자원 공유 커뮤니티를 위한 인증 기술과 신뢰관계사슬*

김 정 곤** · 김 신 곤***

Authentication and Trust Relationship Chaining for Resource Sharing Community

Kim, Jeong Gon · Kim, Shin Kon

〈Abstract〉

This article proposed the authentication protocol for peer-to-peer resource sharing community. The proposed protocol does not require a priori information for generating and exchanging authentication key.

Also this protocol can provide the delicate access control by allowing the user(authenticator) to assign the trust level to the authentication supplicant, which can be used to decide if the resource providing node will accept the resource sharing request from a resource requesting node. Trust Relationship Chaining provides the environment where trust levels (included in the trust table) of nodes in the resource sharing community are propagated among nodes when trust tables are exchanged between two nodes engaged in mutual authentication process and authentication refresh so that any two nodes which are not directly mutual-authenticated can assign the trust level each other for the access control for resource sharing.

In the proposed protocol a node can implements the authentication refresh continuously to verify the effectiveness of authentication after mutual authentication so that the authentication of new node or authentication revocation(effectiveness cancellation) of the departed node can be propagated to the all the nodes in RSC and eventually safe resource sharing community is configured.

Key Words : Authentication, Access Control, Resource Sharing, Trust Relationship Chaining

Ⅰ. 서론

통신 기술의 발달과 함께 무선 이동 단말기(Wireless Mobile Devices)는 점점 소형화되고 인간 친화적으로 되어 거의 모든 사람들의 일상생활에 침투되어 사용되고 있다. 이러한 무선 통신 단말기들은 사용자들이 이동 중에 통신용으로 사용하는 것 외에 가전 기기나 MP3플레이어, PDA같은 생활용품으로 사용되고 있어서 유비쿼터스 컴퓨팅 환경 하에서 고정된 네트워크(Fixed

* 본 논문은 2007년도 광운대학교 교내 학술연구비 지원에 의하여 연구되었음.
** 한세대학교 IT학부 조교수(제1저자, 교신저자)
*** 광운대학교 경영정보학과 교수

Infrastructure Network)에 의존하지 않고 언제 어느 때나 모바일 애드혹 네트워크(Mobile ad hoc network, 이하 애드혹 네트워크)를 기반으로 하여 단말기(이하 노드 또는 UMO; Ubiquitous Mobile Object와 병행하여 사용) 사이에 자율적으로 자원(예를 들면, 다른 모바일 기기의 메모리나 디스플레이 장치, 스피커 등)을 공유하는 자원 공유 커뮤니티(RSC; Resource Sharing Community)를 생성하는 것이 가능하게 되었다[1-4].

이러한 커뮤니티에서는 개인은 다른 참여자가 제공하는 자원을 사용하기도 하고 커뮤니티의 또 다른 구성원에게 자신의 UMO가 보유한 자원을 제공하기도 하므로 사용자에게 편리함을 제공하지만 자원을 제공하는 개인과 자원을 사용하는 개인 사이에 안전한 접근 및 사용을 보장하는 보안 프레임워크(Security Framework)는 매우 중요하다고 할 수 있다.

자원 공유를 구현하기 위한 다양한 방법들은 현재까지 많은 연구가 진행되고 있는 상황이며[5-7] 이와 함께 RSC에 적용할 수 있는 보안 기술에 대해서도 아직까지 연구가 진행되고 있다[6-7].

앞서 설명한 바와 같이 RSC는 애드혹 네트워크를 기반으로 하는 네트워크로서 네트워크 근원(Network Origin) 이나 네트워크의 일시성(Network Transiency)과 같은 RSC의 특성에 알맞은 커뮤니티 구성 방법과 보안 기술을 적용하여야 [8] 안전한 RSC를 운영할 수 있다.

본 논문에서는 애드혹 네트워크를 기반으로 생성되는 RSC를 위한 인증 기술을 제안하였다. II에서는 P2P의 특성을 갖는 기기들이 인증에 사용할 정보를 사전에 공유하지 않는 상황에서 인증키를 생성하는 보안 프로토콜의선행연구(2.1)와 본 논문에서 제안하는 인증 프로토콜의 설계목표(2.3)를 정리하였다. 3.1에서는 애드혹 RSC에 적용할 수 있는 보안 인증 프로토콜을 제안하였으며 3.2와 3.3에서는 신뢰관계사슬(TRC; Trust Relationship Chaining)을 이용한 RSC의 구성에 대하여 설명하였다. 3.4에서는 제안하는 보안 인증 모델을 자원공유에 적용하는 방법을 설명하였으며 3.5에서는 제안하는 인증기술의 안전성에 대하여 설명하였다. 3.6에서는 기존의 인증프로토콜과 제안하는 인증프로토콜을 보안 시스템 운영측면에서 비교하였다. IV에서는 본 논문에서 제안하는 보안 인증 프로토콜 문에개선 가능성에 대하여 설명하였고 V에서는 본 논문에서 제안하는 인증 기술의 장점을 요약하였다.

Ⅱ. 자원공유 커뮤니티 인증기술

2.1 선행연구

애드혹 네트워크를 이용하는 응용분야에 적용이 가능한 인증과 키 교환(authentication and key exchange)의 문제를 해결할 수 있는 방법들이 아직까지 만족할만한 수준에 있지 않음에도 불구하고 이 문제를 다루는 관련 연구는 많지 않다[9]. 지금까지 연구된 애드혹 네트워크 연구들은 보안 프로토콜을 시행하기 전에 기기들이 미리 인증에 필요한 정보들을 사전에 공유하고 있어서 성공적으로 인증이 이루어지는 것을 전제로 하면서도 인증과 키 교환이 어떻게 이루어지는지를 설명하지 않았다[9].

선행 연구들 중에서 서버를 사용하지 않는 인증 프로토콜과 관련된 연구에는 사전 인증(pre-authentication) 또는 기기들의 물리적 접촉(physical contacts of devices)과 [10-12] 서버가 없는 상황을 보상하기 위한 방법으로 사용자가 개입하여 패스워드를 공유(password sharing)[13]하는 사전인증의 방법이 있다.

Asokan[13]이 지적한 바와 같이 안전한 세션을 설정하려는 노드들은 세션을 설정하기 전에 사전 정보(a priori context)를 반드시 공유하여야 한다. Asokan이 고려한 환경은 학술대회의 회의실 안에 있는 소수의 사람들이 미팅하는 동안에 자신들의 랩톱 컴퓨터를 이용하여 무선 네트워크 세션을 설정하는 경우이다.

[13]에서 저자가 제안한 방법은 회의실 안에 있는 사람들 사이에서, 예를 들면, 칠판에 패스워드를 쓰고 그 패스워드를 공유하는 것을 한 가지 방법으로 제시하였다.

그렇지만 이러한 방법으로 공유한 패스워드는 사전 공격(dictionary attack)[14]에 취약하기 때문에 이 패스워드를 직접 키로 사용할 수는 없으며 이러한 약점을 보완하기 위하여 저자는 공유한 "약한 공유 키(weak password key)"에서 "강한 공유 키(strong shared key)"를 생성하는 "패스워드 인증 키 교환(password authenticated key exchange)"을 사용할 것을 제안하였다. 이러한 보안 메커니즘은 간단한 사전정보를 사전 인증으로 사용할 수 있는 장점이 있지만 애드혹 네트워크 안에 있는 모든 사용자들이 물리적으로 같은 방안에 있어야 하고 연결을 하기 전에 상호간에 서로 알고(신뢰하고) 있어야 하는 조건을 필요로 하는 단점이 있다.

