

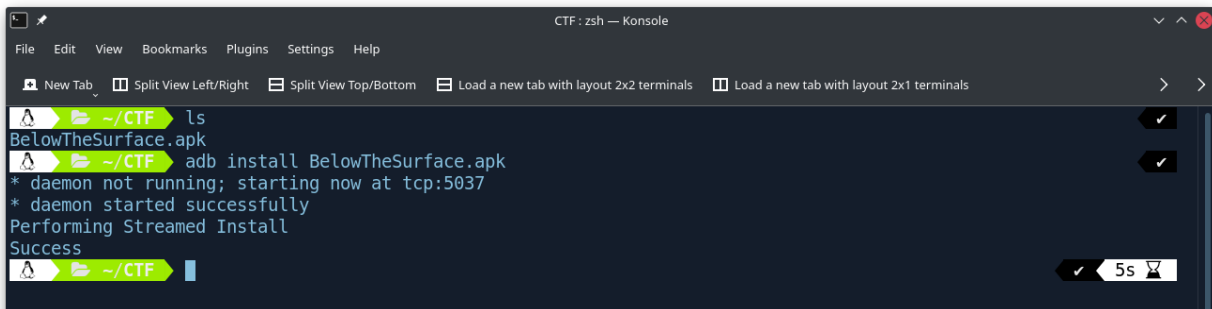
# Below The Surface

Category : Reverse,Android

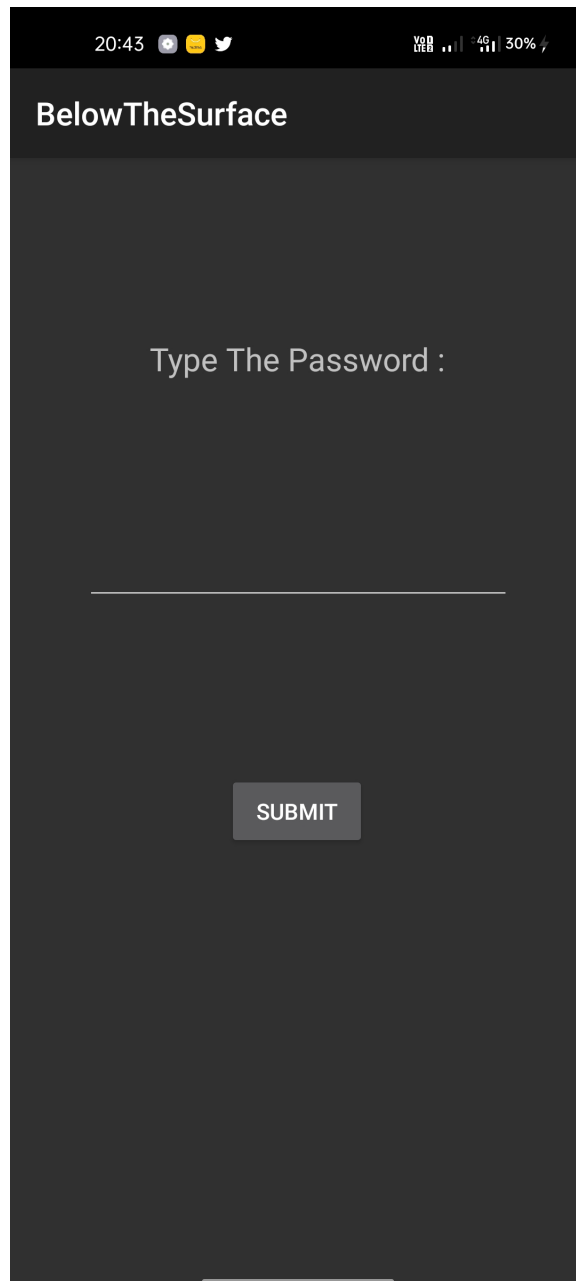
Difficulty : Easy

## 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.

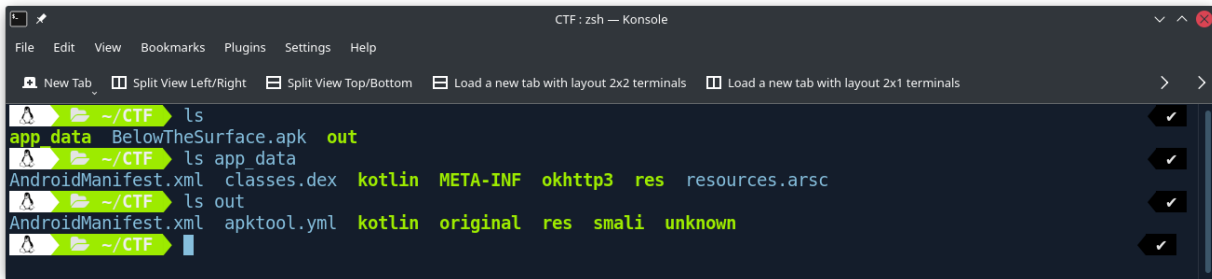
→ **Unzip** the apk file and store the output in **app\_data** folder

```
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)

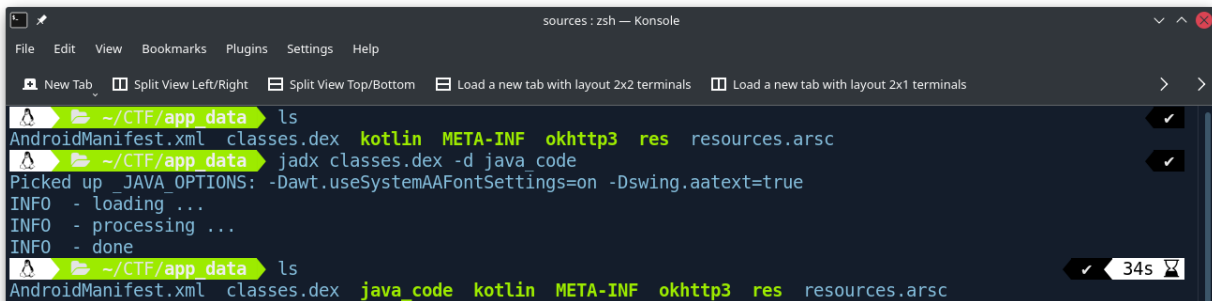
```
CTF : zsh — Konsole
File Edit View Bookmarks Plugins Settings Help
New Tab Split View Left/Right Split View Top/Bottom Load a new tab with layout 2x2 terminals Load a new tab with layout 2x1 terminals
~/CTF ls
app_data BelowTheSurface.apk
~/CTF apktool d BelowTheSurface.apk -o out
Picked up JAVA_OPTIONS: -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
```

Contents inside both the folders

