BFT를 이용한 악의적인 사용자를 식별하는 그룹키 일치 방식[1)]

홍 동 연* 김 광 조*

*카이스트 정보보호대학원

Group Key Agreement with Byzantine Fault Tolerant for identifying malicious users

Dongyeon Hong* Kwangjo Kim*

*Graduate School of Information Security. KAIST

요 약

암호시스템은 안전하지 않은 채널에서 통신 상대방과 보안성을 유지하기 위해 사용된다. 암호시스템을 사용하기 전 2자간 혹은 그룹 내 비밀키 교환이 우선 수행된다. 근래 두 명이 아닌 그룹 대상의 앱 또는 작업 환경이 많아짐에 따라 그룹 내 키 교환이 중요해지고 있다. 그러나 그룹 내 악의적인 사용자가 존재하는 경우 심각한 문제가 발생한다. 이에, 그룹키 공유 전 악의적인 사용자를 색출하여 제거하는 과정은 필수적이다. 기존 논문들은 해당 과정 때문에 그룹키 공유가 지체될 수 있다. 본 논문에서는 이와 같은 상황을 해결하고 더 나아가 비잔틴 장애 허용까지 보장하는 방법을 제안하고자 한다.

I. 서론

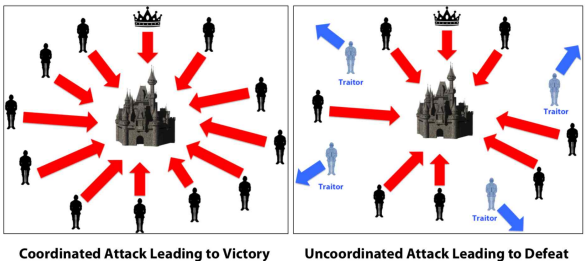
1.1 배경

블록 암호 등 대칭키 암호와 공개키 암호는 다양한 영역에서 사용된다. 공개키 암호는 대칭키 암호보다 상대적으로 느리다는 단점이 있기 때문에 정보를 주고받을 때 대칭키 암호가 사용된다. 대칭키 암호는 사용자 간에 비밀키가 이미 공유되어 있다는 것을 전제로 하기 때문에 비밀키 공유는 중요한 이슈 중 하나이다. 근래에 두 명이 아닌 그룹 단위 앱과 작업 환경이 증가하여 그룹 구성원이 동일한 비밀키를 갖는 알고리즘이 필요하다. 그룹 구성원이 단기간 동안 공유하는 비밀키를 그룹키라 한다. 2자간 키 교환과 달리 그룹키를 공유할 때 악의적인 사용자가 그룹 내부에 존재할 가능성이 있다. 그룹키 노출과 민감한 정보가 노출될 수 있는 위험 때문에 악의적인 사용자의 식별 및 처리는 중요한 과정이다. [1-5]는 식별 및 처리 과정이 있는 논문이다. 그러나 악의적인 사용자의 방해로 그룹키 공유가 지체되는 문제점이 있다. 이에, 본 논문에서는 기존 식별 과정에 합의 알고리즘을 적용하여 문제를 해결하고 비잔틴 장애 허용(BFT, Byzantine Fault Tolerance) [6]까지 보장하는 그룹키 알고리즘을 제안하고자 한다.

1.2 논문 구성

본 논문의 구성으로 Ⅱ장은 관련 논문을 살펴본다. Ⅲ장에서 비잔틴 장애 허용과 이를 위한 알고리즘을 간략히 살펴본다. Ⅳ장은 기존에 논문이 갖고 있던 문제에 대해 다룬다. Ⅴ장에

1) 본 연구는 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행되었습니다. (No.2017-0-00555, 양자컴퓨터 환경에서 래티스 문제를 이용한 다자간 인증키교환 프로토콜 연구)



[그림 1] 비잔틴 장군 문제

서는 Ⅲ장에서 다루었던 문제 해결을 위해 알고리즘을 적용하고 어떻게 문제를 해결하는지 보인다. Ⅵ장에서는 결론과 추후 연구에 대한 가능성을 제시한다.

