Android Mobile Pentest 101

Bài 4 – Dịch ngược

Mục tiêu: Học được 1 số kĩ năng dịch ngược app

Inside the apk

- APK (Android application package) là 1 file chứa các thành phần cần thiết cho việc cài đặt app trên thiết bị android
- Giống như windows có file .exe, thì android cũng có file .apk, quá trình cài đặt tương tự. Việc cài đặt thủ công (cài bằng file apk) được gọi là side-loading.
- Dưới đây là các thành phần thường có trong file apk:

- **META-INF/:** Thư mục này có ở trong file APK đã sign, nó bao gồm các danh sách file trong APK và chữ kí của chúng
- **lib/:** Các file của thư viện native (file có đuôi *.so) được lưu vào thư mục con của lib/ như x86, x86_64
- **res/:** Folder này gồm tất cả các resource XML, drawables (PNG,JPEG) với các loại khác nhau từ mdpi, hdpi, sw600dp... cho tới resource ngôn ngữ
- **AndroidManifest.xml**: Mô tả tên, phiên bản, và nội dung của file APK
- **classes.dex**: Bao gồm mã nguồn đã biên dịch, được chuyển đưới dạng Dex bytecode
- **resources.arsc**: Một số tài nguyên và định nghĩa được biên dịch và lưu trữ ở đây. Các resource này lưu vào APK mà không cần nén để truy cập nhanh hơn lúc runtime.

Inside the apk

- APK là 1 file nén, cụ thể là zip format-type, dựa trên JAR file format, với .apk là phần mở rộng.
- Thử unzip nó xem:

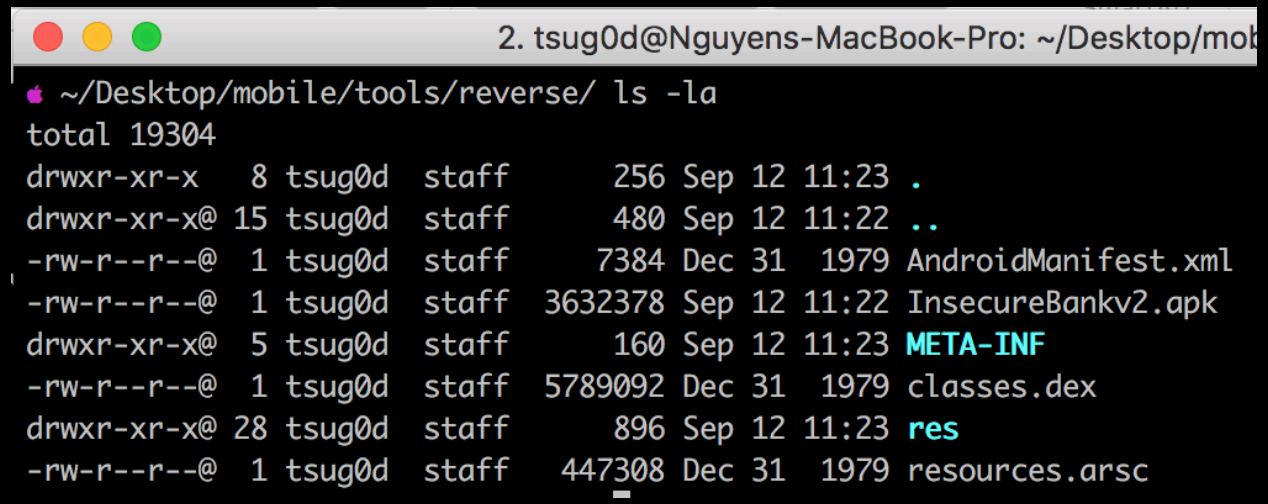
2. tsug0d@Nguyens-MacBook-Pro: ~/Desktop/mobile/tools/reverse (zsh)

```
~/Desktop/mobile/tools/reverse/ file InsecureBankv2.apk
InsecureBankv2.apk: Zip archive data, at least v2.0 to extract
~/Desktop/mobile/tools/reverse/ unzip InsecureBankv2.apk
Archive:  InsecureBankv2.apk
  inflating: AndroidManifest.xml
  inflating: META-INF/CERT.RSA
  inflating: META-INF/CERT.SF
  inflating: META-INF/MANIFEST.MF
  inflating: classes.dex
  inflating: res/anim/abc_fade_in.xml
  inflating: res/anim/abc_fade_out.xml
  inflating: res/anim/abc_grow_fade_in_from_bottom.xml
  inflating: res/anim/abc_shrink_fade_out_from_bottom.xml
  inflating: res/anim/abc_slide_in_bottom.xml
  inflating: res/anim/abc_slide_in_top.xml
  inflating: res/anim/abc_slide_out_bottom.xml
  inflating: res/anim/abc_slide_out_top.xml
  inflating: res/color/abc_background_cache_hint_selector_material_dark.xml
  inflating: res/color/abc_background_cache_hint_selector_material_light.xml
  inflating: res/color/abc_primary_text_disable_only_material_dark.xml
  inflating: res/color/abc_primary_text_disable_only_material_light.xml
  inflating: res/color/abc_primary_text_material_dark.xml
```

unzip-ing

Inside the apk

- Những thành phần nói ở trên đã xuất hiện ☺



- Nhưng như vậy thì chưa đủ để dịch ngược (Vì đây là những nội dung đã được biên dịch, sẽ ở dạng binary).
- Do đó, ta phải biên dịch ngược (decompile) nó!

