BÁO CÁO THỰC HÀNH

**Môn học: Bảo mật Web và ứng dụng**

**Lab 4: Pentesting Android Applications**

*GVHD: Ngô Đức Hoàng Sơn*

1. **THÔNG TIN CHUNG:**

*(Liệt kê tất cả các thành viên trong nhóm)*

Lớp: NT132.O12.ATCL.2- Nhóm 4

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Đỗ Thị Yến Ly | 21520337 | [21520337@gm.uit.edu.vn](mailto:21520337@gm.uit.edu.vn) |
| 2 | Lê Hoàng Oanh | 21521253 | 21521253@gm.uit.edu.vn |
| 3 | Nguyễn Đại Bảo Duy | 21520772 | [21520772@gm.uit.edu.vn](mailto:21520772@gm.uit.edu.vn) |

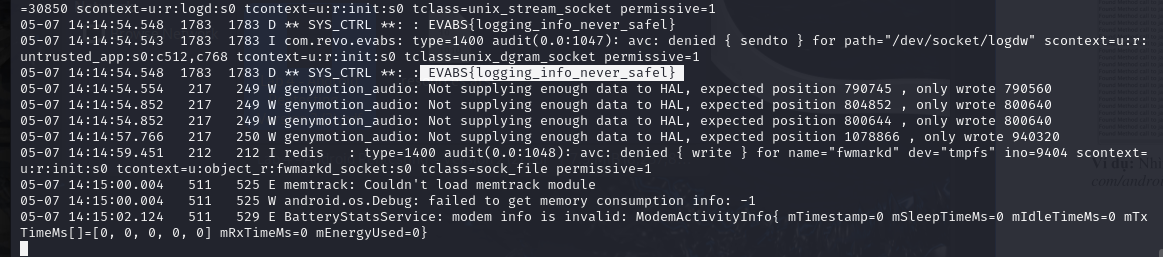
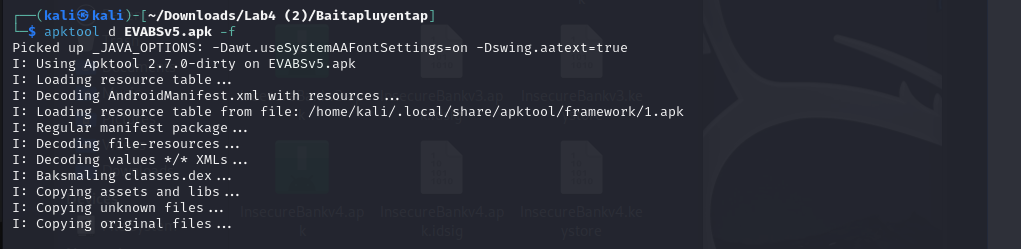
1. **NỘI DUNG THỰC HIỆN:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Kết quả tự đánh giá** |
| 1 | **Yêu cầu 1** | 100% |
| 2 | **Yêu cầu 2** | 100% |

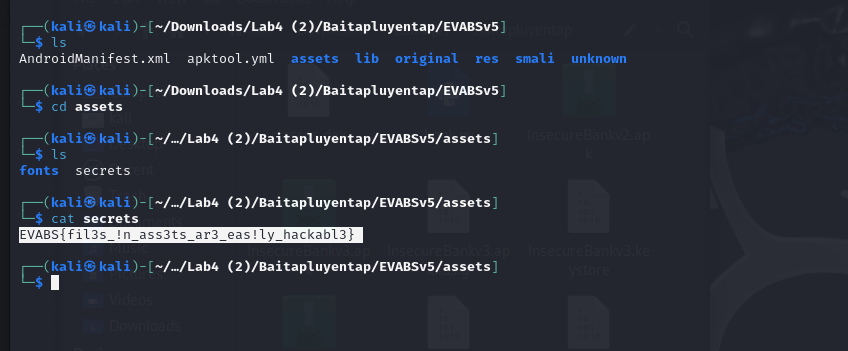
**Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.**

BÁO CÁO CHI TIẾT

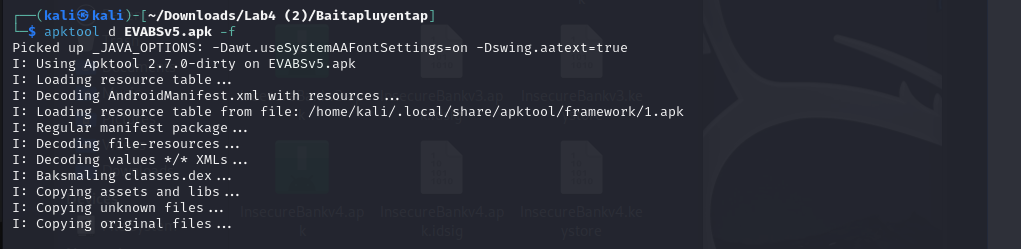
1. **EVABS**
2. **Challenge 1: Debug Me**

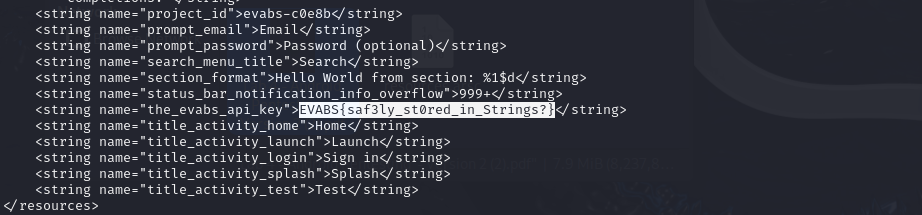
Sử dụng câu lệnh ”logcat” để bắt log từ ứng dụng => lấy được flag

1. **Challenge 2: File Access**

Hint liên quan đến ”assets” => tìm kiếm trong thư mục sau khi decode ứng dụng và lấy được flag

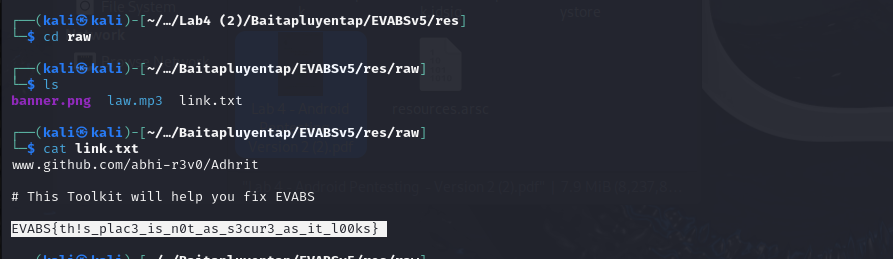
1. **Challenge 3: Strings**

Thực hiện decode ứng dụng



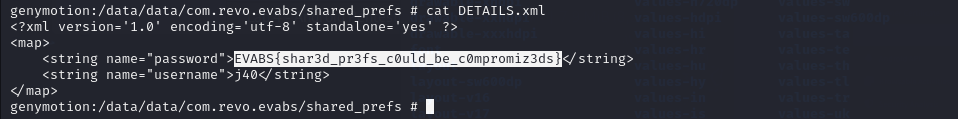
Hint đề cập đến các chuỗi string => tìm kiếm nơi mà lưu trữ giá trị của các chuỗi string (tại res/values) và có được flag

1. **Challenge 4: Resources**



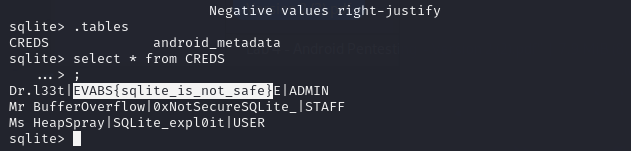
Tương tự challenge 3 => tìm kiếm trong các thư mục lưu trữ key và có được flag