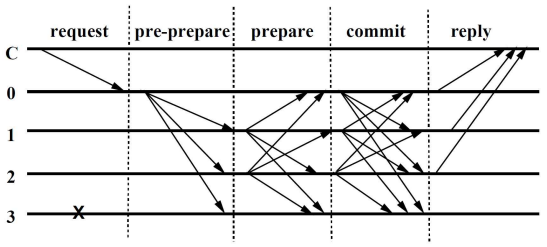
II. 관련 논문

[1-5]의 그룹키 공유 과정은 크게 5단계로 구분할 수 있다.

(1) 각 사용자를 등록하고 (공개키, 비밀키)를 생성한다. **(2)** 그룹키를 위한 서브키 (subkey)와 검증에 필요한 값 (*validity*)을 생성하고 (subkey,*validity*) 메시지 (*msg*)를 다른 사용자들에게 전송한다. **(3)** 받은 메시지에서 서브키를 복구하고 타당성 검증을 수행한다. **(4)** 타당성 검증에 문제가 발생한 경우, 해당 사용자 서브키의 타당성을 재검증한다. 검증 결과에 따라 사용자를 그룹에서 제거한다. **(5)** 그룹키를 생성한다.

[1]은 이산 대수 문제와 선형 함수, [2-4]는 RSA와 해시함수, [5]는 타원 곡선과 겹선형 페어링 등 각각 사용한 암호기법이 다르다.

[3]은 악의적인 사용자가 정직한 사용자의 비밀키를 알지 못해도 정당한 사용자처럼 보일 수 있는 문제점을 지적하며 *validity*에 타임스탬프를 추가하고 전자 서명하는 것을 해결 방안으로 제안한다. [4]는 악의적인 사용자가 두 개의 subkey K_1, K_2를 만들고 정직한 사용자 U에게 (K_1, $validity_1$)을 다른 사용자들에겐 (K_2, $validity_2$)를 보낸다. U가 실패 메시지를 보내어 4단계를 수행하지만 U를 제외한 모든 사용자는 (K_2, $validity_2$)가 타당하기 때문에 U를 그룹에서 제거한다. 이에 대한 해결 방안으로 2단계 *msg*를 해시하여 전자 서명을 하고, 4단계 수행 시 U가 받은 (K_1, $validity_1$)의 타당성을 확인하는 것을 제안한다.



[그림 2] PBFT 알고리즘

III. 배경 지식

3.1 비잔틴 장애 허용

그림 1처럼 중앙에 있는 성을 점령하기 위해 성 주변 군부대는 같은 시기에 공격이 필요하다. 비잔틴 장군 문제는 부대들 간 공격 또는 후퇴 시기를 결정할 때 발생하는 딜레마이다. 부대는 메시지를 보내 의견을 교환하며 두 가지의 실패가 있다. 의도한 실패와 의도하지 않은 실패로 나눌 수 있으며 의도한 실패는 메시지 내용에 혼란을 주어 합의 실패를 유도하는 경우를, 의도하지 않은 실패는 메시지 누락 혹은 파괴로 구분한다. 두 가지 실패에 내성이 있는 경우 BFT라고 한다.

3.2 PBFT 알고리즘[7]

위 3.1을 위해 1999년 Castro는 비동기 통신에 PBFT(Practical Byzantine Fault Tolerance)를 제시하였다. 과정은 아래와 같이 (그림 2)

(1) 클라이언트가 리더 노드에 요청을 전달한다. **(2)** 리더 노드가 명령을 순차적으로 다른 노드에 전달한다. **(3)** 각 노드는 명령을 받게 되면 리더를 포함한 모든 노드에 회신한다. **(4)** 각 노드는 수신된 명령을 $f+1$개 이상 수신하면 지정한 명령을 수행한다. (그룹 전체 크기를 n이라고 할 때 $f = \lfloor (n-1)/3 \rfloor$ 이다.) **(5)** 클라이언트에 메시지를 반환한다.

