Hide `n seek with Android App Protections & Beat `em

By Aan @petruknisme

HELLO! I'm…

Aan Wahyu a.k.a petruknisme

- Lead Security Consultant @ Horangi
- Infosec Enthusiast & Part-time coder
- Passionate with OSINT, RE, and Red Team

Background

Along with the rapid development of mobile application technology meeting various needs and providing convenience to their users. In this case, significant developments are also needed in the **security aspect** that guarantees **privacy** and **security**, especially user data.

Therefore, a **penetration testing** or **application audit** process is needed to ensure that an application is suitable for use by the public. If this is not done properly, there is a high possibility of **data leakage** resulting in losses on the user side and on the company.

Background

To prevent **modification**, **manipulation** or **hacking**, it is not uncommon for developers to apply **protection** to applications. This aims to **minimize losses** that could occur.

As someone who works as a penetration tester, For applications can be tested properly, it is often necessary to bypass the implemented protection so that the required tools or processes can run according to the predetermined penetration testing workflow.

Table of contents

01

Root Detection

Root Detection

To prevent **modifications** or **manipulations** that impact the application, root detection is implemented. Basically, the application will **check files** that are indicated to be part of the rooted device and when it finds this indication, the application will **prevent access** or perform an **exit/crash** so that the application cannot run on a rooted device.

Root Detection Methods

Checking the BUILD tag for test-keys

Checking SU binary and installed root package

Checking dangerous props

Checking common root cloaking apps

Checking permission for system directory

Many more

Sample Root Detection

```
    /* JADX INFO: Access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.SupportActivity, android.app.Acti
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_root_detection);
        Button rootBt = (Button) findViewById(R.id.rootCheck);
        rootBt.setOnClickListener(new View.OnClickListener() { // from class: owasp.sat.agoat.RootDetectionActivity$onCreate$1
            @Override // android.view.View.OnClickListener
            public final void onClick(View it) {
                if (RootDetectionActivity.this.isRooted()) {
                    Toast.makeText(RootDetectionActivity.this.getApplicationContext(), "Device is rooted", 1).show();
                } else {
                    Toast.makeText(RootDetectionActivity.this.getApplicationContext(), "Device is not rooted", 1).show();
                }
            }
        });
    }

    public final boolean isRooted() {
        String[] file = {"/system/app/Superuser/Superuser.apk", "/system/app/Superuser.apk", "/sbin/su", "/system/bin/su", "/system/xbin/su", "/data/local/x
        boolean result = false;
        for (String files : file) {
            File f = new File(files);
            result = f.exists();
            if (result) {
                break;
            }
        }
        return result;
    }

    public final boolean isRooted1() {
        try {
            Runtime.getRuntime().exec(new String[]{"su", "ls /data/data/"});
            return true;
        } catch (IOException e) {
            System.out.println(e);
            return false;
        }
    }
}
```

RootBeer

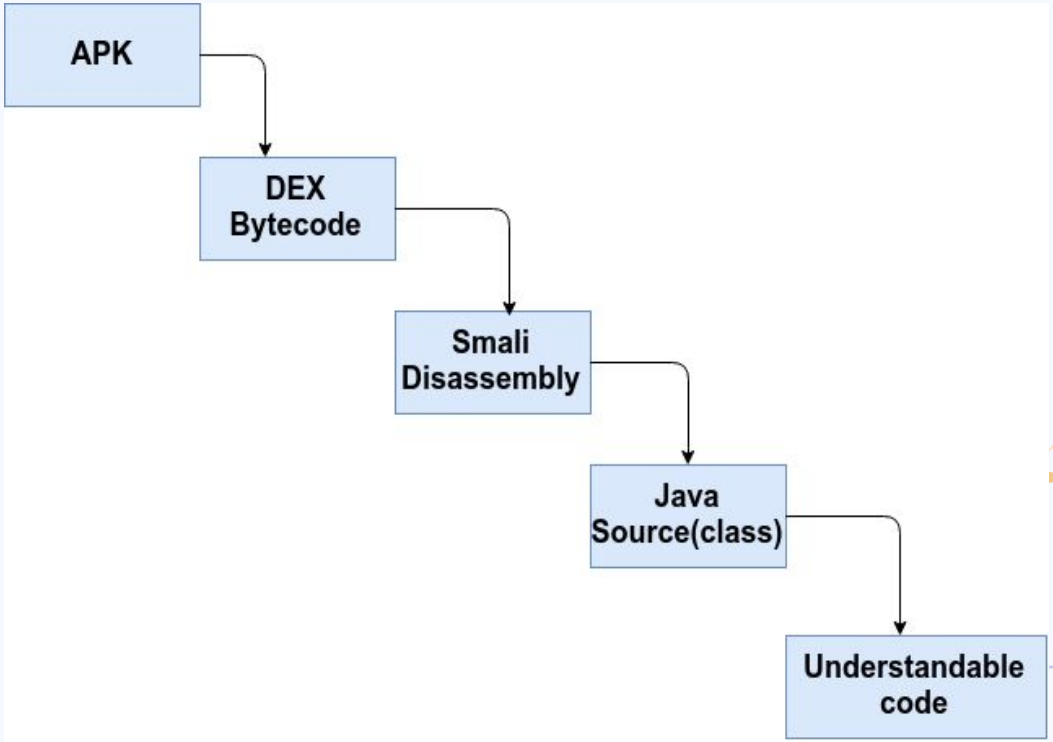
```
import java.util.NoSuchElementException;
import java.util.Scanner;

/* loaded from: classes.dex */
31  public class RootBeer {
        private boolean loggingEnabled = true;
        private final Context mContext;

32      public RootBeer(Context context) {
34          this.mContext = context;
        }

43      public boolean isRooted() {
            return detectRootManagementApps() || detectPotentiallyDangerousApps() || checkForBinary("su") ||
checkForDangerousProps() || checkForRWPaths() || detectTestKeys() || checkSuExists() || checkForRootNative() ||
checkForMagiskBinary();
        }

        @Deprecated
54      public boolean isRootedWithoutBusyBoxCheck() {
55          return isRooted();
        }

65      public boolean isRootedWithBusyBoxCheck() {
            return detectRootManagementApps() || detectPotentiallyDangerousApps() || checkForBinary("su") || checkForBinary(
"busybox") || checkForDangerousProps() || checkForRWPaths() || detectTestKeys() || checkSuExists() || checkForRootNative(
) || checkForMagiskBinary();
```

Sample Root Detection

```
    }

    void showRootStatus() {
        boolean isrooted = doesSuperuserApkExist("/system/app/Superuser.apk") || doesSUexist();
        if (isrooted) {
            this.root_status.setText("Rooted Device!!");
        } else {
            this.root_status.setText("Device not Rooted!!");
        }
    }

    private boolean doesSUexist() {
        Process process = null;
        try {
            process = Runtime.getRuntime().exec(new String[]{"/system/xbin/which", "su"});
            BufferedReader in = new BufferedReader(new InputStreamReader(process.getInputStream()));
            if (in.readLine() == null) {
                if (process != null) {
                    process.destroy();
                }
                return false;
            } else if (process != null) {
                process.destroy();
                return true;
            } else {
                return true;
            }
        } catch (Throwable th) {
            if (process != null) {
                process.destroy();
            }
            return false;
        }
    }

    private boolean doesSuperuserApkExist(String s) {
        File rootFile = new File("/system/app/Superuser.apk");
        Boolean doesexist = Boolean.valueOf(rootFile.exists());
        return doesexist.booleanValue();
    }

    protected void changePasswd() {
        Intent cP = new Intent(getApplicationContext(), ChangePassword.class);
        cP.putExtra("uname", this.uname);
        startActivity(cP);
    }

    protected void viewStatment() {
        Intent vS = new Intent(getApplicationContext(), ViewStatement.class);
        vS.putExtra("uname", this.uname);
        startActivity(vS);
    }
```

Reverse Engineering Apk Process

Several processes are required in reverse engineering an APK. Start by unpacking the APK using an archive extractor such as WinRAR, WinZip, etc. After that, the dex file will be disassembled and decompiled into java source which is still a java class file.



01.1

Manual Modification

Bypassing root detection using manual modification of smali

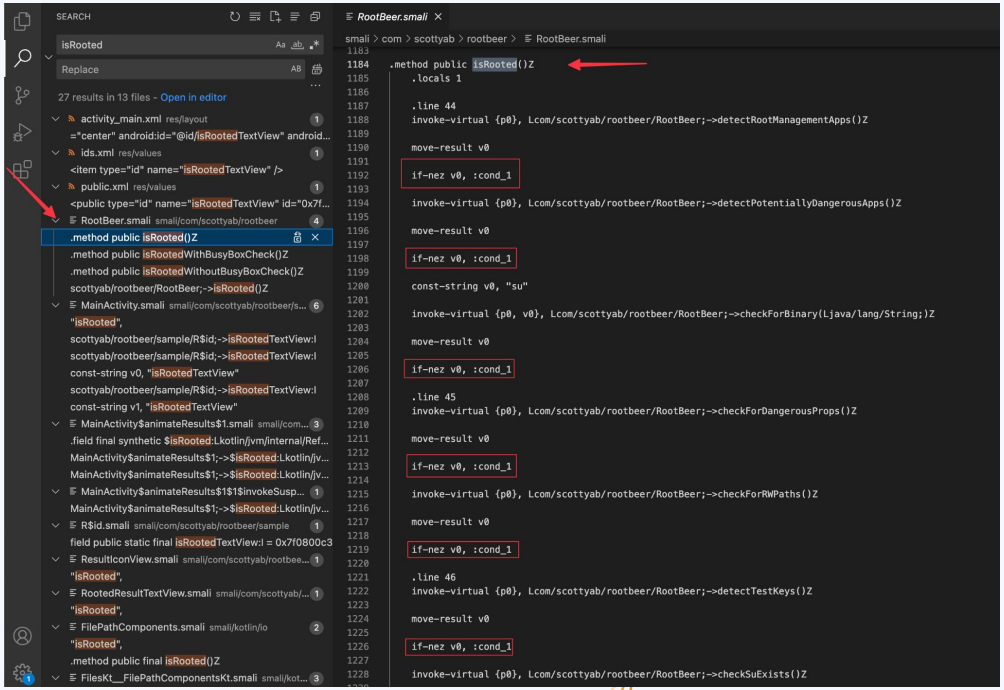
Extracting Apk

To bypass root detection using the smali patch method, the first step that must be done is to extract the content in the apk using apktool

```
❯ apktool d RootBeer\ Sample.apk
I: Using Apktool 2.8.1 on RootBeer Sample.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/petruknisme/Library/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
I: Copying META-INF/services directory
```

Searching the right function

Then we can search for the word isRooted with the help of a code editor. In this example case, the isRooted function is in the Rootbeer.smali file



As you can see, isRooted function are calling other function for checking:

1. Su Binary
2. RW Paths
3. Root via Native Checks
4. Magisk Specific checks
5. etc

Bypassing checkForSuBinary

In the **checkForSuBinary()** function, change **move-result v0** to **const/4 v0, 0x0** with the aim of making the value of the variable **v0** false instead of to taking a dynamic value from the result of the **checkForBinary()** function

```
.method public checkForSuBinary()Z
        .locals 1
        const-string v0, "su"

        .line 160
        invoke-virtual {p0, v0}, Lcom/scottyab/rootbeer/RootBeer;->checkForBinary(Ljava/lang/String;)Z

        const/4 v0, 0x0 <-- Edited

        return v0
```

Bypassing checkSuExist

In the **checkSuExist()** function, change value of **v0** to **0x0** after **if-eqz v2**

```
.method public checkSuExists()Z
    if-eqz v2, :cond_0

    const/4 v0, 0x0 <-- Edited

    :cond_0
    if-eqz v1, :cond_1

    .line 405
    invoke-virtual {v1}, Ljava/lang/Process;->destroy()V

    :cond_1
    return v0

    :catchall_0
    nop

    if-eqz v1, :cond_2

    invoke-virtual {v1}, Ljava/lang/Process;->destroy()V

    :cond_2
    return v0
.end method
```

Bypassing checkForRWPaths

In the **checkForRWPaths()** function, add **return v1** at the end of the function so that the function always returns false because the **v1** variable is already false.

```
.method public checkForRWPaths()Z

    --------------------SNIPPET--------------
    const/4 v1, 0x0
    return v1 <-- Added

    goto/16 :goto_0

    :cond_9
    return v5
.end method
```

Bypassing checkForRootNative

In the **checkForRootNative()** function, change the **v1** value at the end of the function to **0x0** so that the returned value is false.

```
.method public checkForRootNative()Z
--------------------SNIPPET-------------

        if-lez v0, :cond_2
        const/4 v1, 0x0 <-- Edited
        :catch_0
        :cond_2
        return v1
.end method
```

Bypassing checkForMagiskBinary

In the **checkForMagiskBinary()** function, change **move-result v0** to **const/4 v0, 0x0** with the aim of making the value of the **variable v0** false instead of taking a dynamic value from the result of the **checkForBinary()** function

```
.method public checkForMagiskBinary()Z
    .locals 1

    const-string v0, "magisk"

    .line 167
    invoke-virtual {p0, v0}, Lcom/scottyab/rootbeer/RootBeer;->checkForBinary(Ljava/lang/String;)Z

    const/4 v0, 0x0 <-- Edited
    return v0
.end method
```

Rebuild, sign and install

```
~/Downloads/IDSECCONF via ☕ v17.0.8  14s
❯ apktool b RootBeer/ -o repacked.apk && java -jar uber-apk-signer-1.3.0.jar -a repacked.apk && adb
install repacked-aligned-debugSigned.apk
I: Using Apktool 2.8.1
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: repacked.apk
source:
        /Users/petruknisme/Downloads/IDSECCONF
zipalign location: BUILT_IN
        /var/folders/wq/qx9fwmh5713g58xyqjl0g3g80000gn/T/uapksigner-3288691317891163224/mac-
zipalign-33_0_24436693956892857862.tmp
keystore:
        [0] 161a0018
/private/var/folders/wq/qx9fwmh5713g58xyqjl0g3g80000gn/T/temp_6593179767234436346_debug.keystore
(DEBUG_EMBEDDED)

01. repacked.apk

        SIGN
        file: /Users/petruknisme/Downloads/IDSECCONF/repack
        checksum: 9e6919f5ca108145f2faf6aef558168dc8352c88
        - zipalign success
        - sign success

        VERIFY
        file: /Users/petruknisme/Downloads/IDSECCONF/repack
        checksum: fe44a001c3d2d6594bba77e11d8e000a7ef13941
        - zipalign verified
        - signature verified [v1, v2, v3]
                2 warnings
                Subject: CN=Android Debug, OU=Android, O=
```

SHA256withRSA

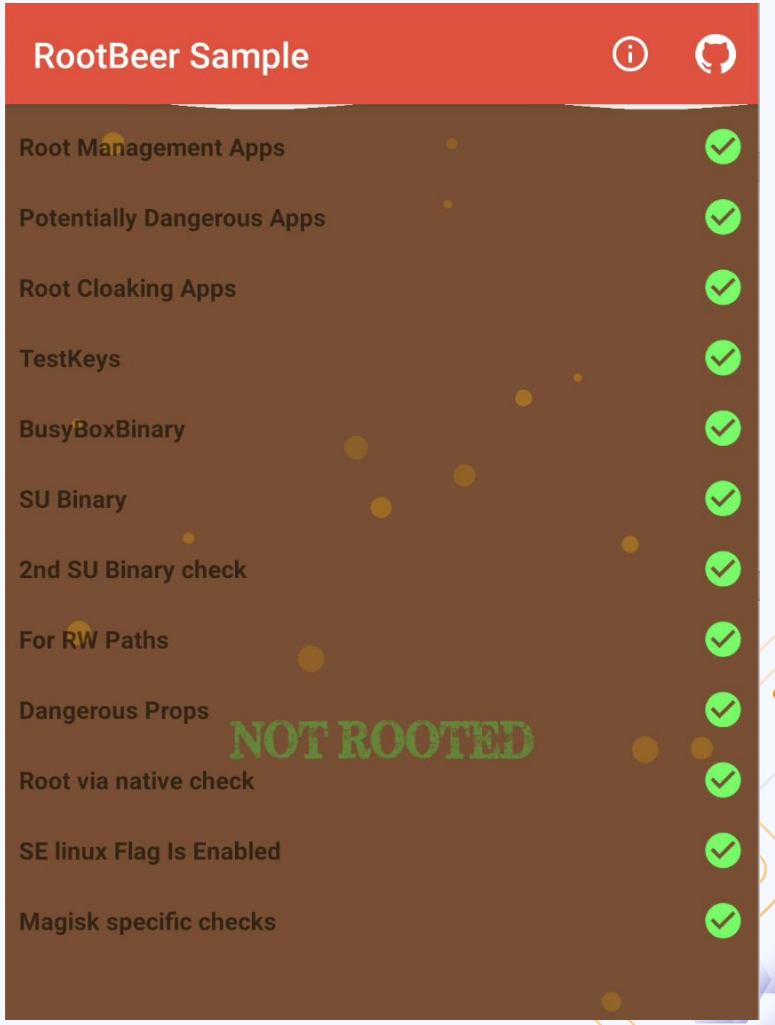
SHA256: 1e08a903aef9c3a721510b64ec764d01d3d094eb954161b62544ea8f187b5953 /

