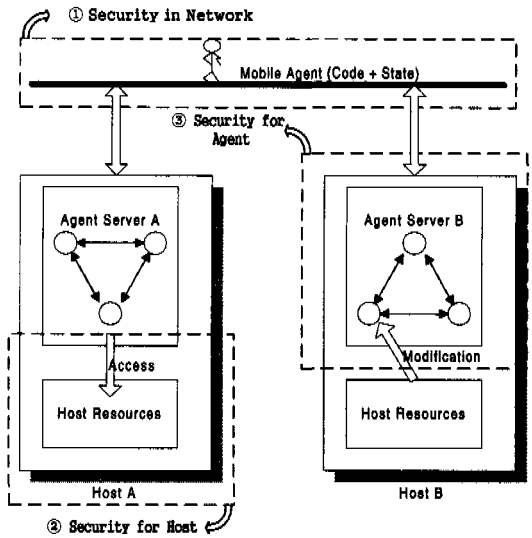

그림 2. 이동에이전트 시스템의 보안문제

Ⅲ. Digital Multi-signature

1. 디지털 서명

디지털 서명은 전자적인 정보를 이용하여 디지털 메시지에 서명하는 것으로 메시지 인증(message authentication)과 사용자 인증(user authentication)[9]을 할 수 있어야 한다. 메시지 인증이란 정보가 변경되지 않고 원래의 정보 그대로임을 보증하는 기

능이다. 사용자 인증은 사용자 A가 바로 그 A임을 증명하는 기능으로서, 사용자 A가 사용자 B와 협력하여 A가 B에게 A임을 증명할 수 있으나 제 3자인 X는 A로 위장하여 B에게 자신이 A라고 증명할 수 없고 B 또한 제 3자인 D에게 심지어는 자기자신에게도 A라고 증명할 수 없는 기능이다.

디지털 서명 방식에는 RSA 암호 시스템을 이용하는 방식, 이산대수 문제의 어려움에 근거를 둔 ElGamal 암호를 이용하는 방식, 변형된 Knapsack 문제를 이용한 방식 그리고 ID를 이용한 디지털 서명 방식 등이 있다.

서명의 바람직한 특성은 위조하기 어렵고 증명하기 쉬워야 한다. 디지털서명은 심볼들의 스트링으로 구성되며 이것은 손으로 쓴 서명과는 다르게 서명할 때마다 다르다. 이는 각 디지털서명이 서명하는 메시지의 함수가 되고 timestamp와 함께 사용함으로써 실현 가능하다. 또한 각 서명자에게 유일성을 보장하고 위조를 방지하기 위하여 각 디지털서명은 서명자에게 유일한 비밀키에 의존하는 것이 바람직하다. 검증자는 서명자의 비밀키를 알지 못하더라도 서명을 쉽게 검증할 수 있어야 한다.

공통키 암호 시스템을 이용한 디지털 서명 방법은 서명자와 검증자간의 공통인 비밀키에 의해 서명을 생성 검증하기 때문에 검증자가 서명 메시지를 변조하여 서명을 했을 경우와 같은 분쟁의 문제가 발생했을 때 분쟁을 해결하기 어려우므로 서명에 효과적이지 못하다.

송신자의 신원 확인 문제를 효율적으로 해결할 수 있는 공개키 암호 시스템에서 이루어지는 서명(signature)은 [그림 3]과 같이 표현될 수 있다.

