It's all about Bypassing Android SSL Pinning and Intercepting Proxy Unaware applications.

Kishor balan · Following

6 min read · Nov 27, 2022

Listen Share More

Hola H3ckers,

We all know there are a plenty of articles available for “How to intercept the HTTPS traffic of Android apps” , So here we are not going to cover them. If you have not found any, Refer the following:

Configuring an Android Device to Work With Burp — PortSwigger

Prerequisites:

Familiar with BurpSuite proxy, Basic Android Pentesting and tools such as adb, frida, Objection, Magisk application, Decompiling/Recompiling APK, and APK signing.

Table of Contents

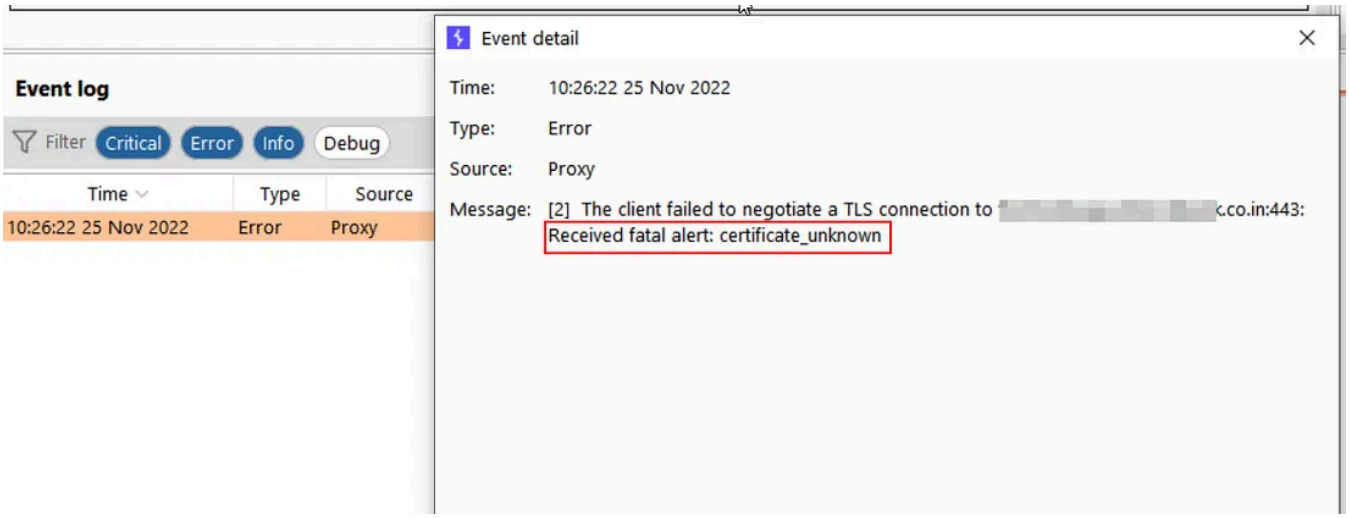
1. Does my target app has SSL pinning??

I got it, that sounds like a joke, because you guys know If the pinning is in place, then we won't be able to capture the HTTPS traffic of our target android application.

2. Wait, How we can confirm the Pinning?

After setting up the proxy in both the device and the proxy server (Burp), Fire up the target application, then do some activities that makes a communication between the target application and their server.

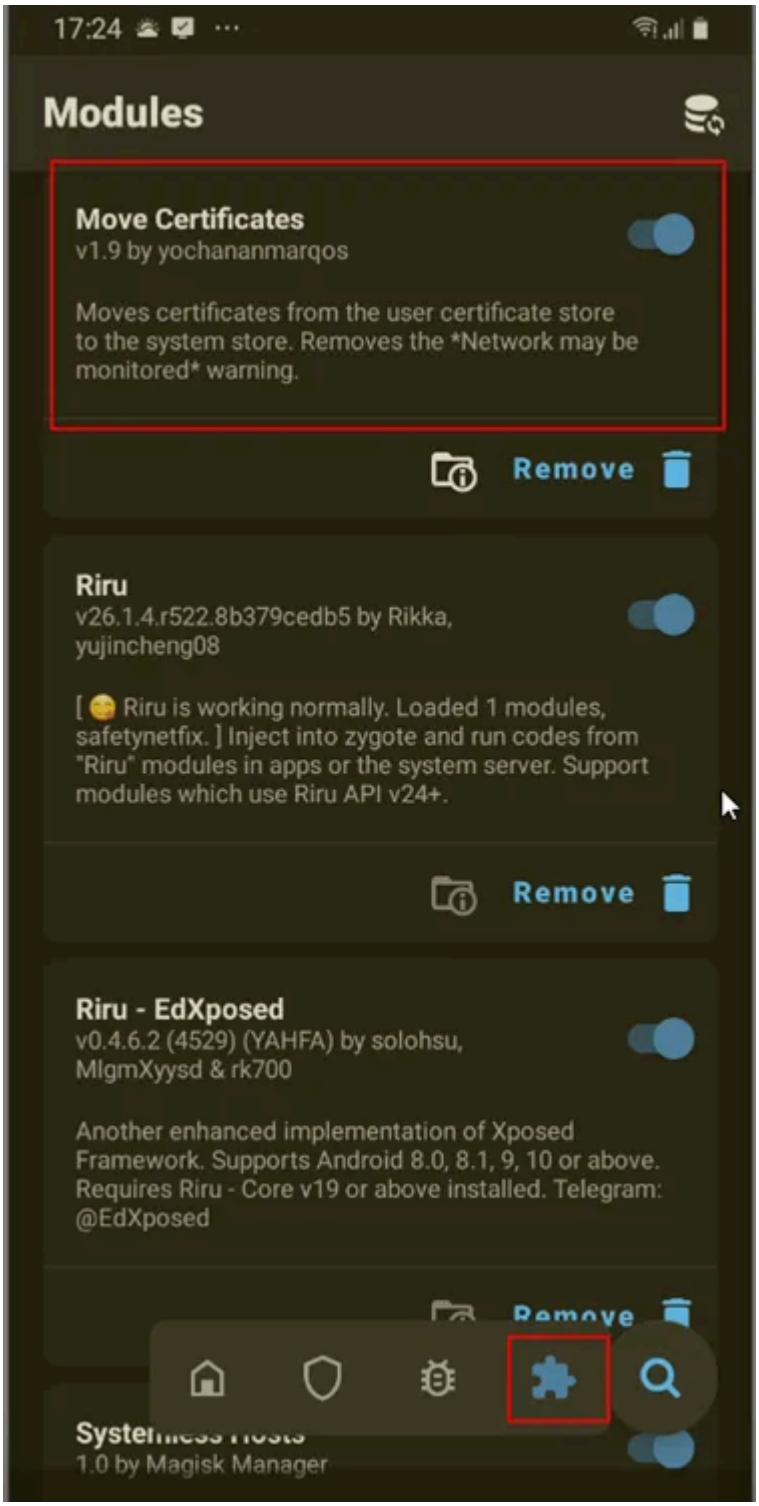
Time to monitor the Burp's dashboard, in specific, the Log section. If the Pinning is in place, then we will be able to see a Certificate error as follows:



2. Time to Bypass

2.1 Move Certificate — Magisk Module:

If your device is rooted with Magisk Application, Then Move Certificate module is one of good option.

