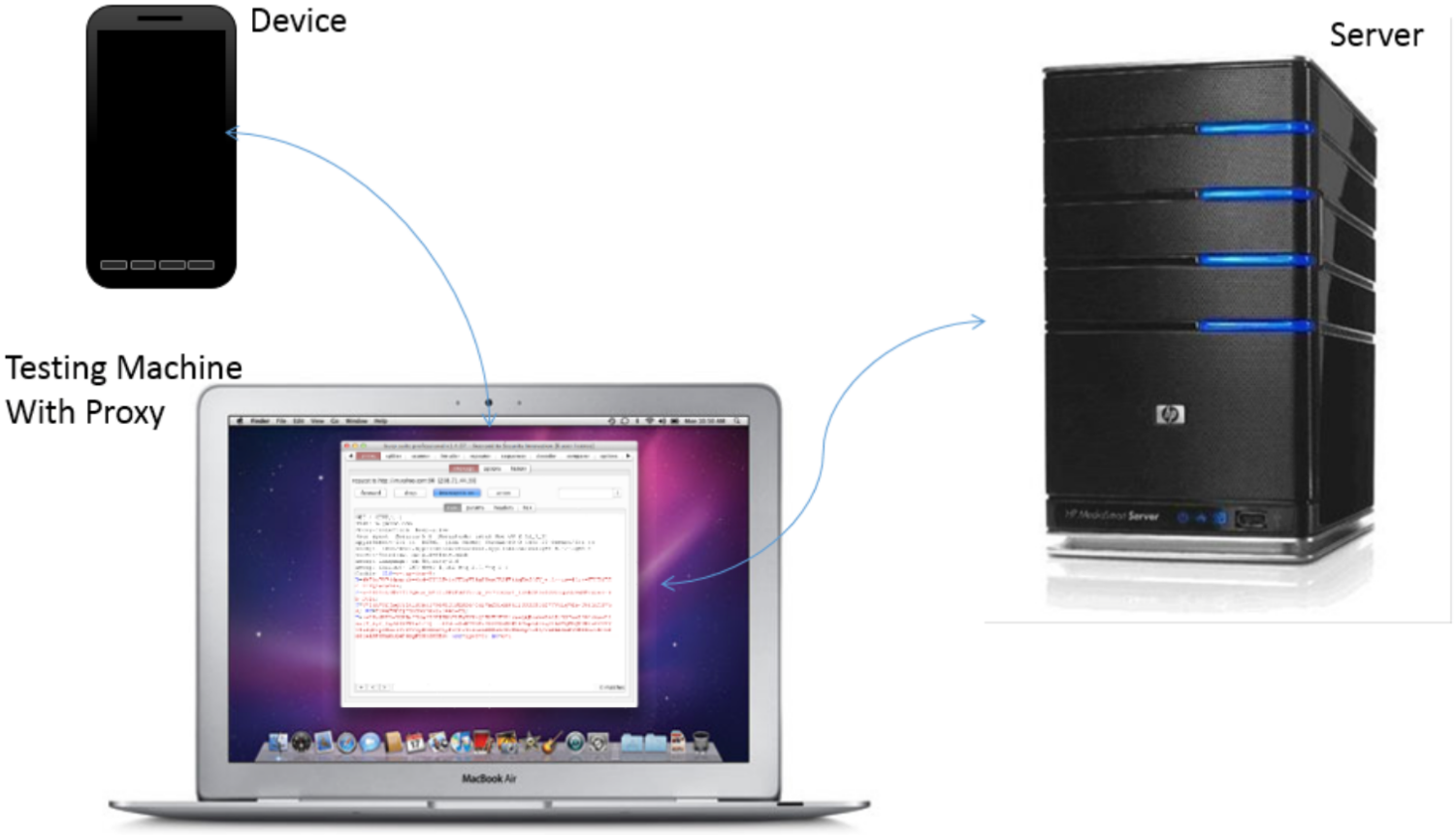
# Module 3: Android Vulnerabilities

- Exploiting Local Storage
- Exploiting Android Components
- Exploiting Weak Cryptography
- Exploiting Side Channel Data Leakage
- Root Detection and Bypass
- Exploiting Network Communication and Cert Pinning
- Exploiting Firebase Databases
- Attacking Google Play Billing

# Android Security Basics
# Proxying Android Traffic



# Steps

- Start Burp inside Linux VM
- Launch Genymotion Android Device
- Setup Genymotion Android to use your Burp as intermediate proxy
- Add Burp Certificate to the Android Device Certificate Store

# Targets

/home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetwork.apk

# Certificate Tip

- Certificates imported to the ***USER*** store are added to:
  - /data/misc/user/0/cacerts-added/
- Certificates imported to the ***SYSTEM*** store are added to:
  - /system/etc/security/cacerts/
- Certificates in the ***SYSTEM*** store are automatically added to the allowlist by the Android OS and **DO NOT NEED** AndroidManifest.xml to be modified

# Automatically Add Burp Certificate to User and System Store

- Use the script from:
  - ***/home/mobile/Desktop/insertCert.sh***
  - ***./insertCert.sh <<BurpIP> <<BurpPort>>***

./insertCert.sh 127.0.0.1 9999

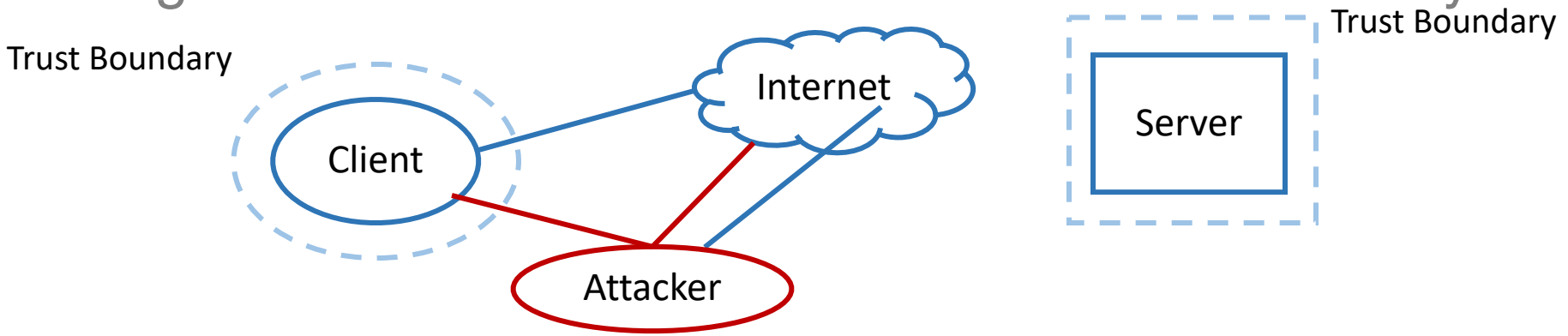
- One Liner (bash):
  - `adb root && adb remount && wget http://127.0.0.1:8080/cert -O burp_wget.crt && openssl x509 -inform der -in burp_wget.crt -out burp_wget.pem && adb shell mkdir -p /data/misc/user/0/cacerts-added/ &&openssl x509 -inform PEM -subject_hash_old -in burp_wget.pem | head -1 | xargs -I{} mv burp_wget.crt {}.0 && adb push *.0 /data/misc/user/0/cacerts-added/ && adb shell ls -1t /data/misc/user/0/cacerts-added/* | head -1 | xargs -I{} adb shell cp {} /system/etc/security/cacerts/ && adb shell chmod 644 /data/misc/user/0/cacerts-added/* && adb shell chown system:system /data/misc/user/0/cacerts-added/* && adb shell chmod 644 /system/etc/security/cacerts/*`

# TASK

- Launch Burp Suite and configure it to intercept traffic
- Configure Genymotion to proxy traffic through your Ubuntu VM
  - **NOTE: BURP certificate is already installed on the Genymotion instance. Do not install the BURP certificate again**
- Make sure you can intercept the HTTP request from ***BuggyNetwork*** application
  - Binary is up on /home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetwork.apk and already installed on Genymotion
- Intercept HTTPS traffic
- Intercept Requests for all of the methods

# Android Security Basics
# Proxying Android Traffic
# Setting up the Test Environment

- Parameter tampering is an attack where the data in messages exchanged between the client and server are modified by the attacker




- The attacker can view and tamper with requests on the client, or at any location between the client and the Trust Boundary
- The tampering can occur in either direction
  - Even with a secure connection

# Bypassing SSL Pinning

# Bypassing SSL Pinning

- SSL Pinning allows developers to specify what certificate is to be considered valid by the application
- Pinning prevents MITM attacks due to mismatch between the user supplied certificate and the packaged certificate

- Techniques:
  - SMALI Manipulation
  - Dynamic Patching
  - Keystore Hijacking
  - Android-SSL-TrustKiller
  - JustTrustMe - Xposed framework
  - Code Manipulation

