와이어 메신저 어플리케이션 프로토콜 분석

김용찬*
*성균관대학교 컴퓨터공학과
quarter814@gmail.com

Wire Messenger Application Protocol Analysis

Young-Chan Kim*
*Dept. of Computer Engineering, Sungkyunkwan University

요 약

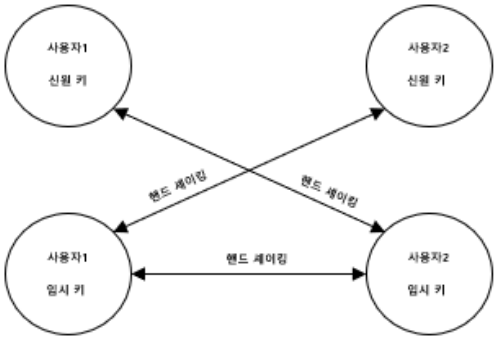
오늘날 인스턴트 메시징은 사용자들에게 개인 및 비즈니스 상의 대화의 편리함을 제공하며 거의 모든 현대인들의 일상에 한 축을 담당하고 있다 해도 과언이 아니다. 이러한 모바일 메신저 어플리케이션들은 사용성을 넘어 사용자들의 개인정보 보호와 메시지의 기밀성 및 무결성을 위해 끊임없이 발전해 왔고 많은 사용자들 역시 언론을 통해 발표되는 메신저 어플리케이션들의 여러 보안 이슈들을 접하게 되면서 사용성을 넘어 강력한 보안을 자랑하는 메신저 어플리케이션을 주목하고 이를 사용하고 있다. 그 중 안전하다고 평가받고 있고 백만 이상의 안드로이드 앱 스토어 사용자가 설치한 와이어 라는 메신저 어플리케이션의 프로토콜 분석을 통해 어떤 방법으로 그들이 강조하고 있는 보안 목표를 달성하고 있는지 확인한다.

1. 서론

일상생활과 떼어 놓을 수 없을 정도로 자주 이용하고 있는 메시징 서비스를 제공하는 모바일 메신저 어플리케이션이 우리의 대화내용 및 사용자의 개인정보 등을 얼마나 안전하게 관리하는지 뚜렷한 인식 없이 사용하는 경우가 대부분이다. 따라서 사용자들이 많이 사용하고 있는 메신저 어플리케이션 중 와이어 라는 특정 어플리케이션을 선정하여 사용자간 대화를 위해 어떻게 키를 분배하고 관리하는지, 메시지를 주고받는 과정에 사용하고 있는 알고리즘, 사용자 등록 과정, 대화를 위한 세션 형성과 같은 프로토콜을 분석함으로써 모바일 메신저 어플리케이션이 공격자가 세션 키를 중간에 획득하더라도 이전 메시지를 볼 수 없는지 또는 이후 메시지를 볼 수 없는지, 종단간 암호화는 잘 수행하고 있는지 등과 같은 보안을 위해 어떤 속성들을 지니고 사용자에 대한 서비스를 제공하고 있는지 파악한다. 더 나아가 이해한 프로토콜을 기반으로 보안상 문제점이 발생할 수 있는 경우에 대하여 정리해 보도록 한다.

2. 이중 래칫 알고리즘의 개요

와이어 어플리케이션 프로토콜의 경우 메시지를 주고받기 위해 이중 래칫 알고리즘을 통해 종단간 암호화를 제공한다. 먼저 래칫 과정을 살펴보면 대화를 형성하고자 하는 두 사용자 각각의 신원 키와 임시 키를 활용하여 디피 헬만 키 교환 방식을 이용한 세 번의 핸드 셰이킹 과정을 통해 비밀 공유를 이뤄낸다. 이 방식은 그림 1 과 같이 나타나며 결과물을 키 파생 함수를 거쳐 메시지를 암호화할 수 있는 키 값을 생성하기 위한 중간 키 값을 만들어낸다. 이와 같이 세션 키의 갱신 및 유지 관리를 위해 디피 헬만 키 교환 방식과 함께 키 파생 함수를 결합하여 사용하는 방식을 이중 래칫 알고리즘이라 한다.



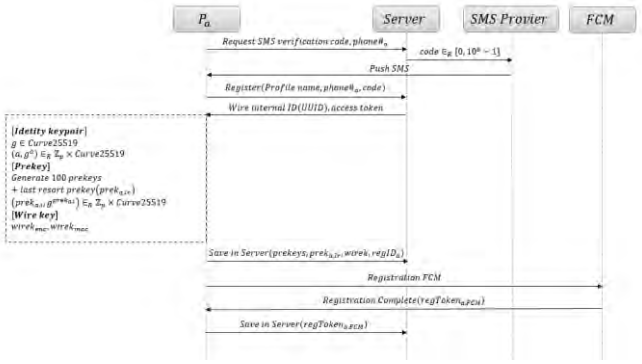
(그림 1) 세션 키 형성 과정.