Inside the apk

- APKTool sẽ được chúng ta sử dụng để decompile
- apktool giúp decompile file AndroidManifest về lại định dạng file xml nguyên bản, cũng như resources.arsc file và classes.dex file thành 1 ngôn ngữ gọi là SMALI
- Gõ lệnh:

apktool d InsecureBankv2.apk

- Sau khi hoàn thành, thư mục InsecureBankv2 sẽ được tạo ra, bạn sẽ tìm thấy các thư mục nói ở trên (bao gồm thư mục chứa smali code).

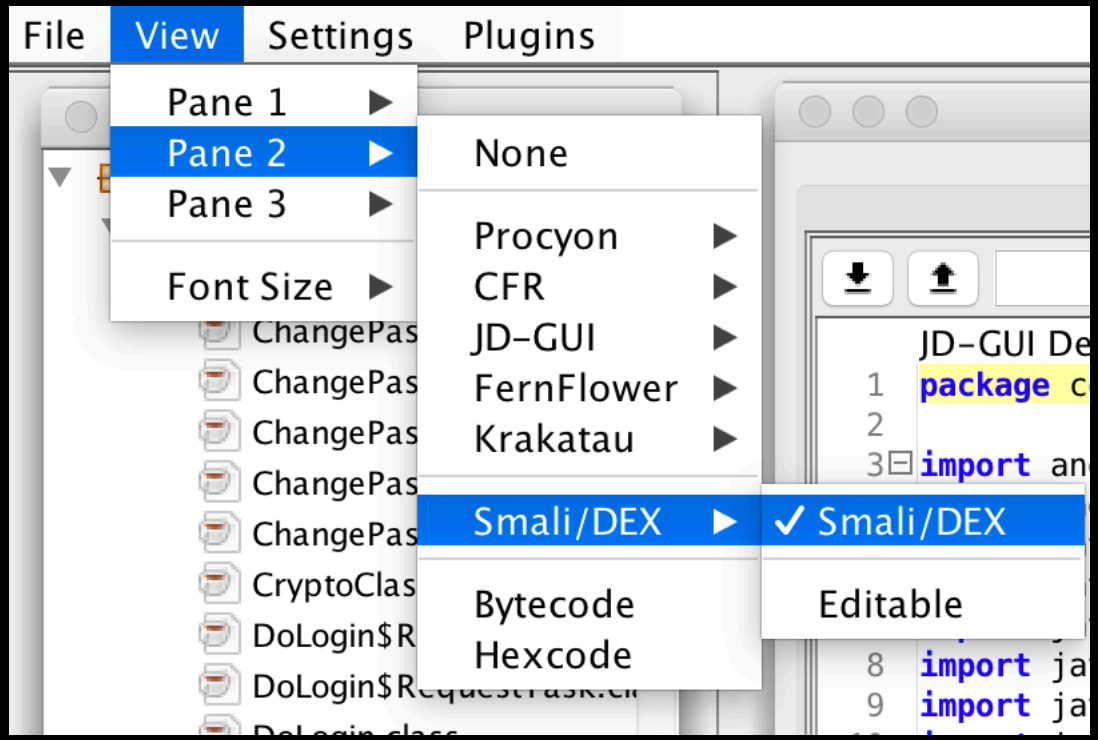
```
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/ ls
AndroidManifest.xml apktool.yml         original                res                     smali
```

- Đến bước này, bạn có thể lấy java source bằng tool dex2jar (output is jar file)
- Sau đó đọc nội dung các jar file đã sinh ra bằng jd-gui

Ở trên là những bước cơ bản khi chuẩn bị dịch ngược 1 file APK, để tự động hoá quá trình đó, ta sử dụng tool ByteCode Viewer (remember? ☺)

