제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003. 5)

JAVA 기반의 이동 에이전트 보안 모델에 의한 전자서명 메카니즘

최길환 · 신민화 · 이대영 · 배상현
조선대학교 전산통계학과
e-mail:ckhplc@hanmir.com

Digital Signature Mechanism by Mobile Agent Security model of JAVA base

Kil-Hwan Choi* · Min-Hwa Shin · Dae-Young Lee · Sang-Hyun Bae
Dept of Computer Science and Statistics, Chosun University

요 약

현재 이동 에이전트 및 이의 전송과 실행을 위한 이동 에이전트 시스템의 구현에 Java가 많이 사용되고 있지만 Java의 기본적인 보안 모델은 이동 코드의 기능 확장성에 제한을 주는 문제점이 있다. 본 논문에서는 전자서명을 Java 기반의 이동 에이전트에 적용함으로써 시스템의 보안을 유지하면서 이동 에이전트의 기능 확장성을 보장할 수 있음을 보였다.

또한 이동 에이전트의 전자서명과 이의 망 관리로의 적용을 위해서는 망 관리국(NMS:Network Management Station)에서의 서명자 관리나 공개키 관리 등 전자서명과 관련된 기능 외에 망 관리 이동 에이전트의 등록과 전송, 실행 및 실행 관리를 위한 구성이 필요하다. 따라서 이를 위해 작성한 망 관리국과 관리 대상 시스템의 구성 모델을 보였으며, 제안한 구성방식의 동작을 검증하기 위해 망 관리 응용 예를 작성하고 평가하였다.

제안한 구성 방식을 사용하면 전자서명 처리로 인한 속도 저하의 문제가 있지만, 이동 에이전트의 사용으로 인해 얻어지는 부하 분산과 실시간 관리, 망 확장성 증대의 장점 이외에도 관리 기능 및 서비스 추가가 용이한 장점이 있다.

1. 서론

신속하고 다양한 정보통신 서비스에 의존하는 정보화 사회로의 진행이 가속되면서 유무선망의 구분 없는 개인 통신 서비스나 음성과 데이터 및 화상정보가 통합되는 새로운 통신 서비스가 요구되고 있다. 이러한 서비스를 지원하기 위한 통신망의 규모가 커지고 복잡성이 증가하고 있으며, 망 요소 간 또는 망 간 이질성이 증대됨에 따라 망 요소들의 효율적인 관리가 필요하다.

현재까지 사용되는 대부분의 모델은 분산 시스템의 클라이언트/서버(Client/Server) 구조에 기반을 둔 매니저/에이전트(Manager/Agent) 구조를 채택하고 있다. 이는 관리 기능을 가지고 있는 NMS(Network Management Station)와 관리 대상인 NE(Network Element)들이 망 관리 프로토콜(Network Management Protocol)을 이용하여 망 관리에 필요한 정보를 교환하고 처리하는 구조를 가진 중앙집중형(Centralized Network Management) 모델이다[2]. 인터넷의 망 관리 표준인 IETF의 SNMP(Simple Network Management Protocol[3])와 OSI에서의 망 관리 표준인 CMIP(Common Management Information Protocol[4])에서 이 모델이 사용되고 있다.

중앙집중형 망 관리 모델은 망 구성 및 관리가 단순하고 보안이 용이하다. 그러나, NMS에 모든 관리 기능이 집중되기 때문에 망 확장성과 실시간 관리 기능이 제한되고, 서비스의 동적인 추가와 망 통합관리가 어렵다는 단점이 있다[5].

이와 같은 중앙집중형 망 관리 모델의 문제점을 보완하기 위하여 망 관리에 이동 에이전트(Mobile Agent) 기술을 적용하려는 연구가 진행중이다. 이는 이동 코드(Mobile Code) 또는 이동 에이전트라 불리는 실행 가능한 코드 자체가 네트웍을 통해 해당 시스템으로 전송되어 실행되는 방식으로서, 현재 대부분의 분산 환경을 이루고 있는 클라이언트/서버 기술인 RPC(Remote Procedure Call)나 메시지 교환과는 대조되는 개념이다[6].

