情報保護學會論文誌
第17卷 第3號, 2007. 6

자기 초기화하는 Ad Hoc 네트워크에서의 대리 서명과 임계 서명 기법을 이용한 인증서 발급 기법*

강전일,† 최영근, 김군순, 양대헌‡
인하대학교 정보보호연구실

Certificate Issuing using Proxy Signature and Threshold Signature in Self-initialized Ad Hoc Network

Jeonil Kang,† YoungGeun Choi, KoonSoon Kim, DaeHun Nyang‡
Information Security Research Laboratory, INHA University

요 약

Ad Hoc 네트워크 환경에서 노드 스스로가 자신의 공개키와 비밀키를 생성하고 이를 시스템 인증 권한자에 의해 서명 받는 자기 초기화(또는 자기 조직화) 기법에서 어떻게 인증서를 발급할 것이냐 하는 것은 인증 권한자가 초기에만 네트워크에 존재하는 시스템에서 해결해야하는 중요한 문제 중에 하나이다. 이러한 문제를 해결하기 위해서 앞선 연구들은 자기 초기화 기법에 관련된 연구들에서는 시스템 인증 권한자 자체를 처음부터 시스템에서 없는 것으로 가정하였거나, 인증서를 획득한 노드가 다시 부분 인증 권한자가 되어 다른 노드들을 부분 서명해주는 인증 확산 방식을 사용하였다. 이 논문에서는 대리 서명과 임계 서명 기법을 사용하여 기존의 연구에서 나타났던 여러 문제를 해결하는 방법을 소개하며, 이를 모의실험 등의 방법을 통해 성능과 보안성에 대해서 살펴보았다.

ABSTRACT

In ad hoc network, especially in the environment which the system authority only exists at the beginning of the network, it is very important problem how to issue the certificates in self-initialized public key scheme that a node generates its certificate with public and private key pair and is signed that by the system authority. In order to solve this problem, early works present some suggestions; remove the system authority itself and use certificate chain, or make nodes as system authorities for other nodes' certificates. In this paper, we suggest another solution, which can solve many problem still in those suggestions, using proxy signature and threshold signature, and prove its performance using simulation and analyse its security strength in many aspects.

Keywords : *Certificate Issuing, Self-initialization, Proxy Signature, Secret Sharing, Threshold Signature, Ad Hoc Network*

Ⅰ. 서 론

Ad Hoc 네트워크에서, 인증서의 사용은 네트워크로의 참여 노드가 늘어남에 따라 발생하는 두 노드 사의의 비밀 통신을 위한 키 분배 문제에 대한 좋은 해결책으로 대두되었다. 인증서는 네트워크에 참여하는 어떠

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었습니다. (IITA-2006-C1090-0630-0028)
접수일: 2006년 12월 8일; 채택일: 2007년 2월 28일
† 주저자, dreamx@seclab.inha.ac.kr
‡ 교신저자, nyang@inha.ac.kr

한 노드라도 확인 가능하며 동시에 충분히 안전한 비밀키로 서명되어 있으므로, 두 노드는 이 시스템 비밀키를 근거로 하여 상대방의 인증서를 신뢰할 수 있으며 서로의 공개키와 자신의 비밀키를 사용하여 안전한 통신을 수행할 수 있다. 그러나 어느 한 노드가 시스템 인증 권한자(System Authority)가 되어 혼자서 시스템 비밀키를 소유함으로써 발생하는 보안 문제와 네트워크에 동적으로 참여하는 모든 다른 노드들의 인증서를 발행해 주어야 하는 부하 문제는 인증서를 Ad Hoc 네트워크에 적용하기 힘들게 한다.

이러한 문제들을 해결하기 위해서 앞서 많은 연구가 수행되었으며, 대표적인 해결방법이 노드 스스로가 자신의 공개키와 비밀키를 생성하는 자기 초기화(Self-initialized, 또는 Self-organized) 방법이 있다. 노드 스스로가 자신의 공개키와 비밀키를 생성하게 함으로써 하나의 노드가 혼자 수행해야 했던 일을 다수의 노드에게 분담시킬 수 있었지만, 누가 어떻게 서명해줄 수 있는지는 다른 문제이다. 이러한 문제에서 자기 초기화 기법은 다시 시스템 인증 권한자가 초기에만 존재하는 모델과 처음부터 존재하지 않는 모델이 존재한다. 시스템 인증 권한자가 존재하지 않는 모델에서는 각각의 인증서는 서로 다른 노드에 의해서 서로 다른 비밀키에 의존하여 독립적으로 서명되고 발행되기 때문에 서로가 서로의 인증서를 확인할 수 있는 방법이 추가적으로 필요하며, 이러한 방법에 대해서는 Srdjan Capkun[2]과 Ruidong Li[4]의 연구가 대표적이다. 반면 시스템 인증 권한자가 초기에만 존재하는 모델은 같은 비밀키를 사용하여 서명된 인증서를 모든 노드가 가지고 있으므로 인증서의 확인에 많은 비용이 들지 않으나, 초기 이후 누가 인증서를 발행할 것인가가 문제가 된다. 이에 대한 한 방법으로는 Haiyun Luo의 연구[5]에서 언급한 인증 확산(Diffusion Wave) 기법이 있다.

인증 확산 기법은 여러 부분 서명 인증서를 모아 인증서를 획득한 노드가 다른 노드를 부분 서명할 수 있는 권한을 갖는 방식을 말하며, 그러한 권한을 갖는 노드의 수가 점점 늘어나는 특징을 갖는다. 그러나 이러한 인증 확산 기법은 인증의 순수성에 관한 문제와 이동성을 갖는 공격자가 시스템 비밀키를 획득한 뒤 시스템을 붕괴시키기 위해 다른 곳으로 이동하는 문제가 있을 수 있다. 또한 하나의 노드는 인증서 획득을 위해서 여러 부분 서명 인증서를 각기 다른 노드로부터 얻어야 하기 때문에 요구되는 많은 네트워크 트래픽에 관련된 문제가 있다.

이 논문에서는 위와 같은 문제들을 해결하기 위해 대리 서명(Proxy Signature)과 임계 서명(Threshold Signature) 기법을 동시에 사용하는 인증서 발행 기법을 소개한다.

Ⅱ. 자기 초기화 인증서 관리 기법

2.1. 자기 초기화

자기 초기화란 네트워크에 참여하는 노드가 자신의 공개키와 비밀키를 생성하는 것을 의미한다. 이렇게 자신이 생성한 공개키와 비밀키를 다른 노드와의 통신에 사용하기 위해서는 자신의 공개키가 위조되지 않았음을 증명할 수 있어야 한다. 무선 네트워크에서는 한 홉 거리에 있다고 하더라도 실제로 그 위치에 그 노드가 있는지 별도의 프로토콜이 존재하지 않으면 이를 확인할 수 없으므로, 공개키의 위조 방지를 위해서는 확인할 수 있고 신뢰할 수 있는 비밀키에 의해서 이 공개키가 서명될 필요가 있으며, 보통의 경우 이러한 역할을 인증서가 대신하게 된다.

2.2. 인증서 사슬 방식

만약 시스템에 시스템 인증 권한자가 없어 각기 다른 노드의 비밀키로 인증서가 서명된다면 그 노드의 공개키를 신뢰할 수 있어야 한다. 하지만 일차적으로 신뢰할 수 있는 것은 자신의 공개키 이외에는 존재하지 않는다. 즉, 자신이 서명했던 인증서만이 직접 자신이 신뢰할 수 있다는 것이다. 하지만 인증서를 발급해준 것의 의미가 다시 자신이 서명해준 인증서의 공개키 또한 신뢰 할 수 있다는 의미라면, 어떠한 노드는 다른 노드의 공개키를 신뢰하기 위해서 자신의 공개키로부터 시작하는 인증서 사슬(Certificate Chain)을 생성하여 다른 노드의 공개키를 신뢰할 수 있다. 이렇게 인증서 사슬을 만드는 방법 중에 대표적으로는, 자신이 알고 있는 인증서의 하부 그래프를 그려 이를 두 노드가 교환함으로써 하부 그래프의 교차점을 찾아 인증서 사슬을 만드는 방법[2]과 주위의 노드들에게 인증서를 반드시 발급하도록 하여 라우팅 경로를 역으로 하는 인증서 사슬을 만드는 방법[4]이 있다.

