



2021 incognito CTF

Leap

강민송(부장님)

Write-up

2021-08-02



Notice

본 보고서는 2021 년 incognito CTF 에 출제된 문제 풀이 방법을 기술한 보고서입니다.

본 보고서의 내용은 저작권법에 의하여 보호받는 저작물로 그 저작권은 문제 출제자에게 있음을 알립니다. 따라서 보고서의 내용을 무단 복제 및 배포는 원칙적으로 금지합니다.

본 보고서를 외부에 유출하거나 무단으로 사용하였을 경우에는 관련 규정에 따른 처벌을 받게 됩니다.

목 차

1. 문제 정보	4
1.1 문제 이름.....	4
1.2 문제 기술.....	4
1.3 문제 분야.....	4
1.4 문제 의도.....	4
1.5 문제 힌트 및 주의사항	4
1.6 문제 정답	4
2. 풀이 방법	5
2.1 필요 기술.....	5
2.1 상세 풀이	5

1. 문제 정보

1.1 문제 이름

Leap

1.2 문제 기술

시간이 촉박한 당신에게 주는 문제!

아무리 급해도 둘다리는 두들겨보고 건너야겠지? ▶ Get the flag! ▶

1.3 문제 분야

① Reversing ② Pwn ③ Web ④ Forensics ⑤ Crypto ⑥ Network ⑦ **Mobile** ⑧ Misc
⑨ Programming ⑩ Recon

1.4 문제 의도

모바일의 구조를 파악하고 있는가?

Smali를 통해 소스코드를 응용할 수 있는가?

모바일 서명키를 생성 및 복구 시킬 수 있는가?

1.5 문제 힌트 및 주의사항

힌트 없음

1.6 문제 정답

INCO{I00K b3F0r3 YOU I3^p}

2. 풀이 방법

2.1 필요 기술

Decompile

JADX

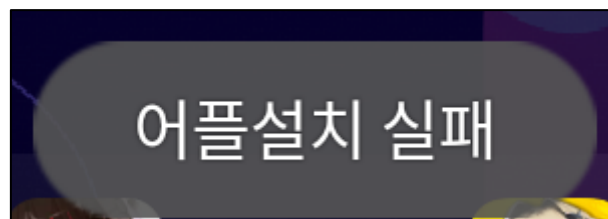
Bytecode-viewer

Smali 응용

Mobile 서명 및 적용

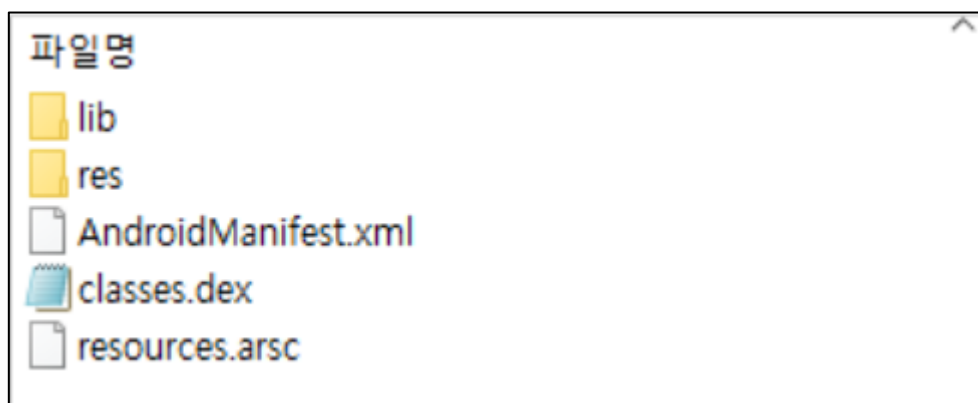
알고리즘 분석

2.2 상세 풀이





[그림 1] NOX 화면

APK 파일의 설치에 실패한다.