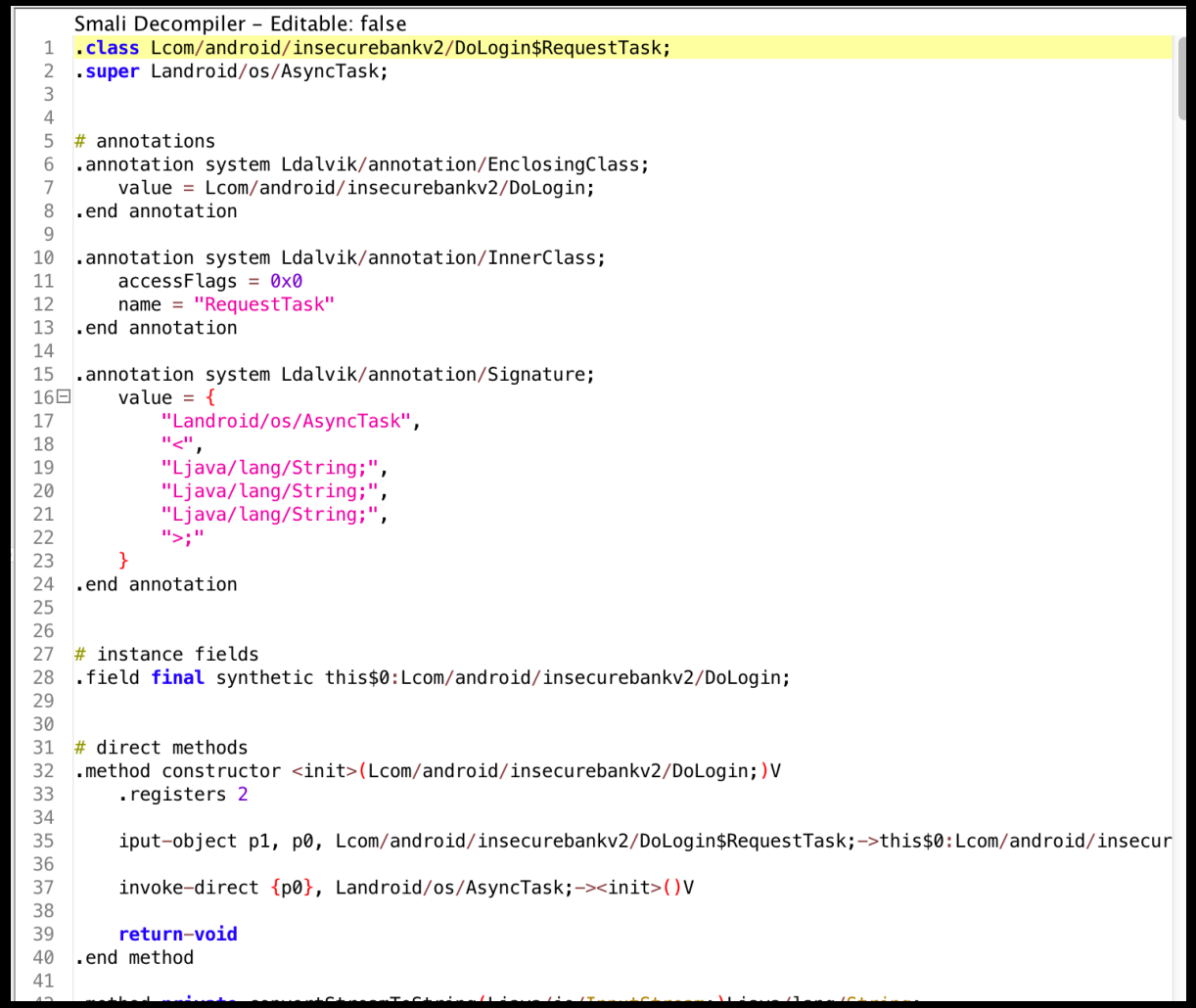
smali

- Bây giờ chúng ta sẽ nói về mã SMALI
- Load file apk vào ByteCode Viewer
- Sau đó chọn như hình dưới



smali

- Bây giờ mở Insecurebankv2.apk/com/android/insecurebankv2/DoLogin$RequestTask.class, chúng ta sẽ thấy đươc smali code của class đó ở panel 2



Smali Decompiler - Editable: false

```
.class Lcom/android/insecurebankv2/DoLogin$RequestTask;
.super Landroid/os/AsyncTask;


# annotations
.annotation system Ldalvik/annotation/EnclosingClass;
    value = Lcom/android/insecurebankv2/DoLogin;
.end annotation

.annotation system Ldalvik/annotation/InnerClass;
    accessFlags = 0x0
    name = "RequestTask"
.end annotation

.annotation system Ldalvik/annotation/Signature;
    value = {
        "Landroid/os/AsyncTask",
        "<",
        "Ljava/lang/String;",
        "Ljava/lang/String;",
        "Ljava/lang/String;",
        ">;"
    }
.end annotation


# instance fields
.field final synthetic this$0:Lcom/android/insecurebankv2/DoLogin;


# direct methods
.method constructor <init>(Lcom/android/insecurebankv2/DoLogin;)V
    .registers 2

    iput-object p1, p0, Lcom/android/insecurebankv2/DoLogin$RequestTask;->this$0:Lcom/android/insecur

    invoke-direct {p0}, Landroid/os/AsyncTask;-><init>()V

    return-void
.end method
```

smali

- Bạn còn nhớ user “devadmin” ở bài trước? Bây giờ cùng xem lại quá trình hoạt động của nó với mã java bytecode (smali) nhé ☺

```
if (this.this$0.username.equals("devadmin"))
{
  localHttpPost2.setEntity(new UrlEncodedFormEntity(localArrayList));
  localHttpResponse = localDefaultHttpClient.execute(localHttpPost2);
}
else
{
  localHttpPost1.setEntity(new UrlEncodedFormEntity(localArrayList));
  localHttpResponse = localDefaultHttpClient.execute(localHttpPost1);
}
```

smali

- Đầu tiên ta đi tìm chuỗi đó trong đoạn smali code

```
407     const-string v5, "devadmin"
408
409     invoke-virtual {v4, v5}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
410
411     move-result v4
412
413     if-eqz v4, :cond_11c
```

- Nó sẽ lấy chuỗi ta nhập vào (v4) và so sánh với chuỗi "devadmin" (v5) bằng cách gọi Java.lang.String.equals() method, nếu đúng thì trả về 1, sai thì trả về 0
- Lưu kết quả trả về vào v4
- Sau đó chương trình gọi câu lệnh điều kiện if-eqz (if equal zero, nếu bằng không), điều đó có nghĩa là nếu chuỗi chúng ta nhập vào không phải là "devadmin", chúng ta sẽ nhảy đến phần code ở :cond_11c

smali

- Chúng ta cùng xem qua đoạn code smali khi if-eqz không thoả và thoả

Nhập vào là “devadmin”

```
413     if-eqz v4, :cond_11c
414
415     new-instance v2, Lorg/apache/http/client/entity/UrlEncodedFormEntity;
416
417     invoke-direct {v2, v1}, Lorg/apache/http/client/entity/UrlEncodedFormEntity;-><init>(Ljava/util/List;)V
418
419     invoke-virtual {v3, v2}, Lorg/apache/http/client/methods/HttpPost;->setEntity(Lorg/apache/http/HttpEntity;)V
420
421     invoke-interface {v0, v3}, Lorg/apache/http/client/HttpClient;->execute(Lorg/apache/http/client/methods/HttpUriRequest;)Lorg/apache/http/HttpResponse;
422
423     move-result-object v0
424
425     :goto_99
```