# Targets

- /home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetwork.apk

- /home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetworkObfuscated.apk

# Network Traffic Analysis - OkHTTP

Target - BuggyNetworkUnobfuscated.apk

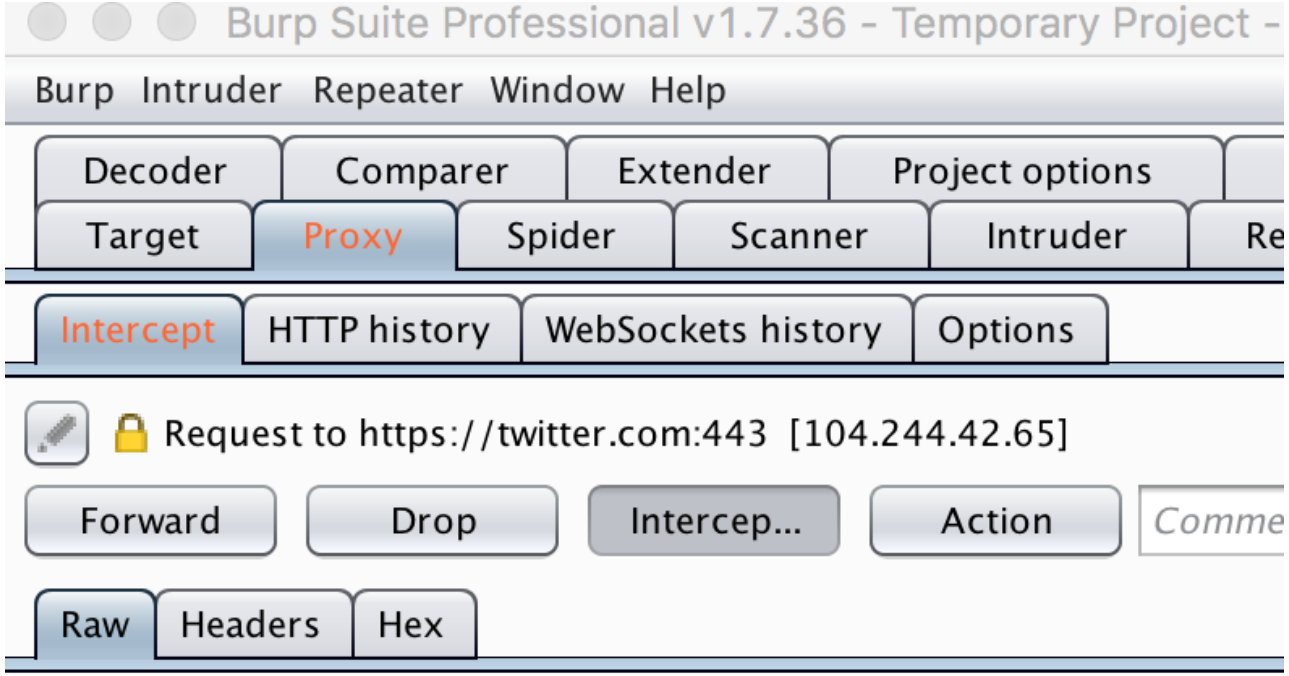
# HttpURLConnection

```java
class HttpURLConnectionGetRequest extends AsyncTask<String, Void, String> {
    @Override
    protected String doInBackground(String... parameters) {
        try {
            URL siteURL = new URL(parameters[0]);
            System.out.println("YES HttpURLConnectionGetRequest ");
            HttpURLConnection httpConn = (HttpURLConnection) siteURL.openConnection();
            httpConn.setRequestMethod("GET");
            httpConn.connect();

            InputStream inputStream = httpConn.getInputStream();
            InputStreamReader inputStreamReader = new InputStreamReader(inputStream);
            BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
            String line = bufferedReader.readLine();
            System.out.println(line);
        } catch (Exception e) {
            e.printStackTrace();
            return "ERROR";
        }
        return "SUCCESS";
    }

    @Override
    protected void onPostExecute(String result){
        super.onPostExecute(result);

        System.out.println("Request Status: " + result);
    }
}
```

# OkHttp

```
class OkHttpClientGetRequest extends AsyncTask<String, Void, String> {
    @Override
    protected String doInBackground(String... parameters) {
        try {
            URL siteURL = new URL(parameters[0]);
            System.out.println("YES OkHttpClientGetRequest ");

            OkHttpClient httpConn = new OkHttpClient();
            Request httpRequest = new Request.Builder().url(siteURL).build();
            Response httpResponse = httpConn.newCall(httpRequest).execute();

        } catch (Exception e) {
            e.printStackTrace();
            return "ERROR";
        }
        return "SUCCESS";
    }
```

# Network Traffic Analysis - OkHTTP




# Network Traffic Analysis

CERTIFICATE PINNING BYPASS - OkHttpClient

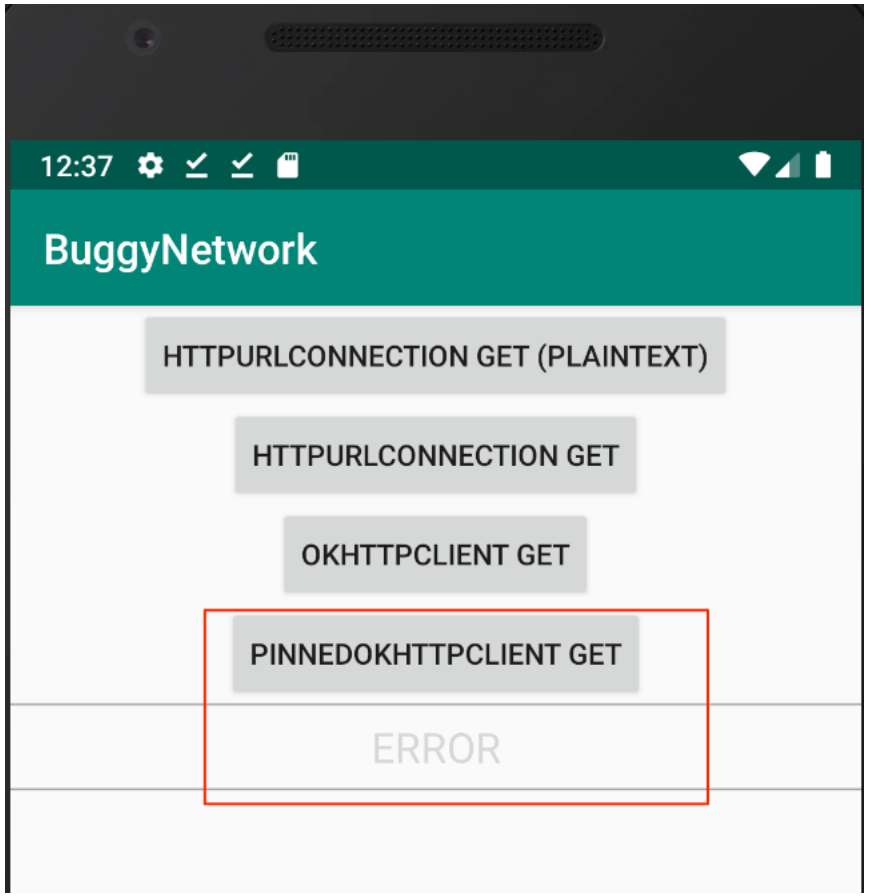
TARGET - /home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetwork.apk

Method – Patching SMALI files

# CERTIFICATE PINNING BYPASS – Patching SMALI Files

- Basic Method worked for other non-certificate-pinned options

- Fails when Certificate Pinning is used



# Patching SMALI Files

- STEP 1 – Understand your Target
- STEP 2 – Patch the necessary files
- Step 3 – Recompile into an APK file
- Step 4  - Re-sign the APK file
- Step 5 – Profit??

# Patching SMALI Files
# STEP 1 – Understand your Target

- Decompile the application and view the source files

com.dns.buggynetwork.MainActivity

```
        });
110     ((Button) findViewById(R.id.button_pinned_ok_http_get)).setOnClickListener(new OnClickListener() {
 95         public void onClick(View v) {
 96             OkHttpClientPinnedGetRequest requestTypeThree = new OkHttpClientPinnedGetRequest();
                try {
100                 MainActivity.this.receivedData = ((String) requestTypeThree.execute(new String[]{MainActivity.th
101                 PrintStream printStream = System.out;
101                 StringBuilder stringBuilder = new StringBuilder();
101                 stringBuilder.append("receivedData = ");
101                 stringBuilder.append(MainActivity.this.receivedData);
101                 printStream.println(stringBuilder.toString());
102                 MainActivity.this.logsStuff.setText(MainActivity.this.receivedData);
```

com.dns.buggynetwork.OkHttpClientPinnedGetRequest

```
class OkHttpClientPinnedGetRequest extends AsyncTask<String, Void, String> {
2       OkHttpClientPinnedGetRequest() {
        }

6       protected String doInBackground(String... parameters) {
            try {
7               URL siteURL = new URL(parameters[0]);
9               String hostname = parameters[1];
0               String pinnedCert1 = parameters[2];
1               String pinnedCert2 = parameters[3];
2               System.out.println("YES OkHttpClientPinnedGetRequest ");
5               new Builder().certificatePinner(new CertificatePinner.Builder().add(hostname, pinnedCert1).add(hos
1               return "SUCCESS";
            } catch (Exception e) {
8               e.printStackTrace();
9               return "ERROR";
            }
        }
```

- Open source libraries FTW !

CertificatePinner.java

```
159         */
160        public void check(String hostname, List<Certificate> peerCertificates)
161            throws SSLPeerUnverifiedException {
162          List<Pin> pins = findMatchingPins(hostname);
163          if (pins.isEmpty()) return;
164
165          if (certificateChainCleaner != null) {
166            peerCertificates = certificateChainCleaner.clean(peerCertificates, hostname);
167          }
168
169          for (int c = 0, certsSize = peerCertificates.size(); c < certsSize; c++) {
170            X509Certificate x509Certificate = (X509Certificate) peerCertificates.get(c);
171
172            // Lazily compute the hashes for each certificate.
173            ByteString sha1 = null;
174            ByteString sha256 = null;
175
176            for (int p = 0, pinsSize = pins.size(); p < pinsSize; p++) {
177              Pin pin = pins.get(p);
178              if (pin.hashAlgorithm.equals("sha256/")) {
179                if (sha256 == null) sha256 = sha256(x509Certificate);
180                if (pin.hash.equals(sha256)) return; // Success!
181              } else if (pin.hashAlgorithm.equals("sha1/")) {
182                if (sha1 == null) sha1 = sha1(x509Certificate);
183                if (pin.hash.equals(sha1)) return; // Success!
184              } else {
185                throw new AssertionError( detailMessage: "unsupported hashAlgorithm: " + pin.hashAlgorithm);
186              }
187            }
188          }
```

```
# virtual methods
.method public check(Ljava/lang/String;Ljava/util/List;)V
```

```
Pin pin = pins.get(p);
if (pin.hashAlgorithm.equals("sha256/")) {
  if (sha256 == null) sha256 = sha256(x509Certificate);
  if (pin.hash.equals(sha256)) return; // Success!
} else if (pin.hashAlgorithm.equals("sha1/")) {
  if (sha1 == null) sha1 = sha1(x509Certificate);
  if (pin.hash.equals(sha1)) return; // Success!
```

okhttp-sources.jar!/okhttp3/CertificatePinner.java

# smali/okhttp3/CertificatePinner.smali

```
.line 178
.local v8, "pin":Lokhttp3/CertificatePinner$Pin;
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hashAlgorithm:Ljava/lang/String;

const-string v10, "sha256/"

invoke-virtual {v9, v10}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result v9

if-eqz v9, :cond_3

.line 179
if-nez v5, :cond_2

invoke-static {v3}, Lokhttp3/CertificatePinner;->sha256(Ljava/security/cert/X509Certificate;)Lokio/ByteString;

move-result-object v5

.line 180
:cond_2
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hash:Lokio/ByteString;

invoke-virtual {v9, v5}, Lokio/ByteString;->equals(Ljava/lang/Object;)Z

move-result v9

if-eqz v9, :cond_5

return-void

.line 181
:cond_3
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hashAlgorithm:Ljava/lang/String;

const-string v10, "sha1/"

invoke-virtual {v9, v10}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
```

```
.line 178
.local v8, "pin":Lokhttp3/CertificatePinner$Pin;
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hashAlgorithm:Ljava/lang/String;

const-string v10, "sha256/"

invoke-virtual {v9, v10}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result v9

if-eqz v9, :cond_3

.line 179
if-nez v5, :cond_2

invoke-static {v3}, Lokhttp3/CertificatePinner;->sha256(Ljava/security/cert/X509Certificate;)Lokio/ByteString;

move-result-object v5

.line 180
:cond_2
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hash:Lokio/ByteString;

invoke-virtual {v9, v5}, Lokio/ByteString;->equals(Ljava/lang/Object;)Z

move-result v9

if-eqz v9, :cond_5

return-void

.line 181
:cond_3
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hashAlgorithm:Ljava/lang/String;

const-string v10, "sha1/"

invoke-virtual {v9, v10}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
```