Stajano[10-11]는 기기들이 "각인(imprinting)"이라는 방법으로 사전정보를 교환한 후에 인증과 키 교환을 하는 사전 인증(pre-authentication, 저자는 사전인증이라는 용어를 사용하지 않았다)을 하는 "부활하는 새끼 오리 프로토콜(resurrecting duckling protocol)"을 제안하였다. 이 프로토콜은 두 단계로 나누어져 있는데 첫 단계는 사전인증(pre-authentication)으로서 새끼 오리 기기(device to be controlled)와 엄마 기기(control device)가 위치 제한 통신 채널(location-limited channel)을 이용하여 각인(imprinting)이라는 비밀 정보(secret data)를 교환을 한다. 위치 제한 통신 채널의 예는 두 기기를 직접 접촉하여 구현하는 것이 가능하다. 두 번째 단계에서는 새끼 오리는 첫 단계에서 교환한 비밀 정보를 제공하는 기기만을 자신을 제어할 수 있는 엄마 기기로 인식을 하게 된다. 즉 사전인증 단계에서 위치 제한 채널을 통해서 교환한 비밀 데이터를 인증을 위한 정보로 사용하는 것이다.

부활하는 새끼 오리 프로토콜은 모바일 기기의 제조자가 유통망에 배포한 모바일 기기가 자신에게 처음으로 비밀 정보를 송신하는 기기를 자신의 엄마 기기로 인식하게 하는 것이 가능하므로 두 기기간의 인증을 구현할 수 있으나[10] 자원공유를 위하여 즉흥적으로 생성되는 RSC에 적용하기에는 적당하지 않은 기술이다.

2.2 적용 환경 및 배경

RSC를 구현하기 위하여 해결하여야 할 요구사항들은 [15] 자원탐색(resource discovery), 자원 퍼블리싱(resource publishing), 중재시스템(coordination system), 신뢰형성(Trust Establishment), 정산 기능(clearing mechanism)이 있다. 일단 UMO 사이에 적절한 인증이 형성이 되면 적절한 키 교환 프로토콜을 사용하여 기밀성, 무결성, 부인봉쇄의 기능을 구현할 수 있으므로 인증 기술은 안전한 RSC를 구현하기 위한 핵심기술이라고 할 수 있다. 특별히 신뢰형성(Trust Establishment)은 자원을 제공하는 UMO가 자원을 요청하는 UMO의 신분(identity)을 확인할 수 있는 인증(Authentication)과 신뢰도가 높은 UMO에 대하여는 더 많은 자원을 제공할 수 있는 접근통제 정책을 통하여 구현하게 되므로 인증기술은 신뢰형성을 위한 중요한 기술이다.

기존의 파일 공유에서와 같이 P2P(Peer-to-peer)를 구현하는 방법에는 순수한 P2P(Pure P2P), 중앙집중식 P2P(Centralized P2P), 하이브리드 P2P(Hybrid P2P)가 있다. 본 논문에서는 유비쿼터스 환경에서 사용자들이 자유롭게 이동하면서 커뮤니티에 자유롭게 참가하거나 이탈할 수 있는 환경에서 적용이 가능하고 특정한 UMO가 인증기관 역할을 하거나 또는 자원 공유 리스트를 생성/보관해야 하는 부담을 지지 않는 순수 P2P 형태의 RSC를 위한 인증 기술(프로토콜)을 제안하였다. 또한 제안하는 인증기술을 자원공유를 위한 접근제어에 적용하는 방법도 제안하였다.

2.3 설계 목표(Design Issues)

RSC의 보안과 관련된 인증 기술을 적용하는 데에 있어서 다음과 같은 설계 목표를 정하고 인증 기술을 연구하고 제안하였다.

첫째, 자원 공유는 애드혹 네트워크의 형태로 구성되어 물리적이거나 논리적인 인프라스트럭처를 통한 지원

(infrastructure support)은 없는 환경에 적용할 수 있는 인증기술이어야 한다.

둘째, RSC에서는 UMO 사용자가 RSC에 참여(join)하거나 또는 이탈(leave)하는 것이 임의적이기 때문에 UMO는 인증을 거친 다른 UMO들에 대하여 지속적으로 인증의 유효성(effectiveness)을 확인하여야 한다(3.1.3 인증 갱신 참조).

셋째, 무선 통신을 사용하는 이동 단말기들에게는 통신 채널은 (하드웨어적인 면에서) 속도, 가용성 면에서 저 사양(low specification)일 수 있으므로 인증을 위한 프로토콜은 효율적이면서 경량이고 적은 대역폭(efficient and small bandwidth)을 사용하여야 한다.

넷째, RSC는 애드혹 네트워크에 기반을 두고 있으므로 일반적인 무선 애드혹 네트워크가 겪는 보안 위협(security threats)을 받을 수 있으나 본 논문에서는 인증 프로토콜이 재생공격(replay attack)을 방지할 수 있도록 설계하였으며 다른 종류의 다양한 보안위협에 대해서는 더 많은 연구가 필요한 상태이다.

2.4 자원 공유 커뮤니티를 위한 가정 (Assumptions)

본 논문에서는 RSC와 관련하여 다음과 같은 일곱 가지의 가정을 하였다.

첫째, RSC 구현을 위한 첫 번째 가정은 자원탐색 단계에서 각 UMO는 자신의 주변에서 "한 홉 거리 내에 있는 다른 UMO들(one hop UMOs)"을 발견하고 (예를 들어, broadcasting을 통하여 구현) P2P의 형태로 상호간에 인증을 할 수 있는 기능을 보유하고 있는 것으로 가정하였다. 상호인증시에 한 노드는 다른 노드에 대한 자신의 신뢰정도(Trust Level)를 수치화된 값으로 배정하여 자신의 신뢰테이블(Trust Table; TT)에 입력할 수 있다.

둘째, UMO 상호간의 인증시에 하나의 UMO 'A'는 특정한 다른 UMO 'B'와는 한 홉 거리 내에 있지만 또 다른 UMO 'C' 와는 한 홉 거리 내에 있지 않지 않은 경우에 UMO 'A'와 UMO 'C'는 별도의 상호인증을 수행하지 않더라도 TRC를 이용하여 UMO 'A'를 인증한 UMO 'B'는 자신의 인증한 UMO 'A'에 관한 정보를 신뢰테이블(TT; Trust Table)을 통하여 UMO 'C'에게 전파시킬 수 있는 것으로 가정하였다 (3.2에서 TRC의 개념을 설명한 <그림 6>과 <그림 7> 참조).

셋째, 모든 UMO 들은 자신만의 고유한 ID (UMO ID 또는 RSC ID)를 가지고 있다 (예를 들면, IP주소나 MAC 주소). 그리고 상호인증과정에서 노드는 RSC ID를 교환하고 (<그림 3>에서 상호인증시에 RSC ID를 교환하는 내용은 생략하였다) TT에는 상호인증에 필요한 정보들을 저장되는 것으로 가정하였다 (3.1.1과 <그림 2> 참조).

넷째, 모든 UMO는 커뮤니티에 가입한 후에 물리적으로 어느 정도의 거리를 이동하여 통신이 불가능한 지역으로 이동할 수 있는 상황을 가정하였다.

다섯째, 각 UMO는 TRC의 특성을 이용하여 자신의 주변에 한 홉 내에 있지 않은 UMO들 중에서 비협조적인 UMO를 발견하여 (예를 들어, 위의 네 번째 가정에서 설명한 바와 같이 통신이 불가능한 지역으로 이동하여 RSC에서 이탈한 UMO를 인식) 이런 UMO들을 자신의 TT에서 유효성취소(effectiveness cancellation)로 설정할 수 있는 기능이 있는 것으로 가정하였다.