하부 그래프 교차 방식은 인증서 사슬을 찾지 못하는

경우가 발생할 수 있을 뿐만 아니라, 개개의 노드가 메모리에 담고 있어야 하는 인증서의 개수가 지나치게 많은 경우가 있다. 라우팅 경로 방식의 경우 하부 그래프 교차 방식과 같은 문제는 비교적 양호해지나 라우팅 경로를 따라 각각의 노드마다 다음 노드에게 하나의 인증서를 더 첨부하여 보내야 하는 문제가 있다. 비록 일반적인 통신은 대부분 지역적으로 일어난다[3]고 하지만 이는 곧 전체적인 네트워크 트래픽을 증가시킨다는 의미와 같다.

2.3. 시스템 서명 인증서 방식

만약 인증서가 시스템 인증 권한자의 비밀키로 서명이 되었다면 어떠한 인증서도 시스템 권한자의 공개키로 확인할 수 있으며, 보통 이 공개키는 모든 노드들이 알고 있다. 하지만 Ad Hoc 네트워크에서는 노들의 이동성 때문에 시스템 인증 권한자가 항상 자신과 연결된 토폴로지 안에 존재하기를 기대하기란 매우 힘들다. 또한 시스템 인증 권한자는 비밀키를 소유하고 있으므로, 공격자가 시스템 인증 권한자를 공격하여 성공할 경우 시스템 자체가 붕괴될 가능성이 존재한다. 따라서 비밀키는 보안을 이유로 여러 노드에게 분배되어야 하며 비밀키 자체를 소유하고 있는 노드가 존재하지 않는 것이 바람직할 것이다. 비밀키에 대한 정보는 여러 노드에 걸쳐 흩어져 있기 때문에 인증서를 원하는 노드는 부분 비밀키로 서명된 부분 서명 인증서를 모아 인증서를 복원하여 사용할 수 있다. 하지만 초기에 시스템 인증 권한자에 의해서 서명된 인증서와 부분 비밀키를 소유한 몇몇 노드들을 향하여 많은 부분 서명 인증서 서명 요청이 있을 것이고 그에 따라 인증서를 획득하는 시간 지연이 길어지게 된다. Haiyun Luo 등은 인증서를 획득한 노드가 다시 부분 비밀키를 획득할 수 있도록 하여 몇몇 노드들에게 집중되는 부분 서명 인증서 요청을 분산시켜 인증서 획득 시간 지연을 줄이는 방법을 그들의 논문[5]에서 언급하였다.

하지만 이러한 인증 확산 방식에서도 전체적인 네트워크 트래픽을 줄이지 못할 뿐만 아니라, 공격자의 입장에서 시스템 비밀키를 복원하기 위해서 단지 자신의 주위에서 일정 수준 이상의 노드만 공격하면 된다.(물론, 이는 하나의 시스템 인증 권한자가 존재하는 것보다 훨씬 높은 보안 수준을 갖는다.) 또한, 인증 확산 방식은 인증의 권한을 인증 받은 노드에게 나누어주는 것과 같으며, 인증을 위한 규칙이 매우 정교하지 않다면 이는 곧 의도하지 않은 방향으로 인증 규칙이 변질될 가능성이 있음을 의미한다.

2.4. 문제 해결을 위한 이 논문에서의 접근 방향

인증서 사슬 방식은 서로가 서로를 도와 통신을 수행하는 Ad Hoc 네트워크의 성질과 잘 들어맞는 것처럼 보인다. 다시 말하자면, 주위에 도와주는 노드가 존재하지 않으면 인증서 사슬 방식에서 노드는 안전한 통신을 수행할 수 없다. 또한 자신을 위해서 뿐만 아니라 다른 노드들의 통신을 위해서도 추가적인 비용을 지불해야하며, 심지어 일어날지도 모르는 통신을 위해서 주변 노드들의 인증서를 가지고 있어야 한다.

두 노드 간 통신의 유연성을 위해서는 시스템 비밀키로 서명된 인증서를 가지고 있는 것이 여러모로 유리하다. 단지 패킷만 전달해주는 도움만으로도 두 노드는 통신을 수행할 수 있을 뿐만 아니라, 다른 노드를 위해서 자신이 다른 노드들의 인증서를 가지고 있을 필요가 없다. 또한 필요할 때에만 인증서를 요구하면 된다. 하지만 Haiyun Luo 등이 제시한 인증 확산 방식은 인증서 획득 지연 시간을 줄였을지 모르나 위에서 지적했던 여러 다른 문제점을 가지고 있다.

이 논문에서는 대리 서명과 비밀 공유 기법을 동시에 사용하여 앞서 지적하였던 문제에 대한 해결책을 찾는다.

Ⅲ. 제안 프로토콜

3.1. 임계 서명 기법과 대리 서명 기법

비밀 공유 기법[8]은 시스템의 공통된 비밀을 여러 참가자가 나누어 갖고, 일정 수 이상의 참가자가 모였을 때 비밀을 복원해 낼 수 있는 기법을 말한다. 이렇게 복원된 비밀은 다시 사용할 수 없으므로, 다른 비밀을 만들어 다시 분배해야할 필요성이 있다. 하지만 많은 응용분야에서 복원된 비밀의 재사용은 보안측면에서 바람직하지 않기 때문에 비밀을 복원하지 않으면서 비밀을 사용한 것과 같은 효과가 있어야만 했다.

임계 서명 기법[1]은 비밀이 여러 참가자들에게 분산된 시스템에서 그 비밀로 문서를 서명 받고자 할 때 여러 참가자들에게 나누어진 부분 비밀을 이용하여 서명

한 후 이 부분 서명(Partial Signature)들을 모아 원래 비밀로 서명한 것과 동일한 서명을 얻어낼 수 있는 방법이다. 이 과정에서 비밀은 복원되지 않는다. 이 논문에서 제안하는 프로토콜에서 임계 서명 기법은 대리 서명자에게 대리 서명키를 분배하기 위해서 사용된다.

대리 서명[6]은 기본적으로 원 서명자(Original Signer)와 대리 서명자(Proxy Signer), 그리고 검증자(Verifier) 세 참여자가 존재한다. 원 서명자는 자신의 서명 권한을 대신할 수 있는 대리 서명자를 선정하여 위임한다. 원 서명자가 대리 서명자가 생성하는 것과 같은 동일한 대리 서명을 생성할 수 있는지 여부에 따라 비보호-프록시(원서명자 또한 대리서명을 생성할 수 있다)와 보호-프록시(대리서명자를 제외한 그 누구도 대리서명을 생성할 수 없다) 두 가지 방식으로 나뉜다[9]. 원 서명자가 유효한 대리서명을 생성하는 것은 대리 서명자의 역할이 명확히 분담되지 못함을 의미하므로 이 논문에서는 보호-프록시를 사용한다. 또한 우리가 사용하려고 하는 대리 서명 기법은 Bilinear Paring에 기반하고 있다.

3.2. Bilinear Paring

CDHP(Computational Diffie-Hellman Problem)의 해결은 어렵지만 DDHP(Decision DHP)의 해결은 쉬운 그룹 $\mathbb{G}$가 있을 때, 이 그룹을 GDH(Gap DH) 그룹이라고 정의한다. 이러한 문제를 GDHP(Gap DHP) 라고 하는데[10] 이런 GDHP 특성을 만족하는 예로 Bilinear Pairing[11]이 있다.

$\mathbb{G}_1$, $\mathbb{G}_2$는 위수(Order)가 소수 q인 순환군이고 P는 $\mathbb{G}_1$의 생성원(Generator)이다. $\mathbb{G}_1$은 덧셈군이고 $\mathbb{G}_2$는 곱셈군이라면 Bilinear Pairing는 타원 곡선 위에서 다음의 성질을 만족하는 맵 $\hat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$이다.

Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$,
$\hat{e}(P+Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$

Non-degenerate: $\hat{e}(P, Q) \neq 1$인 $P, Q \in \mathbb{G}_1$가 존재한다.

Computable: 모든 $P, Q \in \mathbb{G}_1$에서 $\hat{e}(P, Q)$의 계산이 가능한 효율적인 알고리즘이 존재한다.

이러한 성질을 만족하는 타원곡선 위의 점 P, xP, yP, zP가 주어졌을 때 xyP를 구하는 문제는 어려우나 $xyP = zP$가 성립하는지의 문제는 쉽게 해결할 수 있다.

3.3. 기본 프로토콜 동작

이 프로토콜은 논문 [1]과 [9] 등에서 서명 및 그에 관련된 방법들에 기초한다. 기본적으로 시스템 인증 권한자가 존재해서 네트워크 초기에 네트워크를 위해서 초기 인증 노드들에게 시스템 비밀키를 분산시키거나 대리 서명자에게 대리 서명키를 분배한다. 초기 인증 노드들은 오로지 대리 서명 노드에게 부분 서명키를 발행해주는 역할을 한다. 대리 서명 노드는 일반 노드에게 대리 서명된 인증서를 발행해주며, 일반 노드는 오로지 대리 서명자를 통하여 자신의 인증서를 서명 받아야 한다. (물론, 중복된 역할을 수행할 수도 있다.)

3.3.1. 공용 시스템 파라미터 설정

$PK_o = SK_o P \in \mathbb{G}_1$인 PK_o와 SK_o를 시스템의 공개키와 비밀키로 하고, $PK_p = SK_p P \in \mathbb{G}_1$인 PK_p와 SK_p를 대리 서명 노드의 공개키와 비밀키로 한다. $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q$와 $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$를 서로 다른 종류의 해시 함수라고 할 때 ($\mathbb{G}_1$, $\mathbb{G}_2$, $\hat{e}$, q, P, H_1, H_2)를 공용 시스템 파라미터로 설정한다.

3.3.2. 대리 서명 노드를 위한 대리 서명키의 생성

시스템 인증 권한자는 $a_1, a_2, \cdots, a_{t-1} \in_R Z_q^*$와 비밀키 SK_o를 가지고 $f(x) = SK_o + a_1 x + \cdots + a_{t-1} x^{t-1}$를 만족하는 $(t-1)$차 다항식을 선택한다. 시스템 인증 권한자는 모든 초기 인증 노드(First Initiated Node) $I = \{1,2, \cdots, n\}$, $i \in I$ 에 대해서 $x_i = f(i)$를 나누어 준다. 이때, 임계치(Threshold) t에 대해서 $|S| \geq t$인 모든 부분 집합 $S \subset I$의 모든 원소를 모으면 SK_o를 Lagrange 보간법에 의해서 $SK_o = \Sigma_{i \in S} L_i x_i$처럼 복원할 수 있다. 이때, $L_i = \Pi_{j \in S} -x_j/(x_i - x_j)$이다.

대리 서명 노드를 위한 대리 서명키는 두 가지 방법으로 생성될 수 있다. 시스템 비밀키의 부분을 가지고 있는 초기 인증 노드로부터 부분 비밀키를 모아 대리 서명 노드가 대리 서명키를 생성하는 방법과 시스템 인증 권한자가 직접 자신의 비밀키로 부터 대리 서명키를 생성할 수 있는 정보를 대리 서명 노드에게 주는 방법이 그것이다.

1) 위임장(Warrant) w의 생성

위임장 w는 원서명자의 ID와 대리 서명자의 ID가 들어 있어야 한다. 이 경우 원서명자는 시스템 인증 권한

자로 대리 서명자는 프로토콜의 실시간으로 정해진다. 하지만 임계 서명 기법을 사용하여 위임장 w를 대리 서명 노드를 위해 서명해주기 위해서는 w가 동일해야 한다. 이 때, 위임장 w에 적힌 위임장의 원서명자의 ID와 같은 정보는 시스템 인증 권한자가 사전에 지정하여 사용하고 대리 서명자 ID를 실시간으로 바꾸는 것으로 하나의 대리 서명 노드를 위해서 초기 인증 노드들 간에 실시간으로 추가적인 프로토콜을 사용하여 공통으로 서명할 위임장 w를 생성하는 일을 피할 수 있다.

2) 대리 서명키의 생성

시스템 인증 권한자로부터 대리 서명키의 생성: 시스템 인증 권한자는 위임장 w를 생성한 후, 해시 $H_2(w)$를 계산하고 w에 대한 서명 $SK_{o_w} = SK_o H_2(w)$를 생성하여 대리 서명 노드에게 w와 SK_{o_w}를 전달한다.

초기 인증 노드들로부터 대리 서명키의 생성: 초기 인증 노드는 위임장 w에 대해서 자신의 부분키를 이용하여 $x_i H_2(w)$를 계산한 후 이를 대리 서명 노드에게 전달한다. 대리 서명 노드는 t개 이상의 부분 서명을 합하며 $SK_{o_w} = SK_o H_2(w) = \Sigma_{i \in S} L_i x_i H_2(w)$처럼 대리 서명키를 얻을 수 있다. 이 과정에서 SK_o는 이산 대수 문제가 충분히 어렵다면 복원되지 않는다.

대리서명자는 $\hat{e}(SK_{o_w}, P) = \hat{e}(H_2(w), PK_o)$인지 체크한다. 만약 맞는다면 대리 서명에 이용할 비밀키 $SK_w = SK_{o_w} + SK_p H_2(w) = (SK_o + SK_p) H_2(w)$를 생성한다.

3.3.3. 대리 서명 노드를 통한 인증서의 발급

일반 노드는 대리 서명 노드를 통하여 자신이 생성한 인증서를 서명 받아야 한다. 대리 서명 노드는 서명 받지 않은 인증서 Λ에 대해서 유효한 대리 서명(또는 서명 받은 인증서) $\langle \Lambda, c_p, U_p, w \rangle$을 생성한다. $k_p \in_R \mathbb{Z}_q^*$일 때, $c_p = H_1(\Lambda \| \hat{e}(P,P)^{k_p})$, $U_p = c_p SK_w + k_p P$ 이다. 이때 Λ에는 대리 서명을 해주는 대리 서명 노드의 아이디가 반드시 들어가야 한다. k_p와 $\hat{e}(P,P)^{k_p}$, $k_p P$는 유휴 시간에 미리 계산해 놓을 수 있으므로, 인증서 서명을 위해서 실시간으로는 단지 해시 1번과 타원곡선 위에서의 곱셈연산 1번과 덧셈연산 1번만 수행하면 된다.

3.3.4. 인증서의 검증

일반 노드는 다른 노드와의 안전한 통신을 하기 위해서나 그 외에 다른 이유로 상대방의 인증서를 검증할 필요가 있을 것이다. 상대방의 인증서를 신뢰하기 위해서는 서명 받은 인증서 $\langle \Lambda, c_p, U_p, w \rangle$에 대한 대리 서명을 검증해야 하는데, 대리 서명에 사용된 공개키 $PK_w = PK_o + PK_p$를 생성하고 생성한 PK_w와 $H_2(w)$를 이용하여 $c_p = H_1(\Lambda \| \hat{e}(U_p, P)(\hat{e}(H_2(w), PK_w))^{-c_p})$인지 확인함으로써 이를 검증 할 수 있다.

3.4. 인증서 관리와 관련된 다른 주제들

3.4.1. 초기 인증 노드와 대리 서명 노드, 대리 서명 노드와 일반 노드 사이의 인증

누구나 참여할 수 있는 Ad Hoc 네트워크의 특징으로 말미암아 Ad Hoc 네트워크에서 참여 노드들의 인증 후 행동을 보장할 수 있는 사전 인증 방법은 존재하기 힘들다. 또한 누구나 참여할 수 없고 참여자들이 인증을 위한 비밀 키를 소유하고 있어서, 이를 인증에 사용한다고 하더라도 역시 인증 후 행동을 보장할 수 없는 문제는 여전하다. 즉, 인증은 상대방이 누구인지 확인하는 수단으로써의 의미가 강하며 시스템의 안전성을 완벽히 보장할 수는 없다.