이동 에이전트 기술을 망 관리에 적용하면 관리 기능을 NE들로 분산시킴으로써 중앙집중형 관리 모델에서 발생되는 NMS의 부하 집중 문제를 해결할 수 있다. 또한 관리 작업의 결과가 각 NE에서 처리되어 NMS로 전송되므로 NMS와 NE간의 빈번한 정보교환으로 인하여 발생하는 네트웍 부하를 감소시킨다. 또한, 관리 기능을 동적으로 추가하는 것이 용이하며, 이기종 망간의 관리 연동 등의 장점이 있다[7,8]

2. Java 기반의 이동 에이전트 시스템

이동 에이전트가 네트웍을 이동하면서 기능을 수행하기 위해서는 각 플랫폼에 에이전트 실행 엔진(Agent Execution Engine) 또는 에이전트 실행 환경(Agent Execution Environment)이라 불리우는 이동 에이전트를 실행시키는 기능을 가진 모듈이 필요하다.

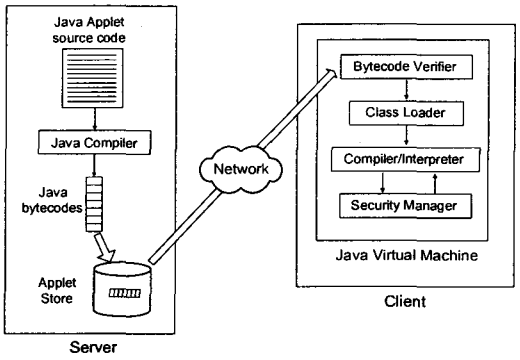
객체 지향적이고 분산 환경에 적합한 네트웍 언어이며 플랫폼에 무관한 이식성 등의 특성 때문에 최근들어 이동 에이전트 시스템 구현에 Java가 많이 사용되고 있다[1,8,9].

본 논문에서도 Java 언어를 이용하여 이동 에이전트 및 실행환경을 구현하였다. 다음에 이동 에이전트와 이의 실행 환경 구성에 Java를 사용할 때Java의 보안 모델로 인해 발생되는 문제점을 기술하였다.

2.1 Java 보안 모델

Java 코드는 분산환경에서 수행되도록 설계되었다. 따라서 네트웍을 통해서 전달된 Java 코드를 수행하는 클라이언트 시스템의 안전성을 보장하는 것은 중요한 사항이다. 특히 이동 에이전트는 네트웍을 통해서 플랫폼간을 이동하면서 실행되기 때문에 적절한 제약이 가해지지 않는다면 시스템의 보안에 많은 문제를 야기 시킬 수 있다.

(그림1)에 Java 애플릿(Applet)의 생성과 전달, 보안사항의 적용 및 수행 과정을 나타내었다.



(그림 1) Java 코드의 생성과 전달, Java 보안 모델

2.2 전자서명과 Signed Applet

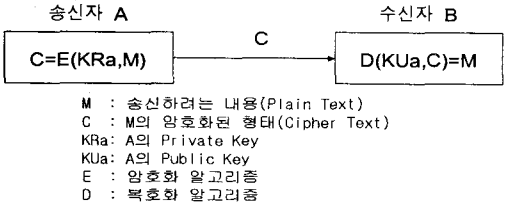
시스템의 보안을 만족시키고 이동 에이전트 기능 확장의 어려움을 극복하기 위해 본 논문에서는 전자서명(Digital Signature) 기술을 이용한다. 망 관리 이동 에이전트를 전자 서명하여 전송하면, 이동 에이전트 엔진이 있는 관리 대상 시스템에서 인증 확인 작업을 거친 후 해당 에이전트에 대해 파일 시스템과 네트웍 접근 제한을 해제하는 것이다. 이러한 방식을 사용하면 보안을 만족시키면서, 망 관리 이동 에이전트의 기능 제한을 극복할 수 있다.