1. **Challenge 5: Shares and Prefs**



Với challenge này ta sẽ truy cập vào dữ kiệu của ứng dụng lưu trên máy local => có được flag

1. **Challenge 6: DB Leak**



Tương tự như challenge 5, ta sẽ tìm kiếm dữ liệu được lưu tại cơ sở dữ liệu của máy và ta có được flag

1. **Challenge 7: Export**

A screenshot of a computer

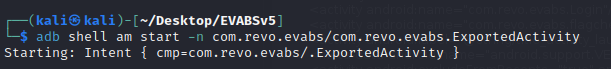
Description automatically generated

Khi kiểm tra manifest, ở ngay dòng thứ 4 trong đã có thông tin về một Activity bị exported:



Khi một activity bị exported, chúng ta có thể khởi động nó bằng adb. Sử dụng adb để trigger các exported activity bằng lệnh:

*adb shell am start -n com.revo.evabs/com.revo.evabs.ExportedActivity*



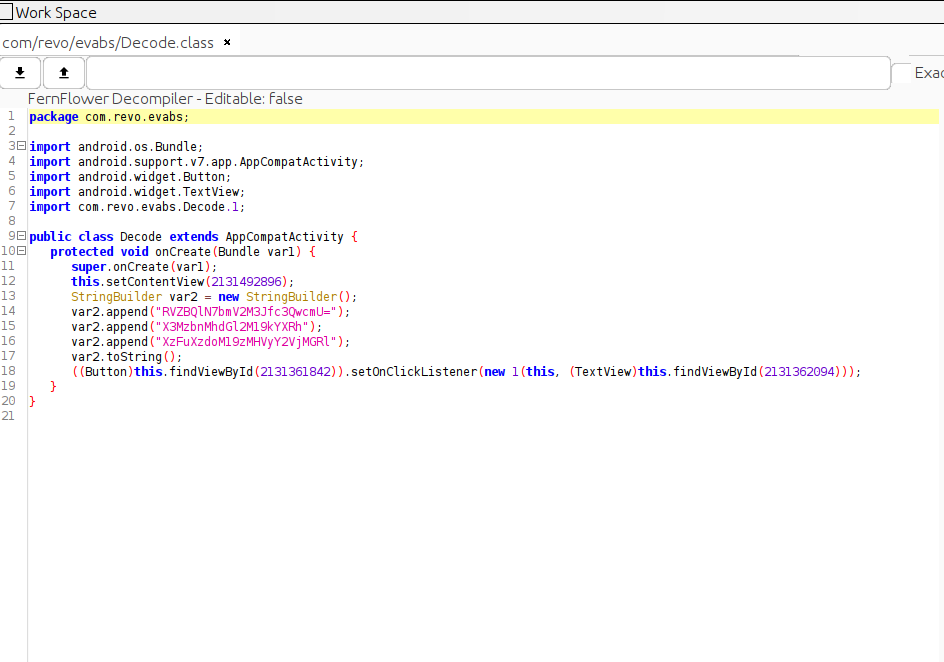
Ta thấy trong thiết bị android xuất hiện flag