Ad Hoc 네트워크와 같이 누구나 참여하여 서로가 서로를 도와 통신을 하는 환경에서는 상대방이 누구인지 확인할 방법이 없으므로, 인증은 상대방이 누구인지 확인하는 수단이 되기 힘들다. 따라서 Ad Hoc 네트워크에서 어떠한 노드의 인증서 발급을 위한 인증은 그 노드의 행동을 보고 판단해야하며, 이는 후처리 기법들과 유사하다. 특히나 초기 인증 노드들이 대리 서명 노드를 인증하기 위해서는 신뢰할 수 있을 만큼의 기간 동안 해당 노드의 행동을 보고 판단해야만 할 것으로 보인다. 물론, 이러한 '과거에도 신뢰할 만 했으니까 미래에도 신뢰할 수 있을 것이다'는 기준은 절대적일 수 없다. 따라서 단순히 인증 후에 인증서를 가지고 있다고 하더라도, 상대방을 전적으로 신뢰할 수는 없으며 꾸준히 상대방의 행동을 지켜보아야 한다.

3.4.2. 인증서의 폐기

만약 초기 인증 노드들이나 다른 대리 서명 노드들로부터 시스템이나 자신에게 위해가 되는 노드를 발견하였다며, 노드들은 자신이 가지고 있는 인증서를 사용하여 CRL(Certificate Revocation List) 정보를 교환할 수 있을 것이다.(물론, 이를 거절할 수도 있다.) 이 CRL에는 인증서를 발급해준 대리 서명 노드에 관련한 정보가 들어 있고, CRL에 들어 있는 인증서의 발급자가 어느 한

대리 서명 노드에 집중되어 있다면 이 대리 서명 노드의 서명 권한을 박탈하는 등의 조치가 추가적으로 필요하다.

3.4.3. 대리 서명 노드의 자격 박탈 및 재지정

이 일은 초기 인증 노드들이 협력하여 시스템 비밀키로 서명된 보고서(Report)를 발행함으로써 가능하다. 보고서는 부분 서명을 받고 이를 합하여 온전히 서명된 보고서는 얻을 수 있다. 이는 같은 토폴로지의 전체 노드에게 플루딩 되고 보고서에 기입된 대리 서명 노드는 더 이상 통신에 참여할 수 없다. 일반 노드는 자신의 인증서를 서명한 대리 서명 노드가 보고서에 기입된 대리 서명 노드와 같다면 자신의 인증서를 폐기한다.

네트워크 안의 대리 서명 노드의 수가 일정 수준 이하로 내려가게 되면 초기 인증 노드들은 다른 대리 서명 노드를 선정한다. 이는 프로토콜에서 강제될 수 있으며, 초기 인증 노드들이 선정할 수도 대리 서명 노드가 되기를 원하는 노드가 이러한 사실을 초기 인증 노드들에게 알릴 수도 있다.

3.4.4. 인증서의 재발급

인증서의 유효시간이 만료가 되었거나, 인증서의 발급자가 서명 권한을 박탈당했거나 하는 등의 이유에서 자신이 소유한 인증서를 재발급 받고 싶을 때는 다른 대리 서명 노드에게 다시 인증서를 발급 받을 수 있다. 필요에 의해서 다른 대리 서명 노드들에게 하나 이상의 인증서를 미리 발급받을 수도 있을 것이다.

3.4.5. 초기 인증 노드의 권한 이양

초기 인증 노드의 수는 네트워크 안에서 일정 수 이상을 유지해야 한다. 그렇지 않으면 대리 서명 노드를 지정할 수 없기 때문에 대리 서명 노드가 줄어들면 네트워크는 더 이상 유지되지 않을 것이다. 그러기 위해서는 초기 인증 노드가 알고 있는 정보를 다른 노드에게 이양해야 하는데, 이를 위해서는 높은 수준의 인증 절차가 필요할 것으로 보인다.

Ⅳ. 모의실험 및 결과

4.1. 모의실험 환경

모의실험은 OMNeT++[14]를 사용하여 수행되었다. 모의실험에의 주요 파라미터는 다음과 같다.

〔표 4〕 모의실험 주요 파라미터

파라미터	내용
토폴로지 크기	300×300㎡ / 600×600㎡
데이터 링크 계층	IEEE 802.11
라우팅 알고리즘	AODV-UU[7][16]
전송 계층	UDP
이동 방식	Random Way Point
이동 속도	최대 10㎧
노드의 수	50개(일반 노드 35개)
메시지의 길이	4096 비트
메시지 발생 빈도	5~10초 당 한 번
모의실험 시간	최대 43200초

300×300㎡ 환경은 주위에 노드들이 적당량이 존재하는 모델이다. 300×300㎡ 환경에서 노드는 70m 전파 반경을 가지고 있다고 하였을 때 이상적으로는 단지 20개 정도만 있으면 모든 노드들이 연결 상태가 된다. 모의실험에서 50개의 노드 사용은 거의 항상 주변에 자신과 통신할 수 있는 노드들이 존재할 수 있다는 의미와 같다. 하지만 600×600㎡ 환경에서는 90개 정도의 노드가 있어야 한다. 게다가 노드의 위치 변화의 무작위성을 고려한다면 더 많은 개수의 노드가 필요하다. 통신을 위해서 매우 열악한 환경과 같다.

모의실험에서는 1㎧에서부터 10㎧까지 한 단계씩 속도를 증가시켰다. 전파 반경이 74m 정도라는 단순히 하나의 노드가 고정되어 있다고 가정해도 전파 도달 영역의 최대 지름은 148m이므로 다른 노드가 움직임으로써 1㎧에서는 최대 148초까지, 10㎧에서는 최대 14초까지 링크가 유지되는 시간이다.

$$Mean\ Dist. = 2r \times \frac{2}{\pi}\int_0^{\frac{\pi}{2}} \sin\theta d\theta \approx 94.21 \qquad (1)$$

위의 수식은 원 위의 한 점에서 다른 점까지의 평균 거리를 구한 것이며 반지름 r에 74를 대입한 값이다. 따라서 평균적으로는 1㎧에서는 94초, 10㎧에서는 9.4초까지 링크가 유지된다고 볼 수 있다. 그런데 스케줄 빈도에 따른 메시지는 5초에서 10초에 한 번씩 무작위로 발생하므로, 1㎧에서는 10초에 한 번씩 메시지가 발생한다 해도 9.4회의 메시지를 보낼 기회가 생기고, 10㎧에서는 5초에 한 번씩 메시지가 발생한다 해도 1.88회의 메시지를 보낼 기회가 생긴다. 평균 7.5초에 한 번 메시지가 발생하므로, 메시지는 평균 발생 빈도는 1㎧

에서는 12.5회, 10%에서는 1.25회의 기회가 있다고 볼 수 있다.

모의실험에서는 UDP를 사용하여 통신을 수행하였다. 메시지의 무결성을 위해서는 TCP가 일반적으로는 더 올바르나, 무선 통신처럼 잦은 통신 장애가 발생하는 네트워크에서 TCP의 혼잡 제어(Congestion Control)은 오히려 네트워크의 혼잡을 가중시키는 것으로 알려져 있다[12][13]. 이러한 문제를 해결하기 위해서 많은 TCP 프로토콜이 제시되었으나 이를 실제 사용해도 되는지는 확신할 수 없었다. 또한 UDP를 사용한다고 하더라도 응용 계층에서 패킷의 오류가 있을 시 재전송을 시도하는 것으로 메시지의 무결성을 보장해줄 수 있을 것으로 보인다.

