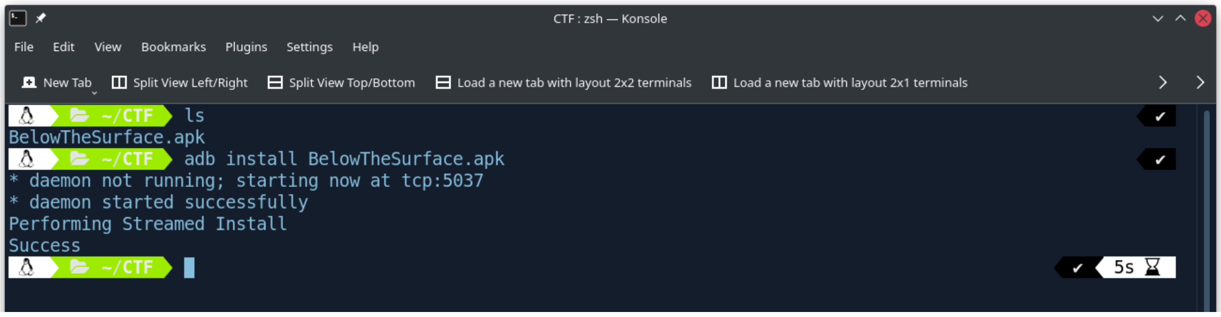
Below The Surface

Category : Reverse,Android

Difficulty : Easy

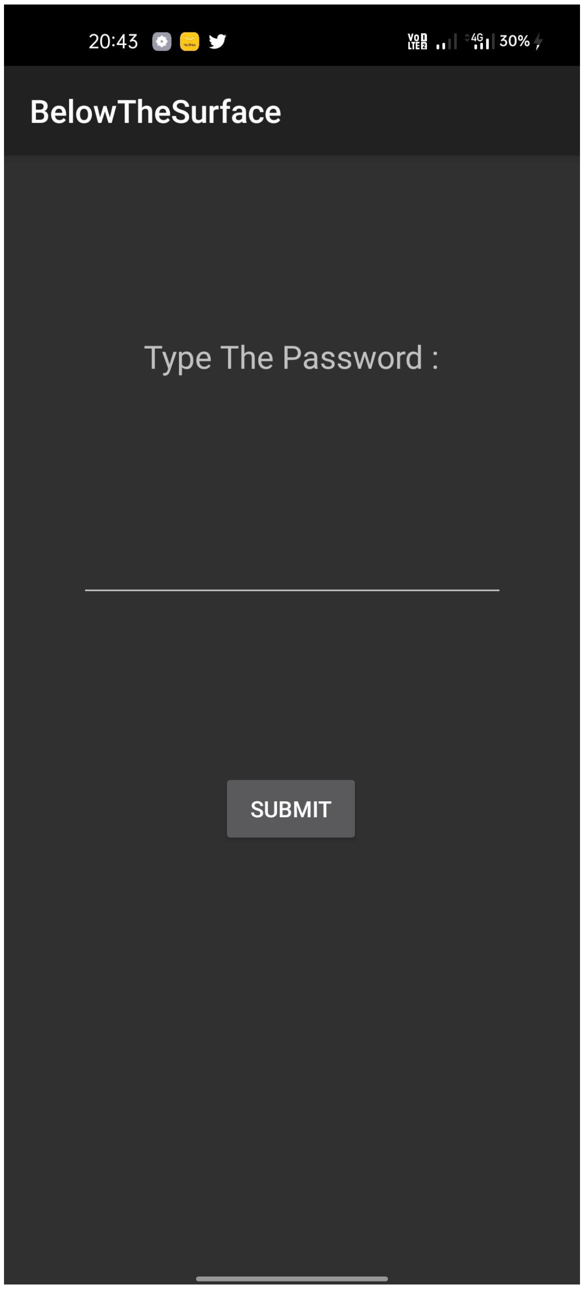
Solution

→ Install the app on your device using adb



CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

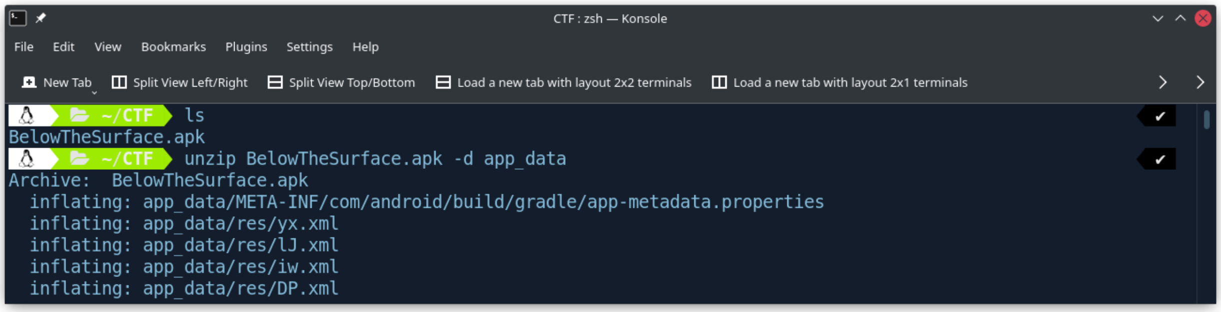
```
~/CTF  ls
BelowTheSurface.apk
~/CTF  adb install BelowTheSurface.apk
* daemon not running; starting now at tcp:5037
* daemon started successfully
Performing Streamed Install
Success
~/CTF 
```



So the app is asking for a password

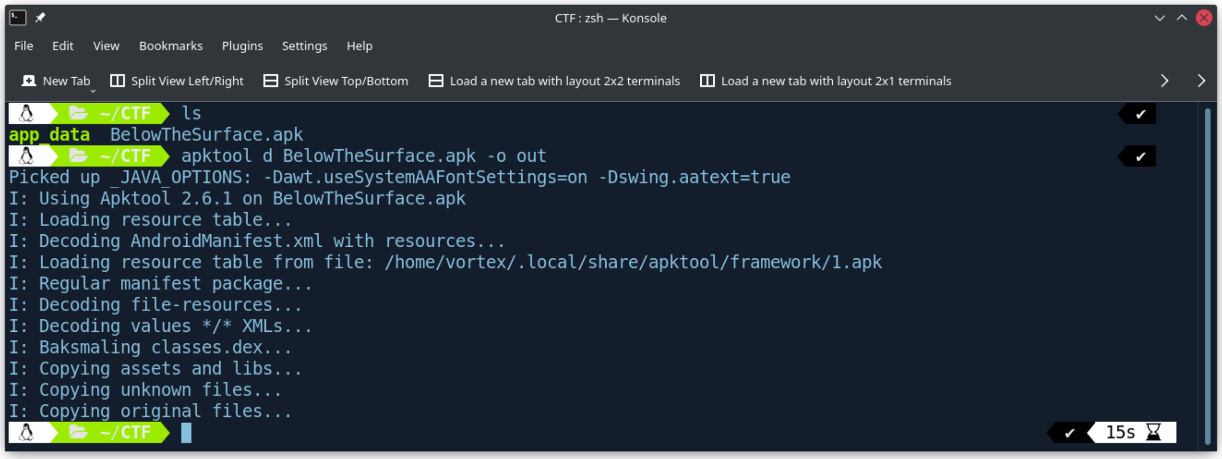
