

301 – Find hidden information

Team Information

Team Name : ForensicGPT

Team Member : Eungchang Lee, Donghyun HA, Hyunwoo Shin, Jinhyeok Son

Email Address : forensicgpt@googlegroups.com

Instructions

Description The police obtained information that the suspect was attempting to leak confidential information by hiding it on his personal smartphone. During the search and seizure process, no traces of classified information could be found. The police asked the suspect where he hid the confidential information on his smartphone, but he said he didn't know. It is known that the suspect has been interested in Android app development. Find confidential information hidden by the suspect.

Target	Hash (MD5)
APKS.zip	7d5de865c82cb5d08a39ae5b523904cf

Questions

Please solve all problems based on UTC+9 time zone.

1. What is the signature information of the APK file where confidential information is hidden? (60 points)
(MD5, SHA1, SHA256 must all be obtained. 20 points each)
2. What is the confidential information decryption algorithm? (90 points)
3. What is the decrypted plaintext of encrypted confidential information? (150 points)

Teams must:

- Describe step-by-step processes for generating your solution.
- Specify any tools used for this problem.

Tools used:

Name:	JADX GUI	Publisher:	skylot
Version:	1.4.7		
URL:	https://github.com/skylot/jadx		


















Name:	VS Code	Publisher:	Microsoft
Version:	May 2023		
URL:	https://code.visualstudio.com/		

Step-by-step methodology:

다운로드 받은 APK-001.zip 압축을 풀어보면 apk파일들을 확인할 수 있습니다.

1) What is the signature information of the APK file where confidential information is hidden? (60 points)

(MD5, SHA1, SHA256 must all be obtained. 20 points each)

 com.yes24.commerce.apk	2023년 4월 26일 오전 9:25
 teamdoppelganger.smarterbus.apk	2023년 4월 25일 오전 6:30
 kr.or.nhic.apk	2023년 4월 25일 오전 6:29
 kr.co.hpoint.hdgm.apk	2023년 4월 25일 오전 6:29
 kr.co.alba.webappalba.m.apk	2023년 4월 25일 오전 6:29
 com.selabs.speak.apk	2023년 4월 25일 오전 6:29
 com.samsung.everland.android.mobileApp.apk	2023년 4월 25일 오전 6:29
 com.order.kyochonchicken.apk	2023년 4월 25일 오전 6:29
 com.mealant.tabling.apk	2023년 4월 25일 오전 6:29
 com.mcdonalds.mobileapp.apk	2023년 4월 25일 오전 6:29
 com.lgt.tmoney.apk	2023년 4월 25일 오전 6:29
 com.firstscreen.weather.apk	2023년 4월 25일 오전 6:28
 com.encar.encarMobileApp.apk	2023년 4월 25일 오전 6:28
 com.oliveyoung.apk	2023년 4월 25일 오전 6:26
 kr.go.keis.worknet.apk	2023년 4월 25일 오전 6:26
 oops.ledscroller.apk	2023년 4월 25일 오전 6:25
 com.mobilefence.family.apk	2023년 4월 25일 오전 6:25

[그림 1] apk파일 목록

압축해제 된 파일들을 수정일을 기준으로 정렬하여 조작 의심 파일을 찾아내었습니다.

다른 apk들은 주로 4월 25일 오전 5시 29분 부터 4월 25일 오전 6시 30분 까지 몰려 있었으나 **com.yes24.commerce.apk**만 다음날인 4월 26일 오전 9시 25분에 수정되어 조작 의심파일로 특정하게 되었습니다.

Jadx-gui를 이용하여 apk파일을 디컴파일한 후, AndroidManifest.xml을 통해 해당 apk의 버전이 2.9.9버전임을 알아내었습니다.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="195" android:versionName="2.9.9" android:compile
<uses-sdk android:minSdkVersion="22" android:targetSdkVersion="31"/>
<supports-screens android:anyDensity="true" android:smallScreens="true" android:normalScreens="true" android:largeScreens="true"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission android:name="android.permission.DISABLE_KEYGUARD"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-feature android:name="android.hardware.telephony" android:required="false"/>
<uses-feature android:name="android.hardware.camera" android:required="false"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE"/>
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="com.google.android.gms.permission.AD_ID"/>
<queries>
```

[그림 2] AndroidManifest.xml

동일한 버전의 원본 apk를 구하기 위하여 apkpure.com을 이용하여 com.yes24.commerce의 2.9.9버전을 다운로드 받았습니다.

(<https://apkpure.com/%EC%98%88%EC%8A%A424-%EB%8F%84%EC%84%9C-%EC%84%9C%EC%A0%90/com.yes24.commerce/variant/2.9.9-APK>)

그 후, apk의 서명 소유자 비교를 통해 원본과 비교를 진행하였습니다.

유효한 APK 서명 v1을(를) 찾았습니다.

서명자 DEVELOPE.RSA (META-INF/DEVELOPE.SF)

