# Database 복호화 키 저장 위치에 따른 Database 보안성 연구

이창원, 최형기
성균관대학교 소프트웨어대학
lcw921@g.skku.edu, meosery@skku.edu

# A Study on the Database security research classified according to Database decryption key storage location

Chang Won Lee, Hyung Kee Choi
Sungkyuankwan Univ

요 약

메신저 애플리케이션은 사용자의 채팅 로그나 전화번호와 같은 개인 정보를 데이터베이스에 저장하며, 비밀번호 관리 애플리케이션은 사용자의 비밀번호 정보를 데이터베이스에 저장한다. 따라서 사용자의 개인 정보들이 들어 있는 데이터베이스를 안전하게 보호하는 것은 매우 중요하다. 본 논문에서는 암호화된 데이터베이스를 복호화하기 위한 키를 저장하는 위치에 따라 애플리케이션을 분류하고, 각 경우 보안성이 어떻게 달라지는지에 관한 연구를 수행했다.

## 1. 서론

메신저 애플리케이션은 사용자의 채팅 로그나 전화번호와 같은 개인 정보를 데이터베이스에 저장하며, 비밀번호 관리 애플리케이션은 사용자의 비밀번호 정보를 데이터베이스에 저장한다. 따라서 사용자의 개인 정보들이 들어 있는 데이터베이스를 안전하게 보호하는 것은 매우 중요하다. 본 논문에서는 암호화된 데이터베이스를 복호화하기 위한 키를 저장하는 위치에 따라 애플리케이션을 분류하고, 각 경우 보안성이 어떻게 달라지는지에 관한 연구를 수행했다. 본 논문에서는 데이터베이스 복호화키 저장 위치를 4가지로 분류하였다. 첫째. 데이터베이스 암호화를 수행하지 않아, 복호화키가 존재하지 않는 경우. 둘째. 기기 내부에 복호화 키를 저장하거나 내부 정보를 통해 복호화 키를 생성하는 경우, 셋째. 애플리케이션 시작 시 사용자가 복호화키를 입력하는 경우. 넷째. 복호화 키를 서버에서 보관하고 있는 경우이다. 본 논문은 이렇게 분류한 데이터베이스 복호화 키 저장 위치에 따른 데이터베이스의 보안성에 관한 연구를 수행하였으며, 데이터베이스 복호화 키 저장 위치에 따른 특성을 보여준다.

## 2. 관련 연구

Digital forensic analysis of encrypted database files in instant messaging applications on Windows operating systems: Case study with KakaoTalk, NateOn and QQ messenger[1] 연구에서 PC 환경에서 카카오톡, 네이트온, QQ 등의 메신저 애플리케이션의 데이터베이스 내부 정보를 사용자의 개인정보 없이 복호화할 수 있음을 보여주었다. 카카오톡의 경우 복호화키가 user Id에서 생성됨을 이용하여 복호화키를 생성하여 데이터베이스를 복호

화하였고, NateOn의 경우 서버를 통해 복호화키를 받지만, 복호화키의 길이와 형식이 고정되어 Brute Force attack을 통해 데이터베이스를 복호화할 수 있었다. QQ의 경우, 서버를 통해 복호화 키를 받지만, 패킷 감청을 통해 복호화 키를 얻을 수 있음을 보였다. jiru는 Kakao Talk에서 채팅 메시지만이 암호화돼 있기 때문에, 채팅 빈도수를 기반으로 user Id를 추측할 수 있으며, 이를 통해 복호화 키를 생성할 수 있음을 보였다[2].

### 3. 분석 애플리케이션 선정

메신저 애플리케이션과 비밀번호 관리 애플리케이션은 사용자 100만 이상 애플리케이션을 선정하였다.

메신저 애플리케이션은 Signal, KakaoTalk, Line을 선정하였으며, 비밀번호 관리 애플리케이션은 SafeInCloud, My Passwords를 선정하였다.

### 4. 분석 방법

첫째, 애플리케이션 내부에 데이터베이스 파일이 있는지 확인한다.

둘째, 데이터베이스가 암호화되어 있는지 확인한다.

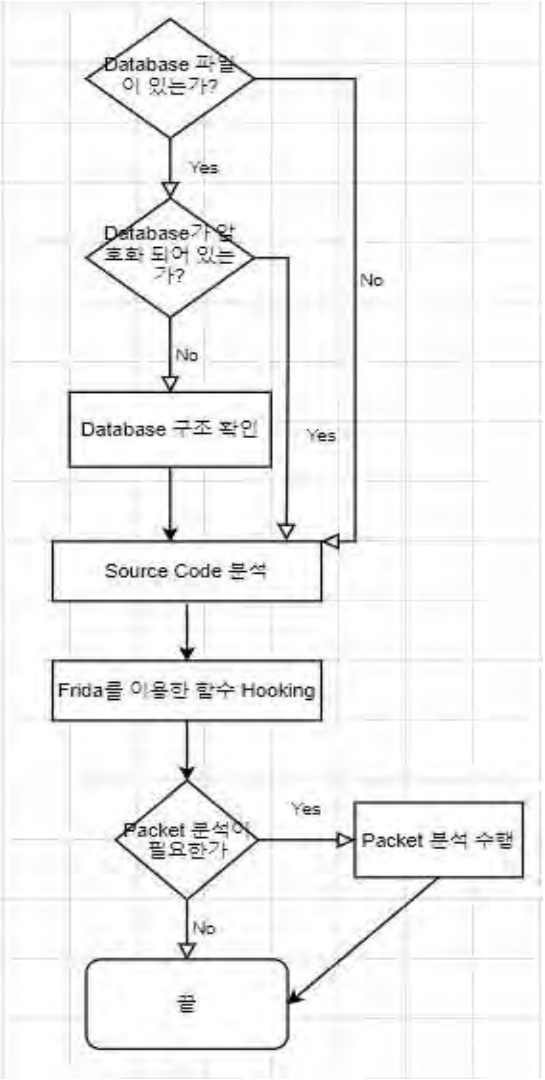
셋째, 데이터베이스가 암호화되어 있지 않다면 데이터베이스의 구조를 확인한다.