Nhập vào không phải là “devadmin”

```
:cond_11c
new-instance v3, Lorg/apache/http/client/entity/UrlEncodedFormEntity;

invoke-direct {v3, v1}, Lorg/apache/http/client/entity/UrlEncodedFormEntity;-><init>(Ljava/util/List;)V

invoke-virtual {v2, v3}, Lorg/apache/http/client/methods/HttpPost;->setEntity(Lorg/apache/http/HttpEntity;)V

invoke-interface {v0, v2}, Lorg/apache/http/client/HttpClient;->execute(Lorg/apache/http/client/methods/HttpUriRequest;)Lorg/apache/http/HttpResponse;

move-result-object v0

goto/16 :goto_99
```

Khá giống nhau ☺

smali

- Xem qua đoạn if-ezq không thoả trước (nhập vào đúng là “devadmin”):

```
413     if-eqz v4, :cond_11c
414
415     new-instance v2, Lorg/apache/http/client/entity/UrlEncodedFormEntity;
416
417     invoke-direct {v2, v1}, Lorg/apache/http/client/entity/UrlEncodedFormEntity;-><init>(Ljava/util/List;)V
418
419     invoke-virtual {v3, v2}, Lorg/apache/http/client/methods/HttpPost;->setEntity(Lorg/apache/http/HttpEntity;)V
420
421     invoke-interface {v0, v3}, Lorg/apache/http/client/HttpClient;->execute(Lorg/apache/http/client/methods/HttpUriRequest;)Lorg/apache/http/HttpResponse;
422
423     move-result-object v0
424
425     :goto_99
```

v2=new UrlEncodedFormEntity;

v2(v1 local_list) == new UrlEncodedFormEntity(v1)

v3.setEntity(v2)

HttpClient.execute(v3)

- Đoạn code smali trên tương đương với đoạn code java ở dưới

```
{
  localHttpPost2.setEntity(new UrlEncodedFormEntity(localArrayList));
  localHttpResponse = localDefaultHttpClient.execute(localHttpPost2);
}
```

Vậy v3 là gì? Next-slide ☺

smali

```
361     const-string v3, "/devlogin"   v3="/devlogin"
362                                                                    v1.append(v3) => v1.append('/devlogin')
363     invoke-virtual {v1, v3}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
364
365     new-instance v3, Lorg/apache/http/client/methods/HttpPost;  v3=new HttpPost
366
367     invoke-virtual {v1}, Ljava/lang/StringBuilder;->toString()Ljava/lang/String;   v1.toString()
368
369     move-result-object v1
370
```

```
371     invoke-direct {v3, v1}, Lorg/apache/http/client/methods/HttpPost;-><init>(Ljava/lang/String;)V   Call HttpPost(v1)
```

- Đoạn code smali trên tương đương với đoạn code java ở dưới

```
116     localStringBuilder2.append("/devlogin");
117     HttpPost localHttpPost2 = new HttpPost(localStringBuilder2.toString());
```

- Tóm lại nếu chúng ta nhập vào là "devadmin", nó sẽ gửi 1 truy vấn POST đến http://mobile-server/devlogin, và cho phép chúng ta login tự do ☺

smali

- Có thể bạn sẽ tự hỏi tại sao phải đọc mã smali? Sao không đọc luôn code java cho nhanh?

Câu trả lời là, chúng ta phải đọc nó, để biết nó ứng với đoạn code java nào, từ đó mới patch được chương trình! (nói sau)

Patching android app

- Bây giờ chúng ta sẽ nói về cách patch 1 app android (không đụng tới smali trong phần này, great ☺)
- Patch đơn giản là tác động và làm thay đổi code của chương trình, khiến nó hoạt động theo ý mình.
- Bạn còn nhớ apktool không? Chúng ta sẽ sử dụng nó để patch, đầu tiên decompile file apk trước:

apktool d InsecureBankv2.apk

```
 ~/Desktop/mobile/tools/reverse/test/ apktool d InsecureBankv2.apk
I: Using Apktool 2.3.4 on InsecureBankv2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/tsug0d/Library/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...
```

Patching android app

- Chúng ta vào thư mục res xem thử nhé (vì sao? Bởi vì trong thư mục này chứa các resource, và thay đổi nó thì rất thú vị ☺)
- Nhanh chóng thấy được thứ thú vị trong InsecureBankv2/res/values/strings.xml

```
61      <string name="create_calendar_message">Allow Ad to create a calendar event?</string>
62      <string name="create_calendar_title">Create calendar event</string>
63      <string name="decline">Decline</string>
64      <string name="hello_world">Hello world!</string>
65      <string name="is_admin">no</string>
66      <string name="loginscreen_password">Password:</string>
67      <string name="loginscreen_username">Username:</string>
68      <string name="mr_media_route_button_content_description">Cast</string>
69      <string name="mr_media_route_chooser_searching">Searching for devices…</string>
```

- Nếu bạn muốn làm 1 pentester giỏi, hay nhà bảo mật ưu tú, thì phải lưu nhớ rằng: “thấy cái gì yes thì chuyển thành no, no thì chuyển thành yes”
- Cho nên chúng ta sẽ sửa nó thành yes

```
64      <string name="hello_world">Hello world!</string>
65      <string name="is_admin">yes</string>
66      <string name="loginscreen_password">Password:</string>
67      <string name="loginscreen_username">Username:</string>
```

Patching android app

- Trở ngược ra folder gốc

```
 ~/Desktop/mobile/tools/reverse/test/ ls
InsecureBankv2      InsecureBankv2.apk  _
```

- Và recompile lại thành file apk:

apktool b InsecureBankv2

```
 ~/Desktop/mobile/tools/reverse/test/ apktool b InsecureBankv2
I: Using Apktool 2.3.4
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...
```