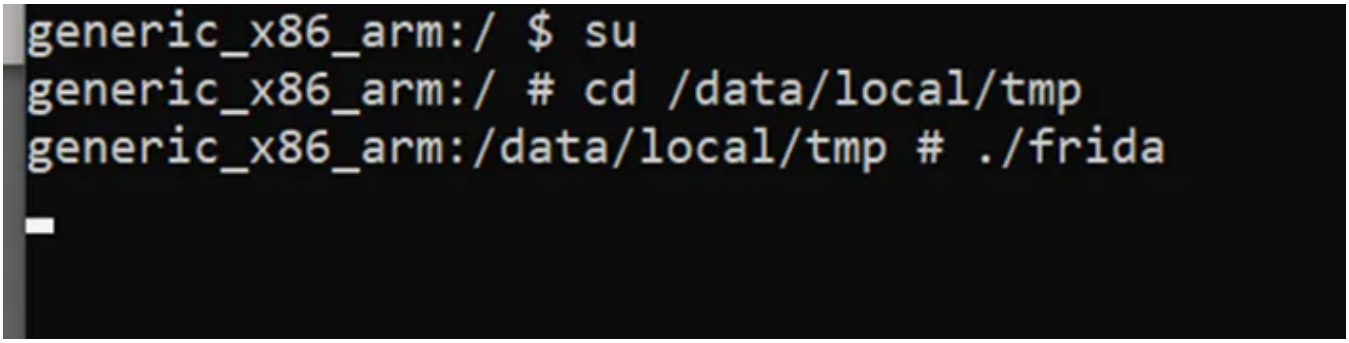


This module will move the user trusted certificates to the system store , making the system (root) trust the Certificate which the user install (Our proxy CA certificate)

2.2 Objection tool

Repo: sensepost/objection: 📱 objection — runtime mobile exploration (github.com)

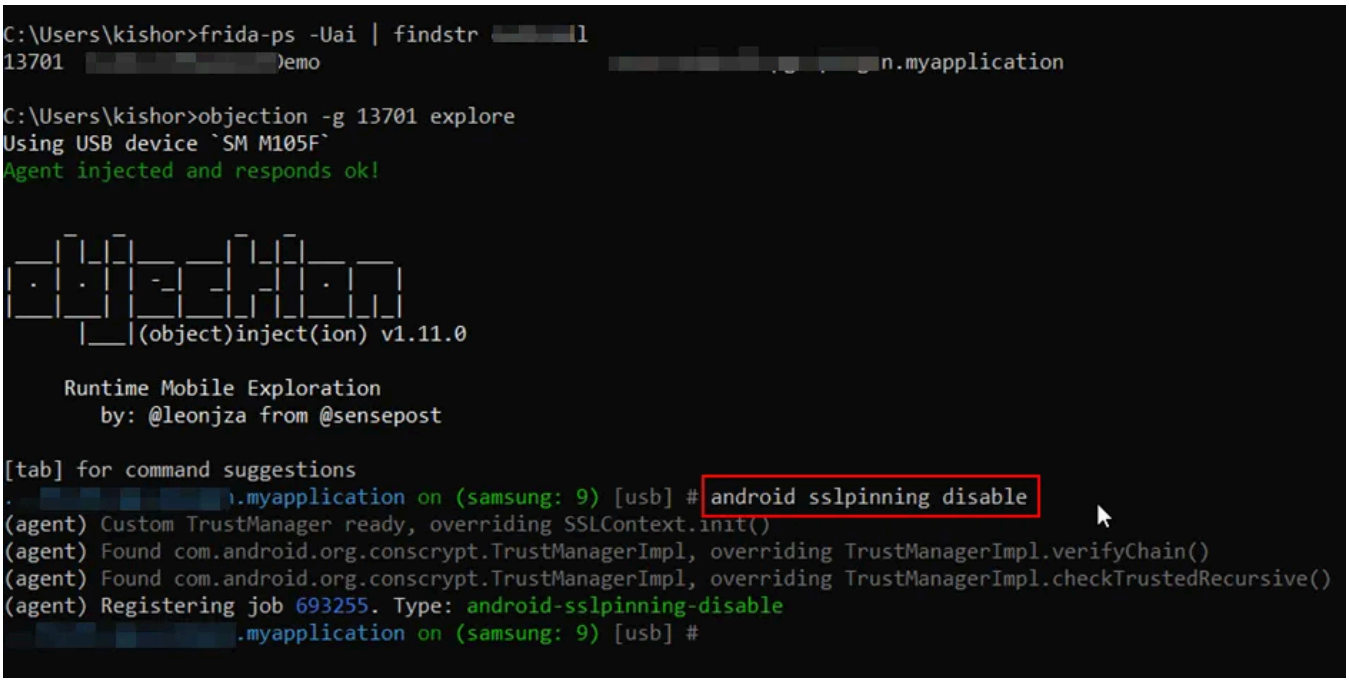
Step 1: Make sure the frida server is running on the android device



Step 2: Attach the target application with objection with the following command:

Objection -g <pkg name/ PID> explore

Then execute the "*android sslpinning disable*" command



Thats it , the script will find the SSL pinning classes and hook them during the runtime in order to byass the Pinning.

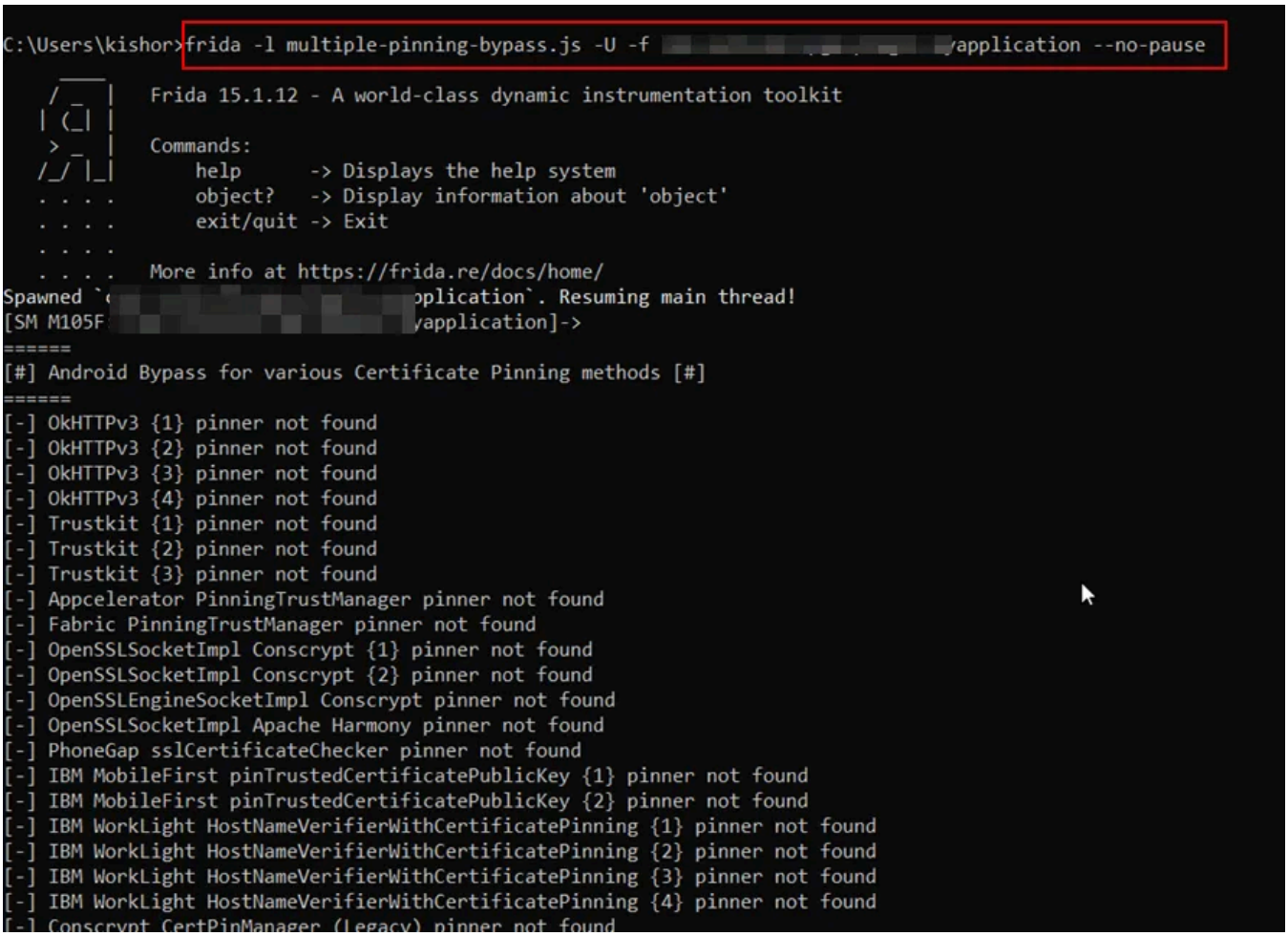
2.3 Frida Framework

Repo: Frida (github.com)

Here comes the most popular and widely used method.