# Patching SMALI Files
# STEP 2 – Patch the necessary files

```
.local v8, "pin":Lokhttp3/CertificatePinner$Pin;
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hashAlgorithm:Ljava/lang/String;

const-string v10, "sha256/"

invoke-virtual {v9, v10}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result v9

if-eqz v9, :cond_3

.line 179
if-nez v5, :cond_2

invoke-static {v3}, Lokhttp3/CertificatePinner;->sha256(Ljava/security/cert/X509Certificate;)Lokio/ByteString;

move-result-object v5

.line 180
:cond_2
iget-object v9, v8, Lokhttp3/CertificatePinner$Pin;->hash:Lokio/ByteString;

invoke-virtual {v9, v5}, Lokio/ByteString;->equals(Ljava/lang/Object;)Z

move-result v9

if-nez v9, :cond_5

return-void
```

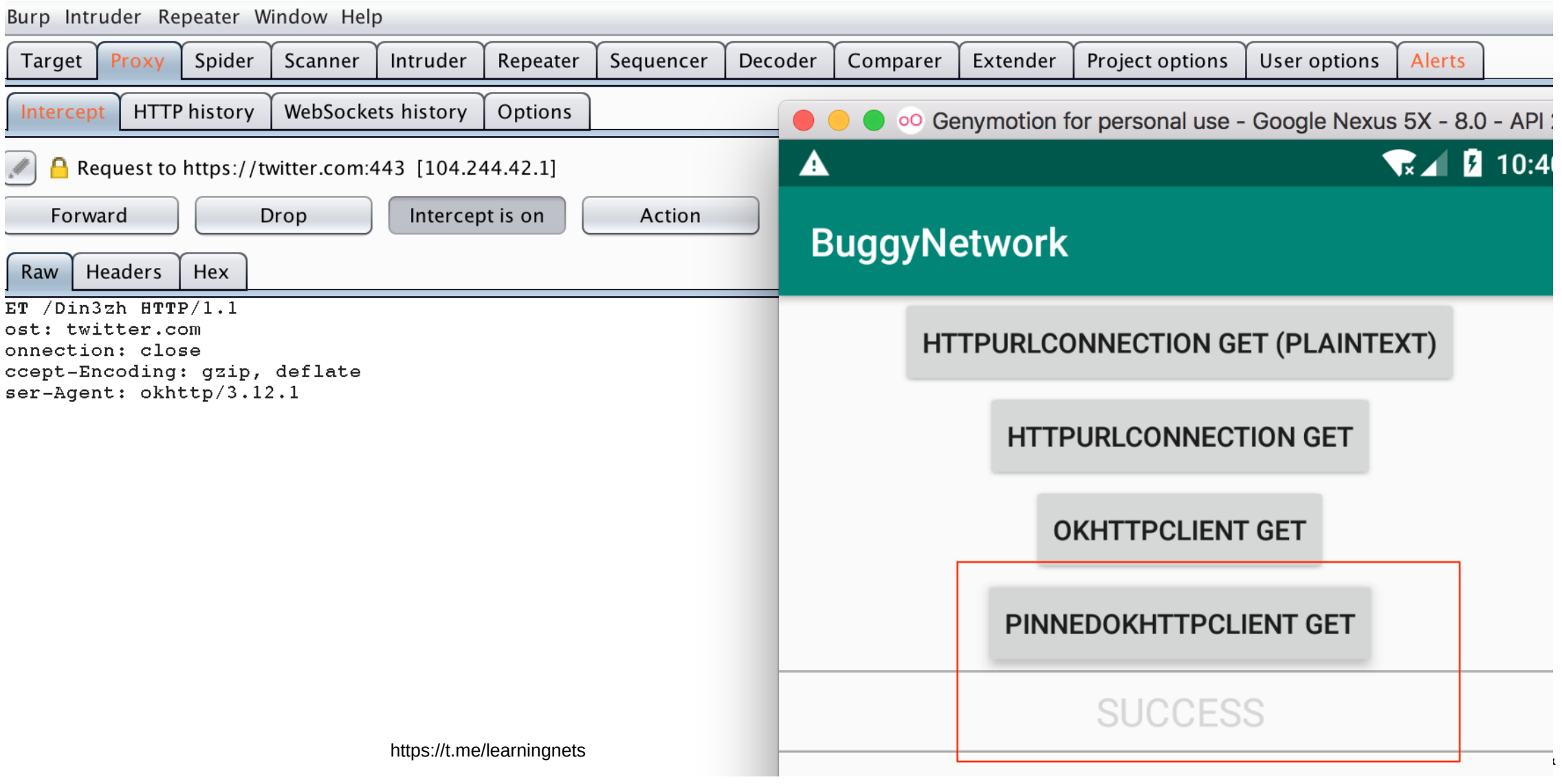
## Patching SMALI Files
## Step 3 – Recompile into an APK file

- `apktool b BuggyNetwork`

# Step 4 – Re-sign the APK file

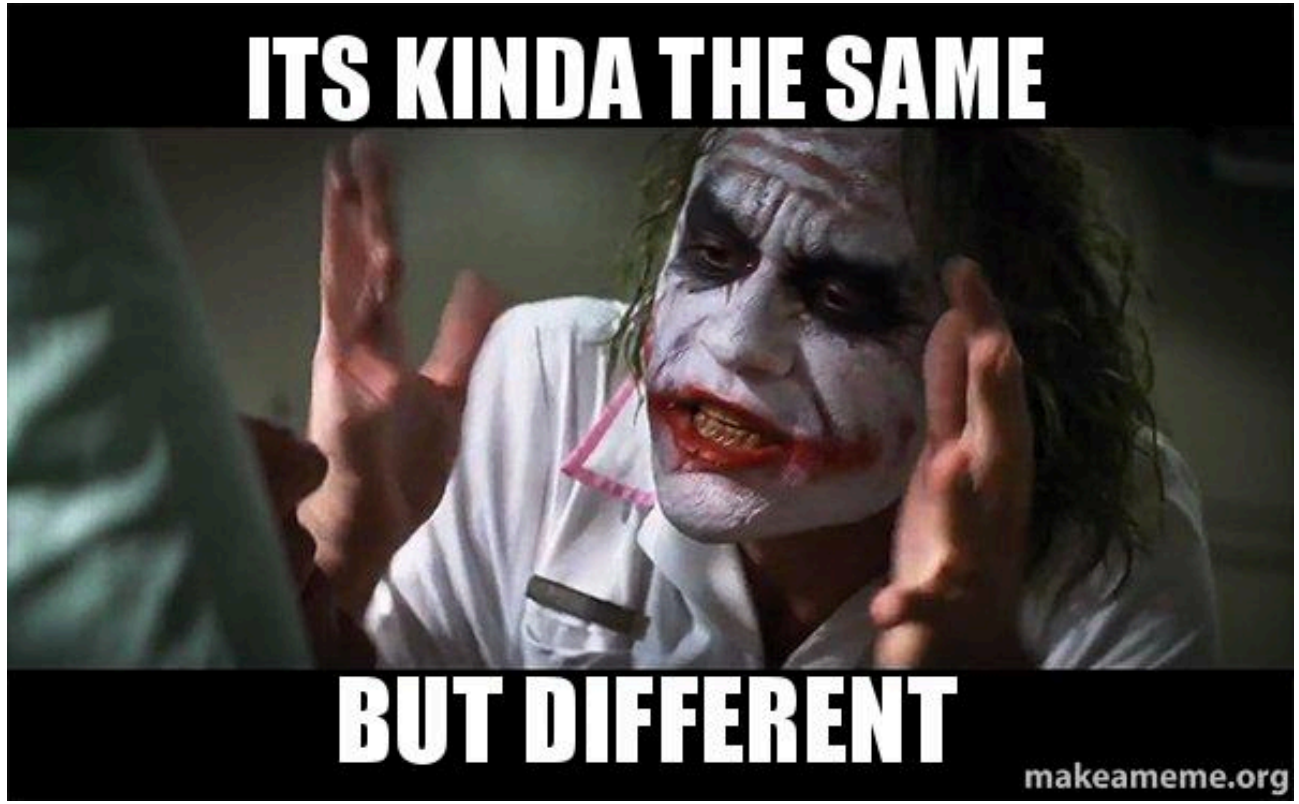
- `keytool -genkey -v -keystore dnskey.jks -alias dnskey -sigalg MD5withRSA -keyalg RSA -keysize 2048 -validity 30`
- `jarsigner -verbose -sigalg MD5withRSA -digestalg SHA1 -keystore dnskey.jks BuggyNetworkUnobfuscatedPatched1.apk dnskey`