A terminal window titled 'CTF : zsh — Konsole' showing a directory listing of the ~/CTF directory. The listing shows files: app\_data, BelowTheSurface.apk, out, AndroidManifest.xml, classes.dex, kotlin, META-INF, okhttp3, res, resources.arsc, apktool.yml, and smali. The user has navigated into the app\_data directory.

```
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
```

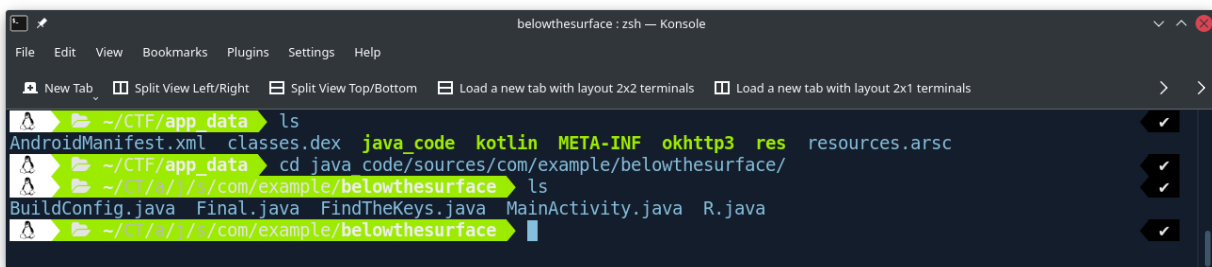
Now inside the **app\_data** folder we have **classes.dex** file from which we can extract the java code using **jadx** tool

A terminal window titled 'sources : zsh — Konsole' showing the execution of the jadx command to extract Java code from classes.dex. The output shows the command being run, the Java options picked up, and the successful extraction of the java\_code directory. The user then lists the contents of the app\_data directory, showing the newly created java\_code directory.

