Ad-hoc환경에서의 안전한 그룹키공유

이원희, 구재형, 황정연, 이동훈*

*고려대학교, 정보보호기술연구센터

A Secure Group-Key Agreement for Ad-hoc networks

Won Hui Lee, Jae Hyung Koo, Jung Yeon Hwang, Dong Hoon Lee*

*Center for Information Security Technologies(CIST) Korea Univ.

요 약

Diffie-Hellman(DH) 키공유 기법[1]이 제안된 후 많은 종류의 키공유 기법들이 연구되었으며, 특히 특정한 그룹 내의 구성원들이 안전한 통신을 할 수 있게 하기 위한 그룹 기반의 키공유 기법들이 제안되었다. 이러한 기법들은 PKI(Public Key Infrastructure) 등을 사용하는 여러 응용들에서 쉽게 사용될 수 있지만, ad-hoc통신망과 같이 특정한 기반구조를 사용할 수 없는 환경에서는 적용하기 어렵다. [2]에서는 ad-hoc 환경에서 사용할 수 있는 그룹(conference) 기반의 키공유 기법을 제안하였지만, n+2 라운드의 통신량이 요구되며, 그룹 구성원들의 위치정보와 같은 부가정보를 알고 있어야 하는 문제점이 있다. 본 논문에서는 ad-hoc 환경에서 그룹 구성원들의 위치정보를 알 필요 없는 효율적인 2-라운드 키공유 기법을 제안한다.

1. 서론

인터넷과 같은 신뢰할 수 없는 통신망에서 제3자에게 정보가 유출되지 않는 안전한 통신을 제공하기 위해서는 교환되는 데이터를 암호화하여야 한다. 이러한 암호통신을 제공하기 위해서는 데이터의 암·복호화에 사용될 키를 안전하게 공유하는 기법이 필수적이다. 1975년에 통신 참여자들이 생성한 난수값을 바탕으로 공유키를 생성하는 Diffie-Hellman 키공유 기법[1]이 제안되었으며, 이후 이를 기반으로 하는 많은 논문들이 제안되었다. 이들 논문들은 Public Key Infrastructure(PKI) 기반의 공개키를 사용하거나 이미 상호 간에 공유되어 있는 비밀값을 사용하기 때문에, 인증기관(CA : Certificate Authority)이나 신뢰할 수 있는 서버 등과 같은 제3의 신뢰기관 (TTA : Trusted Third Party)이 요구된다.

AP(Access Point)를 사용하여 무선 호스트들 간의 데이터 교환뿐만 아니라 유선망이나 다른 통신망에 접속이 가능한 BSS(Infrastructure Basic Service Set) 모델과는 달리 ad-hoc통신망에서의 무선호스트들은 자신의 무선통달거리 내에 존재하는 호스트들과 직접 통신하거나 라우팅 기능을 가진 무선 호스트의 도움을 받아 연결되는 (multi-hop) 호스트들과의 통신만이 가능하다. 또한 망을 구성하는 호스트들이 고정적이지 않고 호스트들의 위치변화에 따라 참여와 탈퇴가 빈번하여 망 구조가 유동적이기 때문에[5][6] ad-hoc망에서는 CA나 TTA를 사용할 수 없다. 물론 ad-hoc환경에서 비밀키 분산 기법[7]을 사용하여 ad-hoc통신망 내의 호스트들로 하여금 CA의 역할을 하도록 하여, 각 호스트들에게 인증서를 발급하는 기법[8][9]도 있지만, 새로이 망에 참여한 호스트를 인증하는 문제와 매번 새로운 인증서를 발급하기 위해 CA의 역할을 하는 호스트들이 참여해야하는 단점이 있다.

위와 같은 특징을 가지는 ad-hoc 통신망에서 상대 호스트와 안정적으로 세션키를 공유하기 위해서 패스워드 기반의 키공유 기법을 사용할 수 있다.

패스워드는 일반적으로 사람이 기억할 수 있는 정보이다. 이러한 정보는 매우 제한된 집합 내에서 선택된 것이므로 낮은 엔트로피를 갖게 된다. 따라서 패스워드로 암호화된 데이터를 갖고 있는 공격자는 사전공격(dictionary attack)을 통해 사용된 패스워드를 찾을 수 있다. 그러므로 낮은 엔트로피를 가지는 패스워드로부터 높은 엔트로피를 가지는 세션키를 생성할 수 있는 기법이 필요하다. 또한 안전한 패스워드 기반의 그룹키 공유 기법을 제공하기 위해서는 다음과 같은 성질을 만족해야한다.

