

비밀번호 관리 어플리케이션의 주요 데이터 복호화 연구 및 보안성 평가 (2024)



융합보안학과 윤세영

유튜브 주소: <https://youtu.be/q6wGrGcFval>

비밀번호 관리 어플리케이션의 주요 데이터 복호화 연구 및 보안성 평가*

김 한 결,^{1*} 이 신 영,¹ 박 명 서^{2*}
¹강남대학교 (학생), ²한성대학교 (교수)

A Study on Key Data Decryption and Security Evaluation for Password
Management Apps*

Han-gyeol Kim,^{1*} Sinyoung Lee,¹ Myungseo Park^{2*}
¹Kangnam University (Undergraduate student), ²Hansung University (Professor)

요 약

인터넷 서비스 급증과 함께 사용자가 다양한 서비스를 이용하게 되면서 계정관리에 어려움을 겪을 수 있다. 이러한 고충을 해결하기 위해 다양한 비밀번호 관리 어플리케이션이 등장하고 있다. 포렌식 관점에서 비밀번호 관리 어플리케이션은 범죄 증거를 획득할 수 있는 단서를 제공할 수 있다. 본 논문에서는 비밀번호 관리 어플리케이션에서 사용자가 저장한 데이터를 획득하는 것을 목적으로 한다. 이를 위해 역공학을 통해 암호화된 데이터를 복호화하고 분석 대상 어플리케이션에 대한 보안성 평가 및 데이터를 안전하게 보관할 수 있는 더 나은 방법을 제안한다.

분석 환경 및 대상 식별

Table 1. Target Application

Name	Version	Package Name
Password Cloud	v.9.3	password.cloud
Password ManageMent	v.1.8.2	net.miyam.dontforgetpassword
Hide Pass	v.1.9.2	com.sisomobile.android.passwordsafe
BeEasy	v.4.6.3	beeeay.a90ms.com.beeasy

HidePass
안전한 비밀번호 저장 앱

비밀번호

로그인

- 데이터를 서버에 저장하지 않고 사용자의 휴대폰에 저장하기 때문에 서버 공격으로 부터 안전합니다.
- 모든 데이터는 보안에 강력한 256비트 암호화 방식으로 저장됩니다.

