Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.34, NO.1, Feb. 2024 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2024.34.1.61

비밀번호 관리 어플리케이션의 주요 데이터 복호화 연구 및 보안성 평가*

김 한 결,[1†] 이 신 영,[1] 박 명 서[2‡]
[1]강남대학교 (학생), [2]한성대학교 (교수)

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

Han-gyeol Kim,[1†] Sinyoung Lee,[1] Myungseo Park[2‡]
[1]Kangnam University (Undergraduate student), [2]Hansung 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.

ABSTRACT

As users use various services along with the rapid increase in Internet services, it may be difficult to manage accounts. To solve these difficulties, various password management applications are emerging. From a forensic point of view, password management applications can provide clues to obtain criminal evidence. The purpose of this paper is to acquire the data stored by the user in the password management application. To this end, we propose a better way to decrypt the encrypted data through reverse engineering, evaluate the security of the application to be analyzed, and safely store the data.

Keywords: Android, Forensic, Reversing, Master Password, Encryption/Decryption

I. 서 론

1.1 배경

비밀번호 관리 어플리케이션은 사용자의 계정 인증이 필요한 서비스 또는 장치에 대해 계정 정보를 저장하는 기능을 가진 어플리케이션이다. 오늘날 인터넷 서비스가 급증하고 다양한 스마트 디바이스가 등장하면서 사용자는 모든 계정 정보를 기억하기 어려워졌다. 또한, 다양한 보안 사고로 인하여 서비스에서 사용자에게 보안성 높은 패스워드 설정을 요구하고 있다. 이러한 상황 속에서 자신의 인증 정보를 안전하게 기록하고 또 자체적으로 복잡한 패스워드를 생성하여 사용자가 기억하지 않아도 자동 입력과 같은 유용한 기능을 제공하는 어플리케이션은 사용자들에게 부담을 줄여준다.

비밀번호 관리 어플리케이션들의 공통된 특징은 어플리케이션 실행 시 잠금 해제를 위해 필요한 마스터 패스워드를 갖는다는 것이다. 마스터 패스워드 분실 시 복구 기능을 지원하는 어플리케이션은 복구 이

Received(10. 31. 2023), Modified(1st: 12. 22. 2023, 2nd: 01. 16. 2024), Accepted(01. 16. 2024)
* 본 연구는 한성대학교 학술연구비 지원과제임
† 주저자, biequal@kangnam.ac.kr
‡ 교신저자, pms91@hansung.ac.kr(Corresponding author)

메일 또는 보안 질문을 통해 복구할 수 있다. 클라우드 백업 기능을 제공하는 어플리케이션도 존재하지만, 기본적으로 마스터 패스워드와 저장된 데이터는 사용자의 디바이스 내에 암호화된 상태로 보관된다.

인증과 관련된 정보를 저장하는 것이 목적인 만큼 비밀번호 관리 어플리케이션은 사용자에게 민감한 정보를 가지고 있다. 따라서 포렌식 관점에서 어플리케이션을 분석하면 범죄자의 중요한 데이터를 획득할 수 있는 단서를 제공해준다.

본 논문에서는 비밀번호 및 계정 관련 데이터를 보관하는 안드로이드 어플리케이션 4종을 대상으로 분석한 결과에 대해 설명한다. 특히, 사용자가 설정한 마스터 패스워드를 복구하고, 암호화된 사용자 관리 데이터를 복호화하는 것을 목적으로 한다. 어플리케이션이 데이터를 보관하는 과정에서 취약한 부분을 식별하였으며, 비밀번호 관리 어플리케이션의 특성을 고려하여 데이터에 접근하기 위해 필요한 3가지 절차를 분류하여 각 어플리케이션별 어떤 절차에서 데이터 획득이 가능한지를 분석하였다.

본 논문의 구성은 다음과 같다. 2장에서는 어플리케이션 분석 환경과 도구, 분석 대상에 대해 서술하고, 3장에서는 어플리케이션별 식별된 암호화 데이터의 복호화 방법을 서술하였다. 4장에서는 어플리케이션별 취약한 보안 요소를 확인하고 이를 보완할 수 있는 방법을 제시하였다. 마지막으로 5장에서는 결론으로 마무리 하였다.