Step 1: Make sure the frida server is running on the android device

Step 2: Attach your target application with frida and run your favorite SSL bypassing script.



Following are my favorite scripts:

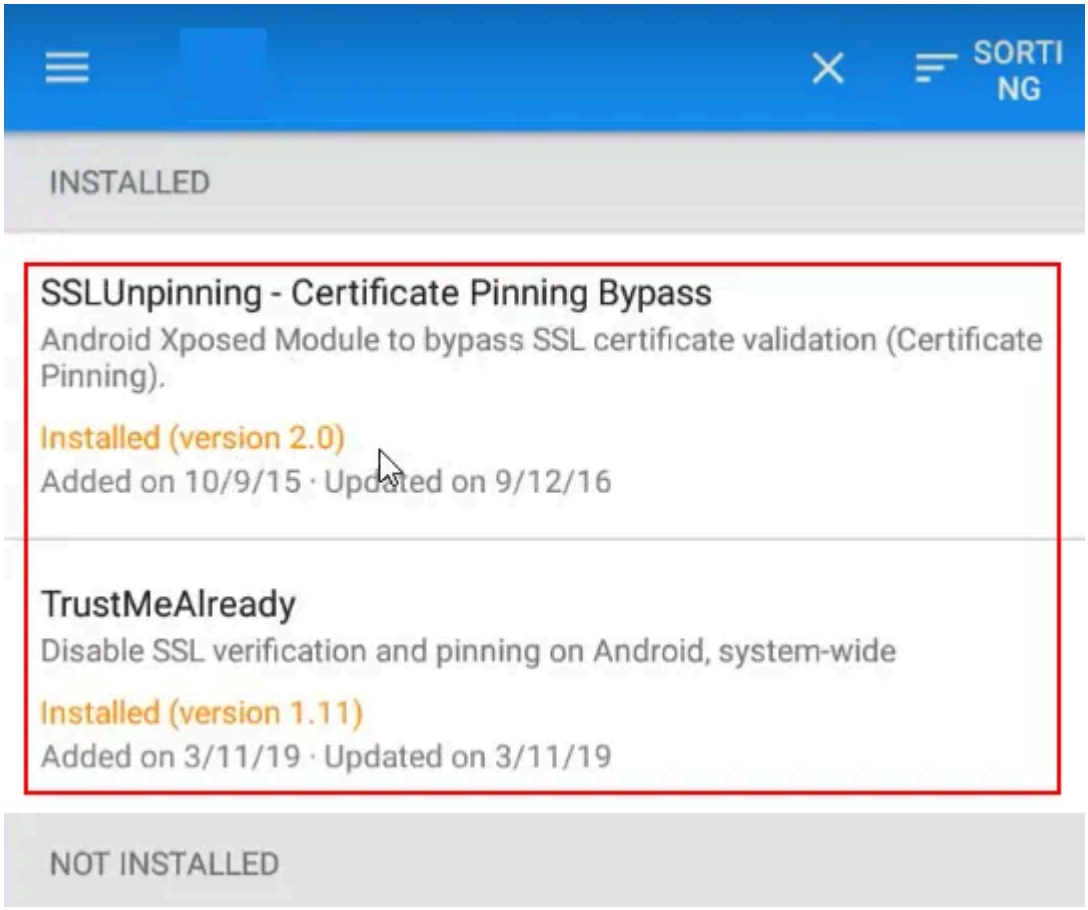
https://codeshare.frida.re/@akabe1/frida-multiple-unpinning/
https://codeshare.frida.re/@pcipolloni/universal-android-ssl-pinning-bypass-with-frida/

2.4 Using Xposed Framework

If your device is rooted with **Xposed framework**, then you can try the following modules to bypass the pinning

1. ac-pm/SSLUnpinning_Xposed: Android Xposed Module to bypass SSL certificate validation (Certificate Pinning). (github.com)

2. ViRb3/TrustMeAlready: 🔓 Disable SSL verification and pinning on Android, system-wide (github.com)



2.5 Using apk-mitm

apk-mitm is a CLI application that automatically prepares Android APK files for HTTPS inspection by modifying the apk files and repacking.

Repo:

shroudedcode/apk-mitm: 🤖 A CLI application that automatically prepares Android APK files for HTTPS inspection (github.com)

apk-mitm can be pulled out using **npm.**

```
C:\Users\kishor>npm install -g apk-mitm

added 142 packages, and audited 143 packages in 34s

18 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Step 1: Run the apk-mitm as shown in below.

```
C:                                                    \tools>apk-mitm app.apk

  apk-mitm v1.2.1
  apktool v2.6.1
  uber-apk-signer v1.2.1

Using temporary directory:
(                                                    )-b0a74f454c40d4581567ba6f78b9ff9d

√ Checking prerequisities
√ Decoding APK file
√ Applying patches
√ Encoding patched APK file
√ Signing patched APK file

Done! Patched file: ./app-patched.apk
```

Thats it, apk-mitm has done its part. Now we can install the patched apk and intercept the application traffic.

2.6 Modifying the network_security_config.xml file

The Network Security Configuration lets apps customize their network security settings through *a declarative configuration file*. The entire configuration is contained within this XML file, and no code changes are required.

Source: Network security configuration | Android Developers

The Network Security Configuration works in **Android 7.0 or higher.**

Step 1: Decompile the android application with apktool or alternatives. And locate the ***network_security_config.xml*** file under ***/res/xml.***

Step 2: The file may look like this if the app has pinned its own CA certificates.

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">example.com</domain>
        <pin-set expiration="2018-01-01">
            <pin digest="SHA-256">7HIpactkIAq2Y49orFOOQKurWxmmSFZhBCoQYcRhJ3Y=</pin>
            <!-- backup pin -->
            <pin digest="SHA-256">fwza0LRMXouZHRC8Ei+4PyuldPDcf3UKgO/04cDM1oE=</pin>
        </pin-set>
    </domain-config>
</network-security-config>
```

Image source: developer.android.com

Step 3: Remove that <pin-set>... </pin-set> tag section and add the following:

```
PS C:[redacted]s> type network_security_config.xml
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">example.com</domain>
        <trust-anchors>
            <certificates src="user" />
            <certificates src="system" />
        </trust-anchors>
    </domain-config>
</network-security-config>
```

Step 4: Now save the file and Re-pack the application using apktool and uber-apk-signer (For signing the modified apk).

Thats it, install our new apk and your are good to go.

3. What if the application is not getting intercepted and also not showing any errors !!

Here the first thing pop-up in my mind is "**Flutter**". The flutter based applications are basically "**Proxy unaware**".

