이동 에이전트 환경에서 역할 기반 접근 제어와 키 관리 기법[1)]

김동우, 송창환, 엄영익
성균관대학교 정보통신공학부
e-mail: {darin, eerien, yieom}@ece.skku.ac.kr

Role-Based Access Control and Key Management Scheme in Mobile Agent Environments

Dongwoo Kim, Changhwan Song, Young Ik Eom
School of Information and Communication Engineering, Sungkyunkwan Univ.

요 약

이동 에이전트는 기존의 클라이언트-서버 환경을 대체하는 분산 컴퓨팅 패러다임이다. 특히 이동 에이전트는 목표를 달성할 때까지 스스로 인터넷 환경을 떠돌며 정보를 수집하고 분석할 수 있도록 설계할 수 있다. 하지만 이동 에이전트가 문제없이 활동하기에 인터넷은 개방적인 환경이며, 많은 경우에 있어 이동 에이전트는 여러 적대적인 호스트들과 접할 수 있다. 이로 인해 이동 에이전트가 안전하게 인터넷상에서 이주 하도록 만드는 것이 큰 관건이 되고 있다. 최근 Volker와 Mehrdad가 이동 에이전트 환경에 있어서 효율적인 접근 제어와 키 관리 메커니즘을 제안하였다. 하지만 이 기법은 이동 에이전트의 이주 대상을 한정시키고, 이주 대상이 많아질수록 키를 관리하는 구조가 커지는 문제점이 있다. 본 논문에서는 이동 에이전트에 역할 모델을 적용함으로써 키 관리에 있어서 그 크기를 줄이고, 인증 센터를 사용하여 이동 에이전트의 이주 대상 호스트를 미리 한정짓지 않도록 하여 이동 에이전트가 자유롭게 이주할 수 있는 접근제어 기법을 제안한다. 본 기법을 이동 에이전트에 적용하면 에이전트의 크기를 줄이고, 이동할 수 있는 호스트의 제약을 줄일 수 있다.

1. 서론

전 세계적으로 인터넷 사용 인구가 기하급수적으로 증가함에 따라, 사업, 문서화 등에 있어 전통적인 방법을 사용하는 대다수의 서비스들이 인터넷을 통해서 사용하는 방법으로 변경되고 있다. 월드 와이드 웹의 확산은 우리의 경제, 문화, 사회의 깊은 곳까지도 영향을 미치고 있다. 이런 다양한 목적의 달성을 위해 복잡한 형태의 분산 컴퓨팅 환경의 네트워크가 구성되고 있다. 하지만 네트워크 대역폭은 제한되어 있고, 통신량 증가로 인한 응답 지연은 인터넷을 심각한 문제로 이끌고 있다. 특히나 빠른 응답시간을 요구하는 웹 기반의 소프트웨어 시스템들에 있어 이러한 지연은 매우 큰 문제이다. 이러한 문제점은 이동 에이전트 기술을 사용함으로써 해결 할 수 있다. 이동 에이전트란 네트워크를 통해 플랫폼 사이를 이주하면서 독자적으로 프로세스를 수행 할 수 있는 프로그램을 의미한다[1]. 이동 에이전트는 실행 코드가 이동하게 하여 목적 호스트에서 실행되므로 통신이 지속적으로 이루어지지 않아도 되기 때문에, 네트워크 부하와 트래픽도 줄일 수 있다. 이러한 이동 에이전트는 다음과 같은 특성을 갖는다[2]. 첫째로 사용자와의 상호작용 없이 스스로 목표를 달성 할 수 있으며, 두 번째로 스스로를 복제하여 전파해 나갈 수 있다. 세 번째로 다른 소프트웨어나 사용자와 협력해서 작업할 수 있으며, 마지막으로 수행할 수 있는 능력의 범위가 정해져 있다는 점이다.