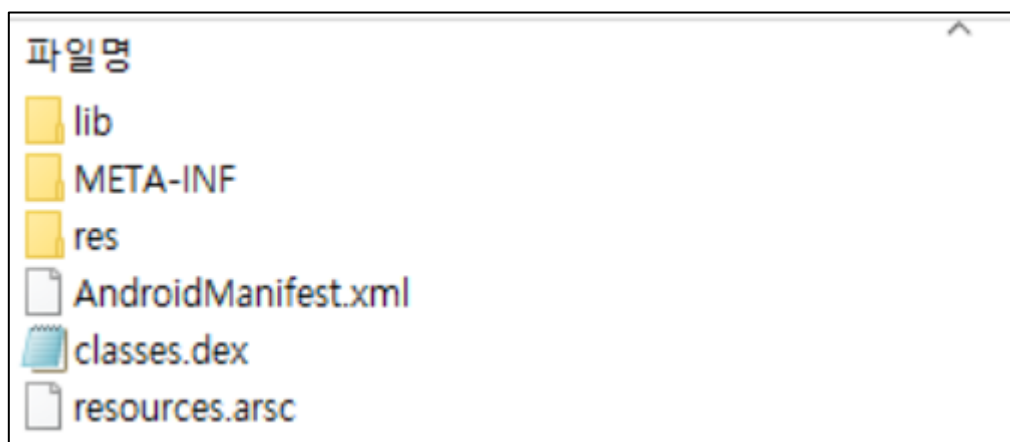
[그림 2] APK 파일 구조

APK 파일의 내부 구조 중에서 META-INF (서명키)가 없는 것을 확인할 수 있다.

이름	수정한 날짜
 INCO_Leap.apk	2021-08-02 오후 11:37
 my-release-key.keystore	2021-08-02 오후 11:37

[그림 3] 서명 및 적용

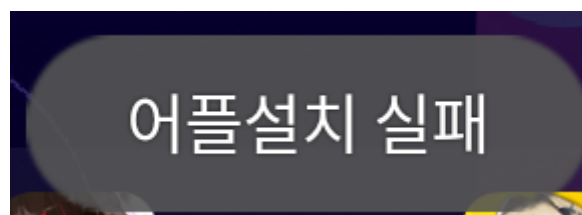
Apk 파일에 서명을 하고 적용시켜준 화면이다.



[그림 4] 서명 및 적용

아까와 달리 meta-info가 생겼음을 알 수 있다.

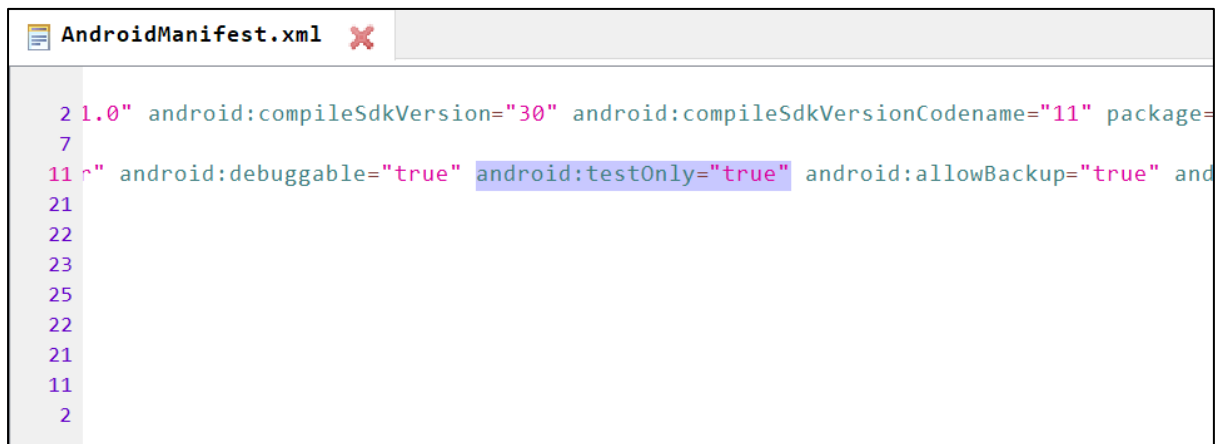
Nox에 설치해보자.



[그림 5] NOX 화면

서명을 했음에도 설치에 실패했다.

JADX를 통해 소스코드를 살펴보자.



[그림 6] AndroidManifest.xml

Manifest를 분석하던 도중, testOnly가 true로 되어있음을 발견했다.

Manifest를 수정하기 위해 apk 파일을 decompile 해준다.