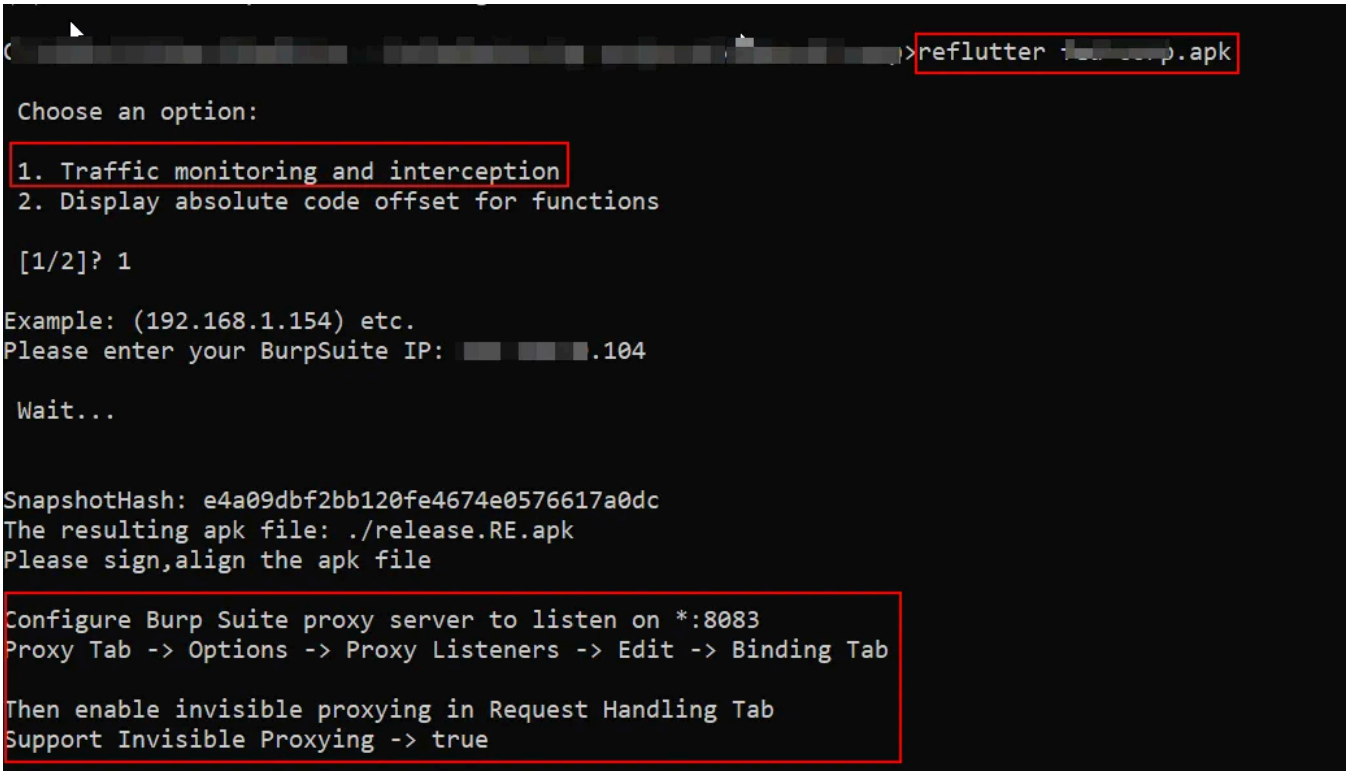
So here comes our hero "**Reflutter**" :

"This framework helps with Flutter apps reverse engineering using the patched version of the Flutter library which is already compiled and ready for app repacking. "

Repo: https://github.com/Impact-I/reFlutter

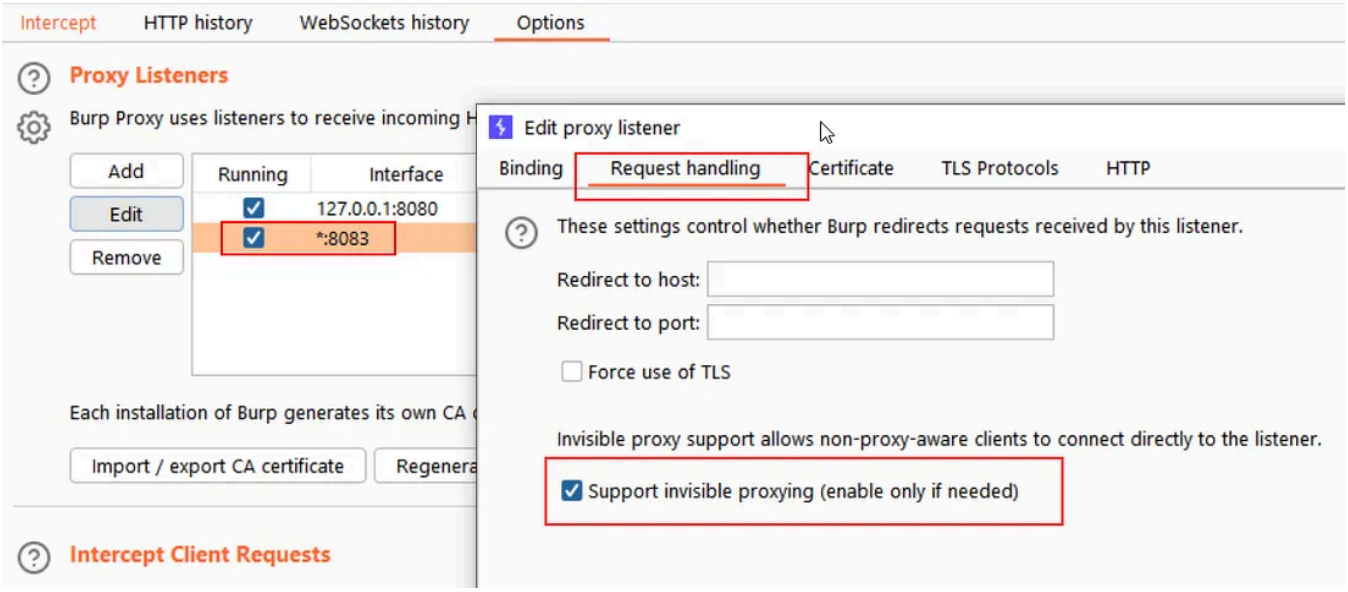
Step 1: install the reflutter using pip

Step 2: Follow the commands illustrated in the below screenshot.



Step 3: Sign the application using uber-apk-signer or any alternatives and install it.

Step 4: Now in Burp proxy, Start listening the port **8083** and also enable "**Support Invisible Proxying**".



That's it peeps, you are all good to go ... !

4. My application is using HTTP only but Still I am not able to Intercept!!

Hmm..That's a kinda weird , But it happens sometimes.

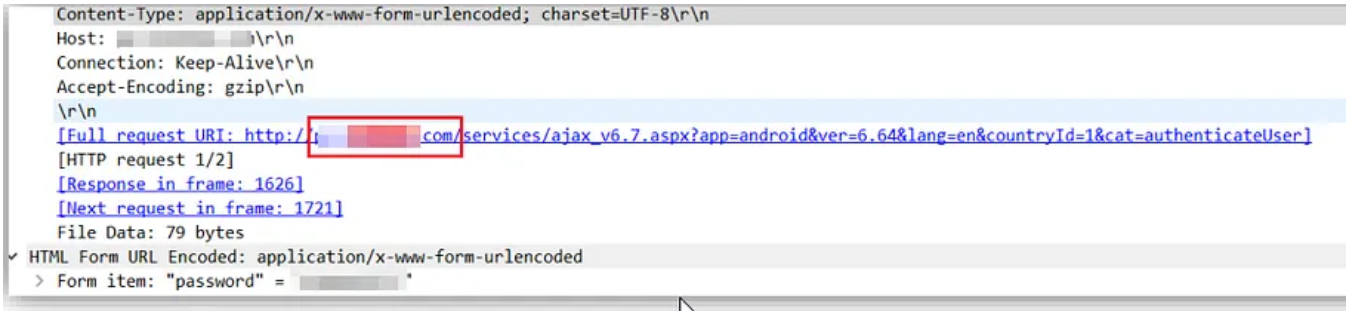
Applications with this behaviour, are basically called "**Proxy Unaware**" applications. Such applications route the traffic directly to the internet without cooperating with system wide Proxy settings.

Time to bypass:

For this method, I would like to thank brother Faris ❤.