이동 에이전트가 네트워크 공간에서 여러 호스트로 이주하는 과정은 수많은 위험들로부터 노출되어 있는 상태이다. 이동 에이전트의 안전한 이주와 실행을 위해 필수적으로 방어해야할 두 가지 유형의 보안 위험이 존재한다[3]. 첫 번째는 악의적인 에이전트들로부터 호스트를 보호하는 것이고, 두 번째는 악의적인 호스트들로부터 에이전트들을 보호하는 것이다. 악의적인 호스트에 노출되었을 때 에이전트 내부에 있는 개인 정보가 노출되거나 변경될 수 있다. 그러므로 실제로 여러 목적으로 사용될 수 있는 이동 에이전트를 개발하는데 있어서 에이전트 내부의 정보 보호는 매우 중요한 요소라고 할 수 있을 것이다[4].

이동 에이전트가 동작하는 과정은 수많은 호스트나 다른 에이전트와 상호작용을 맺는 과정의 연속이다. 이러한 과정을 거치는 호스트나 에이전트들은 경우에 따라 악의적인 목적을 가지고 접근할 수도 있다. 그러므로 이동 에이전트는 악의적인 공격으로부터 자신을 보호할 수 있는 보안 기술을 반드시 가져야 한다.

최근에 이동 에이전트의 보안을 위해 많은 연구들이 수

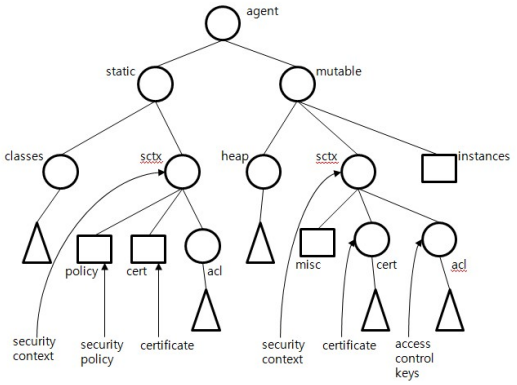
1) 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2008-(C1090-0801-0046))

행되고 있다. 그 중에서도 Volker와 Mehrdad는 트리구조를 기반으로 한 이동 에이전트의 내부 구조를 제안하고 있다[5]. 제안된 기법은 에이전트 인증, 키 관리, 접근제어 등을 지원하고 있다. 하지만 이 기법은 이주 대상을 한정시키며 에이전트의 크기가 커질 위험성을 내포하고 있다. 본 논문에서는 이러한 문제점을 해결하기 위한 기법을 알아본다.

본 논문은 2장에서 Volker와 Mehrdad가 제안한 기법을 알아보고, 3장에서 본 논문이 제안하는 접근제어 기법과 키 관리 기법을 알아본다. 4장에서는 새롭게 제안한 기법을 기존의 기법과 비교해서 장단점을 알아보고, 5장에서는 결론 및 향후 연구 과제를 제시 한 후 논문을 마친다.

2. 관련 연구

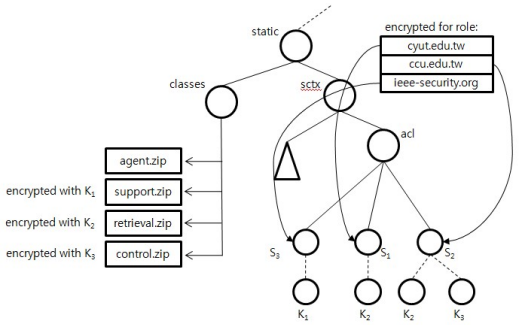
Volker와 Mehrdad가 제안한 접근제어와 키 관리 구조는 에이전트 내부의 개인 정보들을 보호하는 적절한 구조를 가진다. 이들이 제안한 기법의 에이전트의 기본 구조는 그림 1 과 같다[5].



(그림 1) 트리 기반의 에이전트 구조

에이전트는 크게 정적 가지와 동적 가지의 두 부분으로 구성된다. 정적 가지는 에이전트 코드나 보안정책과 같은 변하지 않는 정적인 데이터를 가지며 동적 가지는 클래스의 인스턴트, 힙 영역의 데이터와 같이 상황에 따라 변경될 수 있는 유동적인 데이터들을 가진다.