Perfect forward Secrecy : 패스워드를 알고 있는 사용자만이 해당 세션키를 만들 수 있어야 하며, 사용된 패스워드가 유출되더라도 이전의 세션에서 사용된 세션키를 알 수 없어야 한다.

Contributory key agreement : 세션키가 모든 참여자의 참여값으로부터 생성되는 기법을 contributory 기법이라 부른다. Contributory 기법에서는 특정 사용자 또는 사용자들에 의해 세션키가 제한된 키공간에서 선택될 수 없어야 한다.

Tolerance to disruption attempts : 전송되는 데이터에 대한 수정이나 삭제는 할 수 없으나 삽입은 가능한 공격자에 대해서 안전해야 한다.

[2]에서는 이러한 성질을 만족하는 패스워드 기반의 그룹키 공유 기법을 제안하였다. 그러나 그룹세션키를 만들기 위해서는 그룹구성원들이 순차적으로 각자의 비밀값을 지수승을 해야 하기 때문에 그룹구성원의 수에 따라 라운드 수가 선형적으로 증가하며, 다른 그룹구성원들의 위치정보를 알고 있어야 한다. 본 논문에서는 이러한 단점을 보완한 패스워드 기반의 2-라운드 비대칭 그룹키 공유 기법을 제안한다.

제안된 기법에서는 회의장에 들어오기 전에 안전한 방법으로 그룹간에 공유된 패스워드를 알 수 있는 올바른 그룹구성원들이 ad-hoc통신망을 구성할 수 있는 무선 호스트를 사용하여 소규모 그룹이 회의를 하려 하는 상황을 가정한다. 물론 Ad-hoc통신망만이 구성될 수 있기 때문에 상호인증을 위해서 인증서나 신뢰할 수 있는 KDC(Key Distribution Center)를 사용할 수 없다. 그룹 내에는 회의 주최측의 무선 호스트가 그룹리더로 존재하며, 이 호스트는 좀더 강력한 계산능력을 갖는다.

2. 제안된 Ad-hoc환경에서의 그룹키공유 기법

2.1 제안된 기법

본 논문에서는 ad-hoc환경에서 효율적인 그룹키공유 기법을 제안한다. 그룹통신에 참여할 각 구성원들은 단계(1)에서 자신의 비밀값 S를 랜덤하게 선택하여 생성한 값 g^{S_i}을 그룹구성원들 간에 사전 공유된 패스워드(PWD)로 암호화하여 그룹리더에게 전송한다. 이후 그룹리더는 각 구성원들로부터 받은 값들을 복호화하여 g^{S_i}를 구한 후, 각 값에 자신의 비밀값인 S_i을 지수승한 뒤 곱하여 $k = g^{S_1S_2} \cdot g^{S_1S_3} \cdot \ldots \cdot g^{S_1S_n}$를 구하고, 이 값을 해쉬하여 그룹 세션키 $K = H(k)$를 생성한다. 여기서 $H()$는 일방향 해쉬함수이다. 그룹리더는 자신의 공개값 g^{S_1}을 패스워드로 암호화시킨 값과 각 노드들과 공유될 $g^{S_1S_i}$에 K를 곱한 값 C_i를 각 구성원 M_i에게 전달한다. 이 값을 받은 각 구성원들은 $g^{S_iS_1}$을 구해, C_i에 $g^{S_iS_1}$을 나누어 그룹세션키 K를 구한다.

Setup : *a finite cyclic group* $\mathbb{G} = < g >$

(1) $M_i \rightarrow M_1 : E_{PWD}(g^{S_i}), i = 2, \ldots, n,\ S_i \in_R \overrightarrow{z_p^*}$

(2) $M_1 \rightarrow M_i : E_{PWD}(g^{S_1}), C_i = K^* g^{S_1S_i}, i = 2, \ldots, n$

단계(1)에서 순차적으로 비밀값을 지수승을 하는 [2]와는 달리 각 구성원들은 그룹리더에게 자신의 공개값을 바로 전달하기 때문에, 각 그룹구성원들은 그룹리더의 정보만을 가지고 있으면 된다. 또한 모든 그룹세션키를 생성하기 위한 모든 단계가 그룹리더와 그룹구성원 사이에서만 이루어지기 때문에, 그룹 구성원의 수와 관계없이 라운드수가 일정하게 된다. 또한 그룹리더를 제외한 나머지 구성원들에 대한 계산량은 [2]에서보다 적게 요구된다. 그룹리더와 각 참여자 사이에 공유된 세션 공유값을 곱한 뒤 이를 해쉬시켜 세션키를 생성하고, 세션키에 각 사용자와의 공유키를 곱한 값을 전달하기 때문에 [3]에 비해 계산량이 많지만, 단계(2)에서 세션키를 전달할 때, 곱셈 연산 대신 XOR 연산을 사용하여 계산량을 줄일 수 있다.