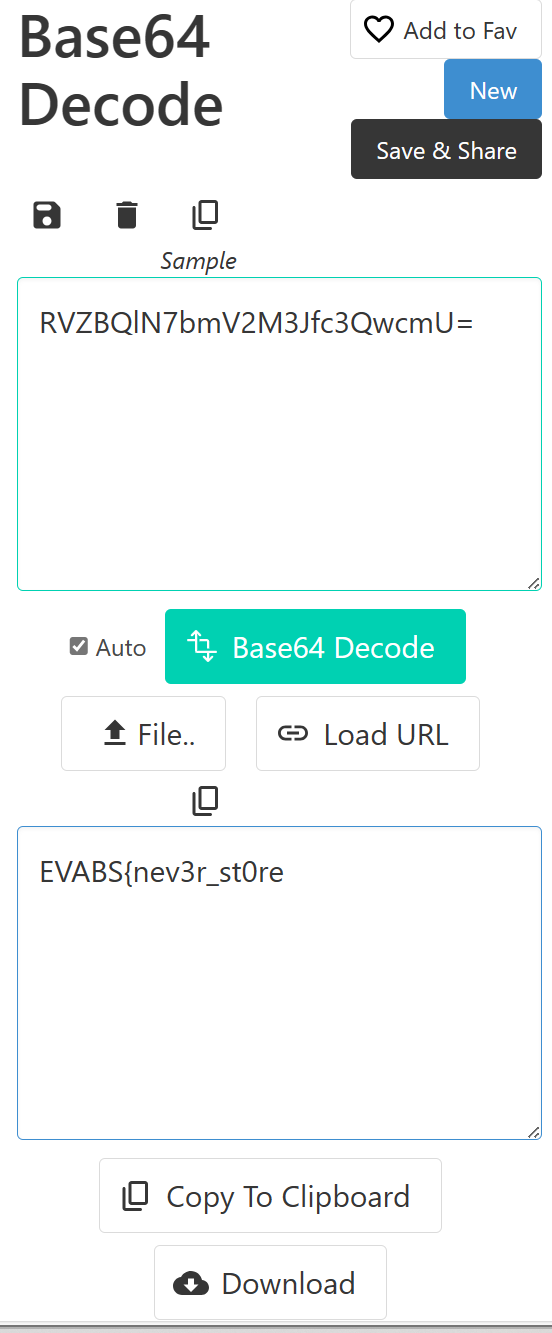
A screenshot of a computer

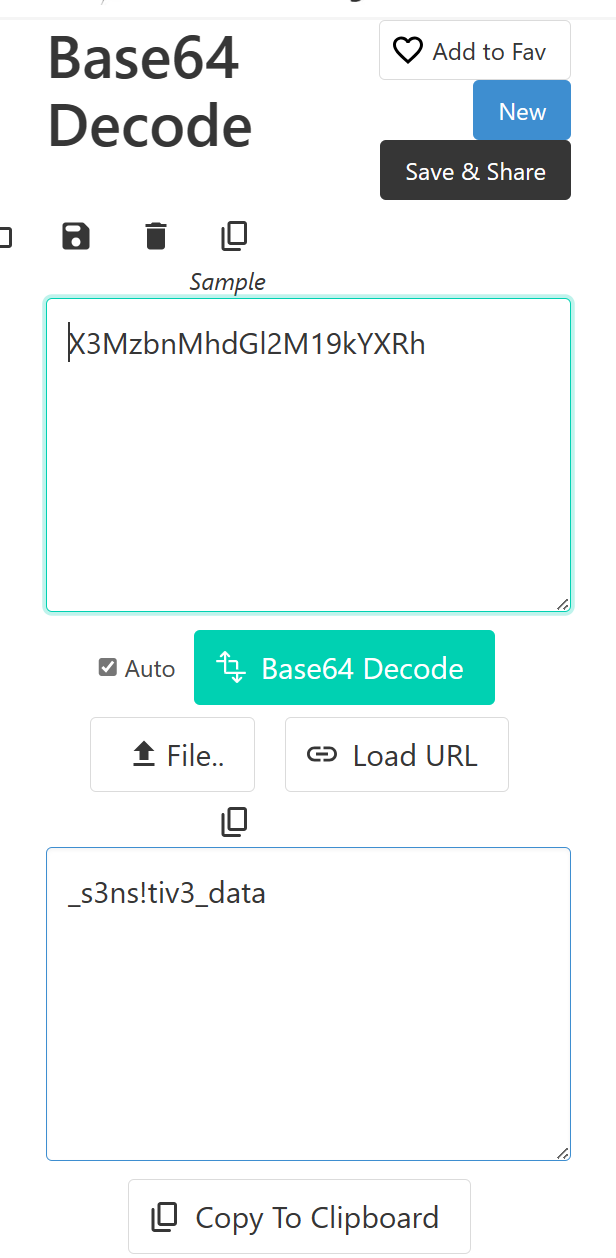
Description automatically generated

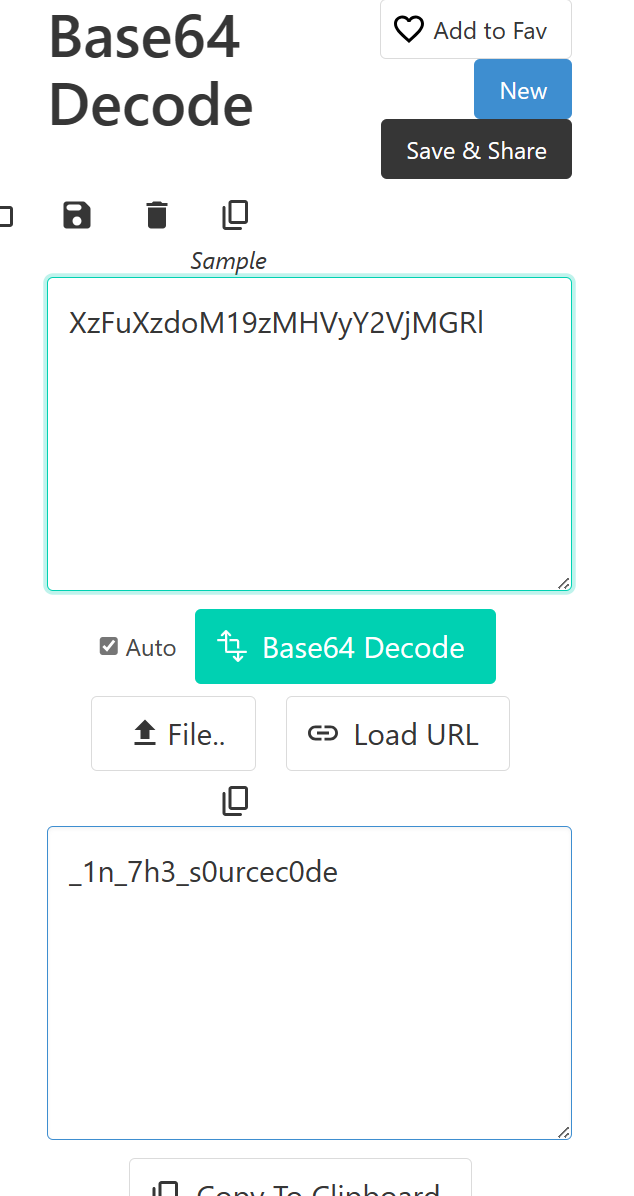
Flag: EVABS{exp0rted\_activities\_ar3\_harmful}

1. **Challenge 8: Decode**



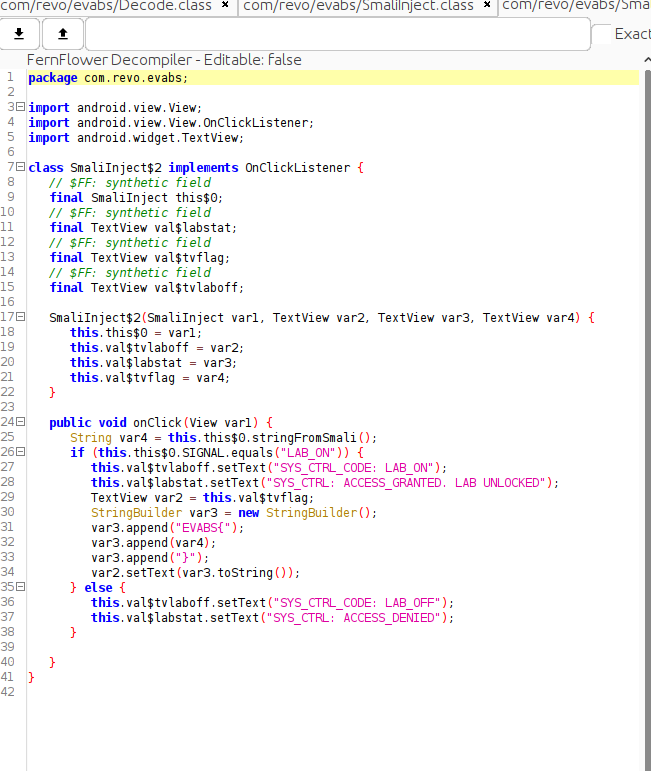
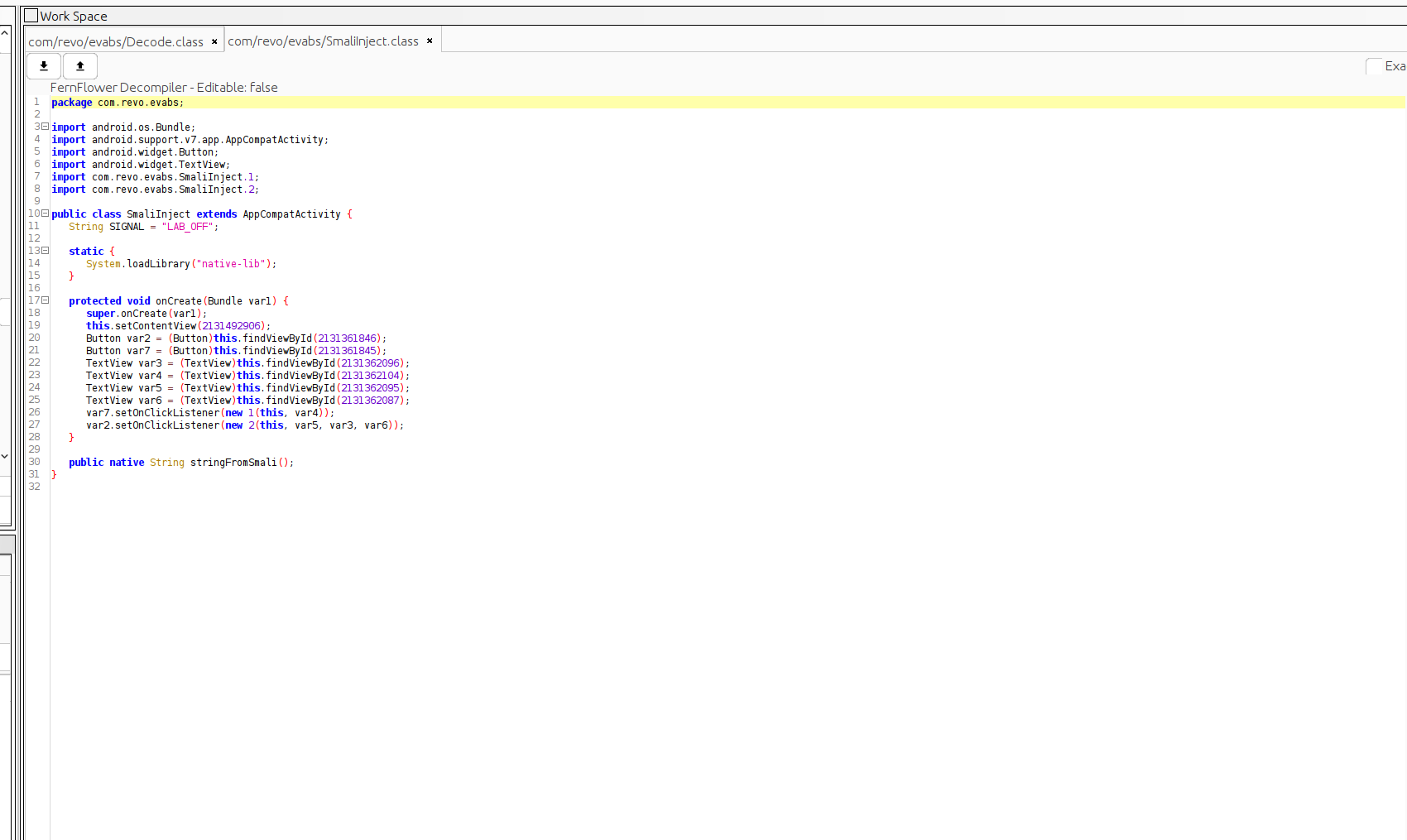


