

Android Mobile Pentest 101

© tsug0d, September 2018

Bài 7.5 – Lab: Practice Frida

Mục tiêu: Luyện kỹ năng hook và dịch ngược

Bài tập

- Cài file challenge2_release.apk (challenge này từ h1-702 2018 CTF):
https://github.com/tsug0d/AndroidMobilePentest101/blob/master/lab/frida_lab/challenge2_release.apk
- Dịch ngược phần mềm, sử dụng frida để crack pin, lấy flag

Hết bài!

Slide tiếp theo là gợi ý để giải bài, trước khi coi thì nhớ tryhard nhé

(No Pain, No Gain!)

Gợi ý

- Brute-force pin
- `public native void resetCoolDown();`

Bế tắc hử? Ok bài giải đây

- Full script:

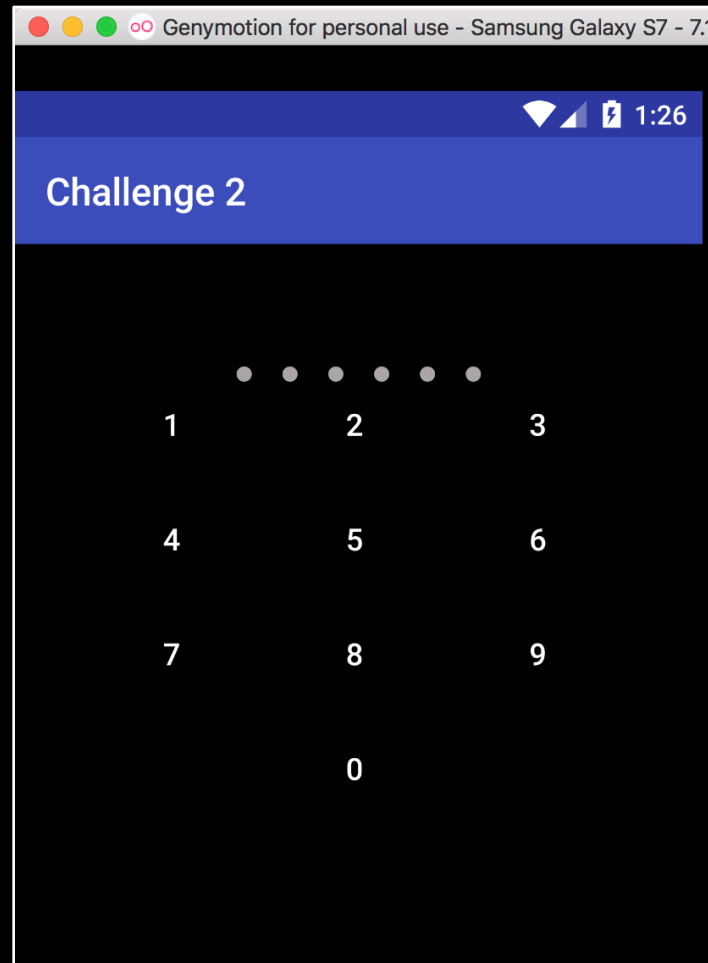
https://github.com/tsug0d/AndroidMobilePentest101/blob/master/lab/frida_lab/lab_frida.py

- Giải thích:

(các slide còn lại)