2.1.1 그룹원 추가

그룹원이 추가된 경우, 새로이 추가된 그룹원M_{n+1}은 자신의 공개값 $g^{S_{n+1}}$을 그룹리더에게 사전에 공유된 패스워드로 암호화하여 전송한다. 그룹리더는 이를 복호화한 값에 새로이 생성된 자신의 비밀값 S_1'을 지수승 함으로써 새로이 참여한 그룹원과의 공유값 $g^{S_1'S_{n+1}}$을 생성한다. 또한 기존 그룹원들과의 공유값 $g^{S_1'S_i}$을 새로이 생성하여 $k' = g^{S_1'S_2} \cdot \ldots \cdot g^{S_1'S_n} \cdot g^{S_1'S_{n+1}}$을 생성하고, 이를 해쉬시켜 새로운 그룹세션키 $K' = H(k')$을 생성한다. 이 값에 각 그룹원과의 공유값 $g^{S_1'S_i}$을 곱한 값 C_i'를 각 구성원 M_i에게 전달한다. 이 값을 받은 각 구성원들은 $g^{S_iS_1}$을 구해 C_i'에 $g^{S_iS_1}$을 나누어 그룹세션키 K'을 구한다.

(1) $M_{n+1} \rightarrow M_1 : E_{PWD}(g^{S_1'})$

(2) $M_1 \rightarrow M_i : E_{PWD}(g^{S_1'}), C_i' = K'^* g^{S_1'S_i}, i = 2, \ldots, n+1$

2.1.2 그룹원 삭제

그룹원 M_j가 삭제된 경우, 그룹리더는 새로이 자신의 비밀값 S_1'을 생성하여 삭제된 그룹원을 제외한 모든 그룹원과의 공유값을 다시 생성하여 $k' = g^{S_1'S_2} \cdot \ldots \cdot g^{S_1'S_{j-1}} \cdot g^{S_1'S_{j+1}} \cdot \ldots \cdot g^{S_1'S_n}$를 생성하고, 이 값에 각 그룹원과의 공유값 $g^{S_1'S_i}$을 곱한 값 C_i'를 삭제된 구성원을 제외한 각 구성원 M_i에게 전달한다. 이

값을 받은 각 구성원들은 $g^{S_i'S_i}$을 구해 C_i'에 $g^{S_i'S_i}$을 나누어 그룹세션키 K'을 구한다.

(1) $M_1 \rightarrow M_i : E_{PWD}(g^{S_1'}),\ C_i' = K' * g^{S_i'S_i}$,
$i = 2, \ldots, j-1, j+1, \ldots, n$

2.2 안전성 분석

본 논문에서 제안한 그룹키 공유기법은 인증을 위해서 패스워드를 사용한다. 패스워드 기반의 그룹키 공유기법이 만족해야할 성질에는 서론에서 살펴본 바와 같이 Perfect forward secrecy, Contributory key agreement, Tolerance to disruption attempts가 있다. 다음은 본 논문에서 제안된 기법이 이 성질들을 만족함을 보인다.

1) Perfect forward secrecy

공격자가 이전 세션의 그룹키 공유 단계에서 사용된 메시지들을 가지고 있고, 그룹간에 공유된 패스워드를 알고 있다 하더라도, 공격자는 이전 세션의 그룹키를 알 수 없다. 공격자는 자신이 알게된 패스워드 P로 이전 세션에서 패스워드로 암호화된 메시지들로부터 $g^{S_1}, \ldots, g^{S_n}$은 알 수 있지만 Diffie-Hellman 문제에 의해서 $g^{S_1S_2}$을 구할 수 없다. 따라서 공격자는 해당 세션의 그룹키를 알 수 없다.