# Step 5 - Profit!!

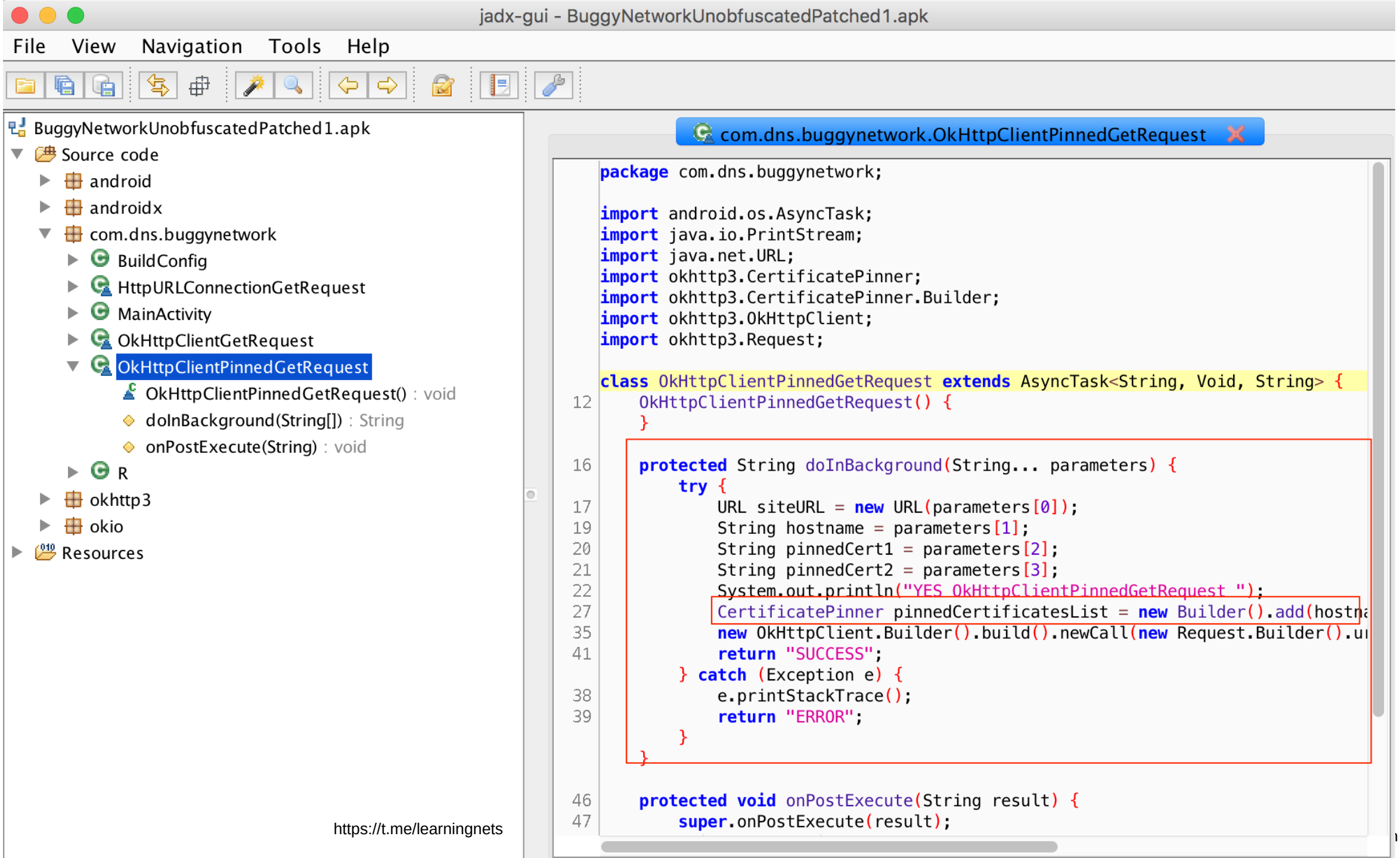


# Alternate Way?

- Look at the file OkHttpClientPinnedGetRequest.java in jadx-gui and find an alternate way, you can bypass certificate pinning.



```
@Override
protected String doInBackground(String... parameters) {
    try {
        URL siteURL = new URL(parameters[0]);

        String hostname = parameters[1]; //eg: twitter.com
        String pinnedCert1 = parameters[2]; //eg: "sha256/BRvG5szpZyF6p3BXtjMBvcFuZDYOrrUzhx2UqcYhkwE="
        String pinnedCert2 = parameters[3]; //eg: "sha256/k2v657xBsOVe1PQRwOsHsw3bsGT2VzIqz5K+59sNQws="
        System.out.println("YES OkHttpClientPinnedGetRequest ");

        CertificatePinner pinnedCertificatesList = new CertificatePinner.Builder()
                .add(hostname, pinnedCert1)
                .add(hostname, pinnedCert2)
                .build();

        OkHttpClient httpConn = new OkHttpClient.Builder()
                .certificatePinner(pinnedCertificatesList)
                .build();

        Request httpRequest = new Request.Builder().url(siteURL).build();

        Response httpResponse = httpConn.newCall(httpRequest).execute();
```



```
invoke-virtual {v6, v7}, Ljava/io/PrintStream;->println(Ljava/lang/String;)V

.line 24
new-instance v6, Lokhttp3/CertificatePinner$Builder;

invoke-direct {v6}, Lokhttp3/CertificatePinner$Builder;-><init>()V

new-array v7, v2, [Ljava/lang/String;

aput-object v4, v7, v1

.line 25
invoke-virtual {v6, v3, v7}, Lokhttp3/CertificatePinner$Builder;->add(Ljava/lang/String;[Ljava/lang/String;)Lokhttp3/CertificatePinner$Builder;

move-result-object v6

new-array v2, v2, [Ljava/lang/String;

aput-object v5, v2, v1

.line 26
invoke-virtual {v6, v3, v2}, Lokhttp3/CertificatePinner$Builder;->add(Ljava/lang/String;[Ljava/lang/String;)Lokhttp3/CertificatePinner$Builder;

move-result-object v1

.line 27
invoke-virtual {v1}, Lokhttp3/CertificatePinner$Builder;->build()Lokhttp3/CertificatePinner;

move-result-object v1

.line 29
.local v1, "pinnedCertificatesList":Lokhttp3/CertificatePinner;
new-instance v2, Lokhttp3/OkHttpClient$Builder;

invoke-direct {v2}, Lokhttp3/OkHttpClient$Builder;-><init>()V

.line 30
invoke-virtual {v2, v1}, Lokhttp3/OkHttpClient$Builder;->certificatePinner(Lokhttp3/CertificatePinner;)Lokhttp3/OkHttpClient$Builder;

move-result-object v2

.line 31
invoke-virtual {v2}, Lokhttp3/OkHttpClient$Builder;->build()Lokhttp3/OkHttpClient;

move-result-object v2

.line 33
.local v2, "httpConn":Lokhttp3/OkHttpClient;
new-instance v6, Lokhttp3/Request$Builder;

invoke-direct {v6}, Lokhttp3/Request$Builder;-><init>()V

invoke-virtual {v6, v0}, Lokhttp3/Request$Builder;->url(Ljava/net/URL;)Lokhttp3/Request$Builder;
```

```
.line 24
new-instance v6, Lokhttp3/CertificatePinner$Builder;

invoke-direct {v6}, Lokhttp3/CertificatePinner$Builder;-><init>()V

new-array v7, v2, [Ljava/lang/String;

aput-object v4, v7, v1

.line 25
invoke-virtual {v6, v3, v7}, Lokhttp3/CertificatePinner$Builder;->add(Ljava/lang/String;[Ljava/lang/String;)Lokhttp3/CertificatePinner$Builder;

move-result-object v6

new-array v2, v2, [Ljava/lang/String;

aput-object v5, v2, v1

.line 26
invoke-virtual {v6, v3, v2}, Lokhttp3/CertificatePinner$Builder;->add(Ljava/lang/String;[Ljava/lang/String;)Lokhttp3/CertificatePinner$Builder;

move-result-object v1

.line 27
invoke-virtual {v1}, Lokhttp3/CertificatePinner$Builder;->build()Lokhttp3/CertificatePinner;

move-result-object v1

.line 29
.local v1, "pinnedCertificatesList":Lokhttp3/CertificatePinner;
new-instance v2, Lokhttp3/OkHttpClient$Builder;

invoke-direct {v2}, Lokhttp3/OkHttpClient$Builder;-><init>()V


.line 31
invoke-virtual {v2}, Lokhttp3/OkHttpClient$Builder;->build()Lokhttp3/OkHttpClient;

move-result-object v2

.line 33
.local v2, "httpConn":Lokhttp3/OkHttpClient;
new-instance v6, Lokhttp3/Request$Builder;

invoke-direct {v6}, Lokhttp3/Request$Builder;-><init>()V

invoke-virtual {v6, v0}, Lokhttp3/Request$Builder;->url(Ljava/net/URL;)Lokhttp3/Request$Builder;

move-result-object v6

invoke-virtual {v6}, Lokhttp3/Request$Builder;->build()Lokhttp3/Request;
```

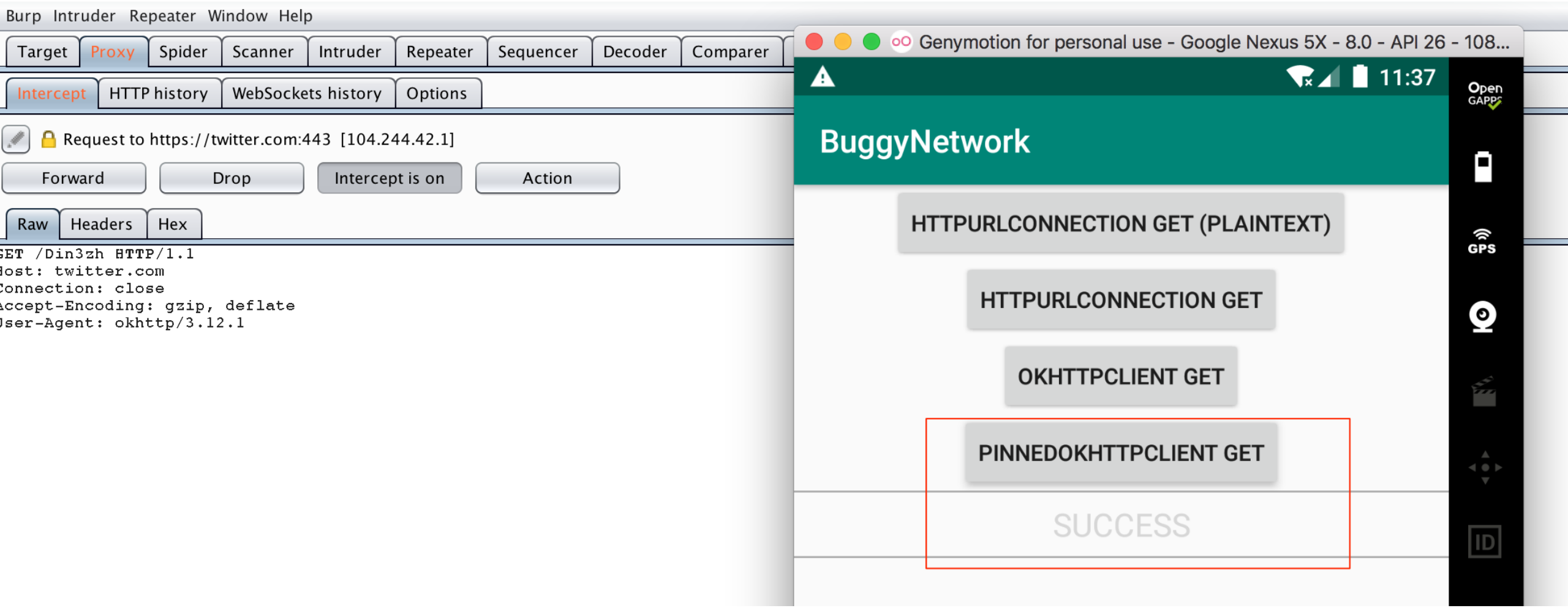
com.dns.buggynetwork.OkHttpClientPinnedGetRequest

```java
package com.dns.buggynetwork;

import android.os.AsyncTask;
import java.io.PrintStream;
import java.net.URL;
import okhttp3.CertificatePinner;
import okhttp3.CertificatePinner.Builder;
import okhttp3.OkHttpClient;
import okhttp3.Request;

class OkHttpClientPinnedGetRequest extends AsyncTask<String, Void, String> {
    OkHttpClientPinnedGetRequest() {
    }

    protected String doInBackground(String... parameters) {
        try {
            URL siteURL = new URL(parameters[0]);
            String hostname = parameters[1];
            String pinnedCert1 = parameters[2];
            String pinnedCert2 = parameters[3];
            System.out.println("YES OkHttpClientPinnedGetRequest ");
            CertificatePinner pinnedCertificatesList = new Builder().add(hostn
            new OkHttpClient.Builder().build().newCall(new Request.Builder().u
            return "SUCCESS";
        } catch (Exception e) {
            e.printStackTrace();
            return "ERROR";
        }
    }
```

com.dns.buggynetwork.OkHttpClientPinnedGetRequest

```java
package com.dns.buggynetwork;

import android.os.AsyncTask;
import java.io.PrintStream;
import java.net.URL;
import okhttp3.CertificatePinner;
import okhttp3.OkHttpClient.Builder;
import okhttp3.Request;

class OkHttpClientPinnedGetRequest extends AsyncTask<String, Void, String> {
    OkHttpClientPinnedGetRequest() {
    }

    protected String doInBackground(String... parameters) {
        try {
            URL siteURL = new URL(parameters[0]);
            String hostname = parameters[1];
            String pinnedCert1 = parameters[2];
            String pinnedCert2 = parameters[3];
            System.out.println("YES OkHttpClientPinnedGetRequest ");
            new Builder().certificatePinner(new CertificatePinner.Builder().add(hostname, p
            return "SUCCESS";
        } catch (Exception e) {
            e.printStackTrace();
            return "ERROR";
        }
    }

    protected void onPostExecute(String result) {
```

