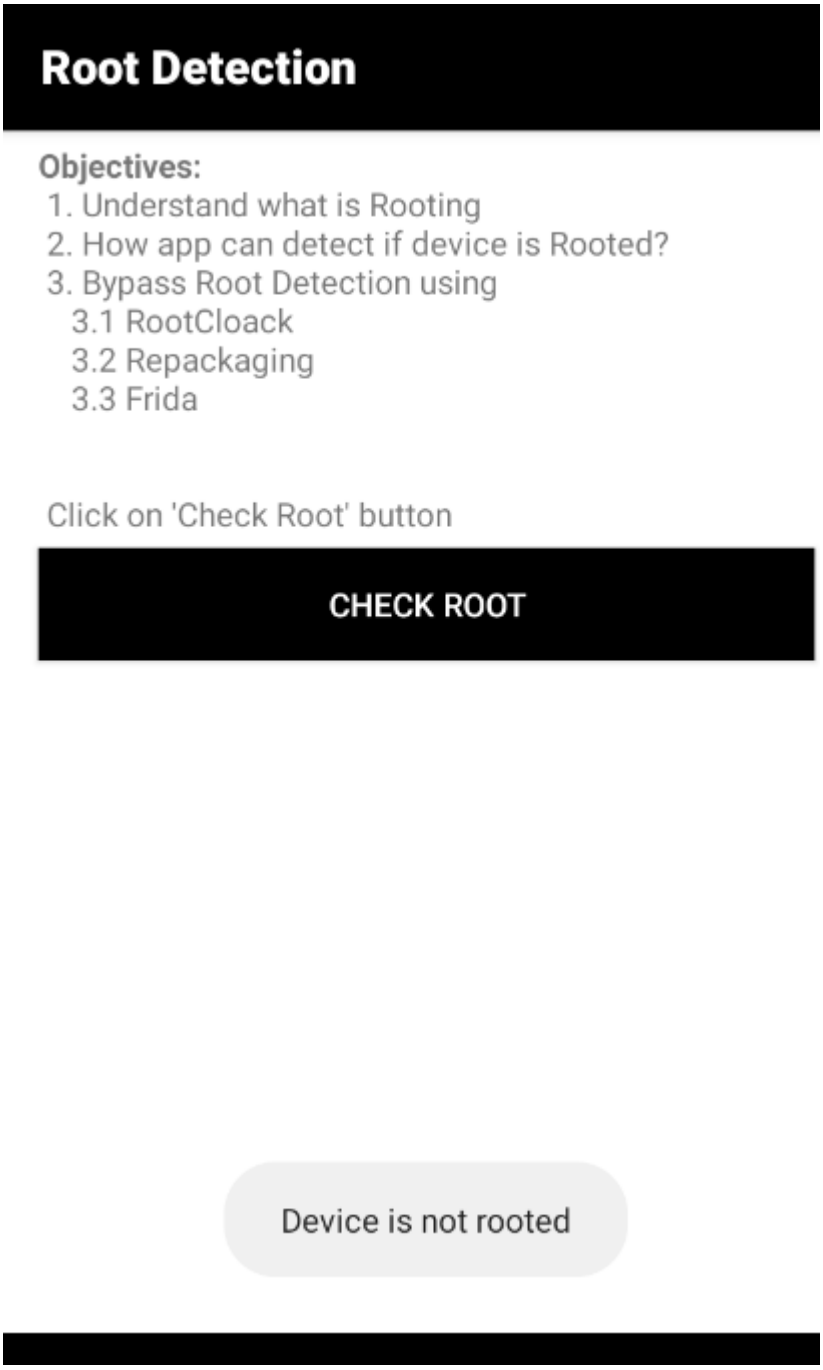
TP : Pentesting d'AndroGoat (Application Android Vulnérable)

L'objectif de ce TP est d'autoriser l'accès au compte root sur un terminal Android.
Lancer l'application et accéder à « ROOT DETECTION », puis cliquer sur « CHECK ROOT ». Un message est affiché indiquant que le système n'est pas rooté.



