A Study of Secure Purchase Protocol with Mobile Agent in Multiple-Hop Environment

"Se-Min Heo", "Yeong-Geun Choe" Soon-Ja Kim
School of Electronics and Electrical Eng., Kyungpook National University
[(urimini), indl@paigong.kmu.ac.kr]

요 약
이동 이전트는 자동으로 여려 채널의 호스트를 이용하여 사용자가 원하는 정보를 수집하거나 물품에 대한 구매 등을 할 수 있게 제공하는 프로그램이다. 그리고 이동 이전트를 사용함으로서 인터넷 환경, 특히 전자상거래에서의 많은 이익을 가져올 수 있다. 하지만 보안적인 프로토콜의 부족으로 이동 이전트의 보급과 사용에 많은 제한이 있다. 즉 악의적으로 호스트에 의해서 이전트의 state나 code가 위조될 수 있고 사용자에게 피해를 주게 된다. 본 논문에서는 Multiple-Hop 환경에서의 이동 이전트를 이용한 데이터 액세스와 구매가 위조 및 변경되는 것을 감지하고 방지할 수 있는 프로토콜을 제안하였다.

1. 서론
Corradi 등은 이동 이전트의 무결성을 보장하기 위해서 TTP(trusted third party)라는 프로토콜과 MH(multiple hop)라는 프로토콜을 제안하였다. [1,2]
TTP 프로토콜은 이동 이전트가 각 사이트를 방문할 때마다 어떤 신뢰 기관을 방문하여 이전트의 무결성을 검증할 수 있도록 설계한 프로토콜이다.
하지만 이전트의 사용자가 신뢰하는 기관을 방문하는 경우에 심한 비용이 든다는 등의 문제를 개선한 것이 MH 프로토콜이다.
이 프로토콜은 TTP 프로토콜보다 단순화된 이전트가 TTP의 상호작용 없이 네트워크를 자유롭게 이용할 수 있도록 설계된 것이다. 각 사이트들은 자신이 견제내의 이전트의 evidence를 저장하고 이를 evidence의 chain에서 송신자에게 전달하여 이용하는 것이고, evidence는 이전 사항에서 계산된 것과 암호학적으로 연관되어 있기 때문에 어떤 악의적인 호스트가 evidence를 위조할 수 없다.
결국 이전트를 owner에게 돌아왔을 때 이전트에 저장되어있는 evidence의 검증은 통해 evidence의 무결성을 증명할 수 있다. 비록 MH 프로토콜이 TTP 프로토콜과 동일한 경로에 어떤 check point가 없기 때문에 이전트의 위조를 즉시 알 수 있다는 단점이 있지만 TTP 프로토콜을 이용하면 이러한 비용이 적게 드는 장점이 있다.
Kotzanikolaou 등은 Sander과 Tschudi가 제안한 undetachable signature의 문제점을 해결하면서 사용자와 서비시어 사이에서 안전한 거래가 이루어질 수 있는 방법을 제시하고 있으며, 그려 인해 이전트가 사용자의 개인키를 노출시키지 않고 서비시어에 메시지에 안전한 사항을 할 수 있는 방법을 제안하였다. [3]
본 논문에서는 앞서 제시한 방법들을 분석하고 그 단점들을 보완하여 MH 환경에서 이전트의 무결성을 저지 못하도록 사용자와 서비시어 사이에 안전한 거래가 이루어낼 수 있는 방법을 제안한다.
논문의 구성은 2절에서는 사용자의 안전한 서명을 위한 Kotzanikolaou 등이 제안한 방법과 프로토콜 분석, 3절에서는 제안하는 프로토콜, 4절에서는 결론을 맺는다.

2. MH Protocol
이전트는 세 부분으로 구성되어 있다. 그 구성요소로 보면 첫째가 CID(code and initialization data), 둘째가 AD(application data), 셋째가 PD(protected data)이다. CID는 이전트의 실행 코드, 사용자의 Identity, 사용자의 정보를 포함하는 request가 포함되어 내용이 변경될 수 없도록 사용자의 개인키로 서명되어 있다. AD는 각 사이트에서 수집된 데이터와 사이트의 Identity가 저장되는 장소이고, 마지막으로 PD는 이전트의 무결성 및 무결성에 관련된 proof들이 저장되는 장소이다. [2]
이전트는 각 사이드의 S_i로부터 마지막 사이드의 S_w까지 이동하면서 정보를 수집한다고 가정한다. 이전트는 먼저 S_w에서 다음 사이드인 S_i의 secret로 사용할 random number T_0를 hash한 값 T_1 hash(T_0)를 구하고 T_1 을 S_i의 공개키로 암호화(ET)한다. 그 다음은 그림 1에 자세히 묘사되어 있다. 그림 1과 같이 이전트가 모든 사이드를 이동한 후 최근 사이트 S_w로 돌아왔을 때 이전트의 무결성을 증명하게 된다.
증명과정은 앞서보면 먼저 secret secret key T_1과 AD part의 S_i, D_i의 내용을 구한다. 이 값들은 M_i와 같은 값을 갖고 있다. 그리고 PD part의 D_s(MIC)과 계산된 MICs를 비교한다. 다음으로 T hash(T_1)와 AD part의 S_i와 D_i의 내용과 앞서 계산해 MICs를 구한다. 앞에서 마찬가지로 MICs와 값이 같은 값을 비교한다. 이는 계산된 마지막 사이드 인 S_w까지 증명을 마치게 되고, signature 증명이 실패할 때까지 계속한다. 만약 K까지의 증명이 성공한다면 이전트의 무결성이 증명되는 것이라고, 도중에 signature 증명이 실패했다면 그 이후의 데이터에 대해서는 무결성이 보장되지 않는 것이다. 즉 이전트의 데이터가 변조되었을 때
지나 위조되었음을 알 수 있다.
이러한 MH 프로토콜을 이용하여 에이전트가 데이터를 수집하고 그 데이터에 무결성을 검증할 수가 있다.
하지만 사용자가 수집된 데이터를 이용해서 어떤 몰론을 수행하고자 할 때 사용자의 신원이 필요하다. MH 프로토콜은 이러한 발전에 대해서는 제시하는 것이 아니다. 다음은 사용자의 이름을 에이전트가 어떻게 안전하게 판단하는지에 대해 설명하고자 한다.