```
유형: X.509
버전: 3
시리얼 번호: 0x42142261
소유자: CN=developer, OU=yes24, O=yes24, L=seoul, ST=seoul, C=82
유효 시작 시각: Tue Apr 25 10:55:40 KST 2023
유효 종료 시각: Wed Apr 24 10:55:40 KST 2024

공개키 타입: RSA
지수: 65537
모듈러스 크기 (비트): 1024
모듈러스: 1133172591065828029881808849351763980318912888520063822087684411668

서명 유형: MD5withRSA
서명 OID: 1.2.840.113549.1.1.4

MD5 지문: 1F 81 4B 7C 3E A1 80 E2 29 31 C6 60 1F D7 75 1F
SHA-1 지문: 9F 3E D0 61 54 83 64 B3 C9 4B 84 DD 2D 67 55 59 AF 34 49 27
SHA-256 지문: 98 2B 7D 04 92 CD 43 42 58 D3 44 91 F9 F3 3C E6 32 2C 24 AE D6
```

[그림 3] 조작 의심 파일의 서명

유효한 APK 서명 v1을(를) 찾았습니다.

서명자 CERT.RSA (META-INF/CERT.SF)

유형: X.509

버전: 3

시리얼 번호: 0x4d192b2c

소유자: CN=Woo Young Seo, O=annex institute, L=Seoul, ST=Yeongdeungpo-gu, C=KR

유효 시작 시각: Tue Dec 28 09:11:24 KST 2010

유효 종료 시각: Wed Dec 15 09:11:24 KST 2060

공개키 타입: RSA

지수: 65537

모듈러스 크기 (비트): 1024

모듈러스: 148184275267172132035677996459930226066037611646129685037790977226954361954959

서명 유형: SHA1withRSA

서명 OID: 1.2.840.113549.1.1.5

MD5 지문: 7E E8 3E EA 72 61 A6 5A FE 23 45 9B 8C 5F 5B D9

SHA-1 지문: 79 D3 AC 94 C0 65 FC 7D 33 DD A8 B3 27 3B FB C0 08 75 7B D0

SHA-256 지문: F9 E9 F9 4F B1 9B 79 3C F3 50 A5 47 9B 38 77 8A 1C 29 4C 13 CF 23 CA F9 E

[그림 4] 원본 파일의 서명

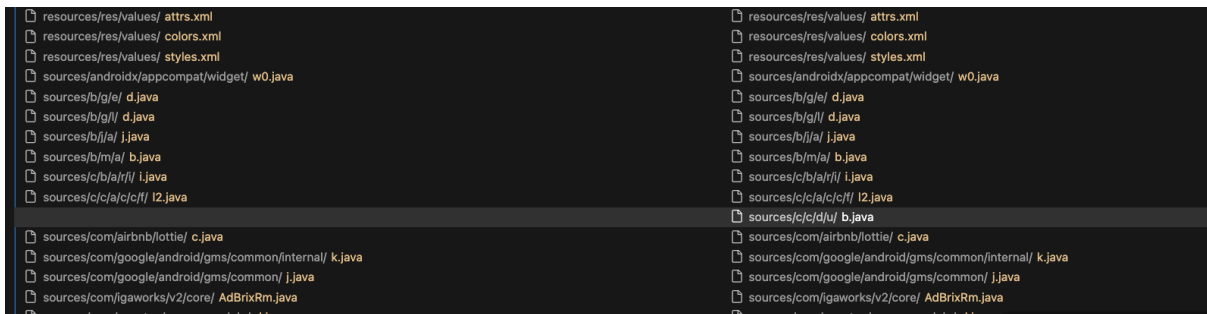
조작 의심 파일의 서명의 CN, O, ST가 원본파일의 서명과는 다른 것을 확인하였습니다. 이는 해당 apk파일이 조작되었음을 나타내고 있습니다.

조작된 파일의 서명 정보

해시명	해시값
MD5	1F814B7C3EA180E22931C6601FD7751F
SHA1	9F3ED061548364B3C94B84DD2D675559AF344927
SHA256	982B7D0492CD434258D34491F9F33CE6322C24AED6727BB0CC8E0DABAFEE53F3

2. What is the confidential information decryption algorithm? (90 points)

Jadx-gui의 모두 저장 기능을 이용하여 조작된 apk와 원본 apk의 디컴파일된 요소들을 저장한 후, VS Code의 폴더 비교 기능을 이용하여 다른 부분을 검사하였습니다.



[그림 5] 비교 결과

결과를 통해 c.c.d.u.b 클래스가 추가되었음을 확인할 수 있었습니다.

