BYPASS ROOT DETECTION OF ANDROID APPLICATIONS

Prepared by -
SAHAD BNU ABID THANGAL

BYPASS ROOT DETECTION OF ANDROID APPLICATIONS

Rooting is the process of gaining root access to your Android devices. Why does someone root his/her device?
It allows you to install custom ROMs, gives you access to root files, etc. Sounds cool, right? Ok, then you might hear that rooted devices are a security risk for user/apps. One of the reasons is any malicious app can access other applications private data or the entire file system. Now it seems to be a real threat.

Implementation of Root detection is an inevitable part of android application security. App developers have been using different root detection methods like check for test keys, directory permissions, installed packages, etc.

Now you have an android app for pentest and tried to run it on a rooted device after installation. If the application works fine, then there is no root detection method implemented. The app will close itself or get crashed if it has root detection methods.

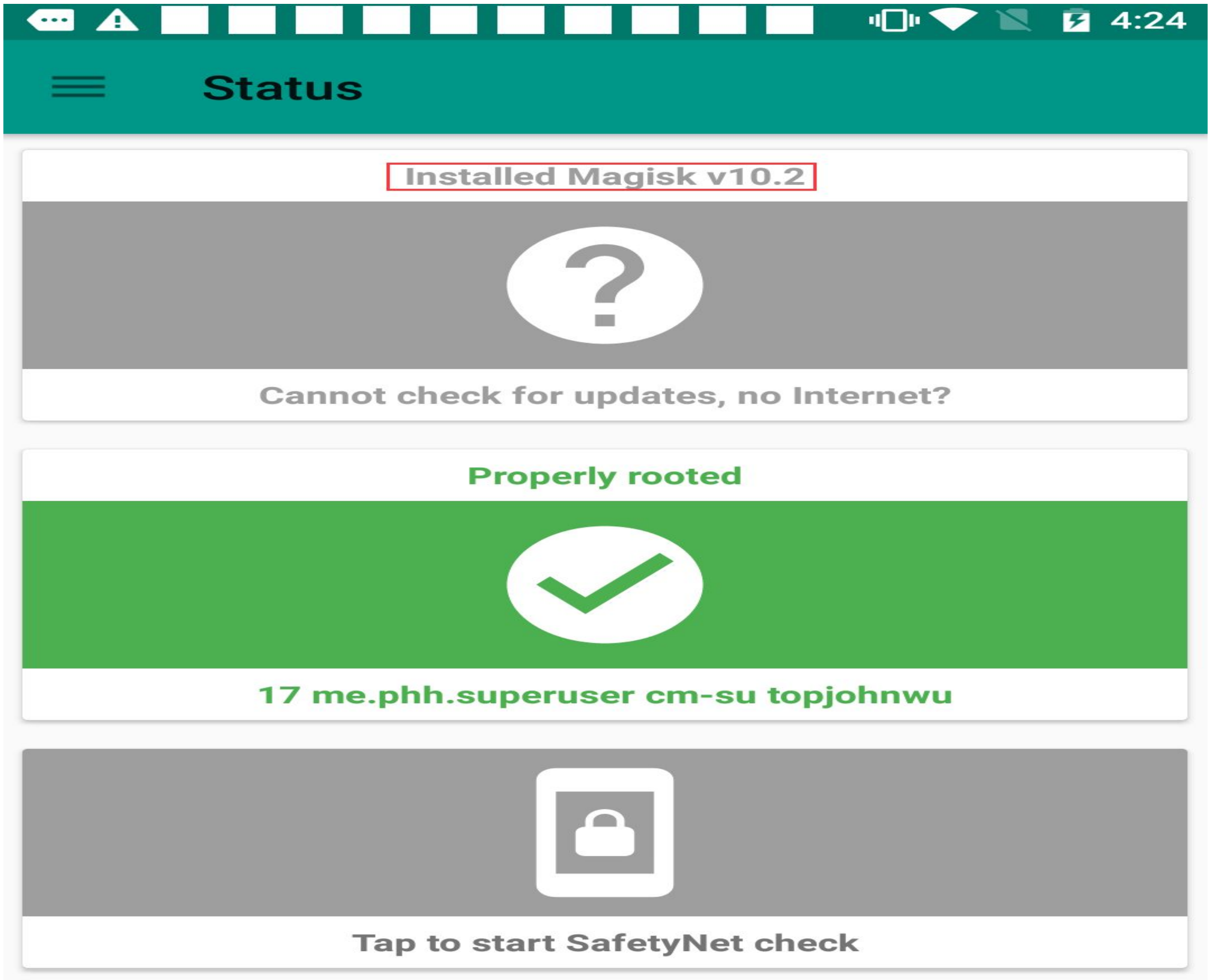
Do you think that the root detection enabled application won't run on the rooted device? Wait, don't give up so easily. There is a chance that you can bypass the root detection of the app and run it on the rooted device.