여섯째, 모든 UMO는 자신의 자원 테이블(Resource Table)을 보유하고 있는 것으로 가정하였다. 예를 들어 일반 데스크톱 컴퓨터인 UMO 'A'는 다른 노드와 공유가 가능한 자원으로 CPU, 키보드, LCD 모니터를 정할 수 있고 각 자원마다 자원을 사용할 수 있는 최소 요구 신뢰 수준 (Minimum Required Trust Level)이 정해져 있다고 가정하였다. 노드 'A'의 사용자가 자신의 CPU를 사용할 수 있는 최소 요구 신뢰 수준을 0.95 이상으로 정하면 노드 A는 TRC를 통하여 계산한 신뢰수준이 0.95이상인 노드 또는 UMO에 대해서만 자신의 CPU를 공유할 수 있게 허락을 한다 (3.4의 <그림 9> 참조).

일곱째, 각 UMO는 자원 공유 보안을 위하여 다른 UMO 에 대하여 적절한 인증을 거친 후에 자신이 보유

한 자원들 중에서 다른 노드들과 공유하기 위하여 제공할 수 있는 자원 테이블(Resource Table)을 전송하는 것으로 가정하였다.

2.5 제안 기술의 특징

현재의 잘 알려진 인증 기술은 중앙집중식(centralized)의 Kerberos, X.509, PKIX 등과 같이 신뢰성 있는 제3의 기관(TTP; Trusted Third Party)인 인증기관(CA; Certificate Authority)이 발행한 인증서(certificate)를 통하여 두 개체가 상대방을 인증하는 방법이 있다[16]. 이 방법은 안정적인 유선 네트워크에서는 유효하게 동작하지만 인증기관의 존재에 대한 불확실성, UMO의 인증서 검증을 위한 계산 능력의 저사양, 단일 장애점(single point of failure) 등의 문제 때문에 애드혹 네트워크에의 적용하는 것은 적절하다고 할 수 없다.

TTP에 의존하지 않는 인증 기술 중에서 잘 알려진 기술은 시도-응답 (Challenge-and-Response) 인증 프로토콜이며[17] 사전에 두 개의 개체가 공유 비밀키(shared secret key)를 공유하는 것이 가능하면 시도-응답을 이용하여 인증을 하는 것도 가능하다. 그러나 두 개체간이 사전에 공유비밀을 공유하는 것이 가능한 것인가 하는 문제는 RSC가 사전계획(planned)에 의해서 생성되는 것인지 또는 RSC가 사전에 알고 있는 사람들 사이에서 생성되는지의 여부에 따라서 적응하여 결정되는 문제이다.

본 논문에서 제안하는 인증기술은 RSC가 사전에 (적어도 RSC에 참여할 사용자들이 사전에 인증하는 데에 필요한 정보를 공유 또는 교환할 정도로 충분한 시간을 가지지 않은 상태로) 계획되지 않고 순수하게 애드혹하게 생성되는 상황에 적용되는 것으로 가정하였다. 또한 인증을 거친 UMO 사이에서 다른 UMO에 대한 유효성(즉, 인증 후에 계속적으로 RSC에 속해있는지 여부)을 검증하고 다른 UMO가 기존의 RSC에 속하지 않은 새로운 (또 다른) UMO를 인증하였는지(즉, 새로운 UMO가 RSC에 신규로 가입하는 경우)를 확인하는 과정만을 고려하였다.

저자들은 위에서 제시한 적용환경에서 설계목표를 구현할 수 있는 보안 인증 모델을 제안하였다. UMO 사이에서 다른 UMO에 대하여 초기 인증 후에 유효성검증을 위하여 지속적으로 갱신(refresh)을 확인하는 방법은 Lamport[18]가 제안한 "단방향 해쉬 체인(one-way hash chain)"을 응용하였으며 UMO 사이에서 상호간의 신뢰(Trust)를 RSC 내에서 전파/공유하는 방법은(아래의 3.2 참조)은 Mobility Based Approach[19]와 인증서사슬(Certificate Chaining) 메커니즘[20]을 효과적으로 조합하여 제안하였다.

Ⅲ. 자원 공유 인증 기술의 제안

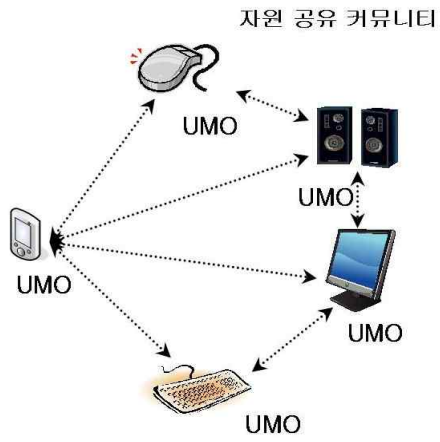
본 논문에서 제안하는 RSC는 TTP가 없는 P2P 서비스 환경을 전제로 하며 상호 인증 절차를 수행한 노드들로 구성된 가상의 커뮤니티이다. 모든 노드는 자신의 주변을 정기적으로 검색을 하여 주변에서 이미 인증을 하지 않은 새로운 노드를 발견하면 상호인증(초기 인증)을 하여 상호간에 RSC회원으로 참여한 것으로 인지를 하고 (3.1.2. 참조) 이미 인증을 마친 노드에 대해서는 정기적으로 인증 갱신을 하는 것으로 가정을 하였다. 또한 사용자가 UMO의 전원을 끄거나 또는 통신이 불가능한 장소로 이동을 하여 인증갱신 때에 반응을 하지 않으면 RSC의 회원의 유효성이 취소(effectiveness cancellation)되는 것으로 가정을 하였다.

각 노드간의 통신은 보안 채널을 통해 안전하게 정보교환이 이루어지는 것으로 가정하였다. 예를 들어, RSC에서 하나의 노드가 자신보다 큰 화면을 갖고 있는 다른 노드의 화면을 사용하는 경우에 동영상을 전송하는 통신은 무선랜을 이용할 수 있지만 상호 인증을 하는 과정에서는 블루투스와 같은 통신 채널을 보안채널로 이용할 수 있다.

상호인증이 끝나면 하나의 노드는 인증을 마친 상대방에 대한 신뢰수준에 대한 정보를 확보하게 되는데

RSC에서는 본 논문에서 제안하는 TRC를 이용하여 하나의 노드가 알고 있는 다른 노드에 대한 신뢰수준을 다른 노드에게 전파시킬 수 있다.

또한 자원을 제공하는 UMO는 자원에 대한 할당된 최소 요구 신뢰수준과 자원을 요청하는 노드의 신뢰수준을 비교하여 자원 제공 여부를 판단함으로써 접근제어(Access Control)를 수행하는 것도 가능하다. RSC의 구조는 다음의 <그림 1>과 같다.



<그림 1> 자원 공유 커뮤니티(Resource Sharing Community)의 구조

3.1 인증 프로토콜

인증 프로토콜은 2 단계로 구분된다. 첫 번째 단계는 초기 인증(Authentication Initialization) 단계로서 RSC에 참여하는 UMO들이 상호간에 처음으로 인증을 하는 단계이고 두 번째 단계는 RSC에서 상호인증을 마친 UMO가 지속적인 회원으로 남아 있는지의 여부를 확인하기 위한 인증 갱신(Authentication Refresh)이다 (3.1.3 인증 갱신 참조).

3.1.1 키와 신뢰테이블(Keys and Trust Table)

제안하는 인증 메커니즘에서는 기기 수준(Device Level)의 인증 처리가 이루어지며 서비스 속도와 자원의 제약적인 환경을 고려하여 최적화된 프로토콜과 알고리즘을 사용한 보안 모델이다. 인증 프로토콜에서 사용되는 보안 키는 네 가지가 있고 사용 목적은 다음과 같다.

