

세션 데이터베이스 복호화 및 주요 아티팩트 분류

2조

20232106 정유진 / 20212052 이동훈

alice2002@kookmin.ac.kr / dhsama51@kookmin.ac.kr

25.04.01

목차

Android Keystore의 개념

데이터베이스 복호화(첨부파일 미포함)

1. 환경 세팅
2. 아티팩트 쌓기
3. 아티팩트 추출
4. 아티팩트 분석(주요 아티팩트 분류)
5. 어플리케이션 분석
6. 향후 연구 계획

Android Keystore의 개념

Android 기기에서 암호키를 안전하게 관리하는 시스템.

TEE (Trusted Execution Environment), StrongBox 등에 의해 키가 보안 HW 외부에 노출되지 않음

→ 기기 외부로 복사하거나 내보낼 수 없음

앱은 키 생성, 저장, 암/복호화, 서명/검증을 위임함

그러나 Nox 등 에뮬레이터는 루팅 + HW 보안 모듈이 없음

→ 평문 또는 약한 보호 상태로 저장되어서 추출 가능



데이터베이스 복호화 - 환경 세팅

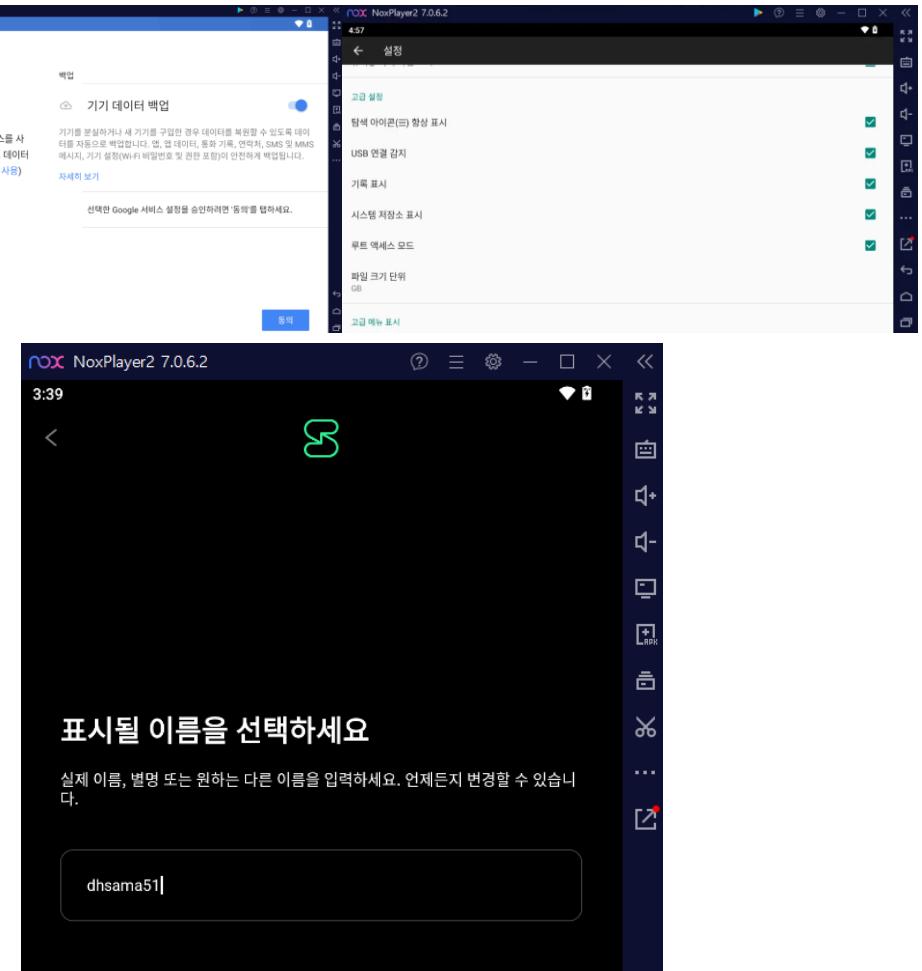
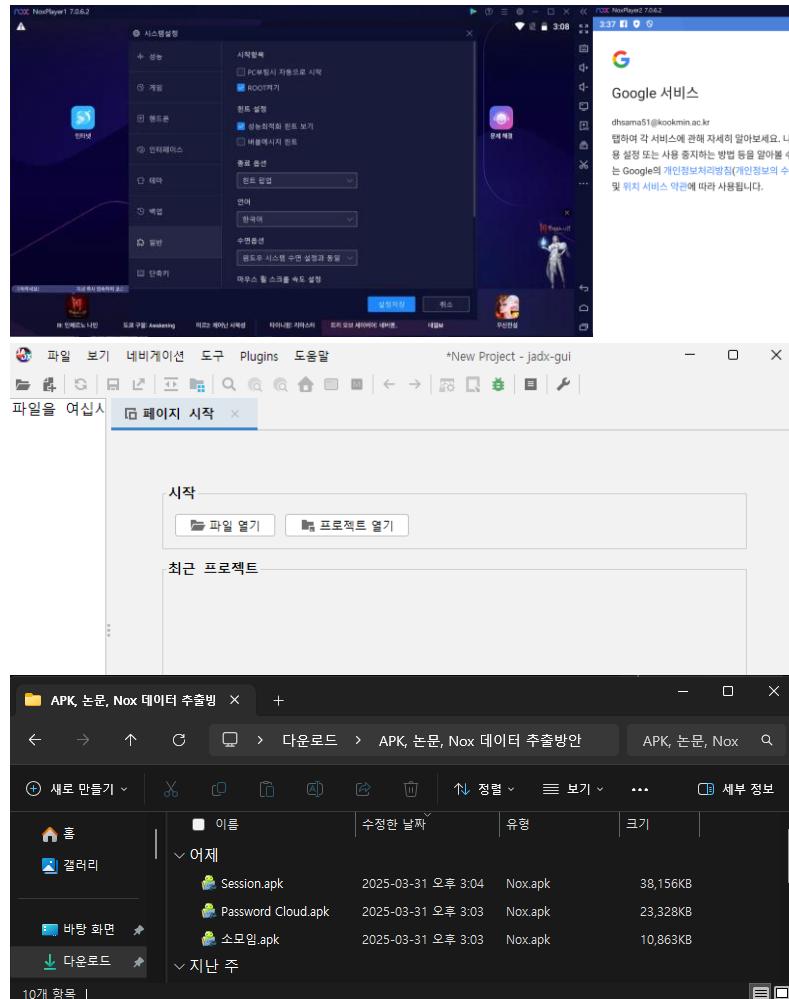
Nox, jadx, Python 사용