![Diagram](image)

지난 사용자는 그의 개인적을 노출시켜서 알고 수용을 위 한 메시지에 서명하기를 원한다. 이를 위해서 Potzaniokou 등은 Sander와 Tschudin이 제안한 Undetachable Signature와 CEF(Computing with Encrypted Function)를 이용하여 RSA signature를 기반으로 사용자의 비밀키를 노출시키지 않고 서명할 수 있게 하는 방법을 제안하였다[3].

먼저, 앞서 사용될 간단한 예를 설명하면, *h*는 하위 함수, Server_ID는 서버의 Identity, Client_ID는 사용자의 Identity, bid_S는 사용자의 주문에 대한 서버의 주문 정보다.

CEF라는 개념을 설명한다면, 어떤 electronic shop에서 물품을 구매할 때 사용자의 signature 함수 *s*가 사용된다. 그래서 signature 함수 *s*를 보호하기 위해서 사용자는 함수 *f*와 함께 암호화(*f*(1) := s)한다. 그리고 에이전트에게 전달될 수 있는 code인 (*h* *f*(*m*), *f*(*s*(*m*)))의 함수 *f*는 *h*로 모의 *n*과 *f*(*s*(*m*))는 주계산되고 서버의 입력값 *x*(Server_ID, Client_ID, bid_S)를 가지고 에이전트가 가지고 있는 함수의 값을 실행하게 된다. 그림 2와 같은 서명 값이 생성된다.

\[ f(x) = m, f(s(x)) = s(f(x)) = s(m) \]

그림 2. 서명 값

RSA 서명은 그림 3에 간단히 설명하였다.

1. 같은 크기의 큰 random 수 p, q를 생성
2. n = pq, *d* = (p-1)(q-1)를 계산
3. 1 < *c* < *p*, gcd(*c*, *d*) = 1로 random 수 e 선택
4. 1 < *d* < *n*, *ed* = 1(mod *n*)의 유일한 수 e 계산
5. 개인키는 *d* 이고, 공개키는 *n*과 *e*이다.

그림 3. RSA 서명 기법

\[ k(=h^d \mod n) \]는 *h*에 대한 사용자의 RSA 서명이다. *f*(*s*(*m*))는 사용자의 RSA 함수 *s*() = (*f*() mod *n*)의 암호화 된 함수 *s*로 *f*이다. 즉 그림 4와 같은 유도식

\[ f(s(x)) = s*f(x) = s(f(x)) = s(h^d) = (h^d)^e = (h^{ed}) = k^d \]

그림 4. 서명 값의 유도식

에이전트가 보내낸 후 에이전트의 수행 단계에서 이동 에이전트는 서버에 의해 앞서 설명한 입력 *x* 값을 통해 실행된다. 그러므로 그림 5와 같이 RSA 서명 값, undetachable signature인 *m*과 *z*를 얻게 된다. 따라서 에이전트는 사용자의 비밀키의 노출 없이 서명을 완성하게 된다.

\[ m = f(x) = h^d \mod n \]
\[ z = f(s(x)) = k^d \mod n = (h^d)^e \mod n = m \mod n = s(m) \]

그림 5. RSA 서명 값 *m*, *z*(undetachable signature)
3장에서 언급된 프로토콜의 개념도는 그림 4와 같다. parameter는 앞에서 설명한 RSA에서 사용된 것과 동일하여다. d는 사용자의 개인키, n과 e는 사용자의 공개키, C는 Client_ID이다.

그림 6. RSA 기반의 undetachable signature

본 논문에서는 Corradi 등이 제안한 MH 프로토콜과 Kotzanikolou 등이 제안한 서명방식을 분석하고 그 단점을 해결하였고, 또한 제안한 두 가지 방식을 접목시켜 MH 환경에서의 안전한 구체가 얻어질 수 있는 방법을 제안하였다. 제안한 방식을 사용함으로써 MH 환경에서의 이동 예인트를 이용한 데이터의 무결성가 보장되고, 안전한 구체가 가능하게 된다.

5. 결론
본 논문에서는 Corradi 등이 제안한 MH 프로토콜과 Kotzanikolou 등이 제안한 서명방식을 분석하고 그 단점을 해결하였다. 또한 제안한 두 가지 방식을 접목시켜 MH 환경에서의 안전한 구체가 얻어질 수 있는 방법을 제안하였다. 제안의 방안은 이동 예인트의 보안적인 구체들의 해결을 위해 더욱이 다양한 연구가 지속적으로 요구된다.

6. 참고문헌

그림 7. 예인트 이동 개념도