Patching android app

- File apk vừa recompile sẽ nằm ở thư mục “dist”

```
 ~/Desktop/mobile/tools/reverse/test/ ls -la InsecureBankv2/dist
total 6568
drwxr-xr-x  3 tsug0d  staff       96 Sep 14 11:24 .
drwxr-xr-x  9 tsug0d  staff      288 Sep 14 11:24 ..
-rw-r--r--  1 tsug0d  staff  3361945 Sep 14 11:24 InsecureBankv2.apk
```

- Cài nó thôi nhỉ?

```
 ~/Desktop/mobile/tools/reverse/test/ cd InsecureBankv2/dist
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ adb install InsecureBankv2.apk
InsecureBankv2.apk: 1 file pushed. 58.7 MB/s (3361945 bytes in 0.055s)
        pkg: /data/local/tmp/InsecureBankv2.apk
Failure [INSTALL_PARSE_FAILED_NO_CERTIFICATES]
```

- Oops, Failed! Bởi vì Every new compiled Android .apk needs to be signed if it is going to be installed on a phone (nôm na là file compiled Android .apk mới phải được sign trước khi cài đặt)

Patching android app

- Vậy thì sign thôi ☺
- Đầu tiên tạo 1 cái key:

keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -keyalg RSA -keysize 2048 -validity 10000

```
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -keyalg RSA -keysize 2048 -validity 10000

Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  nguyen
What is the name of your organizational unit?
  [Unknown]:  nguyen
What is the name of your organization?
  [Unknown]:  tsu
What is the name of your City or Locality?
  [Unknown]:  hcm
What is the name of your State or Province?
  [Unknown]:  hcm
What is the two-letter country code for this unit?
  [Unknown]:  sg
Is CN=nguyen, OU=nguyen, O=tsu, L=hcm, ST=hcm, C=sg correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
        for: CN=nguyen, OU=nguyen, O=tsu, L=hcm, ST=hcm, C=sg
Enter key password for <alias_name>
        (RETURN if same as keystore password):
Re-enter new password:
[Storing my-release-key.keystore]
```

Patching android app

- Sau đó xài jarsigner với key vừa tạo để sign:

jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore InsecureBankv2.apk alias_name

```
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore InsecureBankv2.apk alias_name

Enter Passphrase for keystore:
   adding: META-INF/MANIFEST.MF
   adding: META-INF/ALIAS_NA.SF
   adding: META-INF/ALIAS_NA.RSA
  signing: resources.arsc
  signing: res/mipmap-mdpi/ic_launcher.png
  signing: res/anim/abc_slide_in_bottom.xml
  signing: res/anim/abc_slide_out_top.xml
  signing: res/anim/abc_fade_out.xml
  signing: res/anim/abc_slide_in_top.xml
  signing: res/anim/abc_grow_fade_in_from_bottom.xml
  signing: res/anim/abc_shrink_fade_out_from_bottom.xml
  signing: res/anim/abc_slide_out_bottom.xml
  signing: res/anim/abc_fade_in.xml
```



```
  signing: res/drawable-ldrtl-hdpi-v17/abc_ic_ab_back_mtrl_am_alpha.png
  signing: res/drawable-ldrtl-hdpi-v17/abc_ic_menu_cut_mtrl_alpha.png
  signing: res/drawable-ldrtl-hdpi-v17/abc_spinner_mtrl_am_alpha.9.png
  signing: AndroidManifest.xml
  signing: classes.dex
jar signed.
```

Patching android app

- Chúng ta đổi tên file để tránh nhầm lẫn:

mv InsecureBankv2.apk InsecureBankv2_patched.apk

- Và thử cài lại xem:

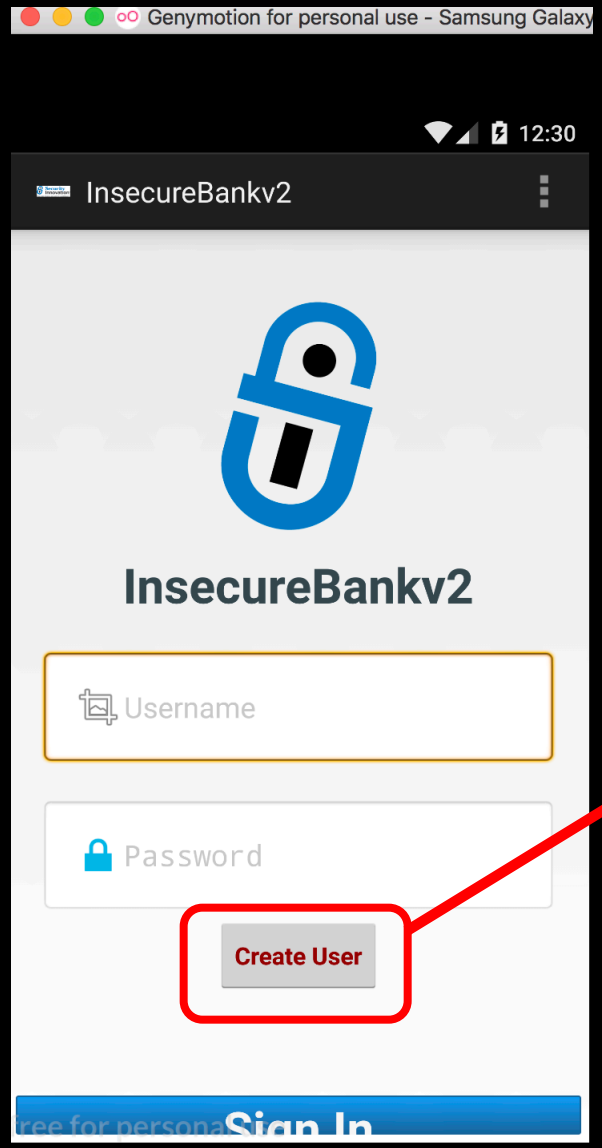
```
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ mv InsecureBankv2.apk InsecureBankv2_patched.apk
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ adb install InsecureBankv2_patched.apk
InsecureBankv2_patched.apk: 1 file pushed. 93.8 MB/s (3395957 bytes in 0.035s)
        pkg: /data/local/tmp/InsecureBankv2_patched.apk
Failure [INSTALL_FAILED_UPDATE_INCOMPATIBLE]
```