1.2 관련 연구

패스워드 매니저에 대해 데이터 획득을 위한 다양한 보안 취약점 분석 및 안전한 설계, 분석 프로세스에 대한 연구가 진행되었다. 남기훈 등 5명은 안드로이드 패스워드 매니저 어플리케이션에 대한 취약점 분석과 이에 대한 대응 방안에 관해서 설명하였다[1]. 정혜라 등 2명은 로컬 컴퓨터에 저장된 패스워드 저장소의 보안 취약점을 분석하는 절차와 시나리오 제시 및 공격 실험을 통한 보안 취약점을 확인하였다[2]. 박준성 등 2명은 크롬 브라우저의 패스워드 매니저에 대해 복호화 요소 파악 및 도구 제안, 디지털 포렌식 원칙을 준수하며 정보를 획득할 방안을 제시하였다[3]. Yasasw. Kumarakalva 등 5명은 패스워드 매니저의 필요성에 따라 직접 패스워드 매니저를 개발하고 이에 대한 안전한 설계 방법을 상세하게 설명하였다[4]. Chaitanya Rahalkar등 2명은 패스워드 매니저의 특징을 분석하고 안전한 설계 방법을 제시하였다[5].

본 논문에서는 사용자 비밀번호를 관리하는 안드로이드 어플리케이션에서 데이터 획득과 암호화 알고리즘별 복호화 방법, 대상 어플리케이션의 취약 요소 및 보완 방안에 관해 설명하였다.

II. 분석 환경 및 대상 식별

2.1 분석 환경

Table 1.은 분석 대상 비밀번호 관리 어플리케이션을 나타낸다. 분석 대상 어플리케이션은 데이터 복호화 방법이 공개되지 않았으며 취약한 데이터 저장 방식을 사용하는 어플리케이션으로 선정하였다. 선정된 어플리케이션을 통해 비밀번호 관리 어플리케이션에서의 데이터 획득 방법 및 보안성 향상을 위한 방법을 제시하였다.

모든 어플리케이션은 논문 작성일 기준으로 가장 최신 버전을 사용했고 구글 플레이스토어에 배포된 어플리케이션을 대상으로 한다.

Table 2.는 비밀번호 관리 어플리케이션 분석을 위해 사용한 도구와 장치이다. 기본적으로 모든 실험 및 분석을 윈도우10 데스크탑에서 진행하고, 분석 대상이 되는 어플리케이션 데이터를 획득하기 위해 루팅 된 단말기를 이용하였다. 어플리케이션의 APK (Android Application Package) 파일과 앱 데이터 추출은 ADB (Android Debug Bridge)의 명령인 pull을 이용하였다. 추출된 데이터 중 데이터베이스 파일은 DB Browser for SQLite를 사용하여 내용을 확인하였고, 어플리케이션 정적 분석을 위해 Jadx를 사용하였다. 또한, 획득한 이미지의 바이너리 값을 통해 원본 파일과 비교하기 위해 HxD를 사용하였다. 각 대상 어플리케이션의 APK

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

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

파일을 대상으로 정적 분석을 진행하였으며 각 어플리케이션에서 사용한 암호화 알고리즘에 대응할 수 있는 복호화 알고리즘을 구현하고, 실험을 통해 정상 작동하는 것을 확인하였다.

2.2 분석 대상 식별

본 논문은 주요 데이터를 마스터 패스워드와 사용자 관리 데이터로 정의한다. 여기서 사용자 관리 데이터란 사용자가 생성 및 저장한 데이터를 의미한다.

Table 3.은 분석 대상 어플리케이션들의 주요 데이터와 획득경로를 나타낸 것이다. xml 파일과 데이터베이스 파일에서 마스터 패스워드 및 사용자 관리 데이터를 획득할 수 있었다.

Table 3. Target File

Application	Main Data	Path	Value
Password Cloud	Master Password	shared_prefs/password.cloud_preferences.xml	value named Password Cloud1
	Encryption Key		value named Password Cloud
	User Data	databases/Temp.db	campo,dato_campo column in campi table
Password Management	Master Password	databases/donotforgetpassword	masterKey column in masterKey table
	User Data		id, passwordcolumn in model table
Hide Pass	Master Password	shared_prefs/com.sisomobile.android.passwordsafe_preferences.xml	value named password
	User Data	databases/my_password.db	id, passwordcolumn in safe table
BeEasy	Master Password	shared_prefs/beeeasy.a90ms.com.beeasy_preferences.xml	value named passcode
	User Data	databases/write.db	userid, userpw column in WriteTable table

안드로이드 어플리케이션에서 데이터를 저장하는 주요 방법은 Shared Preferences를 이용하는 방법과 데이터베이스를 활용하는 방법으로 나뉜다. 본 논문에서는 대상 어플리케이션에서 두 가지 위치에 대해 조사하여 주요 데이터를 획득한다. 이후 획득한 데이터에 대한 복호화 과정은 3장에서 설명한다. 주요 데이터가 위치하는 장소와 각 장소에서 획득한 데이터를 암호화 방식에 따라 복호화하여 원본값을 획득하는 과정을 자세히 설명하였다.