Now, Lets extract all the data from the app.

→ **Unzip** the apk file and store the output in **app_data** folder


CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

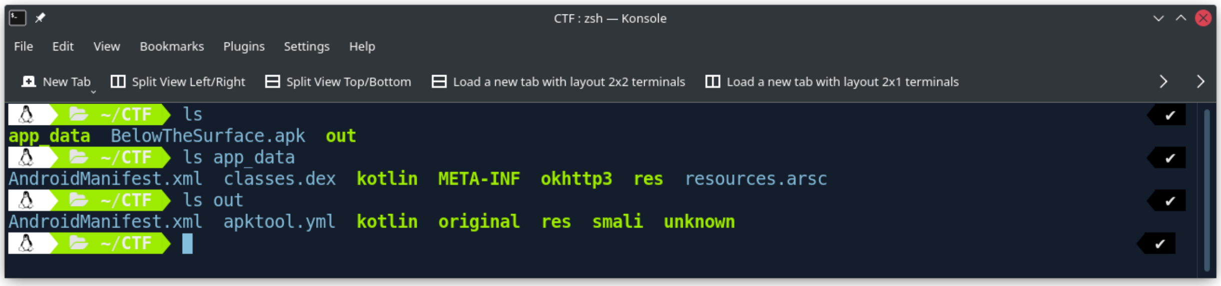
```
~/CTF  ls
BelowTheSurface.apk
~/CTF  unzip BelowTheSurface.apk -d app_data
Archive:  BelowTheSurface.apk
  inflating: app_data/META-INF/com/android/build/gradle/app-metadata.properties
  inflating: app_data/res/yx.xml
  inflating: app_data/res/lJ.xml
  inflating: app_data/res/iw.xml
  inflating: app_data/res/DP.xml
```