I will explain two methods that will help you to bypass the root detection.

1. Magisk Hide

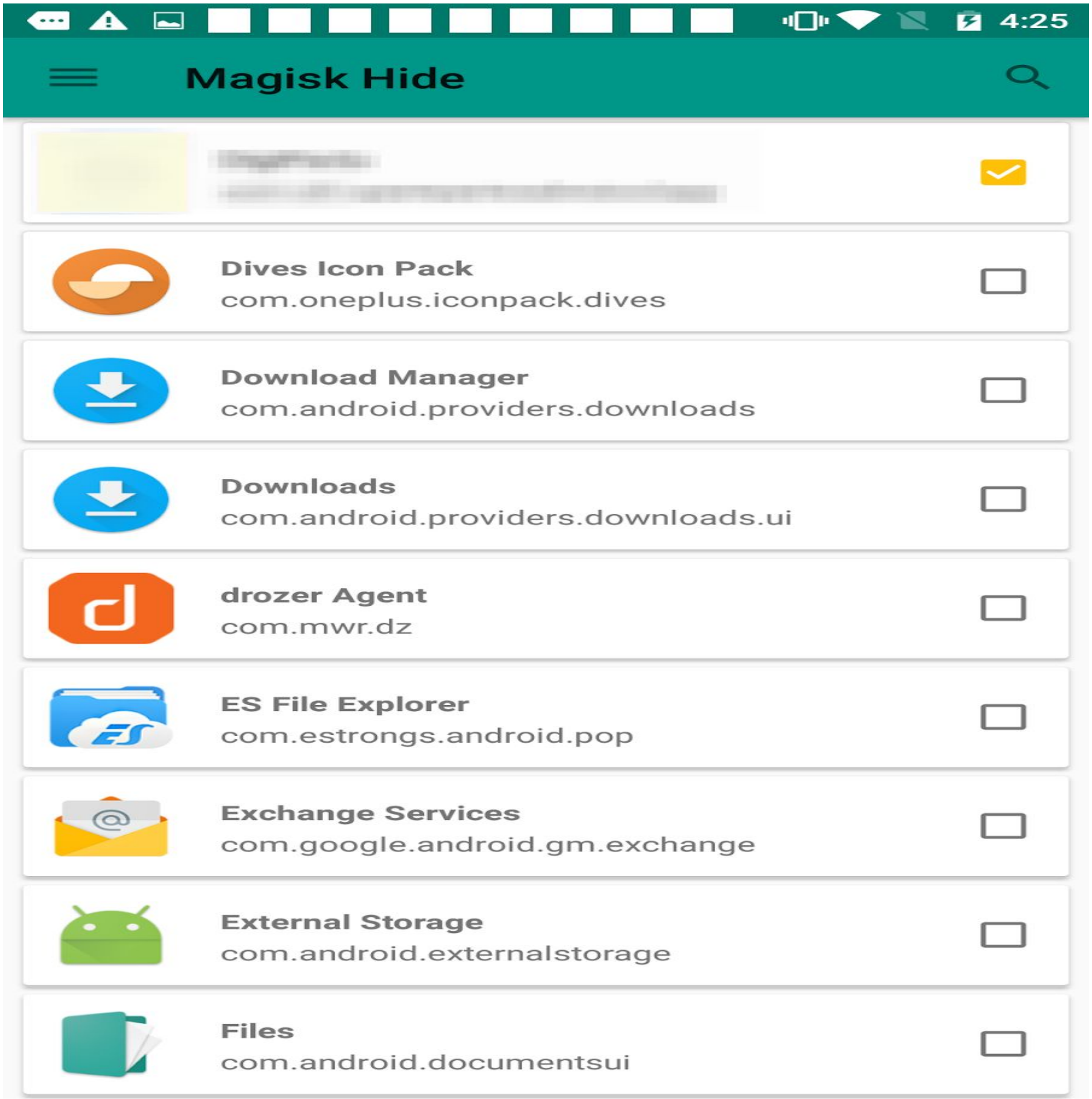
Magisk Manager's Magisk Hide module hides the root status of the device from selected apps in the list. So the root detection enabled apps cannot detect that the device has rooted.
Additionally, it has the package name randomization feature which cloaks the Magisk Manager from root detection.

step 1: Open Magisk Manager app.



4:24
Status
Installed Magisk v10.2
Cannot check for updates, no Internet?
Properly rooted
17 me.phh.superuser cm-su topjohnwu
Tap to start SafetyNet check

step 2: Select "Magisk Hide" option from the left side panel and choose the target app from the app list.