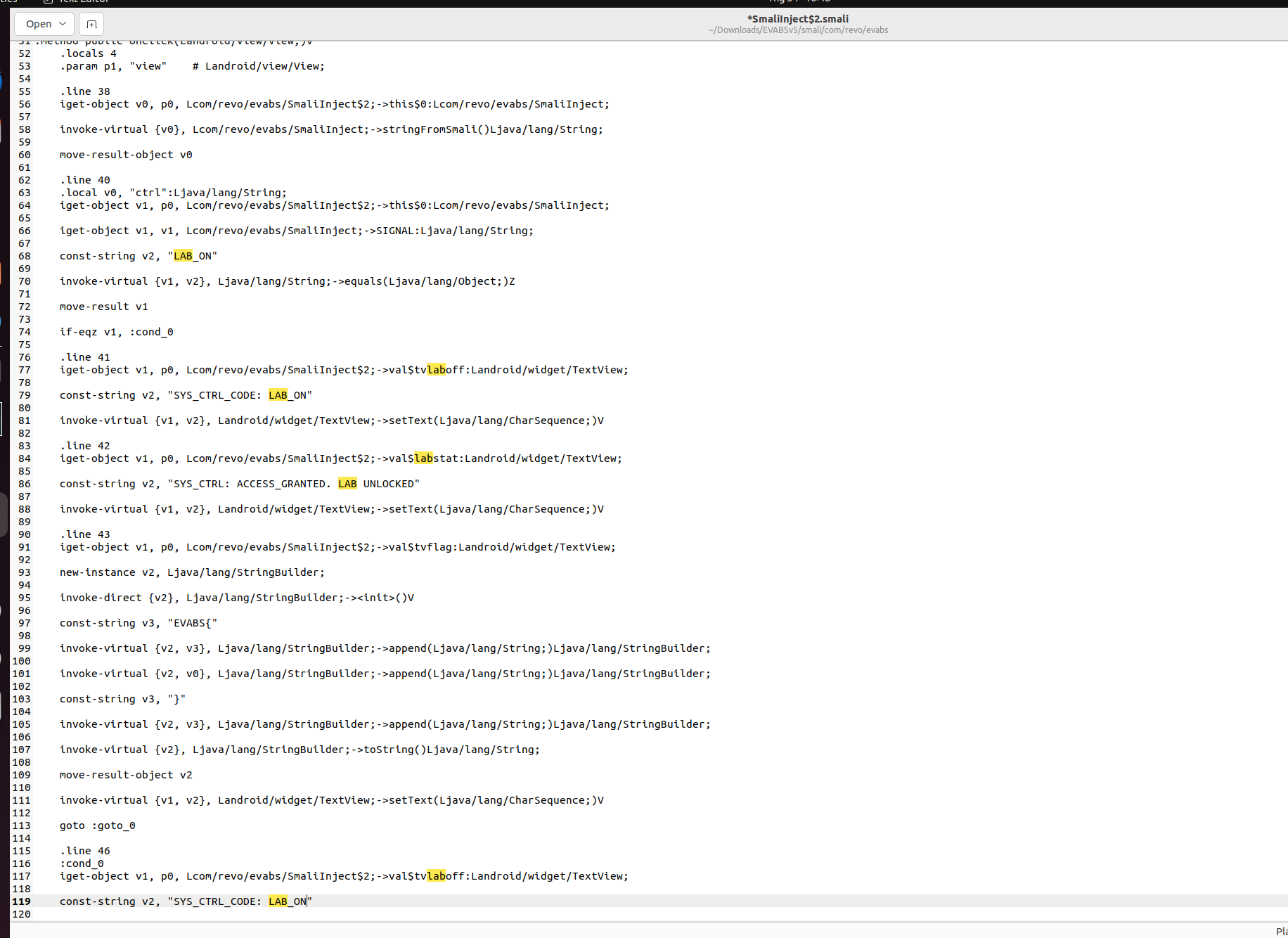




EVABS{nev3r\_st0re \_s3ns!tiv3\_data \_1n\_7h3\_s0urcec0de}

1. **Challenge 9: Smali Injection**





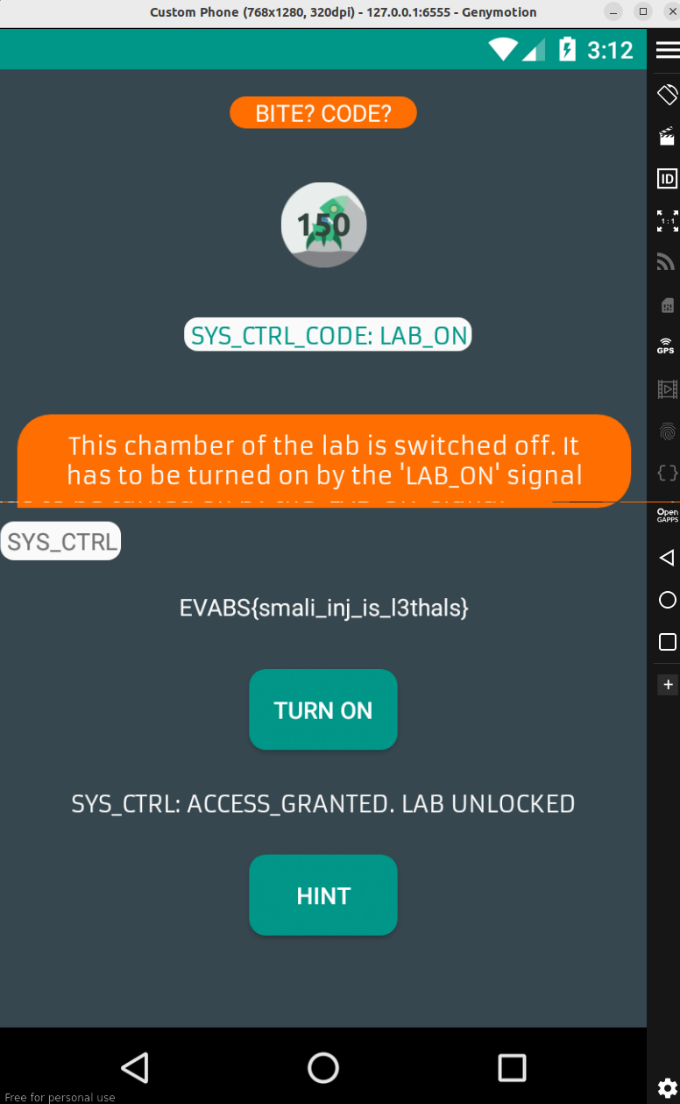
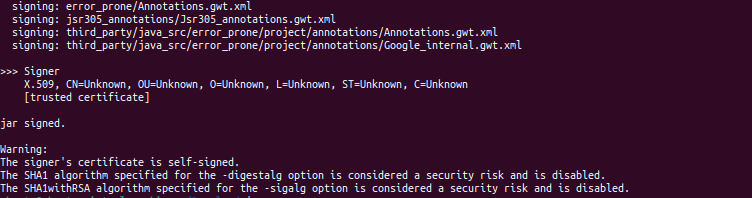
* Ta thấy cmt dòng “if-eqz” sẽ làm cho nó luôn in cờ

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated



1. **Challenge 10: Interception**

**A screenshot of a chat

Description automatically generated**

Configure BurpSuite để lắng nghe port 8083 trong mục Proxy.