III. 주요 데이터 획득 및 복호화

포렌식 관점에서 최종 목표인 사용자 관리 데이터를 획득하기 위해 Fig. 1.와 같이 총 세 가지 절차로 분석을 수행한다. 첫 번째 절차(case 1)는 마스터 패스워드를 획득한 경우로 바로 어플리케이션 잠금을 해제하여 목표 데이터를 획득할 수 있다. 하지만, 마스터 패스워드를 획득하지 못한 경우 사용자 관리 데이터에 접근할 수 없다. 두 번째 절차(case 2)는 대상 기기에서 어플리케이션 데이터를 추출하여 사용자 관리 데이터를 획득하는 것이다. 평문 상

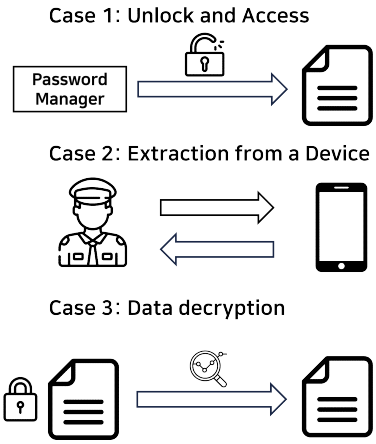

Fig. 1. Three Cases for Data Acquisition

태로 보관된 사용자 관리 데이터를 바로 획득할 수 있으나, 암호화되어 있다면 해당 데이터는 활용할 수 없다. 마지막 세 번째 절차(case 3)에서는 암호화된 마스터 패스워드 또는 사용자 관리 데이터를 복호화하는 것이다. 이를 위해 대상 어플리케이션을 역공학 분석하여 밝힌 데이터의 암호화 논리를 적용하여 해당 데이터를 복호화할 수 있다.

3.1 Password Cloud

Password Cloud 어플리케이션은 이메일, 신용카드 정보, 웹 사이트 계정 등과 같은 고정된 양식뿐만 아니라 커스텀 양식을 제공하여 사용자가 원하는 데이터를 다양한 형태로 저장할 수 있는 기능을 제공하고 있다.

어플리케이션의 마스터 패스워드는 암호화되어 관리되지만, 모든 사용자 관리 데이터가 Temp.db 데이터베이스 파일의 campi 테이블에 평문으로 저장된다. 따라서 앞서 설명한 (case 2)에 해당하는 절차로 사용자 관리 데이터 획득이 가능하나, 다양한 비밀번호 어플리케이션의 마스터 패스워드 관리 방법을 도출하기 위해 암호화 알고리즘 분석을 통해 마스터 패스워드를 복호화하는 과정을 설명한다.

3.1.1 마스터 패스워드 복호화

마스터 패스워드는 32바이트의 암호화키를 사용해 AES256-ECB-PKCS5Padding을 통해 암호화된 후 16진수 값을 16진수 문자열로 변환하는 HEX 인코딩 과정을 거쳐 password.cloud_preferences.xml

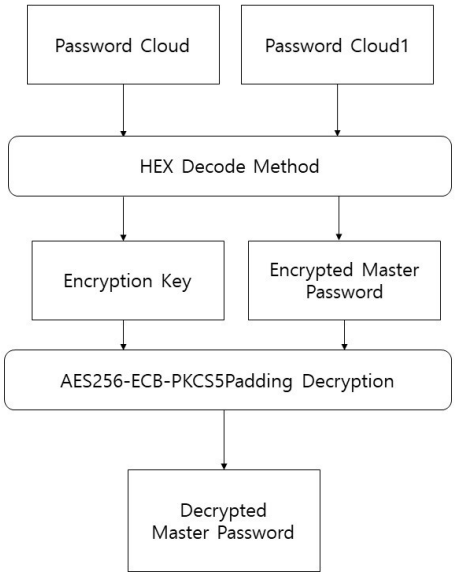

Fig. 2. Decrypt the master password of Password Cloud

파일의 PasswordCloud1 영역에 저장된다. 암호화키는 HEX 인코딩된 후 PasswordCloud 영역에 저장된다. 따라서 복호화를 위해선 우선 xml파일에 있는 마스터 패스워드와 암호화키를 모두 16진수 문자열을 16진수 값으로 변환하는 HEX 디코딩을 적용해야 한다. 이렇게 획득한 암호화 키와 암호화된 마스터 패스워드를 이용해 원본 마스터 패스워드를 획득하기 위해 AES256-ECB-PKCS5Padding을 통한 복호화를 진행한다.

Fig. 2.은 암호화된 마스터 패스워드의 복호화 과정을 도식화한 것이다.