이 기법은 에이전트 내의 기밀 자료들의 보안을 위해서 공개키 암호방식과 대칭키 암호방식을 기반으로 한 기법을 사용하고 있다. 먼저 보안을 필요로 하는 파일들을 대칭키 암호방식으로 암호화 한다. 그 후 static /sctx/acl 폴더 내에 방문할 호스트들의 호스트명을 사용하여 폴더를 생성한다. 각각의 폴더들은 해당 호스트들이 접근하도록 허가 받은 파일들의 복호화 키를 가지고 있다. 만약 호스트가 특정한 파일에 접근하려고 한다면 그 호스트는 자신의 호스트명과 같은 폴더에서 그 파일의 복호화 키를 찾을 수 있다. 이러한 acl폴더의 하위 폴더는 해당하는 호스트들의 공개키로 암호화되어 있어서 자신의 호스트명으로 된 폴더 이외의 폴더는 복호화 하여 복호화 키를 추출할 수 없다. 각각의 호스트는 오직 해당하는 폴더만을 자신이 가진 개인키로 복호화 할 수 있다. 그림 2 는 이러한 에이전트 구조를 잘 나타내어 준다[5].



(그림 2) 호스트별 암호화를 통한 키 관리 구조

하지만 이 기법에는 다음과 같은 단점이 존재한다. 첫째로 에이전트는 홈 플랫폼에서 이주하기 전에 방문하게 될 모든 호스트들의 명단을 파악해야한다. 에이전트는 이주하기 전에 각각의 호스트들에게 필요한 권한을 할당해 주고 acl 폴더 내에 호스트 폴더 구성을 해야 하기 때문에 미리 방문할 호스트의 명단을 모두 가지고 있어야 한다. 이로 인해 에이전트가 그때그때 상황에 맞는 호스트들 사이를 이주하지 못하고 미리 지정된 호스트만 방문할 수 있게 된다. 이는 이동 에이전트의 자율성에 심각한 문제를 가져다준다. 둘째로 방문할 호스트가 많아질수록 에이전트의 크기가 커진다. 각각의 호스트들은 자신의 호스트 명으로 된 폴더를 가진다. 이런 폴더 내부에는 암호화 키들이 저장되기 때문에 방문할 호스트가 많아지면 많아질수록 에이전트의 사이즈가 커지게 되는 부작용을 초래한다.

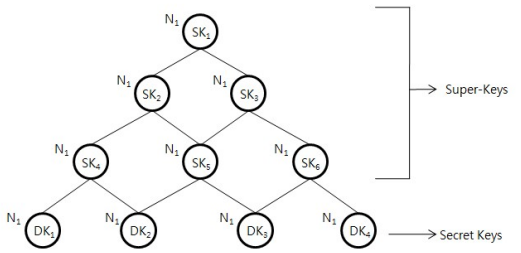
사이즈가 작고 이동 경로에 대한 제약사항이 없을수록 이상적인 에이전트의 구조이지만 위에서 살펴본 방식의 경우 이동경로에 제약이 가해질 수밖에 없는 구조를 택하고 있다. 본 논문에서는 이러한 단점을 극복 할 수 있는 새로운 형태의 에이전트 구조를 제시하겠다.

3. 제안 시스템

본 논문에서는 접근제어와 키 관리를 위한 두 가지 새로운 에이전트 모델을 제시한다. 첫 번째는 에이전트에 계층구조의 키 관리 기법을 사용해 역할 모델을 적용하는 것이고, 두 번째는 인증 서버를 통한 키 분배 기법을 적용하는 것이다.