```
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
AndroidManifest.xml classes.dex java_code kotlin META-INF okhttp3 res resources.arsc
```

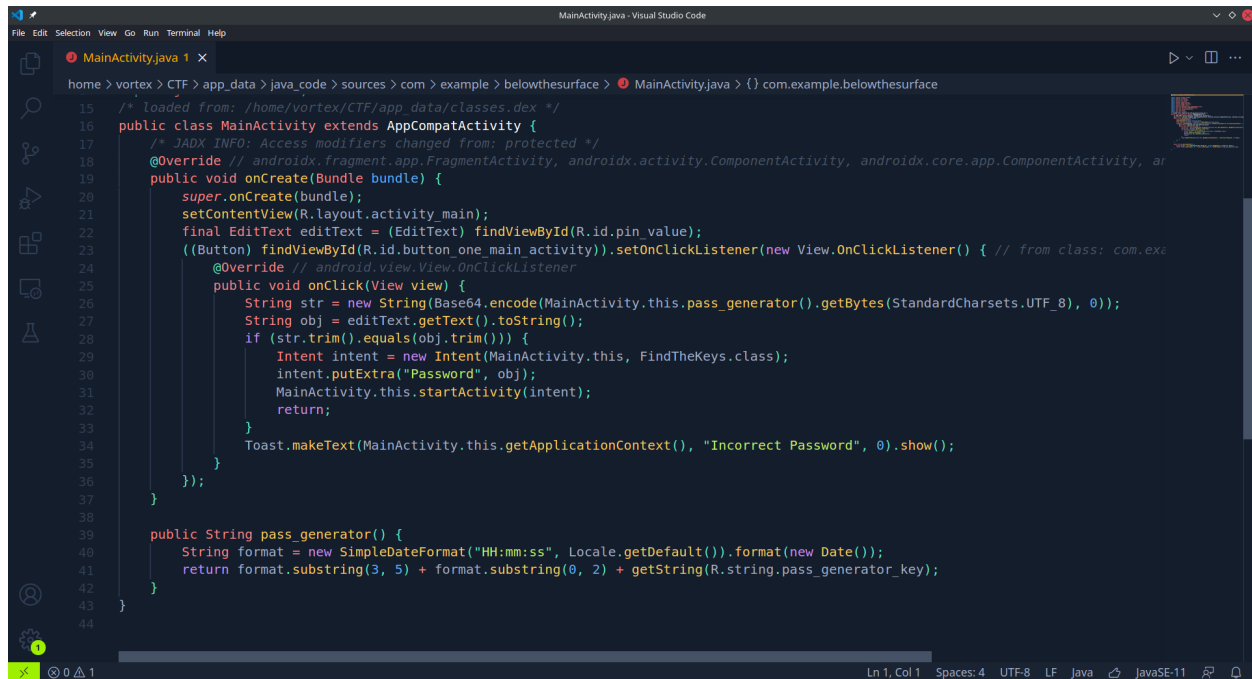
now go to the following directory and you would be able to see the java program files.

directory path → **java\_code/sources/com/example/belowthesurface**

A terminal window titled 'belowthesurface : zsh — Konsole' showing the directory listing of the extracted Java code. The user has navigated into the java\_code/sources/com/example/belowthesurface directory. The listing shows files: BuildConfig.java, Final.java, FindTheKeys.java, MainActivity.java, R.java, and R.java.