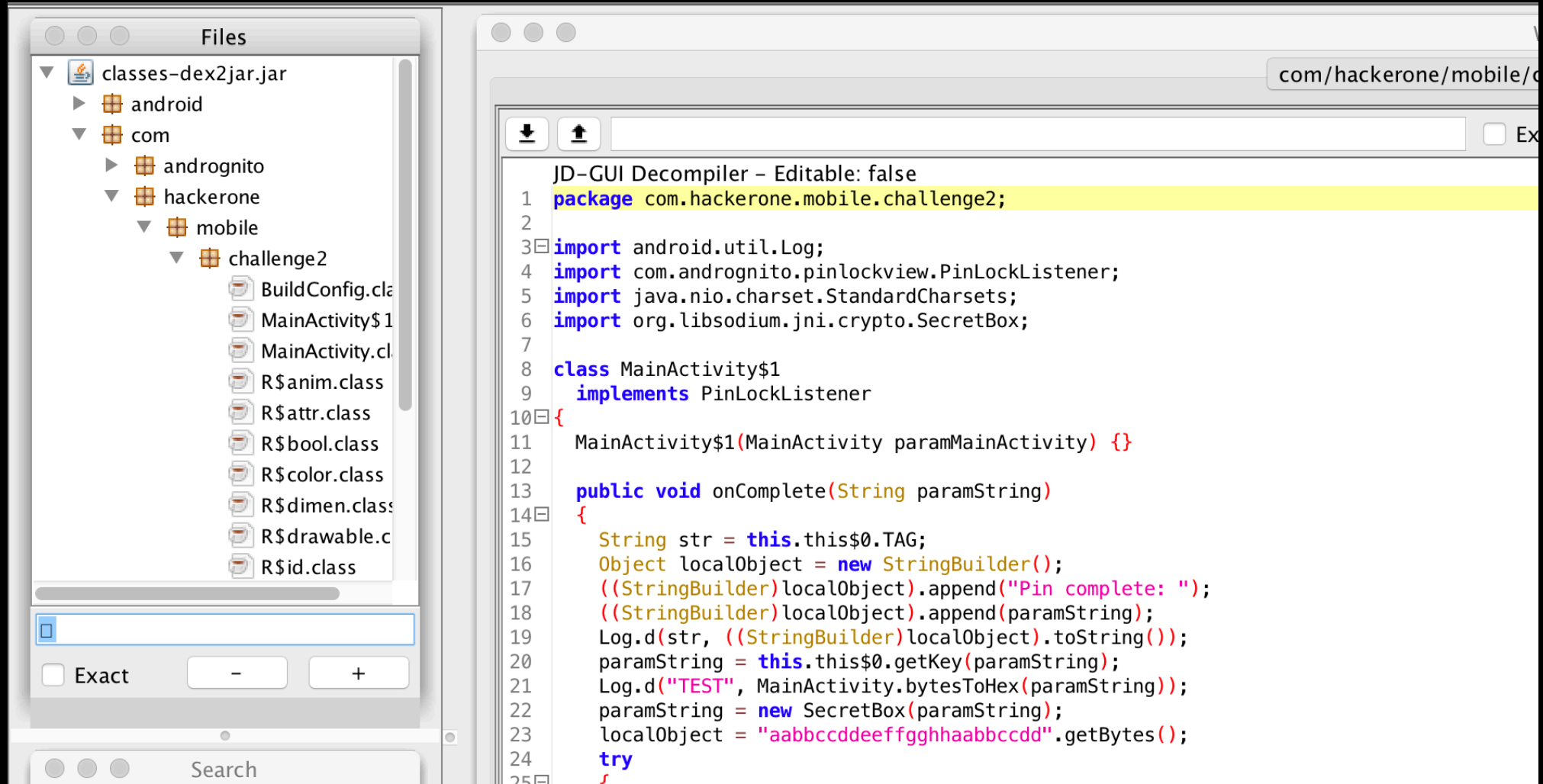
Solution

- App này yêu cầu chúng ta sử dụng điện thoại đời android mới, do đó không thể xài điện thoại ảo **galaxy s6 – 5.1.0 – api 22** như thường nữa
- Chuyển qua xài **genymotion virtual Samsung Galaxy S7 – 7.1.0 – API 25**.
- Cài app lên, mở ra, nhìn sơ qua thì có vẻ ta phải nhập đúng pin để mở khoá:



Solution

- Không thấy thêm gì, nên chuyển qua dịch ngược file apk để phân tích
- Unzip, sử dụng dex2jar để chuyển classes.dex thành jar file, kéo jar file vừa sinh ra vào Bytecode Viewer, ta có code:



Solution

- Chúng ta thấy nhiều lệnh **Log.d**, do đó bật **logcat** lên và sử dụng app để có cái nhìn tổng quan
- First pin number (1):

```
09-27 13:48:42.704 7691 7691 D PinLock : Pin changed, new length 1 with intermediate pin 1
```

- Second pin number (2):

```
09-27 13:48:42.704 7691 7691 D PinLock : Pin changed, new length 1 with intermediate pin 1
09-27 13:48:55.153 7691 7691 D PinLock : Pin changed, new length 2 with intermediate pin 12
09-27 13:48:55.483 265 570 D AudioFlinger: mixer(0xf0e03980) throttle end: throttle time(11)
```

- Third, Fourth, Fifth pin number (3...4...5):

```
09-27 13:48:42.704 7691 7691 D PinLock : Pin changed, new length 1 with intermediate pin 1
09-27 13:48:55.153 7691 7691 D PinLock : Pin changed, new length 2 with intermediate pin 12
09-27 13:48:55.483 265 570 D AudioFlinger: mixer(0xf0e03980) throttle end: throttle time(11)
09-27 13:49:11.286 7691 7691 D PinLock : Pin changed, new length 3 with intermediate pin 123
09-27 13:49:11.595 265 570 D AudioFlinger: mixer(0xf0e03980) throttle end: throttle time(11)
09-27 13:49:14.703 7691 7691 D PinLock : Pin changed, new length 4 with intermediate pin 1234
09-27 13:49:16.353 7691 7691 D PinLock : Pin changed, new length 5 with intermediate pin 12345
```

Solution

- Sixth pin number (6):

```
09-27 13:48:42.704 7691 7691 D PinLock : Pin changed, new length 1 with intermediate pin 1
09-27 13:48:55.153 7691 7691 D PinLock : Pin changed, new length 2 with intermediate pin 12
09-27 13:48:55.483 265 570 D AudioFlinger: mixer(0xf0e03980) throttle end: throttle time(11)
09-27 13:49:11.286 7691 7691 D PinLock : Pin changed, new length 3 with intermediate pin 123
09-27 13:49:11.595 265 570 D AudioFlinger: mixer(0xf0e03980) throttle end: throttle time(11)
09-27 13:49:14.703 7691 7691 D PinLock : Pin changed, new length 4 with intermediate pin 1234
09-27 13:49:16.353 7691 7691 D PinLock : Pin changed, new length 5 with intermediate pin 12345
09-27 13:49:24.337 7691 7691 D PinLock : Pin complete: 123456
09-27 13:49:24.337 7691 7691 D TEST : 6C034108003A2DF4DC02C4C2D8D63CF2ACA689FE7D70E3891CA70C34A59CF28F
09-27 13:49:24.337 7691 7691 I org.libsodium.jni.NaCl: librarypath=/system/lib:/vendor/lib
09-27 13:49:24.338 7691 7691 D PROBLEM : Unable to decrypt text
09-27 13:49:24.338 7691 7691 W System.err: java.lang.RuntimeException: Decryption failed. Ciphertext failed verification
09-27 13:49:24.339 7691 7691 W System.err: at org.libsodium.jni.crypto.Util.isValid(Util.java:47)
09-27 13:49:24.339 7691 7691 W System.err: at org.libsodium.jni.crypto.SecretBox.decrypt(SecretBox.java:56)
09-27 13:49:24.339 7691 7691 W System.err: at com.hackerone.mobile.challenge2.MainActivity$1.onComplete(MainActivity.java:42)
09-27 13:49:24.339 7691 7691 W System.err: at com.andrognito.pinlockview.PinLockView$1.onNumberClicked(PinLockView.java:56)
09-27 13:49:24.339 7691 7691 W System.err: at com.andrognito.pinlockview.PinLockAdapter$NumberViewHolder$1.onClick(PinLockAdapter.java:191)
09-27 13:49:24.339 7691 7691 W System.err: at android.view.View.performClick(View.java:5637)
09-27 13:49:24.339 7691 7691 W System.err: at android.view.View$PerformClick.run(View.java:22429)
09-27 13:49:24.339 7691 7691 W System.err: at android.os.Handler.handleCallback(Handler.java:751)
09-27 13:49:24.339 7691 7691 W System.err: at android.os.Handler.dispatchMessage(Handler.java:95)
09-27 13:49:24.339 7691 7691 W System.err: at android.os.Looper.loop(Looper.java:154)
09-27 13:49:24.339 7691 7691 W System.err: at android.app.ActivityThread.main(ActivityThread.java:6119)
09-27 13:49:24.339 7691 7691 W System.err: at java.lang.reflect.Method.invoke(Native Method)
09-27 13:49:24.339 7691 7691 W System.err: at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:886)
09-27 13:49:24.339 7691 7691 W System.err: at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:776)
```

