情報保護學會論文誌
第14券 第3號, 2004. 6

IEEE 802.15.3 High Rate WPAN을 위한 분산된 인증기관을 가지는 PKI 메커니즘 연구

박 정 우,[a)†‡] 양 대 헌[b)], 송 주 석[a)]
연세대학교[a)], 인하대학교[b)]

A Study on PKI Mechanisms with distributed CA for IEEE 802.15.3 High Rate WPAN

Jung-woo Park,[a)†‡] Dae-hun Nyang[b)], Joo-seok Song[a)]
Yonsei University,[a)] Inha University[b)]

요 약

IEEE 802.15.3 High Rate WPAN(Wireless Personal Area Network, 이하 HR-WPAN)은 홈네트워크와 같은 개인 용도의 장치들 사이의 빠른 무선 연결을 지원하기 위해 연구되어 왔다. 장치들의 안전한 사용을 위한 보안 요구에 따라 MAC 계층에서의 대칭키 암호 시스템을 채택하고 있지만, 키를 안전하게 교환하는 방법에 대해서는 언급하고 있지 않다. 또한 앞으로 도래할 유비쿼터스(Ubiquitous) 컴퓨팅 환경의 핵심 기반 기술이 된다는 점에서 하나의 Piconet으로만 구성된 단순한 토폴로지를 벗어나 Child Piconet이 복잡하게 연결된 토폴로지 상에서 같은 Piconet에 속하지 않는 DEV들 간의 Secure Relationship을 맺는 과정에 대한 연구도 필요하다. 따라서 이 논문에서는 공개키 기반 구조를 사용하여 안전한 키 교환을 통한 Secure Membership을 맺는 과정을 연구하였으며, 또한 Child Piconet이 복잡하게 연결된 토폴로지에서 DEV들 간의 Secure Relationship을 맺기 위하여 공개키 기반 구조에 계층 구조를 추가한 프로토콜을 제안하였다.

ABSTRACT

IEEE 802.15.3 High Rate WPAN is a mechanism for wireless home network such as PDAs, digital video camcoder, etc. While symmetric keys are used for MAC layer security, the process of establishing a secure membership or a secure relationship is outside of the scope of the standard. In addition, to prepare for ubiquitous environment in the near future, it is important to study the process of establishing a secure relationship between DEVs in different dependent piconets. This paper proposes a secure model and a process of establishing a secure relationship using PKI without a trusted certificate authority.

Keywords: *HR-WPAN Security, Threshold Cryptography, PKI, Ad-hoc Security*

Ⅰ. 개 요

Ad-Hoc 네트워크에 공개키 기반 암호화 기법(PKI, Public Key Infrastructure)을 사용하려고 할 때, 가장 문제가 되는 점은 신뢰된 인증기관(Trusted Certificate Authority)이 없는 것이다. HR-WPAN은 Ad-Hoc 네트워크의 한 부분으로 신뢰된 인증기관에서 발행하는 인증서(Certificate)를 가질 수 없기 때문에 공개키 기반 암호화 기법을 바로 적용하는데 문제가 있다. 신뢰된 인증기관 없이 공개키 기반 암호화 기법을 사용하는 방법으로 Threshold Cryptography[2]가 있다. Threshold Cryptography에

접수일: 2004년 4월 7일; 채택일: 2004년 6월 7일
†‡ 주저자, 교신저자, lovia@emerald.yonsei.ac.kr

서는 인증기관 역할을 하기 위해 개인키 Share를 가지고 있는 N개의 서버 중에서 K개의 서버가 부분 전자서명을 한 인증서 조각을 모으면 인증기관이 발행하는 인증서와 일치하는 인증서를 만들 수 있다. 이 논문에서 제안하는 HR-WPAN에 사용되는 공개키 기반 암호화 기법은 Threshold Cryptography를 이용한다. 개인키 Share를 가지고 있는 서버는 Piconet을 구성하는 DEV들이며 Piconet마다 각기 다른 개인키를 가진다. 개인키 Share는 인증서를 가지고 있지 않은 DEV가 Piconet에 접속하려고 할 때 DEV의 인증서에 전자서명을 하기 위한 용도로 사용된다.

PNC(Piconet Coordinator)는 Secure Piconet에 참여하는 것이 허락된 DEV목록을 가지고 있다고 가정한다.[1] PNC는 DEV가 Piconet에 참여하기를 희망할 때 이 목록을 참조하여 결정한다. 따라서 DEV를 인증하는 과정은 필요하지 않다. Piconet을 구성하는 DEV들은 DEV가 소속된 Piconet에서 Threshold Cryptography를 이용하여 발행한 자신의 인증서를 가지고 있다. DEV는 자신의 인증서에 있는 공개키를 사용하여 PNC로부터 PNC-DEV management key를 수신하며, 다른 DEV와 Peer-to-peer management key를 교환한다. Child Piconet을 포함한 각 Piconet은 독립적인 개인키/공개키 쌍을 가진다. HR-WPAN의 DEV는 소속된 Piconet의 DEV하고만 점 대 점(Peer to Peer) 통신을 할 수 있으므로 개인키 Share를 직접 전송할 수 없기 때문이다.

II에서는 HR-WPAN에 대한 전반적인 구조를 살펴보며, III에서는 Threshold Cryptography를 이용한 공개키 기반 구조를 적용한 모델을 제안하였다. IV에서는 제안한 공개키 기반 구조를 사용한 전자서명 방법과 계층 구조를 추가한 모델을 제안하였으며, V에서는 성능평가를 하였고 VI에서 결론을 맺고 있다.

II. IEEE 802.15.3 High Rate WPAN

1. MAC 개요

HR-WPAN을 구성하는 기본요소는 DEV이다. DEV 중에서 계산능력, 저장공간, 전원 등의 기능이 가장 뛰어난 DEV가 PNC가 되며, PNC와 최대 255개까지의 DEV가 참여하여 Piconet을 만든다. Piconet에서의 통신은 PNC가 DEV에게 할당해준 CTA(Channel Time Allocation)동안 점 대 점 방식으로 이루어지며, 크기가 작은 데이터나 관리 데이터는 CAP(Contention Access Period)동안 CSMA/CA 방식으로 송수신 될 수 있다. 따라서 하나의 수퍼프레임(Superframe)은 비컨(Beacon) 프레임, CAP, CTAP(CTA Period)로 이루어진다.

