

BÁO CÁO THỰC HÀNH

Môn học: Pháp chứng kỹ thuật số

Kỳ báo cáo: Buổi 05 (Session 05)

Tên chủ đề: Mobiles Forensics

GVHD: Đoàn Minh Trung

Ngày báo cáo: 07/05/2024

Nhóm: 07

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT334.O21.ATCL.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Đình Bảo Long	21522303	21522303@gm.uit.edu.vn
2	Nguyễn Tấn Phát	21522447	21522447@gm.uit.edu.vn
3	Ngô Minh Thiên	21522623	21522623@gm.uit.edu.vn
4	Đào Vĩnh Thịnh	21522632	21522632@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:¹

STT	Công việc	Kết quả tự đánh giá
1	Kịch bản 01	100%
2	Kịch bản 02	100%
3	Kịch bản 03	100%
4	Kịch bản 04	100%

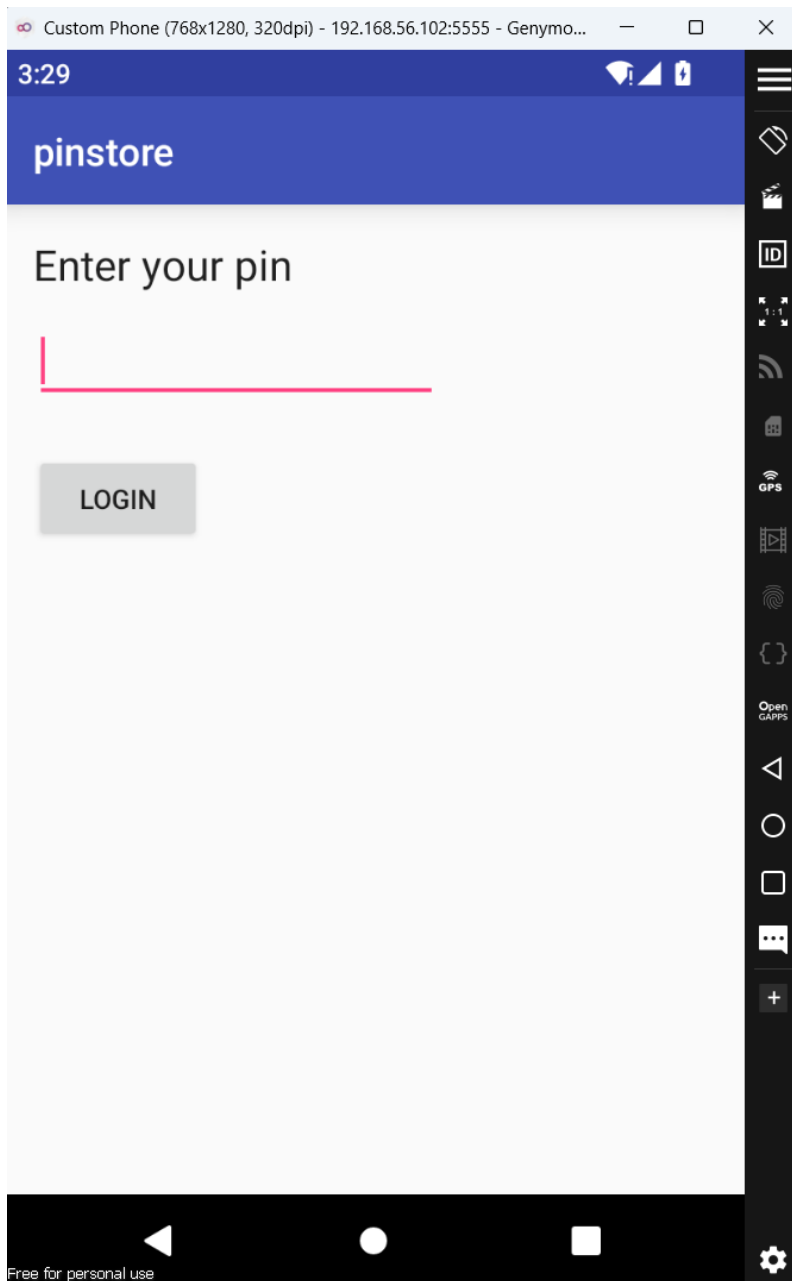
¹ Ghi nội dung công việc, các kịch bản trong bài Thực hành,

BÁO CÁO CHI TIẾT

1. Kịch bản 01. Thực hiện phân tích ứng dụng Android

- Mô tả: Phân tích ứng dụng Android, tìm mã PIN trong ứng dụng để tìm flag.
- Tài nguyên thực hiện: pinstore.zip
- Yêu cầu – Gợi ý: Sử dụng các công cụ dịch ngược (decompile) trên mã nguồn Android để phân tích.

Đáp án:



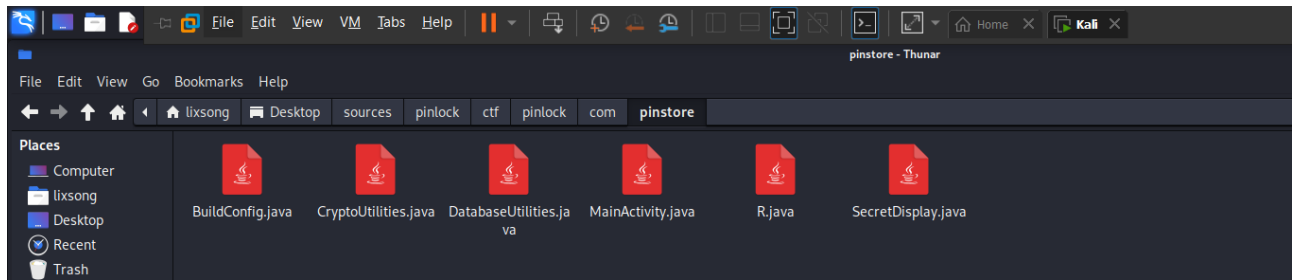
- Khi cài đặt thì nó yêu cầu ta nhập pin
- Nếu sai thì in ra "Incorrect Pin, try again"
- Ta dùng công cụ jadx để decompile file apk

```
(lixsong@kali)-[~/jadx]
$ jadx -d /home/lixsong/Desktop /home/lixsong/Downloads/resources-session05/pinstore.apk
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
INFO - loading ...
INFO - processing ...
INFO - done

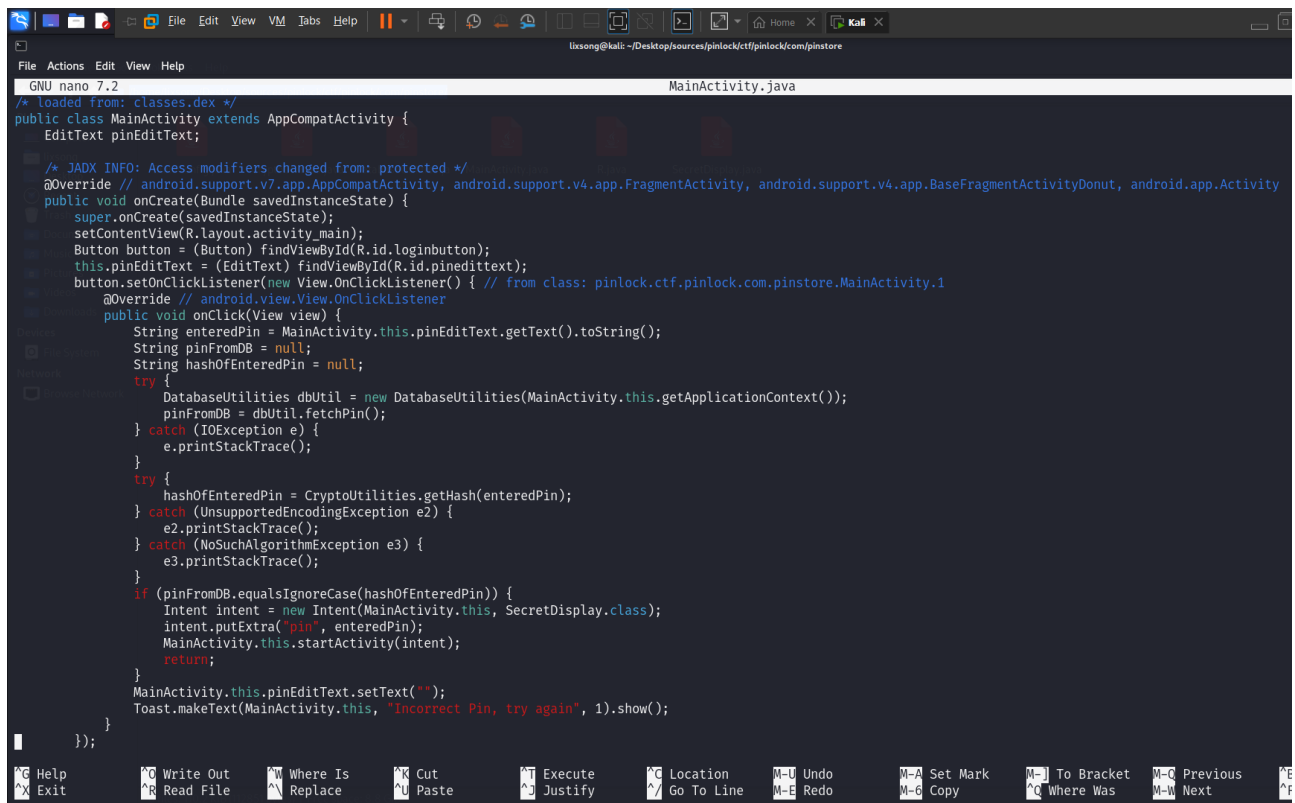
(lixsong@kali)-[~/jadx]
$
```

Trong đường dẫn

“/home/lixsong/Desktop/sources/pinlock/ctf/pinlock/com/pinstore/” ta thấy có các file dưới đây.