A screenshot of a computer

Description automatically generated

Cấu hình máy Android để sử dụng Proxy. Nhập IP máy chính mà chạy BurpSuite và nhập port đã được cấu hình trong BurpSuite listener

A screenshot of a computer

Description automatically generated

Chọn Export CA certificate trong BurpSuite để xuất chứng chỉ CA dưới định dạng DER. Ta lưu nó là 2.der

A screenshot of a certificate

Description automatically generated

Android muốn chứng chỉ ở định dạng PEM, và có tên tệp tin bằng giá trị subject\_hash\_old được thêm vào với phần mở rộng .0

Sử dụng openssl để chuyển đổi từ định dạng DER sang PEM, sau đó xuất giá trị subject\_hash\_old và đổi tên tệp tin.

A black screen with white text

Description automatically generated

Sao chép chứng chỉ đến thiết bị. Chúng ta có thể sử dụng adb để sao chép chứng chỉ, nhưng vì nó phải được sao chép vào /system, chúng ta phải remount nó thành writable. Với quyền root, điều này dễ dàng với adb remount.  
Sau đó, chỉ cần mở một shell (adb shell) và di chuyển tệp tin vào /system/etc/security/cacerts và đặt quyền chmod là 644.

Cuối cùng, chúng ta phải khởi động lại thiết bị hoàn toàn bằng cách sử dụng lệnh adb reboot.

A computer screen shot of a blue screen

Description automatically generated

Sau khi thiết bị khởi động lại, truy cập vào Settings -> Security -> Trusted Credentials để xem chứng chỉ mới "Portswigger CA" đã được thêm vào như một CA được tin cậy hệ thống

A screenshot of a computer

Description automatically generated

Ta bật Intercept trong BurpSuite và ta nhấn nút Receive trên màn hình của challenge thì ta thấy bắt được gói tin.

A screenshot of a computer

Description automatically generated

Ta gửi gói tin đã bắt được vào Repeater và nhấn Send thì ta không thấy hiện flag mà chỉ hiện rat trang sau.

A screenshot of a computer

Description automatically generated

Sau khi tìm hiểu thì có vẻ trang web host cho lab này đã hết hạn nên không truy cập được nữa

A screenshot of a computer screen

Description automatically generated

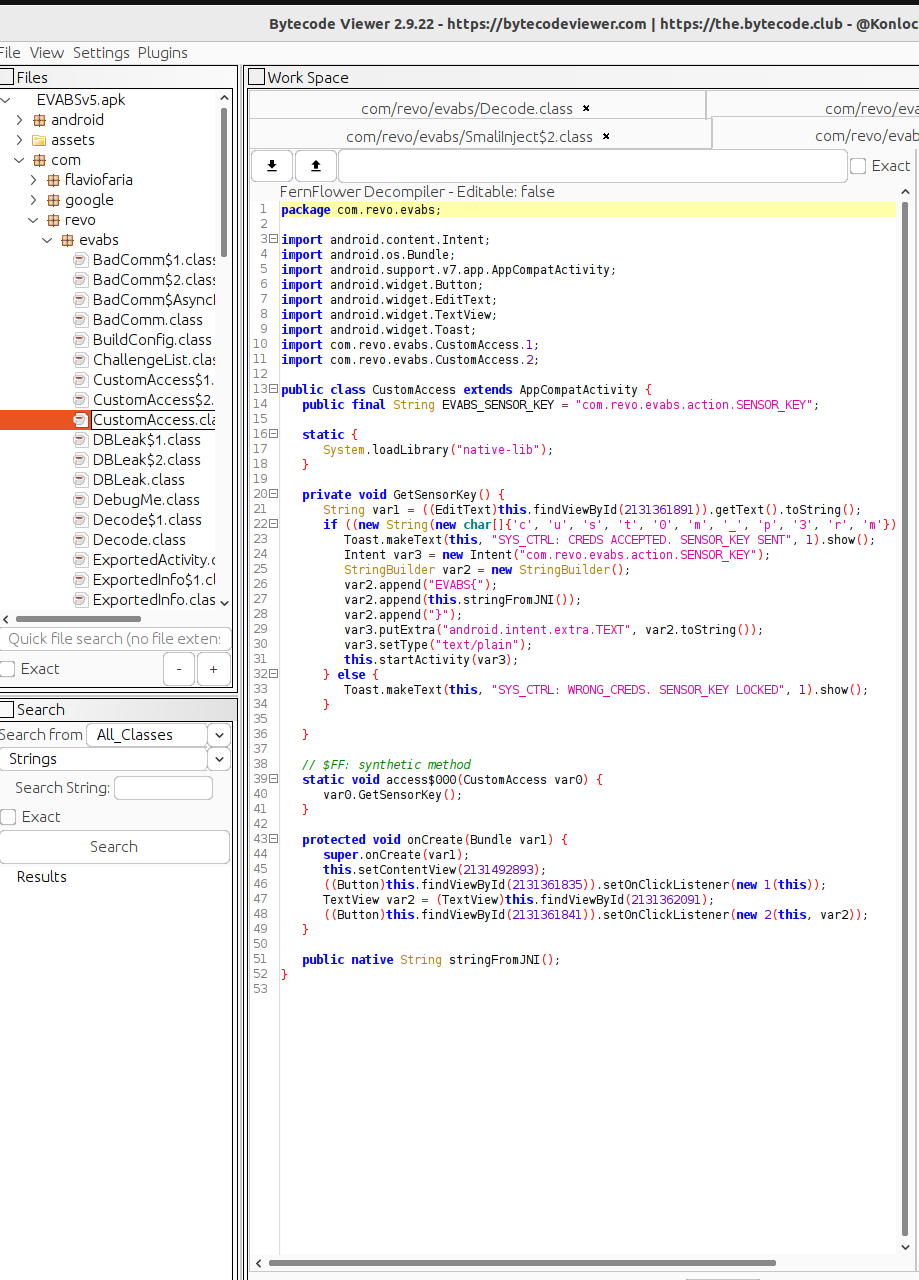
1. **Challenge 11: Custom Access**

Level này yêu cầu ta nhập input vào.

A screenshot of a computer

Description automatically generated

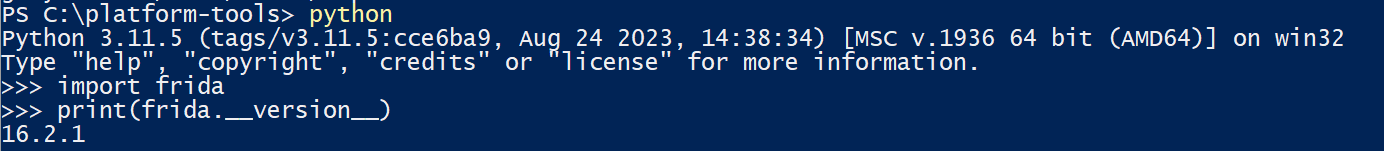
Đọc source trên bytecode viewer:



Ta thấy input ở đây là chuỗi “cust0m\_p3rm”, nếu nhập đúng input thì flag sẽ được tạo ra vào truyền vào intent com.revo.evabs.action.SENSOR\_KEY bằng hàm putExtra().

Ta có thể hook hàm putExtra() bằng Frida để nó in ra flag.