Solution

- Như đã thấy trong log, ta có thể đoán rằng nếu **pin length < 6**, chương trình in ra số pin hiện tại
- Nếu **pin length = 6**, nó sẽ tạo ra 1 chuỗi kí tự (nhìn giống cái key) và bung ra lỗi

D PROBLEM : Unable to decrypt text

- Sau đó in ra stack trace:

```
09-27 13:49:24.338 7691 7691 W System.err: java.lang.RuntimeException: Decryption failed. Ciphertext failed verification
09-27 13:49:24.339 7691 7691 W System.err:     at org.libsodium.jni.crypto.Util.isValid(Util.java:47)
09-27 13:49:24.339 7691 7691 W System.err:     at org.libsodium.jni.crypto.SecretBox.decrypt(SecretBox.java:56)
09-27 13:49:24.339 7691 7691 W System.err:     at com.hackerone.mobile.challenge2.MainActivity$1.onComplete(MainActivity.java:42)
09-27 13:49:24.339 7691 7691 W System.err:     at com.andrognito.pinlockview.PinLockView$1.onNumberClicked(PinLockView.java:56)
09-27 13:49:24.339 7691 7691 W System.err:     at com.andrognito.pinlockview.PinLockAdapter$NumberViewHolder$1.onClick(PinLockAdapter.java:191)
09-27 13:49:24.339 7691 7691 W System.err:     at android.view.View.performClick(View.java:5637)
09-27 13:49:24.339 7691 7691 W System.err:     at android.view.View$PerformClick.run(View.java:22429)
09-27 13:49:24.339 7691 7691 W System.err:     at android.os.Handler.handleCallback(Handler.java:751)
09-27 13:49:24.339 7691 7691 W System.err:     at android.os.Handler.dispatchMessage(Handler.java:95)
09-27 13:49:24.339 7691 7691 W System.err:     at android.os.Looper.loop(Looper.java:154)
09-27 13:49:24.339 7691 7691 W System.err:     at android.app.ActivityThread.main(ActivityThread.java:6119)
09-27 13:49:24.339 7691 7691 W System.err:     at java.lang.reflect.Method.invoke(Native Method)
09-27 13:49:24.339 7691 7691 W System.err:     at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:886)
09-27 13:49:24.339 7691 7691 W System.err:     at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:776)
```

- Có thể “**123456**” không phải là mã pin đúng => “**Ciphertext failed verification**” => Error!

Solution

- Bây giờ chúng ta sẽ xem qua code để biết chính xác chuyện gì xảy ra
- Class **MainActivity.class**:

```
10 public class MainActivity
11     extends AppCompatActivity
12 {
13     private static final char[] hexArray = "0123456789ABCDEF".toCharArray();
14     String TAG = "PinLock";
15     private byte[] cipherText;
16     IndicatorDots mIndicatorDots;
17     private PinLockListener mPinLockListener = new MainActivity.1(this);
18     PinLockView mPinLockView;
19
20     static
21     {
22         System.loadLibrary("native-lib");
23     }
24
25     public static String bytesToHex(byte[] paramArrayOfByte)
26     {
27
28
29
30
31
32
33
34
35
36
37
38     public native byte[] getKey(String paramString);
39
40     protected void onCreate(Bundle paramBundle)
41     {
42         super.onCreate(paramBundle);
43         setContentView(2131296283);
44         this.cipherText = new Hex().decode("9646D13EC8F8617D1CEA1CF4334940824C700ADF6A7A3236163CA2C9604B9BE4BDE770AD698C02070F571A0B612BBBD3572D81F99");
45         this.mPinLockView = ((PinLockView)findViewById(2131165263));
46         this.mPinLockView.setPinLockListener(this.mPinLockListener);
47         this.mIndicatorDots = ((IndicatorDots)findViewById(2131165241));
48         this.mPinLockView.attachIndicatorDots(this.mIndicatorDots);
49     }
50
51     public native void resetCoolDown();
52 }
53
```