이름	수정한 날짜
lib	2021-08-02 오후 11:44
original	2021-08-02 오후 11:44
res	2021-08-02 오후 11:44
smali	2021-08-02 오후 11:44
AndroidManifest.xml	2021-08-02 오후 11:44
apktool.yml	2021-08-02 오후 11:44

[그림 7] 디컴파일

Apktool로 decompile 한 파일의 내부구조다.

```

1  <?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android="http://schemas.android.com/apk/res/and
2      <application android:allowBackup="true" android:appComponentFactory="androidx.core.app.CoreComponentFactory"
3      android:debuggable="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name"
4      android:roundIcon="@mipmap/ic_launcher_round" android:supportsRtl="true" android:testOnly="false"
5      android:theme="@style/AppTheme">
6      <activity android:name="com.example.inco_leap.MainActivity">
7          <intent-filter>
8              <action android:name="android.intent.action.MAIN"/>
9              <category android:name="android.intent.category.LAUNCHER"/>
10         </intent-filter>
11     </activity>
12 </application>
13 </manifest>

```

[그림 8] AndroidManifest.xml 변조

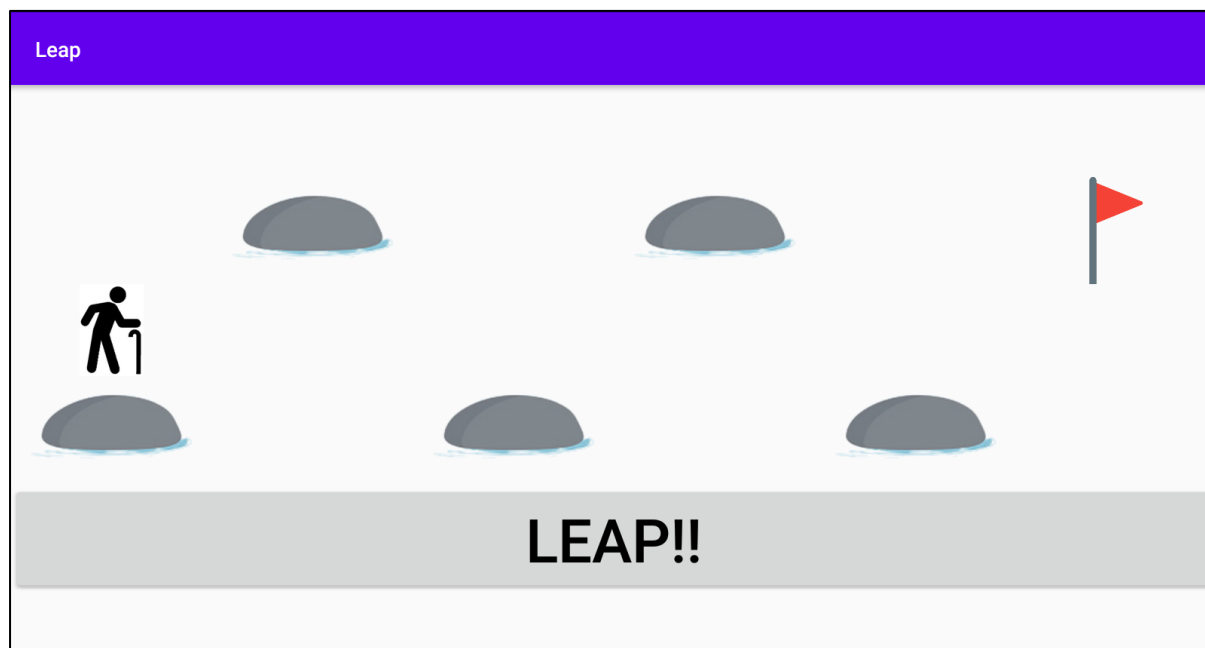
testOnly="true" ⇒ testOnly = "false" 로 바꿔준다.

build	2021-08-02 오후 11:50
lib	2021-08-02 오후 11:44
original	2021-08-02 오후 11:44
res	2021-08-02 오후 11:44
smali	2021-08-02 오후 11:44
AndroidManifest.xml	2021-08-02 오후 11:52
apktool.yml	2021-08-02 오후 11:44
my-release-key.keystore	2021-08-02 오후 11:53
repack.apk	2021-08-02 오후 11:53

[그림 9] 리패키징 및 서명

Decompile 되어있던 폴더를 repack 해준 뒤 서명 및 적용까지 마친 파일의 내부 현황이다.