분석 환경 및 대상 식별

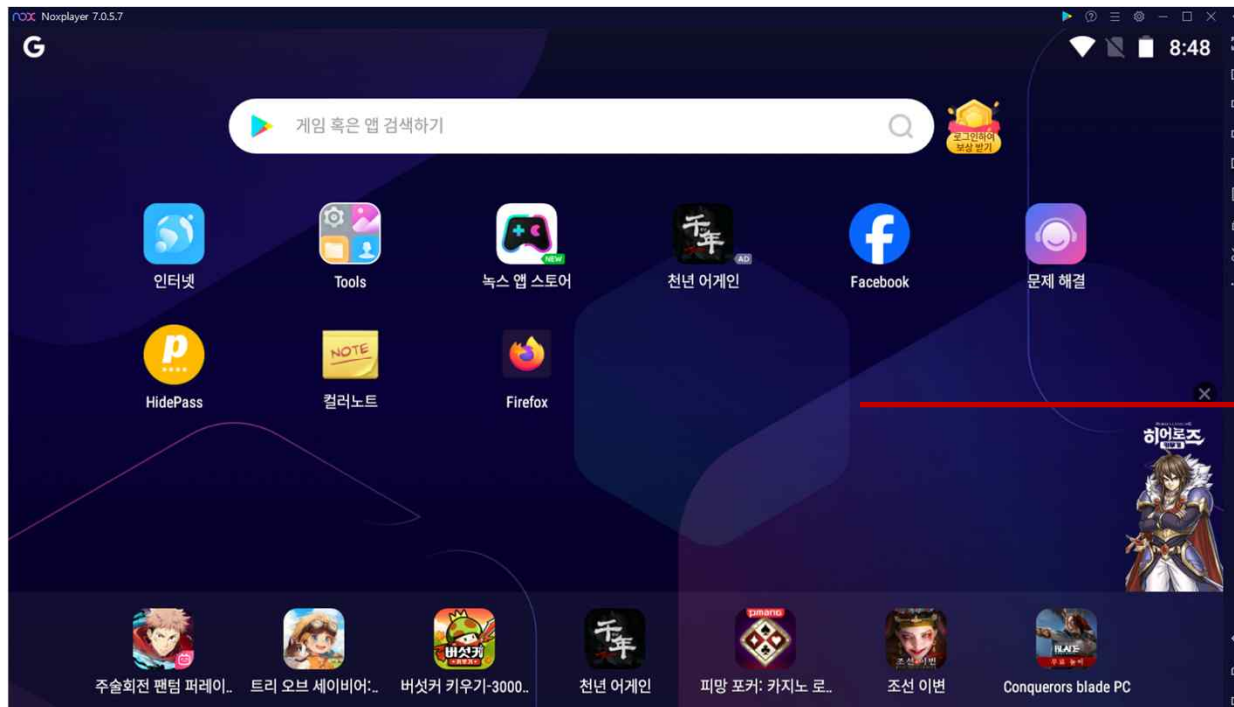
Table 2. Analysis devices and tools

Device and Software	Name	Version
Desktop	AMD Ryzen 3 3300X 4-Core Processor 3.79 GHz, 16.00GB	windows 10
Mobile Device	Pixel 4a	Android 11
Debugging Tool	Android Debugger Bridge	33.0.2
DB Viewer	DB Browser for SQLite	3.29
Raw Data Viewer	HxD	2.5.0.0
Decompiler	Jadx	1.4.3

분석 환경 및 대상 식별

Hide Pass	Master Password	shared_prefs/com .sisomobile.andr oid.passwordsafe _preferences.xml	value named password
	User Data	databases/my_pa ssword.db	id, passwordcol umn in safe table

실습 – 데이터베이스 접근



모바일 앱 실행
에뮬레이터

실습 – 데이터베이스 접근

```
Developer Command Prompt
*****
** Visual Studio 2022 Developer Command Prompt v17.10.4
** Copyright (c) 2022 Microsoft Corporation
*****

C:\Program Files\Microsoft Visual Studio\2022\Community>adb connect 127.0.0.1:62001
adb server version (36) doesn't match this client (41); killing...
* daemon started successfully
connected to 127.0.0.1:62001

C:\Program Files\Microsoft Visual Studio\2022\Community>adb shell
dream2qltechn:/ # whoami
root
```

Android 디버그 브리지(adb)

Android 디버그 브리지(adb)는 기기와 통신할 수 있도록 지원하는 다목적 명령줄 도구입니다. adb 명령어는 앱 설치 및 디버깅과 같은 다양한 기기 작업을 용이하게 합니다. adb는 기기에서 다양한 명령어를 실행하는 데 사용할 수 있는 Unix 셸에 대한 액세스를 제공합니다. 이 도구는 다음과 같은 세 가지 구성요소를 포함하는 클라이언트-서버 프로그램입니다.

- 명령어를 전송하는 **클라이언트**. 클라이언트는 개발 머신에서 실행됩니다. adb 명령어를 실행하여 명령줄 터미널에서 클라이언트를 호출할 수 있습니다.
- 기기에서 명령어를 실행하는 **데몬(adb)**. 데몬은 각 기기에서 백그라운드 프로세스로 실행됩니다.
- 클라이언트와 데몬 간의 통신을 관리하는 **서버**. 서버는 개발 머신에서 백그라운드 프로세스로 실행됩니다.