3.1 역할 모델 기반의 에이전트 구조

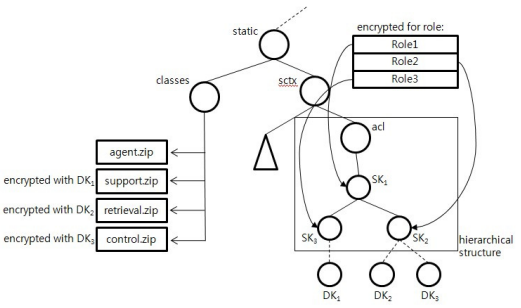
우선 새로운 역할 모델을 적용한 에이전트 구조에 대해 알아보기에 앞서 Akl과 Taylor가 제안한 계층적 구조의 접근제어 기법에 대하여 알아보자. Akl과 Taylor가 제안한 기법의 기본 구조는 그림 3 과 같다[6].



(그림 3) 계층 구조의 키 관리 기법

최상위 노드 N1은 자신의 슈퍼 키 SK1을 사용하여 자신보다 하위에 있는 노드들의 비밀 키를 유도 할 수 있다. 하지만 그 하위에 있는 노드들은 자신보다 상위에 있는 노드의 비밀 키를 유도 할 수 없다. 이는 내부 노드에도 마찬가지로 적용되는데 각각의 노드는 자신보다 하위의 노드들의 비밀 키를 유도해 낼 수 있지만 자신보다 상위에 있는 노드들의 비밀 키를 유도 해 낼 수 없다.

이제 이러한 계층적 구조의 접근제어 기법을 바탕으로 본 논문에서 제안하는 에이전트의 구조에 대해 알아보자. 각각의 에이전트들은 수행해야 할 몇 가지의 역할을 가지게 된다. 그리고 이러한 역할을 수행하는데 있어서 요구되는 파일들 또한 각각 달라질 것이다. 이러한 점에 착안하여 에이전트가 수행할 몇 가지 역할을 분류하고 그 역할에 따라서 필요한 파일들을 나누어 파일을 접근하는 것을 제한하는 기법을 제안한다.



(그림 4) 역할 모델을 적용한 새로운 에이전트 구조

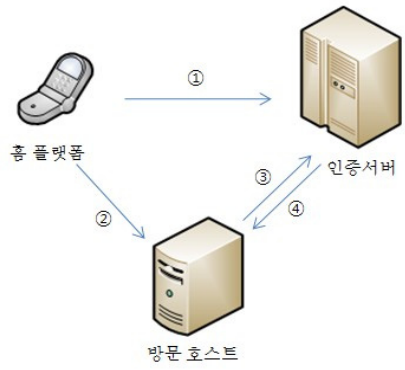
그림 4는 이러한 구조를 적용한 에이전트의 구조를 도식화 한 것이다. 기본적으로 모든 기밀 파일들은 대칭키 암호화 기법(AES, DES, IDEAL 등)을 사용하여 암호화 하도록 한다. DK1, DK2, DK3는 세 개의 기밀 파일인 support.zip. retrieval.zip, control.zip을 대칭키 암호화 기법을 사용하여 암호화 해 놓은 비밀 키 이다. 그 후 에이전트는 생성 될 때 자신이 수행해야 할 역할(Role1, Role2, Role3)에 따라 필요한 파일들을 설정하게 된다. 그림 4의 예에서 에이전트는 Role1을 수행하기 위해서 support.zip. retrieval.zip, control.zip 파일을 필요로 한다. 또한 Role2를 수행하기 위해서 retrieval.zip, control.zip 파일을 필요로 하고 Role3을 수행하기 위해서는 support.zip 파일을 필요로 한다. 그 외의 에이전트 기본 구조는 Volker와 Mehrdad가 제안한 기법을 사용한다. SK1을 획득하게 되면 하위에 존재하는 SK2와 SK3의 비밀 키를 유도 할 수 있으며, 최종적으로 DK1, DK2, DK3을 유도 할 수 있다. 즉 1번 역할에 대한 SK1을 획득하게 되면 3개의 파일에 대한 접근 권한이 생기는 것이다.