- Thường trong MainActivity sẽ là chứa các Activity chính của chương trình



- Đoạn code trong MainActivity

- Ta có thể thấy

+ Chương trình khởi tạo biến để lấy thông tin pin ta nhập vào sau đó khởi tạo biến pinFromDB là null và biến hashOfEnteredPin là null

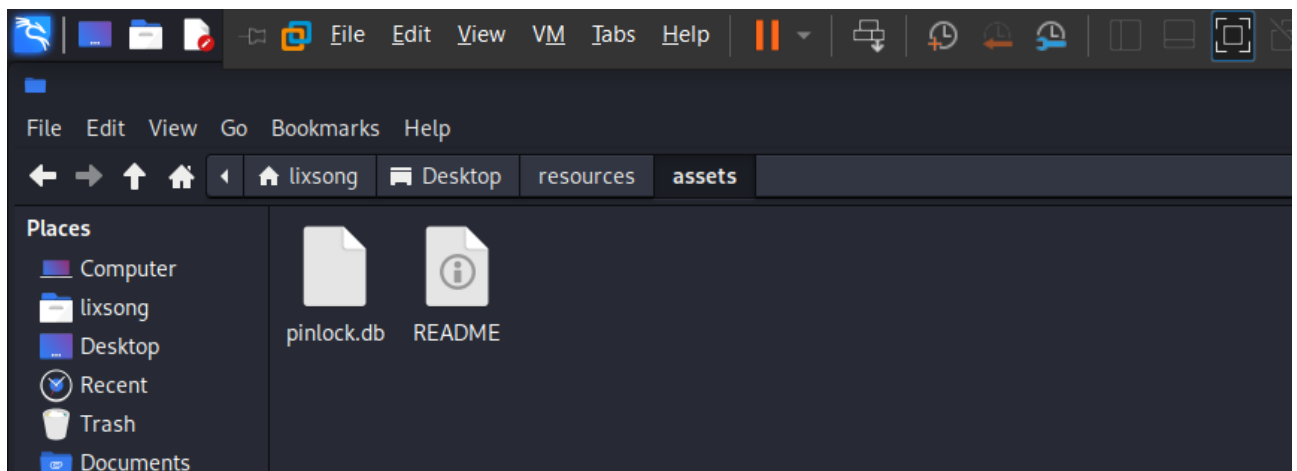
+ Ta có thể đoán là nó sẽ lấy mã pin người dùng nhập và mã PIN từ database. Sau đó nó sử dụng lớp CryptoUtilities để băm mã PIN nhập vào và so sánh với mã PIN từ database

+ Nếu 2 mã PIN khớp thì nó sẽ chuyển qua màn hình SecretDisplay còn nếu không khớp thì sẽ có thông báo được hiển thị

- Vậy ta chỉ cần vào xem database của nó để xem mã hash là được

```
public static String getHash(String input) throws NoSuchAlgorithmException, UnsupportedEncodingException {
    byte[] input_bytes = input.getBytes();
    MessageDigest md = null;
    try {
        md = MessageDigest.getInstance("SHA-1");
    } catch (NoSuchAlgorithmException e) {
    }
    md.update(input_bytes, 0, input_bytes.length);
    byte[] hash_bytes = md.digest();
    String output = getHex(hash_bytes);
    return output;
}
```

- Trong CryptoUtilities.Class ta thấy nó sử dụng SHA-1 để hash mã PIN của ta
- Khi decompile các file và xem trong các thư mục ta thấy trong thư mục “/resources/assets/” có 1 file tên là pinlock.db có vẻ như đây sẽ là file database của ta



- Dùng sqlite3 để xem các tables trong database
- Xem qua các tables thì tables pinDB có vẻ là khớp với SHA-1 của ta

```
File Actions Edit View Help

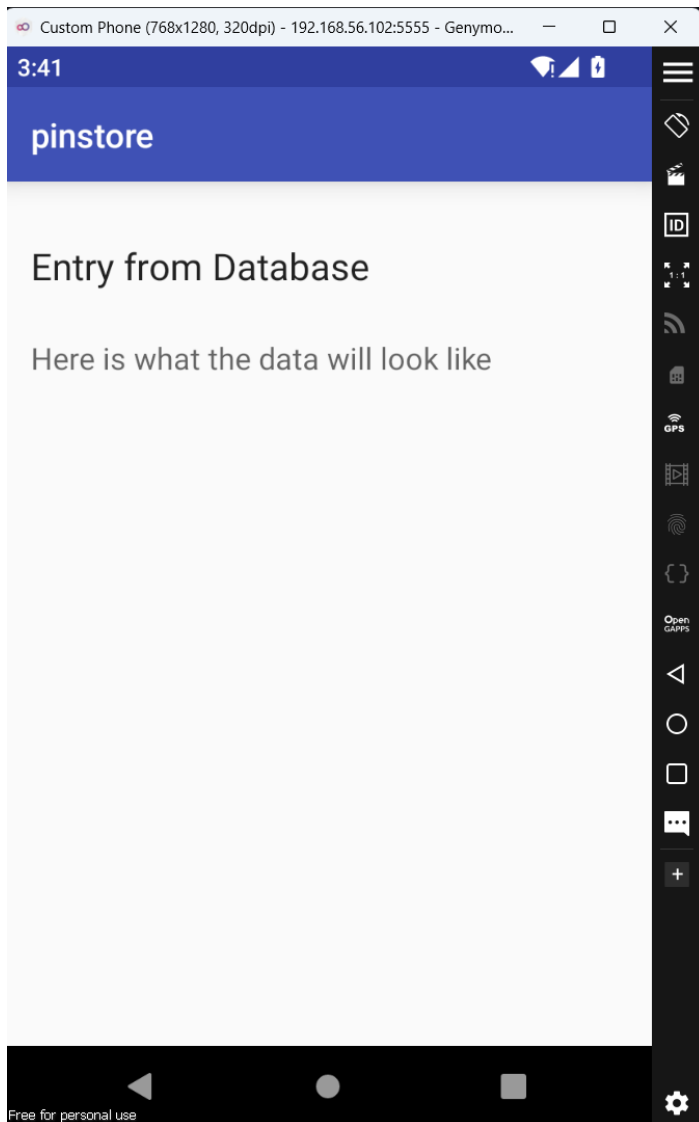
(lixsong@kali)-[~/Desktop/resources/assets]
$ sqlite3 pinlock.db
SQLite version 3.45.1 2024-01-30 16:01:20
Enter ".help" for usage hints.
sqlite> .tables
android_metadata  pinDB          secretsDBv1       secretsDBv2
sqlite> select * from android_metadata
... >
... > ^C
... > ^C
... >
... > ;
en_US
sqlite> select * from pinDB
Dev ... > ;
1|d8531a519b3d4dfebece0259f90b466a23efc57b
sqlite> select * from secretsDBv1
... > ;
1|hcsvUnln5jMdw3GeI4o/txB5vaEf1PFAnKQ3kPsRW2o5rR0a1JE54d0BLkzXPtqB
sqlite> select * from secretsDBv2;
1|Bi528nDlNBcX9BcCC+ZqGQo10z01+GOWSmvxRj7jg1g=
sqlite>
```

Thử dùng tool decode SHA1 online thì có được kết quả là 7498

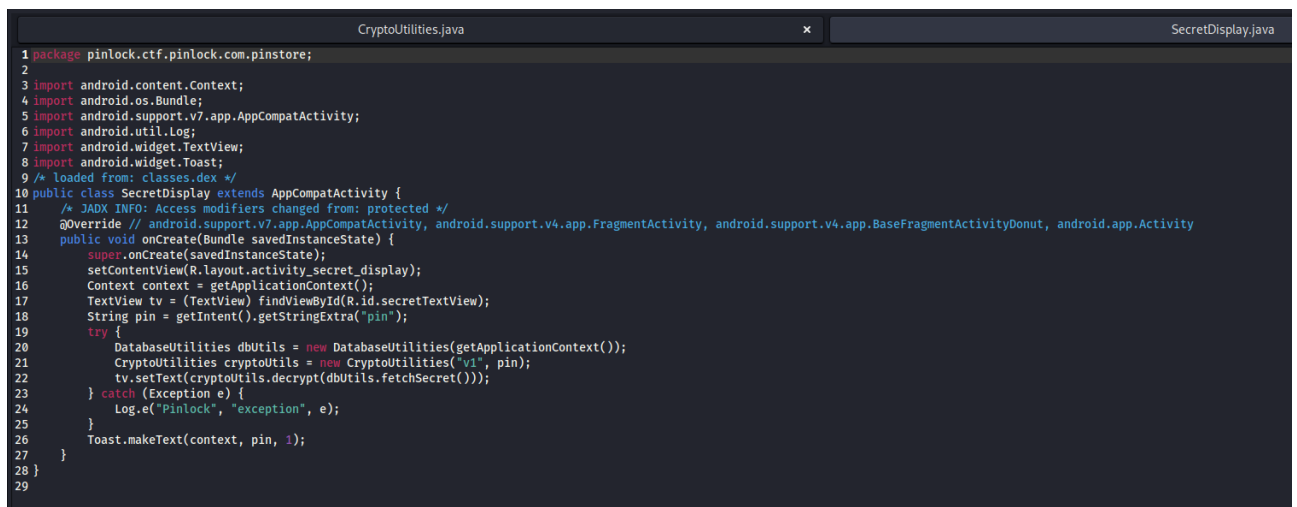
<https://www.dcode.fr/sha1-hash>



Dùng mã pin 7498 login thành công ứng dụng pinstore



- Kết quả khi ta nhập vào mã PIN đúng
- Nhưng có vẻ là chưa ra được flag của ta



- Ta xem code của secretdisplay

- Ta thấy sau khi ta nhập mã pin vào thì nó sẽ lấy giá trị đó và dùng CryptoUtilities để thực hiện mã hóa và giải mã bằng cách truyền phiên bản "v1" và "pin" vào
- Sau đó nó sử dụng hàm fetchSecret() để lấy chuỗi bí mật từ database sau đó chuỗi này được decrypt và kết quả giải mã được sẽ hiển thị lên giao diện

```
public String fetchSecret() throws IOException {
    openDB();
    Cursor cursor = this.db.rawQuery("SELECT entry FROM secretsDBv1", null);
    String secret = "";
    if (cursor.moveToFirst()) {
        secret = cursor.getString(0);
    }
    Log.d("secret", secret);
    cursor.close();
    return secret;
}

public String fetchPin() throws IOException {
    openDB();
    Cursor cursor = this.db.rawQuery("SELECT pin FROM pinDB", null);
    String pin = "";
    if (cursor.moveToFirst()) {
        pin = cursor.getString(0);
    }
    cursor.close();
    return pin;
}
```