실습 - 데이터베이스 접근

```
dream2qltechn:/ # ls
acct      data      init      init.usb.rc  property_contexts  sepolicy      ueventd.rc
bugreports default.prop  init.envIRON.rc  init.zygote32.rc  root              service_contexts  vendor
cache     dev        init.qcom.rc     lib              sbin              storage
charger   etc        init.rc          mnt              sdcard            sys
config    file_contexts.bin  init.superuser.rc  oem              seapp_contexts    system
d         fstab.qcom  init.usb.configfs.rc  proc            selinux_version   ueventd.qcom.rc

dream2qltechn:/ # cd data
dream2qltechn:/data # ls
adb        app-ephemeral  bootchart  drm        mediadrM    ota          security     system_de
anr        app-lib        cache       local      misc        ota_package  ss           tombstones
app        app-private    dalvik-cache  lost+found  misc_ce     property     system       user
app-asec   backup        data        media      misc_de     resource-cache  system_ce    user_de
dream2qltechn:/data # cd data
```


실습 – 데이터베이스 접근

```
dream2qltechn:/data/data # ls
android
android.ext.services
android.ext.shared
com.amaze.filemanager
com.android.Calendar
com.android.backupconfirm
com.android.bluetooth
com.android.bluetoothmidiservice
com.android.bookmarkprovider
com.android.browser
com.android.calculator2
com.android.calllogbackup
com.android.camera2
com.android.captiveportallogin
com.android.carrierconfig
com.android.certinstaller
com.android.contacts
com.android.cts.ctsshim
com.android.cts.priv.ctsshim
com.android.defcontainer
com.android.documentsui
com.android.dreams.basic
com.android.phone
com.android.printservice.recommendation
com.android.printspooler
com.android.providers.blockednumber
com.android.providers.calendar
com.android.providers.contacts
com.android.providers.downloads
com.android.providers.media
com.android.providers.settings
com.android.providers.telephony
com.android.providers.userdictionary
com.android.provision
com.android.proxyhandler
com.android.server.telecom
com.android.settings
com.android.sharedstoragebackup
com.android.shell
com.android.statementservice
com.android.storagemanager
com.android.systemui
com.android.vending
com.android.vpndialogs
```

실습 - 데이터베이스 접근



패키지 이름

com.sisomobile.android.passwordsafe

~ ~ ~ ~ ~

```
dream2qltechn:/data/data # cd com.sisomobile.android.passwordsafe/  
dream2qltechn:/data/data/com.sisomobile.android.passwordsafe # ls  
cache  code_cache  databases  shared_prefs
```

실습 - 데이터베이스 접근

```
dream2qltechn:/sdcard # ls
Alarms  Apps  Download  Music          Pictures  Ringtones  app_test2  app_test4  app_test6  new_pw  test_folder
Android DCIM  Movies  Notifications  Podcasts  app_test   app_test3  app_test5  data      new_test
dream2qltechn:/sdcard # mkdir bbang1110
```

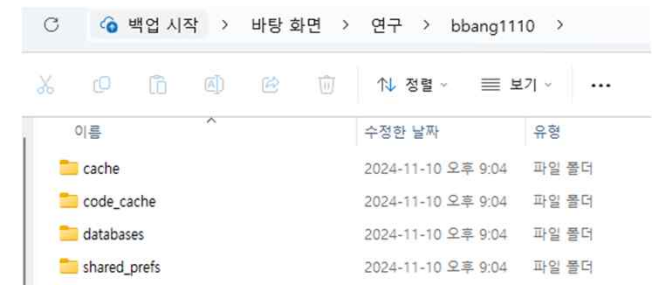
```
dream2qltechn:/data/data/com.sisomobile.android.passwordsafe # ls
cache code_cache databases shared_prefs
p -r /data/data/com.sisomobile.android.passwordsafe/* /sdcard/bbang1110
```

cp [복사할 파일 경로] [복사될 파일 경로]

```
dream2qltechn:/sdcard/bbang1110 # ls
cache code_cache databases shared_prefs
```

완성

```
C:\Users\HJ\Desktop\연구>adb pull /sdcard/bbang1110
/sdcard/bbang1110/: 3 files pulled, 0 skipped. 0.4 MB/s (25698 bytes in 0.058s)
```



이름	수정한 날짜	유형
cache	2024-11-10 오후 9:04	파일 폴더
code_cache	2024-11-10 오후 9:04	파일 폴더
databases	2024-11-10 오후 9:04	파일 폴더
shared_prefs	2024-11-10 오후 9:04	파일 폴더

실습 – 데이터베이스 확인

🔄 백업 시작 > 바탕 화면 > 연구 > bbang1110 > databases

✂️ 📄 📁 🗑️ 🔄 정렬 ▼ ≡ 보기 ▼ ...