기본적으로 프로토콜의 동작에 있어서 서명과 확인에 필요한 시간은 고려하지 않았다. 왜냐하면 대상으로 하는 컴퓨터 환경에서 이를 위해서 사용되는 시간이 응용 계층에서 발생시키는 랜덤 지연 시간보다 작을 것으로 기대하기 때문이다. 랜덤 지연 시간은 물리 계층에서 발생하는 전파 간섭으로 인한 패킷의 유실을 방지하기 위해서 사용되었다. ATmega128과 같은 8-bit CPU의 임베디드(Embedded) 장치에서 타원곡선 위에서의 160 비트의 곱셈에 걸리는 시간은 약 0.81초로 알려져 있다[14]. 일반적으로 Ad Hoc 네트워크에서는 이보다 더 빠른 CPU를 사용할 것이고, 또한 메시지의 발생 주기는 평균 7.5초 정도이기 때문에 인증서의 서명에 걸리는 시간으로 인하여 전체 네트워크의 통신에 지연을 발생할 것으로 보기 힘들다.

4.2. 모의실험을 위한 프로토콜 동작 순서

4.2.1. 인증 확산 프로토콜 동작 순서

이 모의 실험에서 제안 프로토콜의 대조군으로 사용하는 인증 확산 방식에서는 35개의 노드가 각각 15개의 인증된 노드로부터 5개의 부분 서명된 인증서를 얻는 것으로 자신을 인증하며, 자신이 인증된 노드는 다시 다른 노드를 위한 부분 서명된 인증서를 생성할 수 있는 권한을 얻게 하였다.

노드가 네트워크에 참가하였을 때 노드는 어떠한 노드가 인증 권한을 가진 노드인지 모르기 때문에 DW_HELLO 메시지를 브로드캐스트로 주변 노드에게 보냄으로써 자신의 주위에 발급 권한을 가진 노드가 있는지 확인한다.

만약 발급 권한이 있는 노드가 DW_HELLO 메시지를 수신하였다면, 이에 대한 응답으로 DW_REP 메시지를 유니캐스트로 해당 노드에게 전송한다.

DW_REP 메시지를 받은 노드는 자신이 아직 인증되지 못하였다면 DW_REP 메시지를 보낸 노드로부터 부분 서명 인증서를 이미 획득 했는지 확인하고 아직 획득하지 않았다면 인증서 발급에 필요한 정보 담은 DW_REQ_KEY 메시지를 유니캐스트로 해당 노드에게 보내 부분 서명 인증서를 요청한다.

DW_REQ_KEY 메시지를 받은 노드는 자신이 인증된 노드임을 확인하고 부분 서명 인증서를 서명하여 해당 노드에게 DW_REP_KEY 메시지를 유니캐스트로 보낸다.

4.2.2. 대리 서명 프로토콜 동작 순서

제안 프로토콜의 경우 총 50개의 노드 중에서 10개의 노드가 대리 서명 노드에게 키를 줄 수 있는 권한을 갖는 초기 인증 노드로 하고 5개의 노드가 대리 서명 노드로 하였다. 대조군과 동일하게 총 35개의 노드가 자신의 인증서를 발급 받게 하였다. 비교하고자 하는 것은 일반 노드가 인증서를 발급받는데 걸리는 시간이므로 대리 서명 노드는 이미 사전에 대리 서명키를 갖고 있는 것으로 하였다.

일반 노드가 네트워크에 참여하였을 때 대리 서명 노드의 위치를 알 수 없기 때문에 일반 노드는 DN_HELLO 메시지를 브로드캐스트 하여 대리 서명 노드의 위치를 찾는다.

만약 DN_HELLO 메시지를 받은 노드가 일반 노드나 초기 인증 노드일 경우 자신이 알고 있는 대리 서명 노드의 IP 주소를 DN_BROAD_NOTI 메시지를 유니캐스트로 돌려준다.

만약 DN_HEELO 메시지를 받은 노드가 대리 서명 노드인 경우 DN_REP 메시지를 유니캐스트로 반송하여 자신의 존재를 알려준다.

DN_BROAD_NOTI 메시지를 받은 일반 노드는 메시지에 적혀 있는 주소로 자신의 인증서 발급에 필요한 정보를 DN_REQ_CERT에 담아 유니캐스트로 보낸다. 일정 시간을 기다리며 최대 5번까지 DN_REQ_CERT 메시지를 재전송한다. 만약 응답이 없으면 더 이상의 시도를 하지 않는다.

DN_REP 메시지를 받은 일반 노드는 만약 현재 DN_BROAD_NOTI에 의한 DN_REQ_CERT 메시지

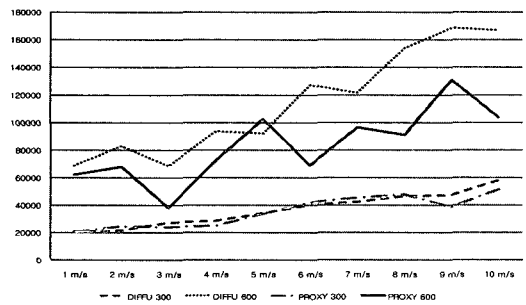
를 시도 중이라면 이 시도를 중단하고 DN_REP를 보낸 노드에게 유니캐스트로 DN_REQ_CERT 메시지를 전송한다.

DN_REQ_CERT 메시지를 받은 대리 서명 노드는 자신의 대리 서명키를 이용하여 인증서를 서명해서 DN_REP_CERT 메시지로 보낸다.

모의실험에 있어서, 두 방식은 35개의 일반노드들이 자신의 인증서를 얻기 위해서 주위와 통신을 시도한다. 두 방식 모두 시스템 인증 권한자는 15개의 노드를 직접 인증했다고 가정하면, 네트워크 토폴로지에는 총 50개의 이동성을 갖는 노드가 존재하게 된다. 인증 확산 방식은 처음에 노드가 자신의 주위에서 비밀을 공유하고 있는 노드를 만날 확률이 3/10이고, 대리 서명 방식은 1/10이다. 대리 서명 방식은 이 상황에 그대로 유지되지만 인증 확산 방식은 지수적으로 증가하게 된다. 하지만 대리 서명 방식에서는 일반노드가 대리 서명 노드를 한 번만 만나면 인증서를 얻어올 수 있지만 인증 확산 방식은 인증서를 가진 노드를 다섯 번 만나야 한다. 각각의 프로토콜에 대해서 이러한 장점과 단점을 가지고 모의실험을 수행했다. 물론, 이러한 설정이 각각의 프로토콜의 성능에 영향을 미치지만, 대리 서명 방식의 경우, 대리 서명 노드의 수를 시스템 인증 권한자가 10개까지(초기 인증 노드와 대리 서명 노드의 역할을 분담하지 않을 때는 15개까지) 조절할 수 있다는 점에서, 이 모의실험은 인증 확산 방식이 더 많은 이득을 가지고 있다고 볼 수 있다.

4.3. 모의실험 결과

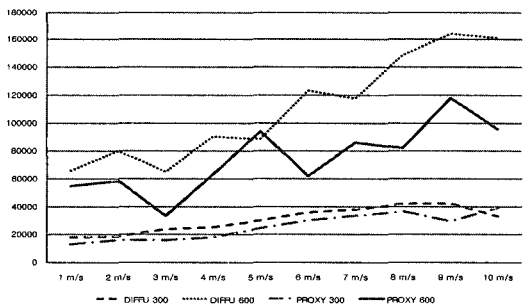
〔그림 2〕는 WLAN에서 받은 총 패킷의 양을 나타낸다. X축은 노드의 이동 속도를 의미하며 Y축은 패킷의 숫자를 의미한다. 300×300㎡의 경우 인증 확산 방식과 대리 서명 방식의 차이가 거의 없는 것으로 나타났다. 하지만 600×600㎡의 경우 인증 확산 방식에 비해 대리 서명 방식이 더 적은 패킷을 보내는 것으로 확인되었다. 전체적으로 노드의 이동 속도가 증가할수록 필요로 하는 패킷의 양이 증가하는 것으로 보인다.