Piconet은 Child Piconet과 Neighbor Piconet을 가질 수 있다. 이때 처음 Piconet은 Parent Piconet이며, Child Piconet과 Neighbor Piconet은 Dependent Piconet이 된다. Neighbor Piconet은 한정된 채널을 공유해서 사용하기 위한 방법으로 Parent Piconet이 Neighbor Piconet에게 CTA를 할당해주는 일 이외에는 독립적이다. Child Piconet은 Parent PNC의 계산능력을 분산시키거나 Piconet의 영역을 확장시키기 위한 방법이다. Child Piconet은 Parent PNC로부터 CTA를 할당받으며 Child PNC는 Parent Piconet의 DEV이므로 Parent Piconet의 DEV와 Child Piconet의 DEV 모두와 통신할 수 있다.

Membership은 DEV가 Piconet에 속해 있음을 나타내며 성공적인 결합(Association) 절차가 끝났음을 나타낸다. 또한 DEV는 같은 Piconet에 속한 DEV와 두 DEV만의 Relationship을 맺을 수 있다. Relationship을 맺은 두 DEV는 다른 DEV가 알지 못하는 두 DEV만의 Peer-to-peer management key와 Peer-to-peer data key를 사용한다.

2. Security 개요

HR-WPAN에는 두 가지의 보안 모드가 존재한다. Mode 0은 MAC계층에서의 보안 정책이 시행되지 않음을 뜻한다. Mode 1은 MAC계층에서의 보안 정책이 시행되는 것으로 Mode 1에서만 Secure Piconet을 구성한다. MAC계층에서 사용되는 키는 PNC-DEV management key, Piconet group data key, Peer-to-peer management key, Peer-to-peer data key의 네 가지이며 HR-WPAN 표준화 문서에서는 PNC-DEV management key, Peer-to-peer management key를 교환하여 Secure Membership과 Secure Relationship을 맺는 과정에 대해 상위 계층에서 해결해야할 문제로 남겨놓고 있다.

III. 시스템 설정

PNC는 DEV가 Piconet에 참여하기를 희망할 때, 허용가능 DEV 목록을 참조하여 결정한다. 따라서, DEV를 인증하는 과정은 필요하지 않다. Piconet을 구성하는 DEV들은 DEV가 소속된 Piconet에서 Threshold Cryptography를 이용하여 발행한 자신의 인증서를 가지고 있다. DEV는 자신의 인증서에 있는 공개키를 사용하여 Piconet PNC로부터 PNC-DEV management key를 수신하며, 다른 DEV와 Peer-to-peer management key를 교환하여 Secure Relationship을 형성한다.

1. 시스템 구성

1.1 인증서 생성

PNC는 Piconet의 개인키/공개키 쌍 $SK=\langle d, n\rangle$ / $PK=\langle e, n\rangle$을 생성하여 공개키를 공개한다. 좌표평면상의 K개의 점을 알고 있을 경우 라그랑지 보간법(Lagrange Interpolation)을 사용하여 유일한 $K-1$차 다항식을 정의할 수 있으므로 PNC는 알려지지 않은 $K-1$차 다항식 $f(x)=d+f_1\cdot x+\cdots+f_{K-1}\cdot x^{K-1}$를 사용하여 $N(>K)$개의 장치 DEV_i의 $DEVID_i$(DEV_i의 장치 식별번호)에 대한 함수값 $f(DEVID_i)$를 계산한다. 이때 장치 DEV_i가 가지는 Secret Share는 $P_i=f(DEVID_i) \bmod n$이다.

인증서를 생성하기 위해서는 비밀키 SK를 사용하여 전자서명을 해야 한다. 그러나 한 DEV가 K개의 Secret Share를 수집하여 d를 계산한다면 비밀($SK=\langle d, n\rangle$)이 드러나게 된다. 따라서, 비밀이 드러나지 않게 하기 위해 Secret Share를 가지고 있는 각 DEV가 부분서명한 인증서 조각을 사용한다.

N개의 DEV중 K개의 DEV가 부분서명한 인증서 조각을 사용하여 인증서를 만드는 과정에서 수식 (1)이 계산된다. $l_j(0)$은 라그랑지 계수(Lagrange Coefficient)이다.

$$\sum_{j=1}^{K}(P_j\cdot l_j(0) \bmod n) \equiv \sum_{j=1}^{K} SK_j \pmod n \quad (1)$$

수식 (1)의 $\sum_{j=1}^{K} SK_j \pmod n$는 n에 대한 모듈로 연산에 대해 d와 같은 값을 가지는 값이므로 K개의 인증서 조각을 곱하는 방법만으로는 완전한 인증서를 생성할 수 없다. 따라서 K-bounded Coalition Offsetting[4] 알고리즘을 사용하여 완전한 인증서를 생성한다.

Secret Share를 분배해주는 역할을 분산시키기 위해 Localized self-initialization[4] 알고리즘과 같은 방법을 이용할 수 있다. Localized self-initialization 알고리즘은 Secret Share를 분배할 수 있는 신뢰된 딜러(Trusted Dealer)가 없을 때, Secret Share를 가지고 있는 DEV들이 새로운 DEV를 위한 Secret Share 조각을 만들어 새로운 DEV에게 전송하면 그것을 사용하여 새로운 DEV의 Secret Share를 만들어 내는 방법이다.

1.2 HR-WPAN으로의 적용

1.2.1 MLME-SECURITY-MESSAGE 형식

이 논문에서 제안하는 메시지들을 전송하기 위해 HR-WPAN 표준화 문서의 MLME-SECURITY-MESSAGE를 사용하며, 이 논문에서 제안한 SecurityInformation 인자의 형식을 그림 1에 정의해 놓았다.