2.2.1 전자서명

전자서명(Digital Signature)은 공개키(Public Key) 방식을 사용하여 사용자 인증과 메시지 불변성을 보장 해주는 기술이다. 공개키 방식이란 사용자마다 자신만이 알고 있는 Private Key와 이에 대응되면서 다른 사람들에게 알려줘야 하는 Public key, 한 쌍의 키들을 암호화와 복호화에 사용하는 것이다. 이 때 Private Key를 사용하여 메시지를 암호화한 경우 Public Key를 사용하여야 해독할 수 있고, 반대로 Public Key를 사용하여 암호화 한 경우는 대응되는 Private Key를 사용하여야 해독 가능하다.

전송하고자 하는 상대의 Public Key로 메시지를 암호화 한 경우에는 원하는 상대만이 자신(수신자)의 Private Key를 사용하여 해독할 수 있으므로 메시지 내용의 기밀성이 필요한 경우에 사용된다.

반대의 경우, 즉 자신의 Private Key로 암호화 한 경우에는 공개되는 Public Key로 누구나 해독할 수 있으므로 내용의 기밀성 보다는 메시지의 작성자를 인증하고 메시지 내용의 불변성을 인증하는 데에 사용되며, 이를 전자 서명이라 한다. (그림2)에서는 송신자 A가 자신의 Private Key(KRa)를 사용하여 암호화 알고리즘 E를 통해 원문 M을 암호화된 형태인 C(Cyphertext)로 변환 후 전송한 것을 나타낸다.

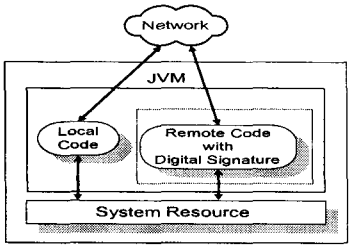


(그림 2) 전자서명 메커니즘

2.2.2 Signed-Applet

Java 모델에서는 네트웍을 통해 이동 가능한 Java 코드를 애플릿(Applet)이라 하며, 이에 대해 앞에서 언급한 Sandbox라 불리는 제한이 적용된다. Signed-Applet이란 JDK 1.1부터 지원되는 것으로써 전자서명을 사용하여 인증된 애플릿을 의미한다. 전자서명 된 Java 애플릿은 이동 후 실행 환경에서 검증과정을 거친 후 Sandbox의 제한을 벗어나서, 마치 Local에서 실행되는 것처럼 파일 시스템과 네트웍 액세스를 자유로이 할 수 있다[10].

Sandbox의 제한에서 벗어나서 실행되는 Signed-Applet을 (그림 3)에서 보였다.



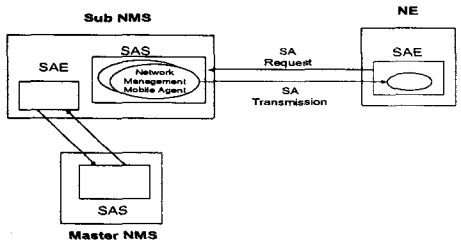
(그림 3) Signed Applet의 실행

3. 망 관리 이동 에이전트 시스템

3.1 전자 서명된 망 관리 응용 이동 에이전트 구성

본 논문에서는 망 관리 기능을 가지며 전자 서명된 이동 에이전트를 SA(Signed Agent)라 한다. SA의 유지, 전달 및 실행을 위해 SAS(Signed Agent Server)와 SAE(Signed Agent Executor), 두개의 기능 모듈을 구성하였다

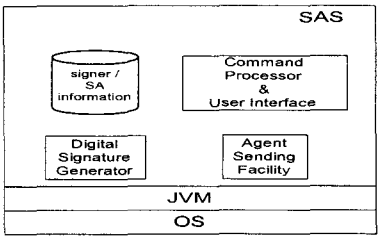
(그림 4)에서 Master NMS 자신은 관리 대상이 아니므로 망 관리 SA들을 유지하는 SAS만을 수행하며, NE는 관리 대상으로서의 역할만을 하므로 SAE만을 실행한다. Sub Master는 자신이 NMS의 역할을 수행하면서 또한 관리 대상일 수 있으므로 SAE와 SAS를 둘 다 실행하고 있다.