Solution

- `onCreate` method sẽ khởi tạo view và biến:

```
protected void onCreate(Bundle paramBundle)
{
    super.onCreate(paramBundle);
    setContentView(2131296283);
    this.cipherText = new Hex().decode("9646D13EC8F8617D1CEA1CF4334940824C700ADF6A7A3236163CA2C9604B9BE4BDE770AD698C02070F571A0B612BBD3572D81F99");
    this.mPinLockView = ((PinLockView)findViewById(2131165263));
    this.mPinLockView.setPinLockListener(this.mPinLockListener);
    this.mIndicatorDots = ((IndicatorDots)findViewById(2131165241));
    this.mPinLockView.attachIndicatorDots(this.mIndicatorDots);
}
```

- Chúng ta có thể thấy 1 thứ thú vị nè, `cipherText` được gán bằng giá trị của `Hex().decode` của một đoạn hex string
- Sau đó chương trình sử dụng `PinLockListener` để thực hiện các thao tác check pin UI
- Trong `MainActivity$1.class`, ta có thể thấy:

```
public void onPinChange(int paramInt, String paramString)
{
    String str = this.this$0.TAG;
    StringBuilder localStringBuilder = new StringBuilder();
    localStringBuilder.append("Pin changed, new length ");
    localStringBuilder.append(paramInt);
    localStringBuilder.append(" with intermediate pin ");
    localStringBuilder.append(paramString);
    Log.d(str, localStringBuilder.toString());
}
```

Solution

- Và:

```
public void onComplete(String paramString)
{
    String str = this.this$0.TAG;
    Object localObject = new StringBuilder();
    ((StringBuilder)localObject).append("Pin complete: ");
    ((StringBuilder)localObject).append(paramString);
    Log.d(str, ((StringBuilder)localObject).toString());
    paramString = this.this$0.getKey(paramString);
    Log.d("TEST", MainActivity.bytesToHex(paramString));
    paramString = new SecretBox(paramString);
    localObject = "aabbccddeeffgghhaabbccdd".getBytes();
    try
    {
        localObject = paramString.decrypt((byte[])localObject, MainActivity.access
        paramString = new java/lang/String;
        paramString.<init>((byte[])localObject, StandardCharsets.UTF_8);
        Log.d("DECRYPTED", paramString);
    }
    catch (RuntimeException paramString)
    {
        Log.d("PROBLEM", "Unable to decrypt text");
        paramString.printStackTrace();
    }
}
```

- `onPinChange`, `onComplete` biểu hiện giống những gì chúng ta đoán dựa vào logcat, bây giờ chúng ta tập trung vào `onComplete` để biết chương trình làm gì khi input 6-length pin

Solution
















- Sau khi Log “Pin complete: XXXXXX”, pin được đượ vào hàm `getKey(string)`, là một hàm `native`:

```
public native byte[] getKey(String paramString);
```

- Android hỗ trợ developers tạo ra các C/C++ binary và load các hàm từ nó vào chương trình. Do đó, `getKey()` `native function` được load vào từ các binary đó.
- Đến thư mục `lib/x86/`, chúng ta tìm được cái lib:

```
~/Desktop/mobile/h1702-2018/lib/x86/ ls  
libnative-lib.so
```

- Mở nó bằng IDA:

	<code>timeval_addMsecs(timeval *,uint)</code>	<code>.text</code>
	<code>timeval_toMsecs(timeval *)</code>	<code>.text</code>
	<code>timeval_sub(timeval *,timeval *)</code>	<code>.text</code>
	<code>timeval_add(timeval *,timeval *)</code>	<code>.text</code>
	<code>timeval_durationFromNow(timeval *)</code>	<code>.text</code>
	<code>get_key_cooldown(void)</code>	<code>.text</code>
	<code>sub_8E0</code>	<code>.text</code>
	<code>Java_com_hackerone_mobile_challenge2_MainActivity_reset...</code>	<code>.text</code>
	<code>Java_com_hackerone_mobile_challenge2_MainActivity_getKey</code>	<code>.text</code>
	<code>sub_A20</code>	<code>.text</code>
	<code>__cxa_finalize</code>	<code>extern</code>
	<code>__cxa_atexit</code>	<code>extern</code>
	<code>__stack_chk_fail</code>	<code>extern</code>
	<code>gettimeofday</code>	<code>extern</code>
	<code>strlen</code>	<code>extern</code>

Solution

- Dịch ngược `getKey` function:

```
1 int __cdecl Java_com_hackerone_mobile_challenge2_MainActivity_getKey(int a1, int a2, int a3, __int16 a4, int a5,
2 {
3     const char *v12; // esi
4     struct timeval tv; // [esp+10h] [ebp-4Ch]
5     int savedregs; // [esp+5Ch] [ebp+0h]
6
7     v12 = (const char *)((*int (__cdecl **)(int, int, _DWORD))(*(_DWORD *)a1 + 676))(a1, a3, 0);
8     gettimeofday(&tv, 0);
9     tv.tv_usec %= 1000000;
10    JUMPOUT(dword_2004, 51, &loc_A79);
11    return sub_A20((int)&savedregs, tv.tv_usec, v12, a1, a2, (int *)a3, a4, a5, a6, a7, a8, a9, a10, a11, a12);
12 }
```

- Hàm này có vẻ sẽ tính toán và trả về giá trị dựa vào `gettimeofday()`, sau đó lấy giá trị đó và 6 digits pin ta nhập vào để đưa vào hàm `sub_A20`.
- `sub_A20` rất dài, nhìn rất rối nên mình ko dig sâu vào, để ý thì thấy nó làm rất nhiều tính toán trong đó, và nó trả về `key` của 6 digits pin input, chương trình sẽ sử dụng `key & nonce` để decrypt `cipherText`

```
try
{
    localObject = paramString.decrypt((byte[])localObject, MainActivity.access$000(this.this$0));
    paramString = new java/lang/String;
    paramString.<init>((byte[])localObject, StandardCharsets.UTF_8);
    Log.d("DECRYPTED", paramString);
}
```


Solution

- Điềm lại xem này giờ chúng ta có gì nè:

```
1 cipherText='9646D13EC8F8617D1CEA1CF4334940824C700ADF6A7A3236163CA2C9604B9BE4BDE770AD698C02070F571A0B612BBD3572D81F99'.decode('hex')
2 nonce='aabbccddeeffgghhaabbccdd'
3 key=getKey(string)
```

- Ý tưởng bây giờ là, sử dụng frida để hook hàm getKey(pin), gọi nó lên với pin theo ý mình, brute force từ 999999 -> 000000, mã pin đúng là mã pin tạo được key giải mã được cipherText!

