Introducing FRIDA

Frida - Introduction

- Tool for reverse engineering and dynamic code instrumentation
- Works by injecting JavaScript engine and console into running processes
- Useful for application information gathering, dynamic manipulation

Using FRIDA – Android Setup

- Run `adb shell getprop ro.product.cpu.abi` to get the version
  - In our case it is arm64-v8a -> frida-server-<version>-android-arm64.xz
- From https://github.com/frida/frida/releases, download frida-server-<version>-android- arm64.xz
- Unzip frida-server-<version>-android-arm.xz using `xz -d <file>`
- Rename the `frida-server-<version>-android-arm` to `frida-server`
- `adb push frida-server /data/local/tmp`

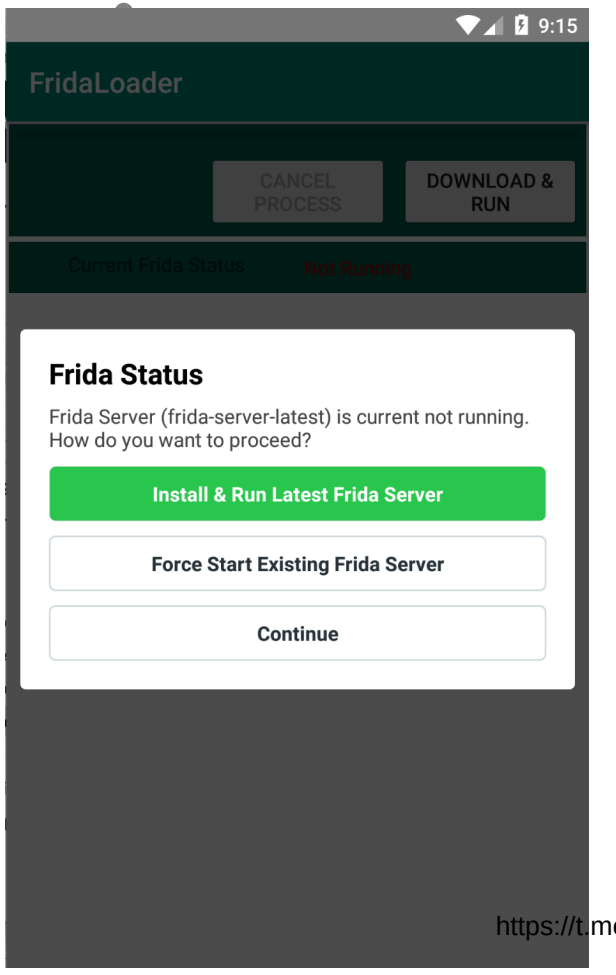
Using FRIDA – Android Setup

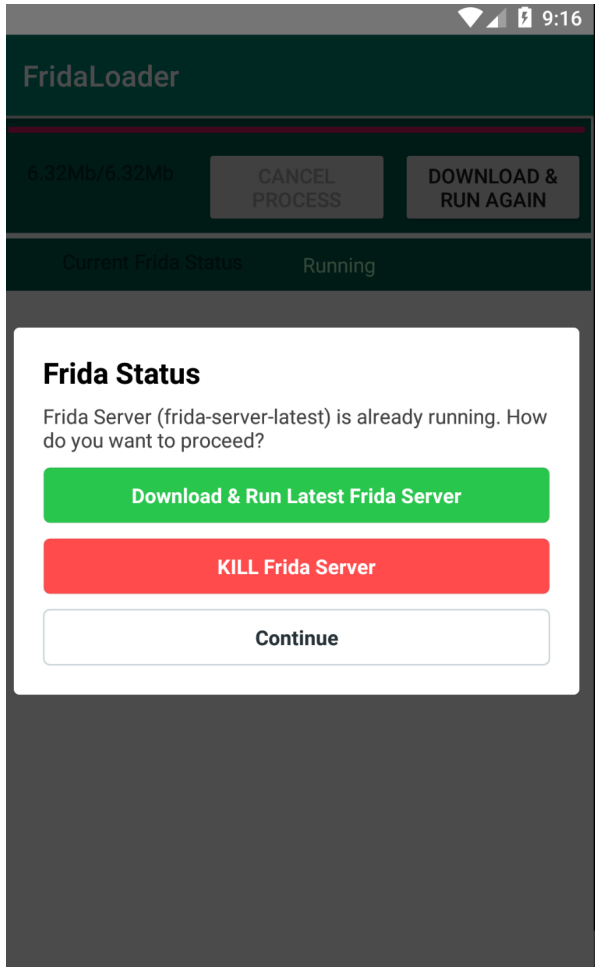
- `adb shell`
- `su`
- `cd /data/local/tmp`
- `chmod +x frida-server`
- `./frida-server &`