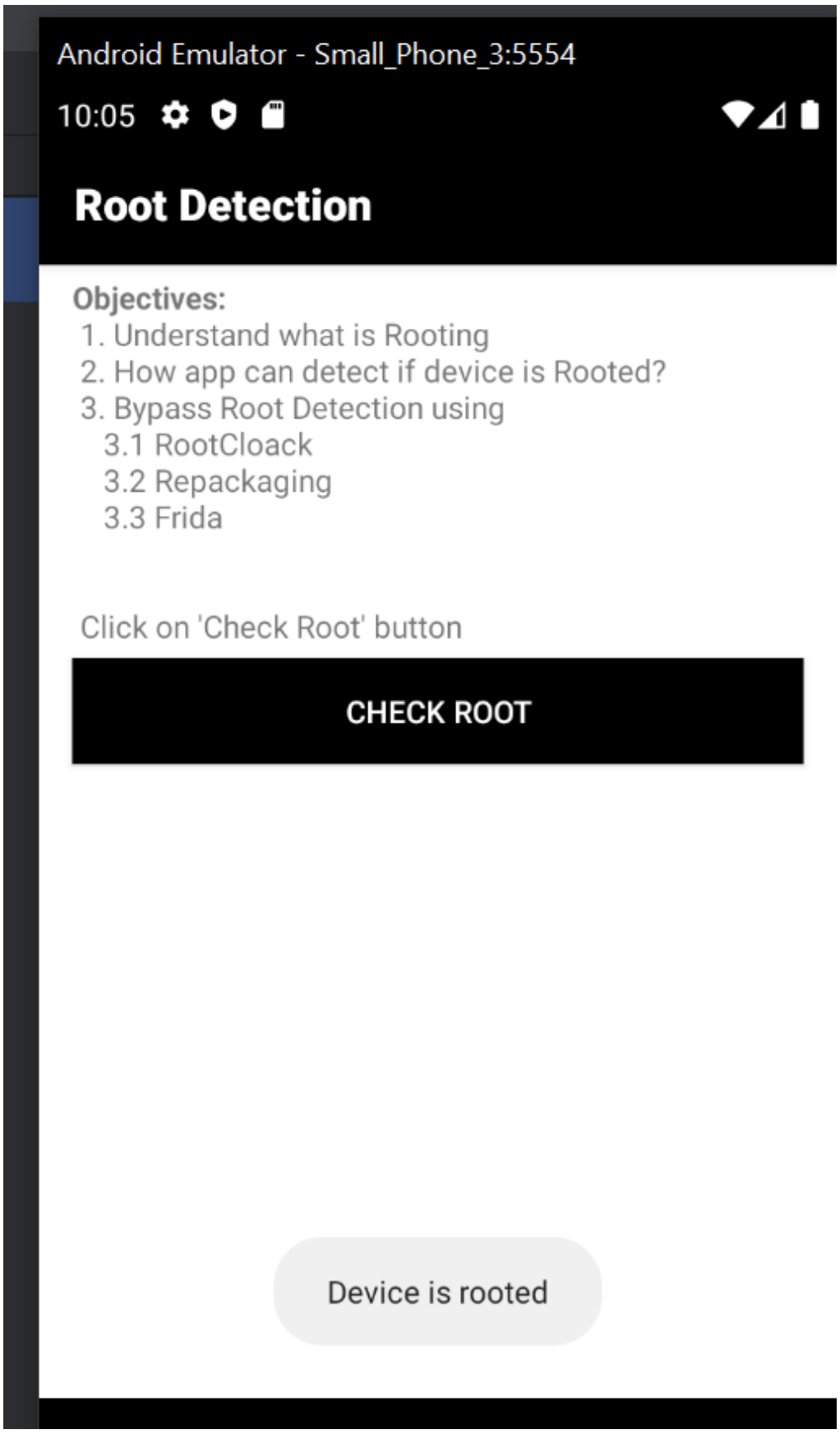
- Au niveau du terminal, il possible de se connecter en ligne de commande avec l'émulateur en utilisant la commande « adb shell », malheureusement il est impossible d'accéder au compte root par la commande « su »