- Đây là hàm fetchSecret() nhưng ta nhập nó lại không phải là flag

```
public SecretKeySpec getKey(String version) throws Exception {
    if (version.equalsIgnoreCase("v1")) {
        Log.d("Version", version);
        byte[] keyBytes = "t0ps3kr3tk3y".getBytes("UTF-8");
        MessageDigest md = MessageDigest.getInstance("SHA-1");
        SecretKeySpec keySpec = new SecretKeySpec(Arrays.copyOf(md.digest(keyBytes), 16), "AES");
        return keySpec;
    }
    Log.d("Version", version);
    byte[] salt = "SampleSalt".getBytes();
    char[] pinArray = this.pin.toCharArray();
    SecretKeyFactory secretKeyFactory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1");
    KeySpec ks = new PBEKeySpec(pinArray, salt, 1000, 128);
    SecretKey secretKey = secretKeyFactory.generateSecret(ks);
    SecretKeySpec keySpec2 = new SecretKeySpec(secretKey.getEncoded(), "AES");
    return keySpec2;
}
```

- Ta xem hàm getKey trong CryptoUtilities class
- Phương thức getKey(String version) trong lớp CryptoUtilities được sử dụng để tạo SecretKeySpec dựa trên phiên bản và pin.
- Nếu phiên bản là "v1", phương thức tạo một SecretKeySpec từ một chuỗi cố định "t0ps3kr3tk3y" và sử dụng thuật toán mã hóa AES.

- Nếu phiên bản không phải "v1", phương thức sử dụng thuật toán PBKDF2 và HMAC-SHA1 để tạo một SecretKey từ pin và salt. Sau đó, tạo một SecretKeySpec từ khóa đã được mã hóa và sử dụng thuật toán mã hóa AES.

- Kết quả là SecretKeySpec tạo ra dựa trên phiên bản và pin được sử dụng cho việc mã hóa và giải mã dữ liệu.

→ Ý tưởng là ta sẽ sử dụng pin và chuỗi secret từ secretsDBv2 là sẽ có flag của ta

Ta có đoạn code khai thác

```
File Edit Search View Document Help
1 import java.io.UnsupportedEncodingException;
2 import java.security.InvalidKeyException;
3 import java.security.MessageDigest;
4 import java.security.NoSuchAlgorithmException;
5 import java.security.spec.InvalidKeySpecException;
6 import java.util.Arrays;
7 import java.util.Base64;
8
9 import javax.crypto.BadPaddingException;
10 import javax.crypto.Cipher;
11 import javax.crypto.IllegalBlockSizeException;
12 import javax.crypto.NoSuchPaddingException;
13 import javax.crypto.SecretKeyFactory;
14 import javax.crypto.spec.PBEKeySpec;
15 import javax.crypto.spec.SecretKeySpec;
16
17
18 public class Bside {
19
20     public static void main(String[] args) {
21         String pin = "7498";
22         String secret1 = "hcsvUnln5jMdw3GeI4o/txB5vaf1PFAnKQ3kPsRW2o5rR0a1JE54d0BLkzXPtqB"; // Pin
23         String secret2 = "B1528nDlNBcX98cCC+ZqQo10z01+GOWSmvxRj7jg1g="; // SecretsDBv1
24         try {
25             byte[] decoded1 = Base64.getDecoder().decode(secret1);
26             byte[] decoded2 = Base64.getDecoder().decode(secret2);
27             Cipher cipher;
28             cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
29             SecretKeySpec key1;
30             key1 = new SecretKeySpec(Arrays.copyOf(MessageDigest.getInstance("SHA-1").digest("t0ps3kr3tk3y".getBytes()), 16), "AES");
31             cipher.init(Cipher.DECRYPT_MODE, key1);
32             String salida = new String(cipher.doFinal(decoded1), "UTF-8");
33             System.out.println("[*] SecretsDBv1 (encrypted): " + secret1);
34             System.out.println("[*] SecretsDBv1 (decrypted): " + salida);
35
36             char[] arrayOfChar2 = pin.toCharArray();
37             byte[] paramString = "SampleSalt".getBytes();
38             SecretKeySpec key2 = new SecretKeySpec(SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1").generateSecret(new PBEKeySpec(arrayOfChar2, paramString, 1000, 128)).getEncoded(), "AES");
39
40             cipher.init(Cipher.DECRYPT_MODE, key2);
41             salida = new String(cipher.doFinal(decoded2));
42
43             System.out.println("[*] SecretsDBv2 (encrypted): " + secret2);
44             System.out.println("[*] Flag: " + salida);
45         }
46         catch (UnsupportedEncodingException | NoSuchAlgorithmException | InvalidKeyException | IllegalBlockSizeException | BadPaddingException | InvalidKeySpecException e) {
47             e.printStackTrace();
48         }
49     }
50 }
```

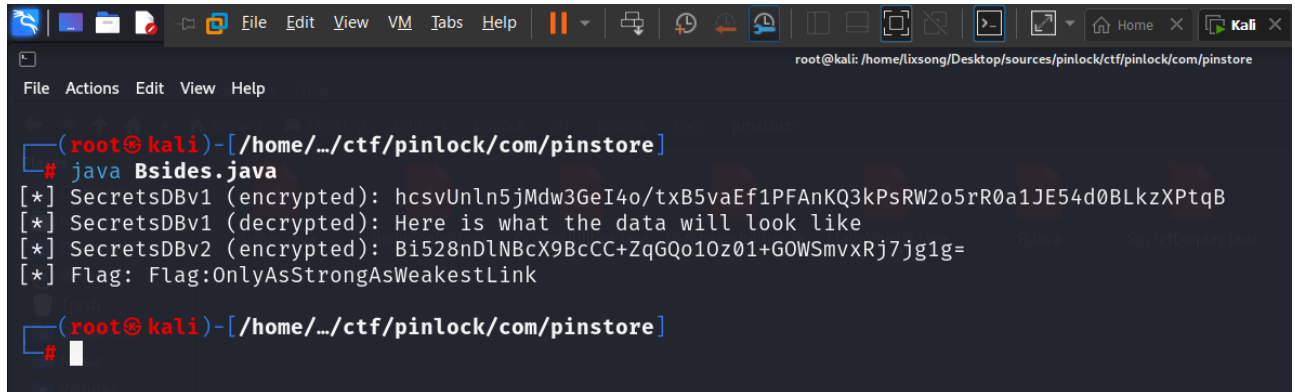
Dưới đây là ý nghĩa của từng phần trong đoạn code:

1. Khai báo các thư viện và các ngoại lệ có thể xảy ra trong quá trình mã hóa và giải mã.
2. Khai báo chuỗi `pin` là một mật khẩu đơn giản, có thể sử dụng để tạo ra khóa mã hóa cho dữ liệu.
3. `secret1` và `secret2` là các chuỗi đã được mã hóa lấy trong pinlock.db
4. Trong phần `try`, đoạn code thực hiện các bước sau:
 - Giải mã `secret1` sử dụng một khóa cụ thể được tạo ra từ chuỗi "t0ps3kr3tk3y" thông qua thuật toán SHA-1. Sau đó, dữ liệu được giải mã bằng thuật toán AES.
 - Tạo một khóa mới từ mật khẩu (`pin`) bằng cách sử dụng phương pháp PBKDF2 với HMAC-SHA1 và một salt cố định là "SampleSalt".
 - Sử dụng khóa mới để giải mã `secret2`.
 - In ra kết quả giải mã của `secret1` và `secret2`.

5. Trong trường hợp xảy ra các ngoại lệ như không thể tìm thấy thuật toán hoặc khóa không hợp lệ, chương trình sẽ in ra thông báo lỗi.

Tóm lại, đoạn code này giải mã hai chuỗi đã được mã hóa bằng AES, một bằng một khóa được tạo ra từ một chuỗi cố định và một bằng một mật khẩu được nhập vào. Điều này giúp bảo vệ dữ liệu nhạy cảm khỏi việc truy cập trái phép.