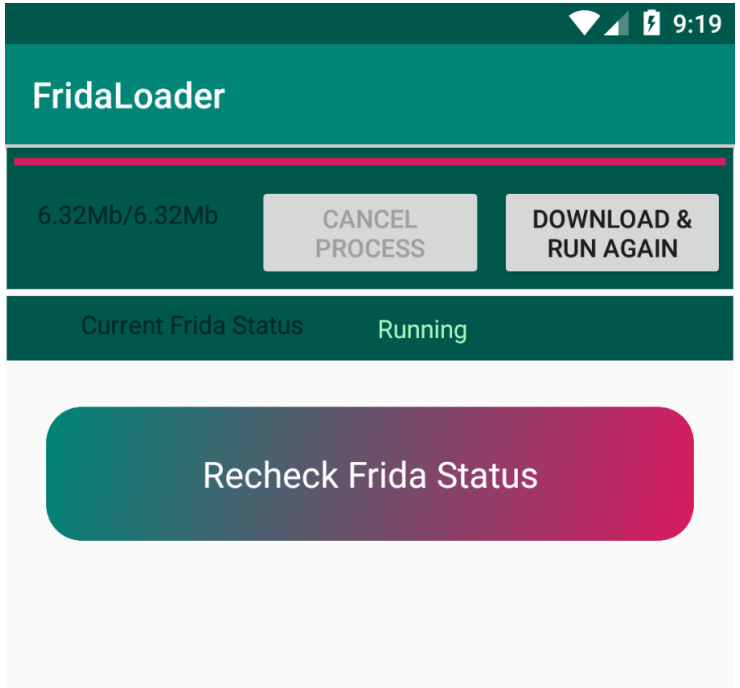
OR

Introducing FridaLoader

- Download from https://github.com/dineshshetty/FridaLoader

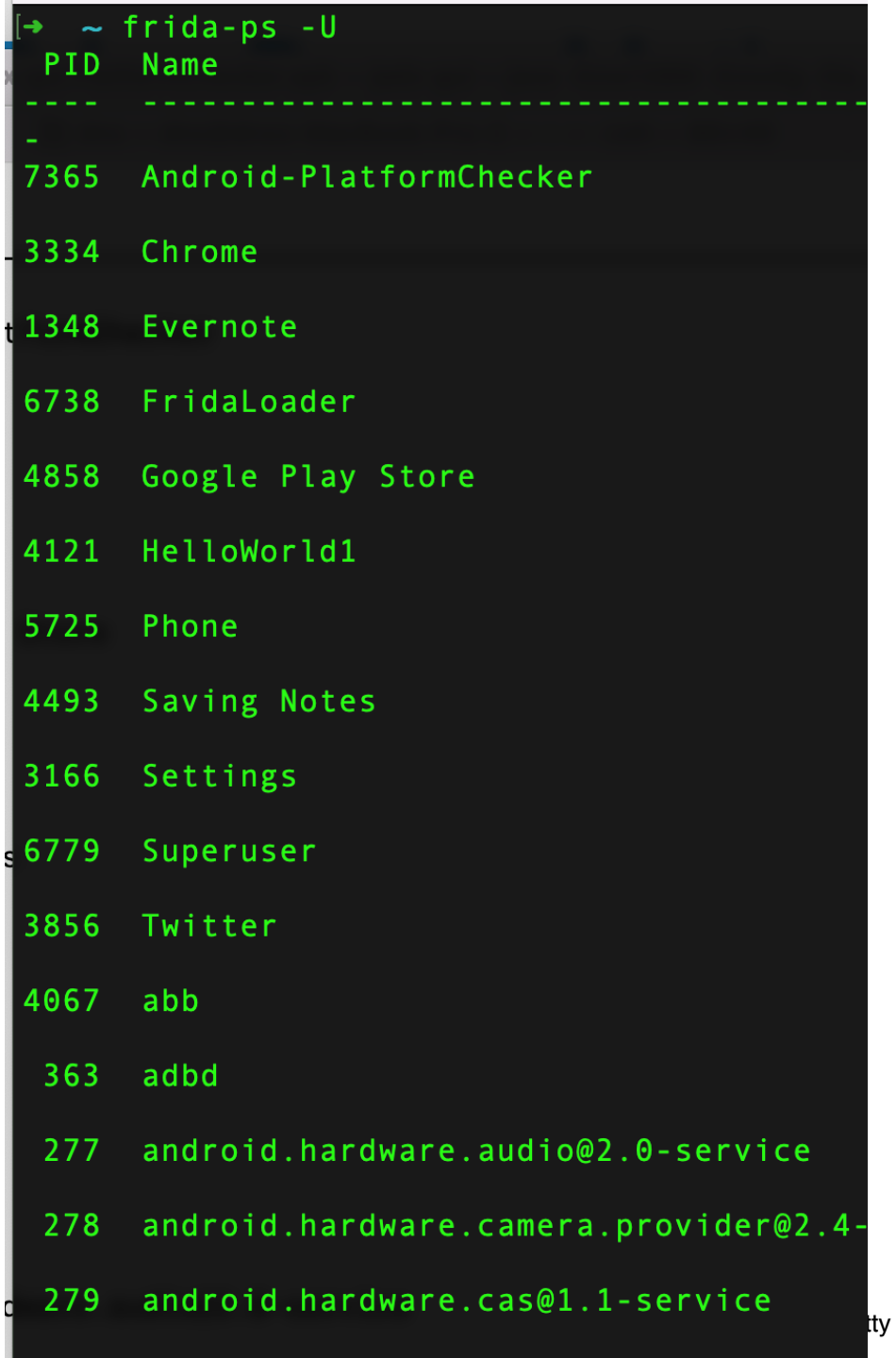






Using FRIDA – Mac Setup

- `pip install frida`
- If Error -> Use -> sudo -H pip install frida --ignore-installed six
- `sudo pip install frida-tools`
- `easy_install pyasn1`
- Run `` `frida-ps -Uai` `` to confirm if Frida works



Problems!

```
➜  ~ frida-ps -U
Failed to enumerate processes: unable to connect to remote frida-
➜  ~
```

```
dns — dns@dns-mac — ~ — -zsh — 80×50
➜  ~ system_profiler SPUSBDataType
USB:

    USB 3.0 Bus:

      Host Controller Driver: AppleUSBXHCILPTH
      PCI Device ID: 0x8c31
      PCI Revision ID: 0x0005
      PCI Vendor ID: 0x8086

        Apple Internal Keyboard / Trackpad:

          Product ID: 0x0274
          Vendor ID: 0x05ac (Apple Inc.)
          Version: 6.24
          Serial Number: D3H52015DF1FTV4AG3ES
          Speed: Up to 12 Mb/sec
          Manufacturer: Apple Inc.
          Location ID: 0x14400000 / 4
          Current Available (mA): 500
          Current Required (mA): 500
          Extra Operating Current (mA): 0
          Built-In: Yes

        Bluetooth USB Host Controller:

          Product ID: 0x8290
          Vendor ID: 0x05ac (Apple Inc.)
          Version: 1.67
          Speed: Up to 12 Mb/sec
          Manufacturer: Broadcom Corp.
          Location ID: 0x14300000 / 5
          Current Available (mA): 500
          Current Required (mA): 0
          Extra Operating Current (mA): 0
          Built-In: Yes

        iPhone:

          Product ID: 0x12a8
          Vendor ID: 0x05ac (Apple Inc.)
          Version: 10.03
          Serial Number: ec96ba[redacted]1
          Speed: Up to 480 Mb/sec
          Manufacturer: Apple Inc.
          Location ID: 0x14100000 / 11
          Current Available (mA): 500
          Current Required (mA): 500
```

Important Note

- Everything you need to run all of the Frida labs including scripts and apk can be found in the folder:
  - /home/mobile/Desktop/vulnapps/frida_labs/

TASK - 2mins

- Launch FridaLoader on Genymotion & start Frida server
- Launch installed RootChecker application in Genymotion
  - Binary can be found at /home/mobile/Desktop/vulnapps/frida_labs/RootChecker.apk
- Find the process-id of the RootChecker application using frida-ps

Note 1 - Make sure to run launch server using FridaLoader.apk
Note 2 - Make sure that target application is in the foreground

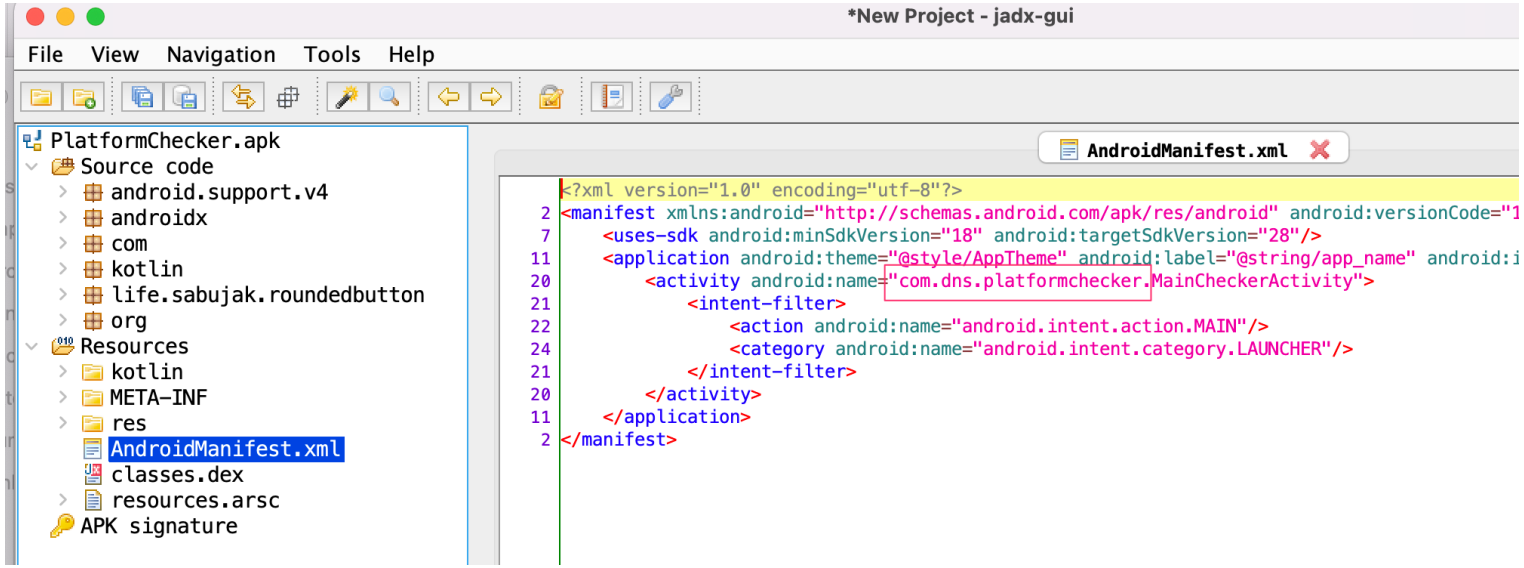
Solution

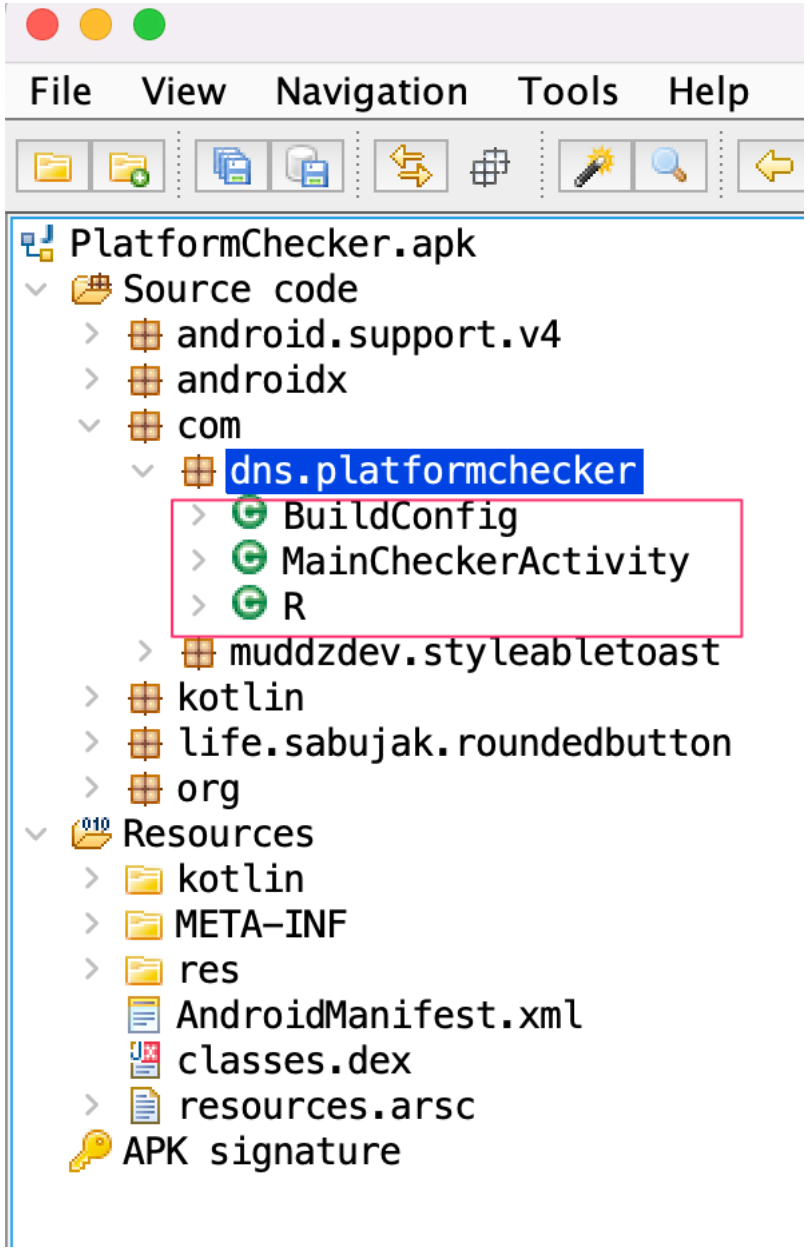
- *`frida-ps -U | grep -i root`*
  - *com.tiagorlampert.rootchecker*