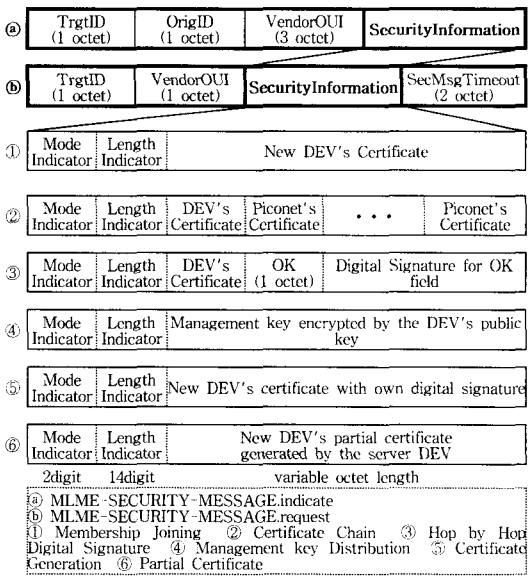


그림 1. MLME-SECURITY-MESSAGE 형식 정의

1.2.2 Secure Membership

HR-WPAN에서는 Secure Membership을 맺기 위한 절차를 상위 계층에서 해결해야할 문제로 남겨 놓고 있다. 그림 2의 *DEV2*는 Piconet에 속해 있지 않은 새로운 DEV이며 *PNC*와 *DEV3*은 Piconet을 구성하고 있는 DEV 들이다. HR-WPAN 표준화 문서에 있는 결합 절차가 성공적으로 끝난 후 *DEV2*가 인증서를 가지고 있지 않을 경우 인증서 생성을 *PNC*에게 요청하며 III.1.2.3에서 제안한 인증서 생성 절차가 이루어진다. 이후 Secure Membership 절차는 그림 2에 제안한 것과 같다.

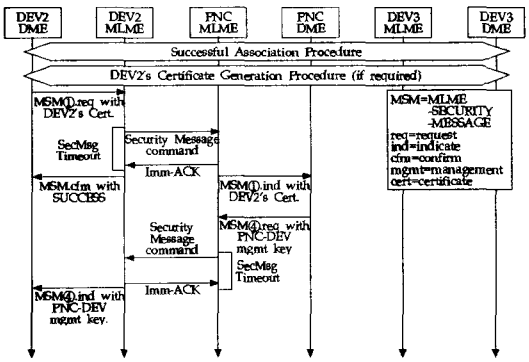


그림 2. Secure Membership 절차

1.2.3 인증서 생성 절차

인증서 생성절차는 새로운 DEV가 Piconet에서 발행한 인증서가 없을 경우 *PNC*에게 요청함으로써 이루어진다. MAC 계층의 성공적인 결합 절차가 끝난 후 인증서 생성 절차가 수행되며 *DEV2*는 자신의 개인키로 전자서명한 인증서를 MLME- SECURITY-MESSAGE ⑤를 사용하여 PNC에게 전송한다. 이후 인증서 생성 절차 과정은 그림 3에 제안한 것과 같다.

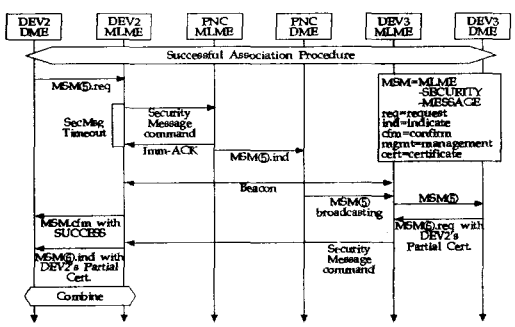


그림 3. 인증서 생성 절차

IV. Secure Relationship을 위한 프로토콜 구조

두 DEV간의 Secure Relationship을 맺기 위해서는 MAC 계층에서 사용하는 Peer-to-peer Management Key를 교환해야 한다. Secure Relationship을 맺는 두 DEV가 같은 Piconet에 속한 DEV 들일 경우 위에서 제안한 공개키 기반 구조 위에서 상대방의 공개키로 암호화하여 전송할 수 있다. 그러나 Dependent Piconet에 속한 DEV와 Secure Relationship을 맺을 경우 각 Piconet에서 사용하는 개인키/공개키 쌍이 다르기 때문에 인증서를 검증할 수 있는 방법이 없으므로 성공적인 Secure Relationship을 맺을 수 없다.

그림 4는 Child Piconet이 포함된 Piconet에서 위에서 제안한 공개키 기반 구조가 적용된 모습을 보여준다. *P*, *P1*, *P2*는 차례로 Parent Piconet, Child Piconet 1, 2이며, 각 Piconet은 각각 다른 개인키/공개키 쌍(예, *SK*/*PK*, *SK1*/*PK1*)을 가지고 있다. DEV는 자신이 속한 Piconet에서 발행한 자신의 인증서(예, C_1DEV_1의 경우 $P1 \ll C_1DEV_1 \gg$)를 가지고 있다. C_1DEV_1이 C_2DEV_2와 Secure Relationship을 맺기 위하여 인증서를 전송했을 경우 C_2DEV_2는 수신한 인증서가 전송도중 훼손되었는지 확인할 수 없다(두 DEV간의 경로는 알려져 있다고 가정한다). 왜냐하면 Piconet *P1*과 *P2*는 다른 개인키/공개키 쌍을 사용하기 때문에 인증서 체인을 연결할 수 없기 때문이다. 따라서 이 논문에서는 인증서 체인을 연결하기 위하여 두 가지 방법, 홉 대 홉 전자서명과 공개키 기반 구조의 계층 구조를 제안한다.

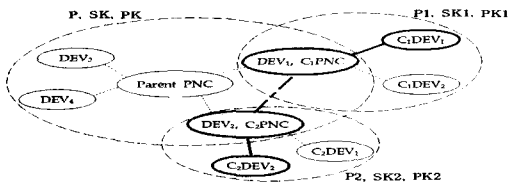


그림 4. Dependent DEV간의 Secure Relationship

1. 홉 대 홉 전자서명

C_1DEV_1과 C_2DEV_2이 상대 DEV의 인증서를 검

증하지 못하는 이유는 인증서에 서명되어 있는 Piconet의 개인키/공개키 쌍이 서로 다르기 때문이다. 만약 라우팅 경로 상에서 라우팅에 참여하는 DEV (PNC)가 인증서의 검증작업을 하고 훼손되지 않았음을 확인하는 표시를 덧붙여 보낸다면 C_1DEV_1과 C_2DEV_2는 상대 Piconet의 공개키를 모르는 경우라 하더라도 수신한 인증서를 검증할 수 있다. 이러한 홉 대 홉 전자서명을 이용하기 위해 그림 1에 MLME-SECURITY-MESSAGE ③을 정의하였으며 추가된 OK표시를 나타내는 필드와 인증서와 OK 표시 필드에 대한 전자서명 필드로 되어있다. 홉 대 홉 전자서명을 사용한 인증서 교환 프로토콜은 그림 5와 같다.