Kết quả :



```
(root@kali)-[/home/.../ctf/pinlock/com/pinstore]
# java Bsides.java
[*] SecretsDBv1 (encrypted): hcsvUnln5jMdw3GeI4o/txB5vaEf1PFAnKQ3kPsRW2o5rR0a1JE54d0BLkzXPtqB
[*] SecretsDBv1 (decrypted): Here is what the data will look like
[*] SecretsDBv2 (encrypted): Bi528nDlNBcX9BcCC+ZqGQo10z01+GOWSmvxRj7jg1g=
[*] Flag: Flag:OnlyAsStrongAsWeakestLink

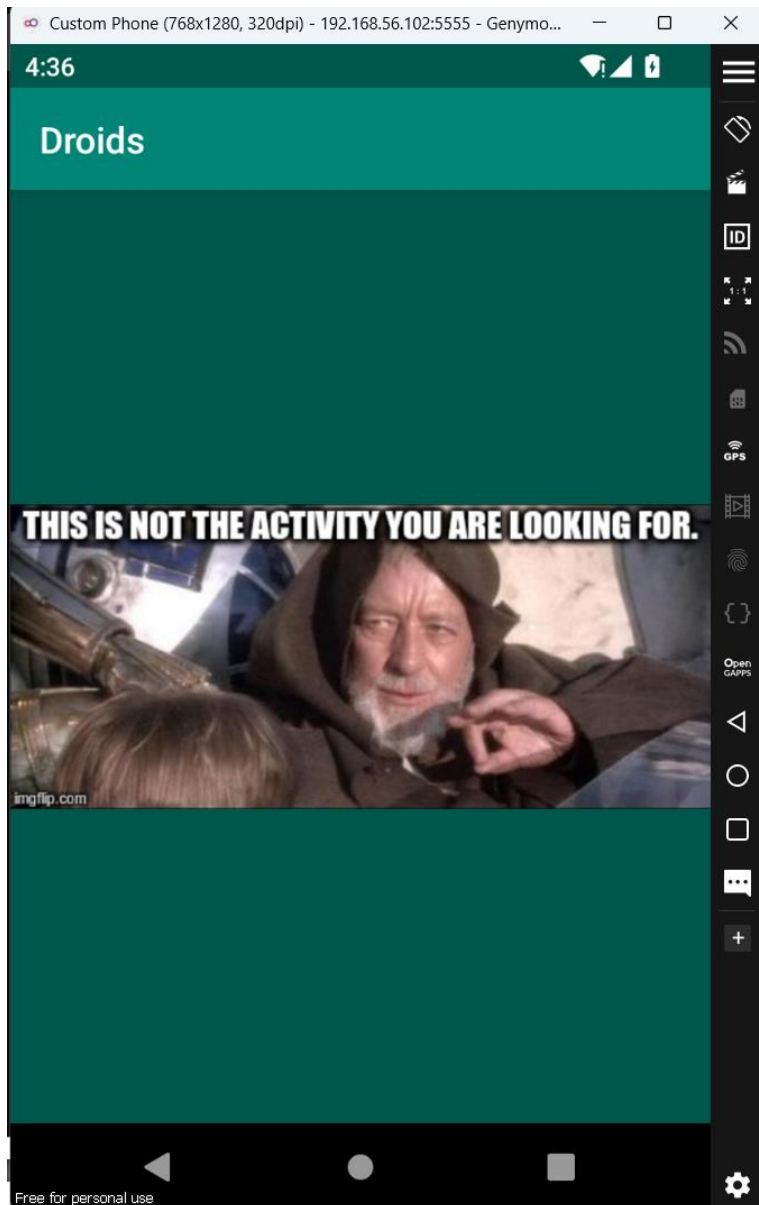
(root@kali)-[/home/.../ctf/pinlock/com/pinstore]
#
```

Flag: OnlyAsStrongAsWeakestLink

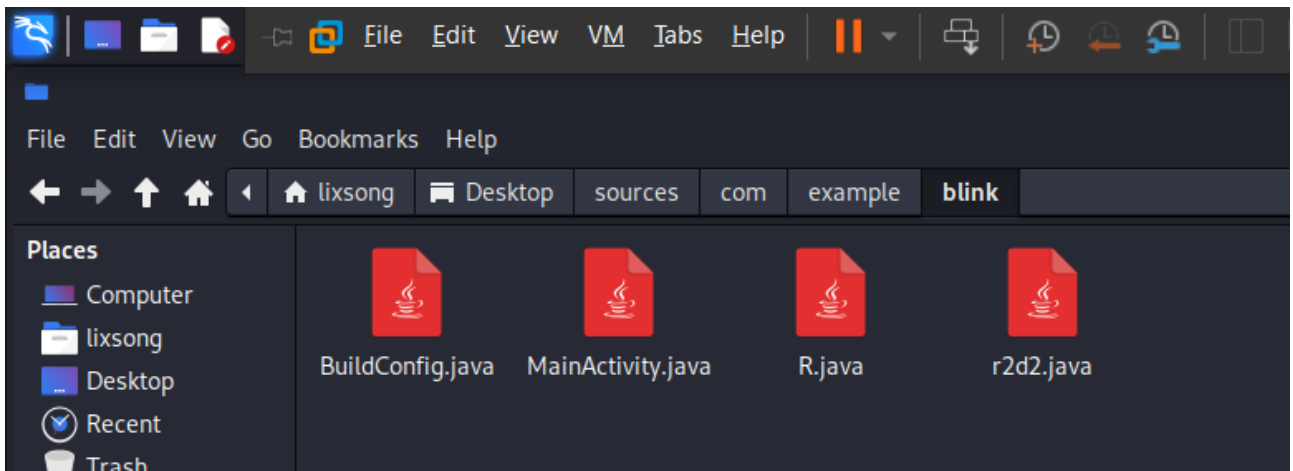
2. Kịch bản 02. Thực hiện phân tích tập tin ứng dụng thu được.

- Mô tả: Ứng dụng kb02 cần được phân tích thành mã smali để tìm flag.
- Tài nguyên thực hiện: kb02_zha.apk
- Yêu cầu – Gợi ý: sử dụng công cụ APKTool/ JADX/ dex2jar/ jdgui/ Android Studio, flag có dạng CTF{....}

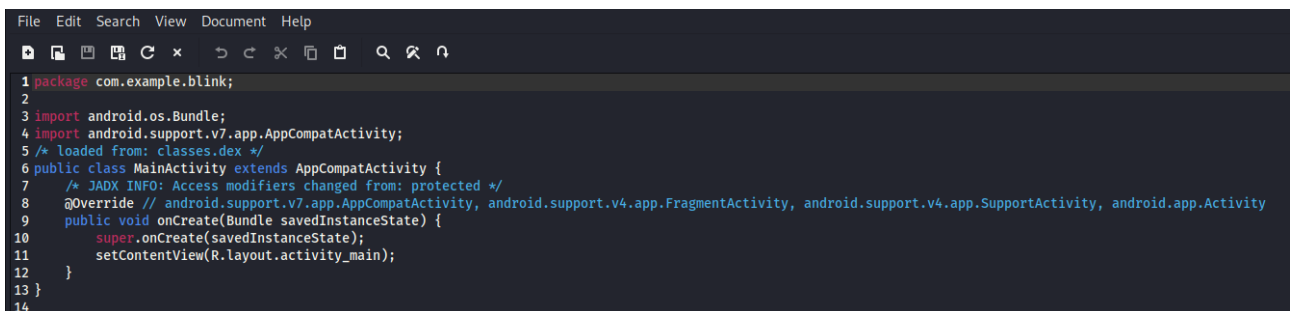
Đáp án:



- Khi cài đặt vào truy cập vào ứng dụng ta được như hình trên
- Dùng jadx để decompile file apk
- Ta thấy có 4 file.

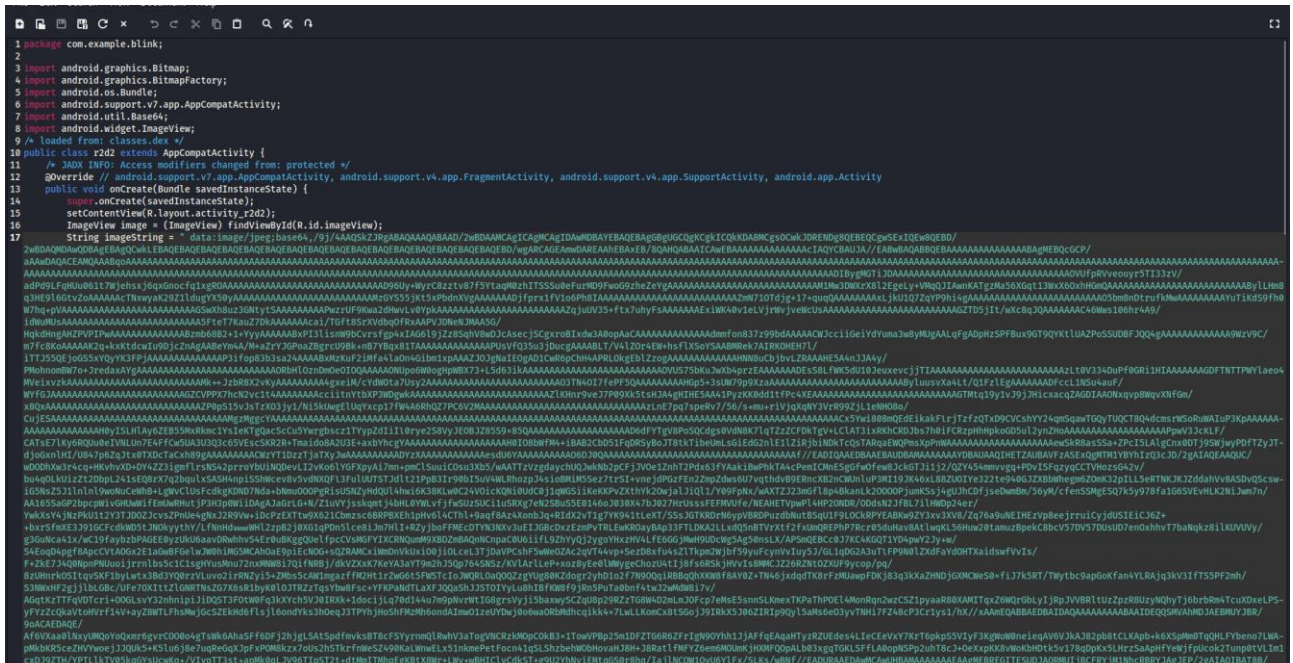


Kiểm tra code MainActivity.java thì không thấy có gì đặc biệt



Kiểm tra code r2d2.java thì thấy có đoạn mã base64 khá dài.