FRIDA

Dump Class Information

Classes? Well…





FRIDA

Dump Methods list

Using FRIDA – Dump Methods - TEMPLATE

```
function listClassMethods(classNameForSearch) {
    var hook = Java.use(classNameForSearch);
    var declaredMethods = hook.class.getDeclaredMethods();
    hook.$dispose;
    return declaredMethods;
}

setTimeout(function() {
    if (Java.available) {
        Java.perform(function() {
            //Change the values for class_name_for_search as needed
            var class_name_for_search = '<<CLASSNAME>>';

            var all_methods_in_class = listClassMethods(class_name_for_search)
            var methods_return = ''

            all_methods_in_class.forEach(function(args) {
                methods_return += String(args) + '\n\n'
            });

            console.log("\n================================================");
            console.log(methods_return);
            console.log("\n================================================");

        })
    }
}, 0);
```

Using FRIDA – Dump Methods - PlatformChecker

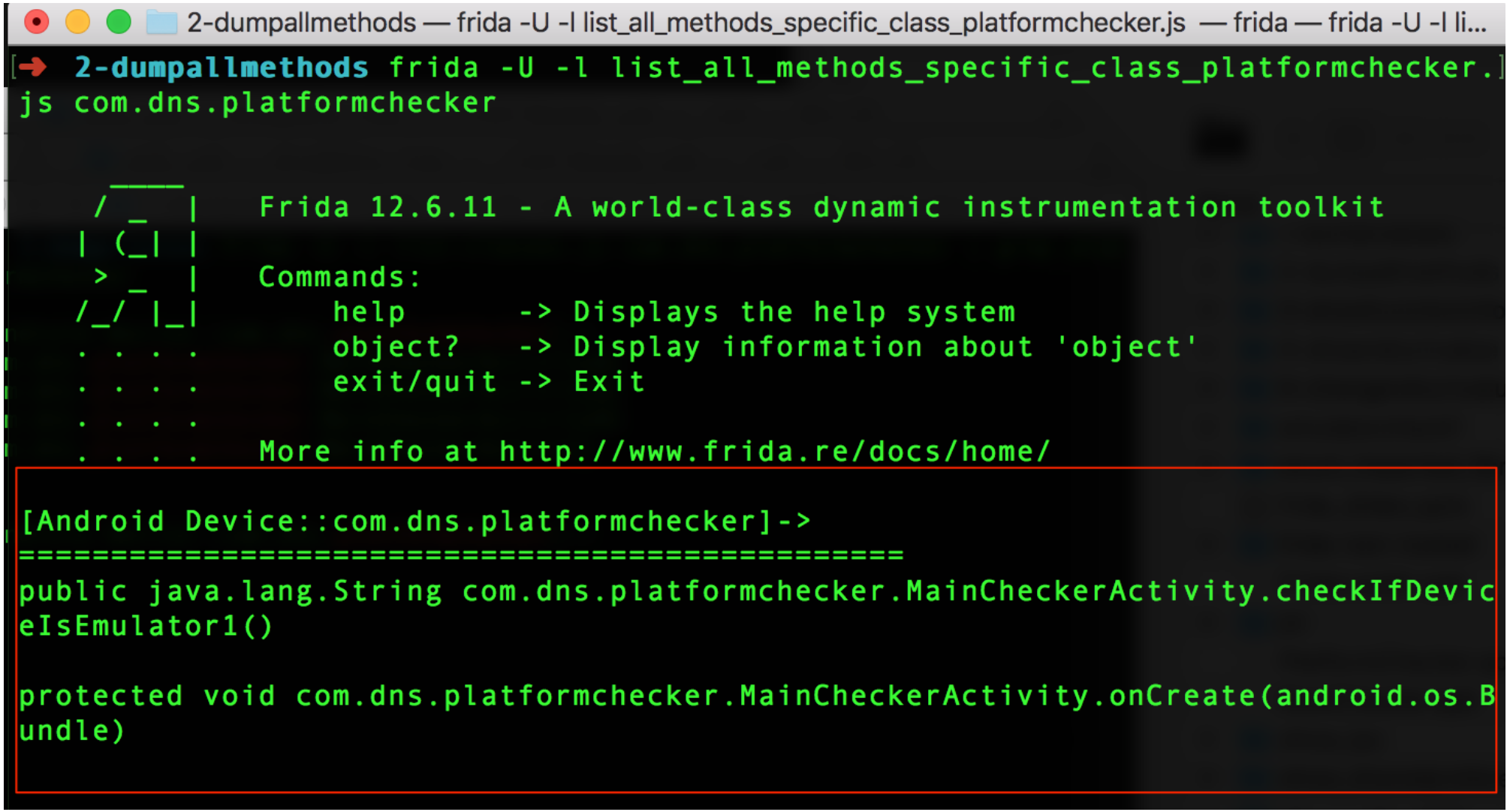
```
function listClassMethods(classNameForSearch) {
    var hook = Java.use(classNameForSearch);
    var declaredMethods = hook.class.getDeclaredMethods();
    hook.$dispose;
    return declaredMethods;
}

setTimeout(function() {
    if (Java.available) {
        Java.perform(function() {
            //Change the values for class_name_for_search as needed
            var class_name_for_search = 'com.dns.platformchecker.MainCheckerActivity';

            var all_methods_in_class = listClassMethods(class_name_for_search)
            var methods_return = ''

            all_methods_in_class.forEach(function(args) {
                methods_return += String(args) + '\n\n'
            });

            console.log("\n================================================");
            console.log(methods_return);
            console.log("\n================================================");

        })
    }
}, 0);
```

Using FRIDA – Dump Methods Information

- Run the application once and then inject script:

```
frida -U -l
list_all_methods_specific_cl
ass_platformchecker.js
"Android-PlatformChecker"
```

```
frida -U -l list_all_methods_specific_class_platformchecker.js "Android-PlatformChecker"
```



/home/mobile/Desktop/vulnapps/frida_labs/2-dumpallmethods/list_all_methods_specific_class_platformchecker.js