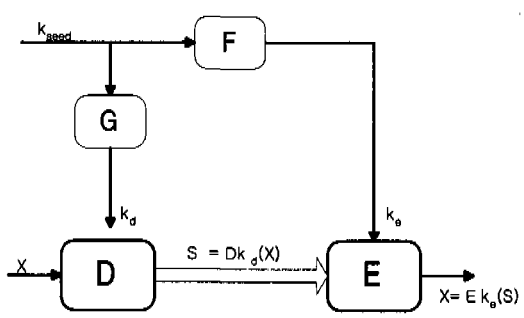

그림 3. 공개키 암호 시스템의 서명원리

위 그림에서 송신자가 메시지 X에 대하여 디지털 서명을 할 경우 송신자는 메시지 X를 자신의 비밀키 K_d와 알고리즘 D를 이용하여 서명메시지 S를 생성한다. 서명 검증자는 서명 메시지 S를 송신자의 공개키 K_e와 암호알고리즘 E를 이용하여 복호한 값이 메시지 X와 같은지를 점검함으로써 검증할 수 있으며 또한 메시지 X에 적절한 redundancy를 추가함으로써 그 메시지가 공개키 K_e에 해당하는 비밀키 K_d에 의해서 서명되었음을 검증할 수 있다.
즉, 공개키 암호 시스템이 서명에 적용되기 위해서는 $E(k_e, D(k_d, X)) = X$ 를 만족해야한다.

이와 같은 디지탈 서명의 기능은 전송되는 정보의 불법변경 여부를 판별할 수 있게 해주는 무결성(Integrity) 기능, 사용자 인증 기능, 정보의 송신이나 수신을 부인할 수 없게 하는 부인방지 (non-repudiation) 기능 등에서 필수적인 도구로서 이용되고 있다.

지금까지 디지털 서명은 한사람이 어떤 메시지에 전자적으로 서명하는 것으로써 단순 서명(single signature)이다. 그러나 대부분의 사무실에서는 계층적인 구조를 가지며 사무실에서 작성한 문서는 기안자 뿐 아니라 상급자들의 서명이 요구된다. 또한 원격회의를 통해 회의결과를 전자문서로 작성하고 최종적으로 회의 참석자들의 동의를 얻어야할 때 참석자들의 서명이 요구된다. 이 경우 단순 서명을 반복해서 적용하여 문제를 해결할 수 있지만, 서명의 길이가 늘어나고, 서명을 검증하려면 서명자의 수만큼 검증과정을 거쳐야 하기 때문에 서명자가 많은 경우 시간이 오래 걸린다는 단점이 있다. 이러한 단순 서명 방식의 문제를 해결하기 위해 나온 개념이 다중 서명방식이다. 즉, 동일한 메시지에 대해 여러사람이 전자적으로 서명하는 것을 디지털 다중서명(Digital Multi-signature)이라 한다.

기존의 디지털 다중 서명 방식으로는 두 개의 큰 소수와 각 서명자의 직위에 따른작은 소수의 곱을 이용하여 RSA 방법을 확대 적용한 Itakura-Nakamura방법, RSA 방식과 같은 전단사 공개키 암호 시스템과 단방향함수를 이용한 Okamoto방법, Fiat-Shmir 서명방식에 근거하여 만들어진 Brickell-Lee-Yacobi방법, Ohta-Okamoto 방법[10] 등이 연구되어있다.

이 중 RSA 공개키 방식으로 운용되며 서명순서에 제약이 없고, 중간서명자가 검증이 가능한 Okamoto 방법을 본 논문에 적용한다.

2. Okamoto 디지털 다중 서명 방법

Okamoto방법은 RSA와 같은 전단사 공개키 암호 시스템과 단방향 해쉬함수를 이용한 다중 서명 방

식이다.

본 논문에서 사용될 기호는 [표 1]과 같이 정의한다.

Okamoto방법은 서명자 i가 공개키 e_i와 비밀키 d_i를 생성하여 공개키인 e_i와 단방향 해쉬함수 $h_i : X_i X_i X_i \ldots X_i \rightarrow X_i$를 공개하고 비밀키 d_i를 보관한다. 서명작업을 위해 첫번째 서명자는 $S_1 = D_{d_1}(h_1(M))$, $M_1 = M$를 수행한다. 그리하여 이 서명메시지 (S_1, M_1)과 자신의 식별자 ID_1을 다음 서명자에게 전송한다. n번째 서명자는 (S_{n-1}, M_{n-1})를 수신하여 앞 서명자의 서명 메시지를 다중 서명 검증식에 의해 검증하고 앞 서명자의 서명메시지에 자신의 서명을 다음과 같이 수행한다.