Solution

- Code:

```
14 hook_script="""
15 function rpad(width, string, padding)
16 {
17     return (width <= string.length) ? string : rpad(width, padding + string, padding)
18 }
19
20 function genPin(pin)
21 {
22     return rpad(6, pin.toString(), '0')
23 }
24
25 Java.perform
26 (
27     function ()
28     {
29         console.log("inside hook script");
30         nonce = Java.array('byte',[ 97, 97, 98, 98, 99, 99, 100, 100, 101, 101, 102, 102, 103, 103, 104, 104, 97, 97, 98, 98, 99, 99, 100, 100]);
31         cipherText = Java.array('byte',[ 150, 70, 209, 62, 200, 248, 97, 125, 28, 234, 28, 244, 51, 73, 64, 130, 76, 112, 10, 223, 106, 122, 50, 54,
32         Java.choose("com.hackerone.mobile.challenge2.MainActivity",{
33             "onMatch":function(instance)
34             {
35                 console.log("[*] Instance found: "+instance);
36                 counter = 0;
37                 for(var i = 999999; i >= 0; i--)
38                 {
39                     pin = genPin(i.toString());
40                     key = instance.getKey(pin);
41                     console.log("Pin: " + pin + "- Key: " + instance.bytesToHex(key));
42                 }
43             },
44             "onComplete":function()
45             {
46                 console.log("Done");
47             }
48         });
49     }
50 );
51 """
```

Solution

- Chạy thử xem:

```
Pin: 999956- Key: 000000000000000000000000B1D4D900B1D4D9ACA689FEACA689FEAC175D27AC175D27
Pin: 999955- Key: 000000000000000000000000B1D4D900B1D4D9ACA689FEACA689FEAC175D27AC175D27
Pin: 999954- Key: 000000000000000000000000B1D4D909B1825CACA689FEA5A6DF7BAC175D27AC175D27
Pin: 999953- Key: 000000000000000000000000B1D4D9DCB3101BACA689FE70A44D3CAC175D27AC175D27
Pin: 999952- Key: 000000000000000000000000B1D4D9098BAFA8ACA689FEA59CF28FAC175D27AC175D27
Pin: 999951- Key: 000000000000000000000000B1D4D9B0B05113ACA689FE1CA70C34AC175D27AC175D27
Pin: 999950- Key: 000000000000000000000000B1D4D900B1D4D9ACA689FEACA689FEAC175D27AC175D27
Pin: 999949- Key: 000000000000000000000009B1825C00000000A5A6DF7BAC175D27AC175D27AC175D27
```

- Tự nhiên nó pause lại ở 999949, sau đó chạy tiếp, và vì 1 lí do nào đó bị timeout luôn > <

```
Pin: 999855- Key: 00000000499B77D800B1D4D9492AA301ACA689FEACA689FEAC175D27AC175D27
Pin: 999854- Key: 00000000499B77D800B1D4D9402AF584ACA689FEA5A6DF7BAC175D27AC175D27
Pin: 999853- Key: 00000000499B77D800B1D4D9952867C3ACA689FE70A44D3CAC175D27AC175D27
Pin: 999852- Key: 00000000499B77D800B1D4D94010D870ACA689FEA59CF28FAC175D27AC175D27
Pin: 999851- Key: 00000000499B77D800B1D4D9F92B26CBACA689FE1CA70C34AC175D27AC175D27
Pin: 999850- Key: 00000000499B77D800B1D4D9492AA301ACA689FEACA689FEAC175D27AC175D27
Pin: 999849- Key: 00000000499B77D809B1825C499B77D8A5A6DF7BAC175D27AC175D27AC175D27
Pin: 999848- Key: 00000000499B77D809B1825C00000000A5A6DF7BE58C2AFFAC175D27AC175D27
Pin: 999847- Key: 00000000499B77D809B1825C2DE2E6E5A5A6DF7BC86ECC1AAC175D27AC175D27
Pin: 999846- Key: 00000000499B77D809B1825C98FCC976A5A6DF7B7D70E389AC175D27AC175D27
Pin: 999845- Key: 00000000499B77D809B1825C492AA301A5A6DF7BACA689FEAC175D27AC175D27
Traceback (most recent call last):
  File "/Users/tsug0d/Desktop/mobile/lab1_frida.py", line 55, in <module>
    script.load()
  File "/usr/local/lib/python2.7/site-packages/frida/core.py", line 192, in load
    self._impl.load()
frida.TransportError: timeout was reached
```

Solution

- Thiệt ra đó là frida timeout, để sửa thì chỉ cần thêm `setTimeout()` function ở ngoài là được

```
setTimeout(function ()  
{  
    function x();  
    function y();  
    Java.perform(function()  
    {  
        //blah blah;  
    }  
})  
, 0);
```

- Vấn đề là, nếu cứ `pause mỗi 51 lần`, thì brute-force `chậm bà cố luôn`, ngồi chờ tới tết ☺