FRIDA

Show Return Value Hook

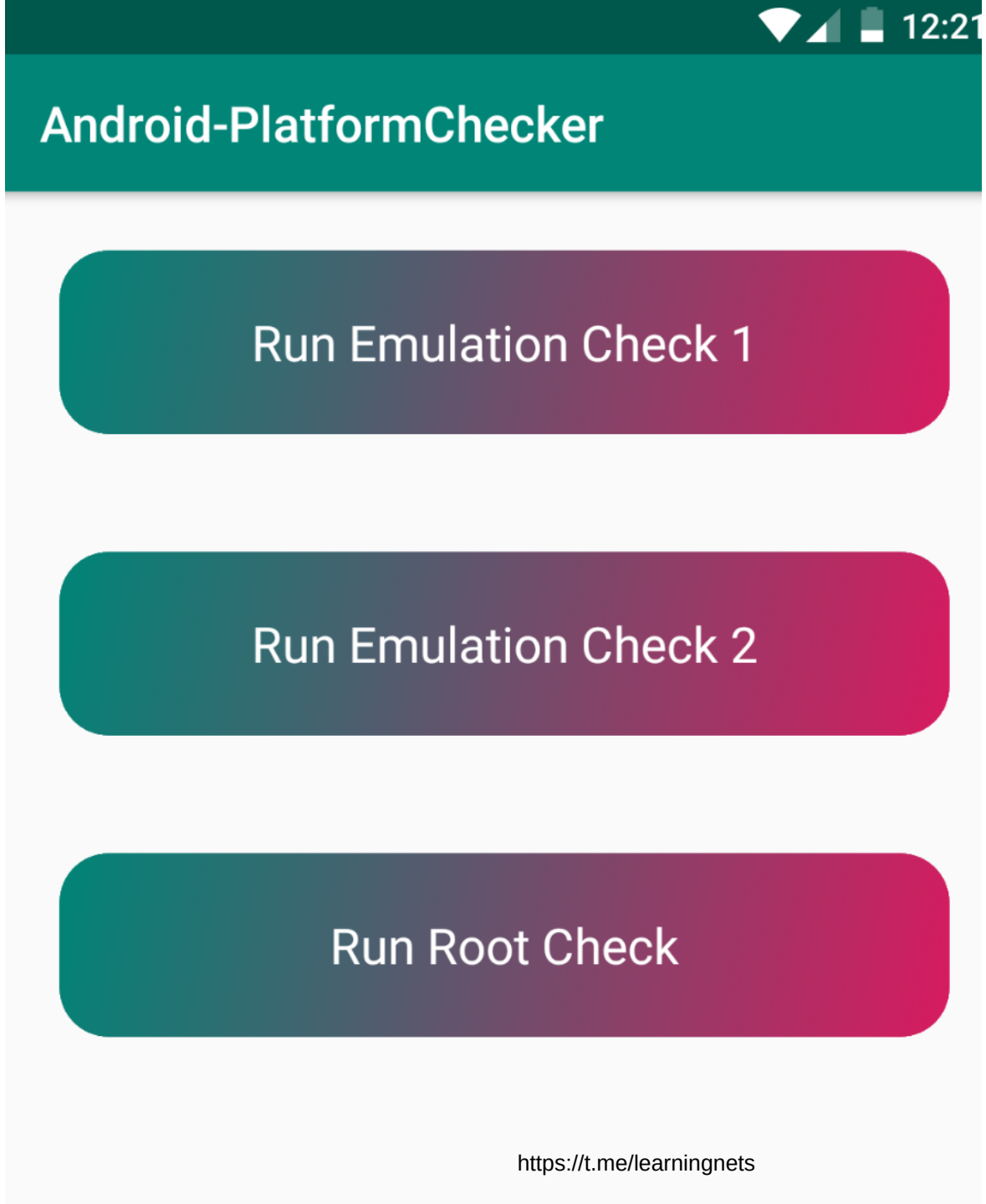
Using FRIDA – Show Return Value Hook Template

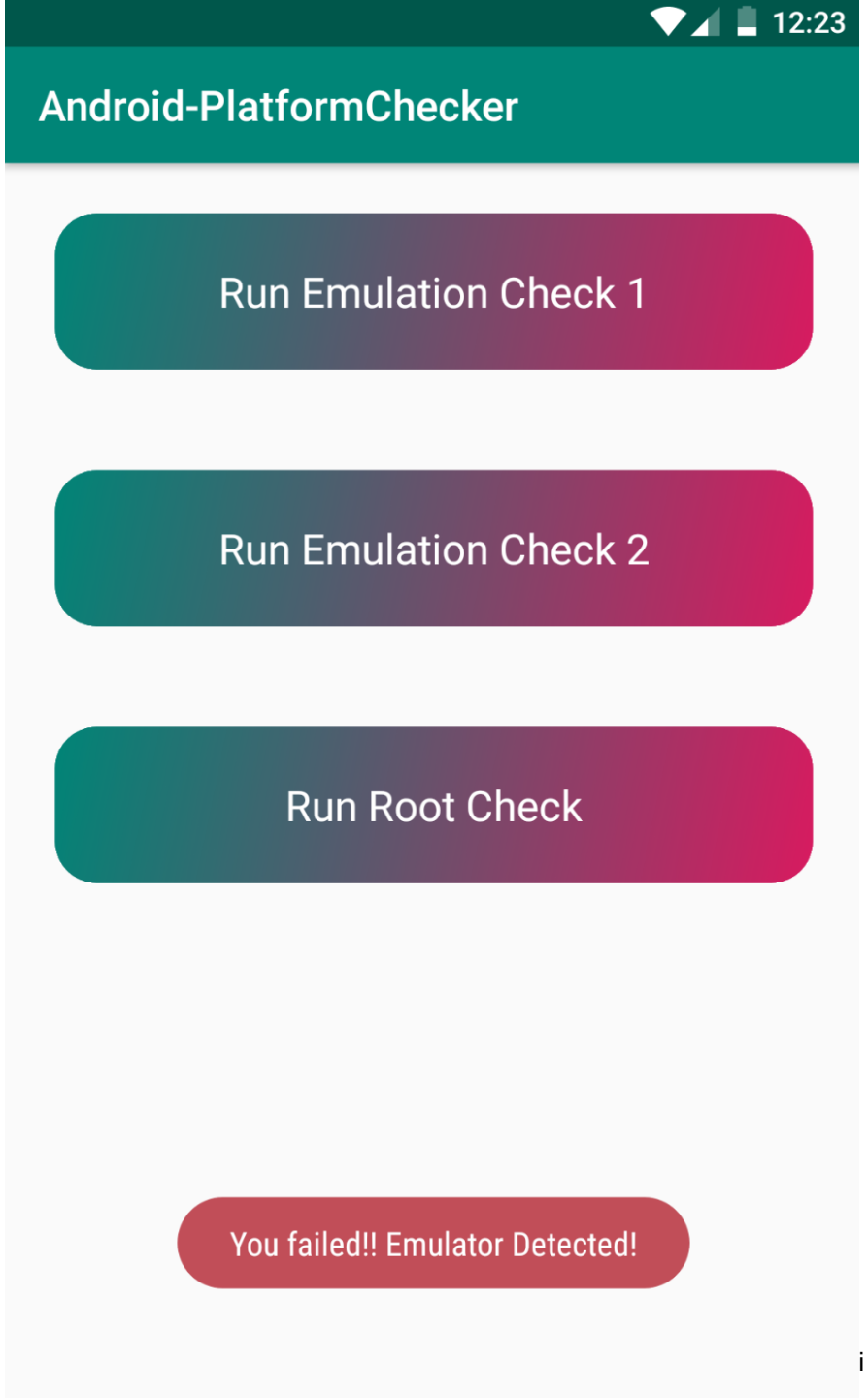
```
Java.perform(function () {
var MainActivity = Java.use("<PackageName.ActivityName>");

MainActivity.<hookedfunctionname>.implementation = function()
{
        console.log("In The Activity");
        var returnValue = this.<hookedfunctionname>();
        console.log("\nOriginal Return Value=",returnValue);
        return returnValue;
};
});
```

Using FRIDA – Show Return Value Hook PlatformChecker

- Let us Reverse Engineer the Application and find out the functions performing the Emulator Detection

- DEMO





```java
RoundedButton emulationCheckButton1 = (RoundedButton) findViewById(R.id.checkEmulationStatus1);
emulationCheckButton1.setOnClickListener( new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        String isEmulator = checkIfDeviceIsEmulator1();

        if(isEmulator.trim().equals("yes"))
        {
            StyleableToast.makeText(getApplicationContext(), text: "You failed!! Emulator Detected!", Toast.LENGTH_LONG
        }
        else if(isEmulator.trim().equals("no")) {
            StyleableToast.makeText(getApplicationContext(), text: "Congratulations!! No Emulator Detected!", Toast.LEN
        } else{
            StyleableToast.makeText(getApplicationContext(), text: "Something went wrong!!", Toast.LENGTH_LONG, R.style
        }
    }
});
```

```java
public String checkIfDeviceIsEmulator1() {
    //Normal function String returnType & No arguments

    if(Build.FINGERPRINT.contains("generic")
            || Build.FINGERPRINT.startsWith("unknown")
            || Build.MODEL.contains("google_sdk")
            || Build.MODEL.contains("Emulator")
            || Build.MODEL.contains("Android SDK built for x86")
            || Build.MODEL.contains("sdk")
            || Build.MODEL.contains("x86")
            || Build.MODEL.contains("unknown")
            || Build.MANUFACTURER.contains("Genymotion")
            || Build.FINGERPRINT.contains("test-keys")
            || Build.PRODUCT.contains("vbox86p")
            || (Build.BRAND.startsWith("generic") && Build.DEVICE.startsWith("generic"))
            || "google_sdk".equals(Build.PRODUCT))
    {

        return "yes";
    }else{
        return "no";
```