표 1. 수식 표기 기호

기 호	정 의
M	서명할 메시지
h	공개된 단방향 함수
Ee_i	키 ei에 의한 공개키 암호함수
Dd_i	키 di에 의한 공개키 복호함수
|N|	N의 비트길이
$[S]^L$	S의 (|S|-L)개의 최상위 비트. 즉, |[S]L| = |S| - L이다
$[S]_L$	S의 L개의 최하위 비트 즉, |[S]L| = L 이다
$^L\{S\}$	상위 (L-|S|)개의 '0'비트 패딩을 갖는 S. 즉, |L{S}| = L이다.
||	연접(concatenation)
ID_i	서명자 i의 ID

만약 $|X_n| > |X_{n-1}|$ 이면

$$S_n = D_{d_n}(^{|X_n|}\{S_{n-1}\}),\ M_n = M_{n-1}$$

그렇지 않으면

$$S_n = D_{d_n}(^{|X_n|}\{[S_{n-1}]_{|X_n|-1}\})$$

$$M_n = M_{n-1} \| [S_{n-1}]^{|x_n|-1}$$

여기서 Xn은 서명자 n의 평문과 암호문의 유한 집합을 나타낸다.

2.1 다중서명 검증식

검증자는 공개키 e_i (i=1, 2, ..., m)을 이용하여 다중 서명 메시지 (S_m, M_m)을 점검한다. 여기서 서명자의 순서는 서명메시지에 첨부된 서명자의 식별자 $(ID_1, \ldots ..ID_m)$에 의하여 표시된다.

(1) 다음 식에 의해서 M_i'와 S_i'(i=1, 2, ..., m)을 구한다. 여기서 $M_m' = M_m$ 이고 $S_m' = S_m$이다.

만약 $|X_i| > |X_{i-1}|$ 이면

$$S_{i-1}' = [E_{e_i}(S_i')]_{|X_{i-1}|} \quad M_{i-1}' = M_i'$$

그렇지 않으면

$$S_{i-1}' = [M_i']_{|X_{i-1}|-|X_i|+1} \| [E_{e_i}(S_i')]_{|X_i|-1}$$

$$M_{i-1}' = [M_i']^{|X_{i-1}|-|X_i|+1}$$

(2) 위의 단계 1에서 얻은 S_1' 와 M_1'가 다음식을 만족하면 다중 서명 메시지 (S_m, M_m)는 유효한 것으로 간주한다.

$$E_{e_1}(S_1') = h_1(M_1')$$

이와 같은 Okamoto 다중서명 방식은Itakura - Nakamura 다중 서명 방식의 서명순서가 제약받는다는 단점을 개선하였으며 다중 서명 길이는 단순 서명 메시지의 길이와 거의 같다. 또한 단방향 해쉬함수를 사용함으로써 다중 서명 발생 및 검증을 효율적으로 처리할 수 있으며 RSA 뿐만아니라 어떠한 전단사 공개키 암호시스템으로도 구성할 수 있다는 장점이 있다. 본 논문에서는 이와 같은 다중서명 기법을 이동에이전트의 보안 문제에 적용하여 새로운 에이전트 보안 구조를 제안한다.

Ⅳ. 다중서명기법을 이용한 보안 이동에이전트의 설계

에이전트가 생성자에 의해 생성된 뒤 에이전트 서버간을 이동하는 도중에 에이전트 자신 혹은 생성자외의 에이전트 서버는 코드를 변경할 수 없다. 또한 이동에이전트 코드는 모든 호스트에서 동일한 형식으로 동작되어야 하고, 직접 해석되거나 다시 컴파일 되지 않고 수행될 수 있는 이식성이 있는 중간언어이어야 한다. 인터프리터는 에이전트의 호스트 자원에 대한 접근을 제어하면서 에이전트를 실행시킨다. 에이전트 서버로부터 받은 에이전트의 코드와 상태에 의해 인터프리터는 에이전트를 실행시킨다. 본 논문의 보안구조시스템은 에이전트를 구동시키는 인터프리터가 신뢰성 있게 구현되어 에이전트 서버에서 구동되고 있다고 가정한다.