Ta thấy đoạn mã base64 này có dạng image jpeg.



Thực hiện chuyển base 64 thành hình bằng lệnh: echo "/9j/4AA..." > base64 -d flag.png

3. Kịch bản 03. Thực hiện phân tích tập tin ứng dụng thu được.

- Mô tả: Một ứng dụng có tính năng ghi nhớ các địa điểm mà người dùng muốn hay không muốn tham quan chỉ bằng dấu tick đơn giản trên bản đồ. Tìm flag.
- Tài nguyên: kb03_yon.apk
- Yêu cầu – Gợi ý: Decompile, chú ý CSDL của ứng dụng.

Gợi ý:

Đầu tiên ta dùng apktool để decompile và mở file assets để đọc Locations.db

```
(lixsong@kali)-[~/Downloads/resources-session05/kb03_yon/assets]
$ sqlite3 Location.db
SQLite version 3.45.1 2024-01-30 16:01:20
Enter ".help" for usage hints.
sqlite> select * from locations;
02/03/2019|37.7842927|-122.4053593|120.0
02/03/2019|37.7838412|-122.4041845|0.0
02/07/2019|37.7863323436302|-122.42828886956|120.0
02/07/2019|37.7851367932719|-122.402353584766|120.0
02/07/2019|37.782343920755|-122.404699847102|0.0
02/07/2019|37.7819822174966|-122.401994504035|0.0
02/07/2019|37.7673615647781|-122.467592954636|120.0
02/07/2019|37.7740060120854|-122.428736798465|120.0
02/07/2019|37.7738507141208|-122.428377382457|0.0
02/07/2019|37.7732178582912|-122.428442761302|0.0
02/07/2019|37.7720162653569|-122.426562532783|0.0
02/07/2019|37.7730111454145|-122.422018200159|120.0
02/06/2019|37.7737341079262|-122.417571097612|120.0
02/06/2019|37.7068383881913|-122.344666644931|120.0
02/06/2019|37.7087049563015|-122.443877533078|0.0
02/06/2019|37.6842005525645|-122.443419881165|0.0
02/06/2019|37.6847434257611|-122.441491037607|0.0
02/08/2019|37.7703669470161|-122.421883132294|0.0
02/08/2019|37.7692135433168|-122.421840216949|0.0
02/08/2019|37.7679922727318|-122.421840216949|0.0
02/08/2019|37.7703330236347|-122.419651534393|0.0
02/08/2019|37.7691456954801|-122.419651534393|0.0
```

Xem thông tin bên trong thì ta thấy được thông tin ngày tháng, toạ độ X, toạ độ Y và màu 120 và 0

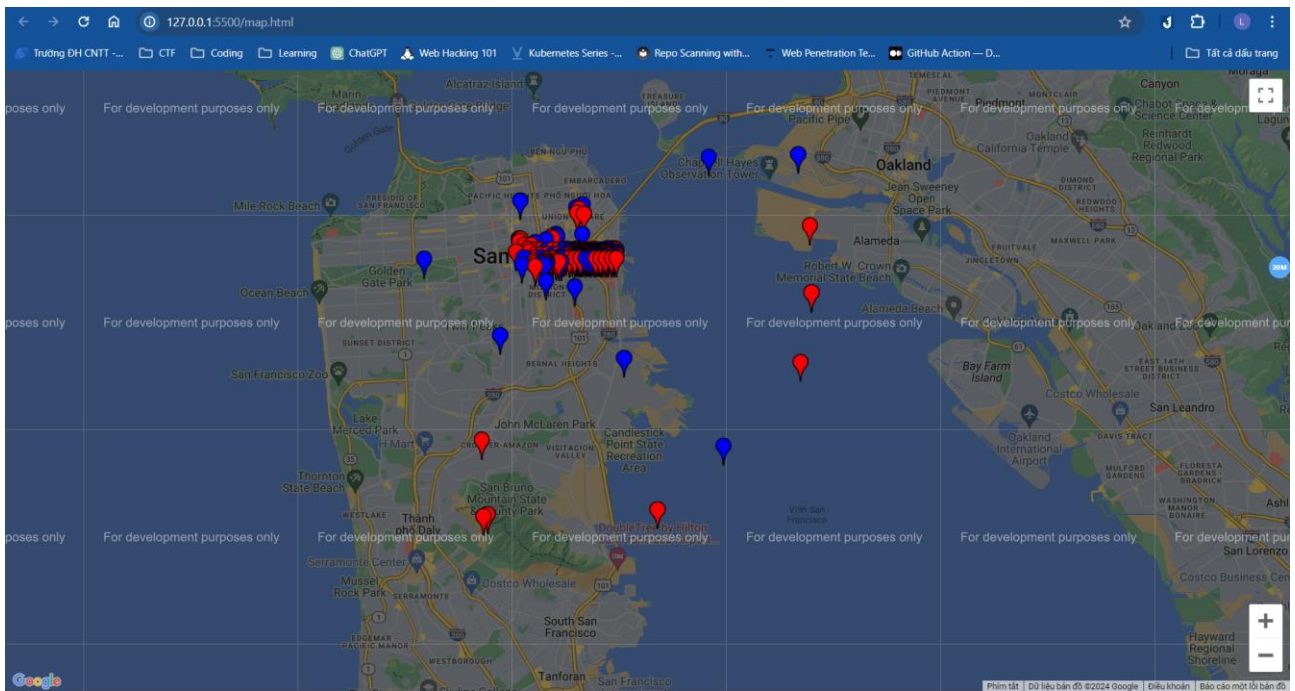
Thực hiện code để đọc data từ file Location và kết hợp api của google maps để thực hiện visualize lên google maps

```
map.py x map.html
map.py
1 import googlemaps
2 import sqlite3
3 import gmplot
4
5 # Load API Key
6 API_KEY = 'AIzaSyBv0pMIEZXngzg8kG6-S0XNYLilIMvfUVY'
7
8 # Connect to db
9 conn = sqlite3.connect('/home/lixsong/Desktop/kb03_yon/assets/Location.db')
10 cursor = conn.cursor()
11
12 # Create GMaps Client
13 gmaps = googlemaps.Client(key=API_KEY)
14
15 # Retrieve data from the db
16 cursor.execute("SELECT * FROM locations")
17 data = cursor.fetchall()
18
19 # Create a GMapsPlotter object
20 gmap = gmplot.GoogleMapPlotter(30.3164945, 78.03219179999999, 13, apikey=API_KEY)
21
22 # Iterate over the data and draw markers on the map
23 for row in data:
24     date, lat, lng, color = row
25     location = (lat, lng)
26
27     if color == 120.0:
28         marker_color = 'blue'
29     else:
30         marker_color = 'red'
31
32     # Add a marker to the map
33     gmap.marker(location[0], location[1], color=marker_color)
34
35 # Save the map to an HTML file
36 gmap.draw('map.html')
37
38 # Close db connection
39 conn.close()
40
```

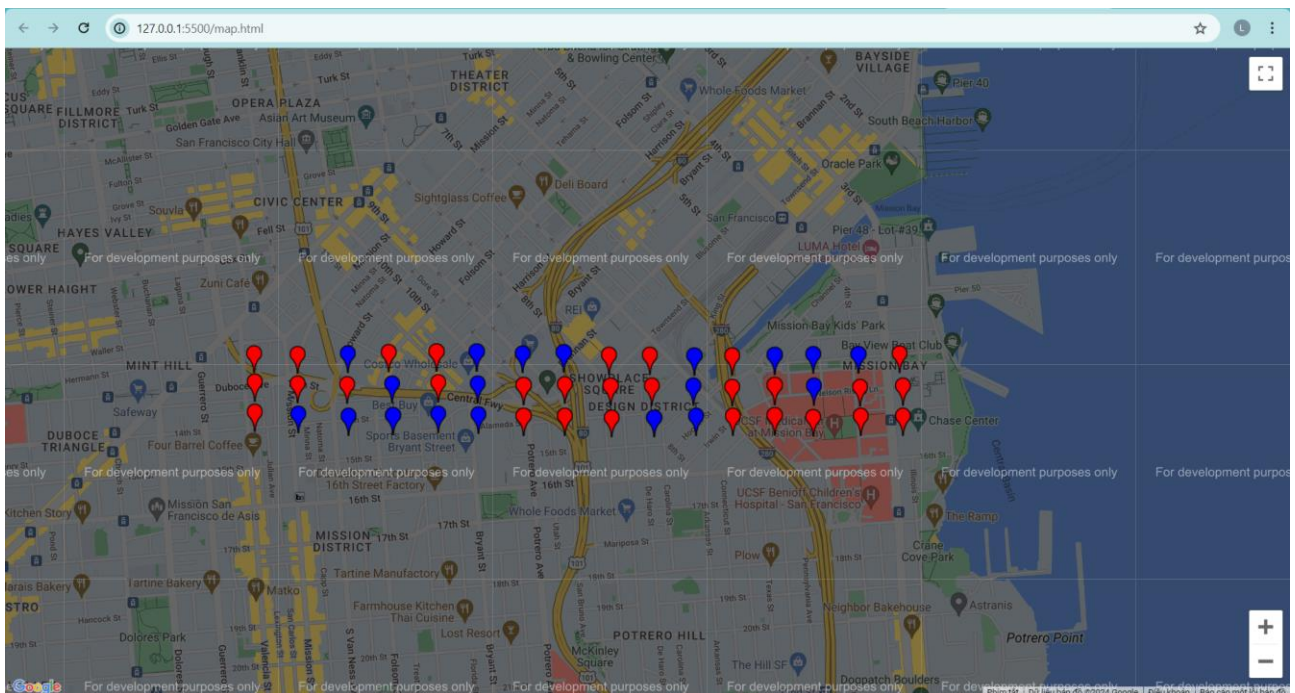
Thực hiện chạy code thì tạo ra file map.html

