Vault 101 Writeup

Target:

> 22f0091b748e368f183d5eb58463789e Vault101-1.1-release.apk

Exploitation:

Use [Jadx]() to decompile apk file.

1. Analyse Binder Interface

There is Binder interface `IVault` used for interaction between `MainActivity` and `VaultService`.

```
/* renamed from: b.c.a.b */
public interface IVault extends IInterface {

    /* renamed from: b.c.a.b$a */
    public static abstract class Stub extends Binder implements IVault {

        /* renamed from: b.c.a.b$a$a */
        public static class Proxy implements IVault {
            ...
        }

        public Stub() {
            attachInterface(this, "com.sctf2020.vault101.IVault");
        }
        ...
    }
    ...
}
```

`VaultService` provides implementation of this interface by extending `IVault.Stub` class.

```
public class VaultService extends Service {
    /* renamed from: a */
    public IBinder binder = new IVault.Stub() /* renamed from:
com.sctf2020.vault101.VaultService$b */ {
        ...
    }

    public IBinder onBind(Intent intent) {
        return this.binder;
    }
    ...
}
```

Meanwhile, `MainActivity` binds to `VaultService` in `onCreate()` method and receiving `IVault` interface
in `onServiceConnected()` method.

```
public class MainActivity extends C0021e implements View.OnClickListener {
    /* renamed from: r */
    public volatile IVault vault;

    /* renamed from: s */
    public ServiceConnection connection = new ServiceConnection() /* renamed
from: com.sctf2020.vault101.MainActivity$a */ {
        public void onServiceConnected(ComponentName componentName, IBinder
iBinder) {
            vault = IVault.Stub.asInterface(iBinder);
            ...
        }

        public void onServiceDisconnected(ComponentName componentName) {...}
    }

    public void onCreate(Bundle bundle) {
        ...
        bindService(new Intent(this, VaultService.class), connection, 1);
    }
}
```

The main part is in `onClick()` method that invokes method `mo3486a()` with user's supplied text in EditText.

```
public void onClick(View view) {
    ...
    boolean a = this.vault.mo3486a(this.editText.getText().toString());
    ...
}
```

2. De-obfuscate strings

All string in the application are scrambled by `C0910c.m2623d()`.

```
Class.forName(C0910c.m2623d("*À/d\u000fÝ0Ç\u0007?ZÔ3Õ<h", 1914561889))
     .getMethod(
             C0910c.m2623d("![sp", -1321742225),
             (Class) Class.forName(C0910c.m2623d(">Ê•0••5Å±nè• Î•4ñ",
-1414862889))
                     .getDeclaredField(C0910c.m2623d("EÐ\u001d@", 828861746))
                     .get(null)
     ).invoke(null, 0);
```

Transformation is symmetric so we can apply it to scrambled strings to retrieve original strings.

```
Class.forName("java.lang.System")
     .getMethod(
             "exit",
             (Class<?>) Class.forName("java.lang.Integer")
                     .getDeclaredField("TYPE")
                     .get(null)
     ).invoke(null, 0);
```

Then remove Java reflection and get simple `System.exit(0);`.