Nox - Root, 백업 설정

파일 탐색기 설정

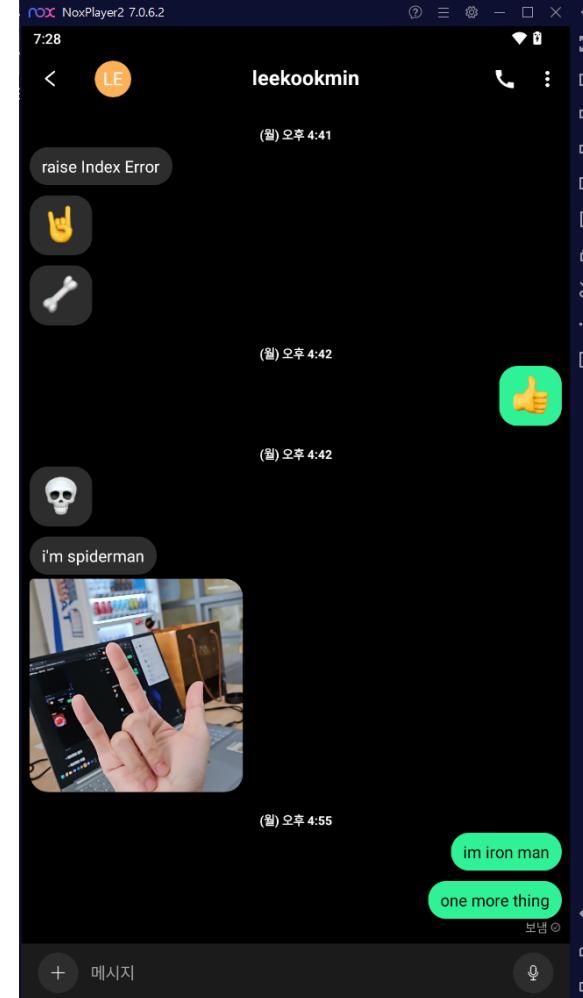
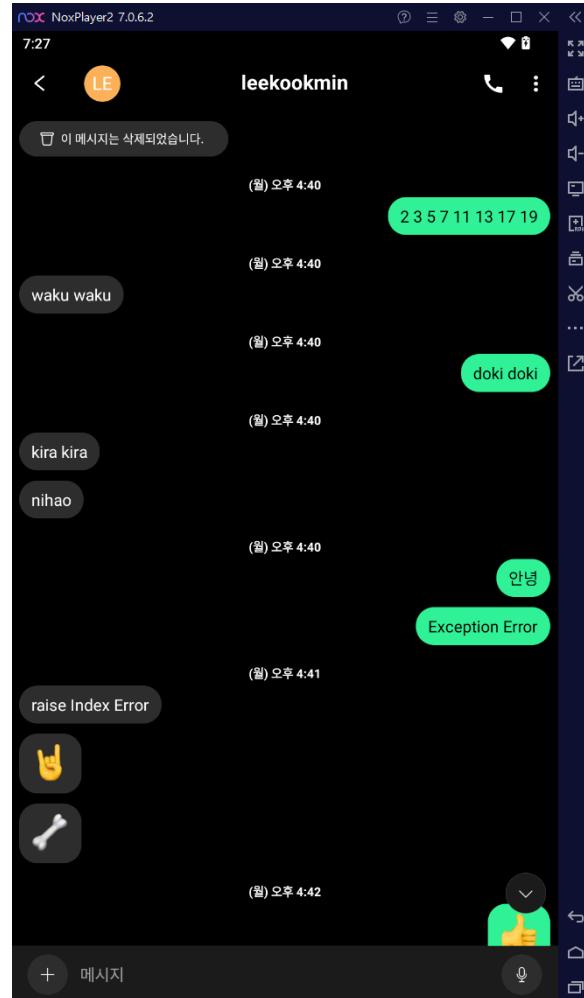
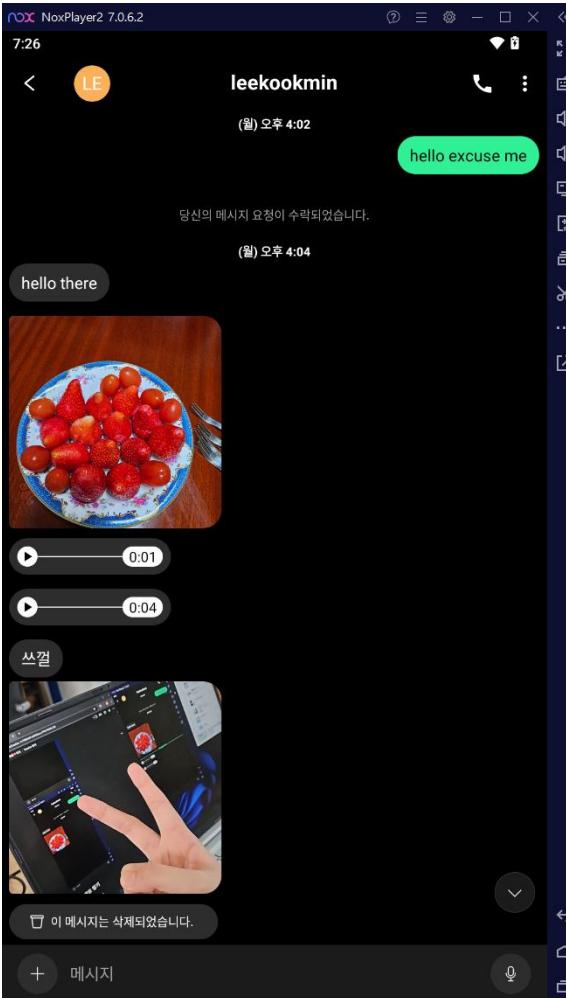
jadx - Session.apk 준비

세션 회원가입



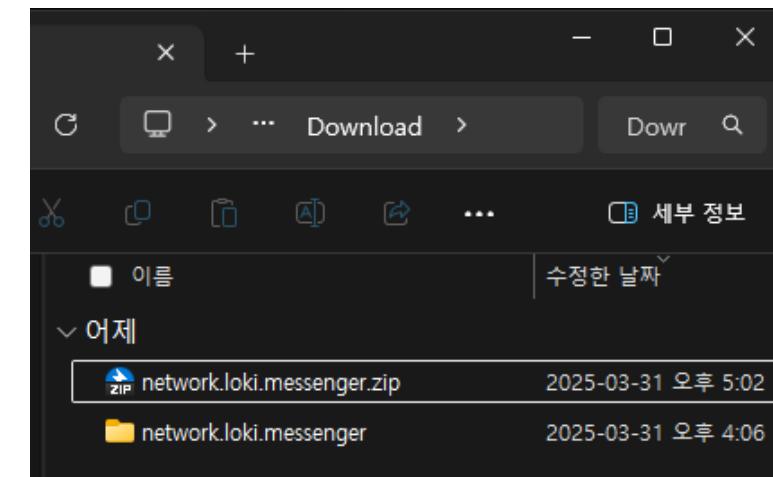
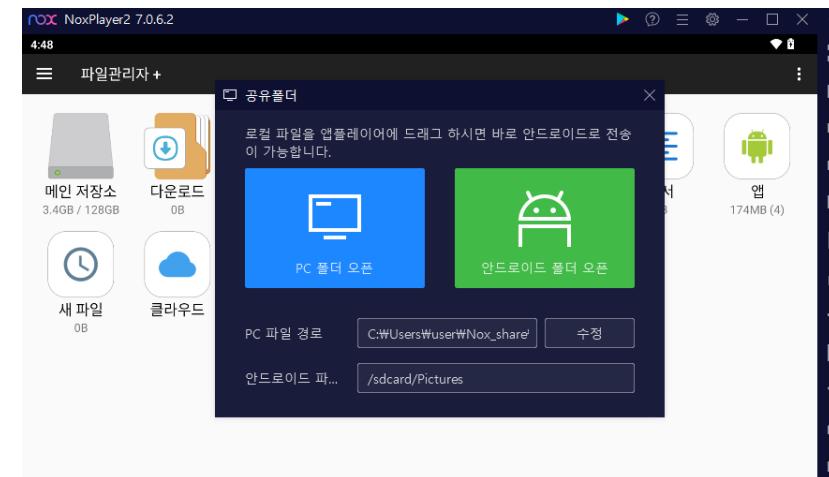
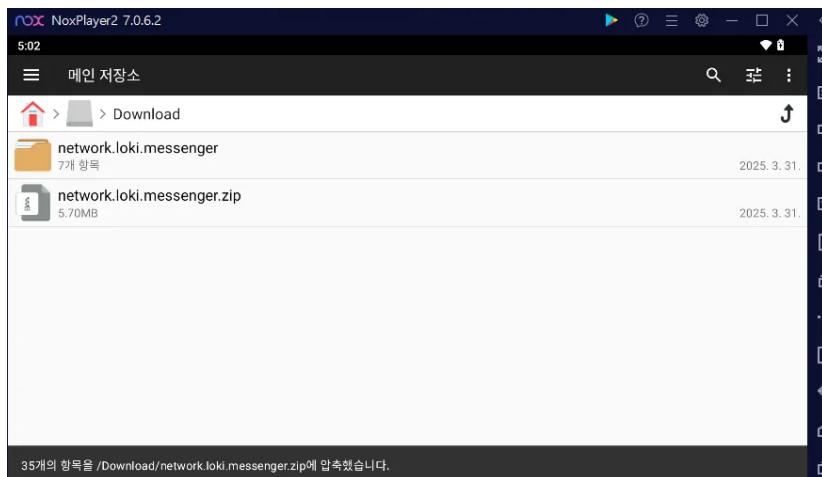
데이터베이스 복호화 - 아티팩트 쌓기

일반 메시지, 사진, 음성 파일, 이모티콘 등 전송, 수신 후 삭제한 메시지 생성



데이터베이스 복호화 - 아티팩트 추출

network.loki.messenger 폴더 Download로 복사, Windows로 이동

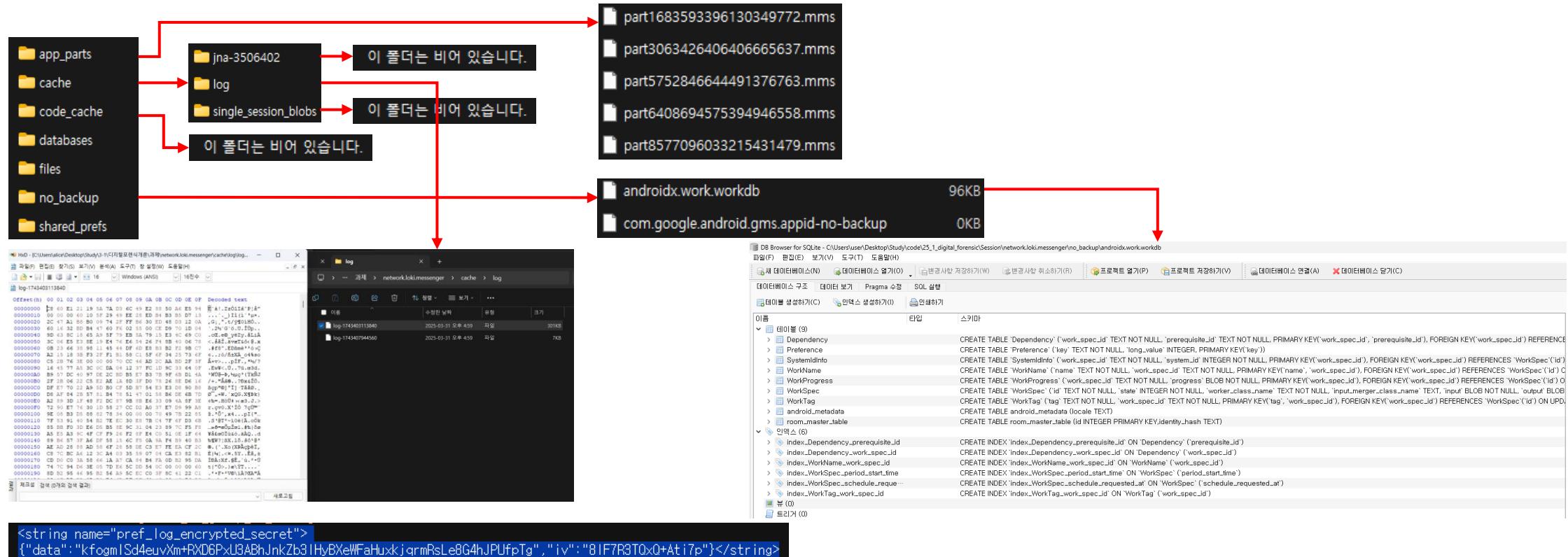


데이터베이스 복호화 - 아티팩트 분석 (1)