→ also use **apktool** to extract data from apk and store it in **out** folder (for viewing resource files)


CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

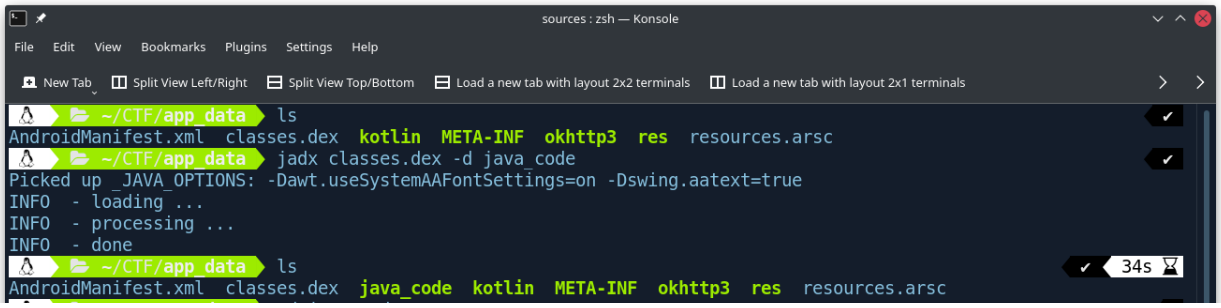
```
~/CTF  ls
app_data  BelowTheSurface.apk
~/CTF  apktool d BelowTheSurface.apk -o out
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
I: Using Apktool 2.6.1 on BelowTheSurface.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /home/vortex/.local/share/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
~/CTF                                                                    ✓  15s
```

Contents inside both the folders


CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

```
~/CTF  ls
app_data  BelowTheSurface.apk  out
~/CTF  ls app_data
AndroidManifest.xml  classes.dex  kotlin  META-INF  okhttp3  res  resources.arsc
~/CTF  ls out
AndroidManifest.xml  apktool.yml  kotlin  original  res  smali  unknown
~/CTF
```

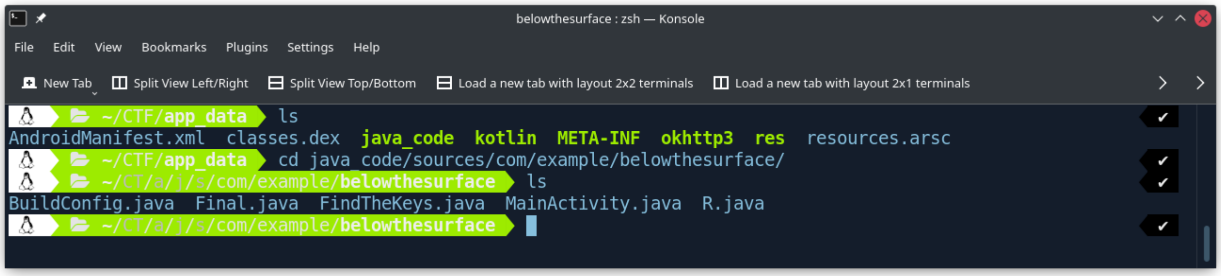
Now inside the **app_data** folder we have **classes.dex** file from which we can extract the java code using **jadx** tool


sources : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

```
~/CTF/app_data  ls
AndroidManifest.xml  classes.dex  kotlin  META-INF  okhttp3  res  resources.arsc
~/CTF/app_data  jadx classes.dex -d java_code
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
INFO  - loading ...
INFO  - processing ...
INFO  - done
~/CTF/app_data  ls                                                    34s
AndroidManifest.xml  classes.dex  java_code  kotlin  META-INF  okhttp3  res  resources.arsc
```