(그림 4) 이동 에이전트 시스템 구성

3.2 SAS (Signed Agent Server)

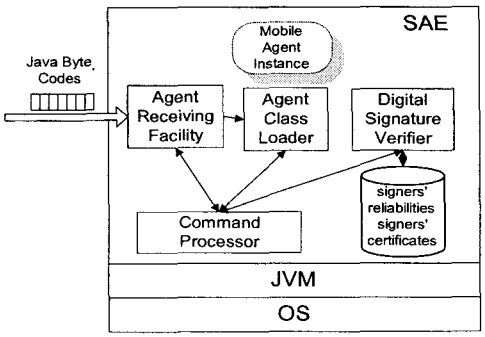
SAS의 기능 모듈을 (그림 5)과 같다. SA를 SAE에 전송하는 기능을 수행하는 Agent Sending Facility와 전자서명을 생성하는 Digital Signature Generator를 사용하여 Command Processor는 서명자 관리, SA 정보 관리, Signed-Agent 수행 관리를 제공한다.



(그림 5) SAS의 구성

3.3 SAE (Signed Agent Executor)

SAE는 관리 대상에서 Daemon 형태로 동작하면서 자신이 요구했거나, SAS에서 전달한 SA에 대해 전자서명을 검사하고 실행시키는 역할을 담당한다. SAE는 관리 대상에서 실행되므로 사용자 인터페이스는 가지지 않는다. 또한 현재 구현된 SAE는 전자서명 된 이동 에이전트를 수신하여 실행시키는 역할만을 하며, 코드 이동이나 에이전트간 통신은 지원하지 않는다. (그림 6) SAE의 구성 모듈을 나타내었다.



(그림 6) SAE의 구성

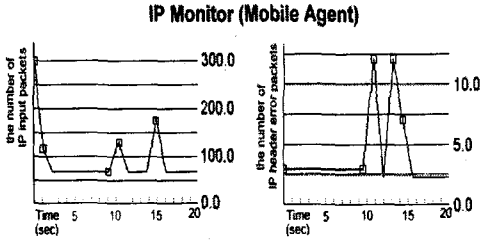
4. 실험 결과 및 평가

(그림 7)-(a) 서버 응용이 표시하는 화면이다. 그래프의 X축은 한 눈금 당 1초씩 시간의 경과를 나타내며, Y축은 해당 시점에서의 각 MIB 값의 증가값, 즉 단위 시간(1초)당 IP 수신 패킷의 수와 오류 패킷의 수를 나타낸다. 서버 응용은 최근 20초 동안에 클라이언트 응용에게서 ipInReceives 값에 대해 5 개의 메시지 이벤트 4건, 평균값 1건을, ipInHdrErrors 값에 대해서는 5 개의 메시지 이벤트 3건, 평균값 2건을 수신했음을 알 수 있다. 따라서 20초 동안에 총 10개의 메시지가 교환되었다.

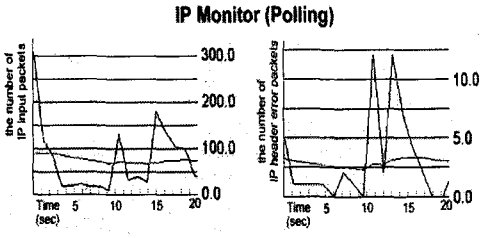
(그림 7)-(b) 비교를 위해 전통적인 SNMP 매니저/에이전트 구조에서 Polling 방식을 사용한 실험

결과이다. 이 방식에서 매니저는 1초에 한 번씩 ipInReceives와 ipInHdrErrors 값을 요구하는 SNMP PDU들을 SNMP 에이전트에게 전달하고, 관리 대상에 있는 SNMP 에이전트는 이에 응답하게 된다. 매니저 프로그램은 수신한 MIB 값들과 평균값을 표시한다. 이 방식에서는 1초당 2개의 요구 SNMP PDU들과 응답 SNMP PDU들이 발생하게 된다. 따라서 20초 동안에 총 80개의 SNMP PDU가 발생하게 된다.