에이전트는 실행 코드와 실행 상태를 가진다. 실행 코드는 에이전트 생성시 만들어지고 소멸할 때까지 변하지 않는다. 따라서 에이전트 실행 코드는 생성자의 디지털 서명을 통해서 코드가 변경되었는지에 대한 확인을 할 수 있다. 실행 상태는 에이전

트가 실행되면서 계속 변화하기 때문에 각 에이전트 서버는 에이전트 실행 상태에 대한 디지털 서명을 함으로써 에이전트 실행 상태 변경에 대한 부인봉쇄를 이룰 수 있다.

1. Notation

AS_i : i번째 에이전트 서버의 ID
$Info_i$: AS_i에서 에이전트의 실행 과정상태정보
C_i : 에이전트가 경유한 i번째 에이전트 서버까지의 갯수
h_i : AS_i의 공개된 단방향 함수
Ee_i : 키 e_i에 의한 공개키 암호함수(AS_i의 공개키)
Dd_i : 키 d_i에 의한 공개키 복호함수(AS_i의 비밀키)
RS_n : 수신 부인봉쇄를 위한 n번째 에이전트 서버의 응답
X_n : 서명자 n의 평문과 암호문의 유한 집합
$|N|$: N의 비트길이
$[S]^L$: S의 ($|S|$-L)개의 최상위 비트. $|[S]^L| = |S| - L$
$[S]_L$: S의 L개의 최하위 비트. $|[S]_L| = L$
$^L\{S\}$: 상위 (L-$|S|$)개의 '0'비트 패딩을 갖는 S. $|^L\{S\}|=L$
$\|$: 연접(concatenation)

2. 알고리즘

(1) 에이전트 서버의 등록

에이전트 서버 i는 RSA 알고리즘에 의해 공개키 e_i와 비밀키 d_i를 생성한 후, 공개키 e_i와 단방향 해쉬함수 $h_i : X_i X_i X_i \ldots X_i \rightarrow X_i$를 공개하고 비밀키 d_i를 자신이 보관한다. 또한 자신의 ID인 AS_i를 Directory server에 등록한다.

$AS_i \rightarrow$ Directory server:e_i, h_i, AS_i

(2) AS_1의 서명발생

에이전트 AS_1에 있었다는 사실을 부인하지 못하도록 다음과 같이 디지털 서명한다. 만일 각각의 에이전트 서버에서의 에이전트 실행과정의 상태를 다른 에이전트 서버에게 노출되지 않고 에이전트 생성자만이 확인할 수 있도록 해야하는 경우에는 이 에이전트 서버의 실행과정의 상태정보를 에이전트 생성자의 공개키로 암호화한다

$AS_1 \rightarrow AS_2 : E_{e_2}(S_1, M_1),\ E_{e_2}(AS_1),$
where $S_1 : D_{d_1}(h_1(M_1)),\ M_1 : Info_1 \| C_1$

(3) AS_2의 검증

AS_1으로부터 받은 (S_1, M_1)이 다음 식을 만족하면 AS_1의 서명 메시지가 유효한 것으로 간주한다.

$$E_{e_1}(S_1) = h_1(M_1)$$

AS_1의 서명 메시지가 유효한 것으로 검증이 되면, AS_2는 AS_1에게 자신이 AS_1의 서명과 메시지를 받았다는 응답을 자신의 비밀키로 서명하여 전송한다.