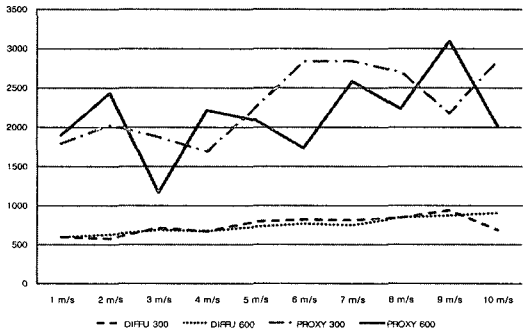
〔그림 2〕 WLAN에서 받은 전체 패킷의 양

〔그림 3〕은 UDP 측에서 보낸 패킷의 총량으로 WLAN 측에서 받은 패킷의 총량과 거의 유사한 결과가 나왔지만 전체적으로 수치가 낮아진 경향이 뚜렷하다. 이러한 차이는 브로드캐스트하는 메시지에 기인한다. 특이하게는 300×300㎡ 환경에서는 미세하게 대리 서명 방식이 인증 확산 방식보다 더 작았다. 하지만 브로드캐스트 메시지가 전체 메시지에서 가장 많은 양을 차지한다고 보았을 때 증가하는 폭이 미미한 것은 브로드캐스트 메시지를 받을 수 있는 노드의 개수가 평균적으로 낮기 때문으로 보인다.

〔그림 4〕는 UDP 측에서 보낸 유니캐스트 패킷의 양이다. 대리 서명 방식이 더 많은 유니캐스트를 시도하는 것으로 알 수 있다. 왜냐하면 DN_BROAD_NOTI 메시지를 사용하여, 대리 서명 노드가 주변에 없다고 하더라도 주위 노드들이 대리 서명 노드의 IP 주소를 가르쳐



〔그림 3〕 UDP에서 보낸 전체 패킷의 양



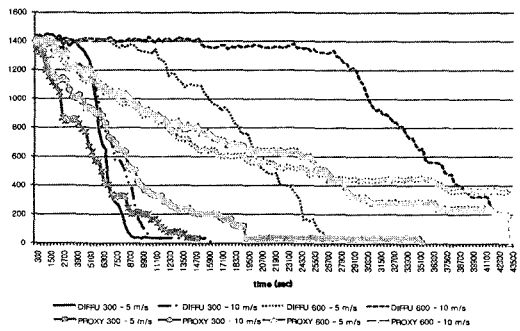
〔그림 4〕 UDP에서 보낸 전체 유니캐스트 패킷의 양

주기 때문에 이러한 IP 주소로의 시도가 증가하기 때문이다.

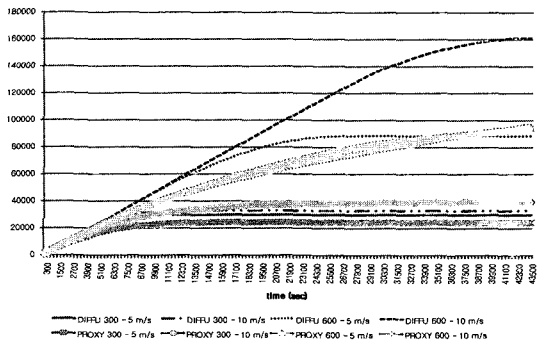
인증 확산 방식은 누군가 자신에게 부분 서명 인증서를 발급해줄 수 있는 노드의 위치를 가르쳐 주지 않기 때문에(너무 많아서 가르쳐 줄 수 없을 뿐더러, 가르쳐 준다 하더라도 그 노드에 관한 정보가 이미 아는 정보가 아니라는 보장도 없으며, 네트워크가 끊임없이 변화하여 그 노드에게 성공적으로 부분 서명 인증서를 얻어올 수 없을지도 모른다.) DW_HELLO 메시지를 통하여 자신이 직접 부분 서명 인증서를 발급해줄 수 있는 노드를 발견해야만 한다. 게다가 복수개의 부분 서명 인증서를 모아야 함으로 더 많은 브로드캐스트 메시지를 보낼 수밖에 없는 것이다.

〔그림 5〕와 〔그림 6〕에서, X축은 시간을 나타낸다. Y축은 누적된 패킷의 양을 나타낸다. 초기에 통신을 수행하는 노드는 모두 35로 동일하므로 두 프로토콜은 초기에 유사한 양의 패킷을 발생시킨다. 300×300㎡ 환경에서 인증 확산 방식과 대리 서명 방식은 유사한 결과를 보인다. 초반에는 대리 서명 방식이 몇몇 노드들에게 더 빨리 인증서를 발급해주어 발생하는 패킷의 양이 빨리 떨어지기 시작하지만 인증 확산 방식에서 점차 부분 서명 인증서를 발급해줄 수 있는 노드의 수가 증가함에 따라 가속적으로 발생하는 패킷의 양이 줄어듦을 확인할 수 있다. 600×600㎡ 환경에서 인증 확산 방식은 네트워크 초기 아주 많은 패킷을 발생시키는 것으로 보이는데, 이는 인증 확산이 초기에 매우 느리게 발생한다는 것을 의미한다. 주위에 노드가 적기 때문에 나에게 필요한 서비스를 제공해줄 수 있는 노드를 만나는 기회가 줄어들고 이로 인하여 많은 시간을 들여 움직여야 한다는 것을 의미한다. 또한 인증 확산 방식은 대리 서명 방식과 다르게 속도에 큰 영향을 받는 것으로 보이는데, 이 이유는 600×600㎡ 환경에서 약 74m의 전파 반경을 갖기 때문에 노드가 10㎧로 움직일 때 단지 하나의 링크는 수 초 정도만 지속되고 또한 주위 노드의 수도 300×300㎡ 환경의 1/4 수준일 것이기 때문으로 네트워크 토폴로지가 굉장히 불안하다는 것이다.

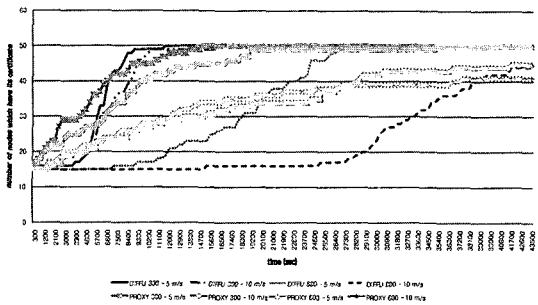
전체적으로 대리 서명 방식은 선형적으로 패킷의 양이 감소하는 것을 볼 수 있으나, 일정한 패킷양이 유지되는 구간이 나타나는 이유는, 노드 이동의 예측 불가능성으로 인하여 몇몇 노드에 대한 환경이 좋지 않은 것



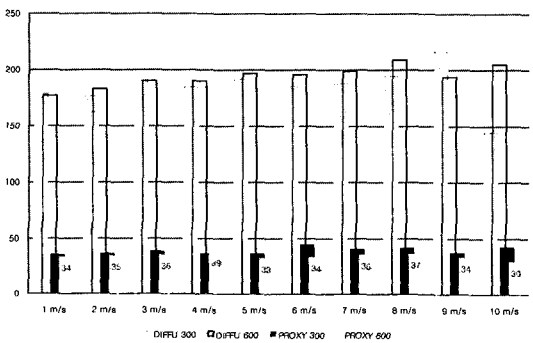
〔그림 5〕 시간에 따른 UDP에서 보낸 300초 누적 패킷의 양



〔그림 6〕 시간에 따른 UDP에서 보낸 총 누적 패킷의 양



〔그림 7〕 시간에 따른 네트워크에서의 인증서를 소유한 노드 개수의 변화



〔그림 8〕 인증서 발급을 위한 서명 횟수 비교

때문으로 보인다. 즉, 노드의 위치에 따라 서비스 받는 환경의 차이가 크다는 것이다.

누적 패킷의 양을 나타내는 그림에서 알 수 있다시피, 인증 확산 방식은 초반에 많은 패킷을 발생시키고 점차 그 기울기가 급격히 줄어든다. 하지만 대리 서명 방식은 조금 더 완만한 기울기를 가지고 있다. 그렇기 때문에 600×600㎡ 환경에서 37,000초 부근에서 총 패킷의 양이 역전되는 현상을 볼 수 있다. 이 이유는 위에서 설명했던 대리 서명 방식의 노드의 위치에 따른 환경의 차이로 설명할 수 있을 것이다.