- Oh, vẫn có vấn đề! Lỗi này là do chương trình cũ đã được cài đặt rồi, do đó ta gỡ nó ra và cài cái mới vào

```
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ adb uninstall com.android.insecurebankv2
Success
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ ls
InsecureBankv2_patched.apk my-release-key.keystore
 ~/Desktop/mobile/tools/reverse/test/InsecureBankv2/dist/ adb install InsecureBankv2_patched.apk
InsecureBankv2_patched.apk: 1 file pushed. 92.4 MB/s (3395957 bytes in 0.035s)
        pkg: /data/local/tmp/InsecureBankv2_patched.apk
Success
```

Patching android app

- Okay, mở lên xem



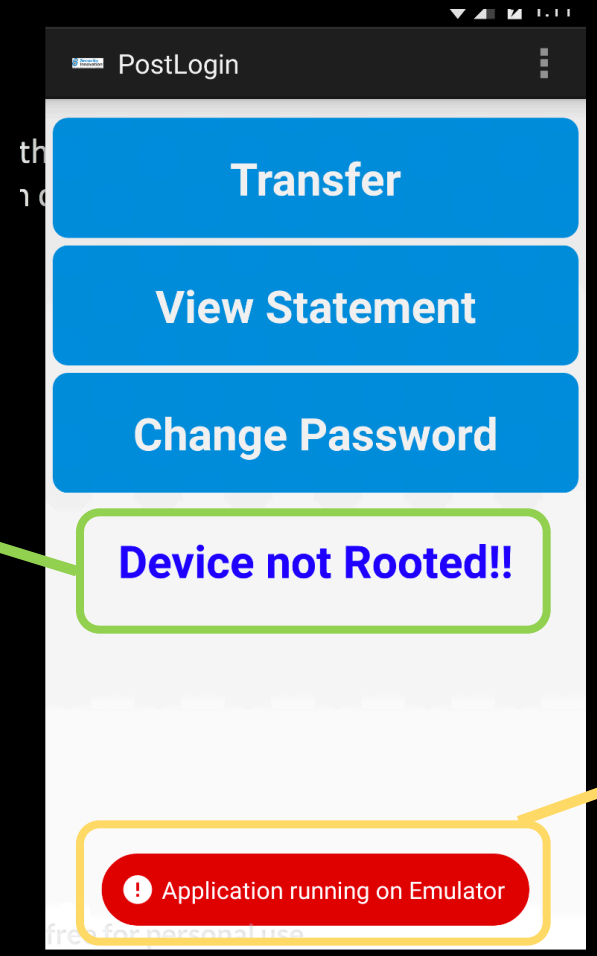
Button Create User xuất hiện nè,
cái này chỉ only admin mới thấy

Detection

- Chúng ta đi đến 1 phần quan trọng khác
- Chương trình hầu như sẽ có các đoạn code để kiểm tra để ngăn chặn kẻ tấn công phân tích, thường là:

1. Root Detection (or anti-root)
2. Emulator Detection (or anti-vm)

- Về lại app, ta thấy:

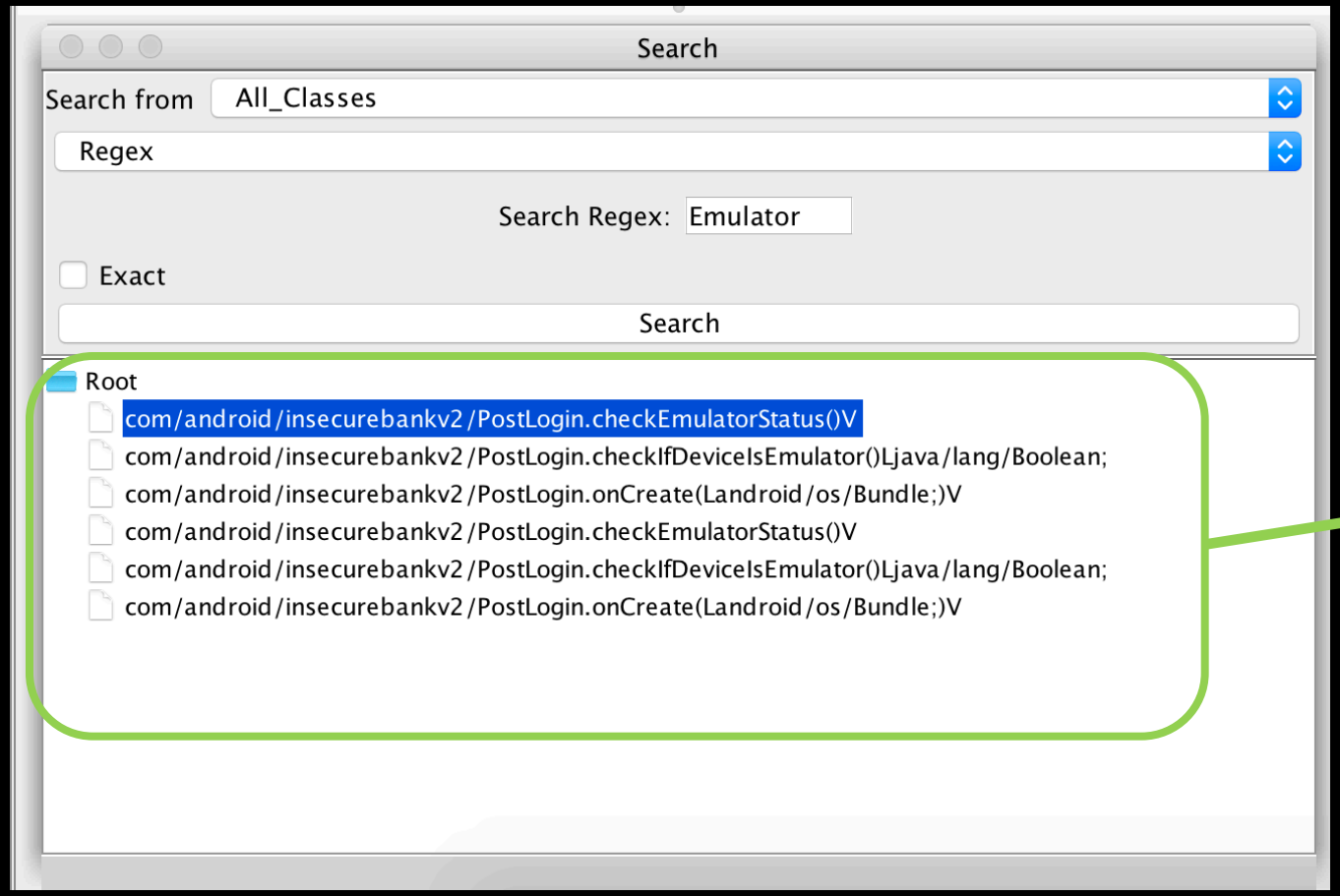


Có code check, nhưng mà check sai nên chúng ta không bị phát hiện, sẽ nói sau ☺

Phát hiện xài giả lập rồi nè, chúng ta đang xài galaxy s6 genymotion đúng không?

Detection -> Anti-emu

- Vậy bypass cái anti-emu trước. Dòng thông báo hiện ở phần PostLogin, nên chúng ta sẽ kiểm tra code ở đoạn đó
- Hoặc tìm kiếm cũng được => Mở Bytecode Viewer, load file apk và tìm chuỗi “Emulator”



Kết quả tìm!

Detection -> Anti-emu

- Chúng ta tìm được đoạn code trong com/android/insecurebankv2/PostLogin.class

```
private void checkEmulatorStatus()
{
  if (checkIfDeviceIsEmulator().booleanValue() == true)
  {
    Toasteroid.show(this, "Application running on Emulator", Toasteroid.STYLES.ERROR, 1);
    return;
  }
  Toasteroid.show(this, "Application running on Real device", Toasteroid.STYLES.SUCCESS, 1);
}

private Boolean checkIfDeviceIsEmulator()
{
  if ((!Build.FINGERPRINT.startsWith("generic")) && (!Build.FINGERPRINT.startsWith("unknown")) &&
    (!Build.MODEL.contains("google_sdk")) && (!Build.MODEL.contains("Emulator")) && (!Build.MODEL.contains("Android SDK built for x86")) &&
    (!Build.MANUFACTURER.contains("Genymotion")) && ((!Build.BRAND.startsWith("generic")) || (!Build.DEVICE.startsWith("generic"))) &&
    (!"google_sdk".equals(Build.PRODUCT))) {
    return Boolean.valueOf(false);
  }
  return Boolean.valueOf(true);
}
```

- Hàm checkEmulatorStatus() gọi hàm checkIfDeviceIsEmulator(), hàm này kiểm tra các chuỗi được định nghĩa trong FINGERPRINT, MODEL, MANUFACTURE, BRAND, DEVICE, etc… để phát hiện, với trường hợp chúng ta là: “Genymotion”
- Lần này không patch dễ như trước được, bởi vì đoạn kiểm tra nằm trong code, nếu muốn sửa thì phải sửa mã SMALI (yes, smali ☺).

Detection -> Anti-emu

- Như vậy bước một là tìm mã smali ứng với đoạn code java kiểm tra emulator
- Chúng ta đã biết hàm kiểm tra là checkEmulatorStatus(), tìm nó

```
.method private checkEmulatorStatus()V
    .registers 4

    const/4 v2, 0x1

    invoke-direct {p0}, Lcom/android/insecurebankv2/PostLogin;->checkIfDeviceIsEmulator()Ljava/lang/Boolean;

    move-result-object v0

    invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z

    move-result v0

    if-ne v0, v2, :cond_13

    const-string v0, "Application running on Emulator"

    sget-object v1, Lcom/marcohc/toasteroid/Toasteroid$STYLES;->ERROR:Lcom/marcohc/toasteroid/Toasteroid$STYLES;

    invoke-static {p0, v0, v1, v2}, Lcom/marcohc/toasteroid/Toasteroid;->show(Landroid/app/Activity;Ljava/lang/String;Lcom/marcohc/toasteroid/Toasteroid$STYLES;I)V

    :goto_12
    return-void

    :cond_13
    const-string v0, "Application running on Real device"

    sget-object v1, Lcom/marcohc/toasteroid/Toasteroid$STYLES;->SUCCESS:Lcom/marcohc/toasteroid/Toasteroid$STYLES;

    invoke-static {p0, v0, v1, v2}, Lcom/marcohc/toasteroid/Toasteroid;->show(Landroid/app/Activity;Ljava/lang/String;Lcom/marcohc/toasteroid/Toasteroid$STYLES;I)V

    goto :goto_12
.end method
```