그림 5에서 전자서명을 하는 방법은 Threshold Cryptography를 이용하지 않으며 IN의 개인키로 전자서명을 한다. 라우팅 경로 상에서 다음 홉의 DEV는 항상 자신과 같은 Piconet에 속해 있으므로 전자서명한 IN의 공개키를 항상 알 수 있다. 그림 4의 C_1DEV_1과 C_2DEV_2이 홉 대 홉 전자서명을 사용하여 인증서를 교환하는 과정은 다음과 같다.

- *SDEV* sends MLME-SECURITY-MESSAGE ①
  - to *DDEV* if *DDEV* is in the same piconet, or
  - to *SDEV*'s PNC
- A intermediate node(*IN*) checks the certificate and/or the digital signature : FAIL
  - Drop the MLME-SECURITY-MESSAGE
- A intermediate node(*IN*) checks the certificate and/or the digital signature : SUCCESS
  - *IN* sends MLME-SECURITY-MESSAGE ③ with 〔*SDEV*'s *piconet*≪*SDEV*≫, OK, Digital Signature using IN's private key〕
- *DDEV* sends MLME-SECURITY-MESSAGE ①
  - to *SDEV* if *SDEV* is in the same piconet, or
  - to *DDEV*'s PNC

그림 5. 홉 대 홉 전자서명을 사용한 인증서 교환 프로토콜

가) C_1DEV_1은 MLME-SECURITY-MESSAGE ①을 C_1PNC에게 전송한다.

나) C_1PNC는 $PK1$을 사용하여 수신한 인증서를 확인하고 OK표시와 자신의 개인키 SK_{C_1PNC}를 사용한 전자서명을 포함시킨 MLME-SECURITY-MESSAGE ③ 〔$P1 \ll C_1DEV_1 \gg$, OK, SK_{C_1PNC} 전자서명〕을 C_2PNC에게 전송한다.

다) C_2PNC는 수신한 OK 전자서명을 확인하고 〔$P1 \ll C_1DEV_1 \gg$, OK, SK_{C_2PNC} 전자서명〕이 포함된 MLME-SECURITY-MESSAGE ③을 C_2DEV_2에게 전송한다.

라) C_2DEV_2는 $P1 \ll C_1DEV_1 \gg$를 직접 확인할 수는 없지만, OK표시와 SK_{C_2PNC}를 사용한 전자서명을 확인함으로써 $P1 \ll C_1DEV_1 \gg$이 훼손되지 않았음을 확신할 수 있다. C_2DEV_2는 MLME-SECURITY-MESSAGE ①을 C_2PNC에게 전송한다.

마) C_2PNC는 $PK2$을 사용하여 수신한 인증서를 확인하고 OK표시와 자신의 개인키 SK_{C_2PNC}를 사용한 전자서명을 포함시킨 MLME-SECURITY-MESSAGE ③ 〔$P2 \ll C_2DEV_2 \gg$, OK, SK_{C_2PNC} 전자서명〕을 C_1PNC에게 전송한다.

바) C_1PNC는 수신한 OK 전자서명을 확인하고 〔$P2 \ll C_2DEV_2 \gg$, OK, SK_{C_1PNC} 전자서명〕이 포함된 MLME-SECURITY-MESSAGE ③을 C_1DEV_1에게 전송한다.

사) C_1DEV_1은 $P2 \ll C_2DEV_2 \gg$를 직접 확인할 수는 없지만, OK표시와 SK_{C_1PNC}를 사용한 전자서명을 확인함으로써 $P2 \ll C_2DEV_2 \gg$이 훼손되지 않았음을 확신할 수 있다.

2. 계층 구조를 이용한 인증서 체인

홉 대 홉 전자서명을 이용하는 방법은 MLME-SECURITY-MESSAGE의 크기를 일정하게 유지시켜 일정한 데이터 양만 증가시키지만 라우팅에 참여하는 모든 DEV가 RSA 개인키/공개키 연산을 수행해야 하므로 계산 능력과 시간을 요구하게 된다. 이러한 RSA 개인키/공개키 연산을 수행하지 않기 위해 인증기관 역할을 하는 Piconet의 공개키 기반 구조의 계층 구조를 제안한다.

제안하는 계층 구조는 Parent Piconet과 Child Piconet이 서로 상대 Piconet의 인증서를 발행하는 방법이다. 그림 4에서 Parent Piconet P는 Child Piconet $P1$과 $P2$의 인증서 $P \ll P1 \gg$, $P \ll P2 \gg$을, Child Piconet $P1$은 $P1 \ll P \gg$을, Child Piconet $P2$는 $P2 \ll P \gg$를 발행한다. Piconet 사

이의 인증서를 상호 발행할 수 있는 이유는 Child Piconet의 PNC가 Parent Piconet, Child Piconet 둘 모두와 Secure Membership을 맺고 있기 때문이다. Piconet에 대한 인증서를 발행하는 절차는 Child Piconet을 구성하는 절차가 끝난 후 실행되며, DEV의 인증서 생성 절차와 같다.

제안한 계층 구조를 사용하여 전체 인증서 체인을 구성하는 프로토콜은 그림 6과 같다. 이때, *SDEV*가 *DDEV*와 Secure Relationship을 맺기를 원하는 경우이며, HR-WPAN의 특성에 따라 *SDEV*와 *DDEV*가 아닌 라우팅 경로상의 DEV는 자신의 Piconet을 가지고 있는 PNC이다.

제안한 인증서 체인을 사용한 Secure Relationship 프로토콜을 사용하여 그림 4의 C_1DEV_1이 C_2DEV_2와 Secure Relationship을 맺는 과정은 다음과 같다.