```
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/1/s/com/example/belowthesurface ls
BuildConfig.java Final.java FindTheKeys.java MainActivity.java R.java
~/CT/a/1/s/com/example/belowthesurface
```

Lets examine the code of MainActivity.java file

A screenshot of the Visual Studio Code editor showing the MainActivity.java file. The code is for an Android application. It starts with a package declaration 'com.example.belowthesurface' and imports 'AppCompatActivity'. The MainActivity class extends AppCompatActivity. It has an onCreate method that calls super.onCreate(bundle), setContentView(R.layout.activity\_main), and initializes an EditText and a Button. The Button has an OnClickListener that calls a onClick method. The onClick method gets the text from the EditText, compares it with a base64 encoded password generated by pass\_generator(), and either starts a new activity or shows a toast message. The pass\_generator method uses SimpleDateFormat to get the current time in HH:mm:ss format and concatenates it with a string resource. The status bar at the bottom shows 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'LF', 'Java', and '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.**

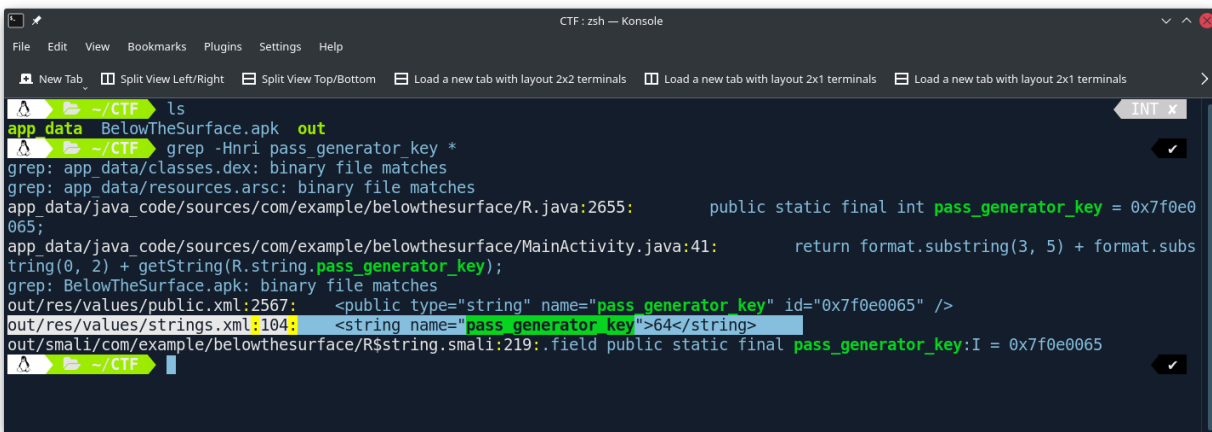
Line 39 : If we have a look inside the **pass\_generator()** function

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 that the format of the value returned is  $\Rightarrow$  **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
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 = 0x7f0e0065;
app_data/java_code/sources/com/example/belowthesurface/MainActivity.java:41: return format.substring(3, 5) + format.substring(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
```

command used  $\rightarrow$  **grep -Hnri pass\_generator\_key \***

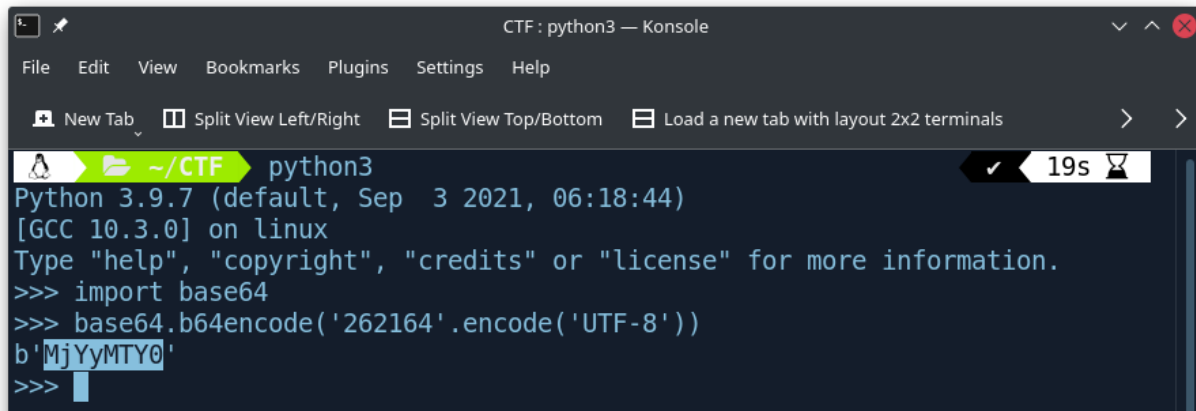
so the value of pass\_generator key is **64**