```
/* loaded from: classes.dex */
public class b {
    public static byte[] iv = {1, 2, 3, 4, 5, 6, 7, 8, 9, 16, 17, 18, 19, 20, 21, 22};

    public static String dbk(String k, String t) throws Exception {
        return dd(k.getBytes(), Base64.decode(t, 0));
    }

    private static String dd(byte[] k, byte[] t) throws Exception {
        SecretKeySpec secretKeySpec = new SecretKeySpec(k, "AES");
        Cipher c2 = Cipher.getInstance("AES/CBC/PKCS5Padding");
        c2.init(2, secretKeySpec, new IvParameterSpec(iv));
        byte[] sk = c2.doFinal(t);
        return new String(sk);
    }
}
```

[그림 6] 추가된 클래스

Java의 Cipher 모듈에서 첫번째 인자는 mode를 선택하는 것으로 2는 DECRYPT 모드임을 확인할 수 있습니다.

Constant Field	Value
DECRYPT_MODE	2
ENCRYPT_MODE	1
PRIVATE_KEY	2
PUBLIC_KEY	1
SECRET_KEY	3
UNWRAP_MODE	4
WRAP_MODE	3

[그림 7] Cipher 모듈의 mode 상수 값

즉, 숨겨진 정보는 **AES/CBC/PKCS5Padding**으로 암호화 되어있음을 유추할 수 있습니다.

3. What is the decrypted plaintext of encrypted confidential information? (150 points)

그림 6을 통해 복호화 하는 함수를 찾을 수 있었고, 해당 함수를 사용하고 있는 dbk함수를 보면, 인자로 k와 t를 받아서 k는 byte로 t는 base64디코딩을 수행하는 것을 알 수 있습니다. 이후 dd에게 인자로 넘기며 k는 key로 사용되고 t는 암호문으로 사용됨을 확인할 수 있습니다. 또한, iv 바이트 배열을 통해 iv 값까지 알아낼 수 있었습니다.

Jadx-gui를 이용하여 dbk함수의 사용을 추적하였으며 ActSetting의 dd함수에서 사용되고 있음을 확인하였습니다.

```
public String dd(String it) {  
    String rv = BuildConfig.FLAVOR;  
    Log.e("developer", "dd");  
    try {  
        String dt = c.c.d.u.b.dbk(this.ky, it);  
        rv = dt;  
        Log.e("developer", "t: " + rv);  
        return rv;  
    } catch (Exception e2) {  
        e2.printStackTrace();  
        return rv;  
    }  
}
```

그림 8 dbk가 사용되는 함수

This.ky를 키로 사용하고 있으며 인자를 암호문으로 사용하고 있는 것을 확인할 수 있습니다. 그리고 This.ky는 해당 클래스파일에서 "ActSettingCreate"로 확인됩니다.

암호문의 경우 폴더 diff를 통해 smali코드를 비교하여 d.a.a에 base64 형식의 데이터가 추가됨을 확인할 수 있습니다.


```

.method public static valueOf(Ljava/lang/String;)Ld/a/a;
.registers 2

const-class v0, Ld/a/a;

const-string v0, "u7AdMnRkEXIXlNFgqlvJuyIrtjEWRY88M00A3GdkSZtN5jbQsVAqS8J9iEkIx3GK"

invoke-static {v0, p0}, Ljava/lang/Enum;-->valueOf(Ljava/lang/Class;Ljava/lang/String;)Ljava/lang/Enum;

move-result-object p0

check-cast p0, Ld/a/a;

return-object p0
.end method

```

[그림 9] 추가된 암호문

따라서 아래의 정보들을 가지고 복호화를 진행하였습니다.

복호화 방식	AES/CBC/PKCS5Padding
암호문	u7AdMnRkEXIXlNFgqlvJuyIrtjEWRY88M00A3GdkSZtN5jbQsVAqS8J9iEkIx3GK
IV	1, 2, 3, 4, 5, 6, 7, 8, 9, 16, 17, 18, 19, 20, 21, 22
key	ActSettingCreate(41 63 74 53 65 74 74 69 6e 67 43 72 65 61 74 65)

간단한 python 스크립트를 작성하여 나온 결과값은 **information was stored in my private cloud**으로 확인할 수 있었습니다.

```

from base64 import b64decode
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

def decrypt_aes_cbc_pkcs5padding(ciphertext_base64, key, iv):
    ciphertext = b64decode(ciphertext_base64) # base64 디코딩

    cipher = AES.new(key, AES.MODE_CBC, iv=iv) # AES/CBC/PKCS5Padding으로 복호화 객체 생성
    plaintext = unpad(cipher.decrypt(ciphertext), AES.block_size) # 복호화 후 패딩 제거

    return plaintext.decode('utf-8') # 바이트를 문자열로 변환

ciphertext_base64 = "u7AdMnRkEXIXlNFgqlvJuyIrtjEWRY88M00A3GdkSZtN5jbQsVAqS8J9iEkIx3GK"
key = b"ActSettingCreate"
iv = b"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x10\x11\x12\x13\x14\x15\x16"
# 복호화
plaintext = decrypt_aes_cbc_pkcs5padding(ciphertext_base64, key, iv)

print(plaintext)

```

[그림 10] 복호화 python 스크립트