Solution

- Xem code đoạn đó thử `Java_com_hackerone_mobile_challenge2_MainActivity_getKey`

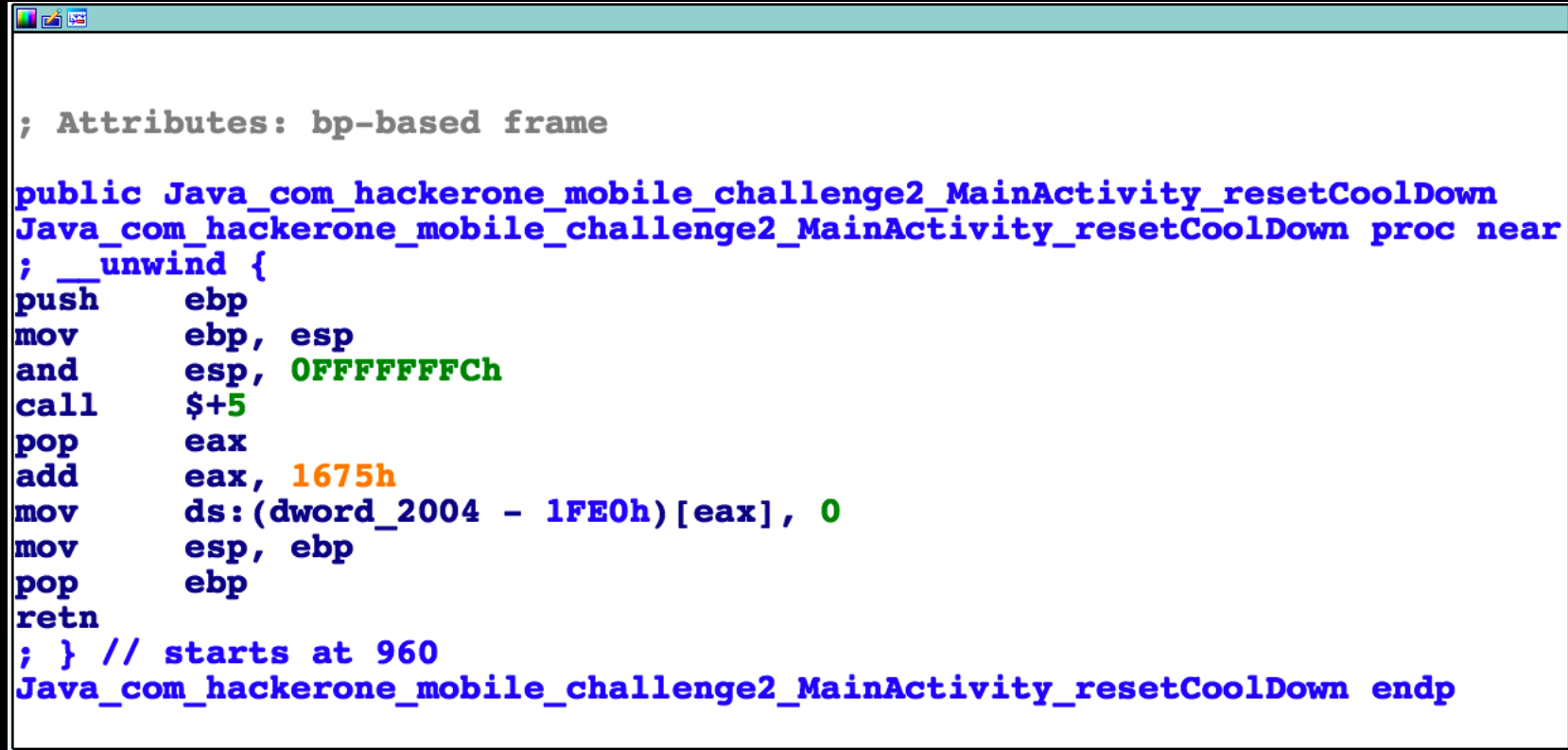
```
mov     eax, ds:(dword_2004 - 1FE0h)[ebx]
cmp     eax, 51
jb      short loc_A79

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
Java_com_hackerone_mobile_challenge2_MainActivity_getKey endp ; sp-analysis failed
```

- Giá trị biến đếm được move vào eax, sau đó so sánh với 51, nếu nhỏ hơn thì jump tới `loc_A79` (là chạy tiếp được), else fail, đó là lí do vì sao nó dừng tại 999949
- Khúc này rõ ràng là để chống brute-force pin, vậy bó tay hử? Thiệt ra vẫn còn 1 function mà chúng ta chưa ngó qua

Solution

- Đó là `Java_com_hackerone_mobile_challenge2_MainActivity_resetCoolDown()`



```
; Attributes: bp-based frame

public Java_com_hackerone_mobile_challenge2_MainActivity_resetCoolDown
Java_com_hackerone_mobile_challenge2_MainActivity_resetCoolDown proc near
; __unwind {
push     ebp
mov      ebp, esp
and      esp, 0FFFFFFFCh
call     $+5
pop      eax
add      eax, 1675h
mov      ds:(dword_2004 - 1FE0h)[eax], 0
mov      esp, ebp
pop      ebp
retn
; } // starts at 960
Java_com_hackerone_mobile_challenge2_MainActivity_resetCoolDown endp
```

- Như hình, nó `move 0` vào `biến đếm`, do đó `reset giá trị` của biến đếm (hoặc nhìn cái tên hàm cũng đủ hiểu), vậy sử dụng nó, thì giá trị của chúng ta sẽ không bao giờ vượt quá 51 => no pause