app_parts: mms 파일인 것으로 보아 멀티미디어 파일(사진, 음성 파일)일 가능성 있음

cache/log: 특이사항을 찾을 수 없고, log 파일 암호화 가능성 있음

no_backup/android.wor.workdb: 앱 설정과 관련된 내용으로 보임

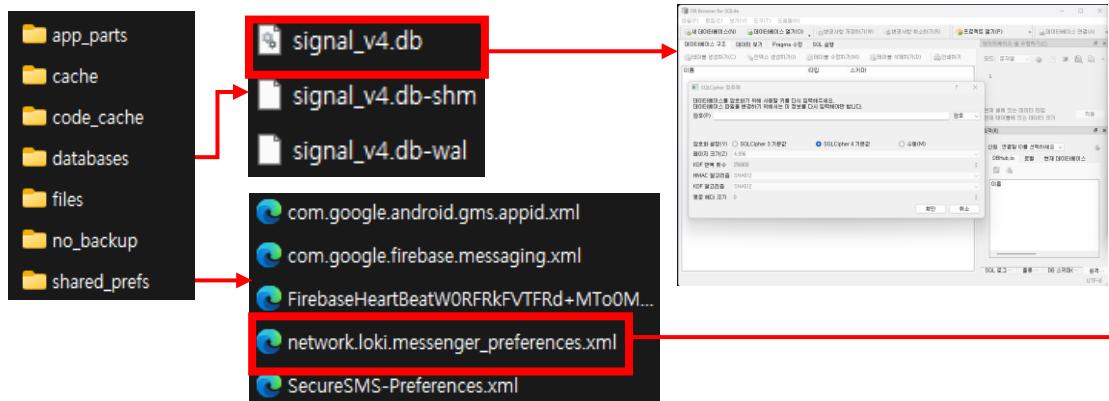


데이터베이스 복호화 - 아티팩트 분석 (2)

shared_prefs/network.loki.messenger_preferences.xml: 암호 정보, 민감한 정보 포함

pref_last_vacuum_time, pref_attachment_encrypted_secret, pref_profile_key,
pref_log_encrypted_secret, pref_database_encrypted_secret, pref_profile_name