3.2 Password Management

Password Management 어플리케이션은 웹 사이트 계정과 은행 계좌 및 카드 정보를 관리할 수 있는 어플리케이션이다. 마스터 패스워드와 사용자 관리 데이터가 모두 암호화되어 donotforgetpassword 데이터베이스 파일에 저장된다.

사용자 관리 데이터의 경우 사용자가 입력한 값 중 비밀번호만 암호화되었고 나머지 값은 평문으로 저장된다. 따라서, 해당 어플리케이션의 복호화 대상은 마스터 패스워드와 사용자 관리 데이터 중 비밀번호이다.

3.2.1 마스터 패스워드 복호화

마스터 패스워드는 16바이트의 암호화키를 사용하여 암호화된다. 입력받은 마스터 패스워드를 AES128-CBC-PKCS5Padding을 사용하여 암호화한 후 Base64로 인코딩 한다. 여기서 암호화 키와 IV (Initialization Vector)는 하드 코딩된 값을 이용하여 생성되기 때문에 사용환경에 상관없이 동일한 값을 갖는다. 복호화 단계에서는 우선 고정된 암호화키와 IV을 획득해야 한다, 획득을 위해서 APK 파일을 Jadx를 통한 정적 분석으로 어플리케이션의 암호화 알고리즘을 식별하였다. 이와 같은 역공학 분석을 통해 다음과 같이 두 값이 모두 동일함을 밝혀냈다.

- IV: aes256-miya-key!
- 암호화키: aes256-miya-key!

이후 Fig. 3.와 같이 암호화 단계와 동일한 암호화키와 IV를 사용하여 AES128-CBC-PKCS5Padding 알고리즘과 Base64 디코딩을 이용해 복호화를 진행한다.

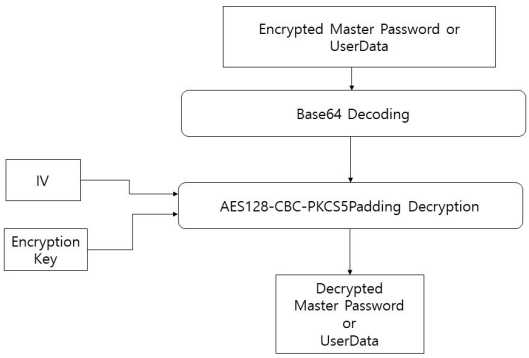

Fig. 3. Decrypt master password or user data for Password management

3.2.2 관리 데이터 복호화

관리 데이터에서 비밀번호는 마스터 패스워드의 암/복호화 단계와 동일한 조건으로 암호화 및 복호화를 진행한다. 따라서 Fig. 3.와 같이 이전과 동일한 암호화키와 IV를 이용하여 Base64 디코딩 및 AES128-CBC-PKCS5Padding을 이용하여 복호화를 진행한다.

3.3 Hide Pass

Hide Pass 어플리케이션은 웹 사이트의 계정 정보만을 기록할 수 있는 어플리케이션이다.

마스터 패스워드는 com.sisomobile.android.passwordsafe_preferences.xml 파일에 암호화된 상태로 저장된다. 또한, 사용자 관리 데이터는 my_password.db 데이터베이스 파일의 safe 테이블에 암호화되어 저장된다. 마스터 패스워드는 영문 대/소문자, 특수문자, 숫자, 공백을 모두 허용하며 최소 1자리, 최대 16자리로 구성된다.

3.3.1 마스터 패스워드 복호화

xml 파일의 password 영역에 암호화된 마스터 패스워드가 존재한다. 해당 값은 SHA-256 해시함수를 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 패스워드 복구 시간을 측정하였다. 또한, 그 이상의 자릿수에 해당하는 마스터 패스워드 복구 시간은 추정치로 작성하였다.

안드로이드 스마트폰에서 입력 가능한 영문 대문자 및 소문자, 숫자, 특수문자를 고려했을 때 한 자리에 들어갈 수 있는 경우의 수는 대략적으로 110개이다. 따라서 x자리를 갖는 패스워드의 경우 110x의 경우의 수를 갖는 것을 확인하였다. 즉, 선형적인 사실에 의해 n번째는 n-1번째 경우의 수의 대략

Table 4. Indiscriminate attack by digit on Hide Pass's master password

Digits	Time(ms)
2	26
3	46
4	295
5	13,329
6	1,435,021
7 (estimate)	157,852,310
16 (estimate)	17,363,754,100

110배가 된다. 이러한 방식의 암호화에서 일정 자리 이상을 갖는 패스워드는 전수조사를 위해서 높은 컴퓨팅 파워가 요구된다.