Using FRIDA – Show Return Value Hook PlatformChecker

```
console.log("<details of the script>");

Java.perform(function () {
var MainActivity = Java.use("com.dns.platformchecker.MainCheckerActivity");


MainActivity.checkIfDeviceIsEmulator1.implementation = function() {
    console.log();
    console.log("MainActivity.checkIfDeviceIsEmulator1 called");
    var returnValue = this.checkIfDeviceIsEmulator1();
        console.log("\nOriginal Return Value checkIfDeviceIsEmulator1 =",returnValue);
        return returnValue;
};

});
console.log("Done!");
```

```
frida -U -l show-return-value-platformchecker.js
"Android-PlatformChecker"
```

```
➜  4-showreturnvalue frida -U -l show-return-value-platformchecker.js  com.dns.platformchecker

     ____
    / _  |   Frida 12.6.10 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at http://www.frida.re/docs/home/
Attaching...
<details of the script>
Done!
[Genymotion Google::com.dns.platformchecker]->
MainActivity.checkIfDeviceIsEmulator1 called

Original Return Value checkIfDeviceIsEmulator1 = yes
```

/home/mobile/Desktop/vulnapps/frida_labs/4-showreturnvalue/show-return-value-platformchecker.js

FRIDA

Change Return Value Hook

Using FRIDA – Change Return Value Hook PlatformChecker

- Let us Bypass the Emulator Detection Check by modifying the return value

- DEMO

Remember Show Return Value Hook Template?

```
Java.perform(function () {
var MainActivity = Java.use("<PackageName.ActivityName>");


MainActivity.<hookedfunctionname>.implementation = function()
{
        console.log("In The Activity");
        var returnValue = this.<hookedfunctionname>();
        console.log("\nOriginal Return Value=",returnValue);
        return returnValue;
};
});
```

New Change Return Value Hook Template

```
Java.perform(function () {
var MainActivity = Java.use("<PackageName.ActivityName>");
MainActivity.<hookedfunctionname>.implementation = function() {
        console.log("In The Activity");
        var returnValue = this.<hookedfunctionname>();
        console.log("\nOriginal Return Value=",returnValue);
        var newReturnValue = <newvalue>;
        console.log("\nNew Return Value=",newReturnValue);
        return newReturnValue;
};});
```

Using FRIDA – Change Return Value Hook PlatformChecker

```
console.log("<details of the script>");

Java.perform(function () {
var MainActivity = Java.use("com.dns.platformchecker.MainCheckerActivity");


MainActivity.checkIfDeviceIsEmulator1.implementation = function() {
    console.log();
    console.log("MainActivity.checkIfDeviceIsEmulator1 called");
    var returnValue = this.checkIfDeviceIsEmulator1();
        console.log("\nOriginal Return Value checkIfDeviceIsEmulator1 =",returnValue);
        var newReturnValue = "no";
        console.log("\nNew Return Value =",newReturnValue);
        return newReturnValue;
};

});
console.log("Done!");
```

/home/mobile/Desktop/vulnapps/frida_labs/5-changereturnvalue/change-return-value-platformchecker.js

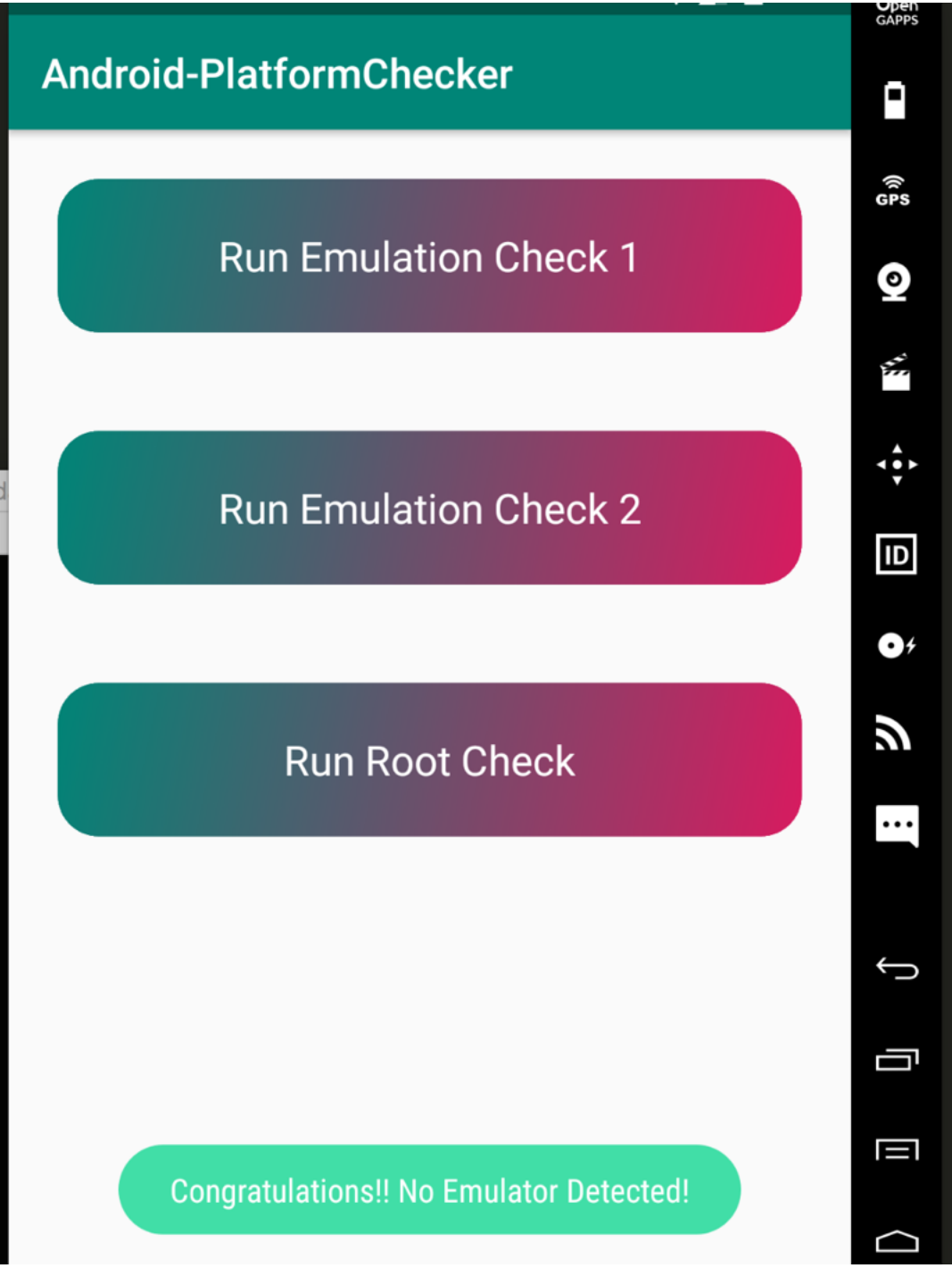
```
console.log("<details of the script>");

Java.perform(function () {
var MainActivity = Java.use("com.dns.platformchecker.MainCheckerActivity");


MainActivity.checkIfDeviceIsEmulator1.implementation = function() {
    console.log();
    console.log("MainActivity.checkIfDeviceIsEmulator1 called");
    var returnValue = this.checkIfDeviceIsEmulator1();
        console.log("\nOriginal Return Value checkIfDeviceIsEmulator1 =",returnValue);
        var newReturnValue = "no";
        console.log("\nNew Return Value =",newReturnValue);
        return newReturnValue;
};

});
console.log("Done!");
```

5-changereturnvalue — frida -U -l change-return-value-platformchecker.js com.dns.platformchecker — frid

..1-dumpclasses

```
[➜ 5-changereturnvalue frida -U -l change-return-value-platformchecker.js  com.dns.platformchecker

    ____
   / _  |   Frida 12.6.10 - A world-class dynamic instrumentation toolkit
  | (_| |
   > _  |   Commands:
  /_/ |_|       help      -> Displays the help system
  . . . .       object?   -> Display information about 'object'
  . . . .       exit/quit -> Exit
  . . . .
  . . . .   More info at http://www.frida.re/docs/home/
Attaching...
<details of the script>
Done!
[Genymotion Google::com.dns.platformchecker]->
MainActivity.checkIfDeviceIsEmulator1 called

Original Return Value checkIfDeviceIsEmulator1 = yes

New Return Value = no
```

```
frida -U -l change-return-value-platformchecker.js "Android-PlatformChecker"
```

TASK - 1

- List the classes from the installed RootChecker application using Jadx-Gui
- List functions from RootChecker application.
  - You can use the script /home/mobile/Desktop/vulnapps/frida_labs/2-dumpallmethods/list_all_methods_specific_class_rootchecker.js

- What functions handles the device emulator check in the RootChecker application?