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
(my lib env) lixsong@long:~/Desktop$ python map.py
(my lib env) lixsong@long:~/Desktop$ ls
kb03_yon kb03_yon.apk map.html map.py my-evabs-key.keystore my-five-key.keystore my_lib_env my-release-key.keystore
(my lib env) lixsong@long:~/Desktop$
```

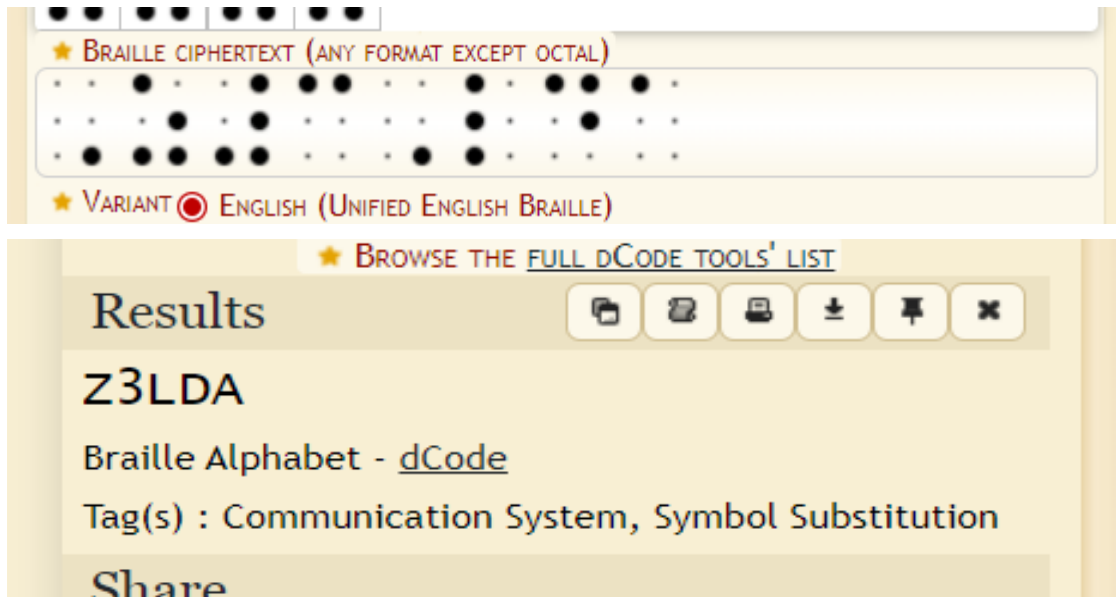
Mở file map.html



- Khi ta load các điểm lên thì ta được các điểm như trên
- Tới đây ta vẫn chưa sử dụng tới dữ kiện cột ngày trong database
- Khi lọc các điểm theo ngày thì ngày “02/08/2019” cho ta các điểm khá là thú vị



- Đây có vẻ như là “Braille”
 - Ta có thể coi điểm màu đỏ là những chấm đen còn màu xanh là chấm trắng
- Dùng <https://www.dcode.fr/braille-alphabet> để decode



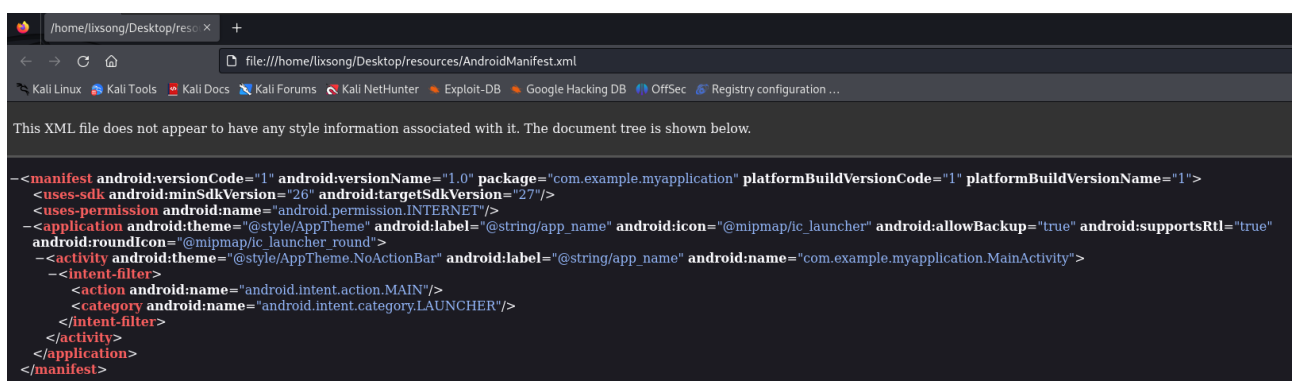
Flag: Z3LDA

4. Kịch bản 04. Điều tra trên tập tin ứng dụng thu được.

- Mô tả: Một ứng dụng thời tiết đơn giản có tính năng thu thập và hiển thị thông tin thời tiết.
- Tài nguyên: kb04_tianqi.apk
- Yêu cầu – Gợi ý: Xác định phiên bản Android đang chạy của ứng dụng. Sử dụng một số công cụ decompile apk như Jadx để phân tích code ứng dụng. Flag có định dạng CTF{...}

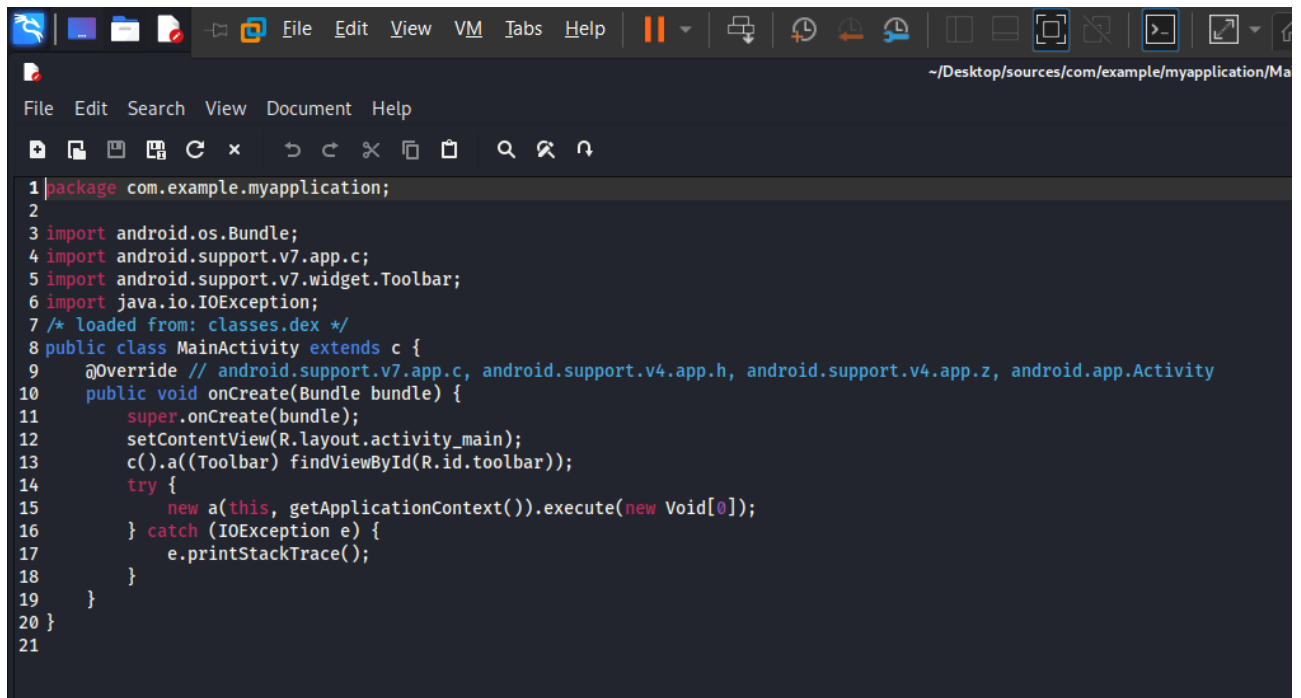
Đáp án:

Dùng jadx decompile file apk và đọc file AndroidManifest.xml để check version



Check thấy android:versionCode = 26 hoặc 27 nghĩa là android 8 hoặc android 8.1

Ta thấy chương trình có activity là com.example.myapplication.MainActivity