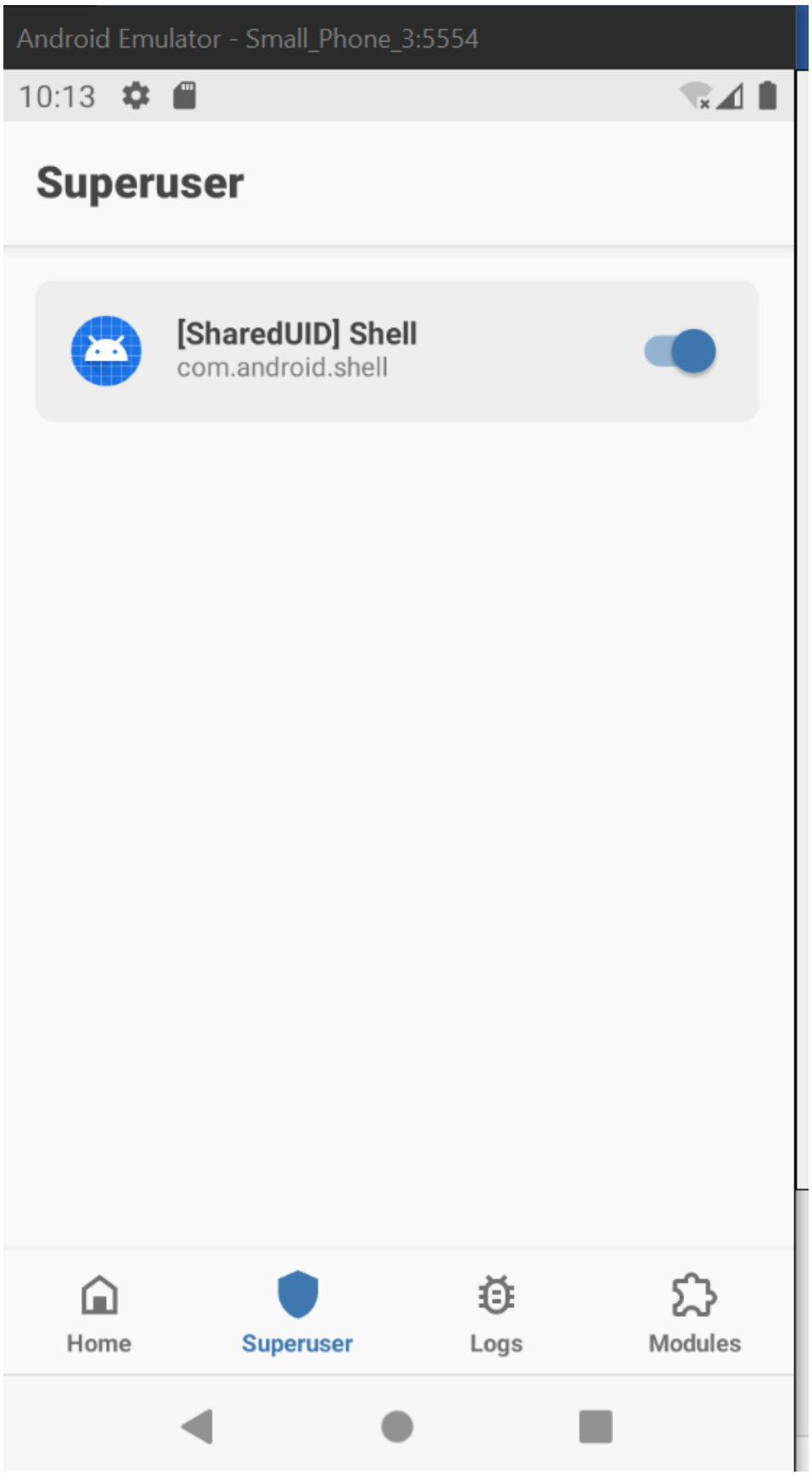
```
C:\Users\HICHAM\AppData\Local\Android\Sdk\platform-tools>adb shell
generic_x86_arm:/ $ su
/system/bin/sh: su: inaccessible or not found
127|generic_x86_arm:/ $
```