# Finally..



# Network Traffic Analysis

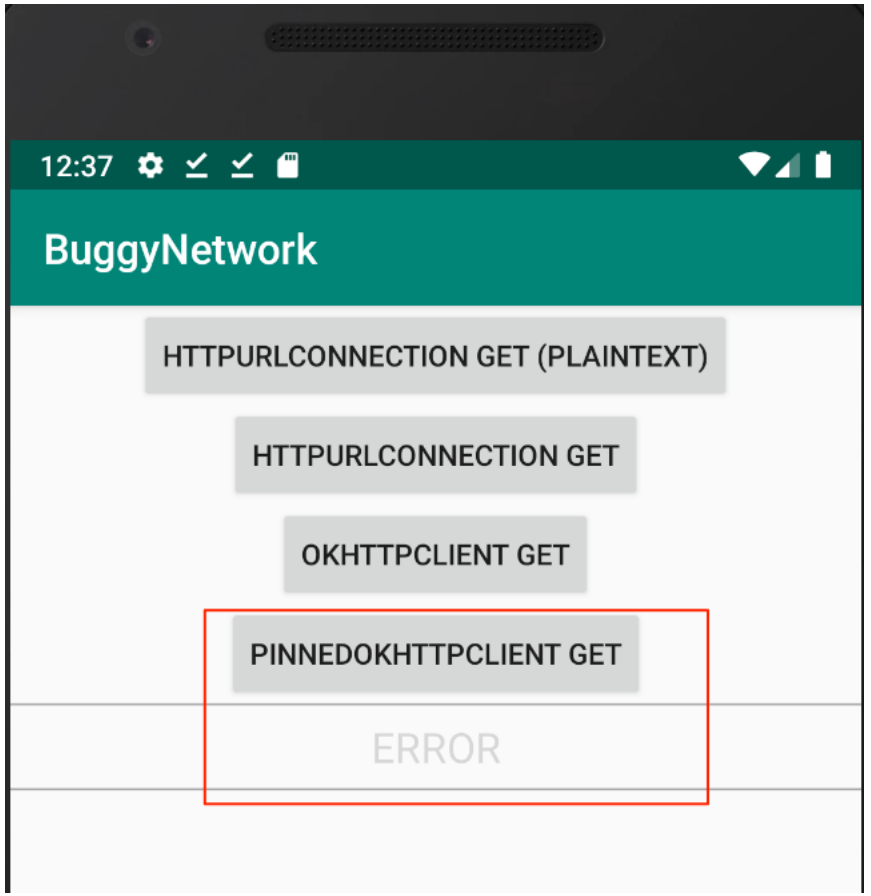
CERTIFICATE PINNING - OkHttpClient

TARGET - /home/mobile/Desktop/vulnapps/network_communication_labs/BuggyNetwork.apk

Method – Dynamic Patching using Frida

# Lets Step Back a little…

- Basic Method worked for other non-certificate-pinned options
- Fails when Certificate Pinning is used



CertificatePinner.java

```
159       */
160      public void check(String hostname, List<Certificate> peerCertificates)
161          throws SSLPeerUnverifiedException {
162        List<Pin> pins = findMatchingPins(hostname);
163        if (pins.isEmpty()) return;
164
165        if (certificateChainCleaner != null) {
166          peerCertificates = certificateChainCleaner.clean(peerCertificates, hostname);
167        }
168
169        for (int c = 0, certsSize = peerCertificates.size(); c < certsSize; c++) {
170          X509Certificate x509Certificate = (X509Certificate) peerCertificates.get(c);
171
172          // Lazily compute the hashes for each certificate.
173          ByteString sha1 = null;
174          ByteString sha256 = null;
175
176          for (int p = 0, pinsSize = pins.size(); p < pinsSize; p++) {
177            Pin pin = pins.get(p);
178            if (pin.hashAlgorithm.equals("sha256/")) {
179              if (sha256 == null) sha256 = sha256(x509Certificate);
180              if (pin.hash.equals(sha256)) return; // Success!
181            } else if (pin.hashAlgorithm.equals("sha1/")) {
182              if (sha1 == null) sha1 = sha1(x509Certificate);
183              if (pin.hash.equals(sha1)) return; // Success!
184            } else {
185              throw new AssertionError( detailMessage: "unsupported hashAlgorithm: " + pin.hashAlgorithm);
186            }
187          }
188        }
```

# So…

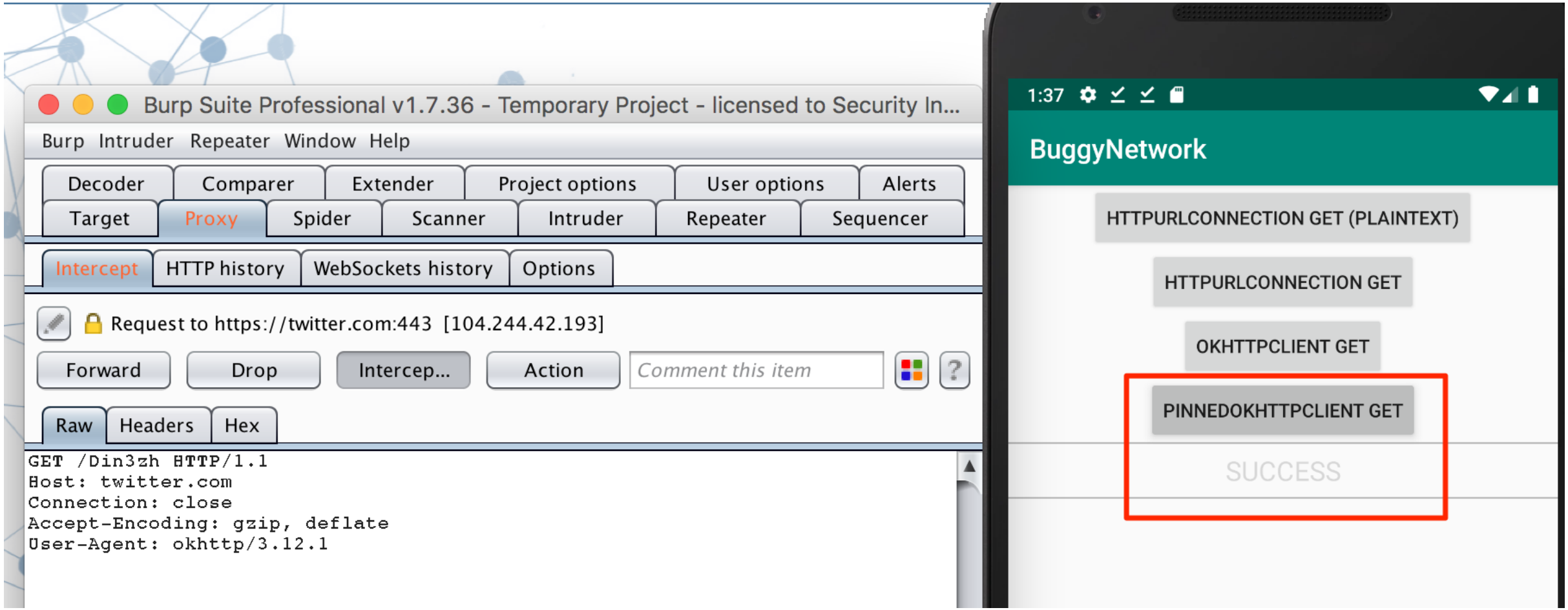
- Class to be hooked – okhttp3.CertificatePinner
- Function to be hooked – check(String, List)
- Manipulate – `return` always

# Frida Script – Bypass Okhttp Pinning

```
console.log("\n********Certificate Pinning Bypass - Okhttp********");

Java.perform(function () {

var CertificatePinner = Java.use("okhttp3.CertificatePinner");

CertificatePinner.check.overload('java.lang.String', 'java.util.List').implementation = function (str1,str2){
    console.log('===>Okhttp CertificatePinner Intercepted')
    console.log('===>Bypassing Okhttp CertificatePinner.check()')
    return ;
};
});
```