2) Contributory key agreement

세션키가 $K = H(g^{S_1S_2} \cdot g^{S_1S_3} \cdot \ldots \cdot g^{S_1S_n})$와 같이 모든 구성원들의 참여값이 사용되고, 계산된 값이 해쉬되므로 contributory를 만족한다. 만약, 그룹리더가 K의 값을 자신이 임의로 선택한 값으로 정한다면, 각 사용자는 K를 구한다음, 단계(2)에서 각 사용자에게 전달되는 G 값들로부터 각 구성원과 그룹리더 사이에 공유된 키 $g^{S_1S_i}$를 구해 이 값들을 곱한 뒤 해쉬시켜 K와 비교함으로써 contributoiry가 만족되었는지 알 수 있다.

3) Tolerance to disruption attempts

단계(1)에서 공격자가 임의의 값 g^r을 그룹리더 M_1에게 보냈다고 가정하자. M_1은 g^r을 패스워드로 복호화하고 이 값에 자신의 비밀값을 지수승하여 $k = g^{S_1S_2} \cdot \ldots \cdot g^{S_1S_{n-1}} \cdot g^{S_1r'}$를 구성할 수 있다. 그리고 단계(2)에서 자신에게 데이터를 보낸 모든 노드에게 G를 전달한다. 패스워드를 알고 있는 그룹구성원들은 자신과 그룹리더와의 공유값 $g^{S_1S_i}$를 구할 수 있으므로 G를 $g^{S_1S_i}$으로 나누어 K를 구할 수 있지만, 패스워드를 모르는 공격자가 $g^{S_1r'}$을 구하기 위해서는 r'을 구해야 한다. 이는 이산로그 문제를 푸는 것이므로 공격자는 그룹리더가 구한 $g^{S_1r'}$을 구할 수 없게되고, 따라서 K를 구할 수 없다.

2.3 효율성 분석

	기법1	기법2	기법3	기법4
라운드수	n+2 라운드	logn 라운드	2라운드	2 라운드
노드별 계산량	M_i : 3 지수승 1 나눗셈, 1 암호화 2 곱셈 M_1 : n 지수승 n-1암호화	*All* : logn 암호화 logn 지수승	M_i : 2 지수승 1 서명, 1 XOR 1 해쉬 M_1 : n-1 지수승 n-1 Verify n 해쉬 n-1 XOR	M_i : 2 지수승, 1 나눗셈, 2 암호화 1 해쉬 M_1 : n-1 지수승, n 암호화, 2(n-1) 곱셈 1 해쉬

표 1: 기법 간 성능비교

※ 기법1 - Asokan의 Diffie-Hellman 그룹키공유 기법

※ 기법2 - Hypercube상에서의 Diffie Hellman 그룹키 공유 기법

※ 기법3 - Bresson 등이 제안한 그룹키공유 기법

※ 기법4 - 본 논문에서 제안한 그룹키공유 기법

표 1은 위에서 언급한 각 기법에서 그룹세션키가 공유되기까지의 라운드 수와 각 노드별 계산량을 비교한 것이다. 이 표에서 볼 수 있는 것처럼 본 논문에서 제안하고 있는 기법의 경우 [2]와는 달리 구성원의 수에 관계없이 라운드 수가 2로 고정되며, 계산량의 경우 [2]에서 일반 구성원들은 3번의 지수승과 1번의 나눗셈연산, 1번의 암호화 과정이 필요하지만, 본 논문에서 제안하는 기법에서는 일반 구성원들은 1번의 지수승과 1번의 나눗셈연산, 1번의 암호화 과정과 1번의 복호화 과정만이 요구된다. 그룹리더인 M_1의 경우 [2]에서는 n번의 지수승과 n-1번의 복호화 과정이 요구되지만, 본 논문의 기법의 경우 n번의 지수승과 n-1번의 복호화 과정 및 1번의 암호화 과정, 2(n-1)번의 곱셈연산이 요구된다. 일반 그룹구성원의 경우 계산량이 기존 기법에 비해 적으며, 그룹리더의 경우 좀더 많은 계산량을 요구하나, 전체 시스템의 경우 [2]보다 비용 면에서 효율적이다. 실제 구현상에 있어서도 본 논문에서 제시한 기법의 경우 사용자가 처음 그룹리더에게 접속할 때 자신의 공유값을 패스워드로 암호화하여 보냄으로써 통신횟수를 줄일 수 있는 반면, [2]에서 제안하고 있는 기법의 경우 그룹리더를 제외한 그룹구성원들은 단계(1)의 메시지를 전달할 노드에 대한 정보를 알아야 한다. 이를 위해서 각 그룹구성원들은

먼저 그룹리더에 접속하여 그룹리더로부터 단계(1)의 처리 순서를 얻어야 한다. 따라서 모든 구성원들이 그룹리더에 접속한 이후에야 단계(1)을 수행할 수 있다. 하지만 그룹세션키를 생성하고, 세션키를 각 구성원들에게 전달하는 방식 때문에 [3]에 비해서는 계산량이 많다.