Tiếp theo, ta cài đặt frida trên win và android với win là frida 16.2.1 nằm trong bộ frida-tools và drida server 16.2.1 trên android. Kiểm tra sau khi cài đặt xong



A blue screen with white text

Description automatically generated



Tiếp tục ta sẽ code python để hooking code, code java sẽ thực hiện để lấy flag

A screen shot of a computer program

Description automatically generated

Sau đó chạy code python, đồng thời trên mobile mở ứng dụng và nhập cust0m\_p3rm

A screen shot of a computer code

Description automatically generated

Flag: EVABS{always\_ver1fy\_packag3sa}

1. **Challenge 12: Instrument**

Khi click "MAP AREA" thì sẽ xuất hiện 2 tọa độ x và y cùng với 1 giá trị bằng x \* y

**A screenshot of a computer

Description automatically generated**

Khi đọc code challenge này ở Frida1.class

A computer screen shot of a program

Description automatically generated

Flag sẽ được in ra nếu (x = a \* b) > i + 150 với i là 1 số int random trong khoảng 0 -> 70. Vì a, b cố định khiến x luôn là 50, như thế thì i random kiểu gì thì x cũng không thỏa mãn được.

Lúc này ta cần hook vào hàm random sao cho để điều kiện trên luôn đúng, ta cho giá trị random là -200.

A screen shot of a computer program

Description automatically generated

Chạy frida server trong máy android



Chạy file code ở máy thật



Ta nhấn vô Map Area thì nó sẽ hiện ra dòng chữ sau, nghĩa là đã chạy thành công

A screenshot of a phone

Description automatically generated

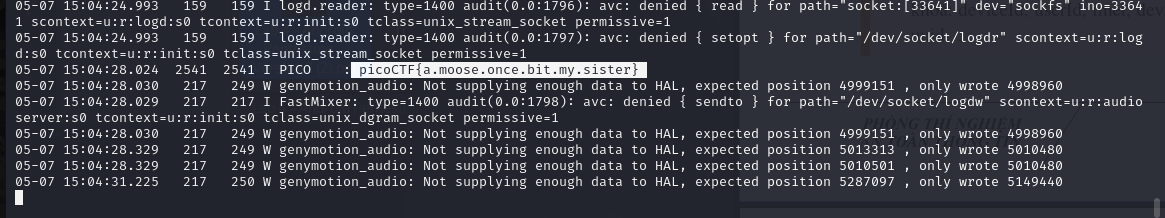
Dùng adb logcat để xem log thì ta thấy flag

A computer screen shot of a blue screen

Description automatically generated

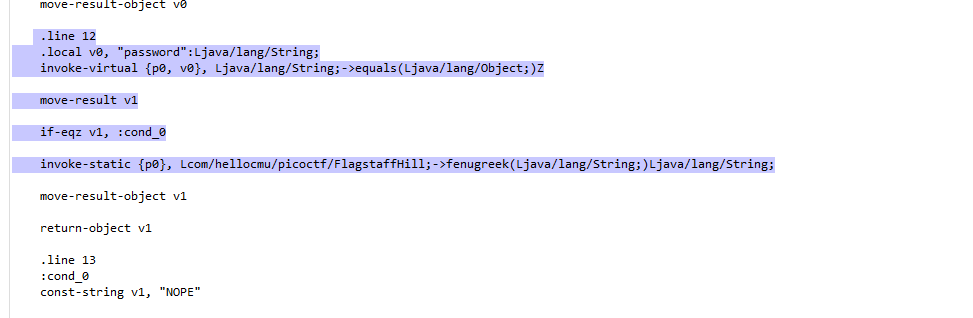
Flag: EVABS{a\_dynam1c\_h00k}

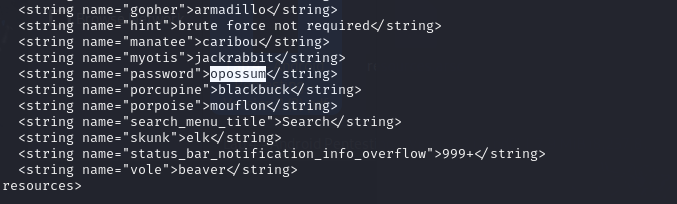
1. **PicoCTF**
2. **Challenge 1:**



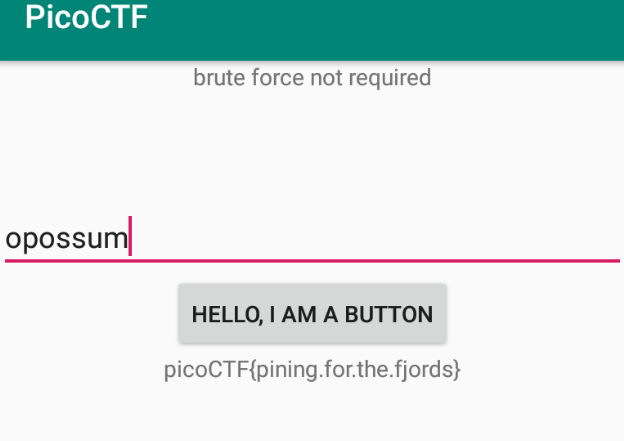
**Giống với challenge 1 phần 1, ta sử dụng “logcat” để bắt các log và có được flag**

1. **Challenge 2:**

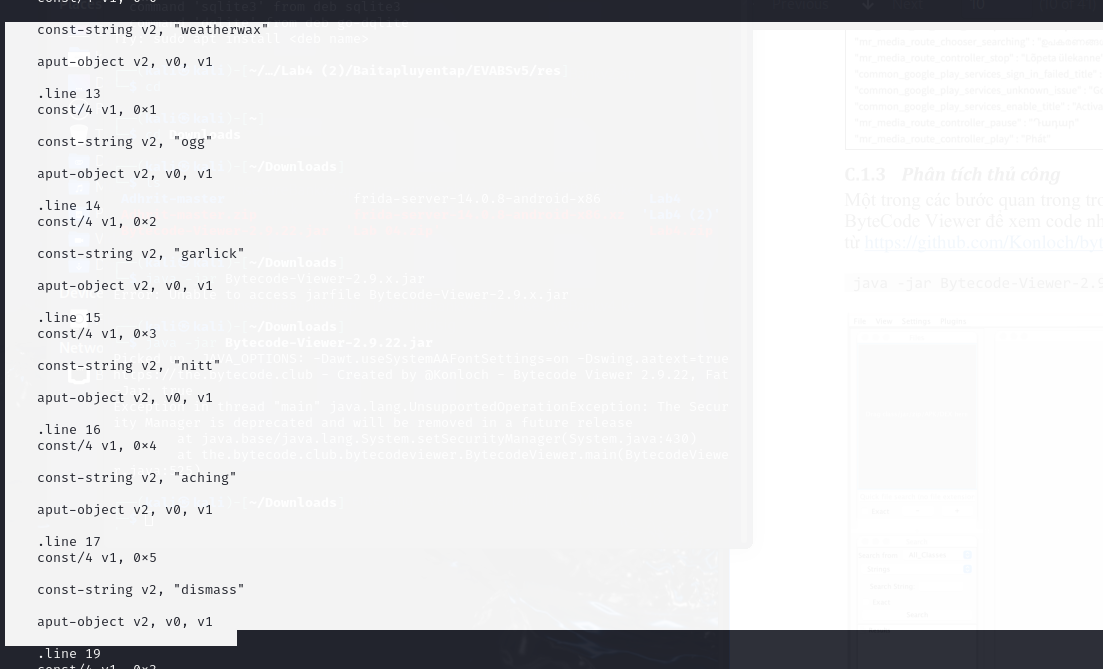
**Đoạn mã ở đây cho ta thấy phải nhập đúng mật khẩu thì mới lấy được flag => phải tìm kiếm mật khẩu được lưu ở đâu**



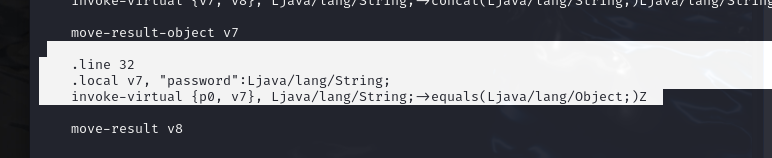
Tìm được mật khẩu là “opossum”