Magisk Hide

[redacted] ☑

Dives Icon Pack
com.oneplus.iconpack.dives ☐

Download Manager
com.android.providers.downloads ☐

Downloads
com.android.providers.downloads.ui ☐

drozer Agent
com.mwr.dz ☐

ES File Explorer
com.estrongs.android.pop ☐

Exchange Services
com.google.android.gm.exchange ☐

External Storage
com.android.externalstorage ☐

Files
com.android.documentsui ☐

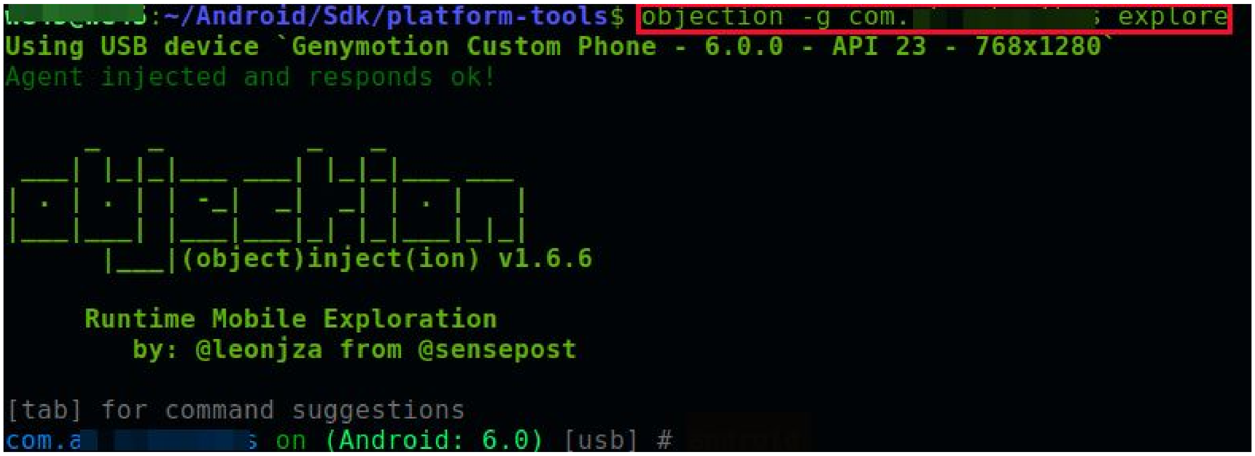
step 3: Restart your target app. Done, Now the app will work on the rooted device.

2.Objection For Android

We can also use Objection mobile runtime toolkit to bypass the root detection methods of an android app. Objection attempts to disable root detection on Android devices by hooking numerous classes such as java.lang.String (for contains()),java.lang.Runtime (for exec()) and java.io.File (for exists()).

Step 1: Start objection with your target app package

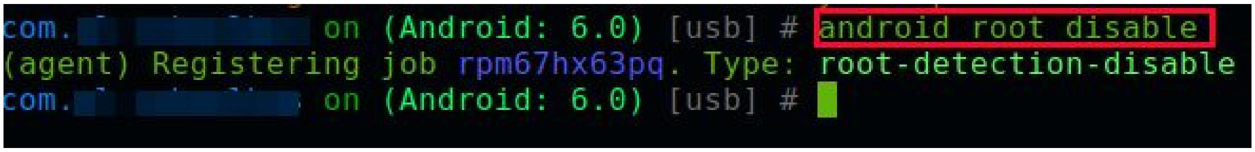
Objection -g package_name explore



```
[redacted]:~/Android/Sdk/platform-tools$ objection -g com.[redacted]s explore
Using USB device `Genymotion Custom Phone - 6.0.0 - API 23 - 768x1280`
Agent injected and responds ok!

     _   _         _   _
 ___| |_|_|___ ___| |_|_|___ ___
| . | . | | -_|  _|  _| | . |   |
|___|___| |___|___|_| |_|___|_|_|
      |___|(object)inject(ion) v1.6.6

     Runtime Mobile Exploration
        by: @leonjza from @sensepost

[tab] for command suggestions
com.a[redacted]s on (Android: 6.0) [usb] #
```

Step2: Run the built-in command that bypasses the root detection,Done.

android root disable



```
com.[redacted] on (Android: 6.0) [usb] # android root disable
(agent) Registering job rpm67hx63pq. Type: root-detection-disable
com.[redacted]s on (Android: 6.0) [usb] #
```

Of these two methods, the first one (Using Magisk Hide) is more effective. These are not only the methods to bypass the root detection. You can try other methods like reverse-engineering the application.