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  $\rightarrow$  21:26

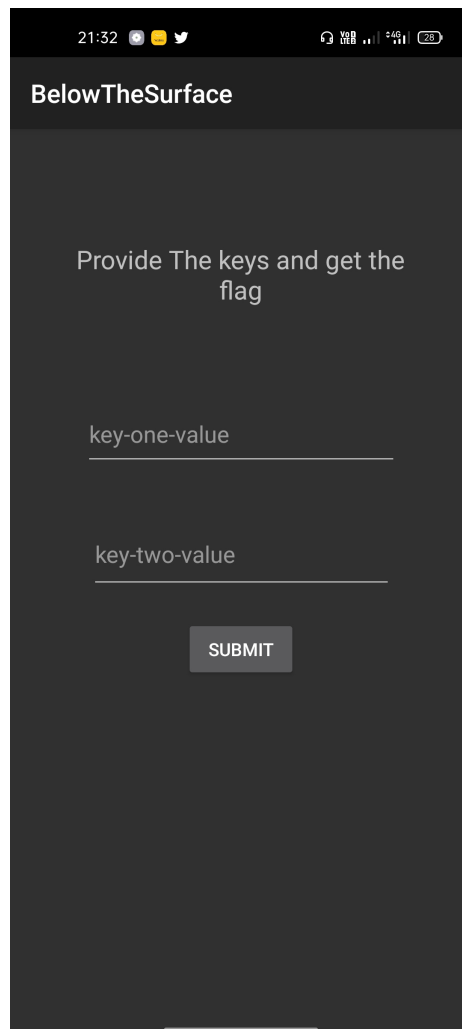
so password will be base64\_encoded ('262164')

A screenshot of a terminal window titled "CTF : python3 — Konsole". The window has a menu bar with "File", "Edit", "View", "Bookmarks", "Plugins", "Settings", and "Help". Below the menu bar is a toolbar with options like "New Tab", "Split View Left/Right", "Split View Top/Bottom", and "Load a new tab with layout 2x2 terminals". The terminal shows a prompt at "~/CTF" with the command "python3" entered. The output of the command is: "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." followed by a Python prompt ">>>". The user enters "import base64", followed by "base64.b64encode('262164'.encode('UTF-8'))", which outputs "b'MjYyMTY0'". The user then enters "MjYyMTY0" and the prompt ">>>" is shown again. A status bar at the bottom right shows a checkmark, "19s", and a clock icon.

```
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
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'
>>> MjYyMTY0
>>>
```

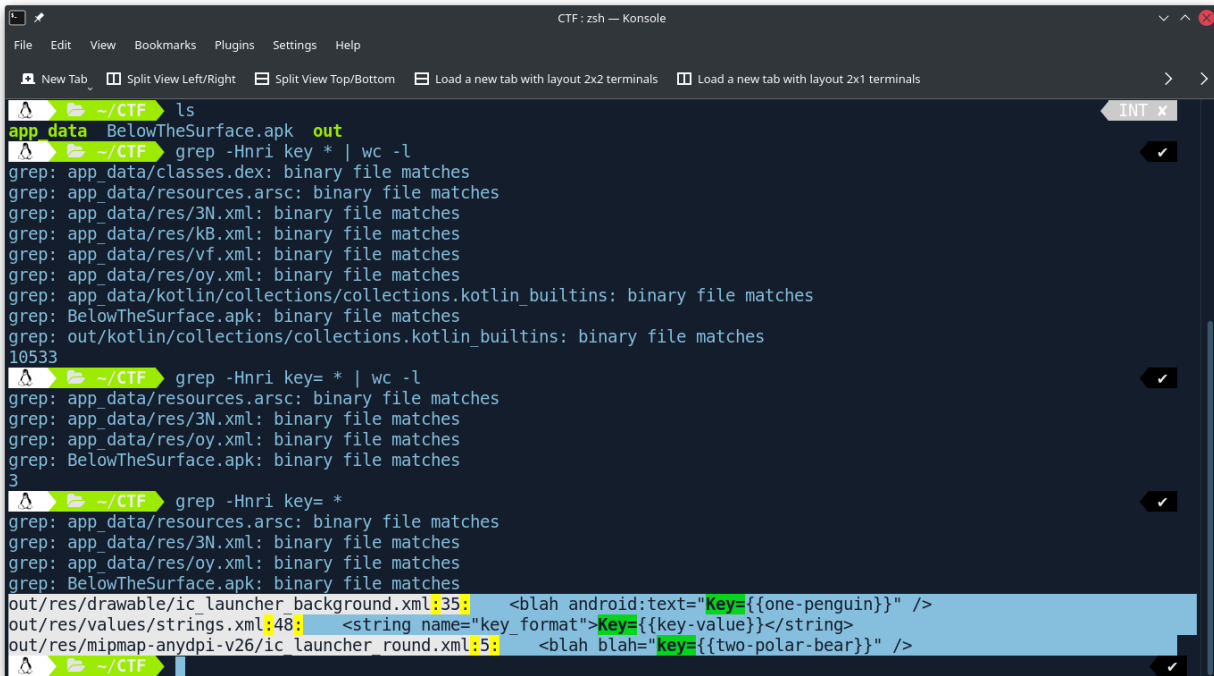
so **MjYyMTY0** will be the password according to current time.