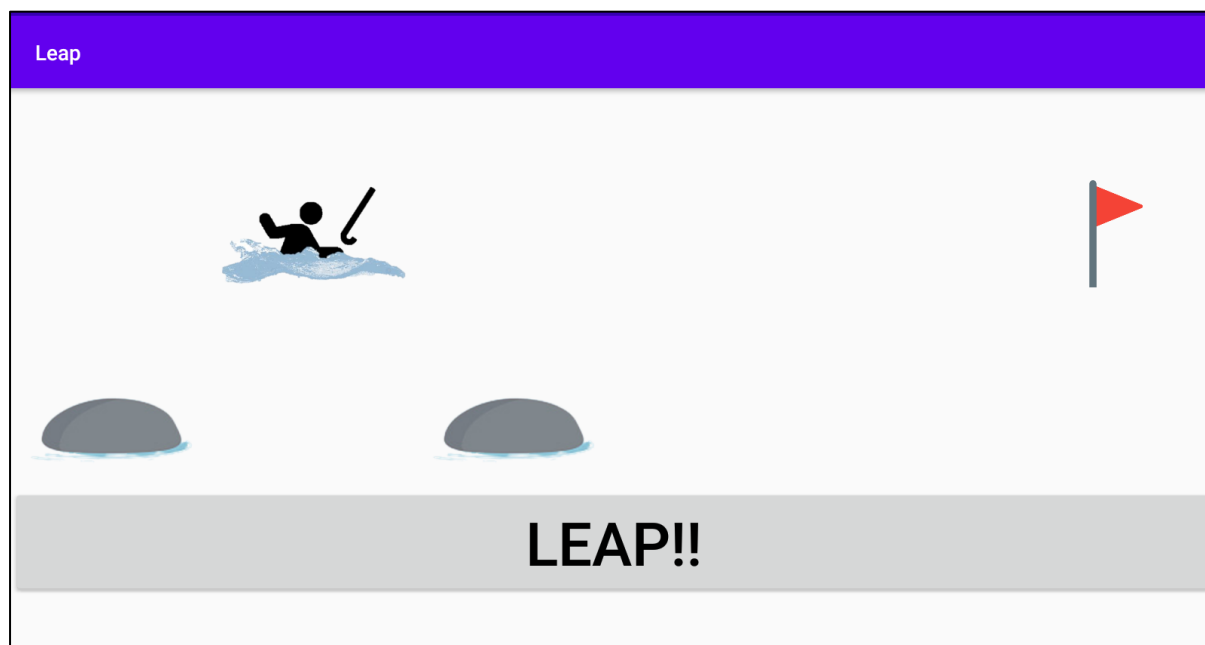
repack.apk를 nox에 설치해보자.



[그림 10] NOX 화면

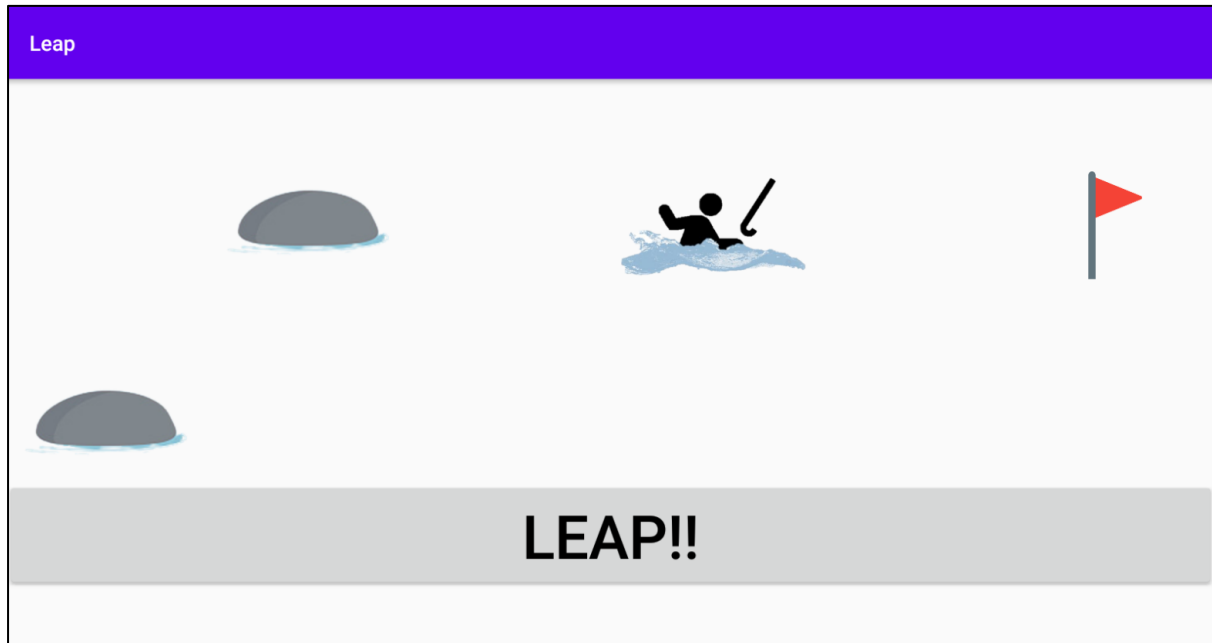
설치가 성공적으로 끝난 문제의 화면은 징검다리위에 사람이 있는 모습이다.