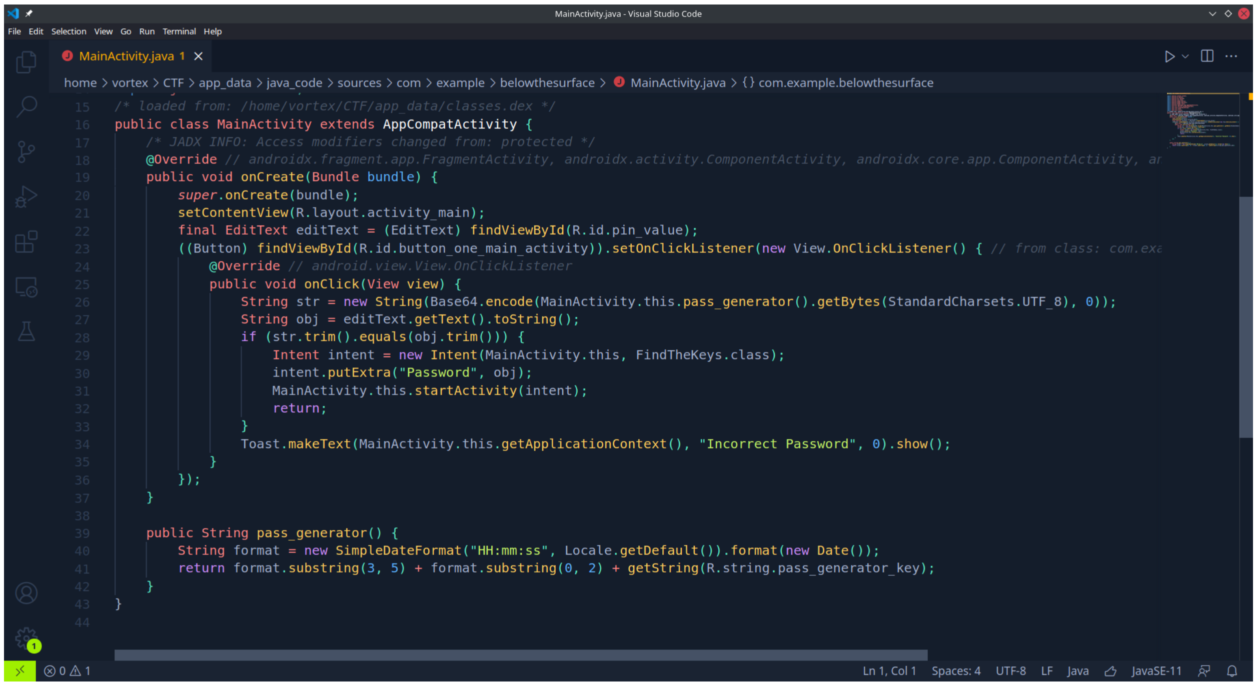
now go to the following directory and you would be able to see the java program files.

directory path → **java_code/sources/com/example/belowthesurface**


belowthesurface : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

```
~/CTF/app_data  ls
AndroidManifest.xml  classes.dex  java_code  kotlin  META-INF  okhttp3  res  resources.arsc
~/CTF/app_data  cd java_code/sources/com/example/belowthesurface/
~/CT/a/j/s/com/example/belowthesurface  ls
BuildConfig.java  Final.java  FindTheKeys.java  MainActivity.java  R.java
~/CT/a/j/s/com/example/belowthesurface
```