3. 와이어 어플리케이션 프로토콜

와이어 어플리케이션 프로토콜의 경우 오픈소스[1]를 제공하고 있고 메시지 분석 시 어플리케이션에서 주고받고 있는 패킷을 참고해 사용자 등록 과정부터 메시지를 암호화하여 주고받는 과정까지 조사해 보았다.

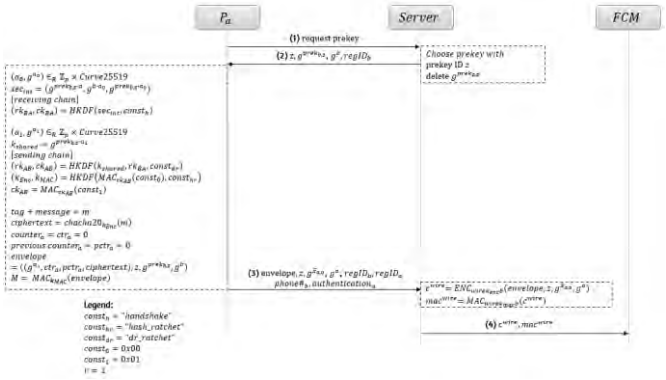
사용자 등록 과정부터 살펴보면 사용자가 소유한 스마트 폰의 전화번호를 통하여 등록이 가능한다. 먼

저 사용자가 와이어 서버에 전화번호와 함께 인증코드를 요청한다. 이에 생성한 인증코드를 SMS 를 통해 사용자에게 전달하고 전달받은 인증코드를 사용자가 전화번호와 함께 서버로 제출하게 되면 이를 확인한 서버에서 사용자에게 와이어 내부 아이디와 통신을 위한 토큰을 제공한다.[2] 이 후 사용자는 앞으로의 와이어를 통한 메시징 서비스를 이용하기 위해 몇 가지 정보들을 와이어 서버에 저장을 하게 된다. 첫 번째로 자신의 신원 키와 임시 키 100 개를 저장하는데 이는 세션을 형성하기 위해 사용된다. 두 번째로 와이어는 파이어 클라우드 메시징 서비스를 이용해 송신자로부터 수신자로 메시지를 전달하는데 메시지의 암호화 과정 이 외에 와이어 서버에서의 또 한 번의 암호화 과정을 거쳐 파이어 클라우드 메시징 서버를 통해 데이터를 전송하기 때문에 이에 필요한 암호화 키와 메시지 인증 키를 저장한다. 세 번째로 사용자 등록 정보를 서버에 저장하는데 이는 누가 누구에게 메시지를 전달하는지 누구의 키를 사용해 서버에서 암호화를 실시하고 파이어 클라우드 메시징 서버로 보낼지 결정한다. 마지막으로 사용자는 파이어 클라우드 메시징 서비스를 이용하기 위해 파이어 클라우드 메시징 서버에 등록을 하고 등록 토큰을 받아와 와이어 서버에 저장을 한다. 위의 내용은 그림 2 를 통해 확인할 수 있다.