```
1 package com.example.myapplication;
2
3 import android.os.Bundle;
4 import androidx.appcompat.app.AppCompatActivity;
5 import androidx.appcompat.widget.Toolbar;
6 import java.io.IOException;
7 /* loaded from: classes.dex */
8 public class MainActivity extends AppCompatActivity {
9     @Override // androidx.appcompat.app.AppCompatActivity, androidx.appcompat.app.AppCompatActivity, androidx.appcompat.app.AppCompatActivity, androidx.appcompat.app.AppCompatActivity
10     public void onCreate(Bundle bundle) {
11         super.onCreate(bundle);
12         setContentView(R.layout.activity_main);
13         Toolbar toolbar = findViewById(R.id.toolbar);
14         try {
15             new a(this, getApplicationContext()).execute(new Void[0]);
16         } catch (IOException e) {
17             e.printStackTrace();
18         }
19     }
20 }
21
```

Đọc sơ qua thì đoạn mã cố gắng thực thi 1 tác vụ bằng cách sử dụng lớp “a”, truyền hoạt động “this” và ngữ cảnh “getApplicationContext()” làm đối số.

Sau đó dòng execute để khởi động thực thi tác vụ.

Ta thì thấy MainActivity.java gợi ý cho chúng ta về a.java, cái mà mở rộng AsyncTask, cho thấy quá trình sẽ diễn ra ở nền

```
import java.io.IOException;
import java.io.InputStreamReader;
import java.math.BigInteger;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.concurrent.TimeUnit;
import org.json.JSONException;
import org.json.JSONObject;

public final class a extends AsyncTask {
    private final MainActivity a;
    private final Context b;

    public a(MainActivity var1, Context var2) {
        this.a = var1;
        this.b = var2;
    }

    private String a() {
        String var3 = "";
        k var1 = k.j();
        if (var1.h == null) {
            var1.h = var1.g.a(var1);
        }

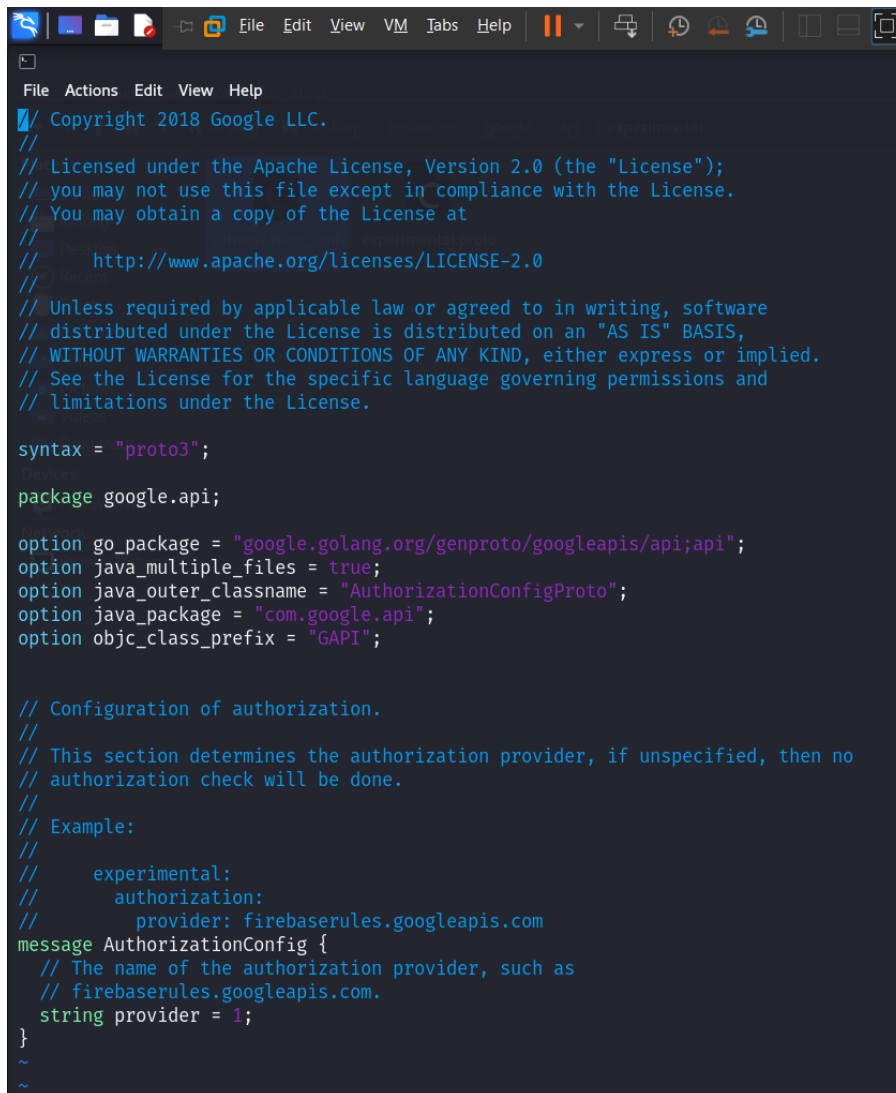
        g var5 = (g)var1.h;
        String var4 = null;
        String var12l = var3;
        String var2 = var4;

        label889: {
            label888: {
                IOException var10000;
                label895: {
                    Utils var6;
                    boolean var10001;
                    try {
                        var6 = new Utils;
                    } catch (IOException var120) {
                        var10000 = var120;
                        var10001 = false;
                        break label895;
                    }

                    var12l = var3;
                    var2 = var4;

                    try {
                        var6.<init>(this.b);
                    } catch (IOException var119) {
```

Tiếp tục ta sẽ thực hiện xem code thì ta thấy chương trình đang sử dụng api key của google thì đây là lab bị outdate nên dùng được nữa.



```
File Actions Edit View Help
// Copyright 2018 Google LLC.
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//      http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

syntax = "proto3";

package google.api;

option go_package = "google.golang.org/genproto/googleapis/api;api";
option java_multiple_files = true;
option java_outer_classname = "AuthorizationConfigProto";
option java_package = "com.google.api";
option objc_class_prefix = "GAPI";

// Configuration of authorization.
//
// This section determines the authorization provider, if unspecified, then no
// authorization check will be done.
//
// Example:
//
//     experimental:
//       authorization:
//         provider: firebase.rules.googleapis.com
message AuthorizationConfig {
    // The name of the authorization provider, such as
    // firebase.rules.googleapis.com.
    string provider = 1;
}
~
~
```

Sau khi phân tích code thì ta có attack vector

Bypass SSL Unpinning → Hook toString → Monitor toString

Thực hiện viết đoạn code để hooking.

```

JS baitap4.js > Java.perform() callback > implementation
1  Java.perform(function(){
2
3      // Step - 1
4
5      var array_list = Java.use("java.util.ArrayList");
6      var ApiClient = Java.use('com.android.org.conscrypt.TrustManagerImpl');
7
8      ApiClient.checkTrustedRecursive.implementation = function(a1, a2, a3, a4, a5, a6) {
9          var k = array_list.$new();
10         return k;
11     }
12
13
14     // Step - 2
15
16     console.log("Hooking Java");
17
18     const StringBuilder = Java.use('java.lang.StringBuilder');
19
20     StringBuilder.$init.overload('java.lang.String').implementation = function (arg) {
21         var partial = "";
22         var result = this.$init(arg);
23         console.log('new StringBuilder("' + result + '");');
24         return result;
25     }
26
27     console.log("Hooking new StringBuilder(java.lang.String)");
28
29
30     // Step - 3
31
32     StringBuilder.toString.implementation = function () {
33         var result = this.toString();
34         console.log('StringBuilder.toString(); => ' + result);
35         return result;
36     }
37
38     console.log("Hooking StringBuilder.toString() hooked");
39
40 }, 0);

```

Đoạn code trên là một ví dụ về việc sử dụng Frida, một công cụ được sử dụng để hook và thay đổi hoạt động của ứng dụng chạy trên Android. Dưới đây là giải thích từng bước của đoạn code:

1. Step 1:

- Sử dụng `Java.perform()` để thực hiện một hàm callback trên các đối tượng Java.
- Lấy ra các class cần hook: `ArrayList` và `TrustManagerImpl` từ gói `java.util` và `com.android.org.conscrypt` tương ứng.
- Ghi đè vào phương thức `checkTrustedRecursive` của `TrustManagerImpl`. Trong hàm ghi đè này, một đối tượng mới của `ArrayList` được tạo và trả về.

2. Step 2:

- In ra thông báo "Hooking Java".
- Lấy ra class `StringBuilder` từ `java.lang`.

- Ghi đè vào hàm khởi tạo của `StringBuilder` nhận một đối số là một chuỗi. Trong hàm ghi đè này, một chuỗi rỗng và một đối tượng `StringBuilder` mới được khởi tạo. Thông điệp "new StringBuilder()" cùng với đối số được in ra.

- In ra thông báo "Hooking new StringBuilder(java.lang.String)".

3. Step 3:

- Ghi đè vào phương thức `toString()` của `StringBuilder`. Trong hàm ghi đè này, giá trị của chuỗi `StringBuilder` được lấy ra và in ra thông điệp "StringBuilder.toString(); => [giá trị của chuỗi]".

- In ra thông báo "Hooking StringBuilder.toString() hooked".

Như vậy, đoạn code trên đang thực hiện việc ghi đè vào các phương thức của các class trong Java như `ArrayList` và `StringBuilder`, và sử dụng Frida để theo dõi và in ra các thông điệp khi các phương thức này được gọi trong ứng dụng Android.

Thực hiện chạy file frida -U -f com.example.myapplication -l /home/lixsong/Desktop/baitap4.js