- 마스터키 (Master Key)
  각 노드에서 개별적으로 생성된 키 정보로 모든 키 생성을 위한 seed 값으로 사용하며 노드 'A'의 마스터 키는 $K_{master-A}$이다.
- 갱신 키(Refresh Key 또는 Session Key)
  노드 'A'의 갱신 키(refresh key)는 $K_A = H(K_{master-A} \| r_A \| r_B)$로 마스터키 $K_{master-A}$, 자신이 생성한 난수 값 (Random Number) r_A, 상대 노드가 송신한 난수 값 r_B 값을 연결(concatenation)한 후에 해쉬계산을 하여 생성하며 인증키를 생성하기 위한 seed로 사용한다. RSC 의 회원으로서 역할을 수행할 경우에만 사용하는 임시키이다.
- 인증 키 (Authentication Key)
  $H_N(K_A)$로서 갱신 키(refresh key)를 N회 만큼 반복하여 해쉬(One Way Hash)하여 생성한 최종 값이다. 인증키를 이용하여 갱신키를 유추하는 것이 불가능하게 설계되었다.
- 인증 토큰(Authentication Token)
  $H_k(K_A)$로서 인증 갱신을 위하여 사용한다. 갱신키를 카운터 k 회수만큼 반복하여 해쉬를 하여 생성하며 인증 갱신을 위하여 사용한다.

인증과 관련된 정보는 노드가 관리하는 TT에 저장이 된다. TT는 신뢰관계(Trust Relationship)가 확립된 각 노드에 대한 RSC ID, Trust Level, 홉 거리(Hop Distance), 유효성 여부, 인증 토큰 값으로 구성된다. TT정보 교환을 통한 TRC를 구성하여 신뢰할 수 있는 RSC을 생성하고 유지한다. TT에서 관리되는 정보의 항목과 내용은 아래와 같다.

- RSC ID
  신뢰관계(Trust Relationship)가 확립된 노드의 식별자(UMO 식별 정보)로서 MAC Address, IP address등 구성된 RSC에서 식별할 수 있는 정보이다.
- 신뢰수준(Trust Level)
  하나의 노드가 다른 대하여 정하는 신뢰수준으로서 0.0과 1.0 사이의 값을 갖으며 1.0은 완전신뢰이고 0.0은 불신임을 나타낸다. 한 홉(hop) 거리의 노드에 대한 신뢰수준은 노드의 사용자가 다른 노드에 대한 신뢰수준을 자신의 UMO에 직접 입력하여 정한다. 두 홉 이상 떨어진 노드에 대한 신뢰도는 TT를 송신한 인접 노드에 대한 신뢰수준과 홉 거리 가중치의 곱으로 계산한다. 예를 들어, 노드 'A'와 노드 'D'의 홉 거리(Hop Distance)가 $d=4$인 경우에 홉 거리 가중치는 $(1-p)^{(d-1)}=(1-0.1)^{(4-1)}=0.729$이며 $p=0.1$은 홉 수의 증가에 따른 신뢰도 감소를 고려하기 위한 신뢰도 감소 인자(attenuation factor)이다.
- 유효성 여부
  대상 노드의 현재의 유효성 상태를 나타내는 정보로서 주기적인 인증 갱신 검증이 실패한 경우 해당 노드는 이 항목이 유효성취소(effectiveness cancellation)로 설정(Setting)된다.
- 인증 키
  갱신 키(Refresh Key)를 통하여 최종 생성된 인증 검증 값으로서 $Auth-Key_A=H_N(K_A)$이다
- 카운터 값(Counter Value) k
  카운터 값 k 는 1보다 크고 최대 카운터 값(Maximum Counter Value) N보다 작은 값($1<k<N$)을 갖으며 인증을 증명하여야하는 노드가 인증 토큰을 생성할 때 사용한다. 인증 갱신을 할 때 마다 이전에 사용한 카운터 값 보다 항상 작은 값을 사용하며 재생 공격(Replay Attack) 방지를 위한 목적으로 사용한다. 아래의 <그림 2>는 노드 'x'의 (다른 노드에 대한) 신뢰 테이블의 예를 보여준 것이다.

노드 x의 신뢰 테이블					
RSC ID	신뢰 수준	홉 거리.	최대 카운터	인증키	유효성
w	0.7	1	1000	$H_N(K_X)$	Yes

<그림 2> 신뢰 테이블(Trust Table)

인증 메커니즘에서 TT 및 신뢰수준 정보 관리는 신뢰할 수 있는 인접 노드간의 주기적인 정보 교환을 통해 이루어진다. 인증갱신을 하면서 두 노드는 TT의 해쉬 정보를 교환을 하고 TT의 변경 유/무를 판단하여 해쉬 값이 다른 경우 TT 정보를 교환한다.

3.1.2 초기 인증 (Authentication Initialization)

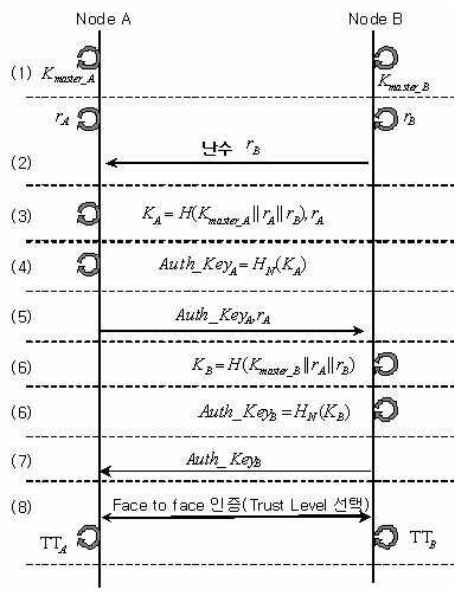
인증 초기화를 위한 절차는 초기 인증 정보 교환과 오프라인 대면확인(i. e. face-to-face confirmation) 단계를 포함하며 애드혹 특성 상 물리적으로 매우 근접하기 때문에 사용자에 의한 식별 및 인증 등 대면 확인 절차가 가능하다. 즉 시스템 적인 측면의 인증 정보 교환과 교환된 인증 정보에 대한 신뢰성을 확보하기 위한 오프라인 대면확인을 결합하여 활용한다. <그림 3>에 나타난 초기 인증 프로토콜을 단계별로 설명하면 다음과 같다.

1 단계. 각 노드는 접속하기 위한 RSC을 위한 마스터키 $K_{master-A}$를 생성한다. 생성된 노드 'A'의 마스터 키 $K_{master-A}$, 노드 'B'의 마스터 키 $K_{master-B}$는 안전하게 생성, 저장 관리된다.

2 단계. 갱신키 K_A, K_B 의 반복 생성을 방지하기 위한 난수 값 r_A, r_B을 생성 및 상호 교환한다. UMO 'B'는 난수 r_B를 UMO 'A'에게 전송한다.

3 단계. 갱신키(Refresh Key 또는 Session key) 생성을 위하여 마스터키와 난수 값 r_A, r_B을 이용하여 다음과 같이 생성한다.

Refresh Key = $K_A=H(K_{master-A}\|r_A\|r_B)$



<그림 3> 초기 인증 프로토콜 (Authentication Initialization Protocol)

4 단계. 갱신키를 N번 반복하여 해쉬 알고리즘을 적용하여 최종 인증키를 생성한다.

5 단계. 생성된 인증 키 값과 갱신키를 생성하기 위해 필요한 난수 r_A을 상대 노드에게 전송한다.

6 단계. 수신 측 노드도 갱신키를 생성한 후 생성된 키 값을 해쉬하여 인증 키 값을 생성한다.

7 단계. 생성된 인증 키 값을 상대 노드에게 전송한다.

8 단계. 상호 인증 처리를 위한 정보 교환 후 대면 확인 등을 통한 상호 신뢰관계(Trust Relationship)가 확립되면 각각의 노드는 TT를 생성하고 노드의 사용자는 상대방 노드에 관한 신뢰수준을 정보를 UMO의 사용자인터페이스를 통해서 입력을 한다 (<그림 4>).