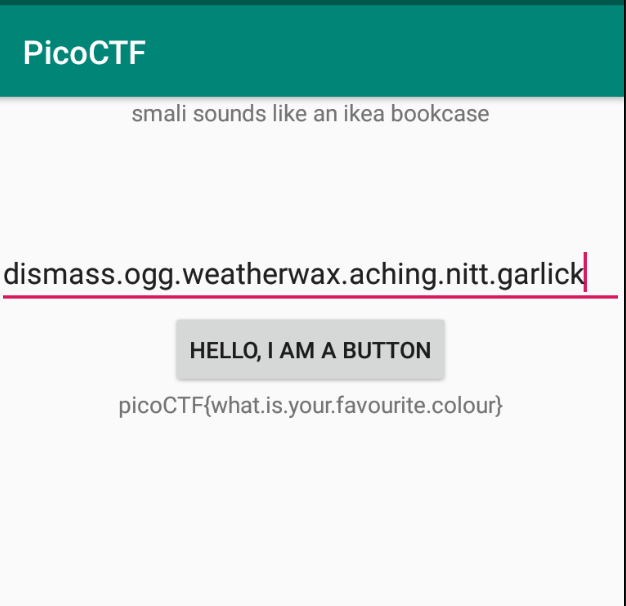
1. **Challenge 3:**



Ở đây là một mảng gồm nhiều chuỗi string khác nhau



Tương tự như challenge trên ta phải nhập một chuỗi string hợp lệ thì mới lấy được flag

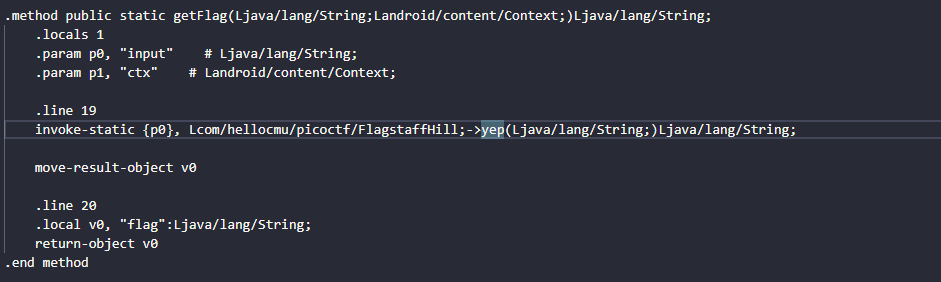


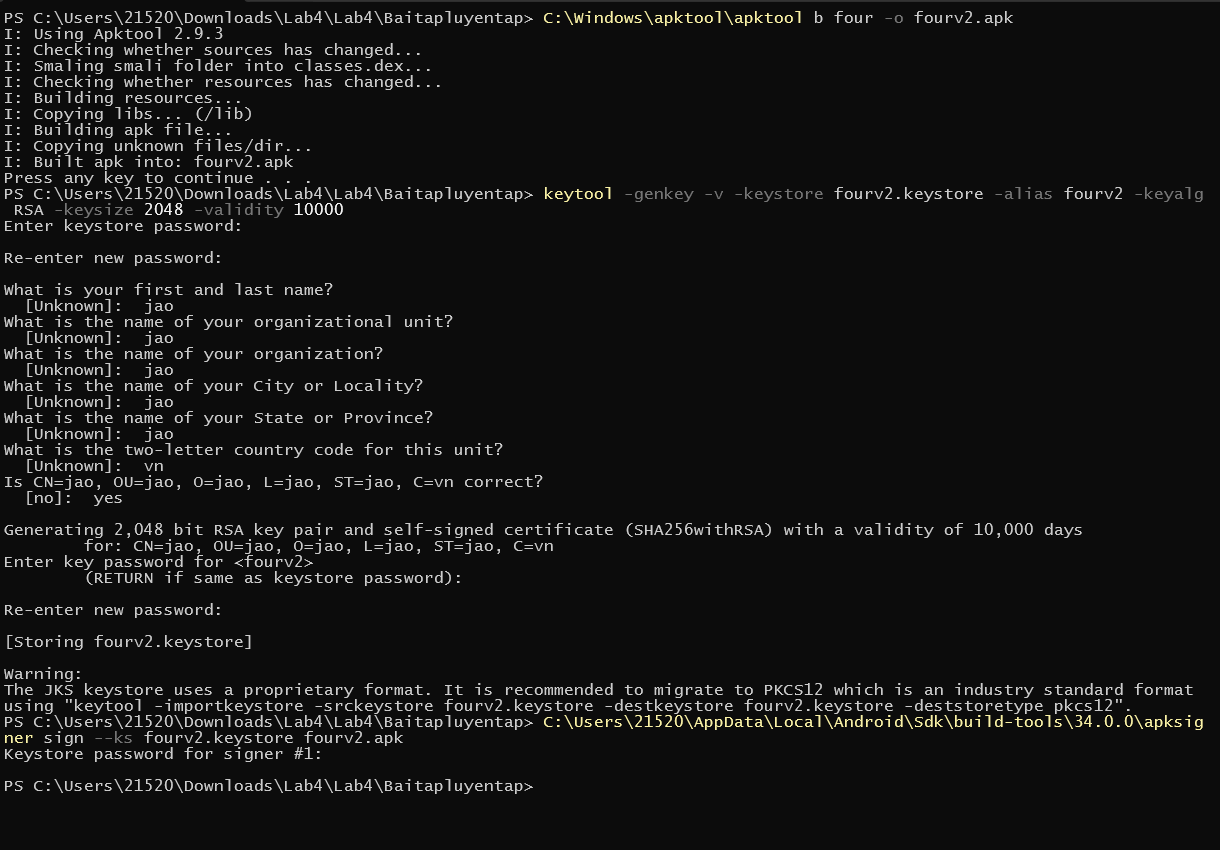
Sau khi sắp xếp lại thì ta lấy được flag

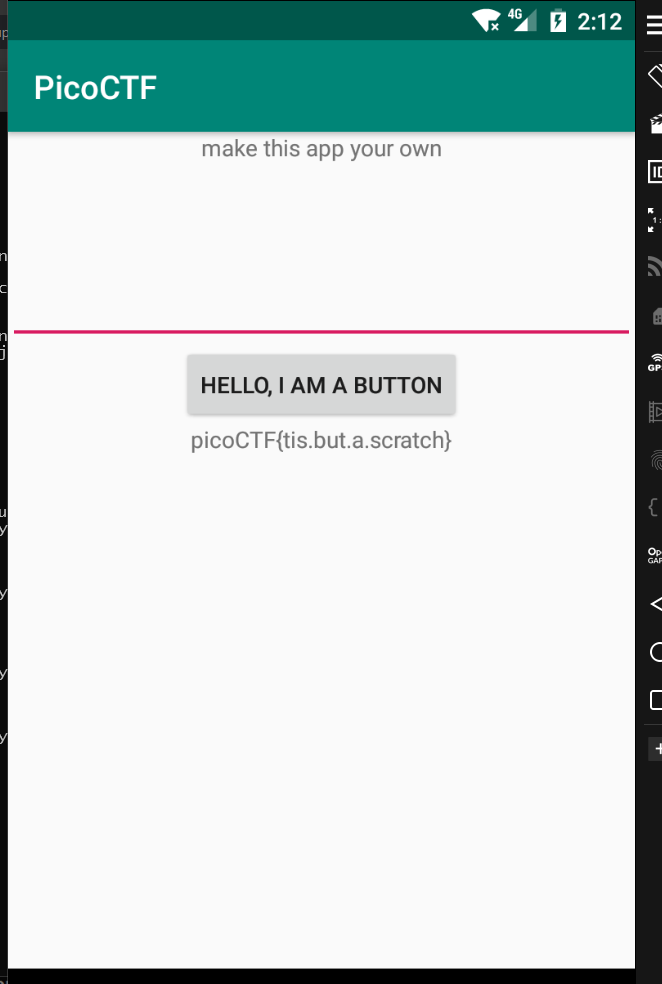
1. **Challenge 4:**



Theo như đoạn mã trên thì hàm getFlag đang return none, để lấy được flag thì phải sửa lại giá trị trả về của nó là true, ở đây sẽ là yep