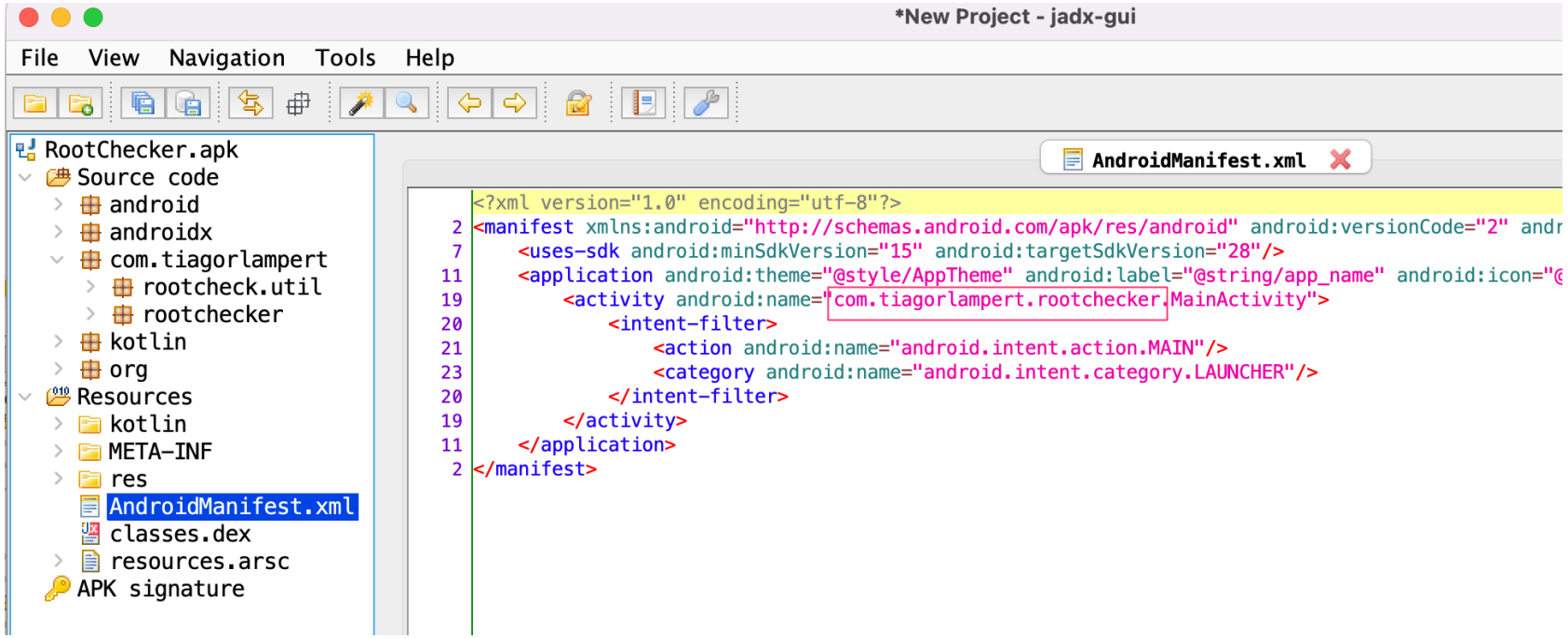
- Use the Function Hook to display the return value of those functions
  - The script is at /home/mobile/Desktop/vulnapps/frida_labs/4-showreturnvalue/show-return-value-rootchecker.js

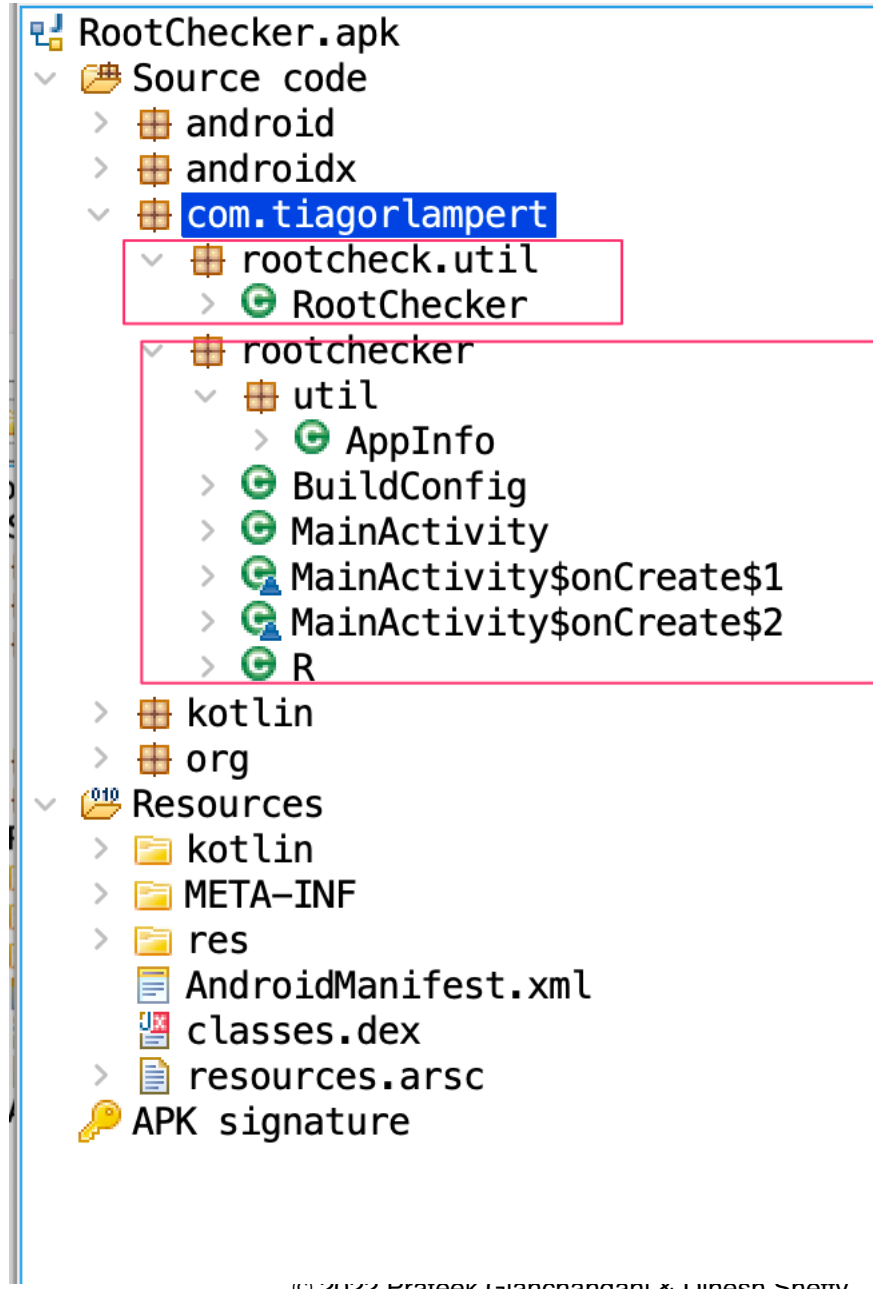
TASK - 2

- Ensure that the RootChecker applications always shows that the device is not rooted.
  - The script is at /home/mobile/Desktop/vulnapps/frida_labs/5-changereturnvalue/change-return-value-rootchecker.js