에이전트는 처음 생성될 때 각각의 역할에 맞게 계층적 구조의 키 관리 구조를 구성하게 된다. 이후 여러 호스트들로 이주하게 되고, 방문한 호스트는 자신이 수행해야 할 역할에 필요한 파일들을 접근하여 임무를 수행하게 된다. 이렇게 역할을 바탕으로 한 에이전트 구조를 사용할 경우 방문해야 할 호스트가 많이 늘어 날 지라도 호스트가 수행해야 할 역할은 몇 가지로 고정되어 있기 때문에 불필요하게 에이전트의 크기가 커지는 것을 방지 할 수 있다.

하지만 제안 기법을 사용하는데 있어서 다음과 같은 문제점이 발생한다. 우선 이전 Volker와 Mehrdad의 기법에는 방문할 호스트들이 각각의 폴더를 가지고 있기 때문에 호스트들은 자신의 공개키로 암호화 된 폴더를 복호화 하여 각 파일들의 복호화 키를 추출해 낼 수 있었다. 하지만 이런 기법은 호스트 별로 폴더를 생성해야 하기 때문에 에이전트의 크기가 지나치게 커지는 단점이 있었다. 이런 문제점을 해결하기 위해 인증 서버를 통한 키 분배 기법을 제안한다.

3.2 인증 서버를 통한 키 분배 기법

Volker와 Mehrdad가 제안한 기법에서 찾아낸 문제점을 해결하기 위해 본 논문에서는 역할기반 접근제어를 위한 인증 서버를 두는 기법을 제안 한다. 인증 서버의 역할은 각각의 호스트들로부터 수행 하고자 하는 역할과 그 역할에 대한 접근 권한을 요청받고, 역할의 타당성에 대해 검토 후 호스트에게 슈퍼 키를 배분할 수 있도록 한다.



(그림 5) 신뢰할 수 있는 호스트를 통한 키 분배기법의 사용

그림 5 는 이 기법을 사용한 키 분배 기법을 보여주고

있다. 우선 홈 플랫폼에서 에이전트가 이주 하게 될 때 모든 역할에 따른 슈퍼 키들을 인증 서버의 공개키를 통해 암호화 하게 된다. 이렇게 암호화 된 슈퍼 키들은 에이전트가 처음 이주를 시작하기 전 인증 서버로 전송된다. 이렇게 슈퍼 키가 이주된 것을 확인 하고 난 후 에이전트는 이주를 시작한다. 에이전트가 특정 호스트에 가서 특정한 역할을 수행하기에 앞 서 호스트는 그 역할을 수행하기 위해 특정 파일에 접근을 요하게 되는데 이 파일의 복호화를 위해 인증 서버로 특정역할에 해당하는 슈퍼 키를 요청하게 된다. 이때 호스트는 받아올 슈퍼 키를 암호화하기 위해 특정한 대칭 키를 같이 전송하게 된다. 인증 서버는 슈퍼 키의 요청을 받아서 슈퍼 키를 요청한 호스트의 대칭 키를 통해서 슈퍼 키를 암호화 하여 해당 호스트로 전송한다. 슈퍼 키를 요청한 호스트는 인증 서버를 통해 암호화 된 슈퍼 키를 전송받고 자신의 대칭 키를 통해 암호화 된 슈퍼 키를 복호화 한다.

이제 슈퍼 키를 획득한 호스트는 자신의 역할을 수행 하게 되고, 자신의 역할에 맞는 파일로의 접근만이 가능하게 된다. 또한 인증 서버는 전체 요청 기록을 로그로 남겨 두어 언제든지 문제가 발생하였을 경우에는 로그를 분석하여 문제를 해결 할 수 있도록 한다.

이러한 기법을 통해 슈퍼 키를 배분함으로써 에이전트 내에 슈퍼 키를 암호화해서 저장할 필요가 없으므로 에이전트의 크기는 줄어들게 되고, 에이전트가 이주하기 전에 이주해야할 모든 호스트들을 알아야 할 필요가 없으므로 에이전트의 이동성에 있던 제약을 풀 수 있게 된다.