버튼을 눌러 FLAG에 도달하면 될 것 같다.



[그림 11] NOX 화면

버튼 클릭 시, 앞으로 전진하되 물에 빠져 그 이상 전진하지 못하고 있다.



[그림 12] NOX 화면

랜덤으로 생성되는 징검다리를 건너서 FLAG를 획득하여야 한다.

JADX를 통해 MAIN 코드를 분석해보자.

```

15 public class MainActivity extends AppCompatActivity {
    int cross = 1;
    int index = 0;
    Random random = new Random();

    public native String stringFromJNI();

    static {
17         System.loadLibrary("native-lib");
    }

    /* access modifiers changed from: protected */
    @Override // androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity, and
24 public void onCreate(Bundle savedInstanceState) {
25     super.onCreate(savedInstanceState);
26     setContentView(R.layout.activity_main);
29     ((Button) findViewById(R.id.leap)).setOnClickListener(new View.OnClickListener() {
        /* class com.example.inco_leap.MainActivity$AnonymousClass1 */

31         public void onClick(View view) {
32             MainActivity.this.func();
33             MainActivity.this.cross();
        }
    });
}

```

[그림 13] MainActivity.java

```
public void onClick(View view) {
```

```
    MainActivity.this.func();
```

```
    MainActivity.this.cross();
```

```
    // 버튼 클릭 시 함수 실행
```

```

152 public void func() {
153     ImageView stone2 = (ImageView) findViewById(R.id.stone2);
154     ImageView stone3 = (ImageView) findViewById(R.id.stone3);
155     ImageView stone4 = (ImageView) findViewById(R.id.stone4);
156     ImageView stone5 = (ImageView) findViewById(R.id.stone5);
157     int nextInt = this.random.nextInt(4);
157     this.index = nextInt;
181     if (nextInt == 0) {
160         stone2.setVisibility(0);
161         stone3.setVisibility(4);
162         stone4.setVisibility(4);
163         stone5.setVisibility(4);
164         Log.d("tttt", String.valueOf(this.index));
181     } else if (nextInt == 1) {
167         stone2.setVisibility(4);
168         stone3.setVisibility(0);
169         stone4.setVisibility(4);
170         stone5.setVisibility(4);
171         Log.d("tttt", String.valueOf(this.index));
181     } else if (nextInt == 2) {
174         stone2.setVisibility(4);
175         stone3.setVisibility(4);
176         stone4.setVisibility(0);
177         stone5.setVisibility(4);
178         Log.d("tttt", String.valueOf(this.index));
181     } else if (nextInt == 3) {
181         stone2.setVisibility(4);
182         stone3.setVisibility(4);
183         stone4.setVisibility(4);
184         stone5.setVisibility(0);

```

[그림 14] MainActivity.java

Func() 함수의 코드를 살펴보면,

```

ImageView stone2 = (ImageView) findViewById(R.id.stone2);           // 선언
this.index = nextInt;                                              // [그림13] 변수 index = 0;
    if (nextInt == 0) {                                           // random 값이 0일 때
        stone2.setVisibility(0);                                   // visible
        stone3.setVisibility(4);
        stone4.setVisibility(4);
        stone5.setVisibility(4);                                   // invisible

```

```
int nextInt = this.random.nextInt(4);
```

//stone의 num이 5까지 존재하고, random은 4까지 (0-3) 돌리는 것으로 보아,
[그림 10]의 첫 돌을 제외한 나머지 4개의 돌이 랜덤하게 나타난다는 코드다.