- *SDEV* sends MLME-SECURITY-MESSAGE ① with [*SDEV's piconet*≪*SDEV*≫]
  - to *DDEV* if *DDEV* is in a same piconet, or
  - to *SDEV's* PNC
- A intermediate node(*IN*) appends the certificate to MLME-SECURITY-MESSAGE ②
  - The certificate : [*IN's piconet*≪*IN's parent piconet*≫] if next hop DEV is in the child piconet, or
  - The certificate : [*IN's parent piconet*≪*IN's piconet*≫]
- *DDEV* sends MLME-SECURITY-MESSAGE ① with [*DDEV's piconet*≪*DDEV*≫]
  - to *SDEV* if *SDEV* is in a same piconet, or
  - to *DDEV's* PNC

그림 6. 인증서 체인을 사용한 인증서 교환 프로토콜

가) C_1DEV_1은 자신의 인증서 $P1 \ll C_1DEV_1 \gg$이 포함된 MLME-SECURITY-MESSAGE ①을 C_1PNC에게 전송한다.

나) C_1PNC는 $P \ll P1 \gg$를 추가한 MLME-SECURITY-MESSAGE ②를 C_2PNC에게 전송한다.

다) C_2PNC는 $P2 \ll P \gg$를 추가한 MLME-SECURITY-MESSAGE ②를 C_2DEV_2에게 전송한다.

라) $P2 \ll P \gg P \ll P1 \gg P1 \ll C_1DEV_1 \gg$이 포함된 MLME-SECURITY-MESSAGE ②를 수신한 C_2DEV_2는 전송 중 훼손되지 않았음을 확인할 수 있다.

마) C_2DEV_2는 자신의 인증서 $P2 \ll C_2DEV_2 \gg$이 포함된 MLME-SECURITY-MESSAGE ①을 C_2PNC에게 전송한다.

바) C_2PNC는 $P \ll P2 \gg$를 추가한 MLME-SECURITY-MESSAGE ②를 C_1PNC에게 전송한다.

사) C_1PNC는 $P1 \ll P \gg$를 추가한 MLME-SECURITY-MESSAGE ②를 C_1DEV_1에게 전송한다.

아) $P1 \ll P \gg P \ll P2 \gg P2 \ll C_2DEV_2 \gg$이 포함된 MLME-SECURITY-MESSAGE ②를 수신한 C_1DEV_1은 전송 중 훼손되지 않았음을 확인할 수 있다.

V. 성능 평가

제안한 구조를 검증하기 위해 시스템을 구성하는데 추가적으로 요구되는 데이터의 양과 시간과 인증서 체인, 홉 대 홉 전자서명을 사용하여 Secure Relationship을 구성하기 위한 데이터의 양과 시간을 계산하였다. 연산장치의 성능에 따른 RSA 키 계산 시간을 표 1에 나타내었다.[4]

표 1. RSA와 인증서 계산 시간 (K=5)[1]

key (bit)	SPEC = 20.5			
	RSA-PK (msec)	RSA-SK (sec)	PCC (sec)	combine (sec)
512	0.093	0.0056	0.0466	0.0928
768	0.124	0.0173	0.1198	0.2416
1024	0.142	0.0386	0.2610	0.5280
1280	0.136	0.0669	0.4590	0.9742
1536	0.133	0.1089	0.7944	1.5598
2048	0.208	0.2462	1.7058	3.4410
key (bit)	SPEC = 12.1			
	RSA-PK (msec)	RSA-SK (sec)	PCC (sec)	combine (sec)
512	0.884	0.0678	0.1835	0.1982
768	1.276	0.2165	0.5973	1.3430
1024	1.324	0.4672	1.1637	1.1978
1280	1.356	0.8734	2.2912	2.4109
1536	1.416	1.4863	3.5820	3.6952
2048	1.036	3.1883	7.7855	8.0324
key (bit)	SPEC = 1.37			
	RSA-PK (msec)	RSA-SK (sec)	PCC (sec)	combine (sec)
512	2.782	0.2347	0.5499	0.6144
768	3.382	0.6403	1.4818	1.6478
1024	4.036	1.2953	3.1738	3.3283
1280	4.065	2.4607	5.5492	5.9019
1536	3.941	3.8545	10.1253	10.4301
2048	3.954	8.3826	20.6606	21.7095

1) SPEC:SPECint95
(20.5≒PentiumIII500, 1.37≒SPARCstation5/85)[4]
RSA-PK:공개키 계산시간 RSA-SK:개인키 전자서명시간
PCC:Secret Share로 전자서명 하는 시간.
combine:인증서 조각을 사용하여 인증서를 만드는 시간

표 2. K에 따른 인증서 계산 시간 (key = 1024bit)

K	SPEC = 20.5		SPEC = 12.1		SPEC = 1.37	
	PCC (sec)	Combine (sec)	PCC (sec)	Combine (sec)	PCC (sec)	Combine (sec)
2	0.260	0.526	1.293	1.334	2.991	3.304
3	0.261	0.528	1.149	1.171	2.998	3.293
5	0.261	0.528	1.164	1.198	3.174	3.328
7	0.263	0.531	1.140	1.207	3.163	3.530
10	0.262	0.537	1.309	1.410	3.099	3.394
20	0.261	0.532	1.308	1.464	3.078	3.458
30	0.261	0.537	1.160	1.510	3.082	3.410

표 3. MLME-SECURITY-MESSAGE의 크기

MSM 유형	메시지 크기(byte)
MSM ①	681
MSM ②	681+675 × 인증서 개수
MSM ③	778
MSM ④	102
MSM ⑤	681
MSM ⑥	681

표 2는 K값의 변화에 따라 인증서 조각을 생성하는 데 요구되는 시간과 인증서 조각을 사용하여 완전한 인증서를 생성하는 데 요구되는 시간을 측정한 표이다.[4]

추가로 발생하는 데이터 양을 계산하기 위해 X.509 인증서의 768bit 키를 가지는 675byte 인증서를 이용하였다.[6] 인증서의 크기를 고려한 MLME-SECURITY-MESSAGE의 크기는 표 3과 같다.

수퍼프레임의 길이는 0~65536μs이며[8], 65536 μs로 고정하였다. 데이터 전송속도는 HR-WPAN의 기본 전송속도인 22Mbps를 사용하였다. 따라서 256개의 DEV가 같은 간격의 CTA를 할당받는 다면 DEV당 256μs의 CTA를 할당받게 되며 704byte를 전송할 수 있다. 하지만 수퍼프레임에 비컨과 CAP를 위한 시간이 존재하므로 한 CTA당 실제 전송 속도는 704byte보다 작게 된다.