Expires: Fri Mar 11 03:10:05 WIB 2044

```
[Tue Sep 19 12:23:36 WIB 2023][v1.3.0]
Successfully processed 1 APKs and 0 errors in 1.05 seconds.
Performing Incremental Install
Serving...
Unknown command: install-incremental
Performing Streamed Install
Success
```

Result

We are successfully bypassing root detection with manual modification of smali files



01.2

Frida Instrumentation

Bypassing root detection using frida instrumentation

Frida Instrumentation

As we know, we need to bypass these protection to be able to run the app in rooted device:

- SU Binary
- RW Paths
- Root via Native Checks
- Magisk Specific Checks

I assume that we already know how to write frida script. For the first step, we will try to hook one of the function to make sure that our script is working

Frida Instrumentation

```
❯ frida -U -l isroot.js -f com.scottyab.rootbeer.sample

     ____
    / _  |   Frida 16.1.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/
   . . . .
   . . . .   Connected to Mi A2 (id=4920eaee)
Spawned `com.scottyab.rootbeer.sample`. Resuming main thread!
[Mi A2::com.scottyab.rootbeer.sample ]-> Fungsi checkForSuBinary sudah ter-hook
```

```
Java.perform(function(){
// Membuat instance untuk kelas RootBeer yang akan di hook
var RB = Java.use("com.scottyab.rootbeer.RootBeer");

// Hook fungsi checkForSuBinary()
RB.checkForSuBinary.overload().implementation = function(){
    console.log("Fungsi checkForSuBinary sudah ter-hook");
    return false
};
});
```

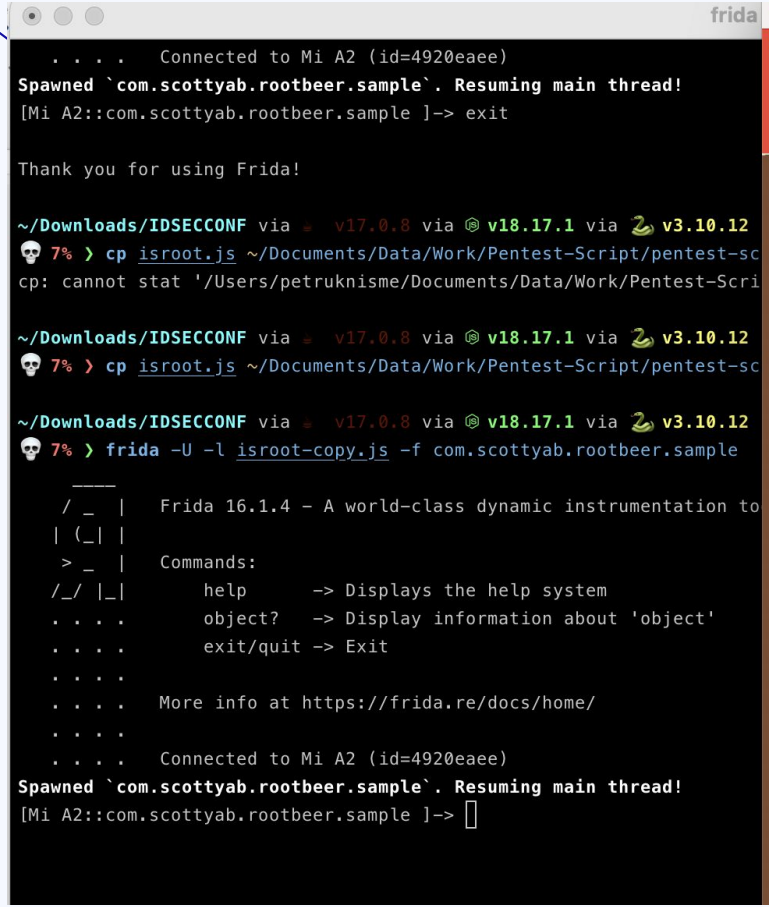
Final frida Script

For the rest of functions, we just need to copy and modify with the function name that we will hook.

```
Java.perform(function(){
  // Membuat instance untuk kelas RootBeer yang akan di hook
  var RB = Java.use("com.scottyab.rootbeer.RootBeer");

  const functionsBypass = [
    "checkForRWPaths",
    "checkForRootNative",
    "checkForMagiskBinary",
    "checkForSuBinary",
    "checkSuExists",
  ];

  functionsBypass.forEach((func) => {
    RB[func].implementation = function() {
      return false;
    };
  });
});
```

But, for keeping it simple(KISS) and follow DRY principle, I've modified the script to be more simple, short, and easy to understand.

Result


frida
. . . . Connected to Mi A2 (id=4920eaee)
Spawned `com.scottyab.rootbeer.sample`. Resuming main thread!
[Mi A2::com.scottyab.rootbeer.sample]-> exit
Thank you for using Frida!
~/Downloads/IDSECCONF via v17.0.8 via v18.17.1 via v3.10.12
7% ❯ cp isroot.js ~/Documents/Data/Work/Pentest-Script/pentest-sc
cp: cannot stat '/Users/petruknisme/Documents/Data/Work/Pentest-Scri
~/Downloads/IDSECCONF via v17.0.8 via v18.17.1 via v3.10.12
7% ❯ cp isroot.js ~/Documents/Data/Work/Pentest-Script/pentest-sc
~/Downloads/IDSECCONF via v17.0.8 via v18.17.1 via v3.10.12
7% ❯ frida -U -l isroot-copy.js -f com.scottyab.rootbeer.sample
Frida 16.1.4 - A world-class dynamic instrumentation to
Commands:
help -> Displays the help system
object? -> Display information about 'object'
exit/quit -> Exit
More info at https://frida.re/docs/home/
. . . . Connected to Mi A2 (id=4920eaee)
Spawned `com.scottyab.rootbeer.sample`. Resuming main thread!
[Mi A2::com.scottyab.rootbeer.sample]->
MI A2
3:44
RootBeer Sample
Root Management Apps
Potentially Dangerous Apps
Root Cloaking Apps
TestKeys
BusyBoxBinary
SU Binary
2nd SU Binary check
For RW Paths
Dangerous Props
NOT ROOTED
Root via native check
SE linux Flag Is Enabled
Magisk specific checks

01.3

Objection

Bypassing root detection using objection

What?

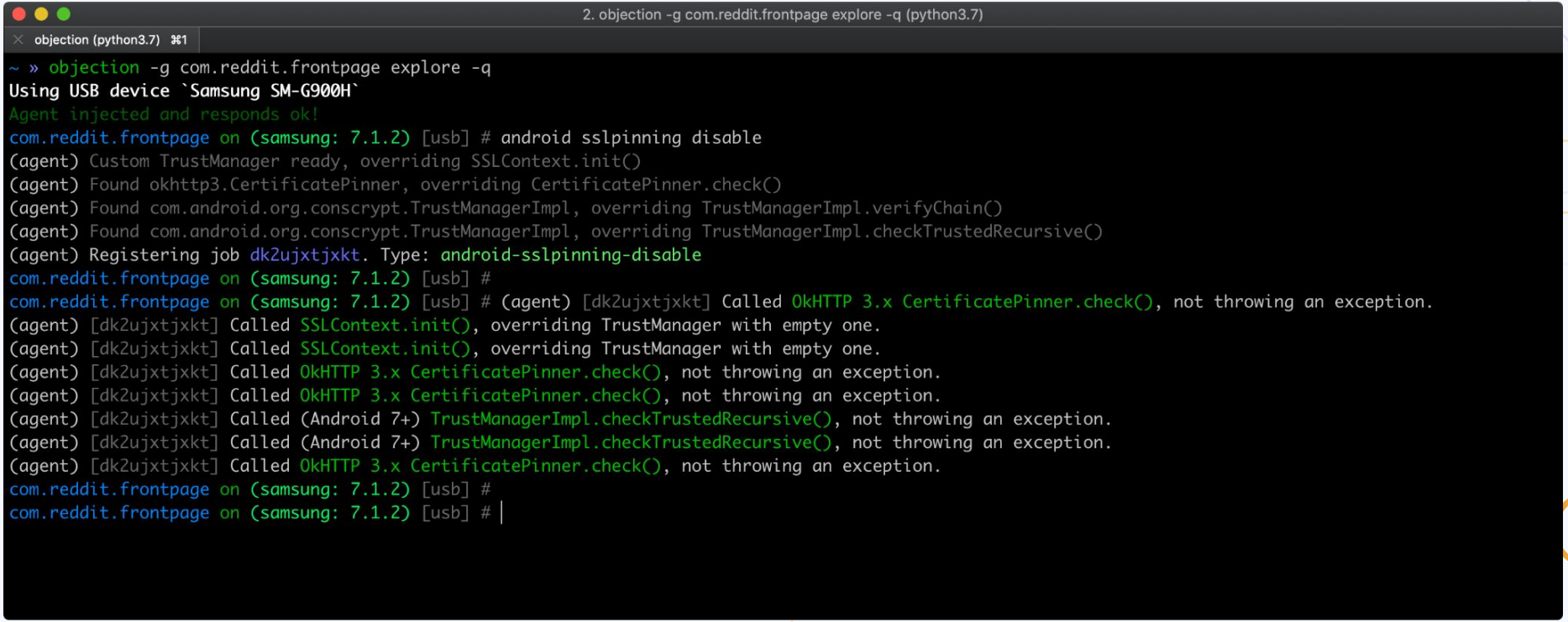
objection is a runtime mobile exploration toolkit, powered by Frida, built to help you assess the security posture of your mobile applications, **without** needing a **jailbreak**.

- Supports both iOS and Android.
- Inspect and interact with container file systems.
- Bypass SSL pinning.
- Dump keychains.
- Perform memory related tasks, such as dumping & patching.
- Explore and manipulate objects on the heap.
- And much, much more...

Installation

Installation is simply a matter of **pip3 install objection.** This will give you the **objection** command. You can update an existing **objection** installation with **pip3 install --upgrade** objection.

What?

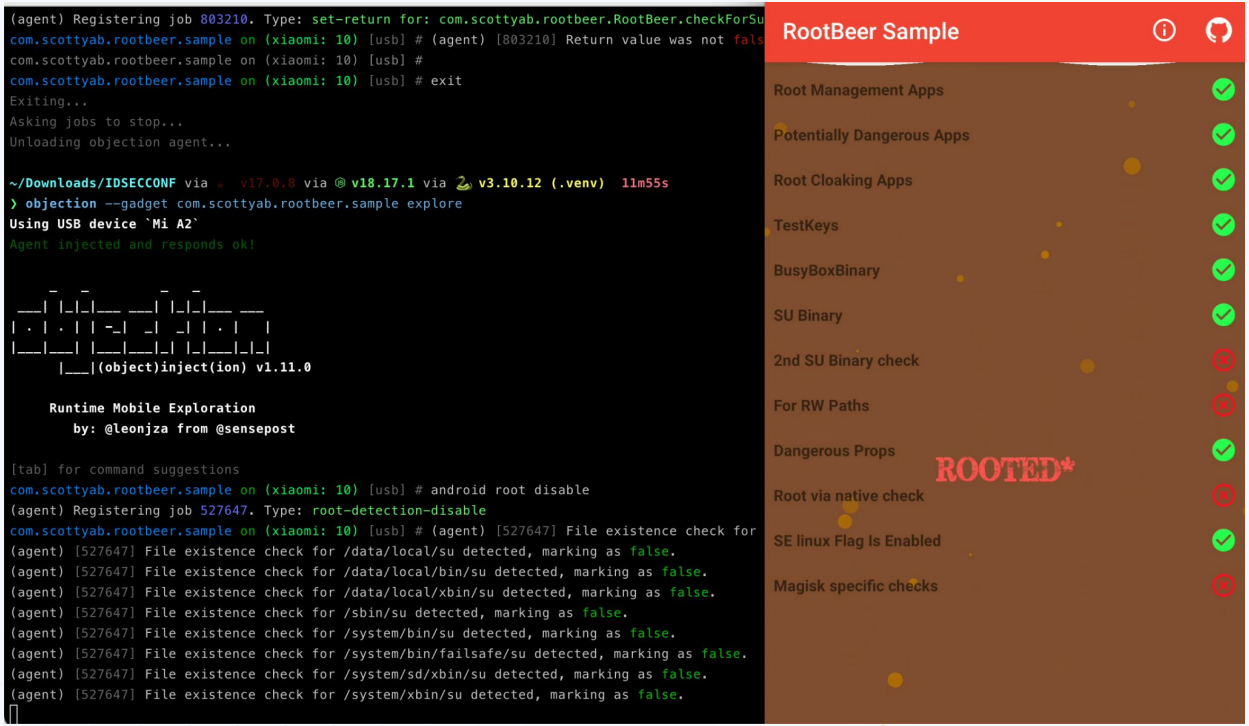


https://github.com/sensepost/objection/wiki/Screenshots

No need script, only command line

```
❯ objection --gadget com.scottyab.rootbeer.sample explore
Using USB device `Mi A2`
Agent injected and responds ok!

     _     _       _     _
 ___| |_ _|_|___ ___| |_ _|_|___ ___
| . | . | | -_|  _|  _| | . |    |
|___|___| |___|___|_| |_|___|_|_|
      |___|(object)inject(ion) v1.11.0

     Runtime Mobile Exploration
        by: @leonjza from @sensepost

[tab] for command suggestions
com.scottyab.rootbeer.sample on (xiaomi: 10) [usb] #
```

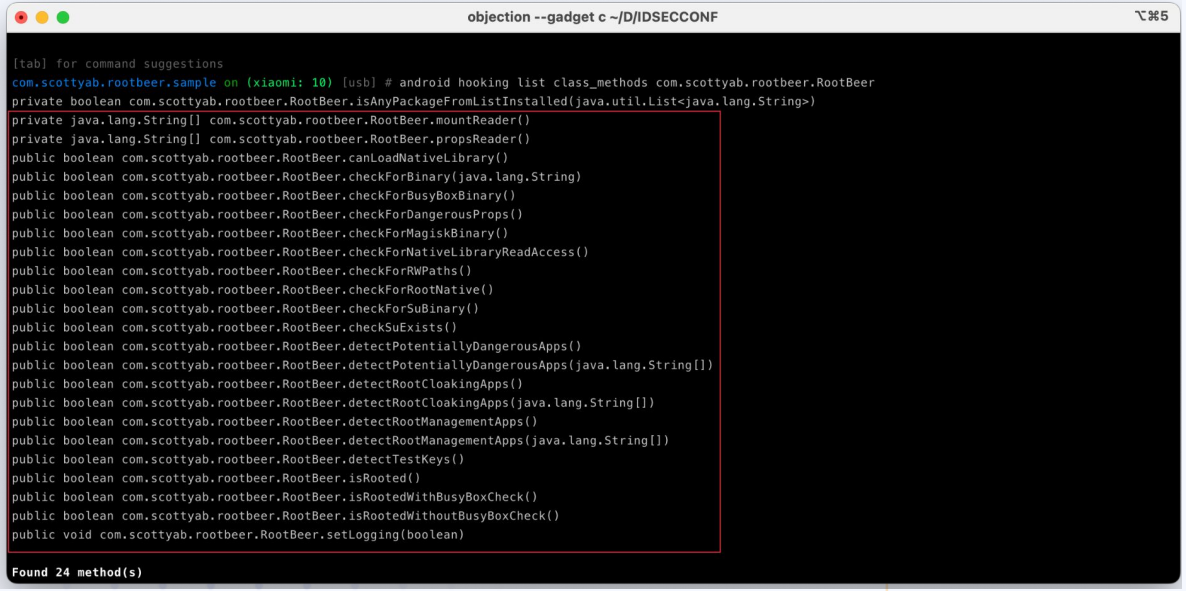
No need script, only command line



Partially success bypassing rootbeer checker with objection command “android root disable”