```
StringBuilder.toString(); => {"type": "service_account", "project_id": "bsides-sf-ctf-2019", "private_key_id": "6dd7fc48a8b1d49edf7f03f74bc47713bec7d989", "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEVAIBADANBgkqhkiG9w0BAQEFAASCBKYYggSIAgEAAQIBAQChMa7X7qZ2Sec4\n5W41r+YXXJ3IwJ28fwyW0PZSoITb/1jQtcKk/tjP6ITr3H5MqKw6Vz/W6w7G5M\nnd21xMFqclwG8N7f+zhIK0XuvRBrS+cMEhw0RbHUBc03ZaghKffalThzPY4x2r\nhh/N8lUy14TBMAgaZC3H3pu19rTG4+ucX06pMnz3/EMzy0SmihR9Xb50KMV/Aq\n7N37BAW80SwRlT11aC8Ch18wddun6FKgVYmMcBwXSjropu8F6MR7vM3F0PEua\nYBZ12pxvVyc1i1MS0w1NtaA1ZBuhRenA1TFwOCt4rIS7BgV5qTS0K5t0W552tH\ntuzt/OVjAgMBAAGCggCAC9T230uU125W1huiXdtb7n/vU1w755yTvhf05Wkb+S\ni+19od5SEsrVh79sDR/n4NEktb1lH5Ulw3jgP08N34qARH0R2THJgxbpad231ry\nekuj+hu2atfA6AC00K+Du+7L7twrS6j8pgLR4d1L22ebvz2lMlMu10652pCIVMyM\nSMNS93vgfmiZotao73dIVwK8PUNRih0gqgLnub574dXkTfKcEpcedyza007Iy\nCoITUVYx007a7BT0Pvyq7m0eWh1SYUd1SoFcFAL1Cet2nm6+c9qRg00AwBC8V\ns9TmR0ePdlrUwPBAJk4YKov/Q0Uv29svE08VwK0Q0GTobS190xm2s+hmC\nPabZw765g/A0dPU4w4+/Ou/RUT+v30umsWf5N7KUXM98npu8LYoullit9+2V\nThTlbyTrnASvAm7hoCpeY80rboN1Vcxj9nFv652jw2CLZ0Lapl0XIVC0axRNDs\nCSyc2LDVVYj8abjzX0CFC05+QK0gC77kpSHM1w17p1m/T9aakdyVlzt1ZU\n+DTRNPXh/WMS6j9vdzGKq31lmiV/SSzVUfW0ax78Vzo1A169rUv/Dst8h7CpGd\nLCz850qg7ubbs7g00K/Zrw5QhV8doegZ0u0ntUfAVL7AnyKG7Vq25LheVmh7eZ\nm94MGMC20WdKgrFXfSWCGRGHM/wXKGV08JmtGwZcnjw+ACRZ7ZANpFqI7fH0Me\ngkyDwhjadvpiay87tP+ar1MvXteS1qCInBgfbcyKw+50Q/Lu1lXs9v0GwhYMQX\njCE56S0054IdIm0CInCvFzK0rStVkyKh1M2G05C1087H1YGS9DRBT5AoGAEWt\nFNrReDz9DapsanCncV9PoMie3LC3T54K713yRNNMs3sHlt7NqxtjyTE+K83/U\nMa+Pld0qYSHCMQ35Rhord7T0922D37grDY080yChLm96vasQTHeF7dDHSUN17i\nZr20a1uixp7/b0qT1P8978UvYj12lrfw+FBa9KcgrbCh8Vvy0ZlgayS07j0Rju\ndnVFr0u0ZRLKAShrIakVkyz2CaakV/1ZNI9mDztj6AIQ0r2UM1d0Qsmj204\n7VwArl3UuNhVxuhg2ocLCUthSYdExHfEujBu6uMQd3j0px87uoaggC7jw2anQC\n3ApTPc3478g/mho9S56QWw==\n-----END PRIVATE KEY-----\n","client_email": "weather-companion-service-acco@bsides-sf-ctf-2019.iam.gserviceaccount.com", "client_id": "116037946827001874660", "auth_uri": "https://accounts.google.com/o/oauth2/auth", "token\nuri": "https://oauth2.googleapis.com/token", "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs", "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/weather-companion-service-acco@40bsides-sf-ctf-2019.iam.gserviceaccount.com"}
```

- Chuỗi có định dạng JSON này là key được sử dụng để ủy quyền trong google cloud storage

- Và file JSON sẽ có định dạng như sau, khi đã sắp xếp lại

```
map.py map.html JS baitap4.js keyjson
1
2 {"type": "service_account",
3   "project_id": "bsides-sf-ctf-2019",
4   "private_key_id": "6dd7fc48a8b1d49edf7f03f74bc47713bec7d989",
5   "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEVAIBADANBgkqhkiG9w0BAQEFAASCBKYYggSIAgEAAQIBAQChMa7X7qZ2Sec4\n5W41r+YXXJ3IwJ28fwyW0PZSoITb/1jQtcKk/tjP6ITr3H5MqKw6Vz/W6w7G5M\nnd21xMFqclwG8N7f+zhIK0XuvRBrS+cMEhw0RbHUBc03ZaghKffalThzPY4x2r\nhh/N8lUy14TBMAgaZC3H3pu19rTG4+ucX06pMnz3/EMzy0SmihR9Xb50KMV/Aq\n7N37BAW80SwRlT11aC8Ch18wddun6FKgVYmMcBwXSjropu8F6MR7vM3F0PEua\nYBZ12pxvVyc1i1MS0w1NtaA1ZBuhRenA1TFwOCt4rIS7BgV5qTS0K5t0W552tH\ntuzt/OVjAgMBAAGCggCAC9T230uU125W1huiXdtb7n/vU1w755yTvhf05Wkb+S\ni+19od5SEsrVh79sDR/n4NEktb1lH5Ulw3jgP08N34qARH0R2THJgxbpad231ry\nekuj+hu2atfA6AC00K+Du+7L7twrS6j8pgLR4d1L22ebvz2lMlMu10652pCIVMyM\nSMNS93vgfmiZotao73dIVwK8PUNRih0gqgLnub574dXkTfKcEpcedyza007Iy\nCoITUVYx007a7BT0Pvyq7m0eWh1SYUd1SoFcFAL1Cet2nm6+c9qRg00AwBC8V\ns9TmR0ePdlrUwPBAJk4YKov/Q0Uv29svE08VwK0Q0GTobS190xm2s+hmC\nPabZw765g/A0dPU4w4+/Ou/RUT+v30umsWf5N7KUXM98npu8LYoullit9+2V\nThTlbyTrnASvAm7hoCpeY80rboN1Vcxj9nFv652jw2CLZ0Lapl0XIVC0axRNDs\nCSyc2LDVVYj8abjzX0CFC05+QK0gC77kpSHM1w17p1m/T9aakdyVlzt1ZU\n+DTRNPXh/WMS6j9vdzGKq31lmiV/SSzVUfW0ax78Vzo1A169rUv/Dst8h7CpGd\nLCz850qg7ubbs7g00K/Zrw5QhV8doegZ0u0ntUfAVL7AnyKG7Vq25LheVmh7eZ\nm94MGMC20WdKgrFXfSWCGRGHM/wXKGV08JmtGwZcnjw+ACRZ7ZANpFqI7fH0Me\ngkyDwhjadvpiay87tP+ar1MvXteS1qCInBgfbcyKw+50Q/Lu1lXs9v0GwhYMQX\njCE56S0054IdIm0CInCvFzK0rStVkyKh1M2G05C1087H1YGS9DRBT5AoGAEWt\nFNrReDz9DapsanCncV9PoMie3LC3T54K713yRNNMs3sHlt7NqxtjyTE+K83/U\nMa+Pld0qYSHCMQ35Rhord7T0922D37grDY080yChLm96vasQTHeF7dDHSUN17i\nZr20a1uixp7/b0qT1P8978UvYj12lrfw+FBa9KcgrbCh8Vvy0ZlgayS07j0Rju\ndnVFr0u0ZRLKAShrIakVkyz2CaakV/1ZNI9mDztj6AIQ0r2UM1d0Qsmj204\n7VwArl3UuNhVxuhg2ocLCUthSYdExHfEujBu6uMQd3j0px87uoaggC7jw2anQC\n3ApTPc3478g/mho9S56QWw==\n-----END PRIVATE KEY-----\n","client_email": "weather-companion-service-acco@bsides-sf-ctf-2019.iam.gserviceaccount.com", "client_id": "116037946827001874660", "auth_uri": "https://accounts.google.com/o/oauth2/auth", "token\nuri": "https://oauth2.googleapis.com/token", "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs", "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/weather-companion-service-acco@40bsides-sf-ctf-2019.iam.gserviceaccount.com"}
```

Thực hiện cài đặt account theo file key.json này nhưng ko thành công có thể do bài này bị outdate rồi.

```
lixsong@long: ~/Desktop
$ gcloud auth activate-service-account --key-file=key.json
ERROR: (gcloud.auth.activate-service-account) There was a problem refreshing your current auth tokens: ('invalid_grant: Invalid grant: account not found', ('error'
: 'invalid_grant', 'error_description': 'Invalid grant: account not found'))
Please run:

$ gcloud auth login

to obtain new credentials.

If you have already logged in with a different account, run:

$ gcloud config set account ACCOUNT

to select an already authenticated account to use.
lixsong@long:~/Desktop$
```

Sau khi tham khảo writeups trên mạng thì dưới đây là các bước nếu đăng nhập thành công.

```
$ gcloud auth activate-service-account --key-file=key.json
Activated service account credentials for: [weather-companion-service-acco@bsides-sf-ctf-

$ gsutil ls -p bsides-sf-ctf-2019 gs://weather-companion
gs://weather-companion/flag.txt
gs://weather-companion/weather.json
```

```
$ gsutil -m cp -r -p bsides-sf-ctf-2019 gs://weather-companion ./
```

Flag: CTF{buck3t_s3at5}