1. 인증서 생성을 위해 요구되는 시간과 데이터 양

인증서를 생성하기 위한 절차는 그림 3에 나타낸 것과 같으며, 인증서를 생성하기 위한 시간(CGT)은 수식 (2)와 같다.

$$CGT = PCC + 3\times Superframe + Combine \quad (2)$$

수식 (2)의 CGT는 인증서 조각을 생성하기 위해 부분 전자서명을 하는 시간(PCC), 인증서 조각을 전송하기 위한 수퍼프레임, 전송 받은 인증서 조각을 Combine하는 시간으로 이루어져 있다.

전송되는 데이터의 양(CGD)은 수식 (3)과 같으며, MLME-SECURITY-MESSAGE ⑤가 N개의 DEV에 전송될 때의 데이터 양과 부분 서명된 인증서가 포함된 N개의 MLME-SECURITY-MESSAGE ⑥이 전송될 때의 데이터 양으로 이루어져 있다.

$$CGD = (1 + N)\times MSM\ ⑤ + N\times MSM\ ⑥ \quad (3)$$

2. Secure Membership을 위해 요구되는 시간과 데이터 양

Secure Piconet에 참여하기 위한 과정은 그림 2에 나타낸 절차와 같으며, Secure Membership을 위해 추가적으로 요구되는 시간(SMT)은 수식 (4)와 같다.

$$SMT = CGT + 2\times Superframe + 2\times RSA\ PK \quad (4)$$

수식 (4)의 SMT는 인증서 생성 시간(CGT), MLME-SECURITY-MESSAGE ①과 ④를 전송하기 위한 수퍼프레임, 공개키를 사용하여 인증서를 확인하고, PNC-DEV management key를 암호화하는 시간으로 이루어져 있다.

데이터의 양(SMD)는 수식 (5)와 같으며 인증서를 생성하기 위한 데이터 양, MLME-SECURITY-MESSAGE ①, ④로 이루어져 있다.

$$SMD = CGD + MSM\ ① + MSM\ ④ \quad (5)$$

수식 (4)를 이용하여 Secure Membership을 맺기 위해 요구되는 시간을 계산한 그래프는 그림 7과 같다. 그림 7에서 보는 바와 같이 연산장치의 성능

이 SPEC=20.5, SPEC=12.1일 경우 키의 크기가 1024bit 이하일 때 Secure Membership을 맺기에 적합한 시간이 요구되며 특히 SPEC=20.5일 때에는 1초 이내의 시간이 요구되므로 HR-WPAN에서 요구하는 시간을 만족한다. 하지만 키의 크기가 커지면 연산장치의 성능이 빨라야 함을 알 수 있다.

수식 (5)를 이용하여 Secure Membership을 맺기 위해 요구되는 데이터 양을 계산한 그래프는 그림 8과 같으며 이때 사용한 인증서는 768bit 크기의 공개키를 가지는 675byte 인증서이다.

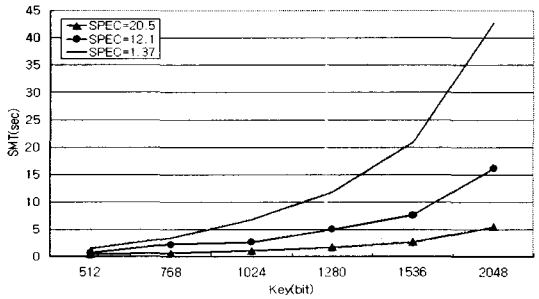


그림 7. Secure Membership을 맺기 위해 요구되는 시간(SMT)

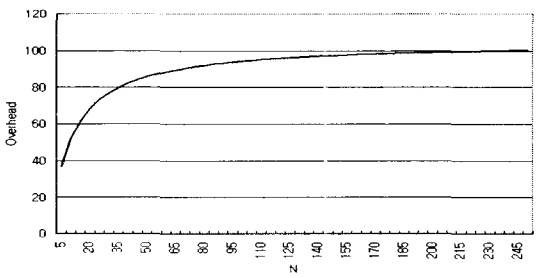


그림 8. Secure Membership을 맺기 위해 요구되는 데이터 양(SMD)

Secure Membership을 맺기 위해 추가로 요구되는 데이터의 양은 DEV가 보안 모드가 적용되지 않은 Piconet에 참여할 때 전송되는 데이터 양과의 비율로 나타내었으며 Piconet 전체에서 증가하는 전체 데이터 양이다. 그림 8에서 보는 바와 같이 인증서를 생성하고 PNC-DEV Management Key를 분배하는 과정은 36.7배에서 100.3배의 추가적인 데이터를 발생시킨다. 이는 Threshold Cryptography에 의해 N개의 DEV에게 675byte의 인증서를 전송하고 다시 수신하는 과정이 있기 때문이다. 250개의 DEV가 Piconet에 결합되어 있을 때 새로운 DEV가 Secure Membership을 맺기 위해 추가로 발생하는 데이터 양은 333.9kbyte이며 한 DEV당 전송 양은 681byte가 된다. 이것은 한 번 또는 두 번의 CTA 동안 전송할 수 있는 데이터 양으로 그림 7에서 보는 바와 같이 적합한 시간 내에 전송할 수 있다.

3. Secure Relationship을 위해 요구되는 시간과 데이터 양

Secure Relationship을 맺기 위한 방법을 두 가지 제안하였다. 인증서 체인을 사용할 때 요구되는 시간($SRCT$)과 홉 대 홉 전자서명을 사용할 때 요구되는 시간($SRHT$)은 각각 수식 (6), (7)과 같다.

$$\begin{aligned} & SRCT \\ &= 2\times\frac{hops\times(hops+1)}{4}\times Superframe \\ &+ hops\times Superframe \\ &+ (2\times hops+1)\times RSA\ PK \\ &+ RSA\ SK \end{aligned} \tag{6}$$

수식 (6)에서 MLME-SECURITY-MESSAGE ②를 목적 DEV까지 전달되기 위해서 각 홉마다 CTA를 할당해야 하며 제안한 인증서 교환 프로토콜에 의해 인증서를 교환한 후 Peer-to-peer management key를 전달하는 과정을 위해 3-way handshake가 필요함을 나타낸다. 또한 MLME-SECURITY-MESSAGE ②는 홉마다 인증서만큼의 크기가 증가하므로 한 개 이상의 수퍼프레임으로 나누어 전송해야 하는 경우가 발생한다. 이것은 한 Piconet을 구성하는 DEV의 수와 관계가 있으며 Dependent Piconet을 포함하는 모든 DEV의 수와도 관계가 있다. 따라서 각 Piconet 마다 15개의 DEV를 가지며 10개의 Piconet으로 구성된 환경을 가정하였다. 나머지 두 항은 인증서를 검증하는 시간과 Peer-to-peer management key를 암호화/복호화 하는 시간이다.