Hook & patch the right function

android hooking list class_methods com.example.app.class



Before patching the functions, we need to know the class first

Set return value

android hooking set return_value com.app.example.class.method

After we know the right class_methods, we can set the return value

01.4

Magisk Zygisk

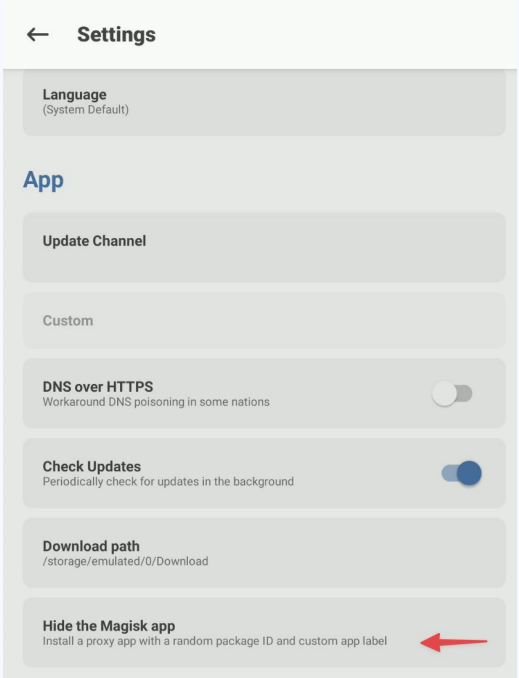
Bypassing root detection using Magisk Zygisk

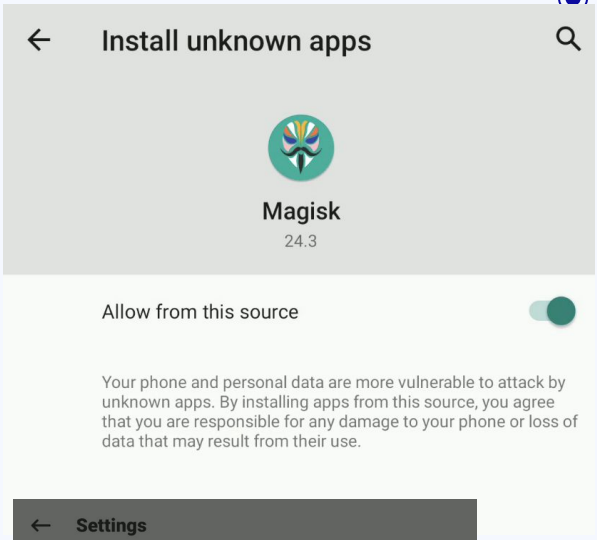
Magisk

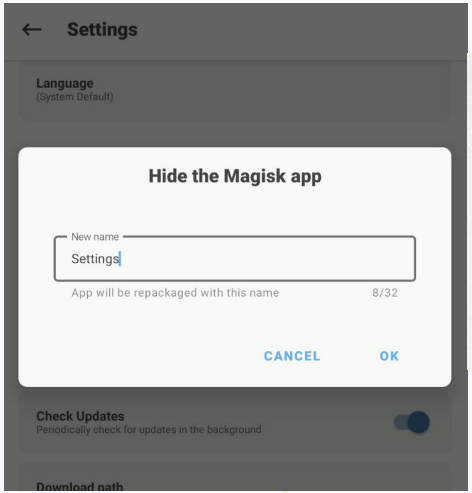
In simple terms, Magisk is a tool to help users gain root access by patching the ROM. For further information regarding installation and configuration, please refer to https://github.com/topjohnwu/Magisk

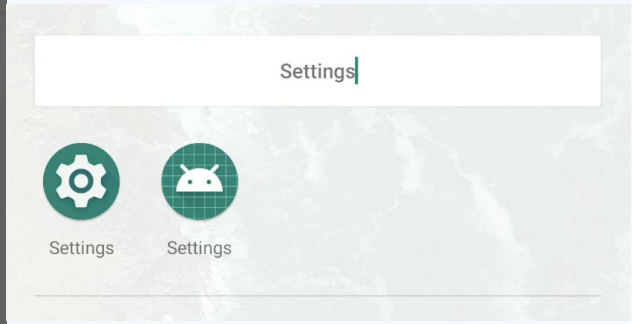
The easiest method is to rely on Zygisk Denylist in the Magisk application. To be able to enjoy **Zygisk**, Magisk version that must be installed is **v24.1+**.

Setup

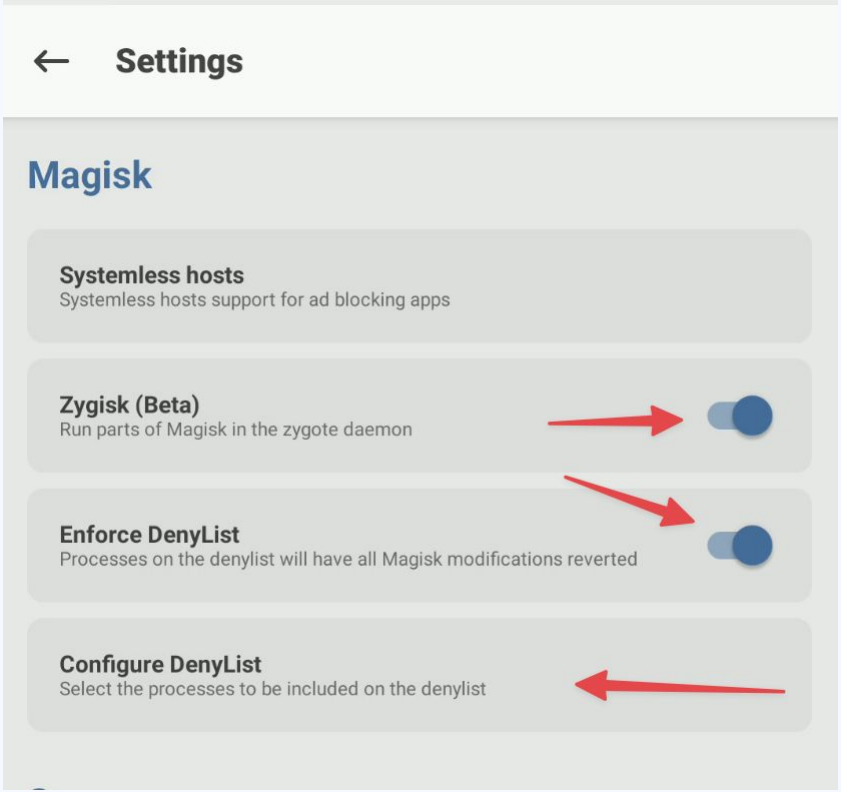

Settings
Language
(System Default)
App
Update Channel
Custom
DNS over HTTPS
Workaround DNS poisoning in some nations
Check Updates
Periodically check for updates in the background
Download path
/storage/emulated/0/Download
Hide the Magisk app
Install a proxy app with a random package ID and custom app label

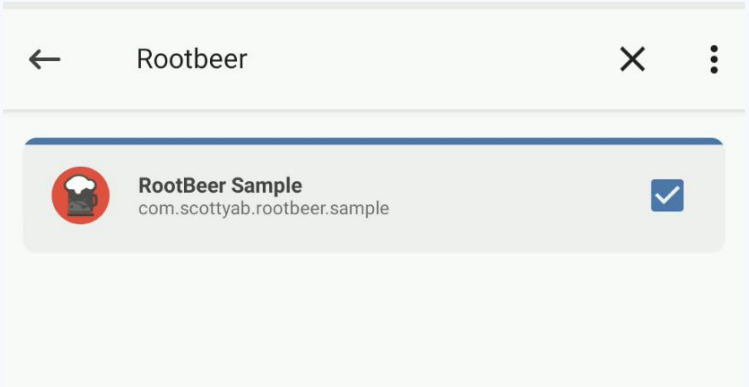

Install unknown apps
Magisk
24.3
Allow from this source
Your phone and personal data are more vulnerable to attack by unknown apps. By installing apps from this source, you agree that you are responsible for any damage to your phone or loss of data that may result from their use.


Settings
Language
(System Default)
Hide the Magisk app
New name
Settings
App will be repackaged with this name
8/32
CANCEL
OK
Check Updates
Periodically check for updates in the background
Download path

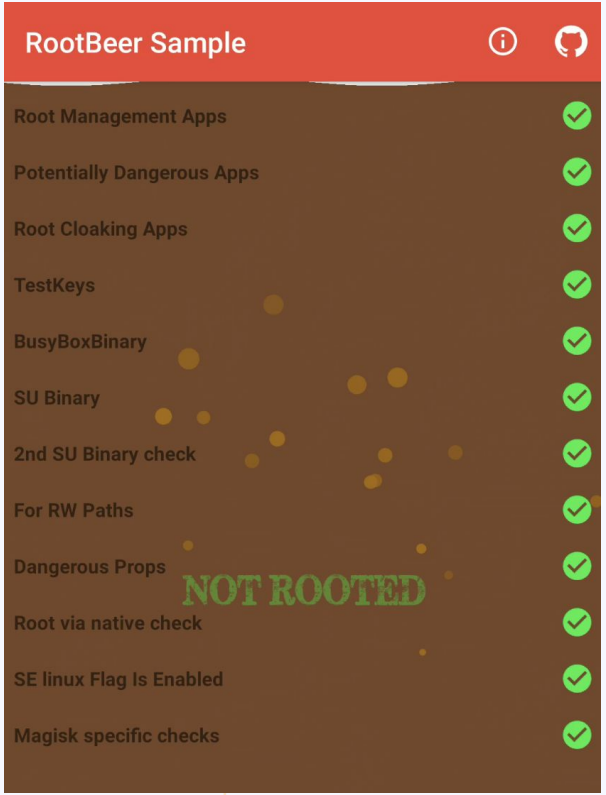

Settings
Settings
Settings

Setup


Settings
Magisk
Systemless hosts
Systemless hosts support for ad blocking apps
Zygisk (Beta)
Run parts of Magisk in the zygote daemon
Enforce DenyList
Processes on the denylist will have all Magisk modifications reverted
Configure DenyList
Select the processes to be included on the denylist


Rootbeer
RootBeer Sample
com.scottyab.rootbeer.sample

Result


RootBeer Sample
Root Management Apps
Potentially Dangerous Apps
Root Cloaking Apps
TestKeys
BusyBoxBinary
SU Binary
2nd SU Binary check
For RW Paths
Dangerous Props
NOT ROOTED
Root via native check
SE linux Flag Is Enabled
Magisk specific checks

02

SSL Pinning

SSL Pinning

SSL pinning is a technique that helps to prevent MiTM attacks by hardcoding the SSL/TLS certificate's public key into the app or device. This means that when the app or device communicates with the server, it will compare the server's SSL/TLS certificate's public key with the one that is hardcoded into the app or device.

https://www.indusface.com/learning/what-is-ssl-pinning-a-quick-walk-through/

02.1

Manual Modification

Bypassing SSL Pinning using manual modification of smali

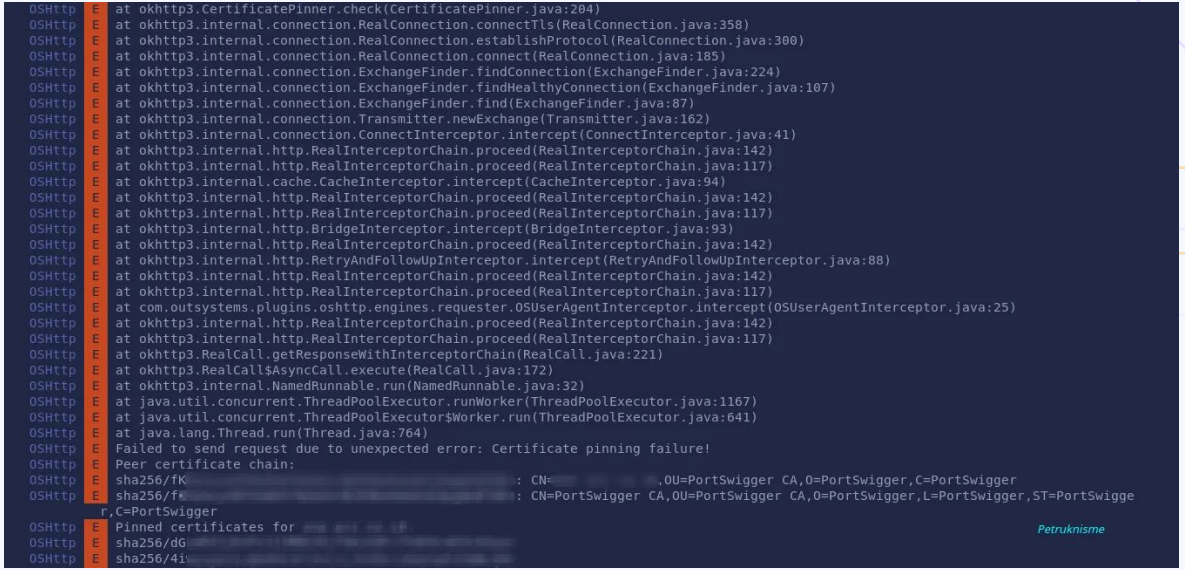
Manual

In this discussion, I will start by explaining how we can bypass the SSL Pinning process by manually modifying smalis. The target that will be used in this case study is an android application that use Cordova Framework.

When connecting to the server, the application refused the connection because the certificate on the device did not match, because I was intercept the traffic using Burp Suite. With the help of “adb logcat | logcat-color”, I can find out the error message given by the application when it refuses a connection to the server.

Manual

As you can see, android certificate pinning is failing and complaining about mismatch certificate hash. Our peer certificate chain is for PortSwigger(BurpSuite), but the Pinned Certificate for a domain(masked) is defined in the android resource file.