/home/mobile/Desktop/vulnapps/network_communication_labs/okhttp_pinning.js

# Frida Script – Bypass Okhttp Pinning



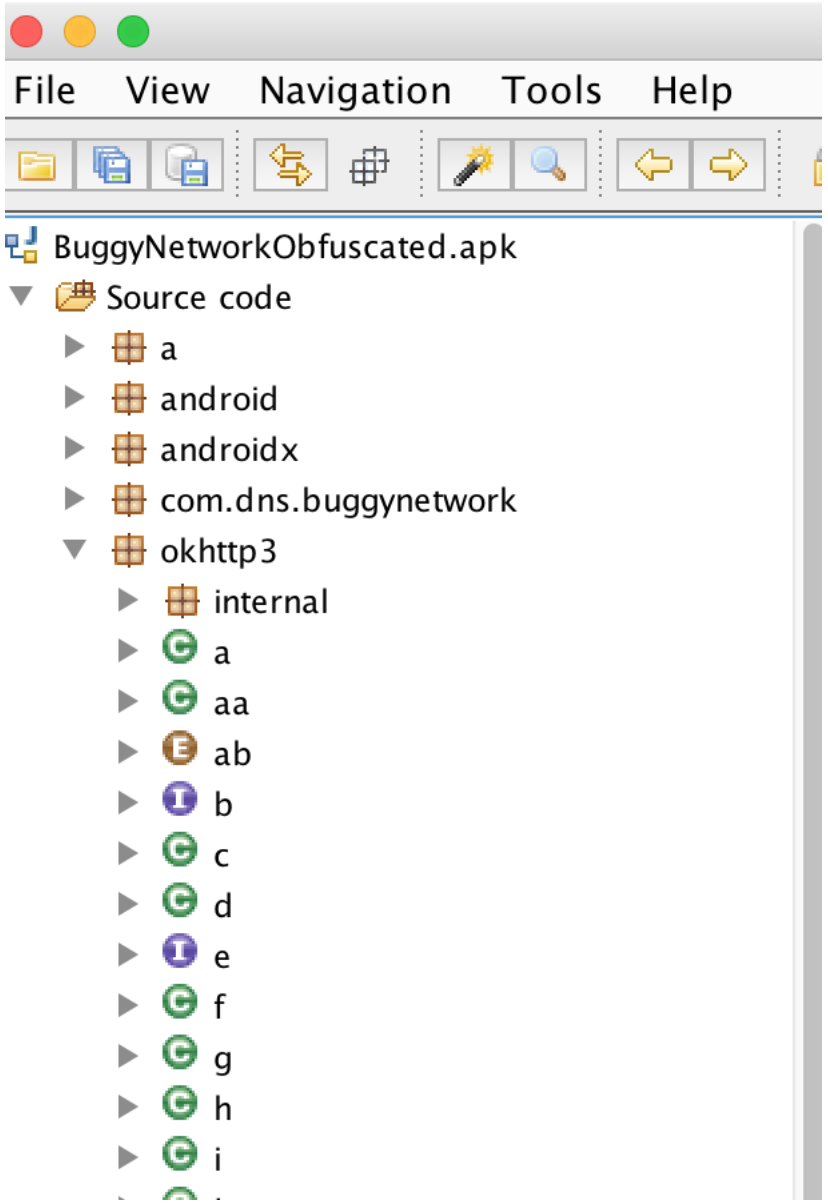
# Network Traffic Analysis

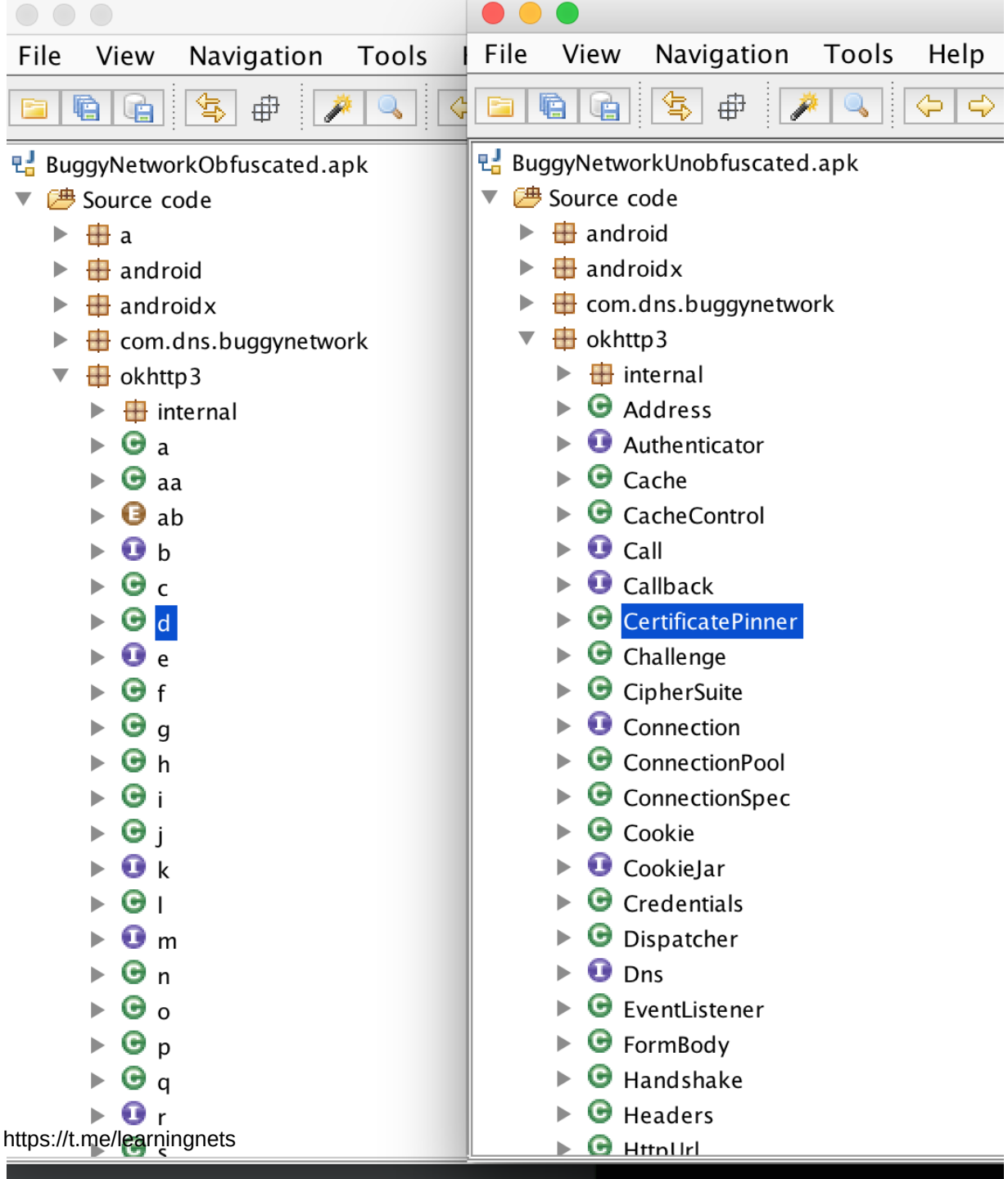
## CERTIFICATE PINNING - OkHttpClient

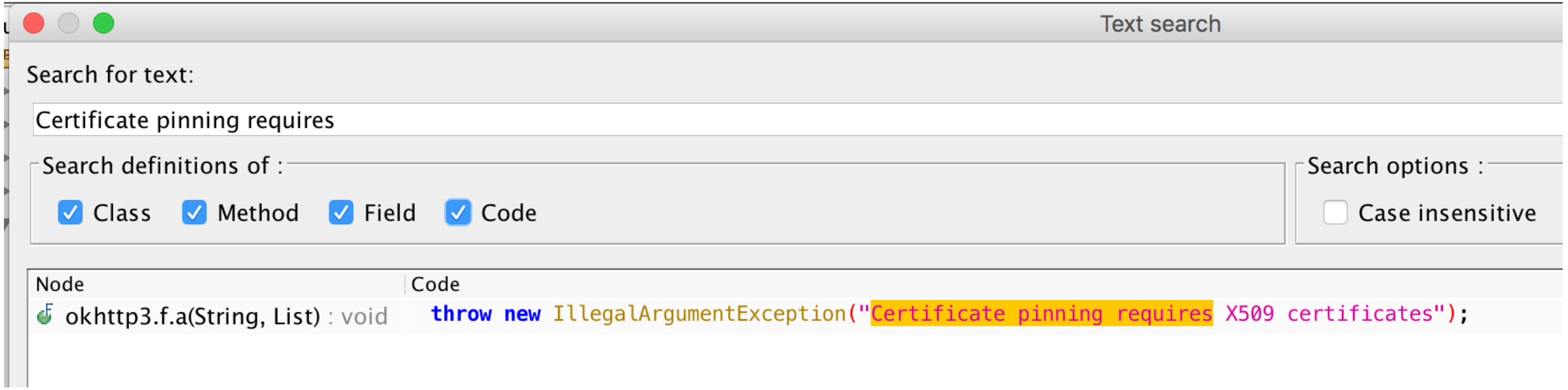
TARGET - /home/mobile/Desktop/vulnapps/
network_communication_labs/BuggyNetworkObfuscated.apk

# Obfuscated APK?

```
Error: java.lang.ClassNotFoundException: Didn't find class "okhttp3.CertificateP
inner" on path: DexPathList[[zip file "/data/app/com.dns.buggynetwork-KY9wPSW8tr
tYQsO9GQ43fQ==/base.apk"],nativeLibraryDirectories=[/data/app/com.dns.buggynetwo
rk-KY9wPSW8trtYQsO9GQ43fQ==/lib/x86, /system/lib]]
    at frida/node_modules/frida-java/lib/env.js:222
    at ensureClass (frida/node_modules/frida-java/lib/class-factory.js:703)
    at frida/node_modules/frida-java/lib/class-factory.js:168
    at /repl1.js:7
    at frida/node_modules/frida-java/lib/vm.js:42
    at E (frida/node_modules/frida-java/index.js:348)
    at frida/node_modules/frida-java/index.js:300
    at frida/node_modules/frida-java/lib/vm.js:42
    at frida/node_modules/frida-java/index.js:280
[Android Emulator 5554::com.dns.buggynetwork]->
```








OR

```
grep -ir "java/lang/String;\[Ljava/lang/String;)L" .
```

```
java -d64 -Xms512M -Xm  ..uscated/smali  java -d64 -Xms512M -X  adb

➜  smali grep -ir "java/lang/String;\[Ljava/lang/String;)L" .
./okhttp3/f$a.smali:.method public final varargs a(Ljava/lang/String;[Ljava/lang/String;)Lokhttp3/f$a;
./com/dns/buggynetwork/c.smali:    invoke-virtual {v5, v3, v6}, Lokhttp3/f$a;->a(Ljava/lang/String;[Ljava/lang/St
ring;)Lokhttp3/f$a;
./com/dns/buggynetwork/c.smali:    invoke-virtual {v4, v3, v2}, Lokhttp3/f$a;->a(Ljava/lang/String;[Ljava/lang/St
ring;)Lokhttp3/f$a;
➜  smali
```