Travail à réaliser

```
C:\Users\HICHAM\Desktop>cd rootAVD

C:\Users\HICHAM\Desktop\rootAVD>rootAVD.bat ListAllAVDs
rootAVD A Script to root AVD by NewBit XDA

Usage:  rootAVD [DIR/ramdisk.img] [OPTIONS] | [EXTRA ARGUMENTS]
or:     rootAVD [ARGUMENTS]

Arguments:
        ListAllAVDs                     Lists Command Examples for ALL installed AVDs

        InstallApps                     Just install all APKs placed in the Apps folder

Main operation mode:
        DIR                             a path to an AVD system-image
                                        - must always be the 1st Argument after rootAVD

ADB Path | Ramdisk DIR| ANDROID_HOME:
        [M]ac/Darwin:                   export PATH=~/Library/Android/sdk/platform-tools:$PATH
                                        export PATH=$ANDROID_HOME/platform-tools:$PATH
                                        system-images/android-$API/google_apis_playstore/x86_64/

        [L]inux:                        export PATH=~/Android/Sdk/platform-tools:$PATH
                                        export PATH=$ANDROID_HOME/platform-tools:$PATH
                                        system-images/android-$API/google_apis_playstore/x86_64/

        [W]indows:                      set PATH=%LOCALAPPDATA%\Android\Sdk\platform-tools;%PATH%
                                        system-images\android-$API\google_apis_playstore\x86_64\

        ANDROID_HOME:                   By default, the script uses %LOCALAPPDATA%, to set its Android Home
                                        directory, search for AVD system-images and ADB binarys. This behaviour
                                        can be overwritten by setting the ANDROID_HOME variable.
                                        e.g. set ANDROID_HOME=%USERPROFILE%\Downloads\sdk

        $API:                           25,29,30,31,32,33,34,UpsideDownCake,etc.

Options:
        restore                         restore all existing .backup files, but doesn't delete them
                                        - the AVD doesn't need to be running
                                        - no other Argument after will be processed
```

```
C:\Users\HICHAM\Desktop\rootAVD>rootAVD.bat system-images\android-30\google_apis_playstore\x86\ramdisk.img
[*] Set Directorys
[-] Test IF ADB SHELL is working
[!] ADB is not in your Path, try to
set PATH=%LOCALAPPDATA%\Android\Sdk\platform-tools;%PATH%
[*] setting it, just during this session, for you
[-] Test IF ADB SHELL is working
[-] ADB connection possible
[-] In any AVD via ADB, you can execute code without root in /data/data/com.android.shell
[*] Testing the ADB working space
[!] /data/data/com.android.shell is available
[*] Cleaning up the ADB working space
[*] Creating the ADB working space
[*] looking for Magisk installer Zip
[*] Push Magisk.zip into /data/data/com.android.shell/Magisk
[-] C:\Users\HICHAM\Desktop\rootAVD\Magisk.zip: 1 file pushed, 0 skipped. 168.9 MB/s (11278270 bytes in 0.064s)
[*] create Backup File
[-] Backup File was created
[*] Push ramdisk.img into /data/data/com.android.shell/Magisk/ramdisk.img
[-] C:\Users\HICHAM\AppData\Local\Android\Sdk\system-images\android-30\google_apis_playstore\x86\ramdisk.img: 1 file pushed, 0 skipped. 293.2 MB/s (1300381 bytes in

[-] Copy rootAVD Script into Magisk DIR
rootAVD.sh: 1 file pushed, 0 skipped. 5.7 MB/s (82110 bytes in 0.014s)
[-] run the actually Boot/Ramdisk/Kernel Image Patch Script
[*] from Magisk by topjohnwu and modded by NewBit XDA
[!] We are in a ranchu emulator shell
[-] Api Level Arch Detect
[-] Device Platform is x86 only
[-] Device SDK API: 30
[-] First API Level: 30
[-] The AVD runs on Android 11
[-] Switch to the location of the script file
[*] Looking for an unzip binary
[-] unzip binary found
[*] Extracting busybox and Magisk.zip via unzip ...
[*] Finding a working Busybox Version
[*] Testing Busybox /data/data/com.android.shell/Magisk/lib/armeabi-v7a/libbusybox.so
[*] Testing Busybox /data/data/com.android.shell/Magisk/lib/x86/libbusybox.so
[!] Found a working Busybox Version
[!] BusyBox v1.34.1-Magisk (2022-03-22 04:11:29 PDT) multi-call binary.
```


Android Emulator - Small_Phone_3:5554
10:05
Root Detection
Objectives:
1. Understand what is Rooting
2. How app can detect if device is Rooted?
3. Bypass Root Detection using
3.1 RootCloack
3.2 Repackaging
3.3 Frida
Click on 'Check Root' button
CHECK ROOT
Device is rooted


Android Emulator - Small_Phone_3:5554
10:13
Superuser
[SharedUID] Shell
com.android.shell
Home
Superuser
Logs
Modules

 Au niveau du terminal, et à l'aide de la commande « adb shell », l'exécution de la commande « su » renvoi un message de permission interdite