그림 9는 수식 (6)의 홉 수를 변화시키면서 Secure Relationship을 맺는 시간을 나타낸 그래프이며 연산장치의 성능이 SPEC=12.1일 때이다. 인증서 체인을 사용할 경우 인증서 체인의 길이가 늘어남에 따라 요구되는 수퍼프레임의 수가 시간을 증가시키는 가장 큰 원인이다.

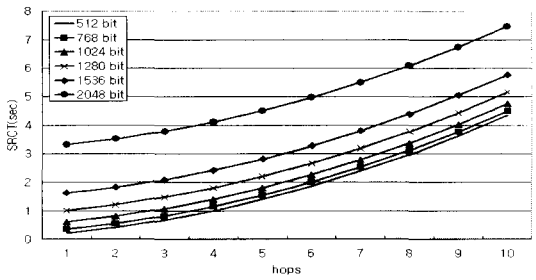


그림 9. 인증서 체인을 사용할 경우 요구되는 시간(SRCT)

$$SRHT = 3\times hops\times Superframe + 2\times(hops-1)\times(RSA\ SK+RSA\ PK) + RSA\ SK + RSA\ PK \quad (7)$$

수식 (7)에서 첫 번째 항목은 3-way handshake를 위한 수퍼프레임의 총 수를 나타낸다. 두 번째 항목은 각 홉마다 전자서명을 하기 위한 개인키 계산과 전자서명을 확인하기 위한 공개키 계산 시간을 나타낸다. 마지막 두 항은 Peer-to-peer management key를 암호화/복호화 하기 위한 시간이다.

그림 10은 홉 대 홉 전자서명을 이용하여 Secure Relationship을 맺기 위해 요구되는 시간을 수식 (7)을 이용하여 홉 수의 변화에 따라 계산한 그래프이다. 연산장치의 성능은 SPEC=12.1을 사용하였다. 홉 대 홉 전자서명을 이용하는 경우는 전자서명을 하기 위한 시간과 전자서명을 확인하기 위한 시간이 가장 큰 원인이다.

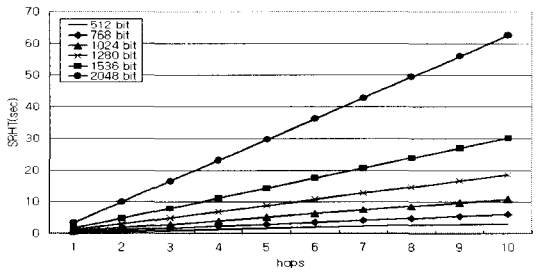


그림 10. 홉 대 홉 전자서명을 사용할 경우 요구되는 시간 (SRHT)

그림 11은 공개키의 크기가 768bit, 1024bit인 경우 인증서 체인과 홉 대 홉 전자서명을 사용하는 방법을 비교한 그래프이다. 그래프에서 보는 것과 같이 인증서 체인을 사용하는 방법이 더 빠른 시간 내에 Secure Relationship을 맺는다. 이것은 인증서 체인을 사용하는 방법이 계산 능력을 많이 요구하는 개인키/공개키 계산을 수행하는 회수가 더 적기 때문이다.

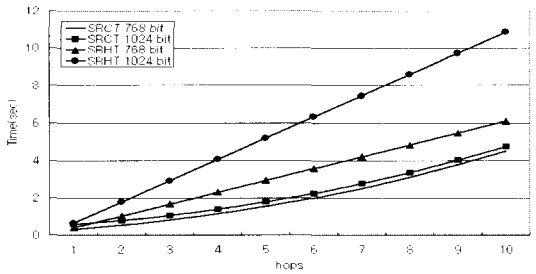


그림 11. SRCT와 SRHT의 비교

수식 (8)과 수식 (9)는 인증서 체인을 사용할 경우(*SRCD*)와 홉 대 홉 전자서명(*SRHD*)을 사용할 경우 요구되는 데이터 양을 각각 나타낸다.

$$SRCD = 2\times\left\{\frac{(hops-1)\times hops}{2}\times Cert. + MSM\ ①\right\} + MSM\ ④ \quad (8)$$

수식 (8)의 첫 번째 항은 라우팅에 참여하는 *IN*이 인증서를 추가하면서 전송함에 따라 증가된 MLME-SECURITY-MESSAGE ②의 데이터 크기를 나타내며 두 번째 항은 MLME-SECURITY-MESSAGE ④를 나타낸다.

$$SRHD = 2\times MSM\ ① + 2\times(hops-1)\times MSM\ ③ + MSM\ ④ \quad (9)$$

수식 (9)의 첫 번째 항은 *SDEV*와 *DDEV*가 처음 보내는 MLME-SECURITY-MESSAGE ①을 나타내며 두 번째 항은 라우팅에 참여하는 *IN*이 전자서명을 추가한 MLME-SECURITY-MESSAGE ③이며 마지막 항은 Peer-to-peer management key를 보내기 위한 MLME-SECURITY-MESSAGE ④이다.

그림 12는 Secure Relationship을 맺기 위해 요구되는 데이터 양을 인증서 체인을 이용하였을 경우와 홉 대 홉 전자서명을 이용한 경우를 비교한 그래프이다. *SRCD*의 경우 홉 수가 증가함에 따라 인증서 크기만큼의 데이터 양이 계속 추가되어 다음 홉으로 전송된다. 따라서 홉마다 일정한 데이터 양을 전송하는 *SRHD*보다 많은 데이터 양이 증가한다.