Lets examine the code of MainActivity.java file


MainActivity.java - Visual Studio Code
File Edit Selection View Go Run Terminal Help
MainActivity.java 1
home > vortex > CTF > app_data > java_code > sources > com > example > belowthesurface > MainActivity.java > {} com.example.belowthesurface
Ln 1, Col 1 Spaces: 4 UTF-8 LF Java JavaSE-11

```
15  /* loaded from: /home/vortex/CTF/app_data/classes.dex */
16  public class MainActivity extends AppCompatActivity {
17      /* JADX INFO: Access modifiers changed from: protected */
18      @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity, ar
19      public void onCreate(Bundle bundle) {
20          super.onCreate(bundle);
21          setContentView(R.layout.activity_main);
22          final EditText editText = (EditText) findViewById(R.id.pin_value);
23          ((Button) findViewById(R.id.button_one_main_activity)).setOnClickListener(new View.OnClickListener() { // from class: com.exa
24              @Override // android.view.View.OnClickListener
25              public void onClick(View view) {
26                  String str = new String(Base64.encode(MainActivity.this.pass_generator().getBytes(StandardCharsets.UTF_8), 0));
27                  String obj = editText.getText().toString();
28                  if (str.trim().equals(obj.trim())) {
29                      Intent intent = new Intent(MainActivity.this, FindTheKeys.class);
30                      intent.putExtra("Password", obj);
31                      MainActivity.this.startActivity(intent);
32                      return;
33                  }
34                  Toast.makeText(MainActivity.this.getApplicationContext(), "Incorrect Password", 0).show();
35              }
36          });
37      }
38
39      public String pass_generator() {
40          String format = new SimpleDateFormat("HH:mm:ss", Locale.getDefault()).format(new Date());
41          return format.substring(3, 5) + format.substring(0, 2) + getString(R.string.pass_generator_key);
42      }
43  }
44
```

Line 25 : If you look inside **onClick** method

Line 26 : there is a string variable **str** which basically takes value from pass_generator() function, base64 encode it and then store value in itself.

Line 27 : there is a **obj** variable which contains the user provided input

Line 28 : its **comparing str and obj variable to see if the password is correct or not**.

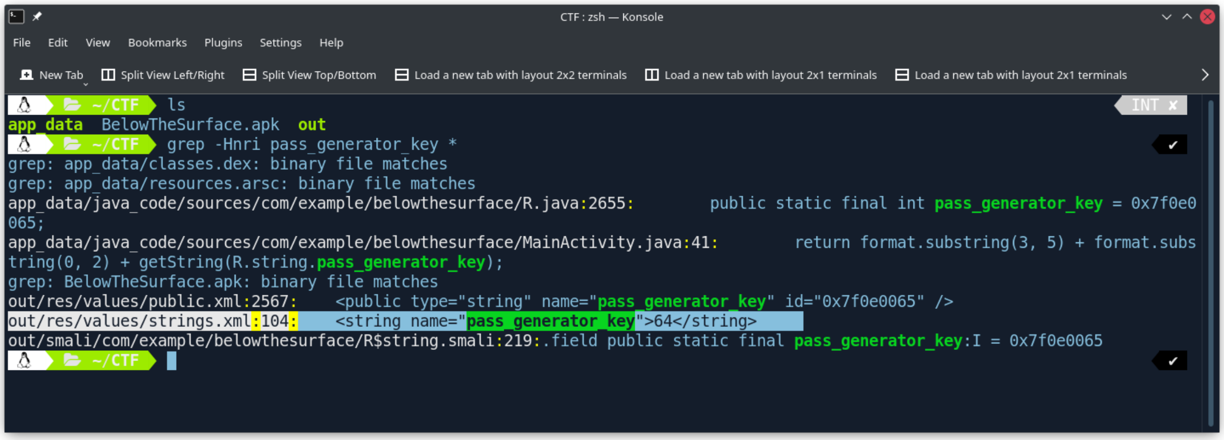
Line 39 : If we have a look inside the **pass_generator()** function

Line 40 :Then we could see that its getting the current time in HH:mm:ss format and storing it in format variable.

> **Time here is in 24 hour format.**

Line 41 : In the return statement we could see the that the format of the value returned is ⇒ **mm + HH + pass_generator_key**

Lets search for the pass_generator key



CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab
Split View Left/Right
Split View Top/Bottom
Load a new tab with layout 2x2 terminals
Load a new tab with layout 2x1 terminals
Load a new tab with layout 2x1 terminals

```
~/CTF  ls                                                                                              INT ✘
app_data  BelowTheSurface.apk  out
~/CTF  grep -Hnri pass_generator_key *                                                                    ✔
grep: app_data/classes.dex: binary file matches
grep: app_data/resources.arsc: binary file matches
app_data/java_code/sources/com/example/belowthesurface/R.java:2655:        public static final int pass_generator_key = 0x7f0e0
065;
app_data/java_code/sources/com/example/belowthesurface/MainActivity.java:41:        return format.substring(3, 5) + format.subs
tring(0, 2) + getString(R.string.pass_generator_key);
grep: BelowTheSurface.apk: binary file matches
out/res/values/public.xml:2567:    <public type="string" name="pass_generator_key" id="0x7f0e0065" />
out/res/values/strings.xml:104:    <string name="pass_generator_key">64</string>
out/smali/com/example/belowthesurface/R$string.smali:219:.field public static final pass_generator_key:I = 0x7f0e0065
~/CTF                                                                                                     ✔
```