$AS_2 \rightarrow AS_1 : E_{e_1}(RS_2)$
$RS_2 : D_{d_2}(S_1, M_1, \text{'}Received\text{'})$

(4) AS_2의 서명

AS_2는 에이전트를 실행하여 얻은 상태정보에 다음과 같이 서명한다.
C_2는 AS_1으로부터 넘겨받은 에이전트 서버의 ID갯수에 1을 더하여 구하며, 이것은 에이전트를 실행한 에이전트 서버가 이전 에이전트 서버들에서 실행된 상태정보와 에이전트 식별자를 제거하는 것을 체크하기 위한 카운트이다.

$AS_2 \rightarrow AS_3 : E_{e_3}(S_2, M_2),\ E_{e_3}(AS_1, AS_2)$

만일 $|X_2| > |X_1|$ 이면
$S_2 : D_{d_2}(^{|X_2|}\{S_1\} \| h_2(Info_2 \| C_2))$
$M_2 : M_1 \| Info_2 \| C_2,\ C_2 : C_1 + 1$
그렇지 않으면
$S_2 : D_{d_2}(^{|X_2|}\{[S_1]_{|X_2|-1}\} \| h_2(Info_2 \| C_2))$
$M_2 : M_1 \| [S_1]^{|X_2|-1} \| Info_2 \| C_2$
$C_2 : C_1 + 1$

(5) 다중서명 검증

n번째 에이전트 서버는 (S_{n-1}, M_{n-1})을 수신하면 다음과 같이 검증한다. 검증과정에서 문제가 발생하는 경우 즉각 홈으로 통보함으로써 에이전트의 실행을 중지할 수 있도록 한다. 공개키 e_i(i=1,2,....,n-1)를 이용하여 다중서명 메시지를 검검한다. 에이전트 서버의 순서는 서명메시지에 첨부된 에이전트 서버의 식별자(AS_1,....,AS_{n-1})에 의하여 표시된다. 각각의 S_i'와 M_i'가 검증식을 만족하면 i번째 에이전트 서버의 서명과 실행상태 정보에 대한 무결성이 증명되며 다중서명 메시지는 유효 한 것으로 간주한다.

여기서 $M_{n-1}' = M_{n-1}$ 이고 $S_{n-1}' = S_{n-1}$이다.

$$[E_{e_i}(S_i')]_{|X_i|} = h_i([M_i']_{|(Info_i \| C_i)|})$$

만일 $|X_i| > |X_{i-1}|$ 이면
$S_{i-1}' : [[E_{e_i}(S_i')]^{|X_i|}]_{|X_{i-1}|}$

$M_{i-1}' : [M_i']^{|(Info_i \| C_i)|}$

그렇지 않으면

$S_{i-1}' : [[M_i']^{|(Info_i \| C_i)|}]_{|X_{i-1}|-|X_i|+1} \| [[E_{e_i}(S_i')]^{|X_i|}]_{|X_i|-1}$

$M_{i-1}' : [[M_i']^{|(Info_i \| C_i)|}]^{|X_{i-1}|-|X_i|+1}$

에이전트를 수신 받은 AS_n이 에이전트를 정지시키는 공격행위에 대하여 AS_n이 에이전트를 받았다는 증명을 AS_{n-1}에게 제공하도록 함으로써 에이전트 turnarround가 없을 경우 문제를 발생시킨 AS를 찾도록한다. 즉, 에이전트를 보낸 AS_{n-1}은 AS_n으로부터 다음과 같은 RS_n을 받아두도록 함으로써 가능하며, 수신메세지를 받지 못하는 경우는 홈에게 AS_n이 문제가 있음을 알리도록한다.