databases/signal_v4.db: 데이터베이스가 암호화되어 있음

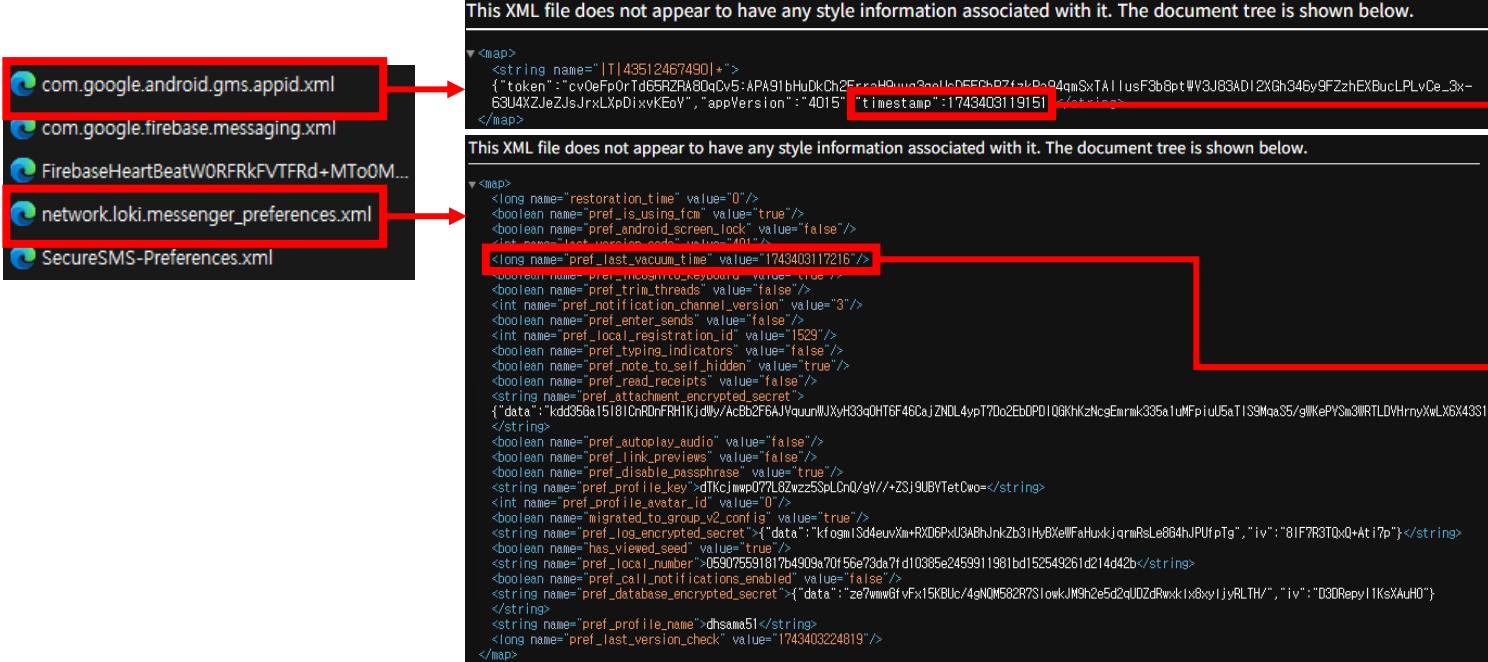


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<map>
    <long name="restoration_time" value="0"/>
    <boolean name="pref_is_using_icm" value="true"/>
    <boolean name="pref_android_screen_lock" value="false"/>
    <int name="last_version_code" value="401"/>
    <long name="pref_last_vacuum_time" value="1743403117216"/>
    <boolean name="pref_incognito_keyboard" value="true"/>
    <boolean name="pref_trim_threads" value="false"/>
    <int name="pref_notification_channel_version" value="3"/>
    <boolean name="pref_enter_sends" value="false"/>
    <int name="pref_local_registration_id" value="1529"/>
    <boolean name="pref_ttyping_indicators" value="false"/>
    <boolean name="pref_note_to_self_hidden" value="true"/>
    <boolean name="pref_read_receipts" value="false"/>
    <string name="pref_attachment_encrypted_secret">
        <data>:kdd356a15181c1nRDnFRH1KjdW/AcBb2F6AJVquonWJXyH3q0HT6F46CaJZNDL4ypT7D02EbDPD1QGKhKzNcgEmmk335a1uMFpiu5aT1S9MqaS5/gWKePVSm3WRTLDvHrnyXwLX6X43SII</data>
    </string>
    <boolean name="pref_autoplay_audio" value="false"/>
    <boolean name="pref_link_previews" value="false"/>
    <boolean name="pref_disable_passphrase" value="true"/>
    <string name="pref_profile_key">dTKcjnwpe0??L8Zwz5SpLCnQ/gV/+ZSJ9UBVTetCwo=</string>
    <int name="pref_profile_avatar_id" value="0"/>
    <boolean name="pref_did_notify_to_group" value="true"/>
    <string name="pref_log_encrypted_secret">(<data>:kfogm1Sd4euVm+RXD6PxU3ABhJnkZb31HyBxeWFaHuxkjqrnRsLe8G4hJPufpTg", "iv": "81F7R3TQxQ+Ati7p")</string>
    <boolean name="has_viewed_seed" value="true"/>
    <string name="pref_local_number">059075591817b4909a70f56e73d7f10305e2459911981bd152549261d214d42b</string>
    <boolean name="pref_call_notifications_enabled" value="false"/>
    <string name="pref_database_encrypted_secret">(<data>:ze?www6fvFx15KB0Uc/4gNOM582R3IowkJM9h2e5d2aU0ZdRwxkIx8xyIjyRLTH/, "iv": "D3DRepy11KsXAuHO")</string>
    <string name="pref_profile_name">dhssama51</string>
    <long name="pref_last_version_check" value="1743403224819"/>
</map>
```

데이터베이스 복호화 - 아티팩트 분석 (3)

shared_prefs/network.loki.messenger_preferences.xml의 **pref_last_vacuum_time**과
shared_prefs/com.google.android.gms.appid.xml의 **timestamp** 비교
Vacuum time은 vacuum 작업(로컬 DB 최적화 관리) 시간으로 추정

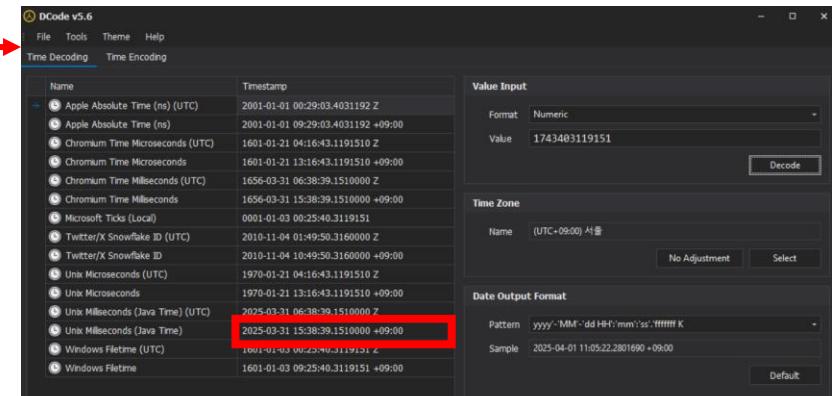
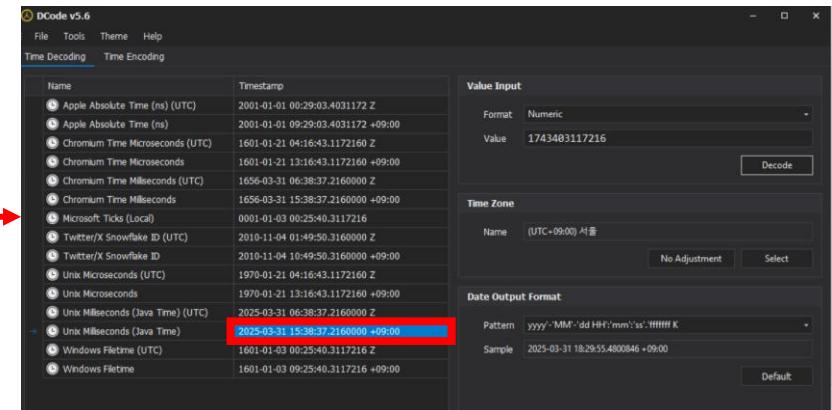


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<map>
    <string name="T|143512467490|*>
        {"token": "cv0eFp0rTd65RZA80aCv5:APA9t1bHu0kCh2FrroHpu...9qjSxTAIIusF3b8ptW3J83AD12XGh346y9FZzhEXBucLPLvCe_3x-63U4XZjeZjsJrLxpDixKEoY", "appVersion": "4015", "timestamp": "1743403119151"}</map>
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<map>
    <long name="restoration_time" value="0"/>
    <boolean name="pref_is_using_fcm" value="true"/>
    <boolean name="pref_android_screen_lock" value="false"/>
    <int name="pref_version_code" value="4015"/>
    <long name="pref_last_vacuum_time" value="1743403117216"/>
    <boolean name="pref_imosimo_keyboard" value="true"/>
    <boolean name="pref_trim_threads" value="false"/>
    <int name="pref_notification_channel_version" value="3"/>
    <boolean name="pref_enter_sends" value="false"/>
    <int name="pref_local_registration_id" value="1529"/>
    <boolean name="pref_typering_indicators" value="false"/>
    <boolean name="pref_note_to_self_hidden" value="true"/>
    <boolean name="pref_read_receipts" value="false"/>
    <string name="pref_attachment_encrypted_secret">
        {"data": "kdd35Ga15181CrRDnFRH1KdlyAcBb2F6A1VquuNjXjH3s3oHT6F46CaJZNDL4ypT7D02Eb0P0I08KhKzNc9Earmk335a1uMfp1uJ5aT1S9Mqa35/gllKePVSm3IRTL0WhrrnyXwLX6X43SII", "type": "string"}</string>
    <boolean name="pref_autoplay_audio" value="false"/>
    <boolean name="pref_link_reviews" value="false"/>
    <boolean name="pref_disable_passphrase" value="true"/>
    <string name="pref_profile_key">dKc1mwp07L6zWzz5SpLCn0/qV//+ZSJ9UBYTetCwo=</string>
    <int name="pref_remote_avatar_id" value="0"/>
    <boolean name="migrated_to_group_v2_config" value="true"/>
    <string name="pref_link_encrypted_secret">{"data": "kfqm1Sd4euVn+RX0DPxUsABhInkzb3IHy8XellFaHuskjqrmRsLe864hJPUpfTg", "iv": "81F7R3TQxQ+A1t7p"}</string>
    <boolean name="has_viewed_seed" value="true"/>
    <string name="pref_local_number">50907181b4909a7015673d7fd10385e2459911981bd152549261d21d442b</string>
    <boolean name="pref_call_notifications_enabled" value="false"/>
    <string name="pref_database_encrypted_secret">{"data": "ze7www6tvfx15k8Uc/4gNQm32R7SlowlJM9h2e5d2qJ02dRwxk18xyijyRLTH/", "iv": "D3DRepy1IKsXAh0"}</string>
    <string name="pref_profile_name">dhsmal1</string>
    <long name="pref_last_version_check" value="1743403224819"/>
</map>
```



데이터베이스 복호화 - 어플리케이션 분석 (1)

사전 정보

주요 데이터를 데이터베이스에 보관하고, 이를 암호화

AES256/GCM/NoPadding 사용

Android Keystore의 구조를 분석하여 암호키 추출

Keymaster: Android Keystore의 모든 키를 관리하는 키.

Blob 형태로 저장되어 있으며 다음과 같은 정보가 들어있음

①Android Keystore 버전 ②Nonce

③암호키 ④암호키 인증을 위한 Tag 및 속성 정보

→ Keymaster Blob이라고 불림

보안 인스턴트 메신저 Session 데이터베이스
복호화 방안 연구

박지수*, 박세준*, 김기윤**, 김종성***

*, **, ***국민대학교(학생, 대학원생, 교수)

Decryption Methods for Secure Instant Messenger Application
Session Database

JiSu Park*, SeJun Park*, GiYoon Kim**, JongSung Kim***

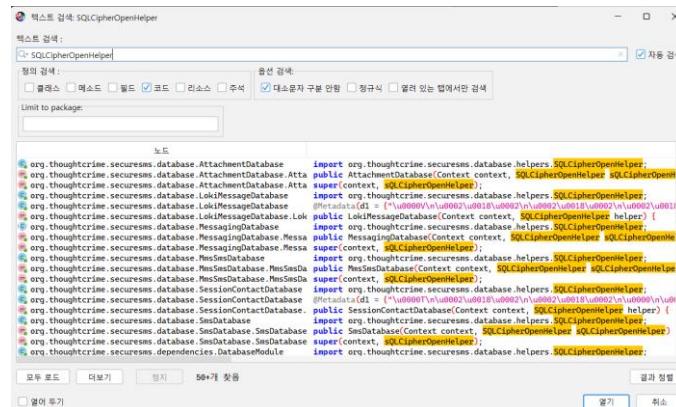
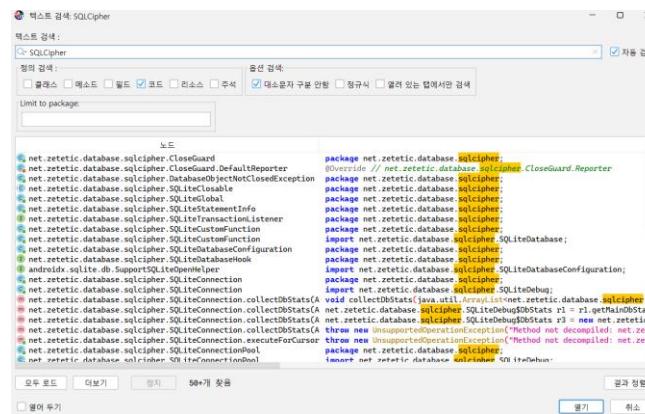
*, **, ***Kookmin University(Student, Graduate student, Professor)

데이터베이스 복호화 - 어플리케이션 분석 (2)

데이터베이스 암호화 정보 확인

SQLCipher로 암호화되어 있음 → 관련 코드 검색

암호화된 데이터베이스 이름인 signal_v4.db 확인



```
/* loaded from: classes4.dex */
public class SQLCipherOpenHelper extends SQLiteOpenHelper {
    private static final String CIPHER3_DATABASE_NAME = "signal.db";
    public static final String DATABASE_NAME = "signal_v4.db"; // highlighted
    private static final int DATABASE_VERSION = 69;
    private static final int MIN_DATABASE_VERSION = 28;
    private static final String TAG = "SQLCipherOpenHelper";
    private static final int lokiV10 = 31;
    private static final int lokiV11 = 32;
    private static final int lokiV12 = 33;
    private static final int lokiV13 = 34;
    private static final int lokiV14_BACKUP_FILES = 35;
    private static final int lokiV15 = 36;
    private static final int lokiV16 = 37;
    private static final int lokiV17 = 38;
    private static final int lokiV18_CLEAR_BG_POLL_JOBS = 39;
    private static final int lokiV19 = 40;
    private static final int lokiV20 = 41;
    private static final int lokiV21 = 42;
    private static final int lokiV22 = 43;
    private static final int lokiV23 = 44;
    private static final int lokiV24 = 45;
    private static final int lokiV25 = 46;
}
```

Jadx decompiler를 통해 분석해본 결과 signal_v4.db 데이터베이스는 SQLCipher로 암호화되었다. 다음은 실제 소스코드 상에서 확인 가능한 SQLCipher를 활용한 데이터베이스 오픈 과정이다(그림 2).