초기 인증이 성공적으로 끝나고 각각의 노드가 TT를 생성을 하면 두 노드는 TT를 교환한다. 두 노드가 상호간에 처음으로 RSC에 참여한 경우라면 노드 'A'의 TT (이하 TT_A)와 TT_B에는 노드 B와 노드 A에 대한 정보만 있기 때문에 TT를 교환한 후에 노드는 자신의 TT를 갱신할 필요가 없다. TT의 교환 및 갱신은 두 가지 경우에 이루어진다. 첫 번째 경우는 두 노드 중에 노드 A가 신규 노드이고 나머지 노드 B가 기존의 RSC에 참여한 노드라면 신규 노드는 인증후에 TT_B를 수신하고 나서 TT_B를 기반으로 하여 TT_A를 갱신하는 경우이다. 두 번째 경우는 인증 갱신 때에 발생할 수 있으며 3.1.3에서 자세하게 설명되어 있다.

<그림 4> 신뢰수준 입력 사용자 인터페이스(예)

앞에서 언급한 바와 같이 사전에 계획하지 않은 애드혹 네트워크를 기반으로 생성한 RSC의 인증은 사전에 인증에 필요한 정보를 공유하지 않고 인증키를 공유하며 인증 과정은 난수의 생성과 해쉬 계산만을 요구하는 경량의 프로토콜이다. 또한 신뢰의 수준은 대면확인을 통하여 정하게 된다. 예를 들어, 사용자 'A'와 사용자 'B'가 학술대회장에서 오프라인 대면 확인을 한다. 이때에 사용자 'A'는 사용자 'B'에 대한 신뢰수준을 자신이 신뢰하는 수준에 따라 0.0~1 0 사이의 값을 입력을 한다. 그리고 자신의 자원을 공유할 때에도 자원에 배정된 최소 요구 신뢰 수준보다 낮은 신뢰수준을 갖는 노드에 대해서는 공유를 하지 않으므로 사용자 'B'가 사용자 'A'의 자원의 남용, 오용 또는 악의적인 공격을 하는 위험 부담을 낮출 수 있다. 따라서 제안하는 인증 기술과 접근 제어 정책은 과거의 인증서 같은 사전정보 공유를 통한 인증과 역할에 기반을 둔 접근제어(RBAC; Role-Based Access Control)와 같은 미세한 접근제어를 할 수 있다.

3.1.3 인증 갱신 (Authentication Refresh)

인증 갱신 목적은 어떤 UMO 'A'가 다른 UMO 'B'를 초기 인증한 후에 UMO 'B' 가 RSC구성원으로서 계속 남아 있는지 확인하기 위한 것이다. 앞서 가정한 바와 같이 자원공유를 위하여 생성된 RSC 참여한 노드들은 처음에 다른 노드와 상호인증하는 절차를 거쳤다 하더라고 RSC내에 남아 있지 않으면 자원을 제공할 수 없으므로 모든 노드는 자신의 TT에 포함된 다른 노드들이 RSC에 있는지 확인하기 위해 정기적으로 확인을 하여야 한다.

<그림 5>의 인증 갱신 절차는 다음과 같으며 그림의 화살표 표시가 있는 원형 심벌(circle with arrow)은 UMO 내에서 계산하는 과정을 표시한다.

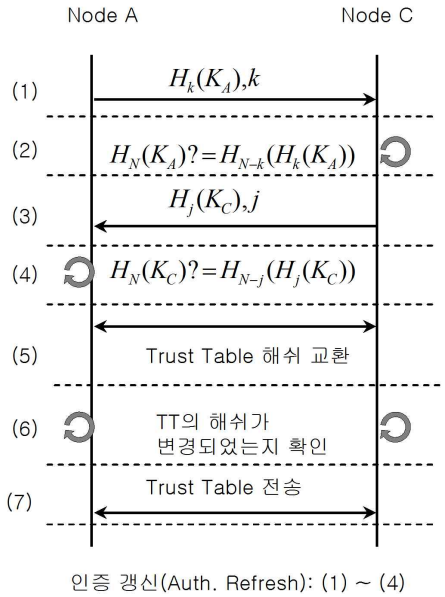
1 단계. 노드 'A'는 자신의 갱신 키(Refresh Key)를 이용하여 인증 토큰 값 $H_k(k_A)$을 생성한 후에 $H_k(k_A)$와 k를 동시에 노드 'C'에게 전송한다.

2 단계. 노드 'C'는 수신한 $H_k(k_A)$값을 $(N-k)$회 추가로 해쉬 계산을 하여 기 보유하고 있는 노드 'A'에 대한 인증 키 값 $H_N(K_A)$과 비교하여 인증 갱신을 한다. 즉, $H_N(K_A)?=H_{N-k}(H_k(K_A))$을 수행한다.

3 단계. 노드 'C'는 상호 인증을 위해 자신의 갱신키를 이용하여 인증 토큰 값 $H_j(k_c)$를 생성한 후에 $H_j(k_c)$와 j를 동시에 전송한다.

4 단계. 노드 'A'는 수신한 $H_j(k_c)$값을 $(N-j)$회 추가로 해쉬 계산을 하여 기 보유하고 있는 노드 'C'에 대한 인증 토큰 값 $H_N(K_C)$과 비교하여 인증 갱신을 한다. 즉, $H_N(K_C)?=H_{N-j}(H_j(K_A))$를 수행한다.

인증갱신 과정에서 상대방 노드가 응답이 없거나 인증토큰이 유효하지 않으면 노드는 자신의 TT에서 유효성 항목을 변경(유효성 취소, effectiveness cancellation)



<그림 5> 인증 갱신 프로토콜 (Authentication Refresh Protocol)

하여야 한다. 인증 갱신에서는 초기 인증에서 사용한 N보다 작은 값을 사용하여야 재생공격으로부터 안전하다. 초기 인증시에 최대 카운터가 N=100,000이라고 가정한다. 예를 들어, 첫 번째 인증갱신때에 $H_{999,999}(k_A)$ (k=999,999)를 사용하였다면 두 번째 인증갱신때에는 이전의 k값보다 작은 k값을 사용하여 (예를 들면, $H_{999,996}(k_A)$와 k=999,996) 인증갱신을 한다. 이렇게 하면 공격자에게 $H_{999,999}(k_A)$와 k=999,999가 노출되었다 하더라도 단방향 해쉬함수의 특성상 공격자가 $k<999,999$를 만족하는 $H_k(k_A)$나 k_A를 유추하기가 어려워 재생공격에 안전한 인증 프로토콜이다.

3.2 TRC 기반의 신뢰할 수 있는 RSC 구성

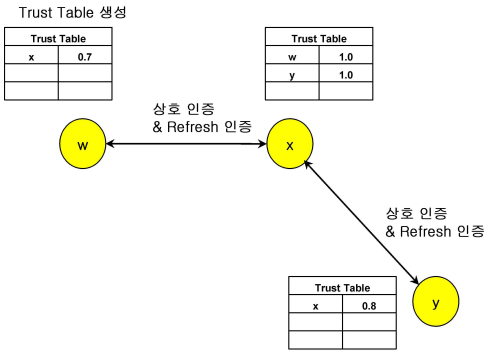
인증갱신의 또 다른 목적은 직접적인 초기 인증 절차를 수행하지 않은 다른 노드에 대한 정보를 이미 인증을 마친 노드를 경유하여 확보하기 위한 기능이라고 할 수 있다. 두 노드는 인증 갱신을 완료한 후에 서로의 TT 해

쉬 값을 교환하고 (<그림 5>의 림 5) 이전의 해쉬 값과 비교하여 다르면 (<그림 5>의 림 6) 상대방 노드의 TT에 새로운 노드가) 되거나 기존의 노드가 삭제된 것으로 판단하여 TT (해쉬 값이 아닌 전체)를 요청하여 값과을 한다 (<그림 5>의 림 7). 예를 들어, 이전의 해와 되거나 기존가 처음에 상호 인증을 한 후에 되거나 기존가 다른 노드나 D해와 상호 인증을 하였다면 이전의 해와 되거나 기존는 갱신 인증을 마친 후에 각각의 TT해쉬 값을 교환하고 노드 'A'는 노드 'C'의 TT를 수신하여 노드 'D'의 참여를 인지하고 TT를 교환한 후에 자신의 TT에 'D'에 과한 정보를 추가하게 된다.