3.3.2 관리 데이터 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 암호화키와 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

복호화 과정은 암호화된 사용자 데이터를 Base64로 디코딩한 후 Fig. 4.와 같이 진행한다. 사용자 관리 데이터는 암호화 단계와 동일한 조건으로 복호화한다. 여기서 하드 코딩된 문자열은 “sisomobile”이고, 암호화된 마스터 패스워드는 64

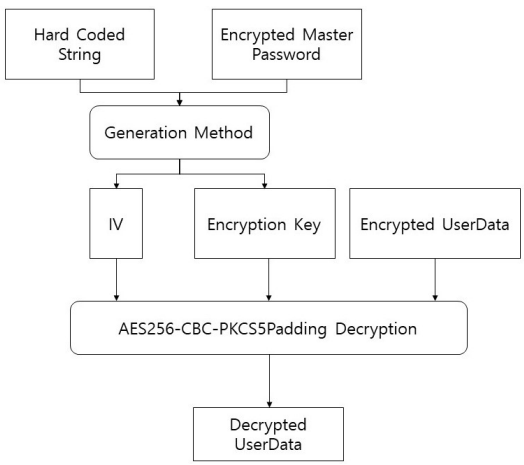

Fig. 4. Decrypting UserData in the Hide Pass

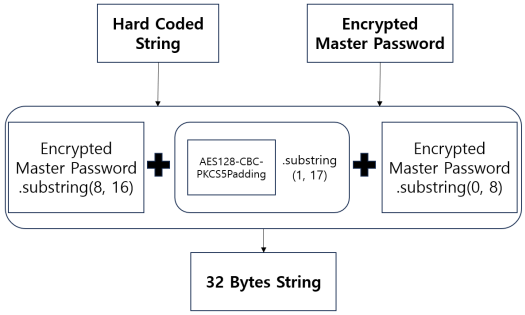

Fig. 5. Generation Method of the Hide Pass

바이트의 문자열이다.

Generation Method에서 암호화키와 IV를 생성하는 과정은 Fig. 5.와 같다. 앞뒤로는 암호화된 마스터 패스워드를 8바이트씩 자른 값이 위치하고 가운데에는 AES128 암호화를 통해 하드 코딩된 값을 암호화한 후 16바이트로 자른 값이 위치한다. AES128 암호화 과정에서 IV는 암호화된 마스터 패스워드를 앞에서부터 8바이트 크기로 자른 값으로 생성하고, 암호화키는 앞에서부터 16바이트로 자른 값이 된다. 최종적으로 생성된 32바이트 문자열에서 해당 값을 앞에서부터 16바이트 자른 값은 IV가 되고, 32바이트 값 자체는 암호화키가 된다.

3.4 BeEasy

BeEasy 어플리케이션은 웹 사이트, 게임 계정 정보와 같은 텍스트 데이터뿐만 아니라 이미지 데이터도 저장이 가능하다. 해당 어플리케이션에서는 마스터 패스워드를 고정된 길이인 6자리의 숫자로 설정하도록 되어있다.

마스터 패스워드는 beeeay.a90ms.com.beeasy_preferences.xml 파일에 암호화되어 저장된다. 또한, 사용자 관리 데이터 및 이미지 데이터는 write.db 데이터베이스 파일에 저장된다.

3.4.1 마스터 패스워드 복호화

Jadx를 통한 정적 분석을 통해 BeEasy 어플리케이션에서의 SHA-1을 통한 암호화 알고리즘을 분석하였다. 이를 통해 xml 파일에 저장된 마스터 패스워드는 사용자가 입력한 원본 마스터 패스워드에 하드코딩된 문자열인 “7xn7@c$”를 앞뒤로 추가하여 SHA-1을 통해 생성된 해시값이라는 것을 식별할

Table 5. Decryption time according to BeEasy's password

Password	Time(ms)
123456	408
598721	1,051
999999	1,635

수 있었다. 이러한 해시값은 전수조사 공격을 통해 원본값을 획득할 수 있는데 이때 평문 데이터에 임의의 문자열이 추가되었다는 점에 유의하여야 한다.

Table 5.는 숫자 6자리로 만들어질 수 있는 임의의 마스터 패스워드에 대해 전수조사 공격을 통해 원본값을 획득하는데 걸린 시간이다. 해당 전수조사 공격은 Table 2.에서 명시된 데스크탑 환경에서 진행하였다. 본 논문에서 시도한 무차별 대입 공격은 000000부터 시작하기 때문에 최악의 경우는 999999가 된다. 이때 걸린 시간은 1,635ms이므로 숫자 6자리로 구성된 패스워드는 늦어도 1.7초 이전에 원본값을 획득할 수 있다.