// 중복된 코드가 너무 많아 스킵된 cross()를 복구해서 코드를 분석해보자.

```

21 public void cross() {
22     ImageView var1 = (ImageView) this.findViewById(2131230951);
23     ImageView var2 = (ImageView) this.findViewById(2131230952);
24     ImageView var3 = (ImageView) this.findViewById(2131230953);
25     ImageView var4 = (ImageView) this.findViewById(2131230954);
26     ImageView var5 = (ImageView) this.findViewById(2131230895);
27     ImageView var6 = (ImageView) this.findViewById(2131230896);
28     ImageView var7 = (ImageView) this.findViewById(2131230897);
29     ImageView var8 = (ImageView) this.findViewById(2131230898);
30     ImageView var9 = (ImageView) this.findViewById(2131230899);
31     ImageView var10 = (ImageView) this.findViewById(2131230900);
32     ImageView var11 = (ImageView) this.findViewById(2131230851);
33     ImageView var12 = (ImageView) this.findViewById(2131230852);
34     ImageView var13 = (ImageView) this.findViewById(2131230853);
35     ImageView var14 = (ImageView) this.findViewById(2131230854);
36     TextView var15 = (TextView) this.findViewById(2131230841);
37     Log.d("ttttcross", String.valueOf(this.cross));
38     int var16 = this.cross;
39     if (var16 != 0) {
40         label61: {
41             label62: {
42                 label63: {
43                     if (var16 != 1) {
44                         if (var16 != 2) {
45                             if (var16 != 3) {
46                                 if (var16 != 4) {
47                                     if (var16 != 5) {
48                                         return;
49                                     }

```

[그림 15] MainActivity.java

Bytecode-viewer로 확인한 cross() 함수다.

```
int var16 = this.cross; // [그림13] 변수 cross = 1;
```

```
if (var16 != 0) {
```

```
    label61: {
```

```
        label62: {
```

```
            label63: {
```

```
                if (var16 != 1) {
```

```
                    if (var16 != 2) {
```

```
                        if (var16 != 3) {
```

```
                            if (var16 != 4) {
```

```
                                if (var16 != 5) {
```

```
                                    // if 문은 총 6개로, cross = 5일 때 [그림16]실행
```

```

123   if (var4.getVisibility() == 0) {
124       var9.setVisibility(0);
125       var5.setVisibility(4);
126       var6.setVisibility(4);
127       var7.setVisibility(4);
128       var8.setVisibility(4);
129       return;
130   }
131
132   if (var4.getVisibility() == 4) {
133       var9.setVisibility(4);
134       var5.setVisibility(4);
135       var6.setVisibility(4);
136       var7.setVisibility(4);
137       var8.setVisibility(4);
138       var14.setVisibility(0);
139       this.cross = 0;
140       return;
141   }
142 }
143
144     var5.setVisibility(4);
145     var6.setVisibility(4);
146     var7.setVisibility(4);
147     var8.setVisibility(4);
148     var9.setVisibility(4);
149     var10.setVisibility(0);
150     var15.setText(this.stringFromJNI());
151 } else {
152     var11.setVisibility(4);
153     var12.setVisibility(4);
154     var13.setVisibility(4);
155     var14.setVisibility(4);
156     var5.setVisibility(0);
157     ++this.cross;
158 }

```

[그림 16] MainActivity.java

```

if (var4.getVisibility() == 0) {           // visible 일 때
    var9.setVisibility(0);                 // visible
    var5.setVisibility(4);
    var6.setVisibility(4);
    var7.setVisibility(4);
    var8.setVisibility(4);                 // invisible
    return;

    // 다른 if 문에서는 cross++ 가 있었는데, cross=4 일때 없음.
    // cross = 5 의 조건문 실행이 아예 안됨.
} if (var4.getVisibility() == 4) {        // invisible 일 때

```

```
        var9.setVisibility(4);
        var5.setVisibility(4);
        var6.setVisibility(4);
        var7.setVisibility(4);
        var8.setVisibility(4);          // invisible
        var14.setVisibility(0);
        this.cross = 0;                 // cross = 0 으로 돌아가라
        return;
    }
    // cross = 5 일 때
    var5.setVisibility(4);
    var6.setVisibility(4);
    var7.setVisibility(4);
    var8.setVisibility(4);
    var9.setVisibility(4);
    var10.setVisibility(0);
    var15.setText(this.stringFromJNI()); // 정답을 보여준다.

    // smali 코드에서 변수 cross 를 찾아 cross=1 ⇒ cross=5 로 변경해주자.
```

```
32
33     .line 16
34     const/4 v0, 0x1
35
36     iput v0, p0, Lcom/example/leap/MainActivity;-->cross:I
37
38     .line 17
39     const/4 v0, 0x0
40
41     iput v0, p0, Lcom/example/leap/MainActivity;-->index:I
42
```