넷째, 애플리케이션을 디컴파일하여 소스코드를 분석한다.

다섯째, code injection library Frida를 사용하여, 복호화 함수 등을 확인하고, 복호화 키의 저장 위치를 파악한다.

여섯째, 패킷 분석이 필요할 경우 패킷 분석을 수행한다.

이를 순서도로 나타내면 다음과 같다.




<그림 1> 분석 방법 순서도

### 5. 연구 결과

메신저 애플리케이션

Line과 Signal의 경우 네트워크 연결 없이 사용 가능하며, 데이터베이스가 암호화되어 있지 않아, 채팅 기록 등의 정보를 바로 확인할 수 있다. Line의 경우는 데이터베이스에서 종단간 암호화에 사용되는 개인키, 공개키 또한 확인할 수 있다.

Kakao Talk의 경우 네트워크 연결 없이 사용 가능하며, 채팅 메시지만이 암호화되어 있다. 채팅 메시지만이 암호화돼 있기 때문에, 채팅 빈도수를 기반으로 user Id를 추측할 수 있으며, 이를 통해 복호화 키를 생성

할 수 있다.

비밀번호 저장 애플리케이션

SafeInCloud와 MyPasswords 모두 네트워크 연결 없이 사용할 수 있다. 데이터베이스의 모든 내용이 암호화되어 있다. SafeInCloud의 경우 데이터베이스의 모든 내용이 암호화되어 있으며, database 폴더가 아닌 다른 폴더에 개인정보를 저장하는 데이터베이스 파일을 저장하고 있다. SafeInCloud은 애플리케이션 시작 시 사용자가 복호화키를 입력하는 경우로 사용자가 입력한 복호화 키를 복호화 키를 사용하여 암호화하여 보관하기 때문에 복호화 키를 모른다면 복호화 키를 얻거나 데이터베이스를 복호화할 수 없다.

MyPasswords의 경우 데이터베이스의 테이블 명은 고정된 문자열로 되어 있으며, 내부의 정보만 암호화되어 있다. 애플리케이션 시작 시 사용자가 복호화키를 입력하는 경우로 복호화키가 SHA-256 알고리즘으로 해싱되어 shared preference에 저장되어 있다. SafeInCloud와 마찬가지로 복호화 키를 모른다면 데이터베이스를 복호화할 수 없다.

| 애플리케이션명 | 네트워크 연결 필요 여부 | 데이터베이스 암호화 여부 | 데이터베이스 암호화 범위 | 복호화키 저장위치 | 복호화키 획득가능 여부 |
|---|---|---|---|---|---|
| LINE | X | X | X | X | |
| Signal | X | X | X | X | |
| KakaoTalk | X | O | 채팅 기록 | 애플리케이션 내부 | O |
| SafeInCloud | X | O | 데이터베이스 전체 | 사용자 입력 | X |
| MyPasswords | X | O | 데이터베이스 내부 정보 전체 | 사용자 입력 | X |

<표 1> 애플리케이션 분석 결과

## 6. 결론

본 논문에서는 데이터베이스 복호화키 저장 위치를 4가지로 분류하였다. 첫째. 데이터베이스 암호화를 수행하지 않아, 복호화키가 존재하지 않는 경우. 둘째. 기기 내부에 복호화 키를 저장하거나 내부 정보를 통해 복호화 키를 생성하는 경우, 셋째. 애플리케이션 시작 시 사용자가 복호화키를 입력하는 경우. 넷째. 복호화 키를 서버에서 보관하고 있는 경우이다.

첫 번째 case의 경우, Database 내부의 정보를 암호화하지 않으므로, 사용자의 정보 없이 확인할 수 있다.

두 번째 case의 경우, 데이터베이스 전체 혹은 부분적으로 암호화되어 있지만, 내부 정보를 통해 복호화 할 수 있다. 특히 네트워크 연결 없이 사용할 수 있는 메신저 애플리케이션의 경우, 대부분 사용자의 정보 없이 데이터베이스 복호화를 수행할 수 있음을 알 수 있다.

세 번째 case의 경우, 애플리케이션 내부의 정보를 사용하는 대신 사용자만 알고 있는 복호화 키를 사용하므로, 사용자가 입력한 비밀번호가 올바른 비밀번호인지 확인하기 위한 절차에 문제점이 없다면 안전하다는 것을 알 수 있다.

네 번째 case의 경우, 관련 연구에서 제시 었던, 고정된 길이와 형식이거나 패킷이 감청되는 경우를 제외하면 안전하다는 것을 알 수 있다.

본 논문은 이렇게 분류한 데이터베이스 복호화 키 저장 위치에 따른 데이터베이스의 보안성에 관한 연구를 수행하였으며, 데이터베이스 복호화 키 저장 위치에 따른 특성을 연구하였다.

## ACKNOWLEDGEMENT

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1A2C1012708).

## 참고문헌

[1]Jusop Choi, Jaegwan Yu, Sangwon Hyun, Hyoungshick Kim, “Digital forensic analysis of encrypted database files in instant messaging applications on Windows operating systems: Case study with KakaoTalk, NateOn and QQ messenger”, Digital Investigation Volume 28, Supplement, Pages S50-S59, April 2019

[2]jiru, kakaodecrypt [Internet], https://github.com/jiru/kakaodecrypt