SQLCipher로 암호화되었다.

데이터베이스 복호화 - 어플리케이션 분석 (3)

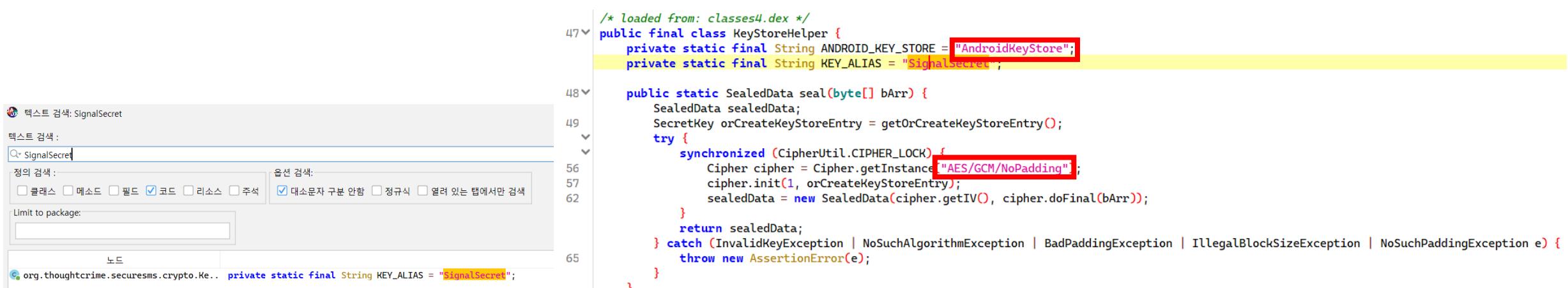
Session 암호키 정보

논문에 언급된 SignalSecret 검색 → 결과 1건

SignalSecret과 AndroidKeyStore이 단 1건 있고,

AES/GCM/NoPadding 사용을 직접 확인

SQLCipher에 사용된 패스프레이즈는 암호화되어 Shared Preference에 저장된다. 이때 암호 알고리즘은 AES/GCM/NoPadding 이었으며, 암호키는 Android Keystore에 저장되었다. Android Keystore에서 Session의 암호키를 구분하기 위한 별칭은 SignalSecret이다(그림 3).



/* loaded from: classes4.dex */
public final class KeyStoreHelper {
 private static final String ANDROID_KEY_STORE = "AndroidKeyStore";
 private static final String KEY_ALIAS = "SignalSecret";

 public static SealedData seal(byte[] bArr) {
 SealedData sealedData;
 SecretKey orCreateKeyStoreEntry = getOrCreateKeyStoreEntry();
 try {
 synchronized (CipherUtil.CIPHER_LOCK) {
 Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding");
 cipher.init(1, orCreateKeyStoreEntry);
 sealedData = new SealedData(cipher.getIV(), cipher.doFinal(bArr));
 }
 return sealedData;
 } catch (InvalidKeyException | NoSuchAlgorithmException | BadPaddingException | IllegalBlockSizeException | NoSuchPaddingException e) {
 throw new AssertionError(e);
 }
 }
}

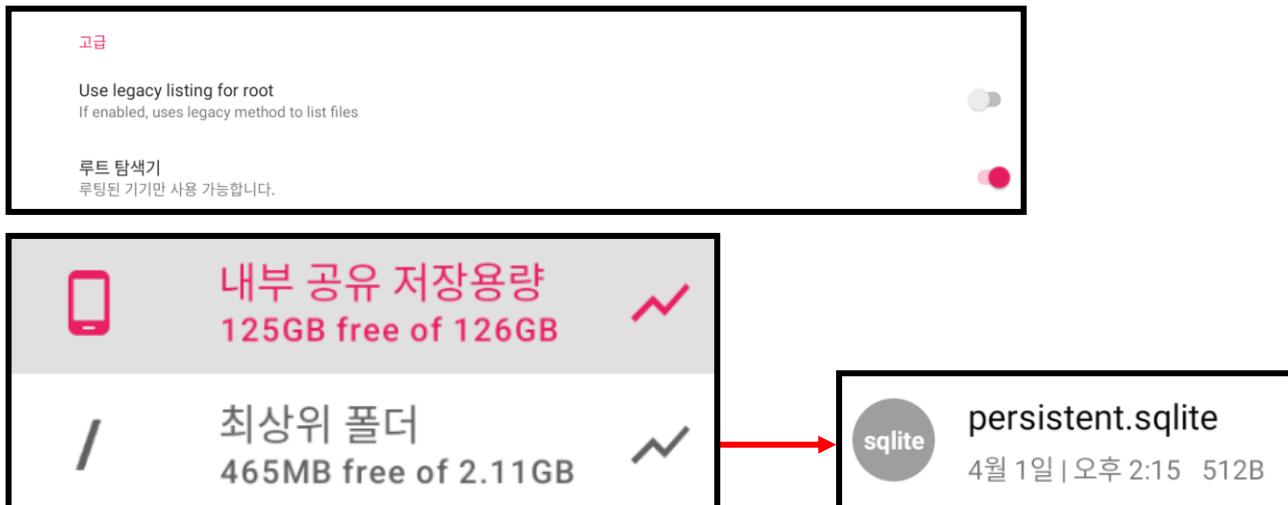
텍스트 검색: SignalSecret
텍스트 검색 :
SignalSecret
정의 검색 :
 클래스 메소드 필드 코드 리소스 주석
옵션 검색:
 대소문자 구분 안함 정규식 열려 있는 탭에서만 검색
Limit to package:
노드
org.thoughtcrime.securesms.crypto.Ke... private static final String KEY_ALIAS = "SignalSecret";

데이터베이스 복호화 - 어플리케이션 분석 (4)

Android Keystore에서 암호키 추출 (1)

Amaze 파일 매니저 이용해 persist.sqlite 추출

안드로이드 12 이상 버전에서의 암호키는 USERDATA/misc/keystore내 persist.sqlite 데이터베이스에 저장된다. 해당 데이터베이스에서



데이터베이스 복호화 - 어플리케이션 분석 (5)

Android Keystore에서 암호키 추출 (2)

keyentry에서 SignalSecret 발견,
blobentry에서 같은 id를 가진 것 발견
0xD~0x1C에서 암호키 추출

이름	타입	테이블(T): keyentry
		id key_type domain namespace alias
blobentry	테이블(7)	1 6926609068309298972 0 0 1017 boot_level_key
blobmetadata	테이블(7)	2 -585827675954575484 0 0 10046 c5635988b7157a3a
grant	테이블(7)	3 -1154276977243341612 0 0 10046 mobstore_encrypt
keyentry	테이블(7)	4 6044086939184787821 0 0 10046 nearby_presence_connection_sec
keymetadata	테이블(7)	5 -6262722488817596635 0 0 10046 nearby_presence_connection_sec
keyparameter	테이블(7)	6 -1994549063571697783 0 0 10046 nearby_presence_connection_sec
version	테이블(7)	7 -3169231379557241686 0 0 10046 nearby_presence_connection_sec
인덱스(6)	인덱스(6)	8 3392414123250361534 0 0 10046 nearby_presence_connection_sec
blobentry_keyentryid_index	인덱스(6)	9 -7823036944918506034 0 0 10046 nearby_presence_connection_sec
blobmetadata_blobentryid_index	인덱스(6)	10 -5423159458800941720 0 0 10046 nearby_presence_connection_sec
keyentry_domain_namespace_index	인덱스(6)	11 6940908571265112464 0 0 10046 nearby_presence_connection_sec
keyentry_id_index	인덱스(6)	12 -9167606571973839702 0 0 10046 nearby_presence_connection_sec
keymetadata_keyentryid_index	인덱스(6)	13 -4036893587965478311 0 0 10062 SignalSecret

이름	타입	테이블(T): blobentry
		id subcomponent_type keyentryid blob
blobentry	테이블(7)	4 4 0 -1154276977243341612 BLOB
blobmetadata	테이블(7)	5 7 0 6044086939184787821 BLOB
grant	테이블(7)	6 10 0 -6262722488817596635 BLOB
keyentry	테이블(7)	7 13 0 -1994549063571697783 BLOB
keymetadata	테이블(7)	8 16 0 -3169231379557241686 BLOB
keyparameter	테이블(7)	9 19 0 3392414123250361534 BLOB
version	테이블(7)	10 22 0 -7823036944918506034 BLOB
blobentry_keyentryid_index	인덱스(6)	11 25 0 -5423159458800941720 BLOB
blobmetadata_blobentryid_index	인덱스(6)	12 28 0 6940908571265112464 BLOB
keyentry_domain_namespace_index	인덱스(6)	13 31 0 -9167606571973839702 BLOB
keyentry_id_index	인덱스(6)	14 32 0 -4036893587965478311 BLOB

데이터베이스 셀 수정하기(C)	
모드:	바이너리
0000	70 4b 4d 62 6c 6f 62 00 00 10 00 00 00 79 fa 7c pKMblob.....v.l
0010	4a 90 a5 53 95 b1 1a 2b 49 7a 82 57 8d 00 00 00 J..S...+Iz.W...
0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030	00 61 00 00 00 03 00 00 30 80 00 00 00 02 00 00 ..a.....0.....
0040	10 20 00 00 00 01 00 00 20 00 00 00 00 01 00 00
0050	20 01 00 00 00 08 00 00 30 60 00 00 00 04 00 00 ..0`.....0.....
0060	20 20 00 00 00 06 00 00 20 01 00 00 00 f7 01 00
0070	70 01 bd 02 00 60 43 5c ec ea 95 01 00 00 be 02 p....`C\.....
0080	00 10 00 00 00 00 c1 02 00 30 c0 d4 01 00 c2 02 ..0`.....0.....
0090	00 30 dd 15 03 00 90 9b 4c f8 38 9a 10 68 .0.....L.8..h

Keymaster Blob을 획득하기 위해서는 keyentry 테이블과 blobentry 테이블을 활용해야 한다. 먼저, 두 테이블을 연결해주는 고윳값이 필요하다. keyentry 테이블에는 alias 컬럼과 id 컬럼이 존재한다. alias 컬럼 내 Session 암호키의 별칭이 존재하는 열의 id 값이 두 테이블을 연결해주는 고윳값이 된다. 해당 값을 blobentry 테이블의 keyentryid 컬럼에서 매칭한 뒤 같은 열에 존재하는 blob 데이터가 Keymaster Blob이 된다. 해당 Keymaster Blob의 offset 0xD~0x1C에는 암호키가 평문 형태로 저장되어 있다(그림 5).