Best Approach

Decompile apk and search for the right string.

```
java -jar apktool.jar d base.apk -o decompiled
I: Using Apktool 2.5.0-dirty on base.apk I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /home/user/.local/share/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

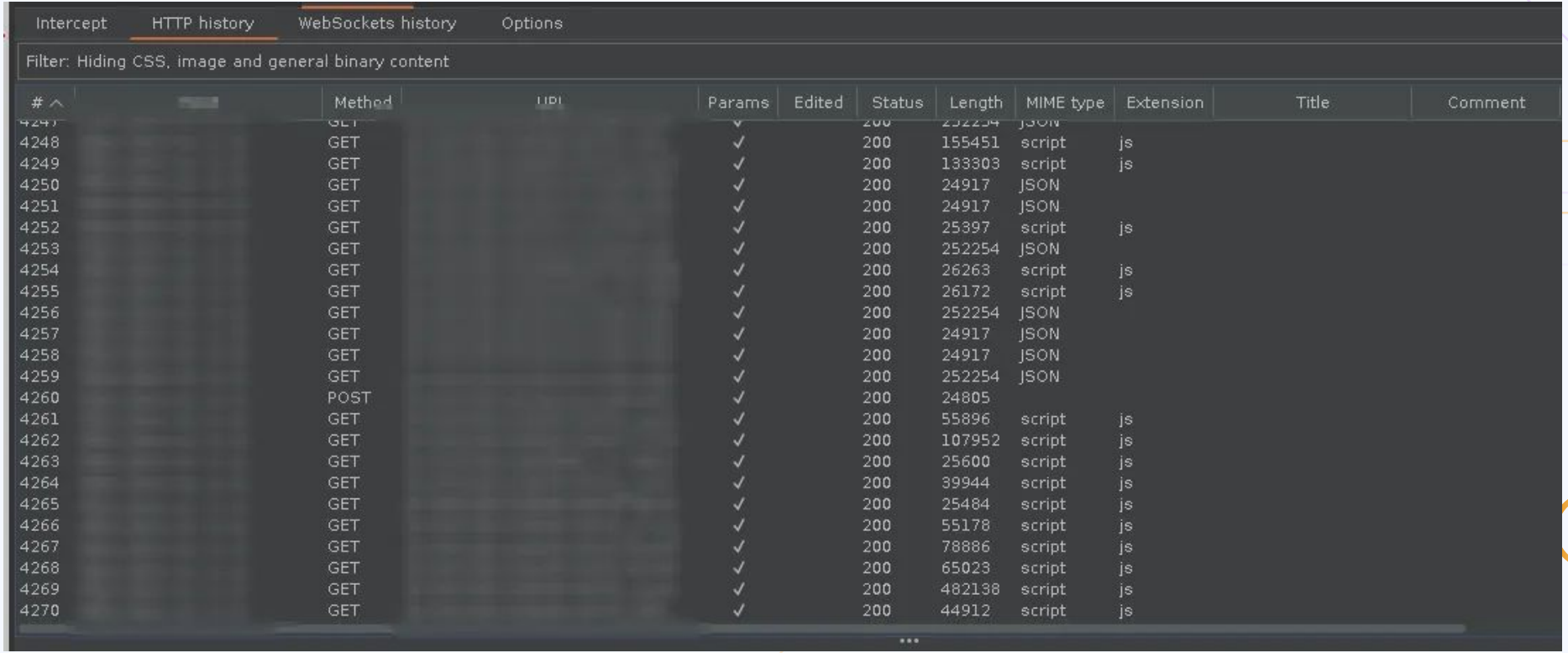
In this case, the file containing the word is in resource/assets/www/pinning/ . After knowing which file is appropriate, the next step is to modify the string sha256/dGxxxxx to sha256/fKxxxxx referring to the previous error message.

```
grep -r "sha256/dGxxxxxxxxxxx" decompiled/*
```

Recompile

```
$ java -jar apktool.jar b decompiled -o app-certpin-bypass.apk
I: Using Apktool 2.5.0-dirty
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...
$ ls
app-certpin-bypass.apk base.apk decompiled
$ java -jar uber-apk-signer.jar -a app-certpin-bypass.apk
source:
        /home/user/apk zipalign location: PATH
        /bin/zipalign
keystore:
        [0] 161a0018 /tmp/temp_10038107399149177087_debug.keystore (DEBUG_EMBEDDED)
        01. app-certpin-bypass.apk

  SIGN
        file: /home/user/apk/app-certpin-bypass.apk (24.53 MiB)
        checksum: 79489a567c31bcc1f82bf45b5d9d6ffa1de2ca71487a538505b31a8b8e01435c (sha256)
        - zipalign success
        - sign success

  VERIFY
        file: /home/user/apk/app-certpin-bypass-aligned-debugSigned.apk (24.65 MiB)
        checksum: 5ade3620151dd81029508faa40fa1a63240cf1e55150702aa79b3928f2bb9703 (sha256)
        - zipalign verified
        - signature verified [v1, v2, v3]
        Subject: CN=Android Debug, OU=Android, O=US, L=US, ST=US, C=US
        SHA256: 1e08a903aef9c3a721541b54ec764e01d3d094eb954161b62544ea8f187b5953 / SHA256withRSA
```

Result



Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

#	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment
4248	GET		✓		200	155451	script	js		
4249	GET		✓		200	133303	script	js		
4250	GET		✓		200	24917	JSON			
4251	GET		✓		200	24917	JSON			
4252	GET		✓		200	25397	script	js		
4253	GET		✓		200	252254	JSON			
4254	GET		✓		200	26263	script	js		
4255	GET		✓		200	26172	script	js		
4256	GET		✓		200	252254	JSON			
4257	GET		✓		200	24917	JSON			
4258	GET		✓		200	24917	JSON			
4259	GET		✓		200	252254	JSON			
4260	POST		✓		200	24805				
4261	GET		✓		200	55896	script	js		
4262	GET		✓		200	107952	script	js		
4263	GET		✓		200	25600	script	js		
4264	GET		✓		200	39944	script	js		
4265	GET		✓		200	25484	script	js		
4266	GET		✓		200	55178	script	js		
4267	GET		✓		200	78886	script	js		
4268	GET		✓		200	65023	script	js		
4269	GET		✓		200	482138	script	js		
4270	GET		✓		200	44912	script	js		

02.2

Objection

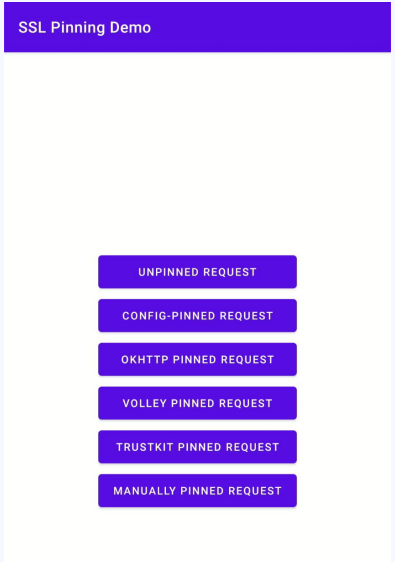
Bypassing SSL Pinning using Objection

Approach

As in the discussion for bypassing root detection, we can also do the same thing with the help of objection. There are two methods that can be done, using the built-in objection feature or manual patching of the return_value from class_method.

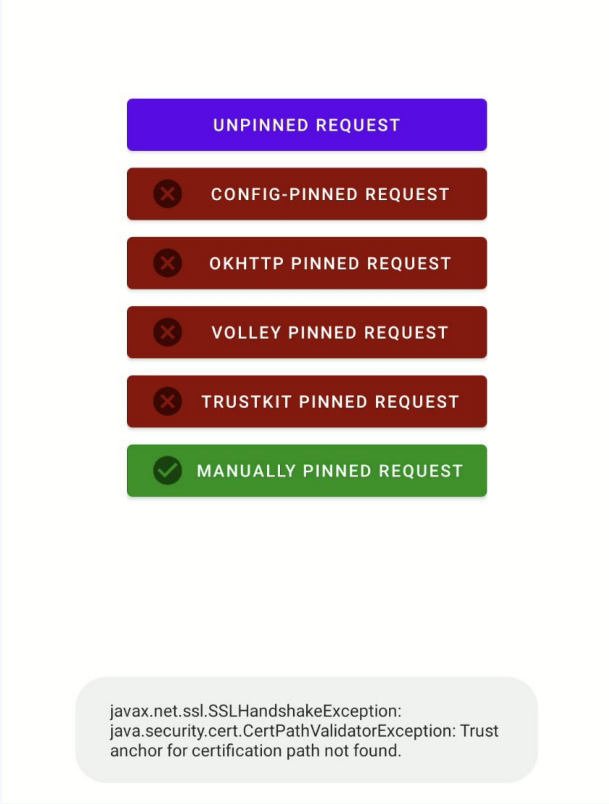
For the study case, we will use

https://github.com/httptoolkit/android-ssl-pinning-demo/releases/download/v1.3.1/pinning-demo.apk

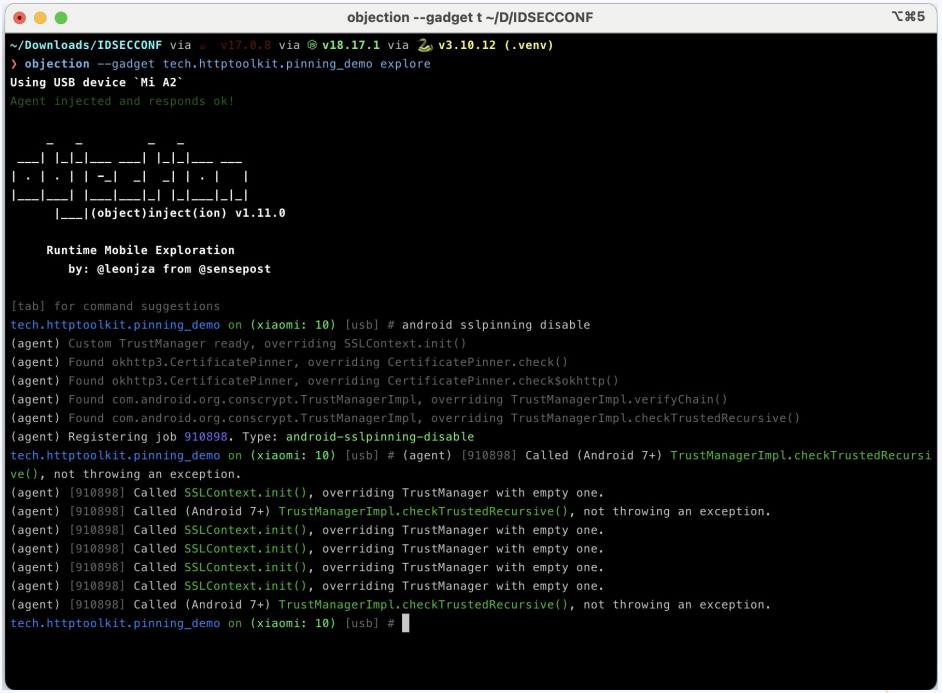


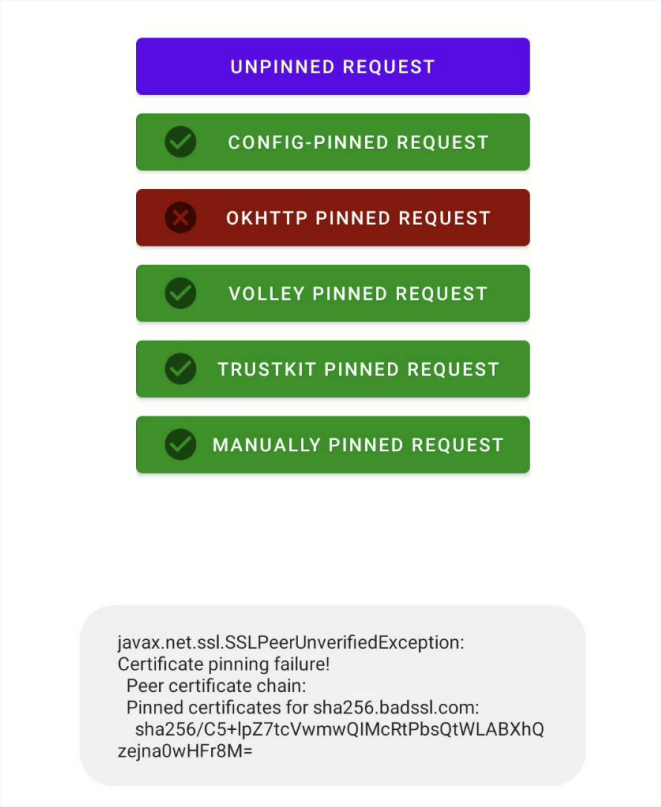
Approach

Error when intercepted with burpsuite

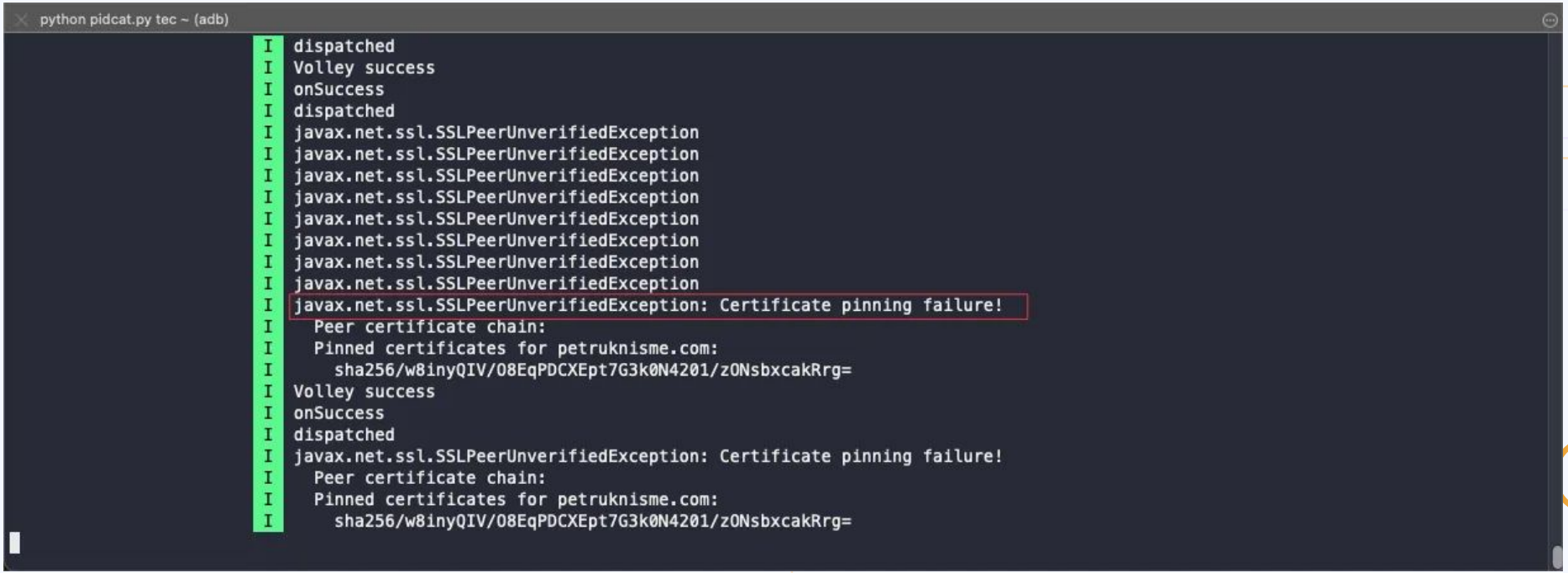


Objection


objection --gadget t ~/D/IDSECCONF
~/Downloads/IDSECCONF via v17.0.8 via v18.17.1 via v3.10.12 (.venv)
objection --gadget tech.httptoolkit.pinning_demo explore
Using USB device `Mi A2`
Agent injected and responds ok!
|_|(object)inject(ion) v1.11.0
Runtime Mobile Exploration
by: @leonjza from @sensepost
[tab] for command suggestions
tech.httptoolkit.pinning_demo on (xiaomi: 10) [usb] # android sslpinning disable
(agent) Custom TrustManager ready, overriding SSLContext.init()
(agent) Found okhttp3.CertificatePinner, overriding CertificatePinner.check()
(agent) Found okhttp3.CertificatePinner, overriding CertificatePinner.check$okhttp()
(agent) Found com.android.org.conscrypt.TrustManagerImpl, overriding TrustManagerImpl.verifyChain()
(agent) Found com.android.org.conscrypt.TrustManagerImpl, overriding TrustManagerImpl.checkTrustedRecursive()
(agent) Registering job 910898. Type: android-sslpinning-disable
tech.httptoolkit.pinning_demo on (xiaomi: 10) [usb] # (agent) [910898] Called (Android 7+) TrustManagerImpl.checkTrustedRecursive(), not throwing an exception.
(agent) [910898] Called SSLContext.init(), overriding TrustManager with empty one.
(agent) [910898] Called (Android 7+) TrustManagerImpl.checkTrustedRecursive(), not throwing an exception.
(agent) [910898] Called SSLContext.init(), overriding TrustManager with empty one.
(agent) [910898] Called SSLContext.init(), overriding TrustManager with empty one.
(agent) [910898] Called SSLContext.init(), overriding TrustManager with empty one.
(agent) [910898] Called SSLContext.init(), overriding TrustManager with empty one.
(agent) [910898] Called (Android 7+) TrustManagerImpl.checkTrustedRecursive(), not throwing an exception.
tech.httptoolkit.pinning_demo on (xiaomi: 10) [usb] #

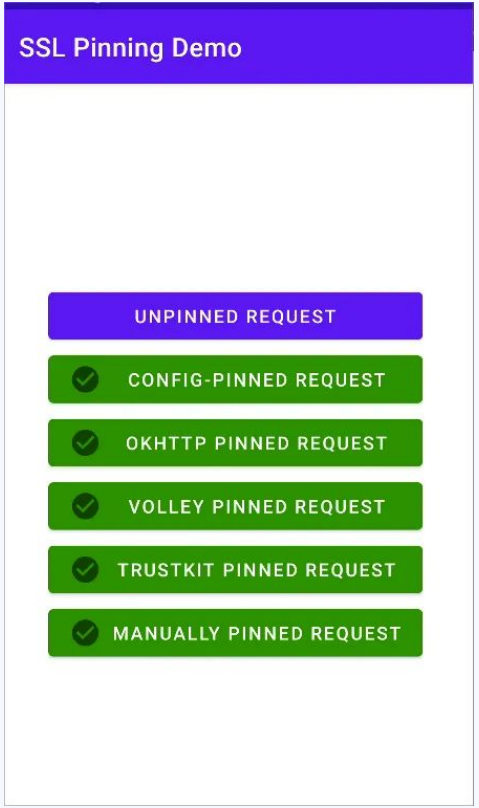

UNPINNED REQUEST
CONFIG-PINNED REQUEST
OKHTTP PINNED REQUEST
VOLLEY PINNED REQUEST
TRUSTKIT PINNED REQUEST
MANUALLY PINNED REQUEST
javax.net.ssl.SSLPeerUnverifiedException:
Certificate pinning failure!
Peer certificate chain:
Pinned certificates for sha256.badssl.com:
sha256/C5+lpZ7tcVwmwQIMcRtPbsQtWLABXhQ
zejna0wHFr8M=

Debugging


python pidcat.py tec ~ (adb)
I dispatched
I Volley success
I onSuccess
I dispatched
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException
I javax.net.ssl.SSLPeerUnverifiedException: Certificate pinning failure!
I Peer certificate chain:
I Pinned certificates for petruknisme.com:
I sha256/w8inyQIV/O8EqPDCXEpt7G3k0N4201/zONsbxcakRrg=
I Volley success
I onSuccess
I dispatched
I javax.net.ssl.SSLPeerUnverifiedException: Certificate pinning failure!
I Peer certificate chain:
I Pinned certificates for petruknisme.com:
I sha256/w8inyQIV/O8EqPDCXEpt7G3k0N4201/zONsbxcakRrg=

Debugging