이름	수정한 날짜	유형	크기
📄 my_password.db	2024-11-10 오후 9:04	Data Base File	16KB
📄 my_password.db-journal	2024-11-10 오후 9:04	DB-JOURNAL 파일	9KB

DB Browser for SQLite - C:\Users\HJ\Desktop\연구\bbang1110\databases\my_password.db

파일(F) 편집(E) 보기(V) 도구(T) 도움말(H)

새 데이터베이스(N) 데이터베이스 열기(O) 변경사항 저장하기(W) 변경사항 취소하기(R) 프로젝트 열기(P) 프로젝트 저장하기(V) 데이터베이스 연결(A) ❌ 데이터베이스 닫기(C)

데이터베이스 구조 데이터 보기 Pragma 수정 SQL 실행

테이블(T): safe 모든 열에서 필터링

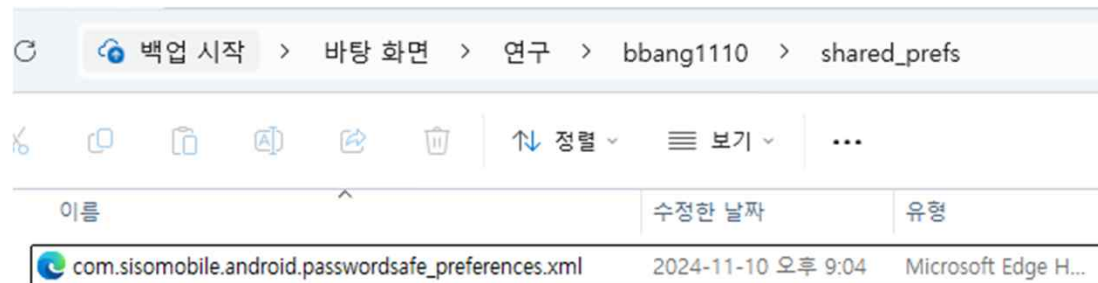
seq	group_name	title	id	password	url	note	memo_title_01
필터	필터	필터	필터	필터	필터	필터	필터
1	1 cBdV86Udj4KdRug9lz+lJg...	p/Ws/6BqX5Bi8B2hUv1DIQ==...	n6UGFa3N/dVJYnXFQE2nEQ...	NbKISjPma4bTrAwkcTLfIQ...	lpKvZtKvUpPOTJQvqQFG+g...		
2	2 cBdV86Udj4KdRug9lz+lJg...	T8EhKOOb095MMGbLyALCUTw...	1q1nbsXHaojenqXr+ooXgQ...	NbKISjPma4bTrAwkcTLfIQ...	VQ/Mp1fiYUvjltbTMwRTw=...		
3	3 cBdV86Udj4KdRug9lz+lJg...	enddbfOujwfeLH0AerhRDQ=...	VI/SK3nLZ5+RU1Hsz4/zJw=...	NbKISjPma4bTrAwkcTLfIQ...			

데이터베이스 셀 수정하기(C)