테이블(T): keyentry
id key_type domain namespace alias state km_uuid
1313877684582408609 0 0 10063 SignalSecret 1 BLOB
테이블(T): blobentry
id subcomponent_type keyentryid blob
65 0 1313877684582408609 pKMblob.....v.l
0000 70 4b 4d 62 6c 6f 62 00 00 10 00 00 00 79 fa 7c pKMblob.....v.l
0010 4a 90 a5 53 95 b1 1a 2b 49 7a 82 57 8d 00 00 00 J..S...+Iz.W...
0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030 00 61 00 00 00 03 00 00 30 80 00 00 00 02 00 00 ..a.....0.....
0040 10 20 00 00 00 01 00 00 20 00 00 00 00 01 00 00
0050 20 01 00 00 00 08 00 00 30 60 00 00 00 04 00 00 ..0`.....0.....
0060 20 20 00 00 00 06 00 00 20 01 00 00 00 f7 01 00
0070 70 01 bd 02 00 60 43 5c ec ea 95 01 00 00 be 02 p....`C\.....
0080 00 10 00 00 00 00 c1 02 00 30 c0 d4 01 00 c2 02 ..0`.....0.....
0090 00 30 dd 15 03 00 90 9b 4c f8 38 9a 10 68 .0.....L.8..h

(그림 3) Android Keystore Key

데이터베이스 복호화 - 어플리케이션 분석 (6)

패스프레이즈 획득

shared_prefs/network.loki.messenger_preferences.xml에서

pref_database_encrypted_secre에서 data, iv 획득

data: "ze7wmwGfvFx15KBUc/4gNQM582R7SlowkJM9h2e5d2qUDZdRwxklx8xyljyRLTH/"

암호화된 패스프레이즈 + GCM 태그 연결

iv: "D3DRepyI1KsXAuHO"

암호화에 사용된 iv가 Base64 인코딩되어 있음

암호화된 패스프레이즈는 network.loki.messenger

_preferences 내 pref_database_encrypted_secret
요소에 JSON 형태로 저장되어 있다(그림 6).

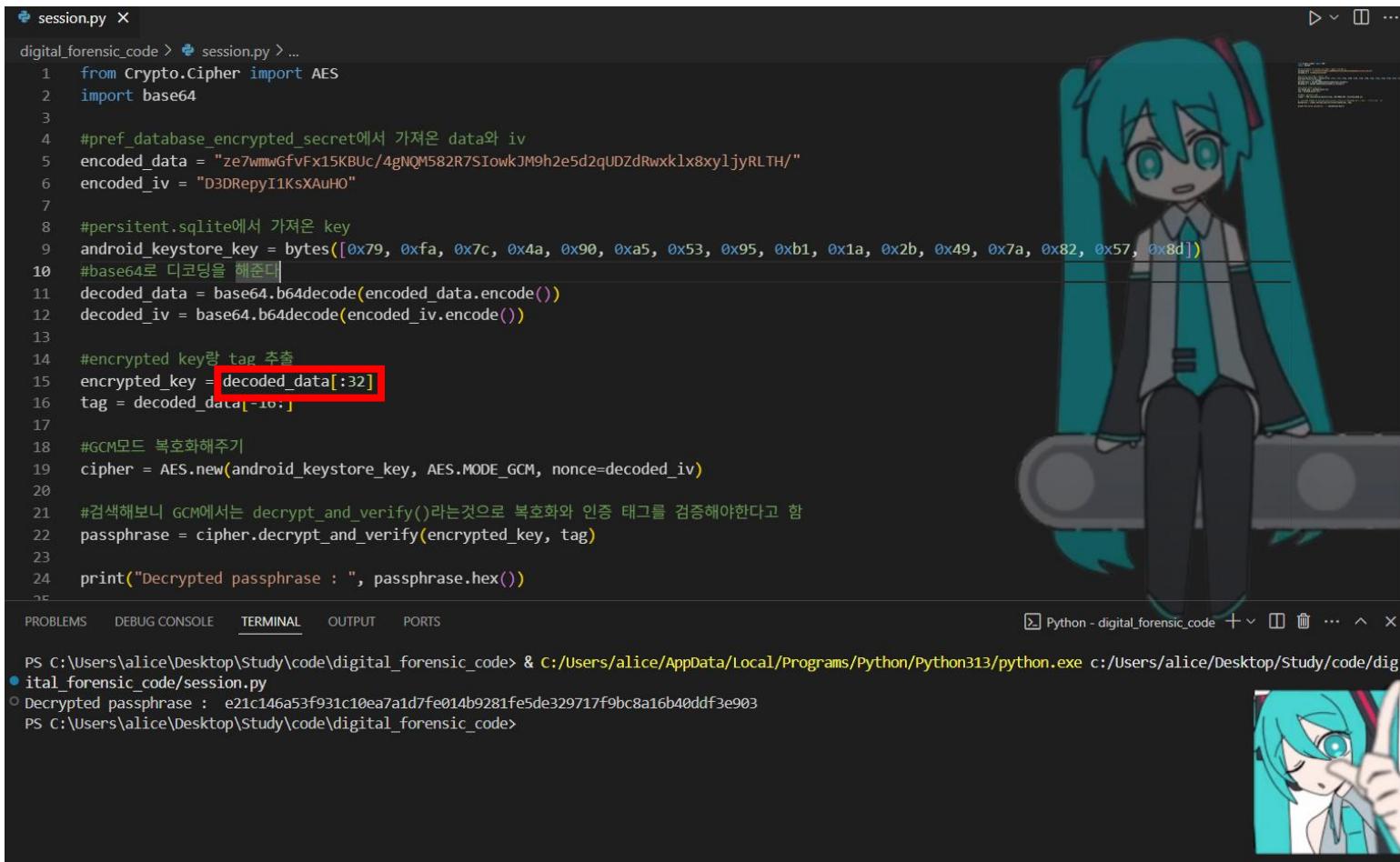
```
<string name="pref_database_encrypted_secret">{"data": "ze7wmwGfvFx15KBUc/4gNQM582R7SlowkJM9h2e5d2qUDZdRwxklx8xyljyRLTH/", "iv": "D3DRepyI1KsXAuHO"}</string>
```

```
<string name="pref_database_encrypted_secret">  
{"data": "ze7wmwGfvFx15KBUc/4gNQM582R7SlowkJM9h2e5d2qUDZdRwxklx8xyljyRLTH/", "iv": "D3DRepyI1KsXAuHO"}  
</string>
```

데이터베이스 복호화 - 어플리케이션 분석 (7)

패스프레이즈 복호화

Python 이용 - 결과: 'e21c146a53f931c10ea7a1d7fe014b9281fe5de329717f9bc8a16b40ddf3e903'



```
session.py
-----
1 from Crypto.Cipher import AES
2 import base64
3
4 #pref_database_encrypted_secret에서 가져온 data와 iv
5 encoded_data = "ze7mmGfvFx15KBUC/4gNQM582R7SIowkJM9h2e5d2qUDZdRwxk1x8xyljyRLTH/"
6 encoded_iv = "D3DRepyI1KsXAUHO"
7
8 #persistent.sqlite에서 가져온 key
9 android_keystore_key = bytes([0x79, 0xfa, 0x7c, 0x4a, 0x90, 0xa5, 0x53, 0x95, 0xb1, 0x1a, 0x2b, 0x49, 0x7a, 0x82, 0x57, 0x8d])
10 #base64로 디코딩을 해준다
11 decoded_data = base64.b64decode(encoded_data.encode())
12 decoded_iv = base64.b64decode(encoded_iv.encode())
13
14 #encrypted key랑 tag 추출
15 encrypted_key = decoded_data[0:32] # This line is highlighted with a red box
16 tag = decoded_data[-16:]
17
18 #GCM모드 복호화해주기
19 cipher = AES.new(android_keystore_key, AES.MODE_GCM, nonce=decoded_iv)
20
21 #검색해보니 GCM에서는 decrypt_and_verify()라는것으로 복호화와 인증 태그를 검증해야한다고 함
22 passphrase = cipher.decrypt_and_verify(encrypted_key, tag)
23
24 print("Decrypted passphrase : ", passphrase.hex())
25
```