command used → **grep -Hnri pass_generator_key ***

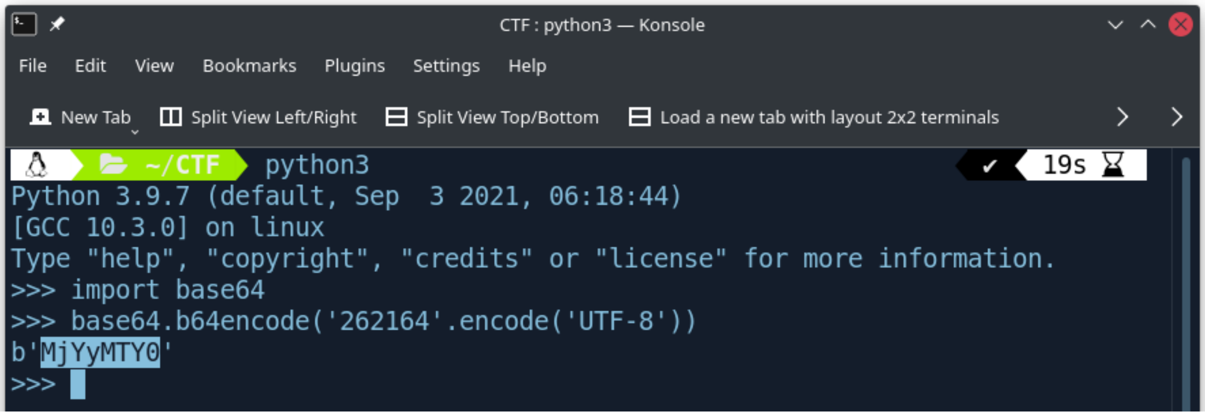
so the value of pass_generator key is 64

So the complete format of the string that pass_generator() function will return will be **mm + HH + 64** or **current_minutes+current_hour+64**

Now we could get the password

For example : time is → 21:26

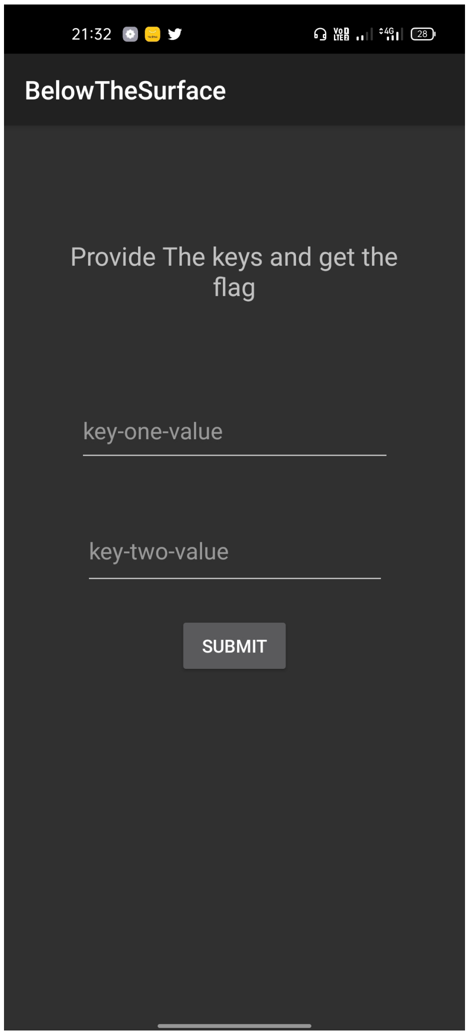
so password will be base64_encoded ('262164')