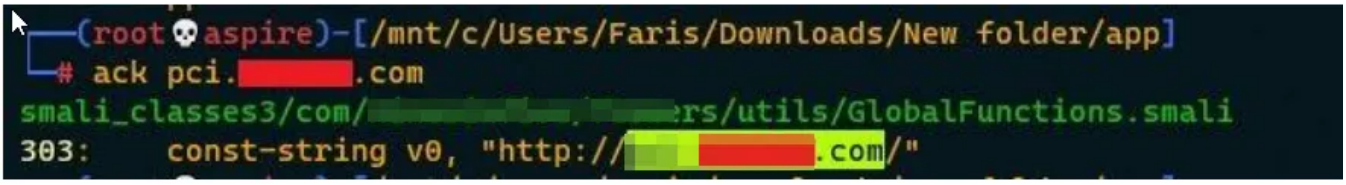
(60) Faris Mohammed | LinkedIn

Step 1: Find out the domain address to which the App is communicating using Wireshark. Shown Below.

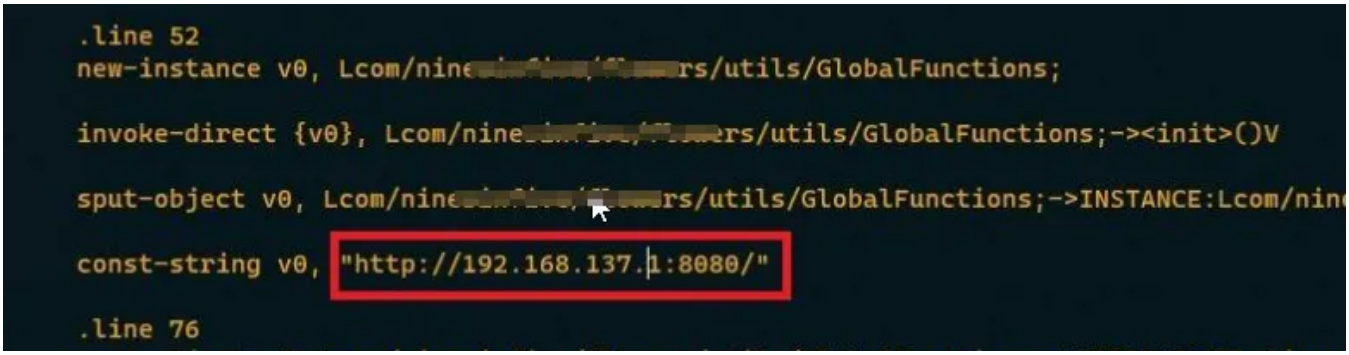


Step 2: Decompile the application using apktool

Step 3: Enter the decompiled folder and use the ack/grep tool to find out the file in which the domain name is mentioned.



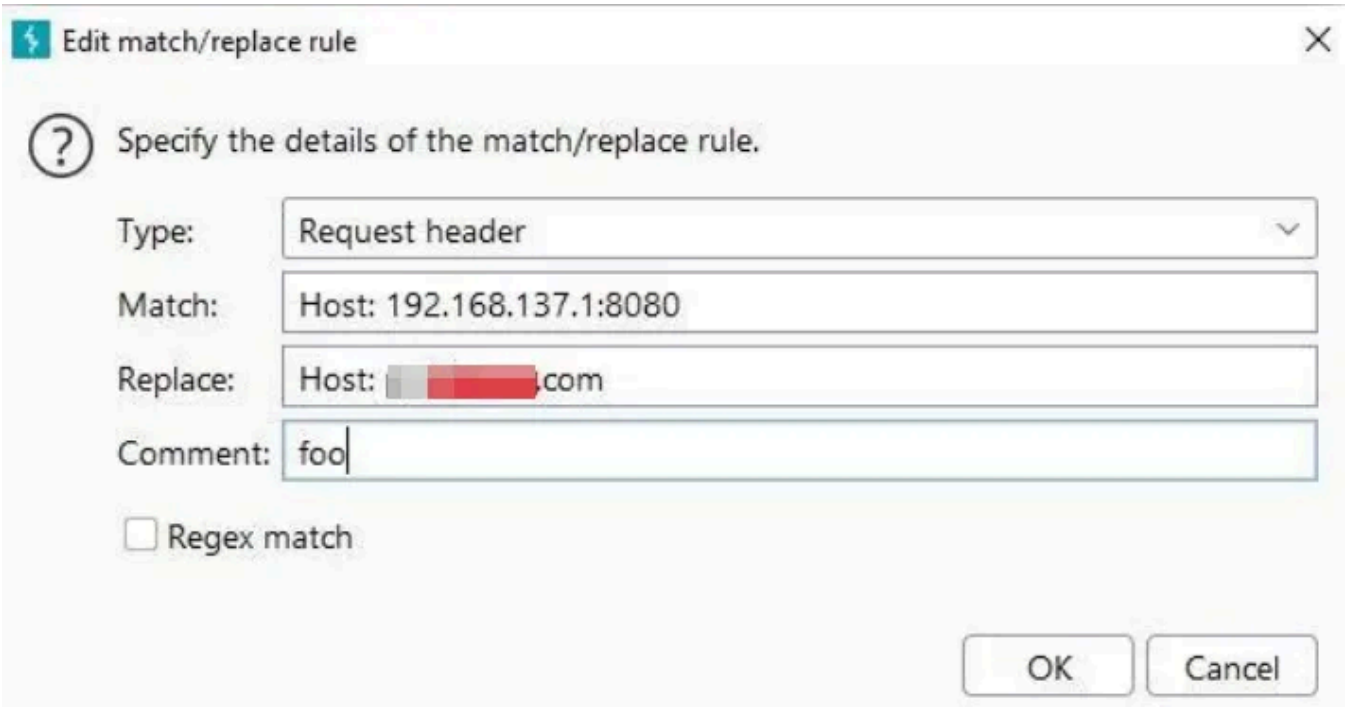
Step 4: Replace the domain name with the IP address and Port of BurpSuite.



Step 5: Re-pack the application, sign it and install it on the android device.

Step7: In the BurpSuite proxy, From the Request handling tab, give redirect host and port as the original domain address which was used by app in the first place.

Step 8: Setup match and replace in the proxy options to replace the Host header value from the burp listener IP address to original domain address of the application



Thats it. Now the application's HTTP traffic will be captured in our Burp proxy.

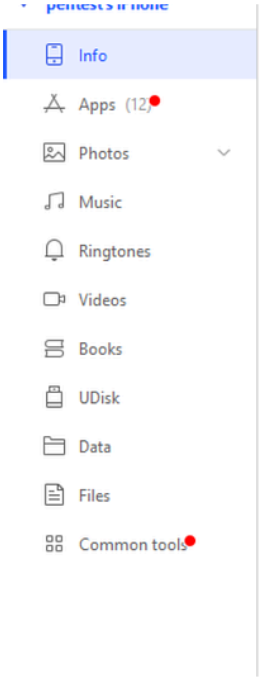
Note: *— Here, Since we replace the hardcoded application domain, we don't need to setup device proxy since the application directly communicates with the hardcoded domain (We have replaced it with our proxy IP).*

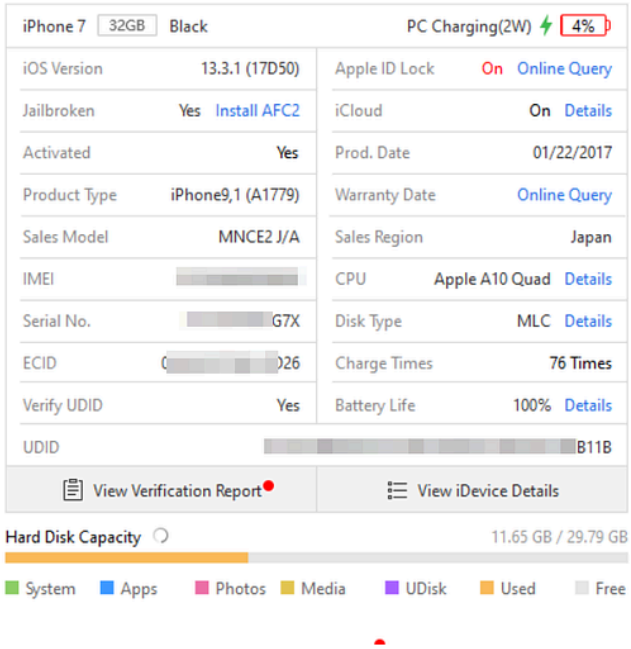
So I think I've done for now. In fact, there are some more other methods that we can use to bypass the android SSL Pinning. I will try to make a Part 2 with that based on your responses.