PBFT 알고리즘을 하기 위해선 리더 선출 작업이 필요하지만 본 논문에서는 다루지 않으며, PBFT는 그룹의 크기에 관계없이 안정적으로

작동한다고 가정한다.

IV. 공격 모델 및 취약점 분석

4.1 공격 모델

사용자들은 그림 3처럼 구성되어 있다고 가정하고 [5]를 따라 악의적인 사용자는 1단계를 따르지 않고 (공개키, 비밀키)를 생성한 사용자로 정의한다. 3단계에서는 msg에서 subkey를 복구하고 $validity$를 통해 타당성을 검증한다. (반대로 msg의 타당성을 검증하고 subkey를 복구할 수 있다.) 검증 결과를 다른 사용자에게 보내며 U_i은 U_j의 subkey가 타당하지 않은 경우 실패 메시지 $v_{i,j}$를 다른 사용자들에게 보낸다. 그룹 내 모든 사용자는 4단계를 수행하여 U_j의 타당성을 재검증한다. 4단계에서 사용자들은 U_j은 msg를 보내야 하며 어떤 사용자도 msg를 받지 못하면 U_j를 제거한다. 값을 받는다면 U_i, U_j를 제외한 다른 사용자는 U_2의 값을 재검증한다. U_j가 타당한 경우 U_i을 그룹에서 제거하고 그 외의 경우 U_j를 제거한다.

4.2 취약점 분석

4.2.1 의도적인 실패

우리는 의도적인 실패 두 가지를 고려하였다. 먼저, 그룹의 크기가 큰 상태에서 악의적인 사용자가 의도적으로 실패 메시지를 보낸다고 가정한다. 그 결과 사용자 검증에 오랜 시간이 소요된다. 두 번째는 악의적인 사용자가 그룹 내에 다수 있을 때 실패 메시지를 하나씩 보낸다고 가정한다. 매번 재검증 과정을 진행하며 최종 그룹키 생성이 지체된다.

4.2.2 의도하지 않은 실패

2단계에서 U_j가 서브키 메시지를 전송하는 과정에 오류 발생 시 해당 메시지를 받은 사용자 U_i는 실패 메시지 $v_{i,j}$를 보낸다. 그러나 4단계를 수행 후 정당한 사용자 U_i는 그룹에서 제거된다.

V. 제안 알고리즘 및 개선 사항

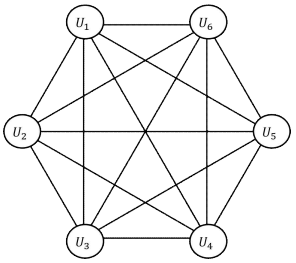


그림 3 그룹 네트워크 구성도

5.1 제안 알고리즘

위 4.1의 공격 모델 중 검증 결과 메시지를 보내 다른 사용자의 정당성을 판단한다. 우리는 이 과정을 합의 알고리즘으로 보고 4.2에서 다룬 문제의 해결과 BFT를 보장하기 위해 다음과 같은 알고리즘을 제안한다. 기존 3단계를 다음과 같이 수정한다.

- **Leader**(): 그룹 내 리더를 선출하고 리더는 어떤 사용자에 대해 검증을 수행할지 결정한다.
- **ValidityCheck**(msg): 각 사용자는 결정된 사용자의 서브키를 복구하고 타당성 검증을 수행한다.
- **Broadcast**($v_{i,j}$): 검증 결과 메시지를 다른 사용자들에게 전송한다.
- **Decision**(): 리더는 검증 결과에 따라 다음 단계를 정한다. 실패 메시지가 $f+1$개 이상이면 4단계를 수행하고 성공 메시지가 $2f+3$개 이상이면 5단계로 넘어간다.

[3, 4]에서 다룬 공격에 내성을 갖기 위해 $validity$에 타임스탬프를 추가하고 msg에 해시와 전자 서명을 수행한다. 또 4단계를 수행할 때 [4]와 같은 실패 메시지를 보낸 사용자의 subkey에도 타당성 검증을 수행하여 다른 키 공격에 내성을 갖도록 한다.