CTF : python3 — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals

```
~/CTF  python3                                        ✓  19s
Python 3.9.7 (default, Sep  3 2021, 06:18:44)
[GCC 10.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import base64
>>> base64.b64encode('262164'.encode('UTF-8'))
b'MjYyMTY0'
>>>
```

so **MjYyMTY0** will be the password according to current time.

so when you type the correct password you will be asked to provide two keys

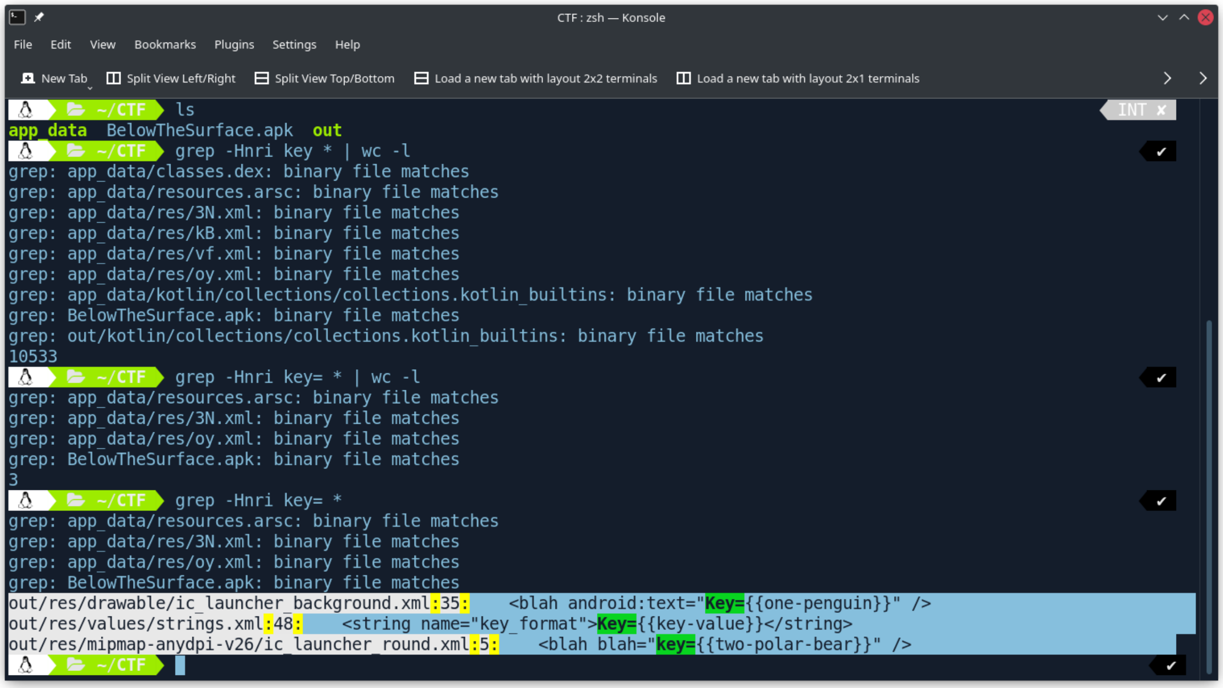


Provide The keys and get the flag

key-one-value

key-two-value

so let’s search for the keys


CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals

```
~/CTF  ls
app_data  BelowTheSurface.apk  out
~/CTF  grep -Hnri key * | wc -l
grep: app_data/classes.dex: binary file matches
grep: app_data/resources.arsc: binary file matches
grep: app_data/res/3N.xml: binary file matches
grep: app_data/res/kB.xml: binary file matches
grep: app_data/res/vf.xml: binary file matches
grep: app_data/res/oy.xml: binary file matches
grep: app_data/kotlin/collections/collections.kotlin_builtins: binary file matches
grep: BelowTheSurface.apk: binary file matches
grep: out/kotlin/collections/collections.kotlin_builtins: binary file matches
10533
~/CTF  grep -Hnri key= * | wc -l
grep: app_data/resources.arsc: binary file matches
grep: app_data/res/3N.xml: binary file matches
grep: app_data/res/oy.xml: binary file matches
grep: BelowTheSurface.apk: binary file matches
3
~/CTF  grep -Hnri key= *
grep: app_data/resources.arsc: binary file matches
grep: app_data/res/3N.xml: binary file matches
grep: app_data/res/oy.xml: binary file matches
grep: BelowTheSurface.apk: binary file matches
out/res/drawable/ic_launcher_background.xml:35:    <blah android:text="Key={{one-penguin}}" />
out/res/values/strings.xml:48:    <string name="key_format">Key={{key-value}}</string>
out/res/mipmap-anydpi-v26/ic_launcher_round.xml:5:    <blah blah="key={{two-polar-bear}}" />
~/CTF
```