Thanks peeps, Happy hacking :)

Cybersecurity Android Security Mobile Pentesting Hacking

More from Kishor balan





Kishor balan

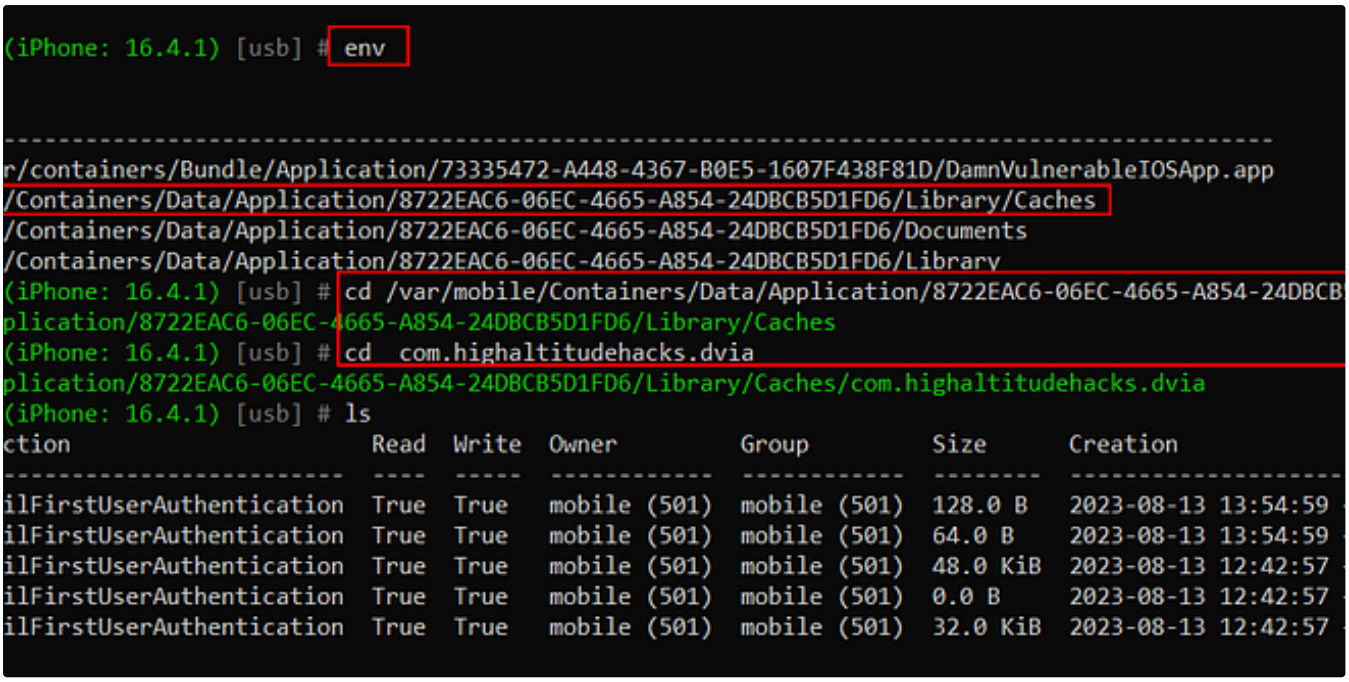
Start your first iOS Application Pentest with me.. (Part- 1)

Hola Heckers,

6 min read · Jan 14, 2023

240

3



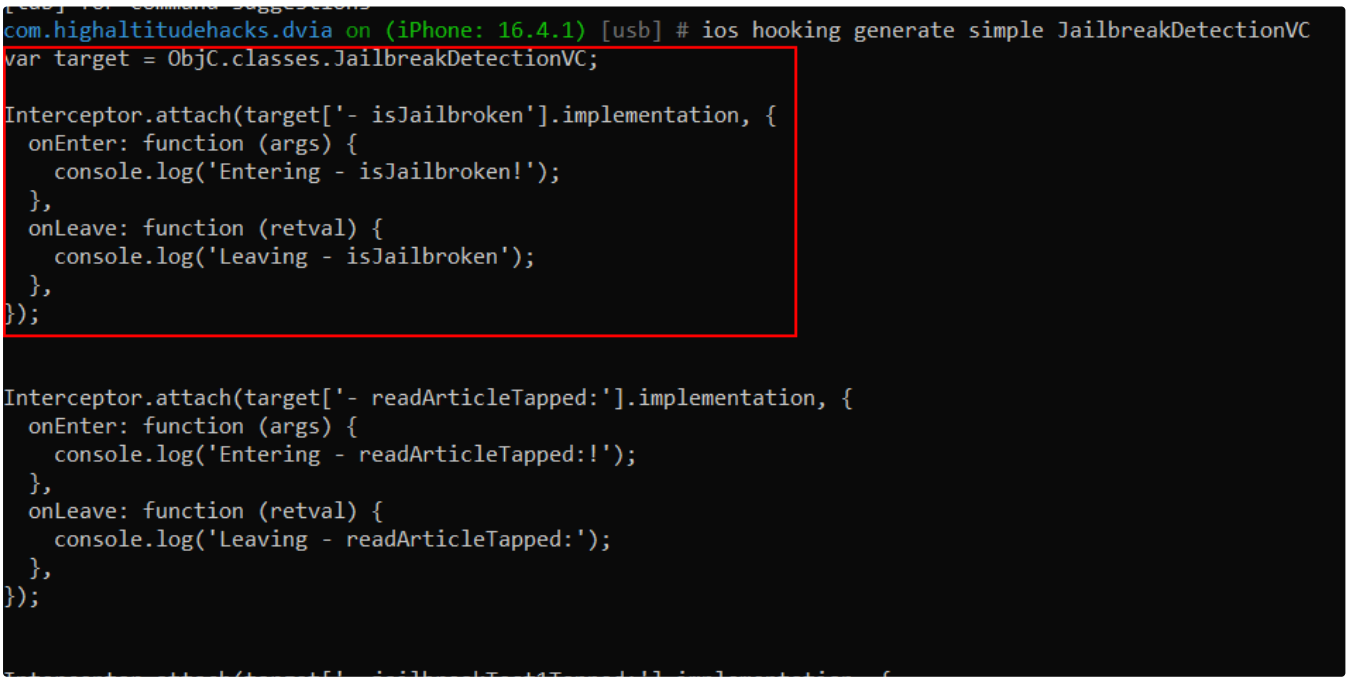
Kishor balan

iOS Pentesting Series Part 2- Into The Battlefield..