$AS_n \rightarrow AS_{n-1} : E_{e_{n-1}}(RS_n)$

$RS_n : D_{d_n}(S_{n-1}, M_{n-1}, 'Received')$

(6) AS_n의 서명

$AS_n \rightarrow AS_{n+1} : E_{e_{n+1}}(S_n, M_n)$,

$E_{e_2}(AS_1, AS_2, \ldots, AS_n)$

만일 $|X_n| > |X_{n-1}|$ 이면

$S_n : D_{d_n}(^{|X_n|}\{S_{n-1}\} \| h_n(Info_n \| C_n))$

$M_n : M_{n-1} \| Info_n \| C_n$

$C_n : C_{n-1} + 1$

그렇지 않으면

$S_n : D_{d_n}(^{|X_n|}\{[S_{n-1}]_{|X_n|-1}\} \| h_n(Info_n \| C_n))$

$M_n : M_{n-1} \| [S_{n-1}]^{|x_n|-1} \| Info_n \| C_n$

$C_n : C_{n-1} + 1$

만약 서명이 마지막 서버에서 이루어졌다면 (S_n, M_n)과 (AS_1, AS_2,...AS_n)을 홈으로 보낸다.

3. 제안된 이동 에이전트 보안 시스템의 분석

에이전트를 실행한 각각의 에이전트 서버는 자신의 서명, 에이전트 실행 상태 정보, 에이전트 이동 경로를 다음 에이전트 서버의 공개키로 암호화하여 전송한다.

에이전트 서버는 이전 에이전트 서버로부터 받은 서명과 에이전트 실행 상태 정보를 자신의 비밀키로 복호화한 후 검증 절차를 거친다.

각각의 에이전트 서버는 에이전트 실행 상태 정보를 해쉬함수를 이용하여 메시지 인증 코드로서 사용한다. 따라서 이전 에이전트 서버로부터 받은 서명과 실행 상태 정보를 검증하는데 있어서 얻어낸 각각의 에이전트 서버의 서명을 공개키로 복호화 한 값과 각각의 에이전트 서버의 실행 상태 정보를 공개된 해쉬함수를 이용한 결과와 비교하여 같은 값일 경우 메시지 즉, 실행 상태 정보의 무결성이 검증되는 것이고 그렇지 않을 경우는 위조 또는 변조되었음을 알 수 있다.

각 에이전트 서버의 서명과 실행 상태 정보는 timestamp를 추가함으로써 재전송을 방지할 수 있고, 각 에이전트 서버는 이전 에이전트 서버로부터 받은 서명과 메시지를 검증한 후 받은 서명과 메시지를 자신의 비밀키로 암호화하여 이전 서버에게 돌려줌으로써 수신 부인 봉쇄를 이룰 수 있다. 제안된 이동 에이전트 보안시스템은 Okamoto 디지털 다중 서명 방법을 변형하여 이전 에이전트 서버의 서명에 각 에이전트 서버에서 실행된 에이전트의 실행 상태 정보를 해쉬함수를 이용하여 메시지 인증 코드로서 추가한 후 서명하였다. 제안된 이동 에이전트 보안 시스템은 서명 순서에 제약이 없고, 중간 서명자가 검증이 가능하다.

V. 결 론

본 논문에서는 최근 많은 연구가 진행되고 있는 에이전트 시스템 중에서 이동 에이전트 시스템에 대하여 살펴보고 이동 에이전트 시스템에서 가장 큰 문제로 지적되고 있는 보안문제에 대하여 기술되었다. 이동 에이전트 시스템에서 야기될 수 있는 보안 문제 중 이동 에이전트가 이동 중 또는 다른 호스트로 이동 한 후 악의적인 호스트로부터 받을 수 있는 공격에 대한 보안으로 공개키 기술에 기반을 둔 디지털 다중서명 방법을 적용하였다. 제안된 이동 에이전트 보안 구조의 장점은 다음과 같다.

첫째, 각 에이전트 서버에 의한 디지털 다중 서명을 이용하여 에이전트 실행 상태 변화를 각 서버마다 검사토록 함으로써 문제 발생시에 즉각 발견할 수 있도록 하였다.

둘째, 사용된 다중 서명은 Okamoto 다중 서명 방식에 근거하므로, 서명순서의 제약이 없다. 즉, 이동 에이전트의 경로 배정 문제에 있어서 이동 에이전트의 생성자가 미리 경로를 배정하여 이동 에이전트는 그 경로를 따라서만 이동하는 경우 뿐만 아니라, 이동 에이전트가 갖는 기본적인 특성인 자율

성과 지능을 바탕으로 이동 에이전트가 직접 이동할 에이전트 서버를 결정하여 이동하는 경우에도 적용될 수 있다.