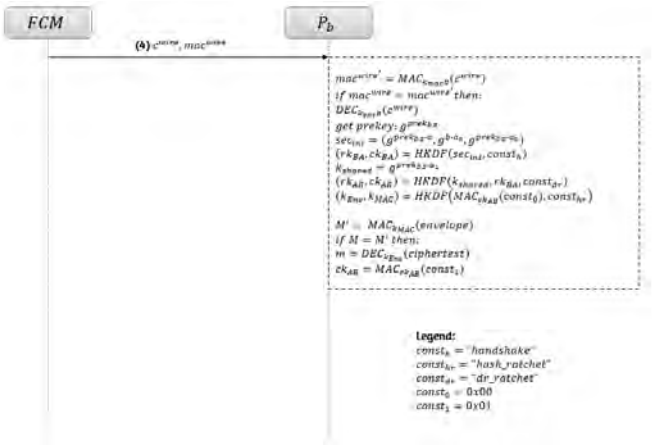
(그림 2) 사용자 등록 과정

사용자간 세션을 형성하는 과정은 첫 메시지를 전송할 때에 진행되는데 메시지 송신자는 서버로부터 통신하고자 하는 수신자의 임시 키를 요청한다. 서버는 수신자의 임시 키와 함께 수신자 등록 정보와 신원 키를 송신자에게 전송하고 이를 받은 송신자는 자신의 신원 키와 임시 키 그리고 수신자의 신원 키와 임시 키를 가지고 디피 헬만 키 교환 방식을 이용한 세 번의 핸드 셰이킹 과정을 통해 비밀 고유 값을 생성한다. 이를 키 파생함수의 인풋 값으로 사용해 중간 키 값을 만드는데 이를 통해 송신할 때 쓰이는 키를 생성하기 위한 키(송신 키)와 수신할 때 받은 메시지의 복호화를 위한 키 값들을 생성하기 위한 키(수신 키)가 만들어진다. 이 후 송신자의 메시지를 송신 키에서 파생된 메시지 암호화 키를 가지고 Chacha20 알고리즘을 사용하여 암호화를 실시하고 암호화된 메시지와 함께 메시지 순서를 파악하기 위해 0 으로 초기화된 카운터와 이전 카운터 그리고 송신자의 신원 키 와 임시 키에 대한 데이터 및 이에 대한 모든 정보를 송신 키에서 파생된 메시지 인증 키를 통해 생성한 메시지 인증 코드 값과 함께 송, 수신자의 정보를 담아 와이어 서버로 전송한다. 와이어 서버는 송신자가 전송한 데이터를 수신자가 서버에 저장해 놓은 암호화 키로 암호화하고 이 값을 메시지 인증 키로 메시지 인증 코드 값을 생성해 파이어 클라우드 메시징 서버로 전송한다. 이는 그림 3 에 나타나 있다.



(그림 3) 세션 생성 및 메시지 송신 과정

수신자는 수신된 데이터를 자신이 가지고 있는 메시지 인증 키로 메시지 인증 코드 값을 생성하고 받아온 메시지 인증 코드 값과 비교하여 일치하면 복호화를 실시한다. 복호화 한 데이터 안에는 송신자가 송신 키와 수신 키를 생성할 때 사용한 송신자의 신원 키와 임시 키가 포함되어 있으므로 수신자는 자신이 가지고 있는 신원 키와 임시 키를 가지고 메시지를 주고받을 수 있는 동일한 환경을 구성할 수 있게 된다.[3] 이는 그림 4 에 나타나 있다. 결국 송신자의 송신 키와 매칭되는 수신자의 수신 키, 송신자의 수신 키와 매칭되는 수신자의 송신 키를 통해 메시지가 암호화되고 복호화 된다.



(그림 4) 메시지 수신 과정

4. 와이어 어플리케이션에서 발생할 수 있는 문제

문제가 발생할 수 있는 가정을 통해 보안에 대해 생각해 보면 만약 공격자가 수신자의 신원 키 와 임시 키 그리고 제 3 자의 신원 키와 임시 키를 알 수 있다고 가정한다면 누구든 제 3 자와 수신자 간 비

밀 고유 값을 생성할 수 있을 것이다. 이를 통해 공격자가 제 3 자인 행세를 하며 수신자에게 메시지를 전달하는 중간자 공격에 취약할 수 있다. 그러나 이는 공격자가 와이어 서버에 자신이 제 3 자임을 인증할 수 있거나 혹은 악의적인 서버에 의해 별도의 인증 과정을 거치지 않고 모든 메시지를 전송하도록 하게 된다면 문제가 발생할 수도 있다. 또한 악의적인 서버라고 가정한다면 카운터 값의 변경을 통해 임의로 메시지의 순서를 조작하는 행위[4] 또한 가능할 것이다.

5. 결론

와이어 프로토콜 분석을 통해 사용자 등록 과정, 세션 형성, 메시지 송수신 과정까지 살펴보았다. 그 중 두 사용자 간 비밀 공유 키를 생성하고 이를 키 파생함수를 이용해 메시지 암호화를 위한 키 값을 획득했다. 이 값을 통해 두 사용자 간의 메시지의 암호화, 복호화가 일어나고 복호화 과정 이후에는 복호화 키가 제거되도록 관리하고 있어 복호화 키를 얻어내기 힘들다. 또한 상대방의 신원 키와 임시 키를 통해 비밀 공유 값을 누구든 생성할 수 있기 때문에 서버에서 별도의 사용자 인증 과정을 거쳐 다른 사용자의 스마트 폰의 획득 없이 다른 사용자인 척하며 메시지를 송신할 수 없다. 메시지 내용에 대해서도 중간에 와이어 서버나 파이어베이스 클라우드 메시징 서버에서 확인할 수 없도록 암호화 과정을 통한 종단간 암호화를 통한 보안이 잘 이루어져 있다. 또한 중간에 키 값이 유출되더라도 세션 키에 대한 갱신이 계속 일어나고 있기 때문에 미래의 정보 및 과거의 정보에 대하여 안전하다고 보여 진다.

참고문헌

[1] https://github.com/wireapp/proteus

[2] https://wire-docs.wire.com/download/Wire+Security+Whitepaper.pdf

[3] Tilman Frosch, Christian Mainka, Christoph Bader, Florian Bergsma, Jorg Schwenk, Thorsten Holz. “How Secure is TextSecure?”, 2016

[4] Paul R ¨ osler, Christian Mainka, J ¨ org Schwenk, “More is Less: On the End-to-End Security of Group Chats in Signal, WhatsApp, and Threema”, 2018