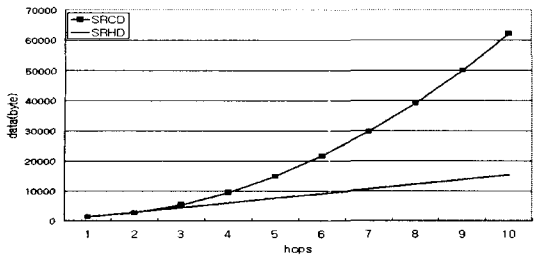


그림 12. SRCD와 SRHD의 비교

그림 11과 그림 12에서는 인증서 체인을 사용한 Secure Relationship을 맺는 과정과 홉 대 홉 전자서명을 사용한 Secure Relationship을 맺는 과정에 대해 시간과 데이터 양에서의 관계를 보여준다. 인증서 체인을 사용하는 경우는 홉 수의 제곱에 비례하여 데이터 양이 증가하며 홉 대 홉 전자서명의 경우보다 전송되는 데이터 양이 많다. 하지만 홉마다 개인키/공개키 연산을 하지 않기 때문에 홉 대 홉 전자서명의 경우보다 Secure Relationship을 맺는데 요구되는 시간은 더 적다.

VI. 결 론

이 논문에서는 개인키/공개키를 사용하여 HR-WPAN의 Secure Membership과 Secure Relationship을 맺는 방법에 대해 기술하였다. 개인키/공개키를 사용하기 위하여 X.509 인증서를 사용하며 인증서에 전자서명을 할 수 있는 신뢰된 인증기관이 없으므로 Threshold Cryptography를 사용하여 Piconet 스스로 인증기관의 역할을 할 수 있는 방법을 제안하였다. 제안한 개인키/공개키 구조를 HR-WPAN에 적용한 후의 검증결과에서 Piconet에 참가하기 위해 요구되는 시간과 추가로 발생하는 데이터의 양 모두 키의 크기가 1024bit 이하였을 때 적합하였다. 같은 Piconet에 속하지 않은 DEV와 Secure Relationship을 맺으려고 할 때 각 Piconet에서 사용하는 Piconet의 개인키/공개키 쌍이 다르므로 인증서를 확인할 수 없는 문제점이 발생한다. 이러한 문제점에 대해 이 논문에서 제안한 방법은 라우팅에 참여하는 DEV의 개인키를 사용하여 홉 대 홉 전자서명을 하는 방법과 공개키 기반 구조에 계층구조를 두어 인증서 체인을 사용하는 방법을 제안하였다. 전자서명을 사용하는 방법은 Secure Relationship을 맺기 위해 요구되는 데이터 양이 더 적은 이점이 있으며, 인증서 체인을 사용하는 방법은 데이터 양은 전자서명을 사용하는 방법에 비해 더 많지만 Secure Relationship을 맺기 위한 시간은 더 적게 요구하였다.

참 고 문 헌

[1] "Draft P802.15.3/D17, Part 15.3: Wireless Medium Access Control(MAC) and Physical Layer(PHY) Specifications for High Rate Wireless Personal Area Networks(WPAN)", February 2003.

[2] Y. Desmedt, Y. Frankel, "Threshold cryptosystems", *Proceedings on Advances in cryptology*. Lecture Notes in Computer Science, pp. 307-315, 1989.

[3] T. P. Pedersen, "A Threshold Cryptosystem without a Trusted Party", *Advances in Cryptology—EUROCRYPT'91*, pp. 522-526, 1991.

[4] J. Kong, P.Zerfos, H.Luo, S. Lu, L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks", *IEEE Computer Society*, Proceedings of the Ninth International Conference on Network Protocols (ICNP'01), PP. 251, 2001

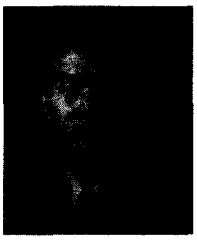
[5] L. Zhou, Z. J. Hass, "Securing Ad Hoc Networks", *IEEE Network*, IEEE Network Magazine vol.13 no.6, 1999.

[6] R. Housley, W. Ford, T. Polk, D. Solo, "RFC 2459, Internet X.509 Public Key Infrastructure Certificate and CRL Profile", *Network Working Group*, IETF, 1999

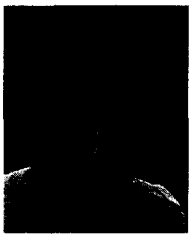
[7] S. Capkun, L. Buttyan, J. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks", IEEE Transactions on Mobile Computing, 2(1), January-March 2003.

[8] "IEEE Standard, Part 15.3: Wireless Medium Access Control(MAC) and Physical Layer(PHY) Specifications for High Rate Wireless Personal Area Networks(WPANs)", September 2003

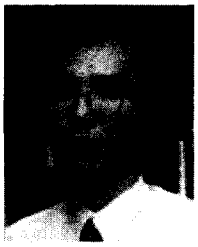
〈著者紹介〉



박 정 우 (Jung-woo Park) 정회원
2003년 2월: 연세대학교 기계전자공학부 컴퓨터과학전공 졸업
2003년 3월~현재: 연세대학교 컴퓨터과학과 석사과정
〈관심분야〉 WPAN 보안, 무선네트워크 보안



양 대 헌 (Dae-hun Nyang)
1994년 2월: 한국과학기술원 과학기술대학 전기 및 전자 공학과 졸업
1996년 2월: 연세대학교 컴퓨터 과학과 석사
2000년 8월: 연세대학교 컴퓨터 과학과 박사
2000년 9월~2003년 2월: 한국전자통신연구원 정보보호연구본부 선임연구원
2003년 2월~현재: 인하대학교 정보통신대학원 전임강사
〈관심분야〉 암호이론, 암호프로토콜, 인증 프로토콜, 무선 인터넷 보안



송 주 석 (Joo-seok Song)
1976년 2월: 서울대학교 전기공학과 졸업
1979년 2월: 한국과학기술원 과학기술대학 전기 및 전자 공학과 석사
1988년 2월: University of California at Berkeley Computer Science Ph.D
1979년 3월~1982년 2월: 한국 전자통신 연구소, 연구 개발
1982년 2월~1982년 6월: 중앙 전기 주식회사, 개발 자문
1985년 9월~1985년 12월: University of California at Berkeley, Teaching Assistant
1985년 12월~1988년 8월: Electronic Research Lab, Research Assistant
1988년 8월~1989년 9월: Naval Postgraduate School, Assistant Professor
1989년 3월~현재: 연세대학교 컴퓨터과학과, 정교수
〈관심분야〉 암호이론, 암호프로토콜, 인증 프로토콜, 유·무선 인터넷 보안