```
Java.perform(function () {

const UnverifiedCertError = Java.use('javax.net.ssl.SSLPeerUnverifiedException');
UnverifiedCertError.$init.implementation = function (str) {

        const stackTrace = Java.use('java.lang.Thread').currentThread().getStackTrace();
        const exceptionStackIndex = stackTrace.findIndex(stack =>
            stack.getClassName() === "javax.net.ssl.SSLPeerUnverifiedException"
        );
        const callingFunctionStack = stackTrace[exceptionStackIndex + 1];

        const className = callingFunctionStack.getClassName();
        const methodName = callingFunctionStack.getMethodName();
        console.log(`      ================`);
        console.log(`      Debugging Stack Trace for SSLPeerUnverifiedException`);
        console.log(`      Thrown by ${className}->${methodName}`);
    };

        try {
            // Bypass OkHTTPv3 {4}
            const okhttp3_Activity_4 = Java.use('okhttp3.CertificatePinner');
            okhttp3_Activity_4.check$okhttp.overload('java.lang.String', 'kotlin.jvm.functions.Function0').implementation = function(a,
                console.log('  --> Bypassing OkHTTPv3 ($okhttp): ' + a);
                return;
            };
            console.log('[+] OkHTTPv3 ($okhttp)');
        } catch (err) {
            console.log('[ ] OkHTTPv3 ($okhttp)');
        }
});
```



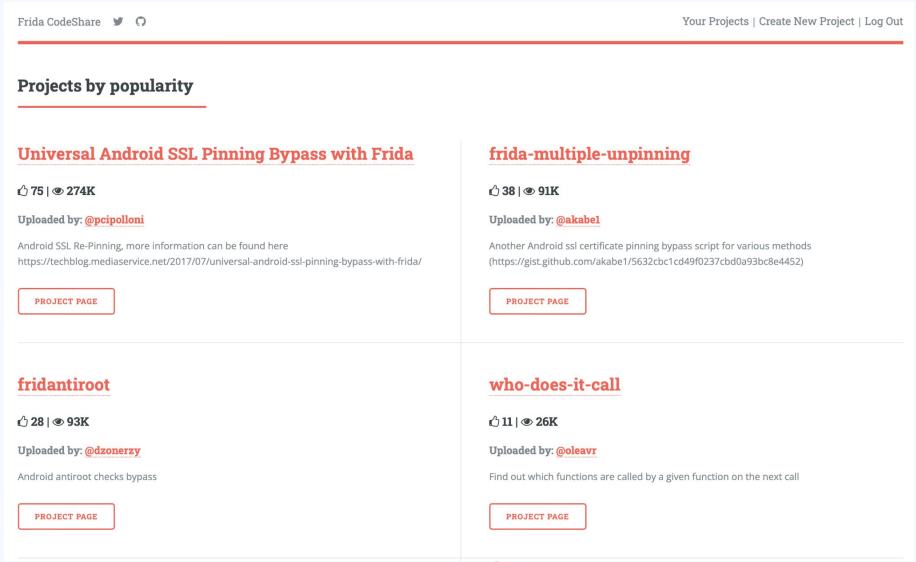
02.3

Frida

Bypassing SSL Pinning using Frida

Frida

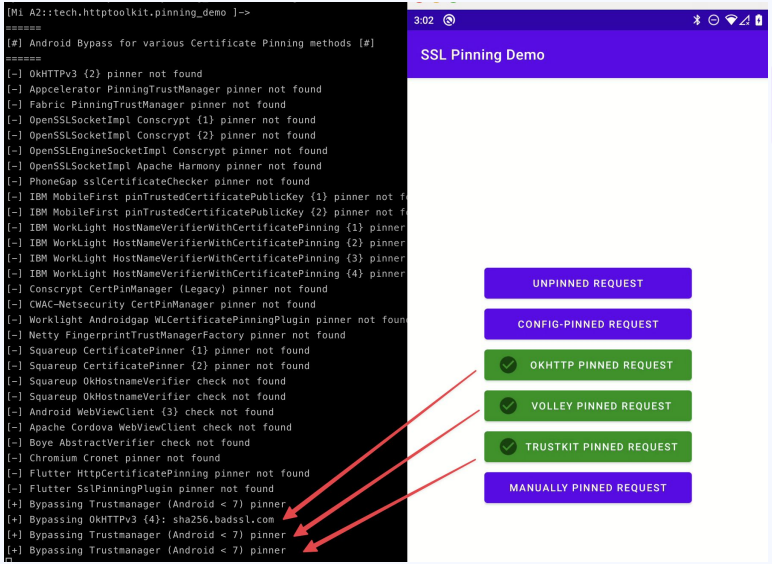
For bypassing SSL Pinning with Frida, we can use frida codeshare or create our own script.



Frida-codeshare

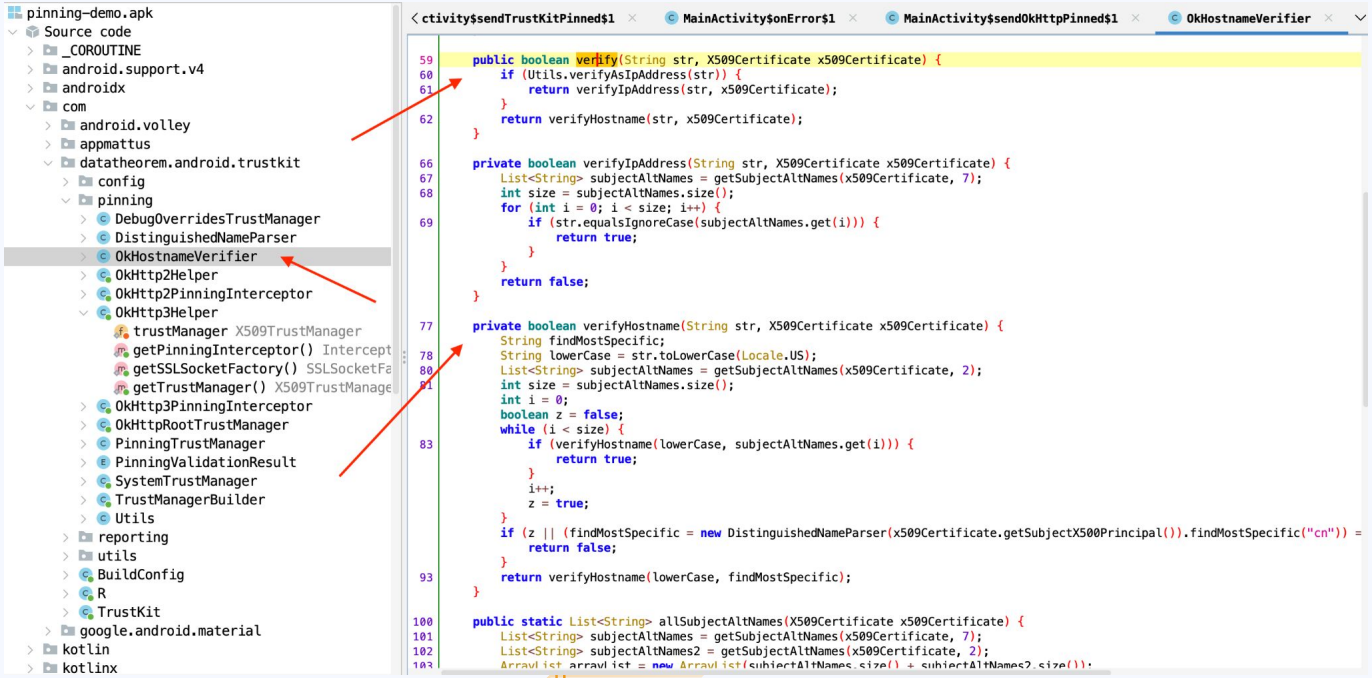
Bypassing previous ssl pinning protection that failed when using objection, we can use frida codeshare from https://codeshare.frida.re/@akabe1/frida-multiple-unpinning/

```
❯ frida --codeshare akabe1/frida-multiple-unpinning -U -f tech.httptoolkit.pinning_demo
     ____
    / _  |   Frida 16.1.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/
   . . . .
   . . . .   Connected to Mi A2 (id=4920eaee)
Spawning `tech.httptoolkit.pinning_demo`...
Hello! This is the first time you're running this particular snippet, or the snippet's source code
has changed.

Project Name: frida-multiple-unpinning
Author: @akabe1
Slug: akabe1/frida-multiple-unpinning
Fingerprint: ce093d7f85ccd594205906dcc5d44c21ebe7992fb7937c288e046483bcbb114c
URL: https://codeshare.frida.re/@akabe1/frida-multiple-unpinning