Polling 간격이나 평균값의 보고 간격 및 이벤트의 정의에 따라 차이가 있겠지만, 이동 에이전트 모델에서는 관리기능의 많은 부분을 관리 대상에서 수행하므로 실시간 관리가 가능하다. 또한 관리 정보에 대한 처리가 NE에서 수행되어서 추상화 된 정보의 형태로 NMS로 전달되기 때문에 NMS의 부하를 분산시키고 교환하는 정보의 양을 줄일 수 있다.



(a) 이동 에이전트 방식



(b) Polling방식

(그림 7) IP 상태 감시 응용 결과

5. 결론 및 향후 과제

일반적인 망 관리 모델은 클라이언트/서버 구조에 근거한 중앙집중형으로서, 망 확장성이나 실시간 관리 및 동적인 서비스 추가에 있어서 단점을 가지고 있다.

본 논문에서는 전자 서명 된 망 관리 이동 에이전트를 위한 시스템을 구성함으로써 보안을 유지하고, 망 관리 응용의 기능 제한을 극복했다. 또한 망 관리 이동 에이전트의 동작 실례를 보이고 성능을 평가함으로써 제안한 시스템의 동작을 검증하고, 이동 에이전트를 이용한 망 관리 모델의 장점을 보였다.

현재에는 망 관리를 위한 이동 에이전트를 일반적인 Java 애플릿 형태로 구성하였지만, 망 관리에 적합한 형태의 망 관리 이동 에이전트의 구조를 고안하는 것이 바람직 할 것이다. 또한 망 관리 이동 에이전트의 실행 엔진 역할을 하는 SAE는 이동 에이전트 코드 및 이동 에이전트가 수집한 데이터를 다른 곳으로 이동시킬 수 있는 기능과 에이전트간 통신 기능이 구현되지 않았으며, 이 역시 망 관리에 적합한 형태로 설계되어야 한다.

참고문헌

[1] 최길환, 배상현, 자바 기반의 이동 에이전트 보안 구조 설계와 암호기능 구현, 한국인터넷정보학회 논문집, Vol.3, No.1, pp61-69, 2002.2.

[2] Roch H. Glitho and Stephen Hayes, Telecommunications Management Network: Vision vs. Reality, IEEE Communication Magazine, March 1995.

[3] J. Case, M. Schoffstall, J. Davin, "A Simple Network Management Protocol (SNMP), RFC 1157, May 1990.

[4] Y. Yemini, The OSI Network Management Model, IEEE Communication Magazine, May 1993.

[5] German Goldszmidt, Yechiam Yemini, "Distributed Management by Delegation, In the 15th International Conference on Distributed Systems, June 1995.

[6] Colin G. Harrison, David M. Chess and Aaron Kershenbaum, "Mobile Agents: Are they a good idea?, IBM Research Report RC 19887, Available From http://www.research.ibm.com/ xv-953-mobag-ps, 1995.

[7] M. Baldi, S. Gai, and G. P. Picco, "Exploiting Code Mobility in Decentralized and Flexible Network Management, In the first International Workshop on Mobile Agents (MA'97), April 1997.

[8] Bieszczad, A. and Pagurek, B., Towards plug-and-play networks with mobile code, In the International Conference for Computer Communications 97 (ICCC 97), pp. 19-21, November 1997.

[9] K. Kotay, D.Kotz, Transportable Agents, In the Third International Conference on Information and Knowledge Management (CIKM 94), December 1994.

[10] Sun Microsystems co., Security and Signed Applets, Available From http://java.sun.com/products/jdk/1.1/docs/guide/security/index.html