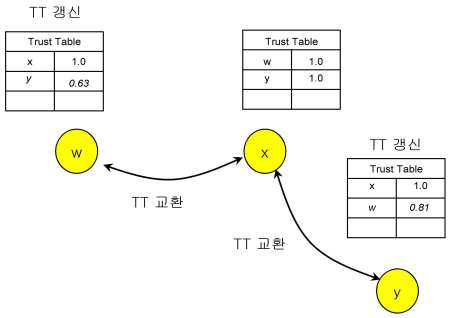
신뢰할 수 있는 네트워크의 구성은 UMO가 상호인증을 할 때에 대면(face-to-face) 접촉을 통하여 입력한 어떤 UMO의 (다른 UMO에 대한) 신뢰수준을 TT의 교환을 통하여 다른 UMO 들에게 전파하는 TRC를 이용하여 구현할 수 있으며 이와 유사한 인증서 사슬(Certificate Chaining)의 개념이 [20]에서 제안되었다.

TRC의 개념을 그림을 통하여 설명하면 아래와 같다. <그림 6>과 <그림 7>의 TT에서는 편의상 다른 항목은 생략을 하고 특정 노드에서 인증을 마친 후에 다른 노드에 부여한 신뢰수준만을 표시하였다. <그림 6>와 <그림 7>에서는 초기에 UMO 'w', UMO 'x', UMO 'y'가 RSC를 생성한 후에 새로운 UMO 'n'이 RSC에 참여할 때에 어떻게 신뢰관계가 사슬로 연결이 되고 신뢰할 수 있는 자원 공유 네트워크가 만들어 지는지를 설명하였다. <그림 6>는 초기에 노드 'w'와 노드 'x', 그리고 노드 'x'와 노드 'y'가 각각 상호인증을 통하여 TT를 생성하는 것을 보여주며 노드 'x'가 노드 'w'를 신뢰하는 신뢰수준 $TL_{x\to w}=1.0$로 가정하였고 나머지 신뢰수준은 $TL_{x\to y}=1.0$, $TL_{w\to x}=0.7$, $TL_{y\to x}=0.8$로 가정하였다.

<그림 7>에서는 노드 'w'와 노드 'x', 그리고 노드 'x'와 노드 'y'가 인증 갱신 때에 TT를 교환하여 노드 'x', 노드 'w', 노드 'y'가 각각 RSC 안에 있는 다른 노드에 대한 정보를 획득하는 것을 보여준다. 노드 'x'는 노드 'w', 노드 'y'와 TT를 교환한 후에도 TT의 갱신은 없다.



<그림 6> 초기 인증을 통하나 RSC 구성



<그림 7> TRC을 통한 Trust Table의 갱신

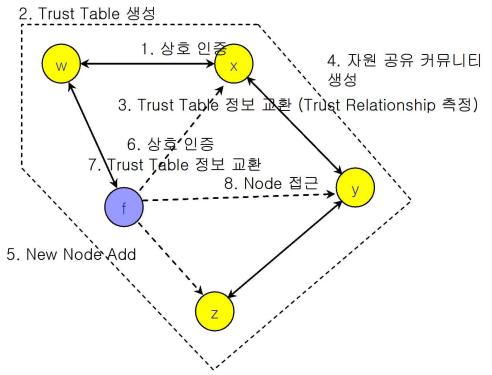
<그림 7>의 노드 'w'의 TT에 나타난 노드 'w'의 노드 'y'에 대한 신회수준 $TL_{w\to y}$은 다음과 같이 계산하며 신뢰도 감소 인자 $d=0.9$로 가정하였다.

$$TL_{w\to y}=TL_{w\to x}\times TL_{x\to y}\times(1-0.1)^{(2-1)}$$
$$=0.7\times1.0\times(0.9)^{1}=0.63$$

또한 RSC에 새로운 노드 'n'이 참여하기 위하여(그림 생략) 노드 'w'와 인증을 거친 후에 노드 'n'과 노드 'w'가 TT를 교환하게 되면 노드 'n'도 RSC 내에 있는 다른 노드에 대한 정보를 획득하게 하게 되어 자신이 직접 인증을 하지 않았더라도 자신이 신뢰하는 노드를 통하여 RSC내의 다른 노드에 대한 신뢰수준을 계산할 수 있다.

3.3 RSC의 구성 절차

앞에서 설명한 내용을 바탕으로 하여 다음 <그림 8>에서는 RSC의 전체적인 구성절차를 설명하였다.



<그림 8> 신뢰성 있는 자원 공유 커뮤니티 구성 절차

1 단계. 노드 'w'와 'x'가 상호 인증 정보 교환 및 오프라인 대면 확인을 통한 신뢰관계(Trust Relationship)를 확립한다. 예를 들어 UMO 'w'는 자신의 기기의 화면에서 UMO 'x'에 대한 보안 신뢰 수준을 0과 1 사이의 숫자로 계량화하여 입력할 수 있다. 노드 'x'와 'y' 그리고 노드 'y'와 'z' 간에도 동일한 절차를 통해 상호 신뢰관계가 확립된다.

2 단계. 노드 'w'는 노드 'x'에 대한 TT를 생성한다. 생성되는 TT는 신뢰관계가 형성된 UMO에 대한 RSC ID, 신뢰수준, 홉 거리, 유효성 여부, 인증키로 구성되어진다.

3 단계. 생성된 TT정보는 인접 노드와 설정된 일정 주기로 상호 교환하여 보유하고 있는 TT정보를 갱신한다.

4 단계. 1~3번 절차를 통해 RSC가 구성이 된다.(노드 'x', 'y', 'z'로 구성된 RSC 생성).

5 단계. 새로운 노드 'f' (또는 UMO)가 RSC에 참여하기를 희망한다.

6 단계. 신규 노드 'f' 는 노드 'w'에 대한 신뢰관계를 확립하기 위하여 대면 확인을 하여 상호 인증 처리를 한다. UMO사용자는 UMO의 응용프로그램에서 다른 UMO에 대한 신뢰 수준(Trust Level)을 설정한다.

7 단계. RSC에 추가된 신규 노드 'f'는 노드 'w'와의 TT 교환을 통하여 자신의 TT를 생성하게 되고 노드 'f'가 생성한 TT에는 노드 'w', 노드 'x', 노드 'y' 그리고 노드 'z'에 관한 정보가 포함되어 있다.

8 단계. 신규 노드 'f'가 다른 노드들과 상호인증을 마치면 노드 'f'는 다른 노드들에게 인지되어 RSC에 추가되어 RSC 회원(Member)으로 구성된다. 주기적인 인증 갱신 메커니즘을 통하여 RSC 구성에 대한 연속성을 보장하고 해당 노드가 RSC를 이탈(Leave)하거나 공격자에 의해 노출된 노드인 경우 인접 노드에 의해 TT 갱신을 통해 RSC로부터 제거된다.