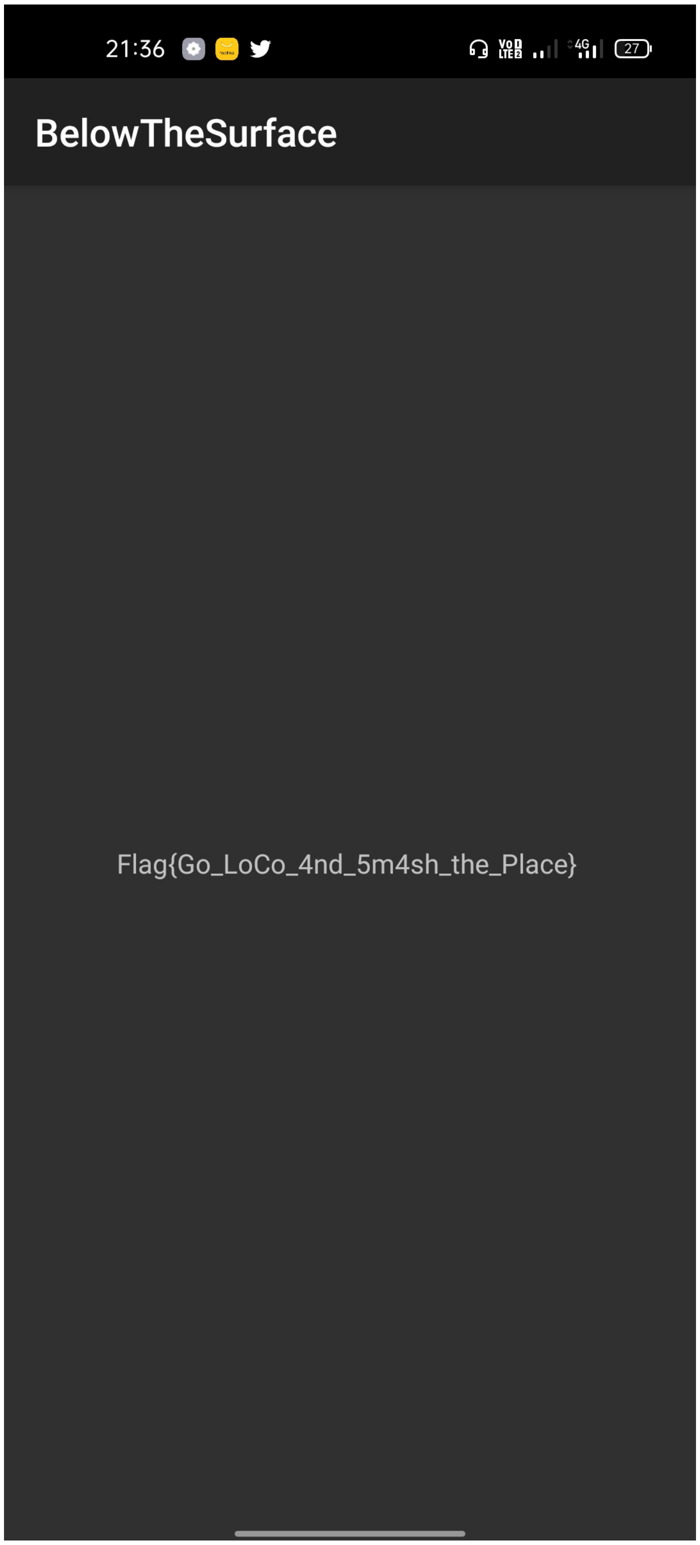
searching for the key values

so the two keys are

one-penguin

two-polar-bear

Provide the two keys and you will get the flag.


21:36

BelowTheSurface

Flag{Go_LoCo_4nd_5m4sh_the_Place}

Flag{Go_LoCo_4nd_5m4sh_the_Place}