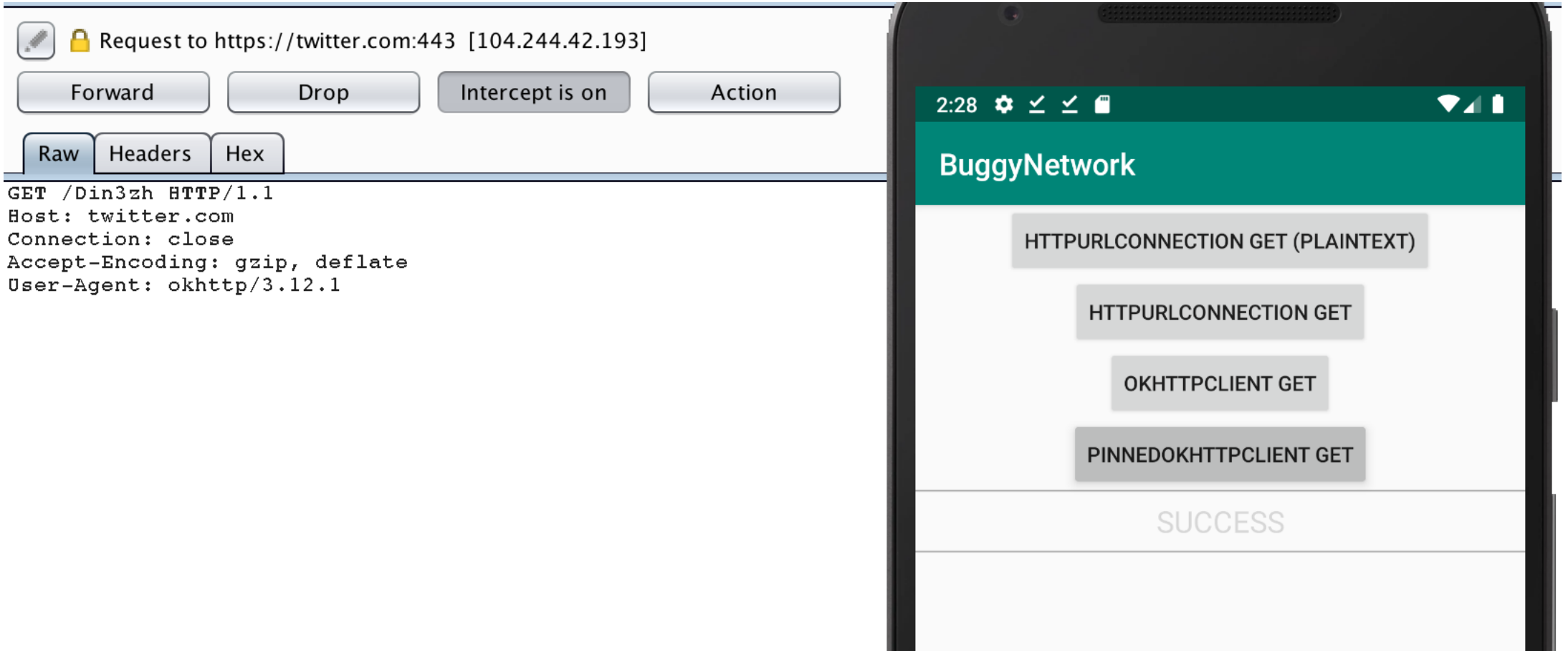
# So…

- Class to be hooked – okhttp3.f
- Function to be hooked – a(String, List)
- Manipulate – `return` always

# Frida Script – Bypass Obfuscated Okhttp Pinning

```
console.log("\n********Certificate Pinning Bypass Obfuscated - Okhttp********");

Java.perform(function () {

var CertificatePinner = Java.use("okhttp3.f");

CertificatePinner.a.overload('java.lang.String', 'java.util.List').implementation = function (str1,str2){
    console.log('===>Okhttp CertificatePinner Intercepted')
    console.log('===>Bypassing Okhttp CertificatePinner.check()')
    return ;
};
});
```

# Frida Script – Bypass Obfuscated Okhttp Pinning



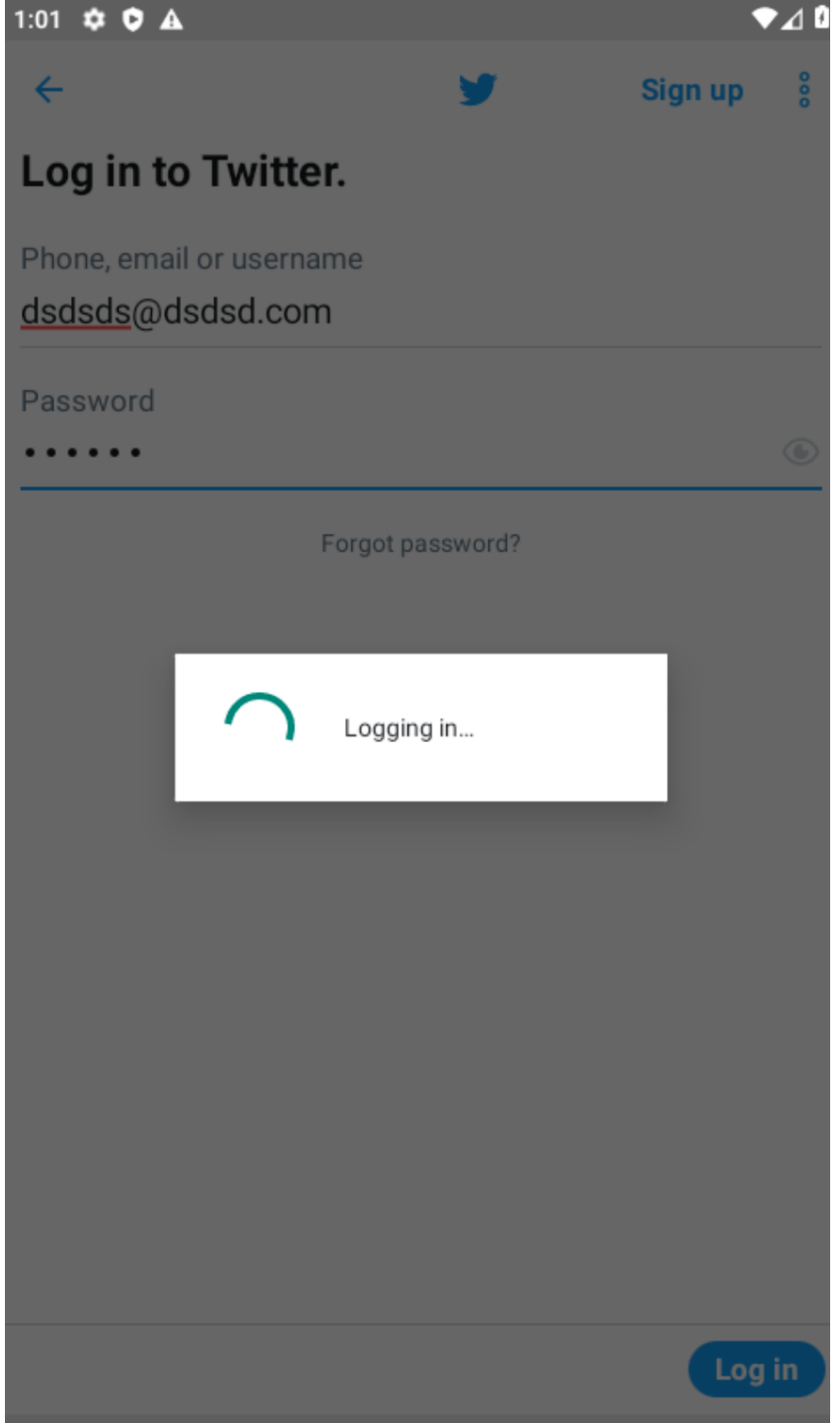
# Network Traffic Analysis

## CERTIFICATE PINNING BYPASS – TrustManager

TARGET – /home/mobile/Desktop/vulnapps/
network_communication_labs/twitter_trustmanager1/Twitter.apk

# GOAL

- Target is the Twitter application on Genymotion
- Use an easier/different approach compared to before – hooking TrustManager



# Hooking TrustManager Template

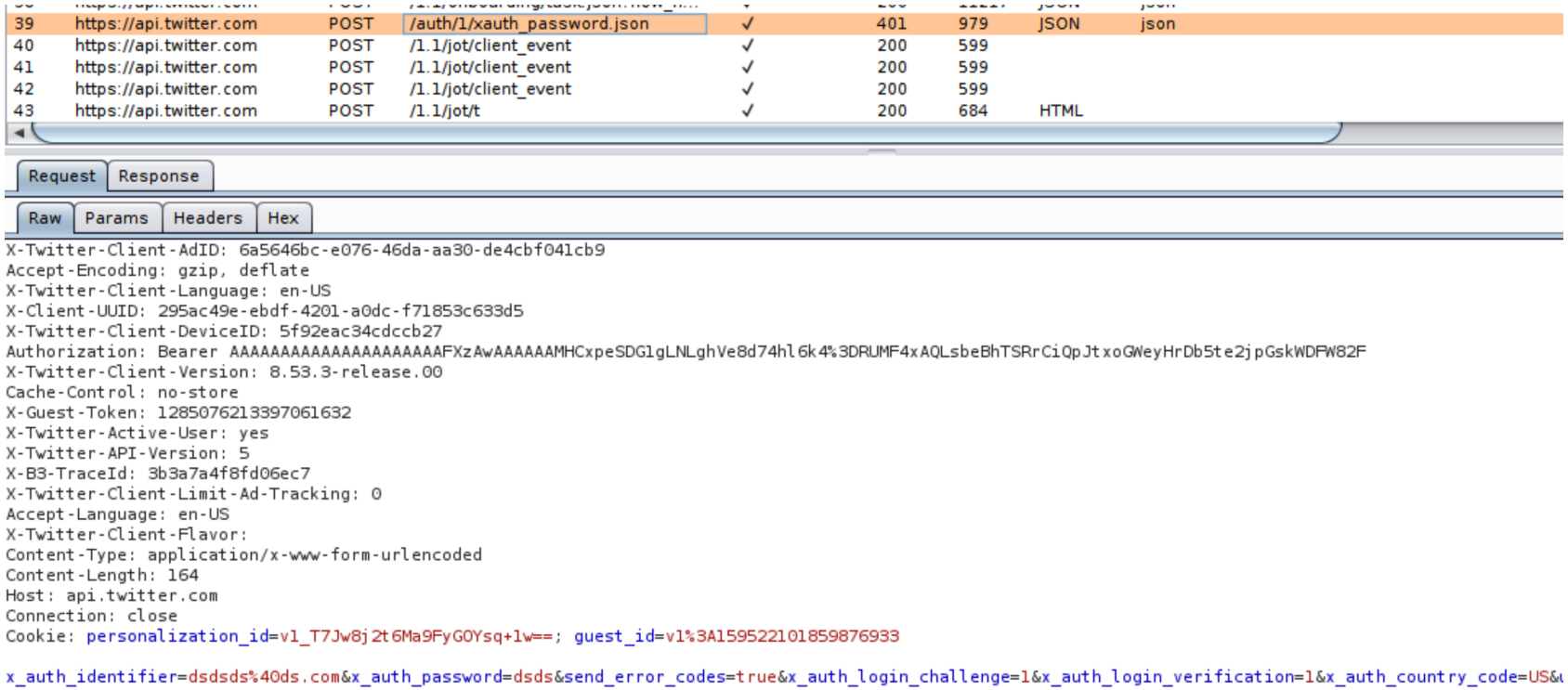
```
var X509TrustManager = Java.use('javax.net.ssl.X509TrustManager');
    var SSLContext = Java.use('javax.net.ssl.SSLContext');
    // Create TEST TrustManager
    var TrustManager = Java.registerClass({
        name: 'com.dns.TrustManager',
        implements: [X509TrustManager],
        methods: {
            checkClientTrusted: function (chain, authType) {
            },
            checkServerTrusted: function (chain, authType) {
            },
            getAcceptedIssuers: function () {
                return [];
            }
        }
    });
```