3.4 자원공유를 위한 접근제어에 TRC 적용

상호인증시에 배정한 신뢰수준을 접근제어에 적용하는 것이 가능하다. 자원공유를 위하여 각 노드는 자신이 다른 노드와 공유할 수 있는 자원 테이블(resource table)을 제공하게 된다. <그림 6>과 <그림 7>의 예에서 보면 노드 'w'는 노드 'x'와 직접 상호인증을 하지는 않았지만 TRC를 이용하여 간접적으로 $TL_{w\to y}=0.63$으로 정하였으므로 노드 'w'는 노드 'y'에게 최소 요구 신뢰 수준이 0.63이하인 자원만을 제공하거나 자원테이블 전체를 제공하더라도 노드'y'는 신뢰수준이 0.63이상인 자원을 (요청하더라도) 사용할 수가 없다.

자원 테이블	
공유 자원	최소 요구 신뢰 수준
CPU	0.95
키 보드	0.8
LCD 모니터	0.5

<그림 9> 노드의 자원 테이블(예)

이와 같이 자원을 제공하는 노드는 다른 노드에 대한 신뢰수준을 직접 또는 TRC를 통하여 간접으로 정하고 자원을 사용하기 위한 최소 요구 신뢰수준을 정하여 미세하게 접근제어를 할 수 있다. 또한, 위의 예와 같이 노드 'w'와 노드 'x'가 직접 상호 인증을 하지 않았더라도 TRC를 이용하여 잘 알지 못하는 노드(사용자)에 대한 신뢰수준을 정할 수 있고 이것을 기반으로 다른 노드에 의한 자원의 남용과 오용을 확률적으로 제어할 수 있는 접근제어 기능을 제공할 수 있다.

3.5 TRC 인증의 보안 공격 안전성

제안하는 인증 프로토콜의 안전성은 대면확인 및 단방향 해쉬 함수를 기반으로 하고 있다. 인증을 요청하는 노드는 인증토큰 $H_k(k_A)$와 정수 k를 전송하고 수신 노드는 $H_N(K_A)?=H_{N-k}(H_k(K_A))$를 계산하여 인증요청을 검증한다.

공격자의 가능한 공격방법은 노드 'A'와 노드 'B' 사이에서 재생공격(replay attack)을 이용하여 신분위장(Impersonation)을 하는 것이다. 이 공격에서는 공격자가 노드 B에 침입하여 TT에 저장되어 있는 $H_N(K_A)$를 확보하고 다른 노드와의 인증갱신때에 노드 'A'로 신분위장을 하는 것이다. 그러나 이 경우에 해쉬함수의 특성상 $H_N(K_A)$에서 K_A를 역추정할 수는 없으므로 재생공격에 취약하지 않다.

또 다른 공격 방법은 해커가 노드 'A'를 침입하여 K_A를 확보한 후에 신분위장을 시도하는 것인데 이 경우에도 노드'A'가 노드 'B'에게 마지막으로 전송한 k값을 확보하지 않는 이상은 해커가 노드 'A'의 신분을 위장할 수 는 없다. 또한 마지막으로 전송한 k값을 확보하여도 노드가 어떤 해쉬 함수를 사용하는지 정확히 알지 못하면 신분위장 공격의 가능성은 낮다고 할 수 있다. 마지막으로 자원의 공유와 관련한 위험은 TRC를 통하여 확신하지 못하는 노드에 대하여 자원의 공유를 제공할 때에 최소 요구 신뢰수준을 통하여 미세하게 접근제어를 하여 다른 노드에 의한 자원의 남용과 오용을 방지하는 확률적인 접근제어 기능을 제공할 수 있다.

3.6 기존 인증 프로토콜과의 비교

본 절에서는 제안하는 인증 프로토콜과 앞에서 제시한 두 가지의 선행 연구를 보안 시스템 운영 관점에서 비교하였다. 비교하는 기준은 인증과 자원공유가 얼마나 유연하게 연동할 수 있는지와 보안 시스템이 노드의 추가 또는 이탈에 대하여 얼마나 유연하게 대처할 수 있는지 비교하여 설명하였다.

Asokan[13]이 제안한 "패스워드 인증 키 교환"은 그룹 지향적인 방법(group oriented solution)으로서 그룹 키 협의(Group key agreement)가 가능하지만 각각의 노드에 대한 인증은 제공하지 않는 단점이 있어서 자원의 공유가 노드와 노드 사이에서 이루어지는 RSC에 적용하기에는 적당하지 않다. 또한, RSC를 위한 보안 시스템을 운영하는 도중에 새로운 노드가 추가되는 경우에는 새로운 키를 생성하여야 하는 부담이 있고 이탈하는 노드가 있는 경우에 이 노드가 기존 노드들의 통신을 도청(eavesdrop)하는 것을 방지할 수 있는 대책을 제공하여야 하므로 보안 시스템 운영에 부담이 되는 프로토콜이다.

Stajano[10-11]가 제안한 부활하는 새끼 오리 프로토콜은 모바일 기기의 제조자가 유통망에 배포한 모바일 기기들에게 적용가능한 방법이지만 보안 시스템을 운영하는 운영자는 모바일 기기의 제조업자가 공인인증서 운영기관가 유사한 수준의 신뢰성을 보유한 경우가 아니면 제조업자가 제공하는 인증 기능에 전적으로 의존할 수 없는 단점이 있다. 또한 본 논문에서 제안한 바와 같이 인증과 접근제어를 유기적으로 연동하여 보안 시스템을 운영할 수 없는 단점이 있어서 자원공유를 위하여 즉흥적으로 생성되는 RSC에 적용하기에는 적당하지 않은 기술이다. 프린터나 프로젝터와 같은 보안 수준이 낮아도 되는 경우에는 PDA와 프린터는 위치제한 채널을 이용

하여 사용자가 그들의 PDA를 프린터를 (또는 프로젝터) 향해 접촉하거나 가리키기만 하면 필요한 인증 정보들을 자동으로 교환하게 되고 이를 이용해서 두 기기들 사이에 안전하게 통신을 하는 것이 가능하지만 이런 경우라고 하더라고 서로 인증하는 동안 두 사용자는 아주 근접한 거리에 있어야 하는 단점이 있어서 사용자들이 회의실 같은 장소에서 구성하는 RSC 네트워크에는 적합하지 않다.

본 논문에서 제안한 인증은 RSC가 즉흥적으로 생성되는 경우에도 사전 정보 필요 없이 인증 키와 갱신 키를 교환할 수 있으며 사용자가 개입하여 다른 노드에 대한 신뢰수준을 직접 입력을 하므로 확률적인 인증을 제공할 수 있다. 또한 인증과 자원공유를 위한 최소 요구 신뢰 수준을 결합하여 자원을 요청하는 노드의 신뢰수준에 따라 접근제어를 유연하게 적용할 수 있으므로 선행 연구와 비교하여 확률적으로 안전한 인증과 접근 제어를 제공할 수 있다. 또한 신규 노드의 추가나 노드의 이탈이 발생하는 경우에도 인증갱신을 통하여 노드의 추가나 이탈을 인지가 노드가 신뢰관계사슬(TRC)을 이용하여 자신이 인지한 정보를 네트워크에 전파하므로 선행 연구와 비교하여 RSC 구성원의 추가나 이탈에 대하여 더욱 능동적으로 대처할 수 있다.