Are you sure you'd like to trust this project? [y/N] y
Adding fingerprint ce093d7f85ccd594205906dcc5d44c21ebe7992fb7937c288e046483bcbb114c to the trust
store! You won't be prompted again unless the code changes.
Spawned `tech.httptoolkit.pinning_demo`. Resuming main thread!
```



Frida scripting

For manual approach, we can search the right function, hook, replace the return value same as in the bypassing root detection



02.4

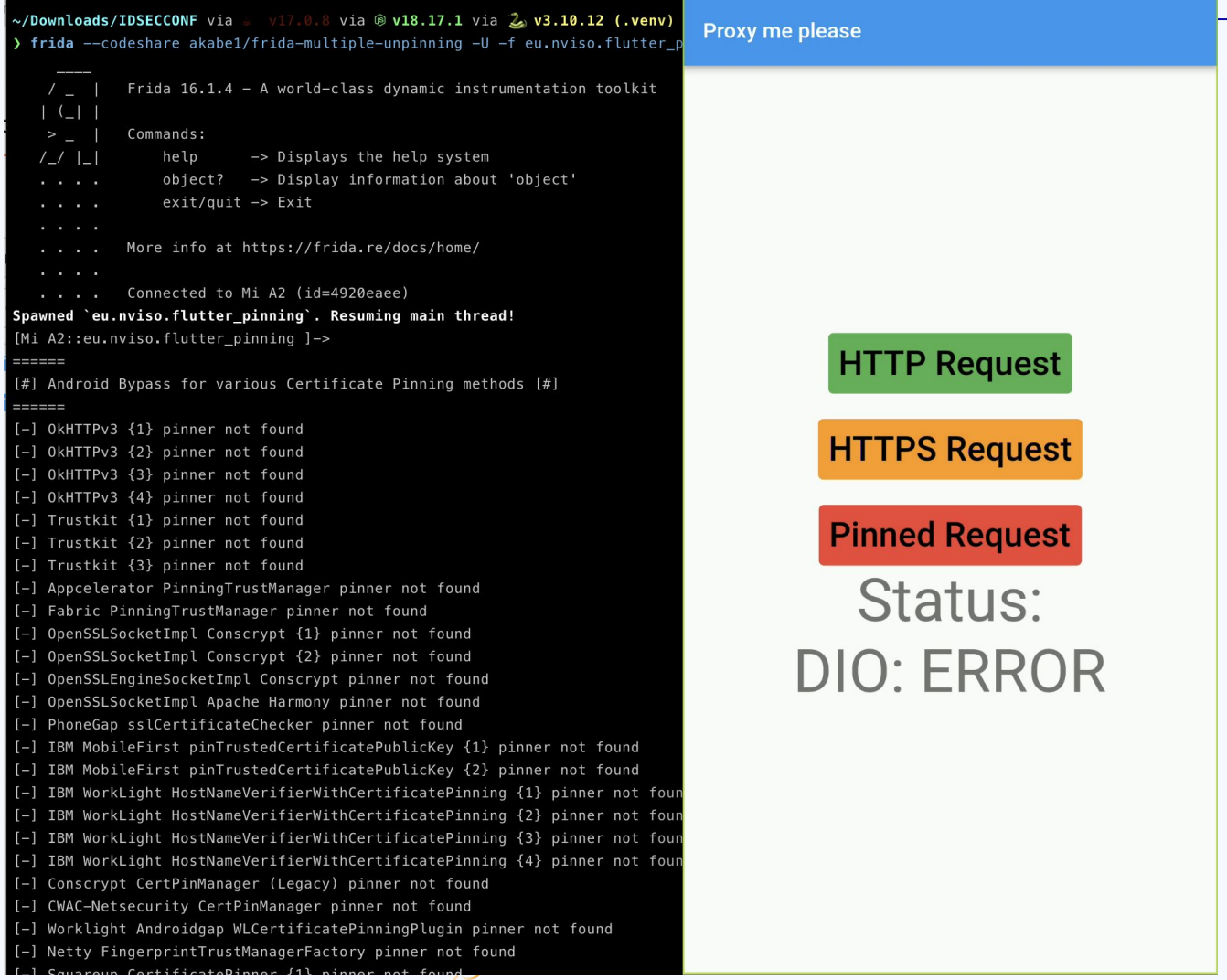
Bypassing Flutter

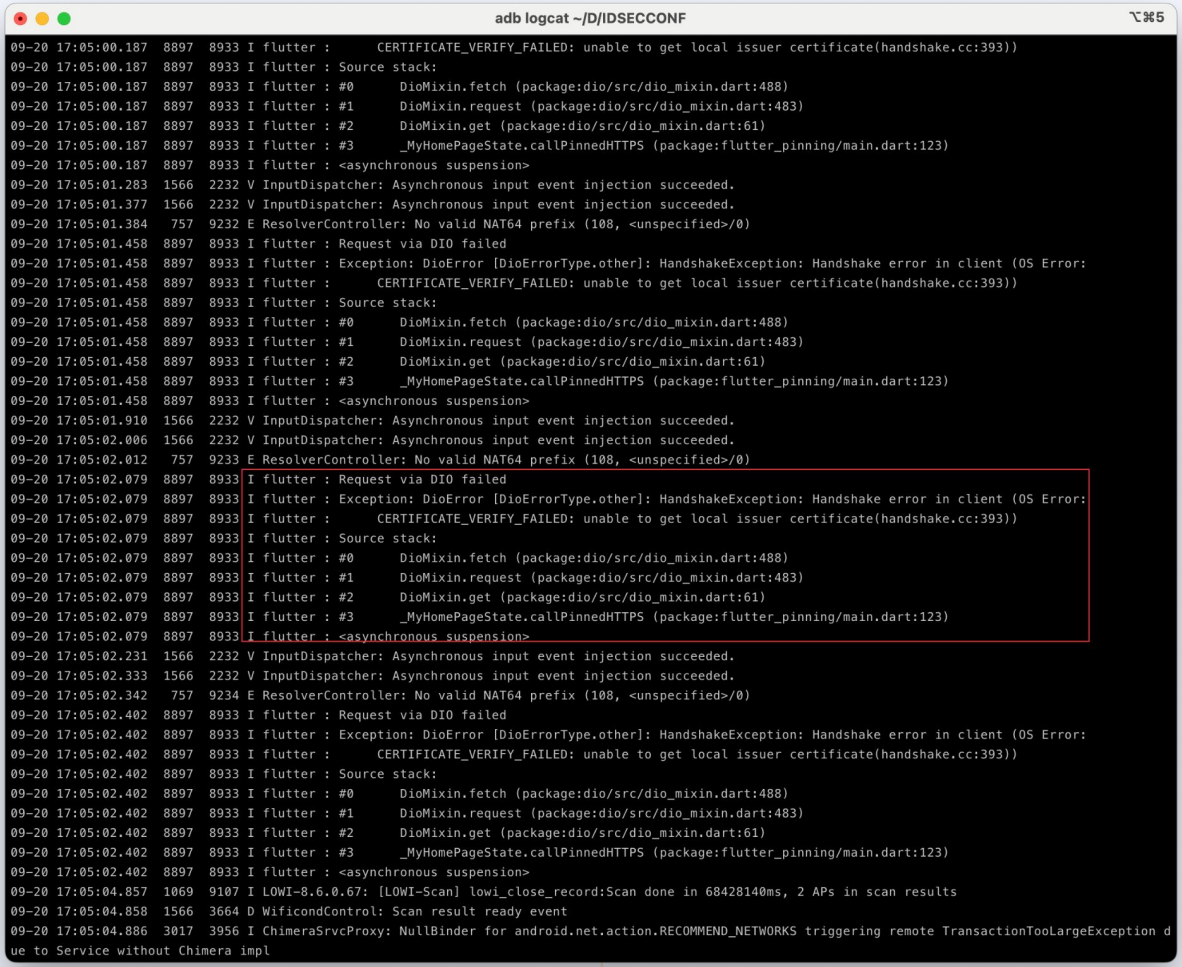
Bypassing Flutter SSL Pinning

Flutter

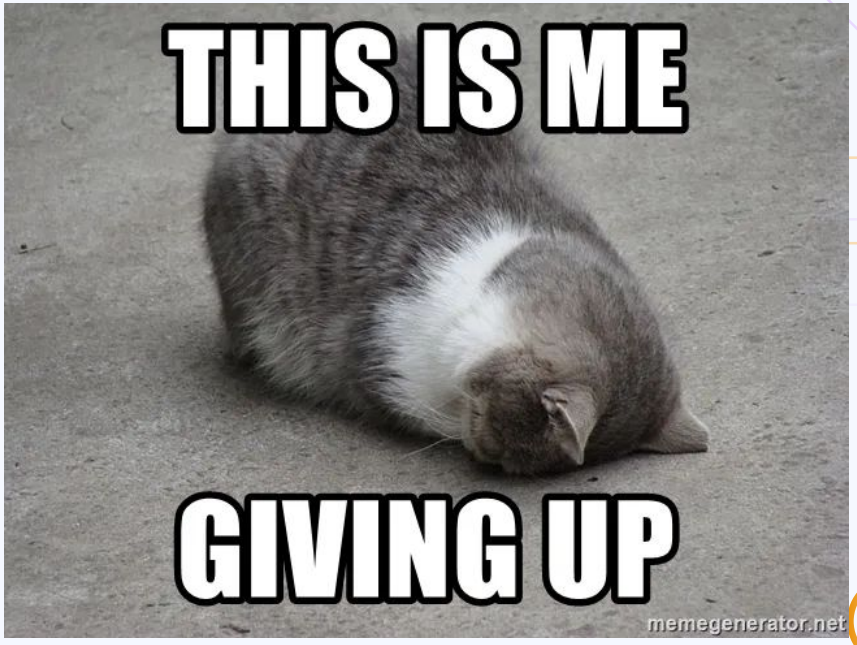
One of the reasons it is difficult to bypass SSL Pinning in Flutter is because Flutter compiles the code into native machine code. This makes common techniques such as method hooking or code injection in SSL Pinning bypass unable to be carried out, even the experiments that we have carried out previously.

Flutter ignores proxy settings on the device so that applications cannot be intercepted. If in the previous case the application would error when passing through the proxy without the SSL pinning bypass process, this does not apply to Flutter because the application will only make a direct connection to the server without passing through the proxy even though it has been set on the device.


~/Downloads/IDSECCONF via v17.0.8 via v18.17.1 via v3.10.12 (.venv)
❯ frida --codeshare akabe1/frida-multiple-unpinning -U -f eu.nviso.flutter_p
Frida 16.1.4 - A world-class dynamic instrumentation toolkit
Commands:
help -> Displays the help system
object? -> Display information about 'object'
exit/quit -> Exit
More info at https://frida.re/docs/home/
Connected to Mi A2 (id=4920eaee)
Spawned `eu.nviso.flutter_pinning`. Resuming main thread!
[Mi A2::eu.nviso.flutter_pinning]->
======
[#] Android Bypass for various Certificate Pinning methods [#]
======
[-] OkHTTPv3 {1} pinner not found
[-] OkHTTPv3 {2} pinner not found
[-] OkHTTPv3 {3} pinner not found
[-] OkHTTPv3 {4} pinner not found
[-] Trustkit {1} pinner not found
[-] Trustkit {2} pinner not found
[-] Trustkit {3} pinner not found
[-] Appcelerator PinningTrustManager pinner not found
[-] Fabric PinningTrustManager pinner not found
[-] OpenSSLSocketImpl Conscrypt {1} pinner not found
[-] OpenSSLSocketImpl Conscrypt {2} pinner not found
[-] OpenSSLEngineSocketImpl Conscrypt pinner not found
[-] OpenSSLSocketImpl Apache Harmony pinner not found
[-] PhoneGap sslCertificateChecker pinner not found
[-] IBM MobileFirst pinTrustedCertificatePublicKey {1} pinner not found
[-] IBM MobileFirst pinTrustedCertificatePublicKey {2} pinner not found
[-] IBM WorkLight HostNameVerifierWithCertificatePinning {1} pinner not foun
[-] IBM WorkLight HostNameVerifierWithCertificatePinning {2} pinner not foun
[-] IBM WorkLight HostNameVerifierWithCertificatePinning {3} pinner not foun
[-] IBM WorkLight HostNameVerifierWithCertificatePinning {4} pinner not foun
[-] Conscrypt CertPinManager (Legacy) pinner not found
[-] CWAC-Netsecurity CertPinManager pinner not found
[-] Worklight Androidgap WLCertificatePinningPlugin pinner not found
[-] Netty FingerprintTrustManagerFactory pinner not found
Proxy me please
HTTP Request
HTTPS Request
Pinned Request
Status:
DIO: ERROR


adb logcat ~/D/IDSECCONF
⌥⌘5
09-20 17:05:00.187 8897 8933 I flutter : CERTIFICATE_VERIFY_FAILED: unable to get local issuer certificate(handshake.cc:393))
09-20 17:05:00.187 8897 8933 I flutter : Source stack:
09-20 17:05:00.187 8897 8933 I flutter : #0 DioMixin.fetch (package:dio/src/dio_mixin.dart:488)
09-20 17:05:00.187 8897 8933 I flutter : #1 DioMixin.request (package:dio/src/dio_mixin.dart:483)
09-20 17:05:00.187 8897 8933 I flutter : #2 DioMixin.get (package:dio/src/dio_mixin.dart:61)
09-20 17:05:00.187 8897 8933 I flutter : #3 _MyHomePageState.callPinnedHTTPS (package:flutter_pinning/main.dart:123)
09-20 17:05:00.187 8897 8933 I flutter : <asynchronous suspension>
09-20 17:05:01.283 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:01.377 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:01.384 757 9232 E ResolverController: No valid NAT64 prefix (108, <unspecified>/0)
09-20 17:05:01.458 8897 8933 I flutter : Request via DIO failed
09-20 17:05:01.458 8897 8933 I flutter : Exception: DioError [DioErrorType.other]: HandshakeException: Handshake error in client (OS Error:
09-20 17:05:01.458 8897 8933 I flutter : CERTIFICATE_VERIFY_FAILED: unable to get local issuer certificate(handshake.cc:393))
09-20 17:05:01.458 8897 8933 I flutter : Source stack:
09-20 17:05:01.458 8897 8933 I flutter : #0 DioMixin.fetch (package:dio/src/dio_mixin.dart:488)
09-20 17:05:01.458 8897 8933 I flutter : #1 DioMixin.request (package:dio/src/dio_mixin.dart:483)
09-20 17:05:01.458 8897 8933 I flutter : #2 DioMixin.get (package:dio/src/dio_mixin.dart:61)
09-20 17:05:01.458 8897 8933 I flutter : #3 _MyHomePageState.callPinnedHTTPS (package:flutter_pinning/main.dart:123)
09-20 17:05:01.458 8897 8933 I flutter : <asynchronous suspension>
09-20 17:05:01.910 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:02.006 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:02.012 757 9233 E ResolverController: No valid NAT64 prefix (108, <unspecified>/0)
09-20 17:05:02.079 8897 8933 I flutter : Request via DIO failed
09-20 17:05:02.079 8897 8933 I flutter : Exception: DioError [DioErrorType.other]: HandshakeException: Handshake error in client (OS Error:
09-20 17:05:02.079 8897 8933 I flutter : CERTIFICATE_VERIFY_FAILED: unable to get local issuer certificate(handshake.cc:393))
09-20 17:05:02.079 8897 8933 I flutter : Source stack:
09-20 17:05:02.079 8897 8933 I flutter : #0 DioMixin.fetch (package:dio/src/dio_mixin.dart:488)
09-20 17:05:02.079 8897 8933 I flutter : #1 DioMixin.request (package:dio/src/dio_mixin.dart:483)
09-20 17:05:02.079 8897 8933 I flutter : #2 DioMixin.get (package:dio/src/dio_mixin.dart:61)
09-20 17:05:02.079 8897 8933 I flutter : #3 _MyHomePageState.callPinnedHTTPS (package:flutter_pinning/main.dart:123)
09-20 17:05:02.079 8897 8933 I flutter : <asynchronous suspension>
09-20 17:05:02.231 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:02.333 1566 2232 V InputDispatcher: Asynchronous input event injection succeeded.
09-20 17:05:02.342 757 9234 E ResolverController: No valid NAT64 prefix (108, <unspecified>/0)
09-20 17:05:02.402 8897 8933 I flutter : Request via DIO failed
09-20 17:05:02.402 8897 8933 I flutter : Exception: DioError [DioErrorType.other]: HandshakeException: Handshake error in client (OS Error:
09-20 17:05:02.402 8897 8933 I flutter : CERTIFICATE_VERIFY_FAILED: unable to get local issuer certificate(handshake.cc:393))
09-20 17:05:02.402 8897 8933 I flutter : Source stack:
09-20 17:05:02.402 8897 8933 I flutter : #0 DioMixin.fetch (package:dio/src/dio_mixin.dart:488)
09-20 17:05:02.402 8897 8933 I flutter : #1 DioMixin.request (package:dio/src/dio_mixin.dart:483)
09-20 17:05:02.402 8897 8933 I flutter : #2 DioMixin.get (package:dio/src/dio_mixin.dart:61)
09-20 17:05:02.402 8897 8933 I flutter : #3 _MyHomePageState.callPinnedHTTPS (package:flutter_pinning/main.dart:123)
09-20 17:05:02.402 8897 8933 I flutter : <asynchronous suspension>
09-20 17:05:04.857 1069 9107 I LOWI-8.6.0.67: [LOWI-Scan] lowi_close_record:Scan done in 68428140ms, 2 APs in scan results
09-20 17:05:04.858 1566 3664 D WificondControl: Scan result ready event
09-20 17:05:04.886 3017 3956 I ChimeraSrvcProxy: NullBinder for android.net.action.RECOMMEND_NETWORKS triggering remote TransactionTooLargeException d
ue to Service without Chimera impl


IF AT FIRST YOU DON'T SUCCEED...
JUST GIVE UP
memegenerator.net


THIS IS ME
GIVING UP
memegenerator.net

02.4.1

reFlutter

Bypassing Flutter SSL Pinning using reFlutter

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. This library has snapshot deserialization process modified to allow you perform dynamic analysis in a convenient way.
Key features:

- socket.cc is patched for traffic monitoring and interception;
- dart.cc is modified to print classes, functions and some fields;
- display absolute code offset for functions
- contains minor changes for successfull compilation;
- if you would like to implement your own patches, there is manual Flutter code change is supported using specially crafted Dockerfile

reFlutter

Install

```
# Linux, Windows, MacOS
pip3 install reflutter==0.7.7
```

Usage

```
impact@f:~$ reflutter main.apk

Please enter your Burp Suite IP: <input_ip>

SnapshotHash: 8ee4ef7a67df9845fba331734198a953
The resulting apk file: ./release.RE.apk
Please sign 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

impact@f:~$ reflutter main.ipa
```

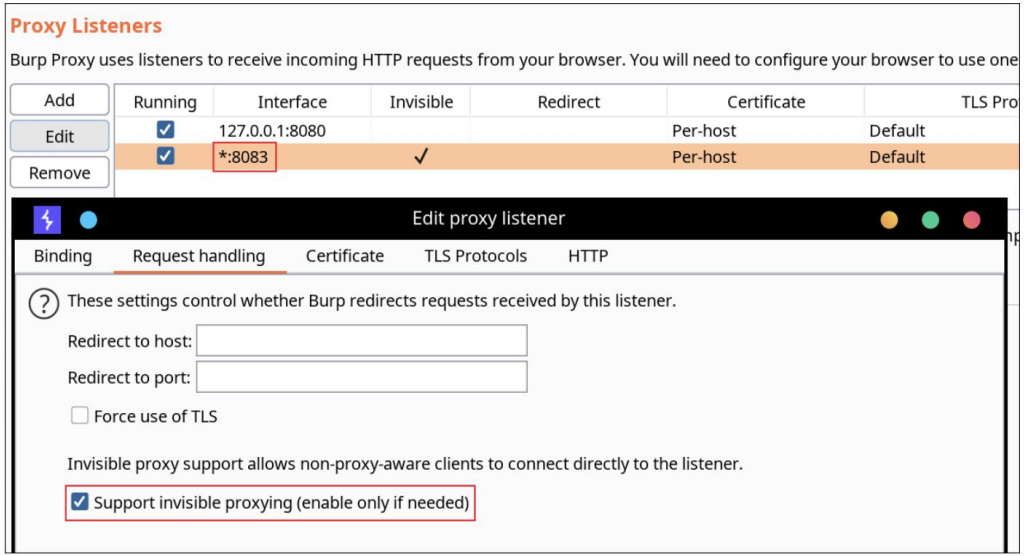
reFlutter