3.4.2 관리 데이터 복호화

사용자 관리 데이터는 텍스트 양식의 데이터인 아이디, 패스워드, 메모 기록이 암호화되어 있고 모두 동일한 알고리즘인 AES128-CBC-PKCS5Padding이 사용되었으며, 암호화 키와 IV는 값이 동일하다. 이때 해당 값은 칼럼 데이터별로 모두 다른 값을 사용한다. 관리 데이터 칼럼별 암호화 키와 IV는 Table 6.의 값을 통해 생성한다. 해당 값은 10바이트 크기의 문자열이며 나머지 6바이트를 null값으로 채운 후 IV와 암호화 키로 사용한다. 마지막으로 Base64를 통해 인코딩된 값이 최종적으로 암호화된 관리 데이터를 의미한다.

복호화 과정은 암호화 과정을 반대로 진행한다. Base64로 디코딩된 데이터를 복호화 알고리즘을 이용하여 복호화하면 원본값을 획득할 수 있다.

이미지 데이터의 경우 원본 데이터가 그대로 저장되어있다. DB Browser for SQLite 도구를 통해 저장된 이미지의 바이너리 값을 확인한 후 HxD를 통해 해당 바이너리 값으로 동일한 확장자의 파일을 생성하여 원본 데이터와 동일함을 확인하였다.

Table 6. String by column of the BeEasy

Column	Encryption Key and IV
userid	writeData1
userpw	writeData2
memo	writeData3

IV. 어플리케이션 보안성 평가

민감한 정보를 저장하는 어플리케이션은 자체적으로 암호화 솔루션을 갖춰 내부 데이터를 보호하고 있다. 하지만 암호화의 핵심이 되는 암호화키의 노출과 구조적인 결함으로 내부 데이터가 유출될 수 있다.

본 장에서는 분석 대상 어플리케이션의 주요 취약 요소를 소개하고 해당 요소에 대해 조금 더 안전한 설계 방법을 제시하였다. Table 7.은 주요 취약 요소를 식별하기 위해 고려한 데이터 암호화 여부 및 암호 알고리즘 설계 방식에 대한 내용이다. 사용자 입력 데이터, 기타 민감한 데이터 및 암호 알고리즘 사용 시 사용되는 주요 매개변수에 대한 암호화 여부를 확인하였다. 또한, 암호 알고리즘 설계 시 하드코딩된 값 사용 여부, 동일한 매개변수 반복사용 여부, 암호화 키 보관 시 암호화 여부 및 어플리케이션의 취약한 비밀번호 규칙을 고려하였다. 이 과정에서 어플리케이션의 취약한 요소는 OWASP (The Open Web Application Security Project) Mobile Top 10 항목에서 해당 여부를 확인하였다[6].

Table 7. Vulnerable element identification items

Category	Consideration
Encryption Status	- Encryption of User-Entered or Sensitive Information - Parameter Encryption for Data Protection
Cryptographic Algorithm Design Method	- Use of Hard coded Values - Use of Identical Parameters - Encryption Status of Encryption Keys - Application of Weak Password Rules

4.1 Password Cloud

Password Cloud 어플리케이션의 마스터 패스워드는 최소 4자리이며 영문 대/소문자, 숫자, 특수문자를 모두 허용한다. 또한, 최대 잠금 해제 시도 실패 시 모든 모듈을 삭제하는 자동 리셋 기능도 제공한다. 그러나 이러한 보안성에도 불구하고 해당 어

플리케이션에는 큰 보안 결함이 존재한다. 바로 사용자 관리 데이터를 암호화하지 않는다는 것이다. OWASP Mobile Top 10에 의하면 암호화하지 않고 데이터를 보관하는 방식은 M9: Insecure Data Storage에 해당한다.

일반적으로 어플리케이션을 사용할 때 잠금을 해제할 수 있는 마스터 패스워드는 AES256 암호화와 HEX 인코딩이 적용되어있다. 이는 마스터 패스워드에 한정하여 3장에서 설명하였던 (case1), (case2), (case3)에 대한 대응책이 갖춰졌음을 의미한다. 하지만 사용자 관련 데이터의 경우 직접 추출하는 (case2)에 대해서는 무방비 상태로 노출된다.

해당 어플리케이션이 (case2)에 대응하기 위해선 사용자 관리 데이터를 암호화해야 한다. 여기서 적용되는 암호화는 대칭키 방식을 사용하고 키를 안전하게 보관하는 방법과 해시함수를 사용하고 동시에 보안성 높은 마스터 패스워드 설정을 요구하는 방법으로 나뉜다.

