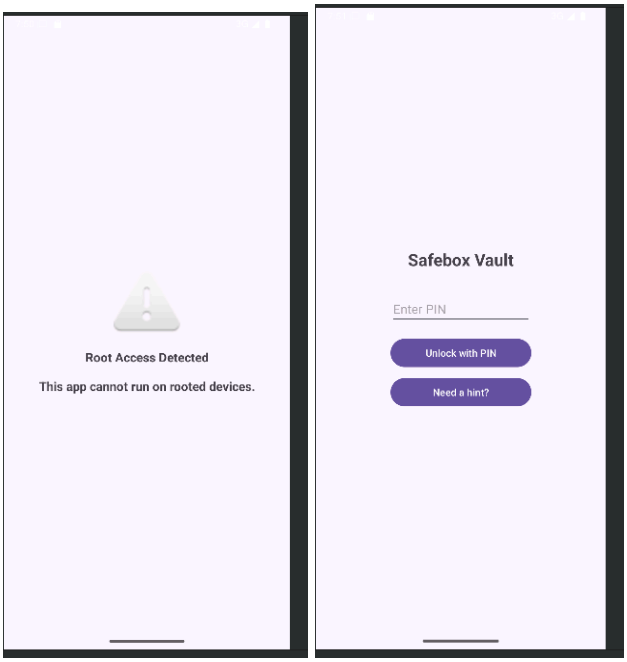
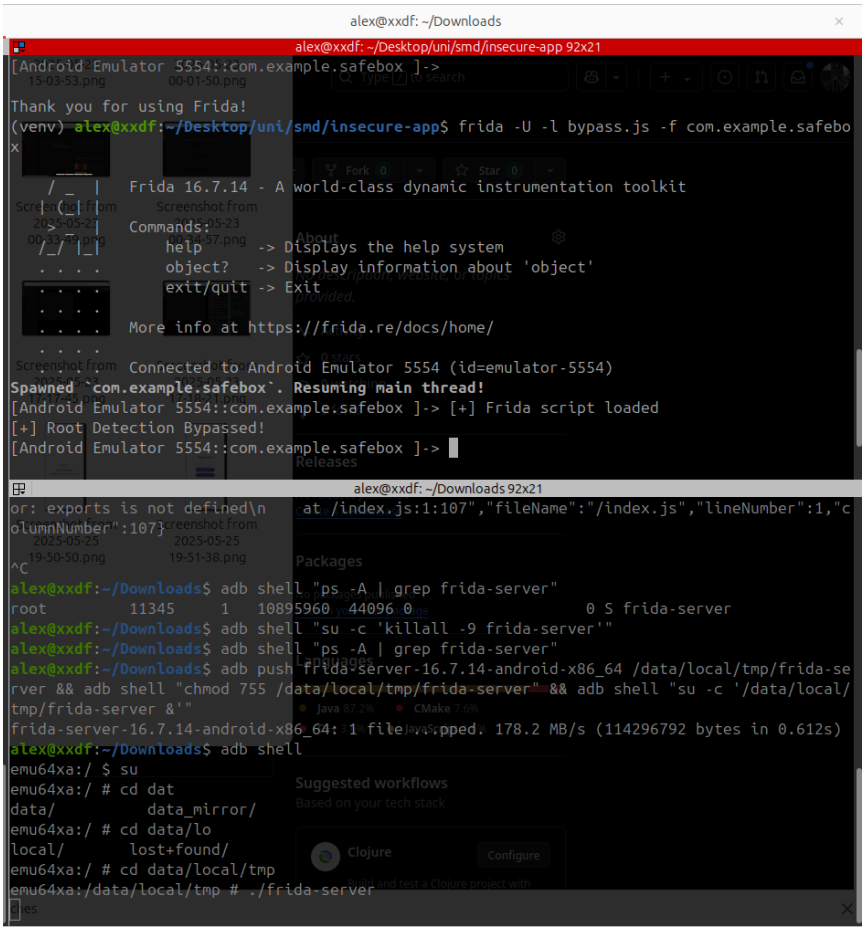
Solutions
Source code @ https://github.com/stefanpantucu/insecure-app

Challenge 1
Use frida to bypass root detection





See bypass.js from the github repo.

Challenge 2

Get the pin from external storage. Can be done via this commands:

adb pull /storage/emulated/0/Android/data/com.example.safebox/files/vault_pin.txt
tail -n 1 vault_pin.txt | base64 -d

Format of the file:
This file contains sensitive data
DO NOT MODIFY OR DELETE
Format: Base64(VAULT_ACCESS_CODE:####:END)
VkFVTFRfQUNDRVNTX0NPREU6NDU5NTpFTkQ=

Challenge 3

See source code for challenge 3. Check the functions for pressing each button.

- Pressing Right decreases the Down counter if it's greater than 0
- Pressing Down decreases the Up counter if it's greater than 0
- Pressing Left decreases the Right counter if it's greater than 0
- Pressing Up decreases the Left counter if it's greater than 0
- up counter must equal leftTouch (which is 1)
- left counter must equal downTouch (which is 2)
- down counter must equal rightTouch (which is 5)
- right counter must equal upTouch (which is 1)

Solution: 3up, 1 right, 5 down, 2 left, 4 right (these are the names on the buttons)

Challenge 4

In the validateFlag method, the app checks the user's input against the actual flag by XORing each byte of the OBFUSCATED_FLAG with a single XOR_KEY.
This means the original flag can be revealed by XORing each obfuscated byte with the key 0x5A.
Flag is: La lautari

Challenge 5:
The flag validation is no longer done in Java. Instead, it’s being handled by a method from a custom Native class - a **JNI (Java Native Interface)** call to **native C/C++ code**.
The flag is hardcoded in the cpp file (or can be seen with ghidra):

```
static const int correct_flag[] = {
   70, 108, 111, 114, 105, 110, 32, 83, 97, 108, 97, 109
};
```

Which is: Florin Salam