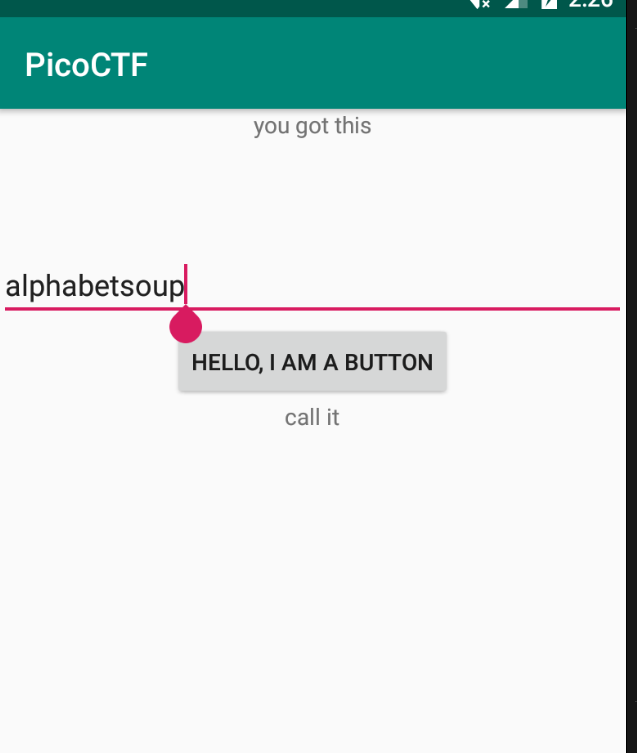
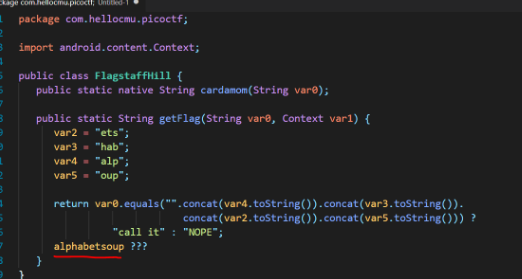


Sau đó ta sẽ thực hiện build lại chương trình và kí chữ kí

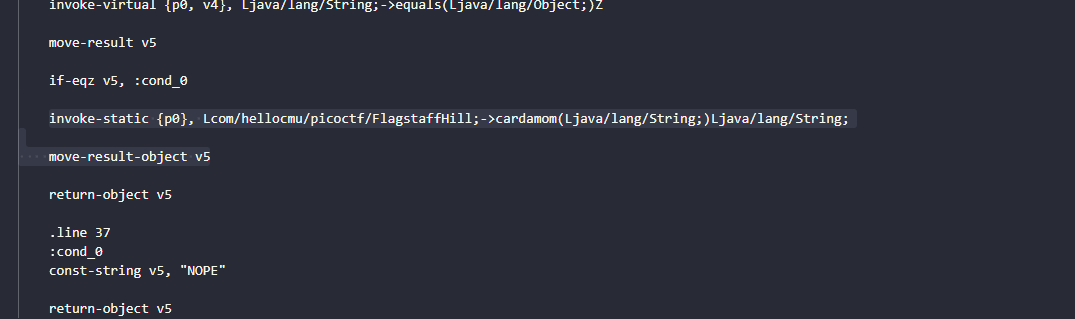


Cài đặt lại chương trình và lấy được flag

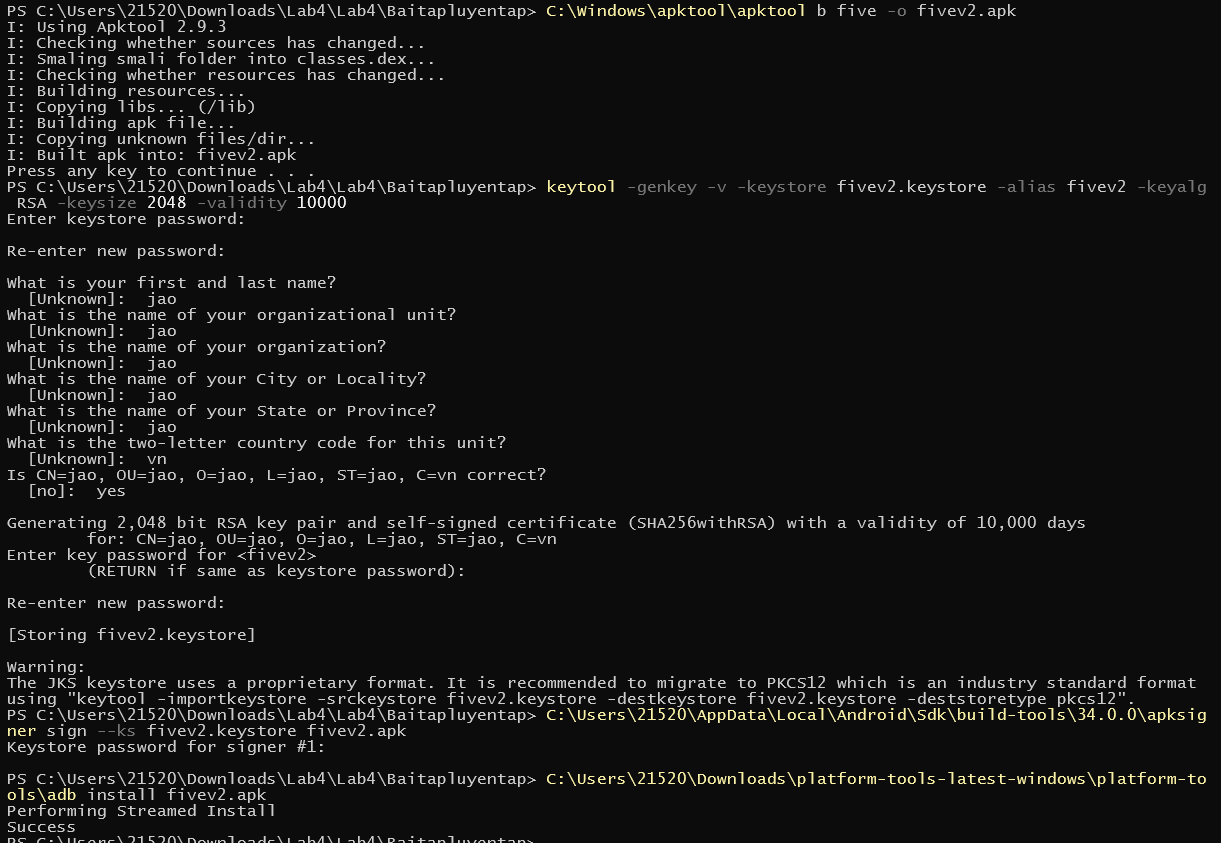
1. **Challenge 5:**



Dựa theo đoạn mã trên thử nhập chuỗi “alphabetsoup” vào thì thấy nó báo “call it” nghĩa là phải gọi một cái gì đó mới lấy được flag - có thể là hàm, biến hoặc một địa chỉ.

Có một hàm “cardamom(String var0)” không được gọi, mặc dù hàm đó cũng nhận input giống hàm getFlag nên ta sẽ sửa lại code để gọi được hàm đó 

Sửa “ const-string v5, "call it" “ thành đoạn mã gọi hàm cardamon được tô ở trên



Tiến hành build lại chương trình và kí chữ kí