본 논문에서 제안하는 기법의 효율성 측정을 위해서 [2]에서 제안하고 있는 기법과의 실제 그룹키 생성 시간에 대한 비교결과를 표 2에서 제시하였다. 측정은 Intel Pentium4 2.0 ~ 2.4 GHz, 256M급 컴퓨터에서 windows 2000 및 windows XP 환경에서 실시되었으며, 실제 프로그램은 Visual C++ 6.0에서 암호라이브러리로써 openssl 2.7.0의 crypto 라이브러리를 사용하여 개발하였다. 측정값의 정확성을 위해서 각 항목별로 6회 측정하여 그 평균값을 사용하였다.

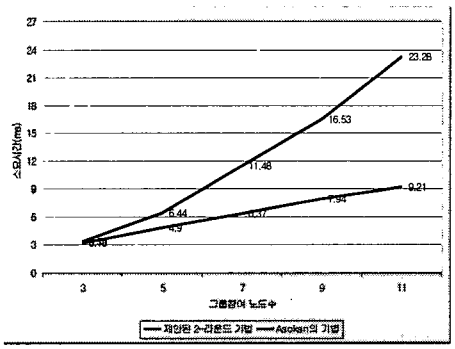

그림 1 : 기법별 그룹키 생성 소요시간

그림 1에서 확인할 수 있듯이 Asokan과 Ginzboorg가 [2]에서 제안한 그룹키 공유 기법의 경우 그룹에 참여하는 노드의 수가 증가할수록 그룹키를 생성하는데 소요되는 시간이 급격히 증가하는 것을 확인할 수 있다. 이는 단계 1에서 그룹참여 노드가 순차적으로 자신의 비밀값을 지수승하기 때문이다. 반면에 본 논문에서 제안한 기법은 그룹참여 노드와 그룹리더 간의 Diffie-Hellman 키교환만을 수행하도록 함으로써 그룹키를 생성하기 위한 통신량을 줄였다.

3. 결론

본 논문에서는 [2]에서 그룹구성원들이 그룹 세션키를 만들기 위해 순차적으로 각자의 비밀값을 지수승함으로써 발생하는 라운드 수를 줄이기 위해, 그룹리더에게 그룹구성원 각자의 공개값을 보내도록 하여, 그룹 세션키를 $g^{S1Sn} \cdot g^{S2Sn} \cdot \ldots \cdot g^{Sn-1Sn}$와 같은 형태로 구성함으로써 그룹리더의 위치정보만을 가지고 2-라운드에 그룹세션키를 구성할 수 있는 기법을 제안하였다. 또한 제안된 기법은 그룹리더를 제외한 일반 그룹구성원들의 계산량이 [2]보다 한번의 지수승 만큼 적게 요구된다. 본 논문에서 제안된 기법을 사용하여 ad-hoc환경에서 좀더 효율적인 그룹키공유를 제공할 수 있다.

[참고문헌]

[1] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory*, vol. IT-22, pp 644-654, Nov 1976

[2] N.Asokan and Philp Ginzboorg. "Key Agreement in Ad-hoc Networks," *Computer Communications*, vol 23:1627-1637, 2000

[3] Emmanuel Bresson, Olivier Chevassut, Abdelilah Essiari, and David Pointcheval, "Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices", The Fifth IFIP-TC6 International Conference on Mobile and Wireless Communications Networks, pages 59-62, October 2003

[4] Steven M. Bellovin and Michael Merrit. "Encrypted key exchange : Password-based protocols secure against dictionary attacks," *In Proceedings of the IEEE Symposium on Research in Security and Privacy*, May 1992

[5] Carlo Kopp, "Ad Hoc Networking," Published in *'System'*, pp 33-40, June 1999

[6] 김동완, "이동 Ad Hoc망 기술 개요," http://kmh.ync.ac.kr/Network2/mobile/2000/itr03-20000300.htm

[7] A. Shamir, "How to Share a Secret", *Communications of ACM* 1979

[8] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks", *IEEE Networks*, Volume 13, Issue 6 1999

[9] H. Luo and S. Lu, "Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks", *Technical Report 200030*, UCLA Computer Science Department 2000

[10] Klaus Becker and Uta Wille, "Communication complexity of group key distribution", In *5th ACM Conference on Computer and Communications Security*, pp 1-6