4.2 Password Management

Password Management 어플리케이션은 일반적인 계정뿐만 아니라 은행 계좌와 카드 정보를 저장할 수 있는 양식을 제공한다. 다만 사용자 관리 데이터에 대한 암호화는 패스워드에 한정되어 적용되어있다. 이는 이메일 또는 계정 ID, 은행 계좌나 카드 번호와 같은 정보가 암호화되지 않고 그대로 노출된다는 것을 의미한다. 이렇게 민감한 정보를 암호화하지 않는 것은 OWASP Mobile Top 10의 M9: Insecure Data Storage에 해당한다. 이로 인해 3장의 (case2)에 의해 정보가 노출될 수 있다. 이에 대응하기 위해선 모든 정보를 로컬 저장소에 저장하는 어플리케이션의 특징을 고려하여 사용자가 입력한 모든 정보를 암호화하여 저장해야 한다.

단순히 암호화만 적용했다고 해서 해당 어플리케이션이 안전하다고 확신할 수 없다. 해당 어플리케이션은 AES128 암호화 과정에서 필요한 IV와 암호화키를 하드 코딩된 값으로부터 생성한 고정된 값을 사용하였고, 또 두 값이 모두 동일하였다. 이러한 설계 방식 때문에 (case3)를 통해 데이터를 복호화 할 수 있다. 이러한 보안적 문제를 해결하기 위해 암호화에 사용하는 IV와 암호화키를 사용자의 입력값에 따라 동적으로 설정하는 방법을 고려해야 한다.

역공학에 취약한 안드로이드의 어플리케이션의 특성상 데이터 암호화를 적용하여도 다양한 보안 위협에 노출될 위험이 있다. [7]에서는 AES 알고리즘을 사용한 DEX 파일 암호화를 통해 불법 복제와 역공학으로부터 어플리케이션을 보호하는 방법에 대해 설명하였다. 저장된 데이터뿐만 아니라 역공학 자체를 방지하는 기술도 데이터를 보호하는 하나의 대안이 될 수 있다.

4.3 Hide Pass

Hide Pass 어플리케이션의 마스터 패스워드는 SHA-256을 통해 생성된다. 이때 해시값 생성 이전

Table 8. Decryption process summary

Name	Decryption Algorithm	Parameter	Output
Password Cloud	AES256/ECB/PKCS5Padding	Encryption Key: 32 bytes String Data: Encrypted Master Password	Decrypted Master Password
Password Management	AES128/CBC/PKCS5Padding	IV: 16 bytes String Encryption Key: 16 bytes String Data: Encrypted Master Password and UserData	Decrypted Master Password and UserData
Hide Pass	SHA-256	Data: Encrypted Master Password	Decrypted Master Password
	AES256/CBC/PKCS5Padding	IV: 16 bytes String Encryption Key: 32 bytes String Data: Encrypted UserData	Decrypted UserData
BeEasy	SHA-1	Data: Encrypted Master Password	Decrypted Master Password
	AES128/CBC/PKCS5Padding	IV: Encryption Key: 16 bytes String	Decrypted UserData

에 임의의 문자열을 추가하여 일반적으로 알려진 SHA-256 해시테이블에 대해 대응이 가능하다. 또한, 단방향 암호화이기 때문에 무차별 대입 공격을 통해서만 원본 데이터를 획득할 수 있는데 일정 길이 이상의 패스워드는 매우 많은 시간을 소비하게 되어 일반적인 컴퓨팅 환경에서는 공격 실패 가능성이 높아진다. 다만, 사용자 관리 데이터는 마스터 패스워드 복구 없이 xml 파일에 저장된 암호화된 마스터 패스워드와 하드 코딩된 문자열을 이용하여 생성된 암호화키에 의해 암호화되기 때문에 (case 3)에 의해 쉽게 노출된다. 이를 위해 IV와 암호화키 생성 시 하드 코딩된 값을 이용하는 것을 피하고, 암호화키를 Android KeyStore에 안전하게 보관하여 외부로부터 보호해야 한다.

4.4 BeEasy

BeEasy 어플리케이션의 마스터 패스워드는 숫자 6자리로 구성되며 SHA-1을 통해 암호화 되었다. 단방향 암호화 방식이지만 각 자리별 경우의 수가 작기 때문에 Table 5.에서 설명했듯이 짧은 시간에 모든 경우의 수를 확인할 수 있었다. 무차별 대입 공격으로부터 마스터 패스워드를 보호하기 위해선 숫자뿐만 아니라 영문 대/소문자, 특수문자와 같이 사용될 수 있는 문자의 수를 증가시켜 경우의 수를 증가시켜야 한다. 사용자 관리 데이터의 경우 대칭키 방식으로 암호화되어 있지만 각 데이터별 암호화 키와 IV가 하드코딩으로 그대로 노출되어 있었다. 이는 OWASP Mobile Top 10 중 M10: Insufficient Cryptography에 해당되며 (case 3)에서 동적 분석을 통해 복호화가 가능해진다. 이를 위해서 암호화키와 같은 민감한 데이터는 하드 코딩 방식으로 저장해서는 안 되며 운영체제에서 제공하는 기본 암호화 솔루션보다 더 강도 높은 솔루션을 통해 보호해야 한다.