모드: 문자열

1 NbKISjPma4bTrAwkcTLfIQ==
2

실습 - 데이터베이스 확인



```
▼<map>
  <string name="trialExpireDate">20241011</string>
  <string name="loginStatus">N</string>
  <string name="password">8100a251377886be50d0aaa2eece5135728ae82374b81725881478ce44ab7be</string>
  <int name="autoLogoutTime" value="0"/>
  <boolean name="isPremium" value="false"/>
  <int name="inCount" value="1"/>
  <string name="passwordRegistStatus">Y</string>
  <string name="clipartStatus">Y</string>
  <string name="premiumExpireDate">20241109</string>
  <string name="passwordHideStatus">Y</string>
</map>
```

```
public class h {
    public static String a(String str) {
        try {
            MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
            messageDigest.update(str.getBytes());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

주요 데이터 획득 및 복호화

3.3.1 마스터 비밀번호 복호화

xml 파일의 password 영역에 암호화된 마스터 비밀번호가 존재한다. 해당 값은 SHA-256 해시값을 통해 만들어진 해시값이기 때문에 일반적인 방법으로는 복호화가 불가능하다. 이러한 경우 사전대입 공격 또는 전수조사 공격을 통해 원본값을 획득할 수 있다. 본 논문에서는 Table 2.에서 명시된 데스크탑 환경에서 전수조사를 진행하였으며, Table 4.와 같이 6자리까지 전수조사 공격을 통해 마스터 비밀번호 복구 시간을 측정하였다. 또한, 그 이상의 자릿수에 해당하는 마스터 비밀번호 복구 시간은 추정치로 작성하였다.

```
public class h {  
    public static String a(String str) {  
        try {  
            MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");  
            messageDigest.update(str.getBytes());
```

```
▼<map>  
  <string name="trialExpireDate">20241011</string>  
  <string name="loginStatus">N</string>  
  <string name="password">8100a251377886be50d0aaa2eece5135728ae82374b81725881478ce44ab7be</string>  
  <int name="autoLogoutTime" value="0"/>  
  <boolean name="isPremium" value="false"/>  
  <int name="inCount" value="1"/>  
  <string name="passwordRegistStatus">Y</string>  
  <string name="clipartStatus">Y</string>  
  <string name="premiumExpireDate">20241109</string>  
  <string name="passwordHideStatus">Y</string>  
</map>
```


주요 데이터 획득 및 복호화

사용자 관리 데이터는 데이터베이스의 safe 테이블에 사용자가 입력한 모든 데이터가 암호화되어 저장된다. 사용자가 입력한 값을 AES256-CBC-PKCS5Padding 암호화 알고리즘을 사용하여 암호화한 후 Base64로 인코딩한다. 이때 사용한 암호화 키와 IV는 xml 파일에 존재하는 마스터 패스워드와 소스 코드상에서 하드 코딩된 문자열을 가지고 어플리케이션에서 구현된 메서드를 통해 동적으로 생성한다. 정적 분석을 통해 Hide Pass 어플리케이션의 암호화 알고리즘을 식별할 수 있었으며, 해당 부분을 분석하여 획득한 16바이트의 IV와 32바이트의 암호화키는 다음과 같다. 해당 값은 마스터 패스워드가 12345일 때 생성된 값이다.

- IV: bb01112atKbeVbpG

- 암호화키:

bb01112atKbeVbpGQs130u/p5994471a

```
public static String a(String str, String str2) {
    try {
        byte[] decode = Base64.decode(str.getBytes(), 0);
        IvParameterSpec ivParameterSpec = new IvParameterSpec(str2.substring(0, 16).getBytes());
        SecretKeySpec secretKeySpec = new SecretKeySpec(str2.getBytes("UTF-8"), "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(2, secretKeySpec, ivParameterSpec);
        return new String(cipher.doFinal(decode), "UTF-8");
    } catch (Exception unused) {
        return "";
    }
}
```

```
public static String b(String str, String str2) {
    try {
        byte[] bytes = str.getBytes("UTF-8");
        IvParameterSpec ivParameterSpec = new IvParameterSpec(str2.substring(0, 16).getBytes());
        SecretKeySpec secretKeySpec = new SecretKeySpec(str2.getBytes("UTF-8"), "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(1, secretKeySpec, ivParameterSpec);
        return Base64.encodeToString(cipher.doFinal(bytes), 0);
    } catch (Exception unused) {
        return "";
    }
}
```

```
public static String a(Context context) {
    try {
        String a2 = y.a(context, "password", "");
        return a2.substring(8, 16) + h.b("sisomobile", a2.substring(0, 16)).substring(1, 17) + a2.substring(0, 8);
    } catch (Exception unused) {
        return "";
    }
}
```