PROBLEMS DEBUG CONSOLE TERMINAL OUTPUT PORTS

PS C:\Users\alice\Desktop\Study\code\digital_forensic_code> & C:/Users/alice/AppData/Local/Programs/Python/Python313/python.exe c:/Users/alice/Desktop/Study/code/digital_forensic_code/session.py

● digital_forensic_code/session.py

○ Decrypted passphrase : e21c146a53f931c10ea7a1d7fe014b9281fe5de329717f9bc8a16b40ddf3e903

PS C:\Users\alice\Desktop\Study\code\digital_forensic_code>

Input : pref_database_encrypted_secret, Android_KeyStore_Key
OutPut : Passphrase

1 : IV ← Base64_Decode(pref_database_encrypted_secret.iv)
2 : Data ← Base64_Decode(pref_database_encrypted_secret.data)
3 : Tag ← Data[-16:]
4 : Encrypted_key ← Data[0: 48] # This line is highlighted with a red box
5 : Passphrase ← AES256/GCM/NoPadding_Decrypt
(Android_Keystore_Key, IV, Encrypted_key, Tag)
6 : return Passphrase

(그림 5) Getting PassPhrase of signal_v4.db

Data[0:48] → 복호화된 패스프레이즈 길이 = 48

≠

AES256/GCM/NoPadding 키 길이 = 32

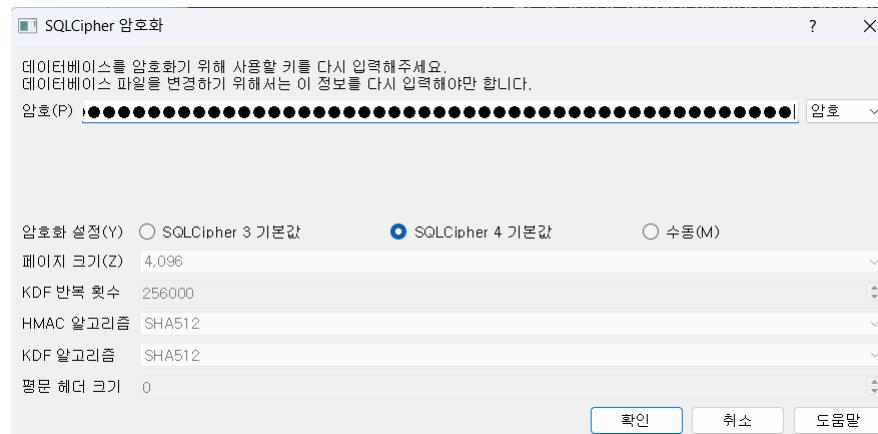
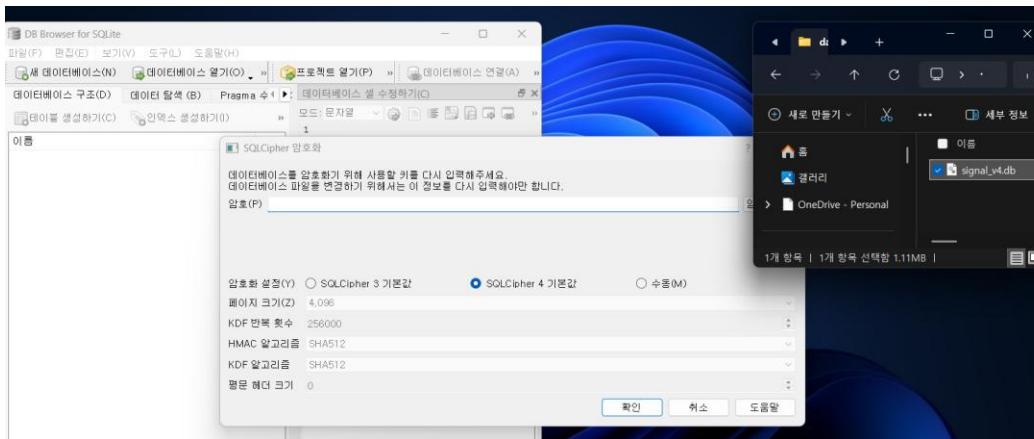
→ Data[0:32]로 진행



데이터베이스 복호화 - 어플리케이션 분석 (8)

데이터베이스 복호화

패스프레이즈 입력을 통해 DB 복호화 확인



The screenshot shows the DB Browser for SQLite interface with the title bar "DB Browser for SQLite - C:\Users\alice\Desktop\Study\3-1\디지털포렌식개론\과제\network.loki.messenger\databases\signal_v4.db". The main window displays the database schema in a table format:

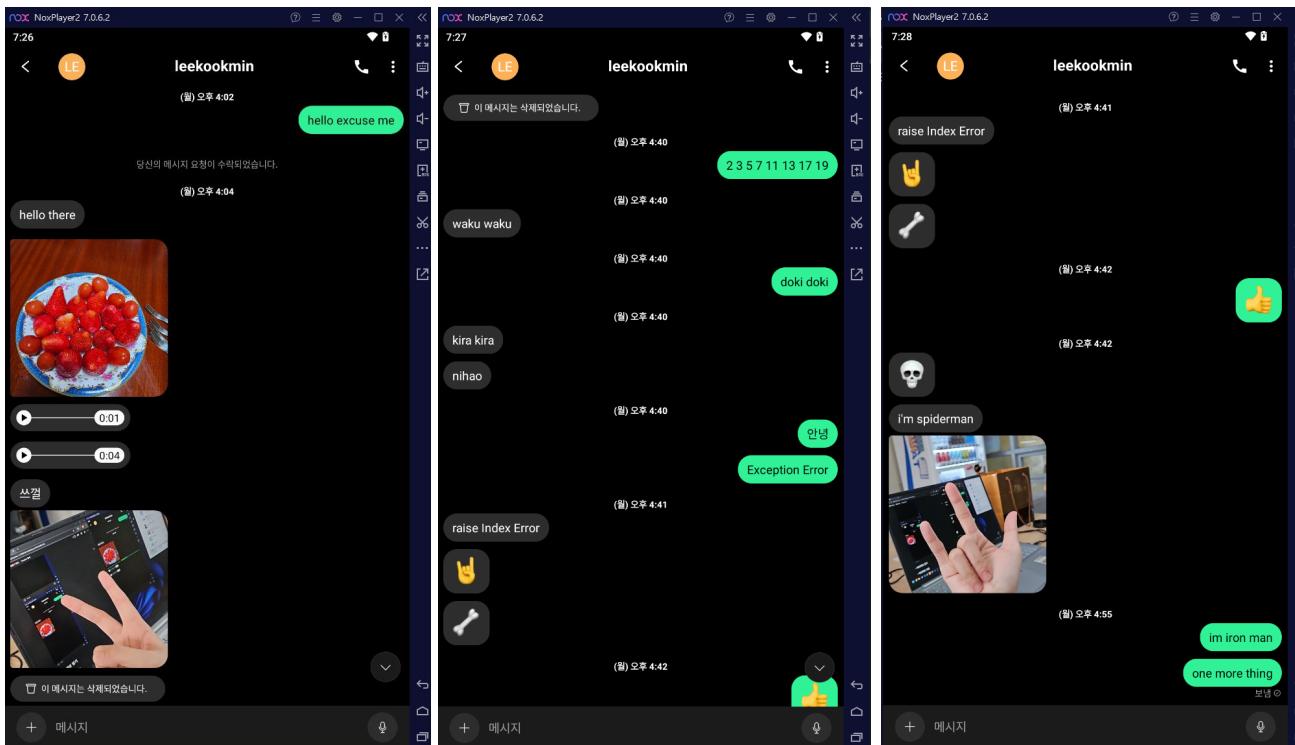
이름	타입	스키마
테이블 (66)		
backup_files	CREATE TABLE b	ackup_files
blinded_id_mapping	CREATE TABLE bl	linded_id_mapping
closed_group_encryption_key_pairs_…	CREATE TABLE cl	osed_group_encryption_key_pairs_…
closed_group_public_keys_table	CREATE TABLE cl	osed_group_public_keys_table
configs_table	CREATE TABLE cc	onfigs_table
drafts	CREATE TABLE dr	afts
emoji_search	CREATE VIRTUAL	emoty_search
emoji_search_config	CREATE TABLE 'e	moji_search_c
emoji_search_content	CREATE TABLE 'e	ontent
emoji_search_data	CREATE TABLE 'e	ata
emoji_search_docsizes	CREATE TABLE 'e	ocsizes
emoji_search_idx	CREATE TABLE 'e	dx
expiration_configuration	CREATE TABLE ex	piration_c
fork_info	CREATE TABLE fo	rk_info
group_member	CREATE TABLE gr	roup_memb
group_receipts	CREATE TABLE gr	oup_recie
groups	CREATE TABLE gr	oups
last_legacy_messages	CREATE TABLE la	st_legacy_m
loki_api_group_chat_auth_token_dat…	CREATE TABLE lo	ki_api_group_
loki_api_last_deletion_server_id_cac…	CREATE TABLE lo	ki_api_las
loki_api_swarm_cache	CREATE TABLE lo	ki_api_sw
loki_error_message_database	CREATE TABLE lo	ki_error_m
loki_group_invites	CREATE TABLE lo	oki_group_i
loki_message_friend_request_database…	CREATE TABLE lo	oki_message_f
loki_message_hash_database	CREATE TABLE lo	oki_message_h

On the right side of the interface, there are several toolbars and panels, including "데이터베이스 셀 수정하기(C)" (Database Cell Modify), "모드: 문자열" (Mode: String), and "활성화된 셀이 없습니다." (No active cell). At the bottom, there are tabs for "SQL", "플", "DB 스키마", and "원".