V. 결 론

본 논문에서는 비밀번호 관리 어플리케이션 4종을 대상으로 데이터 획득 및 암호화된 데이터를 식별하였고, 암호화 알고리즘별 복호화 방법에 대해 설명하였으며, 어플리케이션에 대한 보안성 평가를 진행하였다.

먼저 데이터 복호화를 위해서 비밀번호 관리 어플리케이션의 마스터 패스워드와 사용자 관리 데이터의 저장 위치, 그리고 암호화 알고리즘을 식별하였으며, 이에 대응되는 복호화 알고리즘을 구현하여 데이터 복호화를 진행하였다. Table 8.과 같이 복호화 알고리즘 구현에 있어서 대칭키 암호인 경우 암호화 알고리즘에서 식별한 암호 알고리즘 종류 및 파라미터를 동일하게 사용하여 복호화할 수 있었고, 해시함수인 경우 전수조사 공격을 통해 암호화된 데이터를 입력값으로 넣어 복호화할 수 있었다.

보안성 평가를 위해 3장에서 설명한 3가지 분석 절차를 기준으로 취약 요소 확인과 이에 대한 대응 방안을 제안하였다. 대칭키 방식의 암호화는 암호화키를 안전하게 보관해야 한다. 해시함수를 통한 암호화는 입력받은 패스워드에 암호학적 솔트를 추가하여 사전에 생성된 해시테이블에 대응해야 하고 패스워드 설정 시 강도 높은 패스워드를 요구하여 무차별 대입 공격에 대비해야 한다. 또한, 이러한 암호화 알고리즘을 사용하는 데 있어서 하드 코딩된 값은 지양해야 하며 사용자의 환경별로 다른 암호화 키가 생성되도록 동적으로 키 생성 알고리즘을 구현해야 한다.

References

[1] Gi-Hoon Nam, Byoung-Jin Seok, Seong-Hyeon Gong, Yeog Kim and Chang-Hoon Lee, "Method of hijacking a user account using a password manager vulnerability," Journal of Digital Forensics, 12(1), pp. 9-18, Jun. 2018.

[2] Hyera Jeong and Jaewoo So, "Security of Password Vaults of Password Managers," Journal of The Korea Institute of Information Security & Cryptology, 28(5), pp. 1047-1057, Oct. 2018.

[3] Jungseong Park and Sangjin Lee, "A Study on Acquisition of Google Password Manager Data in Chrome Browser," Journal of Digital Forensics, 15(2), pp. 12-23, Jun. 2021.

[4] Yasasw. Kumarakalva, Vidya Shankar. G. S, Shreekara. K. K, Sridhar. P. H and Asha G. R, "A Secure Password Manager," Journal of

Emerging Technologies and Innovative Research, vol. 4, no. 3, pp. 204-206, Mar. 2017.
〔5〕 Chaitanya Rahalkar and Dhaval Gujar, "A Secure Password Manager," International Journal of Computer Applications, vol. 178, no. 44, pp. 5-9, Aug. 2019.
〔6〕 OWASP Mobile Top 10 | OWASP Foundation, "OWASP Mobile Top 10", https://owasp.org/www-project-mobile-top-10/, Dec. 2023.
〔7〕 JungHyun Kim and Kang Seung Lee, "Robust Anti Reverse Engineering Technique for Protecting Android Applications using the AES Algorithm," Journal of KIISE, 42(9), pp. 1100-1108, Sep. 2015.

〈저자소개〉

김 한 결 (Han-gyeol Kim) 학생회원
2019년 3월~현재: 강남대학교 소프트웨어응용학부 재학
〈관심분야〉 정보보호, 디지털포렌식

이 신 영 (Sinyoung Lee) 학생회원
2018년 3월~현재: 강남대학교 소프트웨어응용학부 재학
〈관심분야〉 정보보호, 디지털포렌식

박 명 서 (Myungseo Park) 종신회원
2013년 2월: 국민대학교 수학과 이학사
2015년 2월: 국민대학교 금융정보보안학과 이학석사
2014년 12월~2017년 2월: 국가보안기술연구소 연구원
2021년 8월: 국민대학교 금융정보보안학과 이학박사
2021년 9월~2022년 2월: 국민대학교 금융정보보안학과 박사후연구원
2022년 3월~2023년 8월: 강남대학교 ICT융합공학부 조교수
2023년 9월~현재: 한성대학교 융합보안학과 조교수
〈관심분야〉 정보보호, 디지털포렌식