5.2 개선 사항

본 장은 4.2에서 다룬 공격들의 해결 방안을 알아보고 기존 논문들과 비교해본다. 4.2.1의 의도적인 실패는 악의적인 사용자가 실패 메시지를 하나씩 보내 5단계로 넘어가지 못하고 4단계를 수행한다. 악의적인 사용자를 전부 처리할 때

표 1 기존 알고리즘들과 비교표

	Huang et. al.[1]	Zhao et. al.[2]	Eslami et. al.[5]	본 연구
암호학적 도구	이산 대수 문제	RSA	타원 곡선	특정화 되지 않음
탐지 방법	선형 함수	RSA 전자 서명 해시 함수	겹선형 페어링	특정화 되지 않음
공격 성공에 필요한 악의적인 사용자의 수	1	1	1	$f+1$
고장 내성 여부	부	부	부	가

까지 많은 시간이 소요되기 때문에 최종 그룹키 공유가 늦춰지게 된다. 그러나 제안 알고리즘은 실패 메시지가 $f+1$개 이상인 경우 4단계를 수행하므로 기존 공격은 그룹키 생성 과정에 영향이 없다. 4.2.2 의도하지 않은 실패는 msg를 보낼 때 오류가 발생하여 생기는 문제이다. 마찬가지로 4단계를 수행하는 실패 메시지 개수의 증가로 해당 문제점을 해결할 수 있다. 또한, 어떤 사용자가 응답하지 않더라도 $2f+3$개의 성공 메시지나 $f+1$의 실패 메시지를 통해 다음 단계로 넘어갈 수 있다. 제안 알고리즘은 PBFT 때문에 BFT 성질을 보장한다. 표 1은 [1,2,5]과 제안 알고리즘을 비교한 표이다. 제안 알고리즘의 암호학적 도구나 탐지 방법은 설계에 따라 달라질 수 있다. 세 번째 항목은 공격에 필요한 악의적인 사용자의 수이며 네 번째 항목은 실패 메시지의 개수이며 네 번째 항목은 의도하지 않은 실패에 대한 내성 여부이다.

VI. 결론 및 향후 과제

본 논문에서는 다자간의 비밀키 공유 과정에 필요한 식별 과정의 효율성 증가를 제시하였다. 기존은 두 가지의 공격이 가능했고 실패 메시지 하나에 사용자 검증을 수행하였다. 본 논문에서는 합의 알고리즘인 PBFT를 적용하여 제시한 두 가지 공격에 내성이 있으며 $f+1$개 이상일 때 식별 과정을 수행함으로 불가피한 재검증 과정을 생략하고 BFT 성질을 보장한다.

향후 연구 과제로는 기존 논문들과 속도를 수치적 비교와 P2P 네트워크 등 다양한 네트워크에 적용과 다른 합의 알고리즘 적용을 할 예정이다. 이후 향상된 그룹키 교환 알고리즘을 제안할 예정이다.

[참고문헌]

[1] K. H. Huang, et. al. A conference key agreement protocol with fault-tolerant capability. Computer Stanards & Interfaces, 31(2), 401-405, 2009.

[2] J. Zhao, et. al. An efficient fault-tolerant group key agreement protocol. Computer Communications, 33 (7), 890-895, 2010.

[3] Z. Wang, Improvement on the fault olerant group key agreement protocol of Zhao et al. Security and Communication Networks, 9(2), 166-170, 2016.

[4] A. Fu, et. al. A secure and efficient fault-tolerant group key agreement protocol. In 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications 310-314, July, 2013.

[5] Z. Eslami, et. al. Provably Secure Group Key Exchange Protocol in the Presence of Dishonest Insiders. IJ Network Security, 18(1), 33-42, 2016.

[6] L. Lamport, et. al. The Byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS), 4(3), 382-401, 1982.

[7] M. Castro and B. Liskov, Practical Byzantine fault tolerance. Proceedings of the third symposium on Operating systems design and implementation, 173-186, February, 1999.