[그림 17] MainActivity.smali

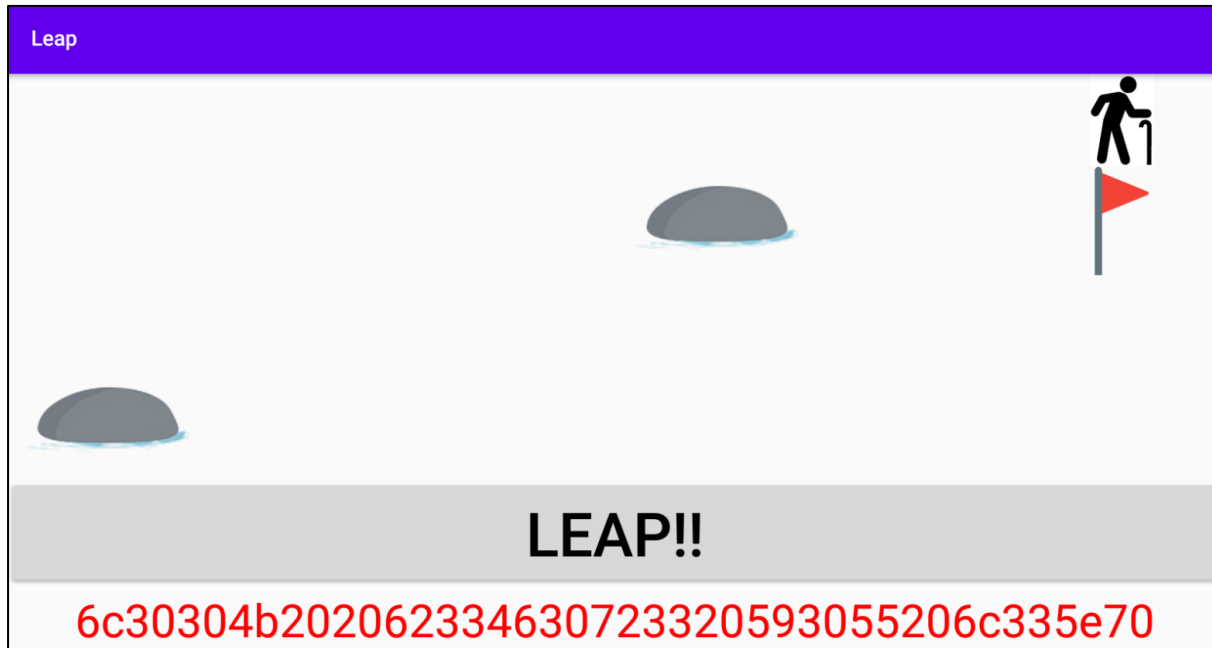
cross = 1 변수의 값을 찾았다.

```
32
33     .line 16
34     const/4 v0, 0x5
35
36     iput v0, p0, Lcom/example/leap/MainActivity;-->cross:I
37
38     .line 17
39     const/4 v0, 0x0
40
41     iput v0, p0, Lcom/example/leap/MainActivity;-->index:I
42
```

[그림 18] MainActivity.smali 변조

cross = 5 로 변조해준다.

변조된 smali 파일을 apktool 을 이용해 repack 해주고, 서명 및 적용까지 끝낸 apk 파일을 nox 에 설치한다.



[그림 19] KEY

기존 NOX 에 설치되었던 Leap.apk 파일을 삭제해주고 새로 repack 한 apk 파일을 설치해준다. 첫번째 돌다리에 있던 사람이 leap 버튼을 한번 누르니 smali 코드 상에서 cross 변수에 줬던 5의 값이 적용되어 flag 에 한번에 도달했다. 아래에 키 값이 나왔다. 해당 키 값을 hex 디코딩을 돌려주어야 제대로 된 값이 나올 것 같다.



[그림 19] KEY

```
>> 6c30304b202062334630723320593055206c335e70
```

```
>> I00K b3F0r3 YOU I3^p
```

KEY 값은 `INCO{I00K b3F0r3 YOU I3^p}` 이다.