TASK - 1 - Solution

Solution - List of classes





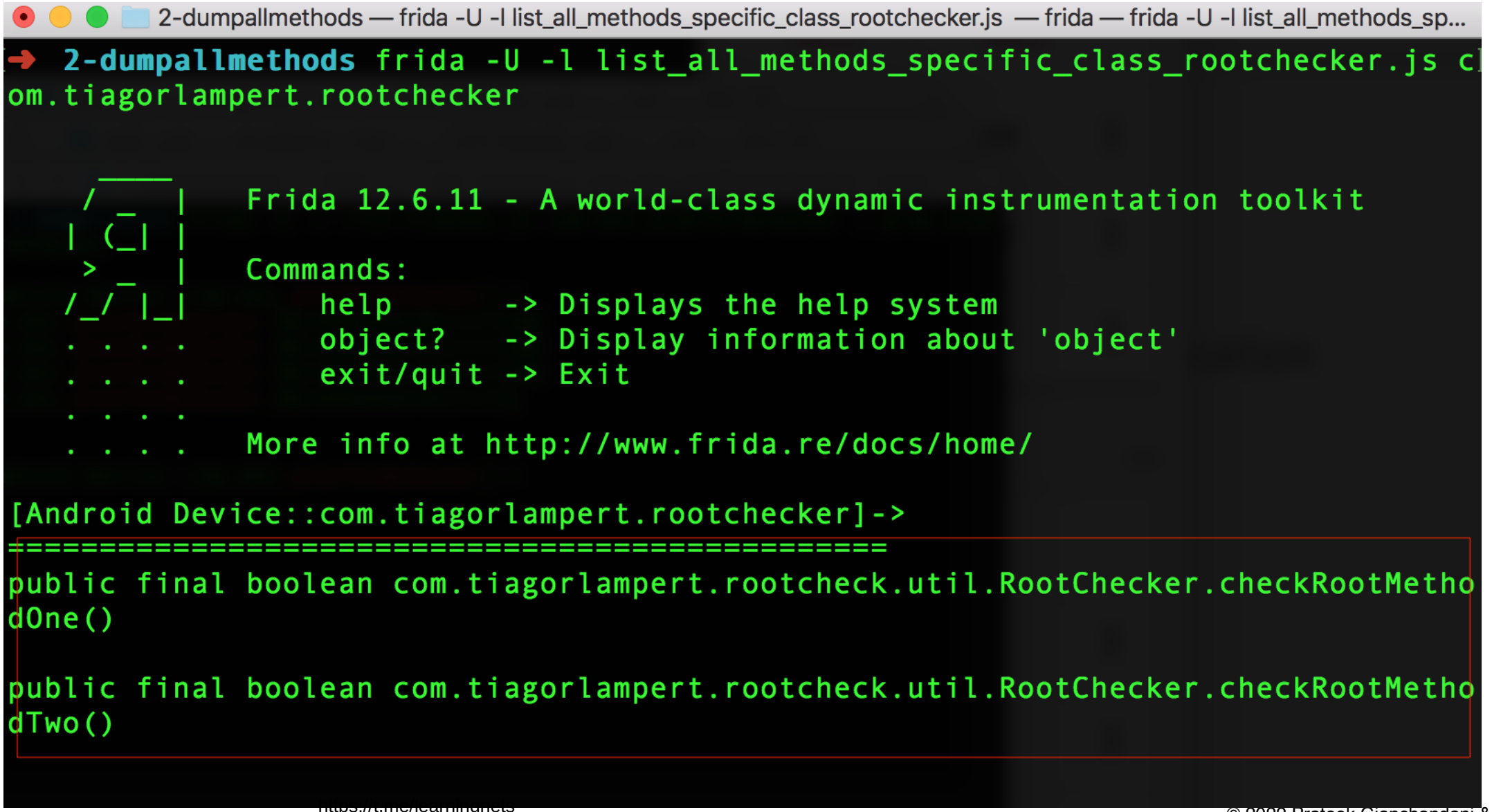
Solution - List of methods

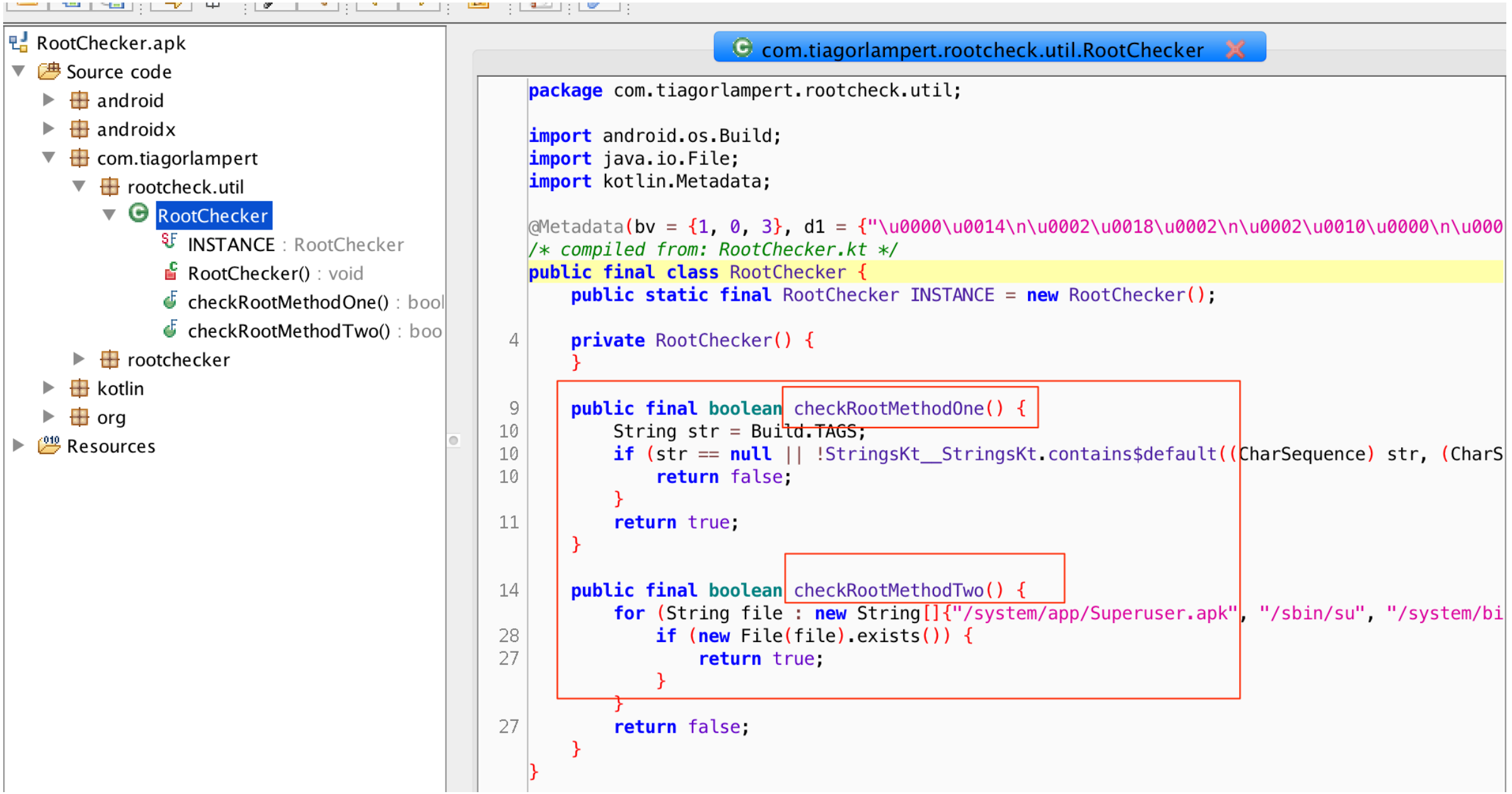
- List the methods from the installed RootChecker application

Using FRIDA – Dump Methods - RootChecker

```
function listClassMethods(classNameForSearch) {
    var hook = Java.use(classNameForSearch);
    var declaredMethods = hook.class.getDeclaredMethods();
    hook.$dispose;
    return declaredMethods;
}

setTimeout(function() {
    if (Java.available) {
        Java.perform(function() {
            //Change the values for class_name_for_search as needed
            var class_name_for_search = 'com.tiagorlampert.rootcheck.util.RootChecker';

            var all_methods_in_class = listClassMethods(class_name_for_search)
            var methods_return = ''

            all_methods_in_class.forEach(function(args) {
                methods_return += String(args) + '\n\n'
            });

            console.log("\n================================================");
            console.log(methods_return);
            console.log("\n================================================");

        })
    }
}, 0);
```

```
frida -U -l list_all_methods_specific_class_rootchecker.js "Root Checker"
```





TASK - Solution

```
console.log("<details of the script>");

Java.perform(function () {
var MainActivity = Java.use("com.tiagorlampert.rootcheck.util.RootChecker");


MainActivity.checkRootMethodOne.implementation = function() {
    console.log();
    console.log("MainActivity.checkRootMethodOne called");
    var returnValue = this.checkRootMethodOne();
        console.log("\nOriginal Return Value checkRootMethodOne =",returnValue);
        return returnValue;
};

MainActivity.checkRootMethodTwo.implementation = function() {
    console.log();
    console.log("MainActivity.checkRootMethodTwo called");
    var returnValue = this.checkRootMethodTwo();
        console.log("\nOriginal Return Value checkRootMethodTwo =",returnValue);
        return returnValue;
};

});
console.log("Done!");
```

TASK - Solution

```
frida -U -l show-return-value-rootchecker.js "Root Checker"
```