# Hooking TrustManager Template

```
//Spoof TrustManager when called
    var TrustManagers = [TrustManager.$new()];
    var SSLContextInitialization =
        SSLContext.init.overload(
        '[Ljavax.net.ssl.KeyManager;', '[Ljavax.net.ssl.TrustManager;',
'java.security.SecureRandom'
    );
    SSLContextInitialization.implementation = function (keyManager, trustManager,
secureRandom) {
        console.log('TrustManager called')
        SSLContextInitialization.call(this, keyManager, TrustManagers, secureRandom);
    };


});
```

# Hooking TrustManager - Twitter

- `frida -U -l bypass_twitter_certificate_pinning.js -f` **`com.twitter.android`** `--no-pause`




/home/mobile/Desktop/vulnapps/network_communication_labs/twitter_trustmanager1/bypass_twitter_certificate_pinning.js

# Network Traffic Analysis

## CERTIFICATE PINNING BYPASS – CertificateFactory

TARGET - /home/mobile/Desktop/vulnapps/
network_communication_labs/Snapchat.apk

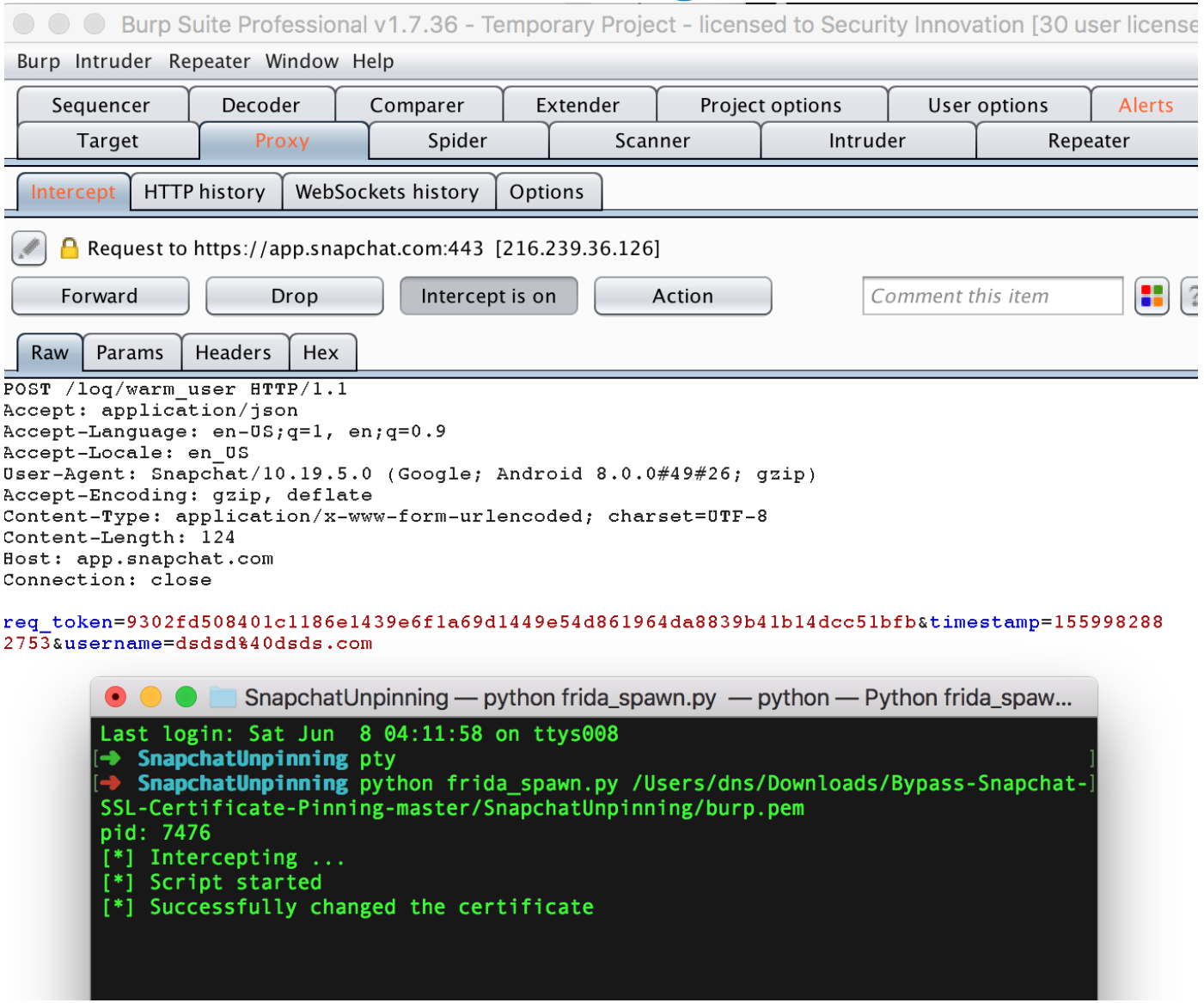
# Bypass Snapchat Certificate Pinning – CertificateFactory Template

```
Java.perform(function () {

    console.log('[*] Script started');

    const certificateArray = Java.use('[Ljava.lang.String;');
    const JavaString = Java.use('java.lang.String');
    var myCertificate = null;

    recv('input', function(value) {
        myCertificate = JavaString.$new(value.payload);
    });

    var HookedClass = Java.use('java.security.cert.CertificateFactory');
    const InputStream = Java.use('java.io.ByteArrayInputStream');
    var inStreamCertificate = InputStream.$new(myCertificate.getBytes());

    var done = false;

    HookedClass.generateCertificate.implementation = function (inStream) {

        if(!done) { // we will change only the first certificate to ours
            console.log("[*] Successfully changed the certificate");
            done = true;
            return this.generateCertificate(inStreamCertificate);
        }
        return this.generateCertificate(inStream);
```

# Bypass Snapchat Certificate Pinning - CertificateFactory

- Refer working script from / home/mobile/ Desktop/vulnapps/ network_communicati on_labs/ Snapchat_Certificate Factory_Pinning.zip

- python frida_spawn.py burp.pem
  - Observe that you can intercept Snapchat traffic



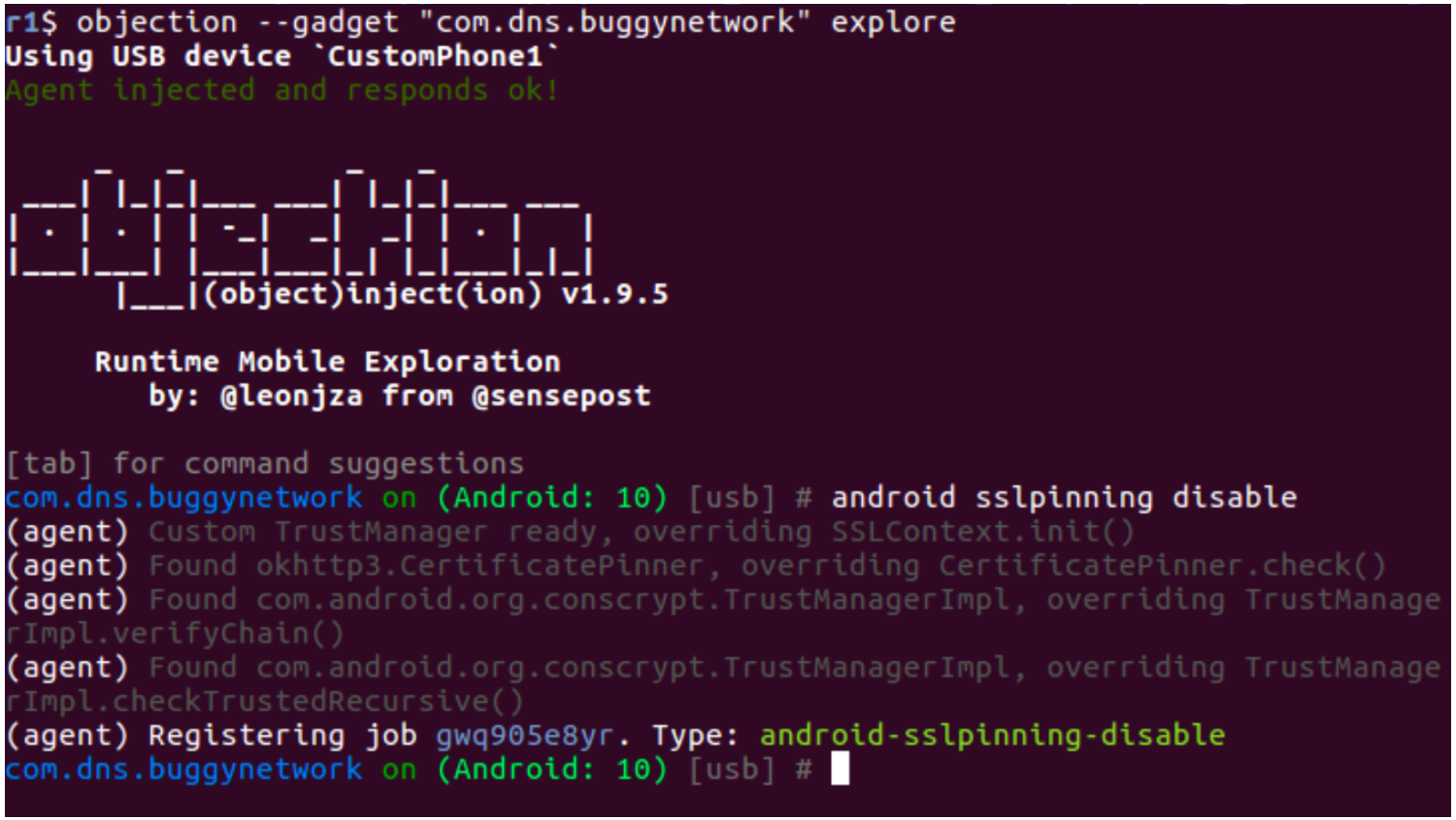
# Network Traffic Analysis

## CERTIFICATE PINNING BYPASS – Automated Technique

TARGET - /home/mobile/Desktop/vulnapps/
network_communication_labs/BuggyNetwork.apk

# Automated Certificate Pinning Bypass

- adb install BuggyNetwork.apk
- pip3 install objection
- objection --gadget "com.dns.buggynetwork" explore
- android sslpinning disable



# TASK

- Bypass Certificate Pinning in the BuggyNetwork application - ***PINNEDOKHTTPCLIENT GET*** using objection