Hola Peeps,

6 min read · Aug 14, 2023

44 2



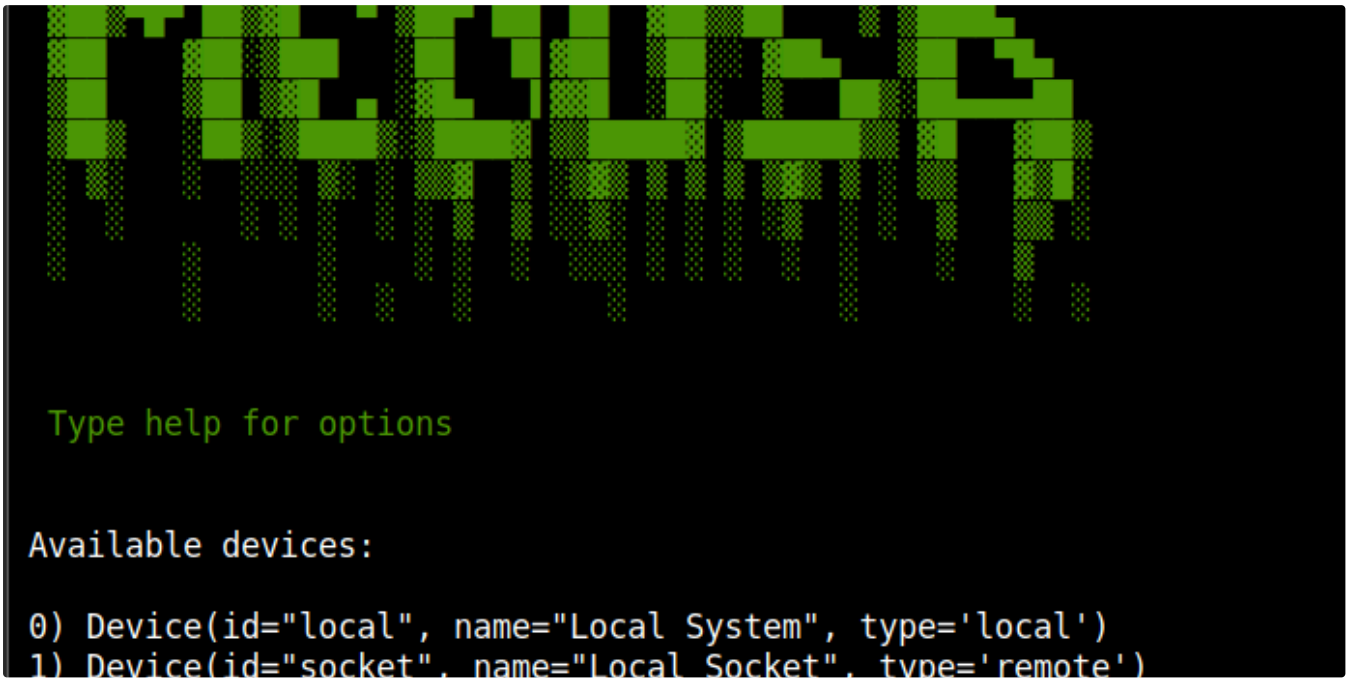
Kishor balan

iOS Pentesting Series Part 3- The Ceasefire

Hola mates,

7 min read · Aug 19, 2023

22



Kishor balan

My fav 7 methods for Bypassing Android Root detection

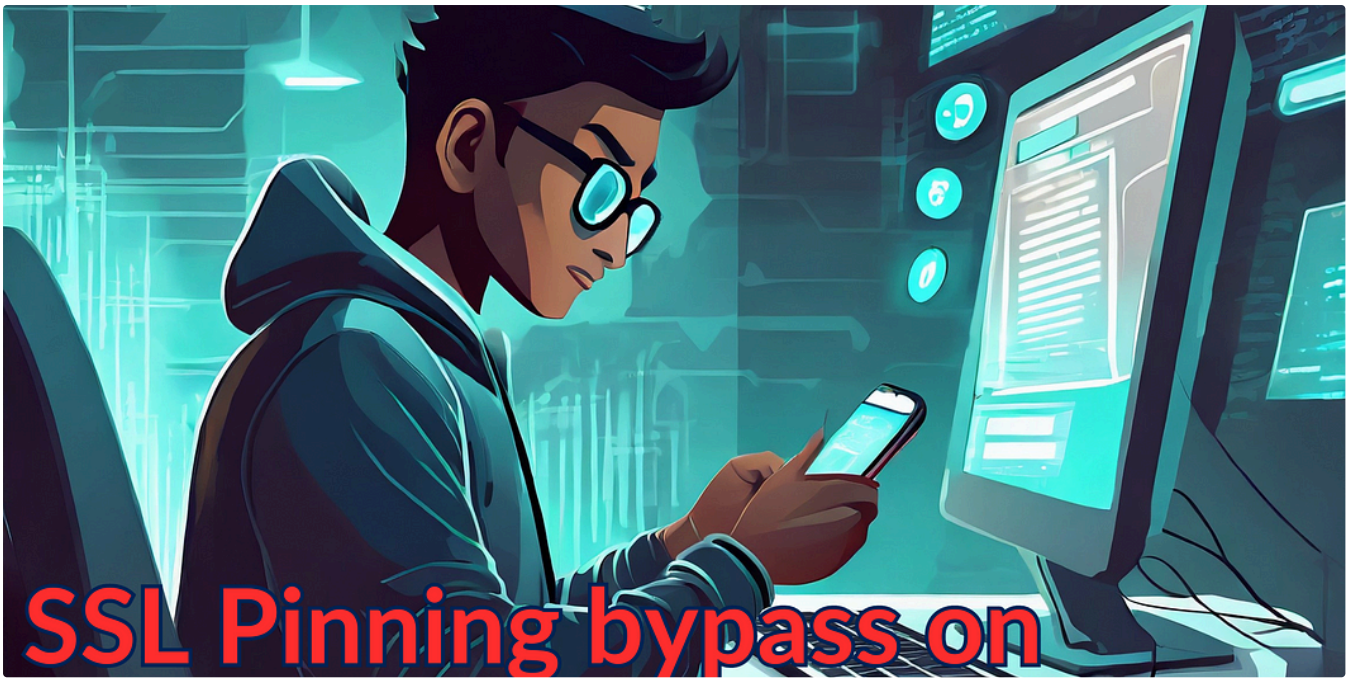
Hola H3ck3rs,

6 min read · Oct 16, 2022

185 2

See all from Kishor balan

Recommended from Medium



Sahil Choudhary

SSL Pinning Bypass on Flutter-Based Android Apps

In the ever-evolving landscape of mobile application security, SSL pinning has become a common defense mechanism to protect sensitive data...