데이터베이스 복호화 - 어플리케이션 분석 (9)

복호화된 데이터베이스에서 대화 확인

주고받은 텍스트를 확인할 수 있었음



sent	protocol	read	status	type	reply_path_present	delivery_receipt_count	subject	body	mismatched_identities	service_center	subscriber
1 3714414	NULL	1	-1	10485783	NULL	0	NULL	hello_world	NULL	NULL	
2 1522201	NULL	1	-1	10485783	NULL	0	NULL	hello excuse me	NULL	NULL	
3 1697984	31337	1	-1	10485780		1	0	NULL	hello there	NULL	GCM
4 3810215	NULL	1	-1	10485783	NULL	0	NULL	2 3 5 7 11 13 17 19	NULL	NULL	
5 3818811	31337	1	-1	10485780		1	0	NULL	waku waku	NULL	GCM
6 3826267	NULL	1	-1	10485783	NULL	0	NULL	doki doki	NULL	NULL	
7 3827020	31337	1	-1	10485780		1	0	NULL	kira kira	NULL	GCM
8 3832320	31337	1	-1	10485780		1	0	NULL	nihao	NULL	GCM
9 3852568	NULL	1	-1	10485783	NULL	0	NULL	안녕	NULL	NULL	
10 3873138	NULL	1	-1	10485783	NULL	0	NULL	Exception Error	NULL	NULL	
11 3878669	31337	1	-1	10485780		1	0	NULL	raise Index Error	NULL	GCM
12 3906442	31337	1	-1	10485780		1	0	NULL	🔥	NULL	GCM
13 3921921	31337	1	-1	10485780		1	0	NULL	⚡	NULL	GCM
14 3930209	NULL	1	-1	10485783	NULL	0	NULL	🔥	NULL	NULL	
15 3930513	31337	1	-1	10485780		1	0	NULL	💀	NULL	GCM
16 3930513	31337	1	-1	10485780		0	NULL	im iron man	NULL	NULL	
17 3951479	NULL	1	-1	10485783	NULL	0	NULL	one more thing	NULL	NULL	

date	date_received	msg_box	read	m_id	sub	sub_cs	body
1 1743404684515	1743404684515	10551316	1	NULL	NULL	NULL	NULL
2 1743404728201	1743404742385	10485780	1	NULL	NULL	NULL	
3 1743406194801	1743406200522	10485780	1	NULL	NULL	NULL	
4 1743406202226	1743406211511	10485780	1	NULL	NULL	NULL	
5 1743406345332	1743406355472	10485780	1	NULL	NULL	NULL	쓰篾
6 1743406699338	1743406709698	10485779	1	NULL	NULL	NULL	이 메시지는 삭제되었습니다.
7 1743407023074	1743407036407	10485780	1	NULL	NULL	NULL	i'm spiderman

향후 연구 계획

보다 자세한 아티팩트 확인 → 표 형식으로 정리

androidx.work.workdb

The screenshot shows the DB Browser for SQLite interface with the database file 'C:\Users\user\Desktop\Study\code\v25_1_digital_forensic\Session\network.loki.messenger\no_backup\androidx.work.workdb' open. The left pane displays the table and index structures:

- 테이블 (9)**
 - Dependency
 - Preference
 - SystemInfo
 - WorkName
 - WorkProgress
 - WorkSpec
 - WorkTag
 - android_metadata
 - room_master_table
- 인덱스 (6)**
 - index_Dependency_prerequisite_id
 - index_Dependency_work_spec_id
 - index_WorkName_work_spec_id
 - index_WorkSpec_period_start_time
 - index_WorkSpec_schedule_requested_at
 - index_WorkTag_work_spec_id
- 트리거 (0)**

The right pane shows the SQL code for creating these tables and indices.

```
CREATE TABLE 'Dependency' ('work_spec_id' TEXT NOT NULL, 'prerequisite_id' TEXT NOT NULL, PRIMARY KEY('work_spec_id', 'prerequisite_id'), FOREIGN KEY('work_spec_id') REFERENCES 'WorkSpec'(id))  
CREATE TABLE 'Preference' ('key' TEXT NOT NULL, 'long_value' INTEGER, PRIMARY KEY(key))  
CREATE TABLE 'SystemInfo' ('work_spec_id' TEXT NOT NULL, 'system_id' INTEGER NOT NULL, PRIMARY KEY('work_spec_id'), FOREIGN KEY('work_spec_id') REFERENCES 'WorkSpec'(id))  
CREATE TABLE 'WorkName' ('name' TEXT NOT NULL, 'work_spec_id' TEXT NOT NULL, PRIMARY KEY(name, 'work_spec_id'), FOREIGN KEY('work_spec_id') REFERENCES 'WorkSpec'(id))  
CREATE TABLE 'WorkProgress' ('work_spec_id' TEXT NOT NULL, 'progress' BLOB NOT NULL, PRIMARY KEY('work_spec_id'), FOREIGN KEY('work_spec_id') REFERENCES 'WorkSpec'(id))  
CREATE TABLE 'WorkSpec' ('id' TEXT NOT NULL, 'start' INTEGER NOT NULL, 'worker_class_name' TEXT NOT NULL, 'input_merger_class_name' TEXT, 'input' BLOB NOT NULL, 'output' BLOB)  
CREATE TABLE 'WorkTag' ('tag' TEXT NOT NULL, 'work_spec_id' TEXT NOT NULL, PRIMARY KEY(tag, 'work_spec_id'), FOREIGN KEY('work_spec_id') REFERENCES 'WorkSpec'(id) ON UPDATE CASCADE)  
CREATE TABLE android_metadata (locale TEXT)  
CREATE TABLE room_master_table (id INTEGER PRIMARY KEY, identity_hash TEXT)
```

CREATE INDEX 'index_Dependency_prerequisite_id' ON 'Dependency' ('prerequisite_id')
CREATE INDEX 'index_Dependency_work_spec_id' ON 'Dependency' ('work_spec_id')
CREATE INDEX 'index_WorkName_work_spec_id' ON 'WorkName' ('work_spec_id')
CREATE INDEX 'index_WorkSpec_period_start_time' ON 'WorkSpec' ('period_start_time')
CREATE INDEX 'index_WorkSpec_schedule_requested_at' ON 'WorkSpec' ('schedule_requested_at')
CREATE INDEX 'index_WorkTag_work_spec_id' ON 'WorkTag' ('work_spec_id')

The screenshot shows the DB Browser for SQLite interface with the database file 'C:\Users\alice\Desktop\Study\3-1\디지털포렌식개론\과제\network.loki.messenger\databases\signal_v4.db' open. The left pane displays the table structures:

- 테이블 (66)**
 - backup_files
 - blinded_id_mapping
 - closed_group_encryption_key_pairs...
 - closed_group_public_keys_table
 - configs_table
 - drafts
 - emoji_search
 - emoji_search_config
 - emoji_search_content
 - emoji_search_data
 - emoji_search_docsizes
 - emoji_search_idx
 - expiration_configuration
 - fork_info
 - group_member
 - group_receipts
 - groups
 - last_legacy_messages
 - loki_api_group_chat_auth_token_dat...
 - loki_api_last_deletion_server_id_cac...
 - loki_api_last_message_server_id_cac...
 - loki_api_swarm_cache
 - loki_error_message_database
 - loki_group_invites
 - loki_message_friend_request_database
 - loki_message_hash_database

The right pane shows the SQL code for creating these tables.

```
CREATE TABLE backup_files  
CREATE TABLE blinded_id_mapping  
CREATE TABLE closed_group_encryption_key_pairs...  
CREATE TABLE closed_group_public_keys_table  
CREATE TABLE configs_table  
CREATE TABLE drafts  
CREATE VIRTUAL TABLE emoji_search  
CREATE TABLE emoji_search_config  
CREATE TABLE emoji_search_content  
CREATE TABLE emoji_search_data  
CREATE TABLE emoji_search_docsizes  
CREATE TABLE emoji_search_idx  
CREATE TABLE expiration_configuration  
CREATE TABLE fork_info  
CREATE TABLE group_member  
CREATE TABLE group_receipts  
CREATE TABLE groups  
CREATE TABLE last_legacy_messages  
CREATE TABLE loki_api_group_chat_auth_token_dat...  
CREATE TABLE loki_api_last_deletion_server_id_cac...  
CREATE TABLE loki_api_last_message_server_id_cac...  
CREATE TABLE loki_api_swarm_cache  
CREATE TABLE loki_error_message_database  
CREATE TABLE loki_group_invites  
CREATE TABLE loki_message_friend_request_database  
CREATE TABLE loki_message_hash_database
```

Q&A

감사합니다

출처: 보안 인스턴트 메신저 Session에 대한 데이터베이스 복호화 방안 연구
<https://developer.android.com/privacy-and-security/keystore?hl=ko>