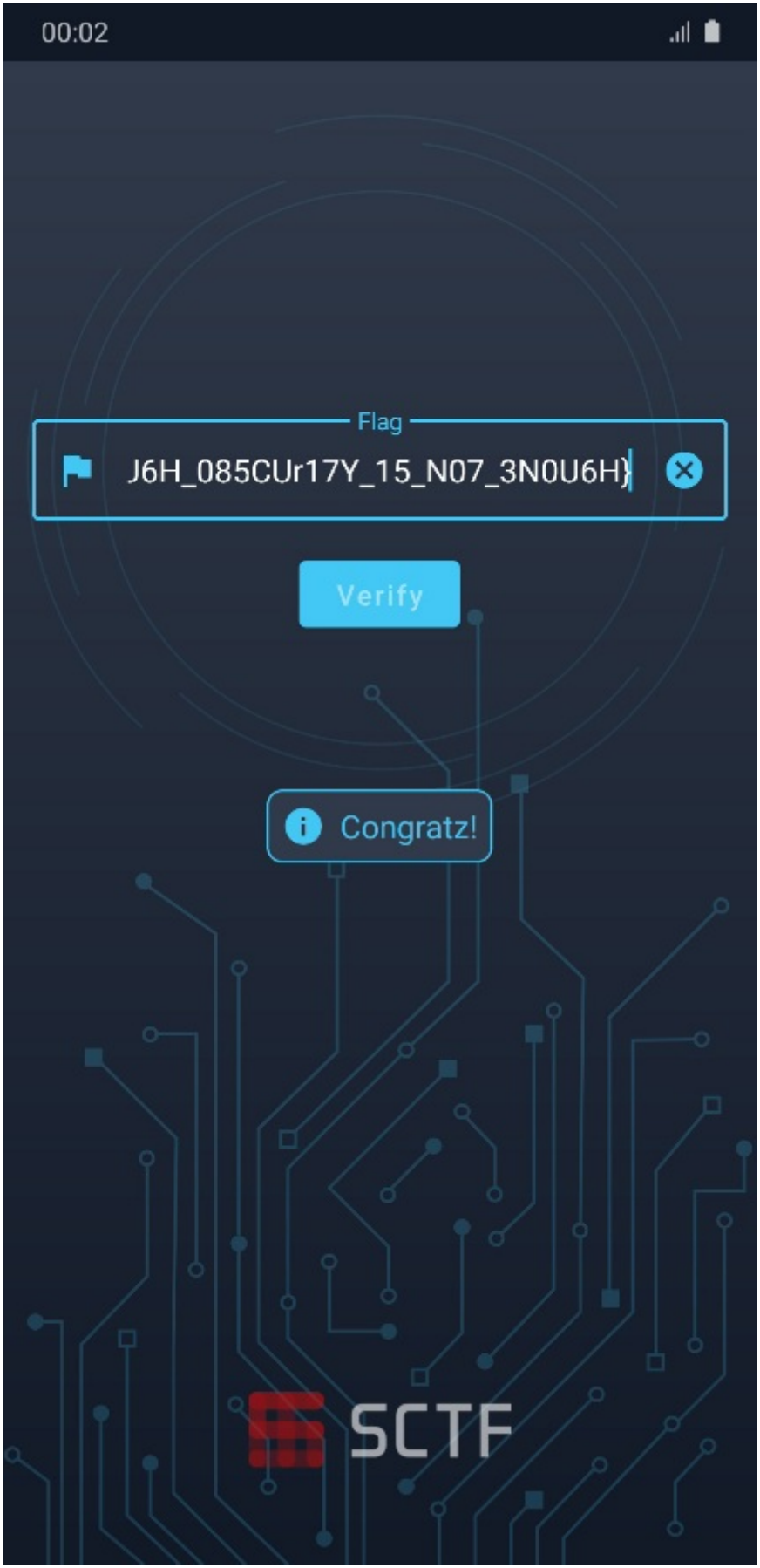
Remove all scrambled strings.
Turns out there is broken AES encryption in CBC mode with the IV equals to the key.

```
/* renamed from: b.c.a.a */
public class AES {
    /* renamed from: a */
    public static byte[] key;

    /* renamed from: b */
    public static byte[] encrypt(byte[] input) {
        try {
            final Cipher instance = Cipher.getInstance("AES/CBC/PKCS5Padding");

            final SecretKeySpec secretKeySpec = new SecretKeySpec(key, "AES");
            final IvParameterSpec ivParameterSpec = new IvParameterSpec(key);

            final int mode = Cipher.ENCRYPT_MODE;
            instance.init(mode, secretKeySpec, ivParameterSpec);
            return instance.doFinal(input);
        } catch (Throwable ignore) {
            throw new RuntimeException();
        }
    }
}
```

The AES key is initialized in `onCreate` function of `VaultService`.
It reads string array `R.array.kind_of_magic` from resources, decode Base64 of every item in array,
un-scramble and get first character to append to the key. Final AES key (and IV) is `PKnJ3BJqymGvKzG2`.

Finally, method `mo3486a()` of `VaultService` receives user's input from UI, encrypts it, encodes Base64, and then compares with value stored in resources `R.string.magic`.

4. PROFIT!
   Flag is `SCTF{53CUr17Y_7HrOU6H_O85CUr17Y_15_NO7_3NOU6H}`.


00:02
Flag
J6H_085CUr17Y_15_N07_3N0U6H}
Verify
Congratz!
SCTF