Ⅳ. 개선 방향

제안하는 인증은 RSC내에서 하나의 노드는 자신을 제외한 모든 노드에 대하여 P2P 형태로 인증을 하고 하나의 노드에 대하여 네 개의 키(마스터 키, 인증 키, 인증 토큰, 갱신 키)를 연관시켜야 하는 보안프로토콜이다. 또한 RSC에 n개의 노드가 있다면 P2P가 양방향인 점을 고려하면 하나의 노드는 $2^{*}\{n(n-1)/2\}$개의 인증 키를 관리하여야 하므로 RSC내의 노드의 수가 증가하면 임의의 노드가 관리하여야 하는 키의 수가 증가하는 단점이 있다. 제안하는 프로토콜은 계산량이 크지 않으므로 관리하는 키의 숫자가 전체적인 성능에 영향을 주지 않을 것으로 예측되나 악의적인 공격에 대비하여서는 관리하는 키의 숫자를 줄이는 연구가 필요할 것으로 보인다.

Ⅴ. 결론

본 논문에서는 P2P 형태의 RSC를 위한 경량의 인증 프로토콜을 제안하였다. 제안하는 인증프로토콜은 사전에 정보를 공유하지 않고 인증키를 교환하는 프로토콜이다. 또한 이 프로토콜은 상호인증시에 상대방에 대한 신뢰수준을 입력하여 인증시에 정한 신뢰수준을 자원공유 때에 적용하여 미세한 접근제어가 가능한 인증 프로토콜이다.

제안하는 TRC 인증 메커니즘은 노드의 신뢰수준 정보를 RSC내의 다른 노드들에게 전파하도록 하여 직접적인 인증 관계가 없는 두 노드사이에서도 (자원공유를 위하여) 신뢰관계사슬 과정을 통해 효과적인 접근제어를 구현하는 기능을 제공한다.

또한 제안한 인증 프로토콜에서는 RSC에 참여하는 UMO가 다른 UMO에 대하여 초기 인증후에도 인증갱신(Authentication Refresh)을 수행하여 인증의 유효성을 지속적으로 확인함으로써 UMO가 RSC에 신규 가입하거나 이탈하더라도 UMO에 대한 인증 또는 유효성 취소에 관한 정보를 전체 RSC에 전파하여 안전한 RSC를 구성할 수 있도록 하였다.

참고문헌

[1] Prigent, Andreaux, Bidan and Heen, "Secure Long Term Communities in Ad Hoc Networks," Workshop on Security of ad hoc and Sensor Networks archive, 2003 Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, Fairfax, Virginia, Pages: 115 - 124.

[2] Rene Meier, Marc-Olivier Killijian, Raymond Cunningham, and Vinny Cahill, "Towards Proximity Group Communication," Workshop on Middleware for Mobile Computing, Heidelberg, Germany, 2001, Nov, IFIP/ACM, TCD-CS-2001-28

[3] Sye Loong Keoh, Emil Lupu and Morris Sloman, "PEACE : A Policy-based Establishment of Ad-hoc Communities," Proceedings of the 20th Annual Computer Security Applications Conference, IEEE Computer Society. pp. 386-395.

[4] Gerd Kortuem, "A Methodology and Software Platform for Building Wearable Communities," Ph. D. Dissertation Graduate School of the University of Oregon, December 2002.

[5] H. Caituiro-Monge, K. Almeroth, M. del Mar Alvarez-Rohena, "Friend Relay: A Resource Sharing Framework for Mobile Wireless Devices," ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH), Los Angeles, California, September 2006. pp. 20~29.

[6] John Holmstrom, "Authentication and Access Control System in Ad Hoc Networks," Master of Science Thesis Report, Royal Institute of Technology, Stockholm, Sweden, Department of Computer Science(DSV), 2004.

[7] Christian Bergstrom, "Security Architecture for Mobile Ad Hoc Networks," Master of Science Thesis Report, Royal Institute of Technology, Stockholm, Sweden, Department of Computer Science(DSV), 2004.

[8] Klas Fokine, "Key Management in Ad Hoc Networks," Master's Thesis, Linköping University, Department of Electrical Engineering, 2002.

[9] K. Hoeper and G. Gong, Models of Authentication in Ad Hoc Networks and Their Related Network Properties, Technical Report CACR 2004-03, Centre for Applied Cryptographic Research, Waterloo, Canada, 2004.

[10] F. Stajano and R. Anderson. "The Resurrecting Duckling: Security Issues for Ad-Hoc Wire-less Networks," In Proceedings of the 7th International Workshop on Security Protocols, B. Christianson, B. Crispo, J. A. Malcolm, and M. Roe (Eds.), LNCS 1796, Springer-Verlag, 1999, pp. 172-194.

[11] F. Stajano. "The Resurrecting Duckling - what next?," Proceedings of the 8th International Workshop on Security Protocols, B. Christianson, B. Crispo, and M. Roe (Eds.), LNCS 2133, Springer-Verlag, 2000, pp. 204-214.

[12] Balfanz, D, Smetters, D. K., Stewart, P. & Wong, H. C., "Talking To Strangers: Authentication in Ad-Hoc Wireless Networks," 2002 Network and Distributed System Security Symposium Conference Proceedings, 2002.

[13] N. Asokan, P. Ginzboorg. "Key agreement in ad hoc networks.", Computer Communications, 23:1627-1637, 2000.

[14] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography. CRC Press, 1997.

[15] Lee W. McKnight, James Howison, Scott Bradner, "Guest Editors' Introduction: Wireless Grids-Distributed Resource Sharing by Mobile, Nomadic, and Fixed Devices," IEEE Internet Computing, 2004, Volume: 8, Issue: 4, pp. 24~31.

[16] H. Luo, S. Lu, "Ubiquitous and Robust Authentication Service for Ad hoc Wireless Networks," Technical Report 200030, UCLA Computer Science Department, 2000.

[17] J. Clark and J. Jacob, "A survey of authentication protocol literature: Version 1.0" (http://citeseer.ist.psu.edu/cache/papers/cs/166/http:zSzzSzwww.cs.york.ac.ukzSz~jaczSzpaperszSzdrareview.pdf/clark97survey.pdf)

[18] A. Perrig, R. Canetti, D. Tygar, and D. Song, "The TESLA Broadcast Authentication Protocol," Cryptobytes, Volume 5, No. 2 (RSA Laboratories, Summer/Fall 2002), pp. 2~13.

[19] Johann Van Der Merwe, Dawoud Dawoud, Stephen McDonald, "A survey on peer-to-peer key management for mobile ad hoc networks," ACM Computing Surveys, Volume 39, Issue 1, 2007.

[20] S. Capkun, L. Buttyan, "Self Organized Public Key Management for Mobile Ad Hoc Networks," IEEE Trans. Mobile Comput. 2, 1, pp. 52~64.

■ 저자소개 ■

김 정 곤
Jeonggon Kim

1998년 9월~현재 한세대학교 정보통신학과 교수
1997년 8월 Texas A&M University Ph. D. in Electrical Engineering
1989년 6월 Georgia Institute of Technology M.S.E.E
1985년 2월 연세대학교 전기공학과(공학석사)
1982년 2월 연세대학교 전기공학과(공학사)

관심분야 : 전자상거래, 인터넷 에스크로 서비스, 비즈니스 모델, 정보보호, 유비쿼터스 컴퓨팅
E-mail : jeongkim@hansei.ac.kr

김 신 곤
Shinkon Kim

1992년 3월~현재 광운대학교 경영정보학과 교수
1989년 6월 Georgia State University, Ph. D. in Business (경영정보학 박사)
1985년 6월 Georgia State University, MS. in Computer Information Systems (컴퓨터정보시스템 석사)
1982년 2월 서울대학교 대학원 경영학과 (재무관리 석사)
1980년 2월 연세대학교 경영학과 (경영학사)

관심분야 : 경영정보시스템, 시스템 분석 및 설계, R&D 평가, 비즈니스 인텔리전스, 고객관계관리 및 데이터마이닝, 비즈니스 모델링 등
E-mail : shinkon@kw.ac.kr

논문접수일 : 2010년 3월 16일
수 정 일 : 2010년 4월 10일
게재확정일 : 2010년 4월 24일