```
❯ java -jar ~/Downloads/IDSECCONF/uber-apk-signer-1.3.0.jar --allowResign -a release.RE.apk
source:
        /Users/petruknisme/Downloads/IDSECCONF
zipalign location: BUILT_IN
        /var/folders/wq/qx9fwmh5713g58xyqjl0g3g80000gn/T/uapksigner-10981596641027246982/mac-
zipalign-33_0_26194567413968502021.tmp
keystore:
        [0] 161a0018
/private/var/folders/wq/qx9fwmh5713g58xyqjl0g3g80000gn/T/temp_4491589597176388583_debug.keystore
(DEBUG_EMBEDDED)

01. release.RE.apk

        SIGN
        file: /Users/petruknisme/Downloads/IDSECCONF/release.RE.apk (16.91 MiB)
        checksum: 337453d0591a7470d94c95a0aa6c463be5540bf3d4aac05f337fc66bc100bb99 (sha256)
        - zipalign success
        - sign success

        VERIFY
        file: /Users/petruknisme/Downloads/IDSECCONF/release.RE-aligned-debugSigned.apk (16.91
MiB)
        checksum: 430852515f4f9ae298f83f97f77583996c22e019eda379f8c27d7a8b015a6c59 (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 03:10:05 WIB 2044

[Wed Sep 20 18:47:01 WIB 2023][v1.3.0]
Successfully processed 1 APKs and 0 errors in 1.59 seconds.
```

Traffic interception

You need to specify the IP of your Burp Suite Proxy Server located in the same network where the device with the flutter application is. Next, you should configure the Proxy in `BurpSuite -> Listener Proxy -> Options tab`

- Add port: `8083`
- Bind to address: `All interfaces`
- Request handling: Support invisible proxying = `True`



HTTP Request

HTTPS Request

Pinned Request

Status:
HTTPS: SUCCESS
(Wed, 20 Sep 2023
08:01:25 GMT)

02.4.1 Frida Pattern Matching

Bypassing Flutter SSL Pinning Ghidra & Frida pattern matching

Concepts

Referring to the article https://blog.nviso.eu/2022/08/18/intercept-flutter-traffic-on-ios-and-android-http-https-dio-pinning/, to bypass SSL pinning on Flutter , we can use the pattern matching method after knowing the pattern of the SSL pinning checking offset address. In simple terms, the steps taken are:

- Find references to the string “x509.cc” and compare them to x509.cc to find session_verify_cert_chain
- Find references to the method you identified in order to identify ssl_verify_peer_cert

```
                    ****************************************************************
                    *                          FUNCTION                            *
                    ****************************************************************
                    undefined FUN_0034b330()
                       assume LRset = 0x0
                       assume TMode = 0x1
     undefined         r0:1           <RETURN>
                    FUN_0034b330+1                                  XREF[0,1]:   005be220(*)
                    FUN_0034b330
0034b330 2d e9 f0 4f     push        { r4, r5, r6, r7, r8, r9, r10, r11, lr  }
0034b334 a3 b0           sub         sp,#0x8c
0034b336 82 46           mov         r10,r0
0034b338 50 20           mov         r0,#0x50
0034b33a 10 70           strb        r0,[r2,#0x0]
0034b33c da f8 98 70     ldr.w       r7,[r10,#0x98]
0034b340 00 2f           cmp         r7,#0x0
0034b342 4c d0           beq         LAB_0034b3de
0034b344 38 68           ldr         r0,[r7,#0x0]
0034b346 00 28           cmp         r0,#0x0
```

https://blog.nviso.eu/2019/08/13/intercepting-traffic-from-android-flutter-applications/

Concepts

```
function hook_ssl_verify_result(address)
{
  Interceptor.attach(address, {
    onEnter: function(args) {
      console.log("Disabling SSL validation")
    },
    onLeave: function(retval)
    {
      console.log("Retval: " + retval)
      retval.replace(0x1);

    }
  });
}
function disablePinning()
{
 var m = Process.findModuleByName("libflutter.so");
 var pattern = "2d e9 f0 4f a3 b0 82 46 50 20 10 70"
 var res = Memory.scan(m.base, m.size, pattern, {
  onMatch: function(address, size){
      console.log('[+] ssl_verify_result found at: ' + address

      // Add 0x01 because it's a THUMB function
      // Otherwise, we would get 'Error: unable to intercept f
      hook_ssl_verify_result(address.add(0x01));

    },
  onError: function(reason){
      console.log('[!] There was an error scanning memory');
    },
    onComplete: function()
    {
      console.log("All done")
    }
  });
}
setTimeout(disablePinning, 1000)
```

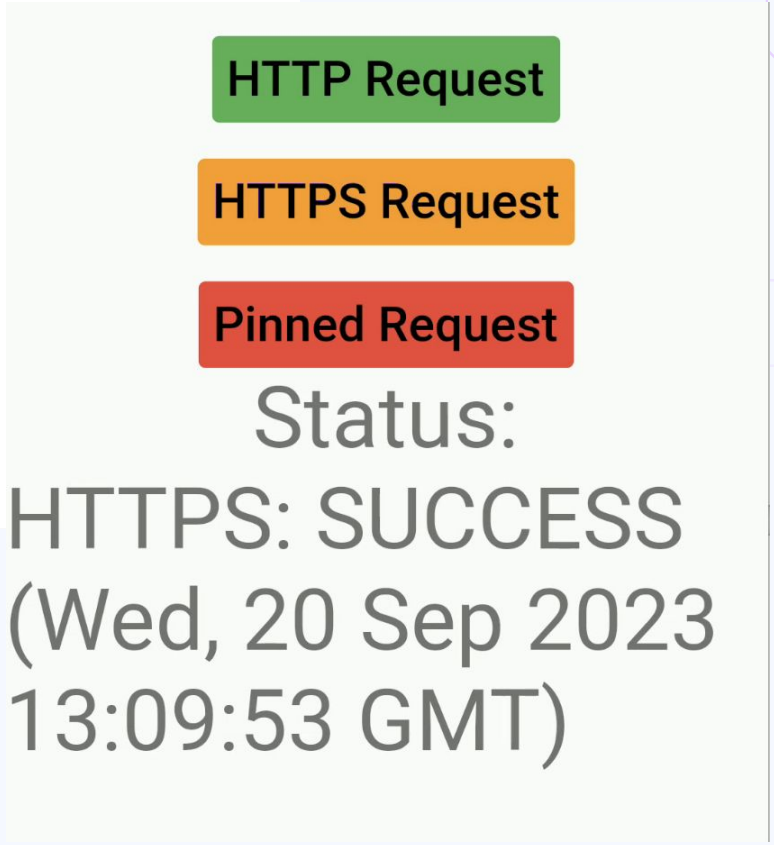
Concepts

Alternatively, we can use Frida's pattern matching engine to search for functions that look very similar to the function from the demo app. The first bytes of a function are typically very stable, as long as the number of local variables and function arguments don't change. Still, different compilers may generate different assembly code (e.g. usage of different registers or optimisations) so we do need to have some wildcards in our pattern.

```
iOS x64: FF 83 01 D1 FA 67 01 A9 F8 5F 02 A9 F6 57 03 A9 F4 4F 04 A9 FD 7B 05 A9 FD 43 01 91 F? 03
00 AA 1? 00 40 F9 ?8 1A 40 F9 15 ?5 4? F9 B5 00 00 B4
Android x64: F? 0F 1C F8 F? 5? 01 A9 F? 5? 02 A9 F? ?? 03 A9 ?? ?? ?? ?? 68 1A 40 F9
Android x86: 2D E9 FE 43 D0 F8 00 80 81 46 D8 F8 18 00 D0 F8 ?? 71
```

https://blog.nviso.eu/2022/08/18/intercept-flutter-traffic-on-ios-and-android-http-https-dio-pinning/

```
❯ frida --codeshare TheDauntless/disable-flutter-tls-v1 -U -f eu.nviso.flutter_pinning
     ____
    / _  |   Frida 16.1.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/
   . . . .
   . . . .   Connected to Mi A2 (id=4920eaee)
Spawning `eu.nviso.flutter_pinning`...
[+] Java environment detected
Spawned `eu.nviso.flutter_pinning`. Resuming main thread!
[Mi A2::eu.nviso.flutter_pinning ]-> [+] libflutter.so loaded
[+] Flutter library found
[!] ssl_verify_peer_cert not found. Trying again...
[+] ssl_verify_peer_cert found at offset: 0x3a251c
```



03

Emulator Detection

Emulator Detection

Emulators are usually used to make it easier for reverse engineers, hackers or pentesters to run applications without needing to have the original device. Because usually, emulators can be easily created and deleted when there are errors or different needs, whereas the original device cannot. It is not uncommon for malware developers to use this check to ensure that their application is able to obtain or control the victim's original data or device. For this case study, we will use https://github.com/reveny/Android-Emulator-Detection/

The app detect emulator with this checks:

- checkHardware();
- checkMounts();
- checkModules();
- checkCPU();
- checkFiles();
- checkCPUArchitecture();

Emulator Detection

```
private fun checkBuildConfig(): Boolean {
    var isEmulator = (Build.MANUFACTURER.contains("Genymotion")
            || Build.MODEL.contains("google_sdk")
            || Build.MODEL.toLowerCase().contains("droid4x")
            || Build.MODEL.contains("Emulator")
            || Build.MODEL.contains("Android SDK built for x86")
            || Build.HARDWARE == "goldfish"
            || Build.HARDWARE == "vbox86"
            || Build.HARDWARE.toLowerCase().contains("nox")
            || Build.FINGERPRINT.startsWith("generic")
            || Build.PRODUCT == "sdk"
            || Build.PRODUCT == "google_sdk"
            || Build.PRODUCT == "sdk_x86"
            || Build.PRODUCT == "vbox86p"
            || Build.PRODUCT.toLowerCase().contains("nox")
            || Build.BOARD.toLowerCase().contains("nox")
            || (Build.BRAND.startsWith("generic") &&
Build.DEVICE.startsWith("generic")))

    return isEmulator
}
```

Concepts

Different from the previous discussion, this time our target is using the Java Native Interface. So, we can't do hooking like we did before. This is because all checking processes occur in libraries that have been compiled into objects. In this example, all the checks are in the libemulatordetector.so file.

The first thing we have to do is find out what JNI functions are available, you can use Frida's help or use nm demangle.

```
❯ nm --demangle --dynamic ~/Downloads/app-debug/lib/arm64-v8a/libemulatordetector.so | grep Java_
00000000000211b0 T Java_com_reveny_emulator_detection_MainActivity_getResult
0000000000021164 T Java_com_reveny_emulator_detection_MainActivity_isDetected
```

Bypass

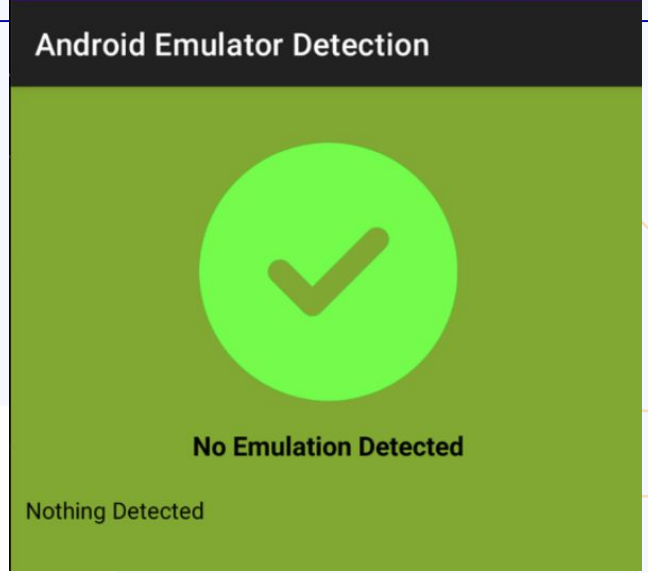
To bypass, the first thing to do is tracing with frida-trace. In this case, I will specifically look for the **dlopen** function because this function is usually used to load shared libraries. It can be seen that what is used is **android_dlopen_ext** to load the shared library file **libemulatordetector.so**. So with this info, we can create code to hook Frida.

```
❯ frida-trace -U -i "*dlopen*" -f com.reveny.emulator.detection
Instrumenting...
__loader_android_dlopen_ext: Loaded handler at
"/Users/petruknisme/Downloads/IDSECCONF/__handlers__/linker64/__loader_android_dlopen_ext.js"
__loader_dlopen: Loaded handler at
"/Users/petruknisme/Downloads/IDSECCONF/__handlers__/linker64/__loader_dlopen.js"
dlopen: Loaded handler at "/Users/petruknisme/Downloads/IDSECCONF/__handlers__/libdl.so/dlopen.js"
android_dlopen_ext: Loaded handler at
"/Users/petruknisme/Downloads/IDSECCONF/__handlers__/libdl.so/android_dlopen_ext.js"
Started tracing 4 functions. Press Ctrl+C to stop.
```

When we run Frida with this code, we can see that libemulatordetector.so is loaded from the android_dlopen_ext function

```
Interceptor.attach(Module.findExportByName(null, 'android_dlopen_ext'), {
    onEnter: function(args) {
        this.path = Memory.readUtf8String(args[0]);
    },
    onLeave: function(retval) {
        console.log(this.path);
    }
});
```

```
❯ frida -U -l example-hook.js -f com.reveny.emulator.detection
Spawned `com.reveny.emulator.detection`. Resuming main thread!
[Phone::com.reveny.emulator.detection ]-> /vendor/lib64/egl/libEGL_emulation.so
/vendor/lib64/egl/libGLESv1_CM_emulation.so
/vendor/lib64/egl/libGLESv2_emulation.so
/data/app/~~ZCD4i-zCV0UIDcAgoKHdJg==/com.reveny.emulator.detection-Zhnwr4_vmjSed2Dn-
yJHGQ==/base.apk!/lib/arm64-v8a/libemulatordetector.so
/vendor/lib64/hw/android.hardware.graphics.mapper@2.0-impl.so
/vendor/lib64/hw/gralloc.ranchu.so
```




```
Interceptor.attach(Module.findExportByName(null, 'android_dlopen_ext'), {
    onEnter: function(args) {
        this.path = Memory.readUtf8String(args[0]);
    },
    onLeave: function(retval) {
        if (!retval.isNull() && this.path.indexOf('libemulatordetector.so') !== -1 ) {
            hookReplaceIsDetected();
        }
    }
});

function hookReplaceIsDetected() {
    Interceptor.attach(Module.findExportByName("libemulatordetector.so",
"Java_com_reveny_emulator_detection_MainActivity_isDetected"), {
        onLeave: function(retval) {
            retval.replace(0);
            console.log("Replacing detection function success");
        }
    });
}
```

rootAVD

newbit @ xda-developers

A Script to...

- root your Android Studio Virtual Device (AVD), with Magisk (Stable, Canary or Alpha)
- patch its fstab
- download and install the USB HOST Permissions Module for Magisk
- install custom build Kernel and its Modules
- download and install AOSP prebuilt Kernel and its Modules

...within seconds.

Install Magisk

Download rootAVD via

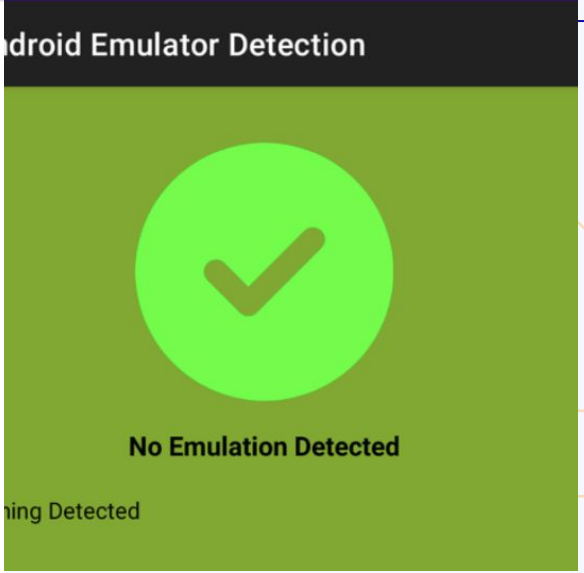
- Click
- `git clone https://gitlab.com/newbit/rootAVD.git`

Preconditions

- the AVD is running
- a working Internet connection for the Menu
- a command prompt / terminal is opened
- `adb shell` will connect to the running AVD

Use Case Examples