〔그림 7〕은 시간의 변화에 따른 인증서 개수의 변화를 의미한다. 인증서를 보유한 노드 수의 변화는 패킷의 변화와 직접적인 연관성이 있다. 왜냐하면 인증서를 보유한 노드는 DN_HELLO나 DW_HELLO 메시지를 더 이상 발생시키지 않기 때문이다. 패킷의 양에서도 유추할 수 있다시피 인증 확산 방식은 급격히 인증서를 보유한 노드의 수가 증가하는 것을 볼 수 있다.

결론적으로, 대리 인증 방식은 네트워크가 안정적이고 노드의 이동이 느릴수록 보다 적은양의 패킷을 사용하여 상대적으로 많은 노드들에게 더 빠르게 인증서를 얻을 수 있도록 한다.

〔그림 8〕은 DW_REP_KEY 메시지와 DN_REP_CERT 메시지의 발생 횟수를 나타낸다. 실제로 이 메시지의 발생 횟수는 노드의 서명 횟수와 정확하게 일치한다. 네트워크의 사정으로 인하여 버려지거나 전송되지 않는 메시지가 있고, 미처 모의실험 시간 안에 요청되지 않는 메시지가 있음을 고려할 때 위 그림의 미묘한 차이는 이유 있어 보인다.

인증 확산 방식에서는 한 노드 당 5번의 서명이 필요하므로 175번 이상의 DW_REP_KEY 메시지가 발생하였다. 대리 서명 방식은 한 노드 당 단지 한 번의 서명이 필요하므로 35번 이상의 DN_REP_CERT 메시지가 발생하였다. 이 차이는 약 5배 차이로 모의실험에 사용되었던 파라미터 수치와 일치한다.

V. 제안 프로토콜 평가

5.1. 성능 평가

위 실험 결과에 미루어 보아, 인증 확산 방식은 최초에 인증을 얻는 시간이 많이 소요된다는 것을 알 수 있었다. 그러나 모든 노드들에게 동일하게 적용된 인증 획득의 기회는 인증 속도를 점점 가속화 시키는 효과를 가져 온다. 대신 많은 노드들이 서로 복수의 부분 서명 인증서를 요구하는 과정에서 많은 양의 패킷이 발생하며 전체적인 네트워크 트래픽을 증가시킨다. 또한 좋은 네트워크 환경에서는 전체 노드에 대해서 안정적인 인증 및 인증서 획득의 기회를 주지만 좋지 않은 네트워크 환경에서는 오랜 시간이 지나도 인증을 성공적으로 하기 힘든 단점이 있다.

대리 서명 방식은, 인증 확산 방식보다 더 적은 양의 트래픽을 사용하여 인증을 수행할 수 있으나, 대리 서명 노드의 수가 충분하지 않은 경우, 네트워크에 존재하는 위치에 따라 노드마다 인증서를 얻는데 시간의 차이가 크게 발생함을 알 수 있다.

또한 실험 결과에서는 드러나지 않았지만 대리 서명 노드를 시스템에서 초기화 시키지 않고 단지 초기 인증 노드들만 생성한다고 하였을 때, 초기 인증 노드들이 대리 서명 노드에게 인증서와 대리 서명키를 나누어주는 과정은 초기 인증 노드의 희소성으로 말미암아 쉽지 않을 것으로 보인다. 따라서 이러한 부분을 해결하기 위해서는 대리 서명 노드 또한 시스템에서 초기화시켜주고, 초기 인증 노드는 대리 서명 노드의 취소와 추가만을 수행하게 해야 하는 것이 올바를 것이다.

5.2. 보안성 평가

5.2.1. 공격자의 노드 공격에 따른 보안성

n를 전체 노드의 개수, t를 비밀을 복원하기 위한 임계 수치, x를 비밀을 가지고 있는 노드의 개수, a를 공격자가 고를 수 있는 노드의 숫자라고 하자. 공격자는 주위의 노드를 공격하여 그 노드가 가진 정보를 가져올 수 있다. a개의 노드를 n개의 노드 중에 골랐을 때, 그 안에서 기대할 수 있는 x개의 숫자는 xa/n개이다. 따라서 공격자가 공격을 성공하기 위해서는 $(xa/n) \geq t$이어야 한다. 그러므로 공격에 성공하기 위해서 공격자는 평균적으로

$$a \geq \frac{nt}{x} \tag{2}$$

만큼의 노드를 공격해야만 한다. 인증 확산 방식의 경우 $x \simeq n$이므로 $a \geq t$이면 된다. 즉, 단지 t개의 노드만 공격이 성공한다면 시스템 비밀을 복원해낼 수 있다.

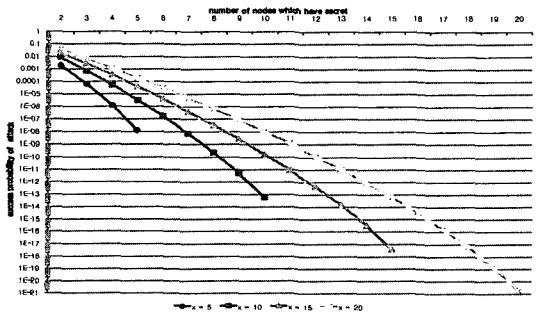
만약, 대리 서명 방식에서 공격자가 비밀을 가진 노

드만을 선택적으로 공격할 수 있다면 인증 확산 방식과 마찬가지로 공격자는 단지 t개의 노드만을 공격하여 성공하면 시스템 비밀을 복원해낼 수 있다. 그러나 만약 공격자가 네트워크 트래픽을 분석할 수 없거나, 비밀을 가진 초기 인증 노드가 매우 조심스럽게 동작하거나 중복된 역할을 수행함으로써 공격자에게 자신이 초기 인증 노드라는 사실을 알아차리지 못하게 하거나, 공격자가 어떠한 사정으로 인하여 주위의 노드들 밖에 공격할 수 없을 경우, 노드를 선택적으로 공격할 수 없다고 하면, 대리 서명 방식의 경우 단지 t개만을 공격에 성공하여 제안 프로토콜을 사용하는 시스템으로부터 공격자가 비밀을 복원할 확률은, 차례대로 선택한 모든 노드들이 비밀을 가지고 있는 노드들이어야 하므로

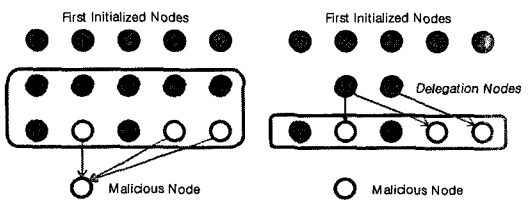
$$\Pr[p] = \frac{x}{n}\frac{(x-1)}{(n-1)}\frac{(x-2)}{(n-2)}\cdots\frac{(x-t+1)}{(n-t+1)} \quad (3)$$

$$= \frac{x!(n-t)!}{n!(x-t)!} = \frac{{}_xP_t}{{}_nP_t}$$

과 같다. [그림 9]는 이러한 확률을 그래프로써 나타낸 것이다. 여기에서 주의해야할 것은, 노드 하나를 공격하여 성공할 확률 항상 1이 아니고 그에 필요한 시간이



[그림 9] 비밀을 가진 노드의 증가와 한계수치에 따른 공격 성공 확률



(a) 인증 확산 방식의 경우 (b) 대리 서명 방식의 경우

[그림 10] 인증 권한의 순수성 문제

충분히 짧다고 볼 수 없으므로 그림에서 보여주는 수치가 $1/2^{64}$이나 $1/2^{128}$ 같은 수치를 의미하지는 않는다.

5.2.2. 인증 권한의 순수성 문제

인증과 인증서를 발급해주는 것은 근본적으로는 다르다. 그러나 인증서를 발급받기 위한 조건이 인증이라고 한다면 인증서를 발급해주는 행위를 인증의 한 부분이라고도 볼 수 있을 것이다. 따라서 대리 서명 노드는 인증 권한을 일부 가지고 있다고 해도 과언이 아닐 것이다. 이런 부분에 있어서 인증 확산과 대리 서명 방식의 인증 권한의 순수성에 대한 논의가 있을 수 있다.