2 min read · Jan 3, 2024

13

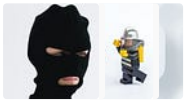
Luxeedd

Mobile Pentest Dynamic Analysis - Hooking with Frida

Introduction

3 min read · Nov 22, 2023

Lists



Tech & Tools

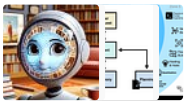
16 stories · 212 saves

Medium's Huge List of Publications Accepting Submissions

285 stories · 2528 saves

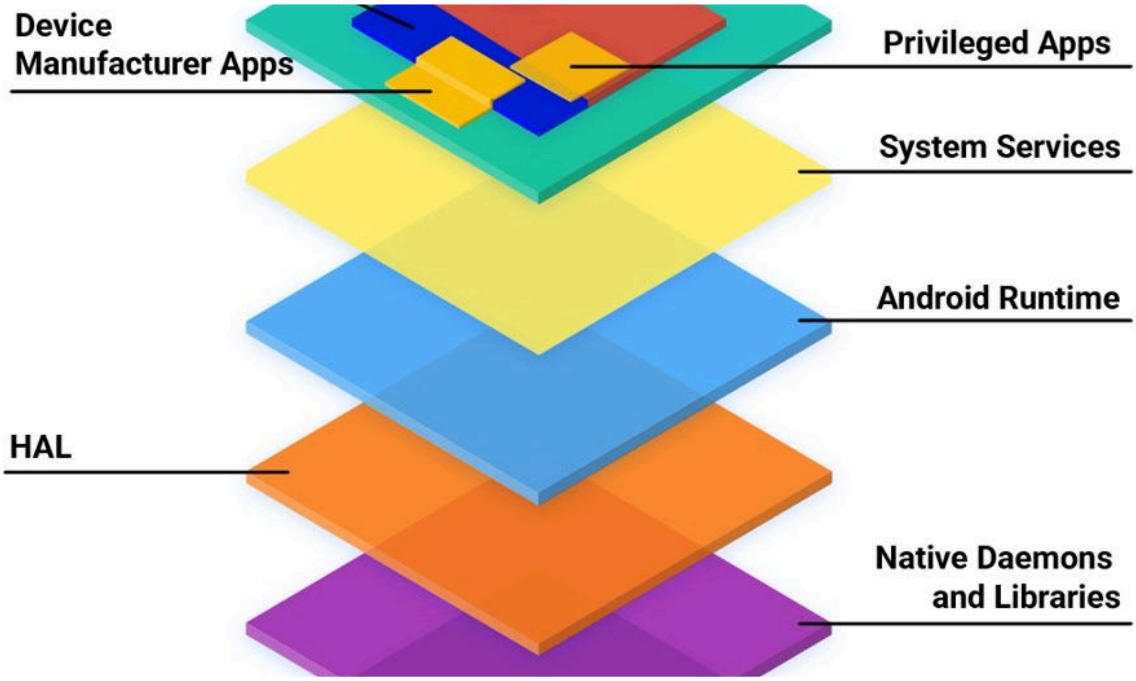
Staff Picks

630 stories · 922 saves



Natural Language Processing

1407 stories · 905 saves

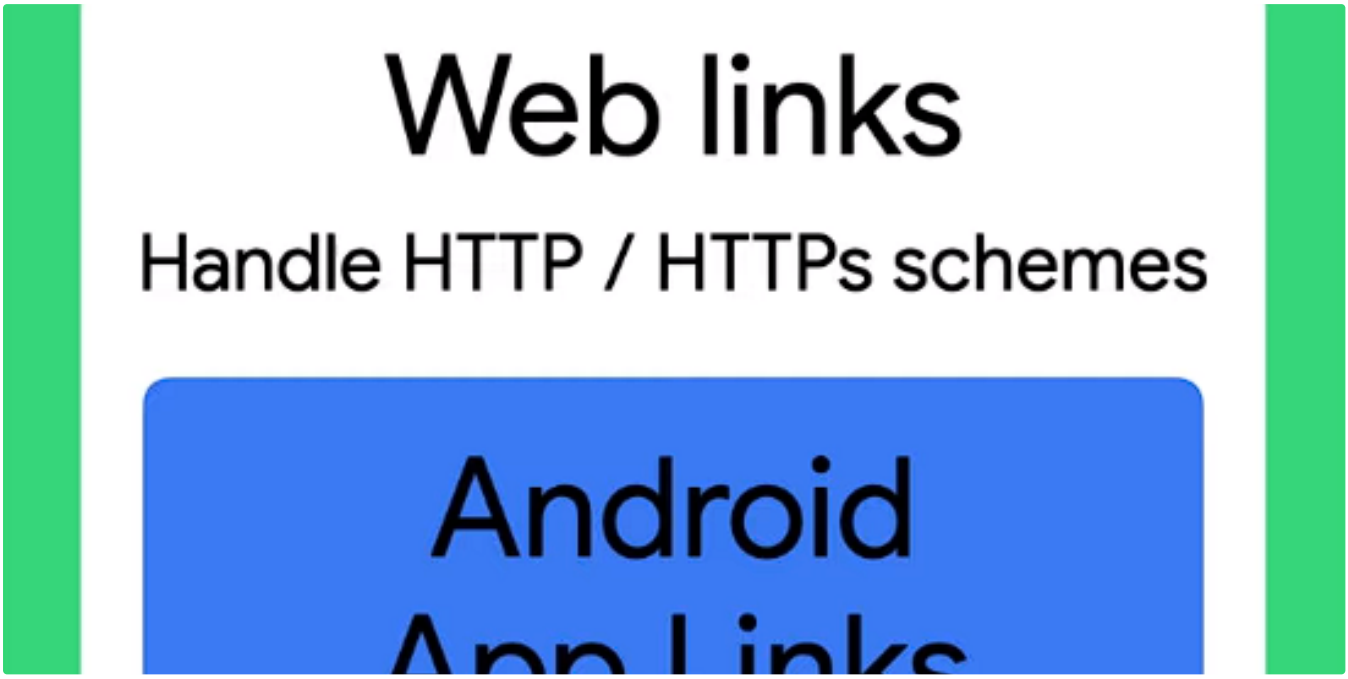


Rudranil Maity

Network Traffic Analysis of Android apps using MITMproxy and Frida

Abstract: With the overwhelmingly increasing number of applications in android marketplace, i.e., google play store; over time with the...

15 min read · Nov 22, 2023



Z4ki

Android Deep Links exploitation

Recently, I have been studying Android penetration testing, and I came across a new topic called ‘Deep Links’. Since it is a new topic to...

5 min read · Oct 30, 2023

197

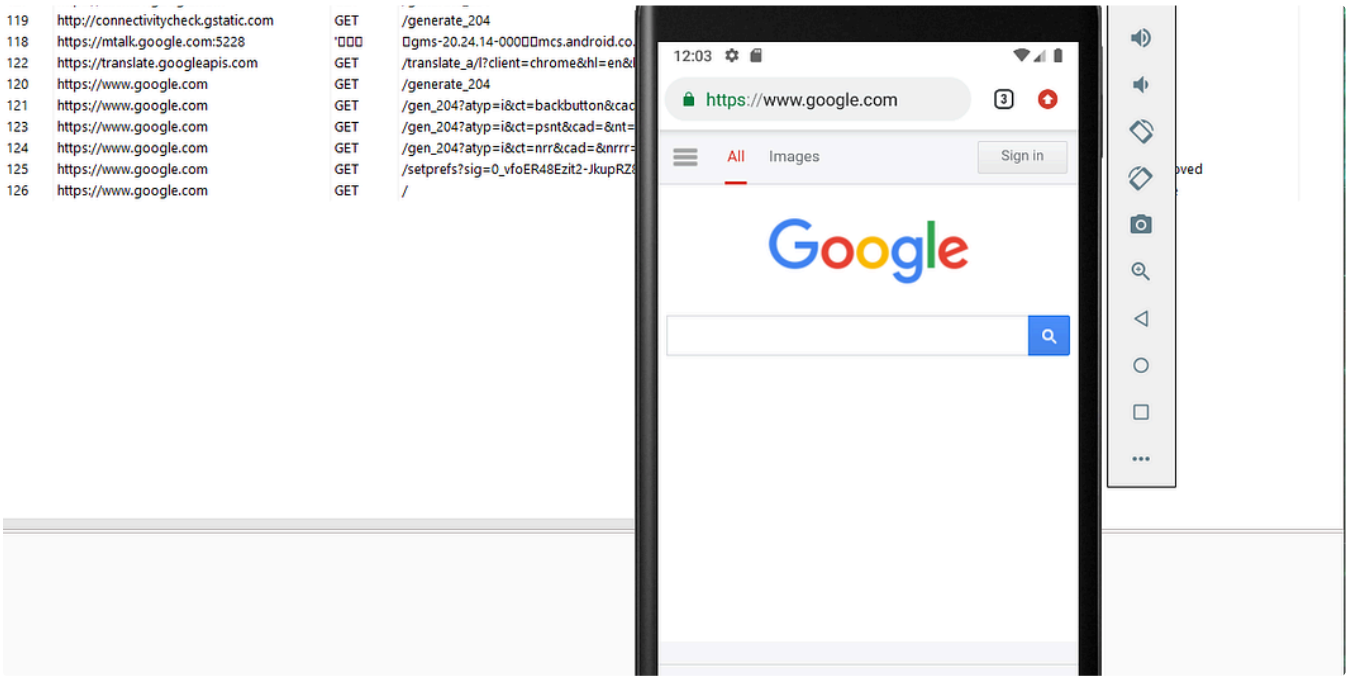
Hacker's Dump

Diving Deep: A Comprehensive Guide to Android Penetration Testing — Part 3

Cracking the Code: A Beginner's Guide to Decoding Android Apps

8 min read · Nov 19, 2023

2



Shayan Ahmed Khan

RevEnge! Reverse Engineering android apps to bypass SSL pinning for mobile app pen-testing

This is the second part of my 2 part blog series on mobile app pen-testing and reverse engineering. In the first part, I have explained how...

5 min read · Oct 31, 2023

 11