Solution

- Trong code ta sẽ khởi tạo `counter=0`, tăng nó sau mỗi vòng lặp, nếu bằng 50 thì call `resetCoolDown()` để reset, ez mmr!

```
32     Java.choose("com.hackerone.mobile.challenge2.MainActivity",{
33         "onMatch":function(instance)
34         {
35             console.log("[*] Instance found: "+instance);
36             counter = 0;
37             for(var i = 999999; i >= 0; i--)
38             {
39                 pin = genPin(i.toString());
40                 key = instance.getKey(pin);
41                 console.log("Pin: " + pin + "- Key: " + instance.bytesToHex(key));
42                 counter+=1;
43                 if(counter==50)
44                 {
45                     instance.resetCoolDown();
46                     counter=0;
47                 }
48             }
49         },
50         "onComplete":function()
51         {
52             console.log("Done");
53         }
54     });
```

Solution

- chạy script, chúng ta lại gặp một vấn đề khác, chương trình lại dừng tại 949284 (hình dưới)

```
Pin: 949288 - Key: 0000000003A2DF4499B77D84010D870E58C2AFFE58C2AFFAC175D27A5A6DF7B
Pin: 949287 - Key: 0000000003A2DF4499B77D86DF23E95E58C2AFFC86ECC1AAC175D27A5A6DF7B
Pin: 949286 - Key: 0000000003A2DF4499B77D8D8EC1106E58C2AFF7D70E389AC175D27A5A6DF7B
Pin: 949285 - Key: 0000000003A2DF4499B77D8093A7B71E58C2AFFACA689FEAC175D27A5A6DF7B
Pin: 949284 - Key: 0000000003A2DF4499B77D8003A2DF4E58C2AFFA5A6DF7BAC175D27A5A6DF7B
```

- Trong logcat, thấy được:

```
JNI ERROR (app bug): global reference table overflow (max=51200)
```

```
global reference table dump:
```

```
  Last 10 entries (of 51200):
```

```
  51199: 0x130aa0a0 byte[] (32 elements)
  51198: 0x130aa070 byte[] (32 elements)
  51197: 0x1309afd0 byte[] (32 elements)
  51196: 0x1309afa0 byte[] (32 elements)
  51195: 0x1309af70 byte[] (32 elements)
  51194: 0x1309af40 byte[] (32 elements)
  51193: 0x1309af10 byte[] (32 elements)
  51192: 0x1309aee0 byte[] (32 elements)
  51191: 0x1309aeb0 byte[] (32 elements)
  51190: 0x1309ae80 byte[] (32 elements)
```

- Global reference table overflow cmnr >_<! theo lý thuyết thì xài **garbage collection** giải quyết được mà mình làm hem được nên bó tay, bạn nào biết thì pm tsublogs@gmail.com chỉ mình hen => như vậy chỉ brute được 1 lần khoảng ~50000 pin, trong trường hợp xấu nhất (giả sử pin nằm từ 000000->050000) thì chúng ta phải chạy $1000000/50000=20$ lần

Solution

- Bây giờ thì chỉ còn coding skill thôi, tạo cái flag để chương trình dừng khi mã pin chính xác
- Nói sơ lại về quá trình mã hoá:

```
paramString = this.this$0.getKey(paramString);
```

```
paramString = new SecretBox(paramString);
```

```
localObject = paramString.decrypt((byte[])localObject, MainActivity.access$000(this.this$0));
```

- Nó sử dụng **SecretBox class của libsodium**, do đó ta dùng frida tạo instance của class này, rồi gọi hàm decrypt ra

```
secretBoxClass = Java.use("org.libsodium.jni.crypto.SecretBox");  
decrypt_result = secretBoxClass.$new(key).decrypt(nonce,cipherText);
```

- Lưu ý là nếu decryption fail, đoạn script sẽ dừng stop, do đó chúng ta phải để nó trong **try – catch – finally statement**
- Ý tưởng bây giờ là, **tạo biến flag = false**, khi nào **decryption success**, ta **gán nó thành true và break the loop**

Solution

- Code:

```
flag = false;
secretBoxClass = Java.use("org.libsodium.jni.crypto.SecretBox");
Java.choose("com.hackerone.mobile.challenge2.MainActivity",{
  "onMatch":function(instance)
  {
    console.log("[*] Instance found: "+instance);
    counter = 0;
    for(var i = 930000; i >= 0; i--)
    {
      pin = genPin(i.toString());
      key = instance.getKey(pin);
      console.log("Pin: " + pin + " - Key: " + instance.bytesToHex(key));

      try
      {
        decrypt_result = secretBoxClass.$new(key).decrypt(nonce,cipherText);
        flag = true;
      }
      catch(err)
      {
        //Do nothing
      }
      finally
      {
        if ( flag == true )
        {
          console.log("Found");
          console.log("Pin: "+ pin );
          break;
        }
      }
    }
  }
});
```