네트워크 설치 이전의 단계에 초기화된 초기 인증 노드들을 1세대 노드라고 하고 네트워크 설치 단계 이후에 진입하여 이 1세대 노드들에 의한 인증단계를 거친 후 네트워크에 합류하게 된 노드를 2세대 노드라 하자. 이러한 2세대 이후의 노드들을 자가 초기화 노드(Self-initialized node)라 한다. 각 세대의 노드를 그 상위 노드들에 의해 인증 받은 노드라고 할 때, 하위 세대로 갈수록 네트워크 생성 시 기대했던 인증권한의 순수성(Originality of Authority)은 떨어지게 될 것이다. 이 말은 세대가 거듭 될수록 인증권한의 안전성 강도가 떨어질 가능성이 있어, 악의적 목적을 가진 노드가 네트워크에 합류할 수 있음을 의미한다. 인증 확산 방식은 이러한 문제점을 가지고 있다.

그러나 대리 서명 방식에서는, 네트워크 설치단계 이전의 인증 권한을 가지고 있는 1세대 노드가 선정한 대리 서명 노드가 그 이후에 네트워크로 진입을 시도하는 노드에게 인증을 해준다. 이 방식에서 새롭게 네트워크에 진입하는 노드들은 대리 서명 노드에 의해 인증은 받으나, 자신이 다른 노드를 인증할 수 있는 권한은 부여받지 않는다. 결국 전체 네트워크상의 노드들은 최대 3세대를 넘지 않는다. 그럼으로써 인증권한의 순수성은 거의 손상되지 않으며 부정노드가 네트워크에 합류할 가능성은 크게 줄어들게 된다.

VI. 앞으로의 과제 및 결론

우리는 이 논문에서 Ad Hoc 환경에서 스스로 초기화하는 인증서 관리 환경에서, 위임장을 발급함으로써, 다른 노드가 대신 서명하게 해주는 대리 서명과 임계치 이상의 노드가 모였을 때 원래의 비밀을 복원할 수 있는 임계 서명 기법을 동시에 사용하는 인증서 관리 기

법을 선보였다. 이는 시스템 인증 권한자에 의해서 초기에 인증된 초기 인증 노드와 초기 인증 노드들에 의해서(또는 시스템 인증 권한자에 의해서) 대리 서명을 수행할 수 있는 대리 서명 노드들로 구성되는 인증서 관리 기법이다. 앞선 실험에 대한 평가와 보안성 분석을 통하여 살펴본 바와 같이, 제안하는 프로토콜은 임계 서명 기법만을 사용하는 인증 확산 방식보다 많은 장점을 갖는다. 임계 서명 기법은 비밀을 분산함으로써 보다 높은 보안을 요구하는 환경에 적합하지만, 이 방법만을 사용했을 경우에는 부분 인증서를 얻어오기 위한 통신량과 인증서 발급에 필요한 연산량 측면에서 불리하다.

그러나 제안하는 프로토콜은 초기에 시스템이 스스로 대리 서명 노드를 설정하지 않는다면 초기 인증 노드들이 이 대리 서명 노드를 만드는데 까지 걸리는 시간이 인증 확산 방식의 인증이 활발히 일어나기 시작하는 시점보다 네트워크 환경적인 문제가 있기 때문에 더 더딜 것으로 예상한다. 실험에서는 인증 확산 방식과 동일한 조건으로 생각할 수 있는 초기 인증 노드와 대리 서명 노드의 수를 정하였고 그에 따라 실험을 수행하였지만, 앞에서 지적한 문제를 해결할 수 있다면 프로토콜의 호환성 측면에서 더 좋을 것으로 기대한다.

참고문헌

[1] Alexandra Boldyreva, "Efficient Threshold Signature, Multisignature and Blind Signature Schemes Based on the Gap-Diffie-Hellman-Group Signature Scheme," Public Key Cryptography, LNCS 2567, pp.31-46, (2003)

[2] Srdjan Capkun, Levente Buttyan, and Jean-Pierre Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," IEEE Transactions On Mobile Computing, Vol. 2, No. 1, (2003)

[3] Jinyang Li, Charles Blake, Douglas S. J. De Couto, Hu Imm Lee and Robert Morris, "Capacity of ad hoc wireless networks," Proc. of ACM Conference on Mobile Computing and Networking, pp.61-69, (2001)

[4] Ruidong Li, Jie Li, Hisao Kameda, and Peng Liu, "Localized Public-Key Management for Mobile Ad Hoc Networks," Proc. of IEEE Global Telecommunications Conference, pp. 1284-1289, (2004)

[5] Haiyun Luo, Jiejun Kong, Petros Zerfos, Songwu Lu and Lixia Zhang, "URSA: Ubiquitous and Robust Access Control for Mobile Ad-Hoc Networks," IEEE/ACM Transactions on Networking, Vol. 12, No. 6, pp.1049-1063, (2004)

[6] Masahiro Mambo, Keisuke Usuda and Eiji Okamoto, "Proxy Signature: Delegation of the Power to Sign Messages," IEICE Transactions on Fundamentals, Vol. E79-A, No. 9, pp.1338-1353, (1996)

[7] Charles Perkins, Elizabeth Belding-Royer, and Samir Das, "Ad Hoc On Demand Distance Vector Routing," RFC3561, (2003)

[8] Adi Shamir, "How to share the secret," Communications of the ACM, Volume 22, Issue 11, pp.612-613, (1979)

[9] Fangguo Zhang, Reihaneh Safavi-Naini, and Chih-Yin Lin, "New Proxy Signature, Proxy Blind Signature and Proxy Ring Signature Schemes from Bilinear Pairings," Cryptology ePrint Archive, (2003)

[10] Tatsuaki Okamoto, David Pointcheval, "The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes," Public Key Cryptography, LNCS 1992, pp.104-118, (2001)

[11] Alfred Menezes, Tatsuaki Okamoto and Scott A. Vanstone, "Reducing elliptic curve logarithms to a finite field," IEEE Transactions on Information Theory, Vol. 39, No. 5, pp.1636-1646, (1993)

[12] T. V. Lakshman and Upamanyu Madhow, "The performance of tcp/ip for networks with high bandwidth-delay products and random loss," IEEE/ACM Transactions on Networking, Vol. 5, No. 3, pp.336-350, (1997)

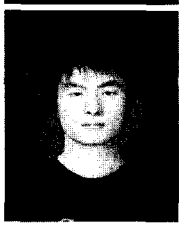
[13] Fabienne Lefevre and Guillaume Vivier, "Understanding tcps behavior over wireless links," Proc. of IEEE symposium on Communications and Vehicular Technology, SCVT-200, pp.123-

130, (2000)
〔14〕 Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle and Sheueling Chang Shantz, "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs," Cryptographic Hardware and Embedded Systems, CHES 2004, LNCS 3156, pp.119-132, (2004)
〔15〕 OMNeT++, http://www.omnetpp.org/
〔16〕 AODV-UU, http://core.it.uu.se/core/

〈著者紹介〉

강 전 일 (Jeonil Kang) 학생회원
2003년 2월 : 인하대학교 컴퓨터 공학과 졸업
2006년 2월 : 인하대학교 정보통신대학원 석사
2006년 3월~현재 : 인하대학교 정보통신공학과 박사 과정
<관심분야> 생체 인식 보안, 무선 센서 네트워크 보안, 무선 인터넷 보안

최 영 근 (YoungGeun Choi) 학생회원
2006년 2월 : 인하대학교 컴퓨터 공학과 졸업
2006년 3월~현재 : 인하대학교 정보통신대학원 석사
<관심분야> 무선 센서 네트워크 보안



김 군 순 (KoonSoon Kim) 학생회원
2006년 8월 : 인하대학교 컴퓨터 공학과 졸업
2006년 9월~현재 : 인하대학교 정보통신대학원 석사
<관심분야> 생체 인식 보안

양 대 헌 (DaeHun Nyang) 정회원
1994년 2월 : 한국과학기술원 과학기술 대학 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 조교수
<관심분야> 암호이론, 암호프로토콜, 인증프로토콜, 무선 인터넷 보안