4. 평가

본 논문에서 제안한 이동 에이전트 구조는 Volker와 Mehrdad가 제안한 기법에 비해 여러 가지 장점을 지닌다. 첫째로 에이전트 크기의 감소이다. 만약 Volker와 Mehrdad가 제안한 기법에서 100개의 호스트를 방문해야 했다면 이동 에이전트 내에 호스트 별로 암호화된 폴더가 1개씩 존재해 총 100개의 폴더가 암호화 되어 에이전트 내부에 존재해야 한다. 이렇게 이동 에이전트의 크기가 커지게 되면 호스트와 호스트 사이를 이동하는 시간이 오래 걸리게 되어 이동 에이전트가 제 시간 내에 역할을 수행하는데 지장을 초래 할 수 있다. 하지만 제안 시스템에서는 호스트별 폴더를 에이전트 내부에 보관하고 있을 필요가 없어짐으로 인해 에이전트의 크기가 줄어들게 되고 그만큼 이동성이 향상되어 제시간 내에 역할을 수행 할 수 있는데 도움을 줄 수 있다. 두 번째로는 에이전트가 이동해야할 호스트의 제약의 감소이다. Volker와 Mehrdad가 제안한 기법에서는 이동 에이전트가 호스트 사이를 이동할 시에 미리 지정된 호스트로만 이동할 수 있었지만 본 논문에서 제안한 기법에서는 현재 상황에 맞게 이동할 호스트를 선택하여 이동 할 수 있게 되었다. 이런 부분은 이동 에이전트의 자율성을 더욱 향상시켜 맡은 임무를 정확하게 수행 하는데 도움을 준다.

5. 결론

이동 에이전트는 가까운 시일 내에 서버-클라이언트 환경을 대체할 새로운 패러다임으로써 중요한 역할을 하게 될 것이다. 하지만 이러한 역할을 수행함에 있어서 보안 취약점은 여러 부분에서 걸림돌이 될 것이다. 본 논문에서 우리는 에이전트의 보안을 위해 새로운 키 관리 방법과 접근제어 기법을 제안하였다.

본 논문에서는 Volker와 Mehrdad가 제안한 계층적 구조의 접근제어 기법을 기본으로 해서, 그 기반 위에 에이전트에 역할 모델을 적용 시키는 기법을 제안하였다. 또한 키 분배를 위해 신뢰 할 수 있는 인증 서버를 도입하는 기법을 채택하였다. 이는 악의적인 호스트들에 의한 에이전트 내의 기밀 데이터로 접근을 막을 수 있는 주요한 방법이다. 본 논문에서 제안한 구조는 이동 에이전트 특성에 적합한 형태의 구조가 될 것이다.

본 논문에서 제안한 기법은 에이전트 내부의 개인정보를 보호하는 방법에 한정되어 있지만 향후 연구에서는 인증센터와 역할 모델의 적용에 관한 연구와 개발이 필요할 것이다.

참고문헌

[1] D. B. Lange and M. Oshima, “Seven Good Reasons for Mobile Agents,” Communications of the ACM, Vol. 42, pp. 88-89, 1999.

[2] T. K. Shih, “Mobile Agent Evolution Computing,” Information Sciences, Vol. 137, pp. 53-72, 2001.

[3] T. Sander and C. Tschudin, “Protecting Mobile Agents Against Malicious Hosts,” Mobile Agents and Security, Lecture Notes in Computer Science, Vol. 1419, pp. 44-60, 1998.

[4] G. Vigna, “Mobile Agents and Security,” Computer Communications, Vol. 22, pp. 1526-1527, 1998.

[5] R. Volker, and J. S. Mehrdad, “Access Control and Key Management for Mobile Agents,” Computer Graphics, Vol. 22, pp. 457-461, 1998.

[6] S. G. Akl and P. D. Taylor, “Cryptographic Solution to a Problem of Access Aontrol in a Hierarchy,” ACM Transactions on Computer Systems, Vol. 1, pp. 239-248, 1983.