Solution

- Chạy lại đoạn script, ta tìm thấy correct pin:

```
Pin: 918278 – Key: 499B77D8B93BFEBB2DE2E6E54010D870C86ECC1AE58C2AFFAC175D271CA70C34
Pin: 918277 – Key: 499B77D8B93BFEBB2DE2E6E56DF23E95C86ECC1AC86ECC1AAC175D271CA70C34
Pin: 918276 – Key: 499B77D8B93BFEBB2DE2E6E5D8EC1106C86ECC1A7D70E389AC175D271CA70C34
Pin: 918275 – Key: 499B77D8B93BFEBB2DE2E6E5093A7B71C86ECC1AACA689FEAC175D271CA70C34
Pin: 918274 – Key: 499B77D8B93BFEBB2DE2E6E5003A2DF4C86ECC1AA5A6DF7BAC175D271CA70C34
Pin: 918273 – Key: 499B77D8B93BFEBB2DE2E6E5D538BFB3C86ECC1A70A44D3CAC175D271CA70C34
Pin: 918272 – Key: 499B77D8B93BFEBB2DE2E6E500000000C86ECC1AA59CF28FAC175D271CA70C34
Pin: 918271 – Key: 499B77D8B93BFEBB2DE2E6E5B93BFEBBC86ECC1A1CA70C34AC175D271CA70C34
Pin: 918270 – Key: 499B77D8B93BFEBB2DE2E6E56DF23E95C86ECC1AC86ECC1AAC175D271CA70C34
Pin: 918269 – Key: 499B77D8B93BFEBB98FCC976098BAFA87D70E389AC175D27AC175D271CA70C34
Pin: 918268 – Key: 499B77D8B93BFEBB98FCC9764010D8707D70E389E58C2AFFAC175D271CA70C34
Pin: 918267 – Key: 499B77D8B93BFEBB98FCC9766DF23E957D70E389C86ECC1AAC175D271CA70C34
Pin: 918266 – Key: 499B77D8B93BFEBB98FCC976D8EC11067D70E3897D70E389AC175D271CA70C34
Pin: 918265 – Key: 499B77D8B93BFEBB98FCC976093A7B717D70E389ACA689FEAC175D271CA70C34
Pin: 918264 – Key: 499B77D8B93BFEBB98FCC976003A2DF47D70E389A5A6DF7BAC175D271CA70C34
Found
Pin: 918264
```

- Bật logcat lên, nhập pin tìm được, we got flag:

```
09-27 20:03:30.902 11681 11681 D PinLock : Pin complete: 918264
09-27 20:03:30.903 11681 11681 D TEST      : 499B77D8B93BFEBB98FCC976003A2DF47D70E389A5A6DF7BAC175D271CA70C34
09-27 20:03:30.903 11681 11681 I org.libsodium.jni.NaCl: librarypath=/system/lib:/vendor/lib
09-27 20:03:30.904 11681 11681 D DECRYPTED: flag{wow_yall_called_a_lot_of_func$}
```

Bonus Solution

- Nhớ hàm `resetCooldown()` hem? Tại sao phải có hàm này nhỉ?
- Giả sử nếu brute-force tới 51 lần, app quit luôn, khỏi cần `resetCooldown()` để chống brute-force nữa 😊
- Bây giờ chúng ta sẽ giải lại bài này, mà không dùng `resetCooldown()` !

Bonus Solution

- Nhiệm vụ bây giờ khá rõ ràng: Khiến giá trị biến đếm không đạt 51 và không xài `resetCoolDown()`
- Frida cung cấp cho ta các method để tương tác với memory dynamically (more like enter god-mode 😊)
- Đầu tiên ta sẽ tìm base-address của `libnative-lib` library (đọc thêm về dynamic linker để hiểu vì sao nhé) sử dụng `Module.findBaseAddress`

```
libnative = Module.findBaseAddress("libnative-lib.so");
```

- `Module.findBaseAddress(name)`: returns the base address of the `name` module, or `null` if the module isn't loaded

Bonus Solution

- Bây giờ ta tìm địa chỉ của biến đếm trên memory
- Trong IDA, hàm `Java_com_hackerone_mobile_challenge2_MainActivity_getKey`:

```
int __cdecl Java_com_hackerone_mobile_challenge2_MainActivity_getKey(int a1, int a2, int a3, __int16 a4, int a5)
{
    const char *v12; // esi
    struct timeval tv; // [esp+10h] [ebp-4Ch]
    int savedregs; // [esp+5Ch] [ebp+0h]

    v12 = (const char *)((*int (__cdecl **)(int, int, _DWORD))(*(_DWORD *)a1 + 676))(a1, a3, 0);
    gettimeofday(&tv, 0);
    tv.tv_usec %= 1000000;
    JUMPOUT(dword_2004, 51, &loc_A79);
    return sub_A20((int)&savedregs, tv.tv_usec, v12, a1, a2, (int *)a3, a4, a5, a6, a7, a8, a9, a10, a11, a12);
}
```

- Ta có thể thấy `JUMPOUT` so sánh giá trị ở `dword_2004` với `51`, do đó cá 1 ăn 10 nó là biến đếm, xem thử:

```
.bss:00002004 dword_2004      dd ?                                ; DATA XREF: LOAD:00000248↑o
.bss:00002004                                     ; LOAD:00000268↑o ...
.bss:00002004 _bss          ends
.bss:00002004
```

Bonus Solution

- virtual address = base address + offset, tính thôi:

```
counter_address = libnative.add(ptr("0x00002004"));
```

- Giờ set giá trị nó thành 1 (which is below 51)

```
Memory.writeInt(counter_address,1);
```

- - Memory.writeS8(address, value),
 - Memory.writeU8(address, value),
 - Memory.writeS16(address, value),
 - Memory.writeU16(address, value),
 - Memory.writeS32(address, value),
 - Memory.writeU32(address, value),
 - Memory.writeShort(address, value),
 - Memory.writeUShort(address, value),
 - Memory.writeInt(address, value),
 - Memory.writeUInt(address, value),
 - Memory.writeFloat(address, value),
 - Memory.writeDouble(address, value) : write the number **value** to the signed or unsigned 8/16/32/etc. or float/double value at **address**.
- A JavaScript exception will be thrown if **address** isn't writable.

Bonus Solution

- Ok, thay hàm resetCoolDown trong đoạn script cũ thành những gì chúng ta đã nói ở trên, here the full PoC script:

https://github.com/tsug0d/AndroidMobilePentest101/blob/master/lab/frida_lab/lab_frida_bonus.py