```
create_fake_boot_img() {

        if $DEBUG; then
                generate_build_prop
        fi

        echo "[*] Creating a fake Boot.img"
        FBHI=$BASEDIR/fakebootheader.img
        FBI=$SDCARD/fakeboot.img
        RAMDISK_SZ="$(printf '%08x' $(stat -c%s $CPIO))"
        PAGESIZE=2048
        PAGESIZE_HEX="$(printf '%08x' $PAGESIZE)"

        echo "[-] removing old $FBI"
        rm -f $FBI $RDF

        printf "\x41\x4E\x44\x52\x4F\x49\x44\x21" > $FBHI  # ANDROID!
        printf "\x00\x00\x00\x00\x00\x00\x00\x00" >> $FBHI # HEADER_VER KERNEL_SZ
        writeLittleEndian $RAMDISK_SZ >> $FBHI # RAMDISK_SZ

        printf "\x00\x00\x00\x00" >> $FBHI # SECOND_SZ
        printf "\x00\x00\x00\x00\x00\x00\x00\x00" >> $FBHI # EXTRA_SZ
        printf "\x00\x00\x00\x00" >> $FBHI
        writeLittleEndian $PAGESIZE_HEX >> $FBHI # PAGESIZE_HEX

        echo "[!] Only a minimal header is required for Magisk to repack the ramdisk"
        #mv $RDF $CPIO

        echo "[*] repacking ramdisk.img into $FBI"
        $BASEDIR/magiskboot repack $FBHI $FBI > /dev/null 2>&1

        test -f "$FBI"
        RESULT="$?"
        if [[ "$RESULT" != "0" ]]; then
                echo "[*] $FBI could not be created"
                echo "[-] Magisk expects a more complete boot.img header as source"

                # fill 00 (to Pagesize 2048)
                truncate -s $PAGESIZE $FBHI
```



03

Frida Detection

Concepts

In some cases, developers create protection so that the application checks the existence of Frida on the device. This is to prevent pentesters/hackers from being able to run Binary Instruments using Frida as in the previous discussion.

There are several mechanisms used to detect the presence of frida, including:

- Detection of named pipes used by Frida
- Detect frida specific thread names
- Detect the port used by Frida
- Etc

For this discussion, I used a customized application to detect frida existence

Concepts

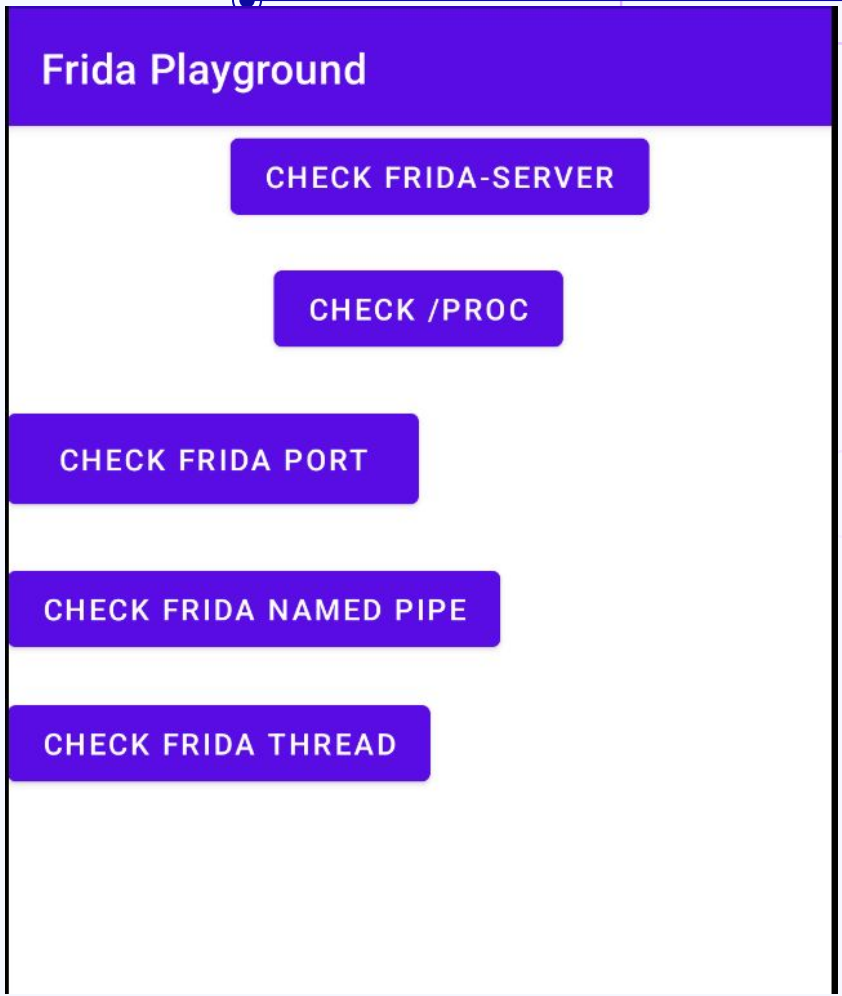
In some cases, developers create protection so that the application checks the existence of Frida on the device. This is to prevent pentesters/hackers from being able to run Binary Instruments using Frida as in the previous discussion.

There are several mechanisms used to detect the presence of frida, including:

- Detection of named pipes used by Frida
- Detect frida specific thread names
- Detect the port used by Frida
- Etc

For this discussion, I used a customized application to detect frida existence

Sample app


Frida Playground
CHECK FRIDA-SERVER
CHECK /PROC
CHECK FRIDA PORT
CHECK FRIDA NAMED PIPE
CHECK FRIDA THREAD

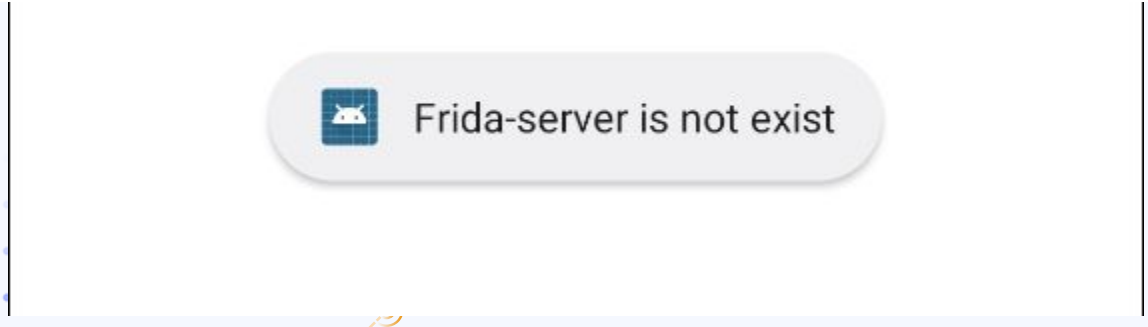
Frida-server detection

The mechanism used by the application to detect frida-server is to check whether there are files in /data/local/tmp/frida-server.

```
extern "C"
JNIEXPORT jboolean JNICALL
MainActivity.isFridaRunning(JNIEnv *env, jobject MainActivity thiz) {
    int result;
    struct stat sb;
    if (stat( path: "/data/local/tmp/frida-server", buf: &sb) == 0) {
        return true;
    }

    return false;
}
```

Frida-server detection

```
const pathString = Memory.allocUtf8String("/data/local/tmp/foo-server");

Interceptor.attach(Module.getExportByName(null, 'stat'), {
    onEnter(args) {
        if(args[0].readUtf8String().includes('frida-server')) {
            args[0] = pathString;
        }
    }
});
```



/proc detection

The mechanism used is to check /proc/self/maps to detect the presence of frida-agent

```
extern "C"
JNIEXPORT jboolean JNICALL
MainActivity.isFridaProc(JNIEnv *env, jobject MainActivity thiz) {
    // TODO: implement isFridaProc()

    FILE *fp;
    char line[512] = {[0]: 0};
    fp = fopen( path: "/proc/self/maps", mode: "r");
    if (fp) {
        while (fgets( dest: line, size: 512, stream: fp)) {
            if (strstr( h: line, n: "frida-agent")) {
                return true;
            }
        }
    }
    return false;
}
```

/proc detection

```
Interceptor.attach(Module.getExportByName(null, 'fopen'), {
    onEnter(args) {
        this.strstr = null;
        if (args[0].readUtf8String() == "/proc/self/maps")
        {
            Interceptor.attach(Module.getExportByName(null, 'strstr'), {
                onEnter(args) {
                    this.arg_1 = args[0].readUtf8String();
                },
                onLeave(retval) {
                    if (this.arg_1.includes('frida'))
                    {
                        retval.replace(0);
                    }
                }
            });
        }
    },
});
```

Default frida port detection

Another mechanism is to check the default Frida port, if the default port can be accessed then that indicates Frida is running

```
Java_com_learn_frida_MainActivity_isFridaPortRunning(JNIEnv *env, jobject MainActivity thiz) {
    // TODO: implement isFridaProc()
    int sock = 0;
    struct sockaddr_in serv_addr;

    if ((sock = socket( af: AF_INET, type: SOCK_STREAM, protocol: 0)) < 0)
    {
        return false;
    }
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);
    if(inet_pton( af: AF_INET, src: "127.0.0.1", dst: &serv_addr.sin_addr)<=0)
    {
        return false;
    }

    if (connect( fd: sock, addr: (struct sockaddr *)&serv_addr, addr_length: sizeof(serv_addr)) < 0)
    {
        return false;
    }
    // Fail (Frida is running)
    return true;
}
```

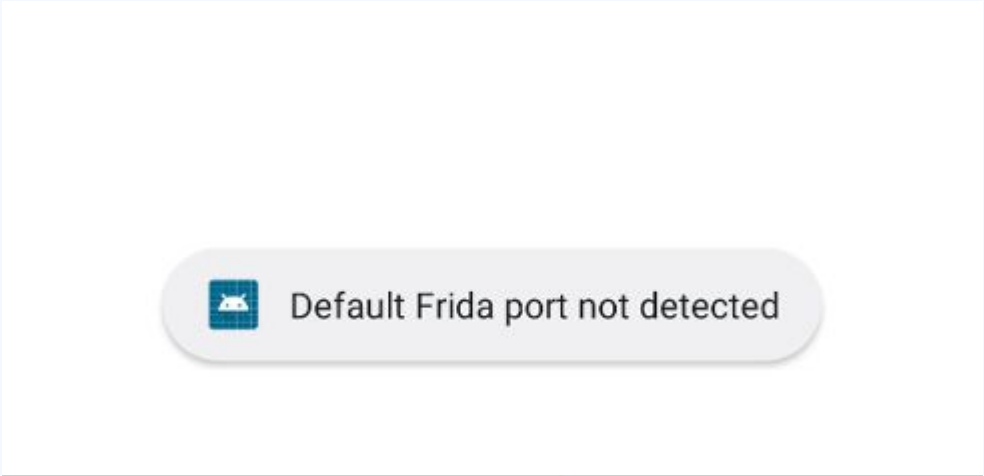
Default frida port detection

To bypass this protection, we can change the default port used when running frida-server with the following command

```
> ./frida-server -l 0.0.0.0:9999

> adb forward tcp:9999 tcp:9999
9999

> frida-ps -H 127.0.0.1:9999
  PID  Name
-----  -----------------------------------------------
12101   Files
12271   Gallery
  912   SIM Toolkit
18034   Settings
 2732      abb
  419      adbd
  971      android.ext.services
  199      android.hardware.atrace@1.0-service
  345      android.hardware.audio.service
  346      android.hardware.authsecret@1.0-service
```

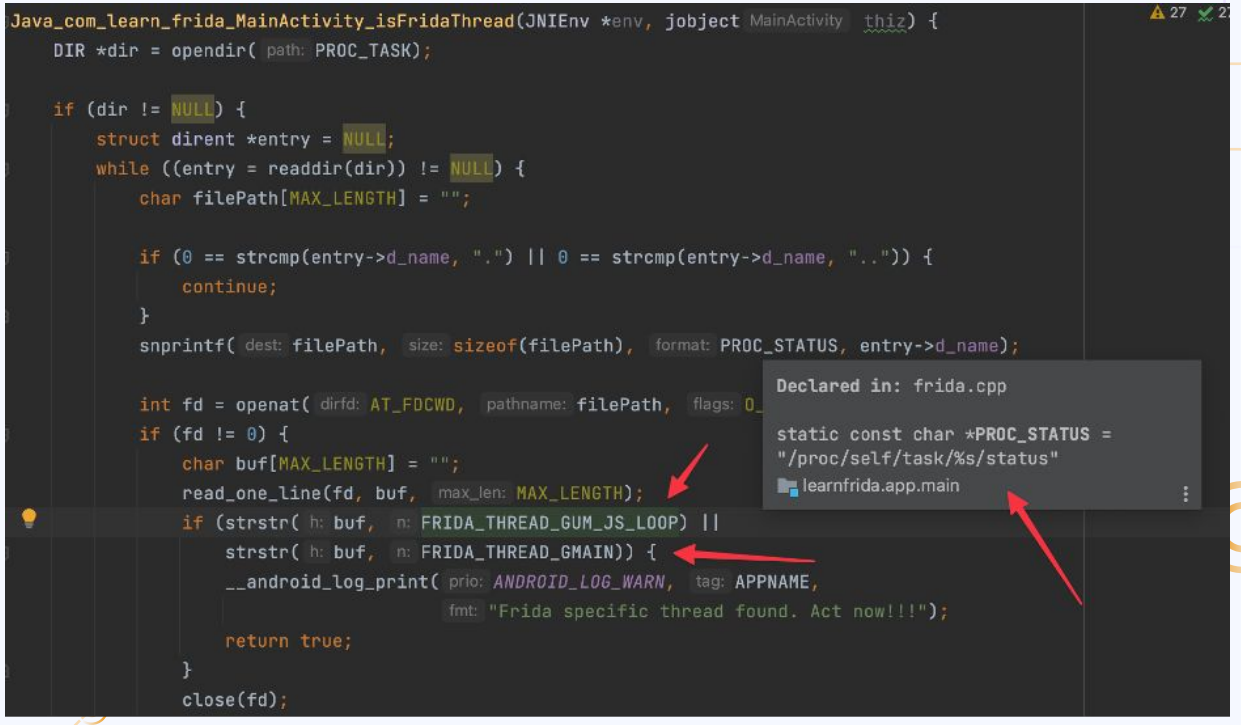
Default frida port detection

```
❯ frida -H 127.0.0.1:9999 -f com.learn.frida -l bypass-maps.js
    . . . .   Connected to 127.0.0.1:9999 (id=socket@127.0.0.1:9999)
Spawned `com.learn.frida`. Resuming main thread!
[Remote::com.learn.frida ]->
```



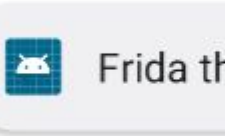
Frida Thread Detection

The application detects the Frida thread by checking /proc/self/task/<PID>/status and comparing its contents with FRIDA_THREAD_GUM_JS_LOOP and FRIDA_THREAD_GMAIN.



Frida Thread Detection

To bypass this protection, we can easily manipulate the **strstr** function every time we find the specified words




```
Interceptor.attach(Module.findExportByName(null, "strstr"),{
    onEnter: function(args){
        this.frida = false;
        var str1 = args[0].readCString();
        var str2 = args[1].readCString();
        if(
          str1.indexOf("gum-js-loop")!==-1 || str2.indexOf("gum-js-loop")!==-1 ||
          str1.indexOf("gmain")!==-1 || str2.indexOf("gmain")!==-1
          ){
            this.frida = true;
        }
    },
    onLeave: function(retval){
        if (this.frida) {
            retval.replace(ptr("0x0"));
        }
    }
});
```

Thanks!

Do you have any questions?

me@petruknisme
t.me/@petruknisme
linkedin.com/in/aancw