- Lưu ý là tên biến, tên điều kiện rẽ nhánh, vv... Sẽ khác nhau tuỳ vào tool decompile

Detection -> Anti-emu

- Một kinh nghiệm nhỏ là, khi dịch ngược 1 phần mềm, không cần hiểu hết phần mềm làm gì, chỉ cần tìm ra điểm G, sửa nó là xong
- Điểm G ở đây là:

```
if-ne v0, v2, :cond_13

const-string v0, "Application running on Emulator"
```

```
:cond_13
const-string v0, "Application running on Real device"
```

- if-ne vx, vy, target == “Jumps to target if vx!=vy. vx and vy are integer values.”
- Đoạn code ở trên có thể diễn tả như sau:

Nếu v0 != v2, thì nhảy tới :cond_13 (:cond_13 là nhánh thiết bị thật => cái chúng ta muốn), nếu không thoả thì nhảy tới nhánh thiết bị mô phỏng

Detection -> Anti-emu

- Vậy bây giờ chúng ta cần phải hiểu rõ v0,v2 là gì, rồi làm 1 số ma thuật, năng lực siêu nhiên để thành thiết bị thật? No no, có 1 cách đơn giản hơn ☺
- Nhìn đoạn này nè:

```
if-ne v0, v2, :cond_13
```

If ... not equal ... jump to cond_13 (right part)

- Vậy điều gì sẽ xảy ra nếu ta bỏ đoạn "not equal" và bắt chương trình luôn nhảy về :cond_13?

=> luồng thực thi sẽ luôn trỏ về :cond_13 ☺

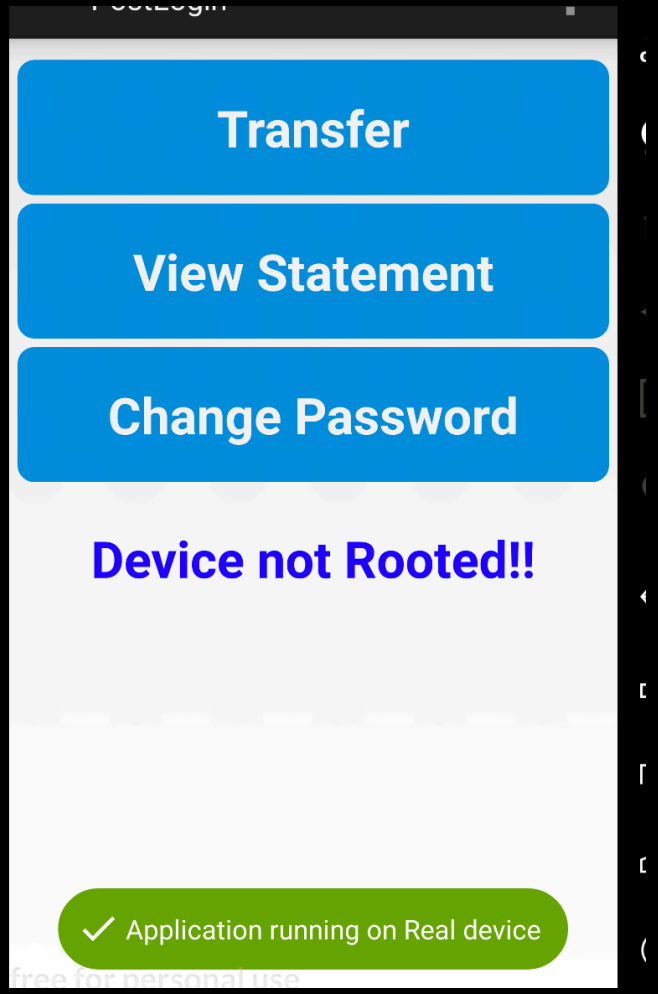
- Thử nào:

```
if-ne v0, v2, :cond_13
```

```
goto :cond_13
```

Detection -> Anti-emu

- Trên ý tưởng đó, thay đổi mã smali trong InsecureBankv2/smali/com/android/insecurebankv2/PostLogin.smali
- Rồi patch nó lại.
- Mở app ra và xem kết quả ☺



Detection -> Anti-root

- Chúng ta sẽ nói về vấn đề tại sao anti-root code không phát hiện được. Đoạn code check root như sau:

```
void showRootStatus()
{
  int i;
  if ((!doesSuperuserApkExist("/system/app/Superuser.apk")) && (!doesSUexist())) {
    i = 0;
  } else {
    i = 1;
  }
  if (i == 1)
  {
    this.root_status.setText("Rooted Device!!");
    return;
  }
  this.root_status.setText("Device not Rooted!!");
}
```

- Nó kiểm tra /system/app/Superuser.apk có tồn tại không?

Nếu có => phát hiện, không thì thôi.

Detection -> Anti-root

- Để xem chuyện gì đã xảy ra, chúng ta vào trong điện thoại ảo trước:

```
 ~/Desktop/mobile/tools/reverse/test/ adb shell
root@vbox86p:/ # uname -a
Linux localhost 4.4.10-genymotion #1 SMP PREEMPT Fri Oct 28 09:28:26 UTC 2016 x86_64 GNU/Linux
```

- Rồi tìm nó thử coi nè:

```
root@vbox86p:/ # ls -la /system/app/Superuser.apk
/system/app/Superuser.apk: No such file or directory
```

Không có, cho nên mới qua được phần kiểm tra

- Thông tin thêm: cái nó cần kiểm tra nằm ở đây ☺

```
root@vbox86p:/ # ls -la /system/app/Superuser/
-rw-r--r-- root     root       927685 2018-05-21 20:16 Superuser.apk
drwxr-xr-x root     root              2018-05-21 20:16 x86
```