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



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}}"/>
```

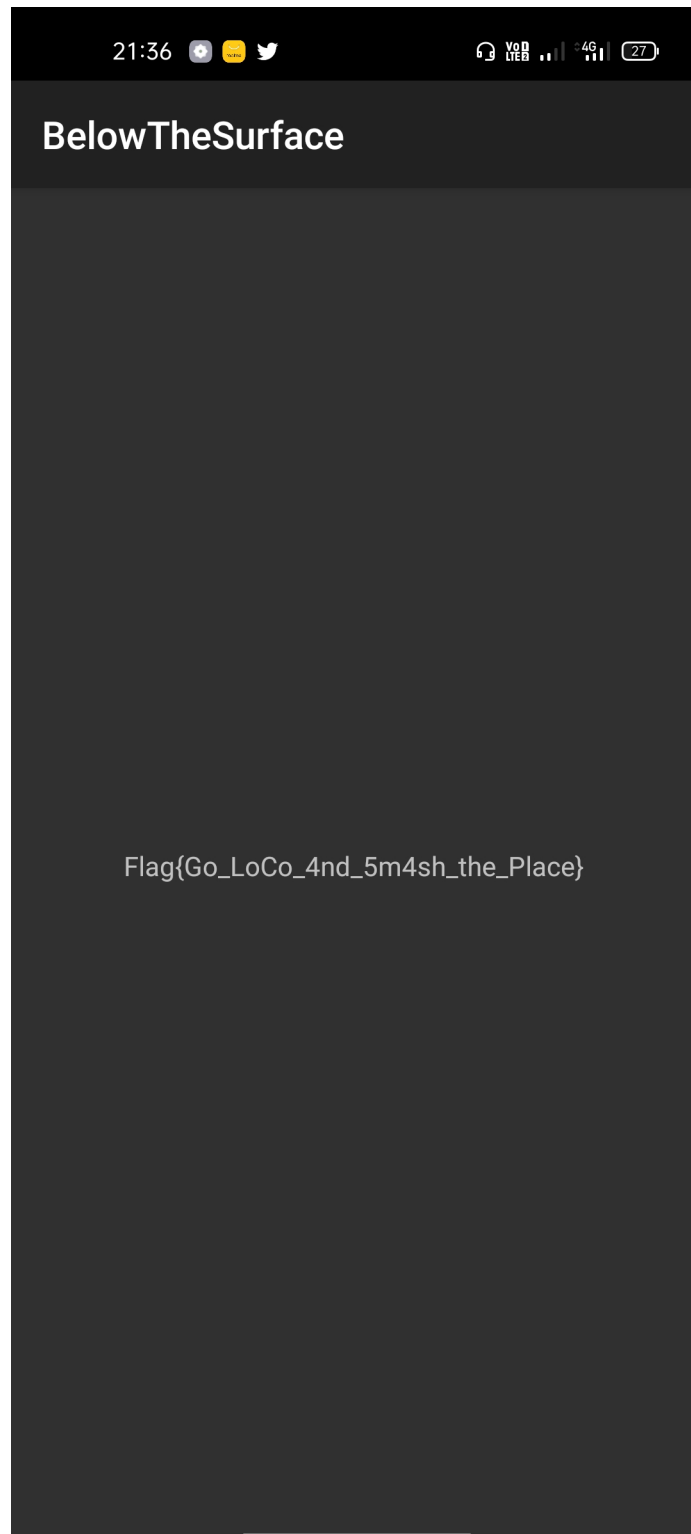
searching for the key values

so the two keys are

**one-penguin**

**two-polar-bear**

Provide the two keys and you will get the flag.



**Flag{Go\_LoCo\_4nd\_5m4sh\_the\_Place}**