셋째, 공개키 암호시스템과 단방향 해쉬 함수를 사용함으로써 정보를 보호하는데 기본적으로 제공되어야 할 기밀성, 인증, 무결성, 부인봉쇄 등의 기능을 제공할 수 있다.

그러나 기본적으로 적용된 전단사 공개키 암호시스템과 일방향함수(one-way function)를 이용한 Okamoto 방법이 RSA방법에 기초하기 때문에 서명시 계산량이 많아져 처리 속도가 느리다는 단점이 있다. 향후 좀 더 수행 능력이 뛰어난 방법으로 실행 상태정보에 대한 보안 문제를 해결하는 구조를 연구하고자 한다.

참 고 문 헌

[1] 이재호, "에이전트 시스템의 연구 및 개발동향", 정보과학회지, 제18권, 제5호, pp.4~9, 2000

[2] 한승완, 임형석, "이동 에이전트 기반 전자상거래 시스템에서의 안전한 이동 에이전트 전송 프로토콜", 정보처리학회지, 제7권, 제5S호, pp.1657~1675, 2000

[3] 김평중, 윤석환, "이동 컴퓨팅을 위한 이동 에이전트 시스템", 정보처리학회지, 제4권, 제5호, pp.67~75, 1997

[4] David Chess, Benjamin Grosof,Colin Harrison, David Levine,Colin Paris, Gene Tsudik, "Itinerant Agents for Mobile Computing"/ Technical Report, IBM Research Division-T.J. Watson Research Center, 1995.

[5] T.Okamoto, "A Digital Multisignature Scheme Using Bijective Public-key Crypto-systems", ACM Trans. on Comp. Systems Vol 6, No 8, pp.432~441, 1988.

[6] Bennet S.Yee, "A Sanctuary for Mobile Agents", DARPA Workshop on Foundations for Secure Mobile Code Workshop, pp.26~28, 1997.

[7] Giovanni Vigna, "Protecting Mobile Agents through Tracing", ECOOP Workshop, 1997.

[8] 백주성, 이동익, "디지털 서명과 감사 도구(Audit trail)를 이용한 이동 에이전트의 보호", 한국정보과학회 학술발표논문집, 제 24권, 제2호, pp.419~422, 1997.

[9] 현대암호학, 한국전자통신연구소, pp.157~170, 1991.

[10] 강창구, "디지탈 다중서명 방식과 응용에 관한 연구", 공학박사학위논문, 충남대학교, pp.41~43, 1993.

김 경(Kyung Kim) 준회원

1996년 : 조선대학교 전산학과 졸업(이학사)

2000년 : 조선대학교 전산학과 졸업(이학석사)

1996년~2000년 : 삼성화재 사원

<주관심 분야> 전자상거래, 네트워크 보안

배 용 근(Yong-Guen Bae) 정회원

1984년 : 조선대학교 컴퓨터공학과 졸업(공학사)

1986년 : 조선대학교 컴퓨터공학과 졸업(공학석사)

1988년~현재 : 조선대학교 컴퓨터공학부 조교수

<주관심 분야> 전자상거래, 병렬처리, 마이크로 프로세서 응용, 멀티미디어

정 일 용(Il-Yong Chung) 정회원

1983년 : 한양대학교 공과대학 졸업(공학사)

1987년 : City University of New York 전산학과 (전산학석사)

1991년 : City University of New York 전산학과 (전산학박사)

1991년~94년 : 한국전자통신연구소 선임연구원

1994년~현재 : 조선대학교 컴퓨터공학부 부교수

<주관심 분야> 네트워크 보안, 전자상거래, 분산시스템 관리, 코딩이론, 병렬 알고리즘