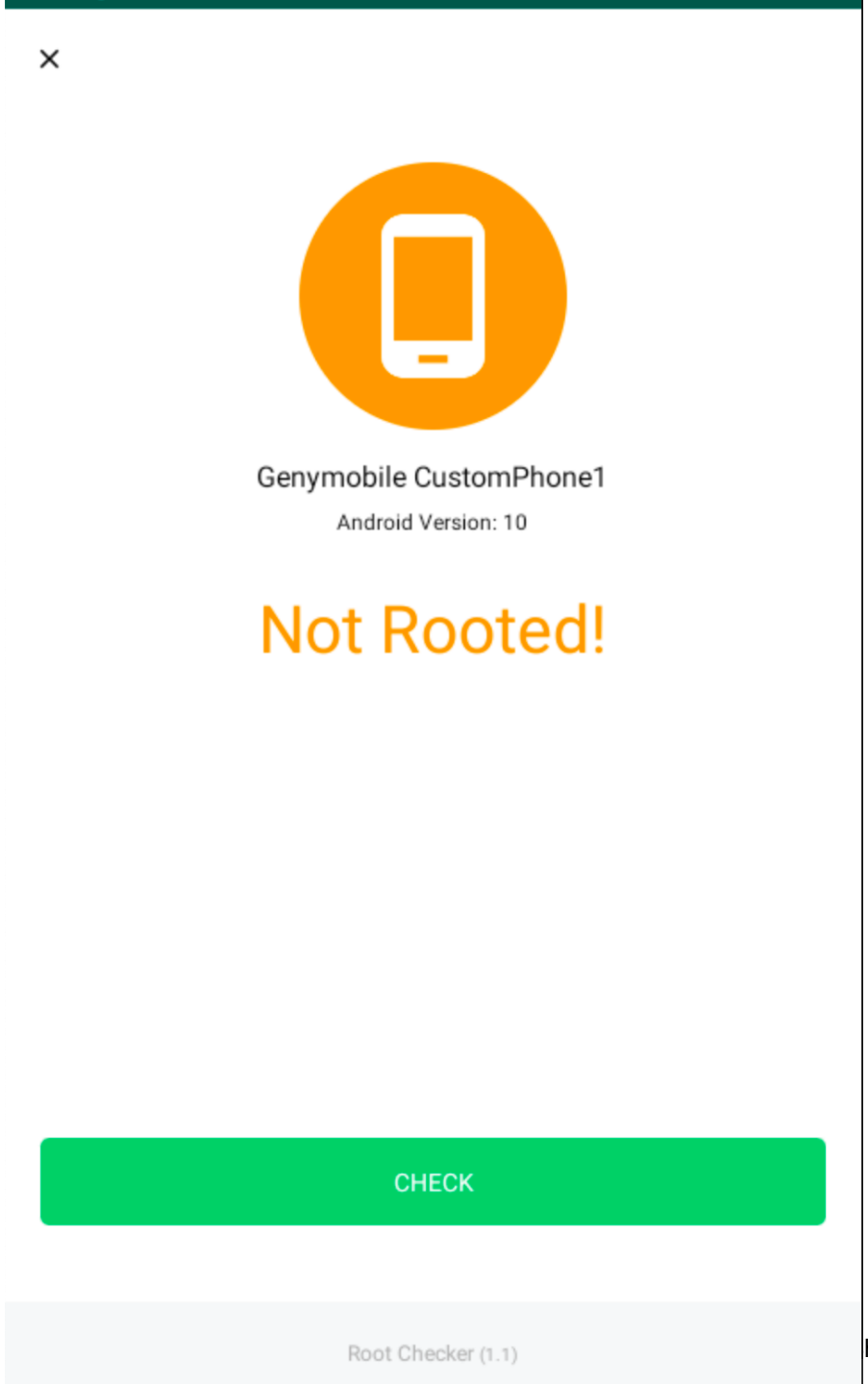
```
➜ 4-showreturnvalue frida -U -l show-return-value-rootchecker.js com.tiagorlampert.rootchecker
     ____
    / _  |   Frida 12.6.10 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at http://www.frida.re/docs/home/
Attaching...
<details of the script>
Done!
[Genymotion Google::com.tiagorlampert.rootchecker]->
MainActivity.checkRootMethodOne called

Original Return Value checkRootMethodOne = true

MainActivity.checkRootMethodTwo called

Original Return Value checkRootMethodTwo = true
```

TASK - 2 - Solution

Using FRIDA – Change Return Value Hook RootChecker

```
console.log("<details of the script>");

Java.perform(function () {
var MainActivity = Java.use("com.tiagorlampert.rootcheck.util.RootChecker");


MainActivity.checkRootMethodOne.implementation = function() {
    console.log();
    console.log("MainActivity.checkRootMethodOne called");
    var returnValue = this.checkRootMethodOne();
        console.log("\nOriginal Return Value checkRootMethodOne =",returnValue);
        var newReturnValue = false;
        console.log("\nNew Return Value=",newReturnValue);
    return newReturnValue;
};

MainActivity.checkRootMethodTwo.implementation = function() {
    console.log();
    console.log("MainActivity.checkRootMethodTwo called");
    var returnValue = this.checkRootMethodTwo();
        console.log("\nOriginal Return Value checkRootMethodTwo =",returnValue);
        var newReturnValue = false;
        console.log("\nNew Return Value=",newReturnValue);
    return newReturnValue;
};

});
console.log("Done!");
```

```
frida -U -l change-return-value-
rootchecker.js "Root Checker"
```



Fancy Method Tracing?

- Welcome Method Info Extractor scripts

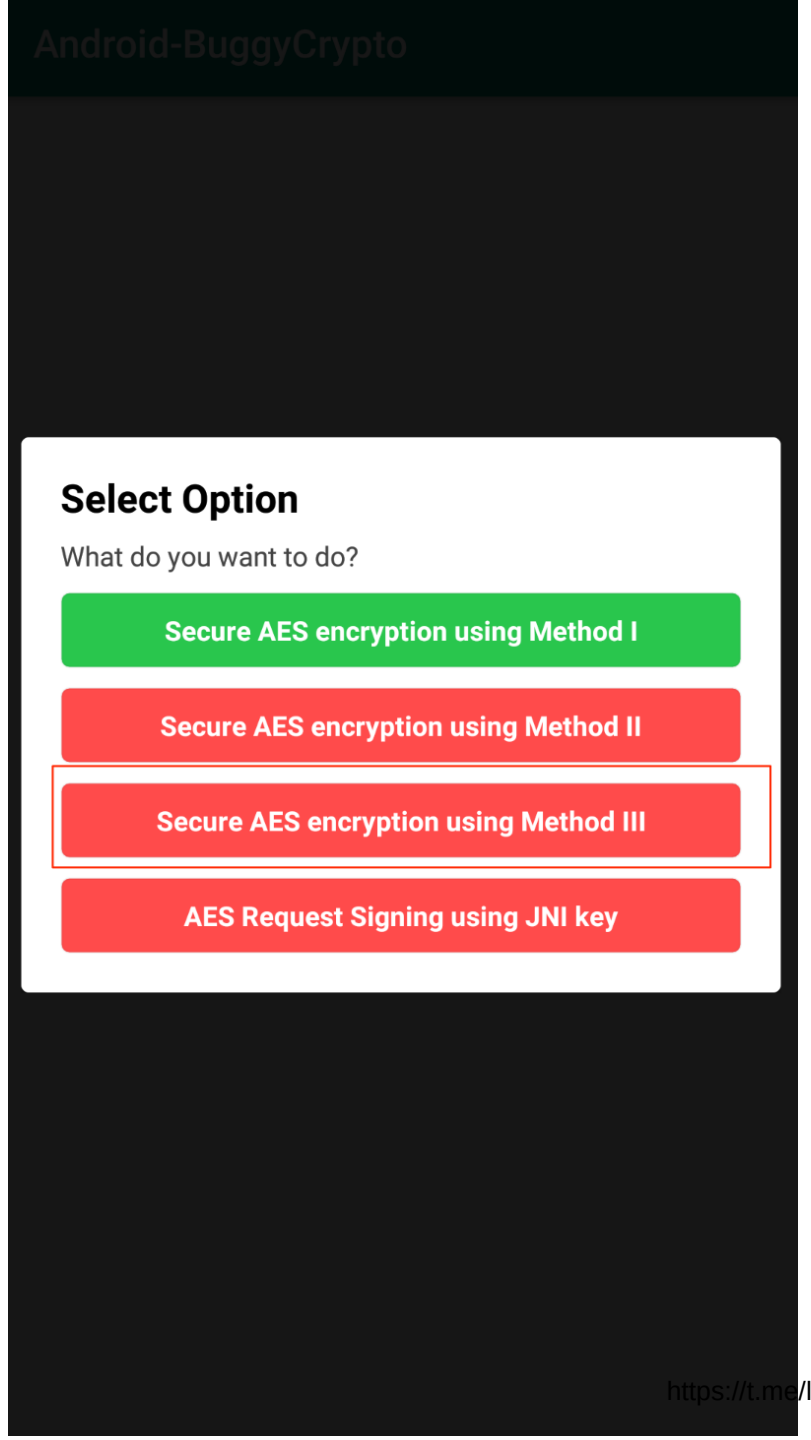
- Location – /home/mobile/Desktop/vulnapps/frida_labs/templates/show_specific_function_info.js

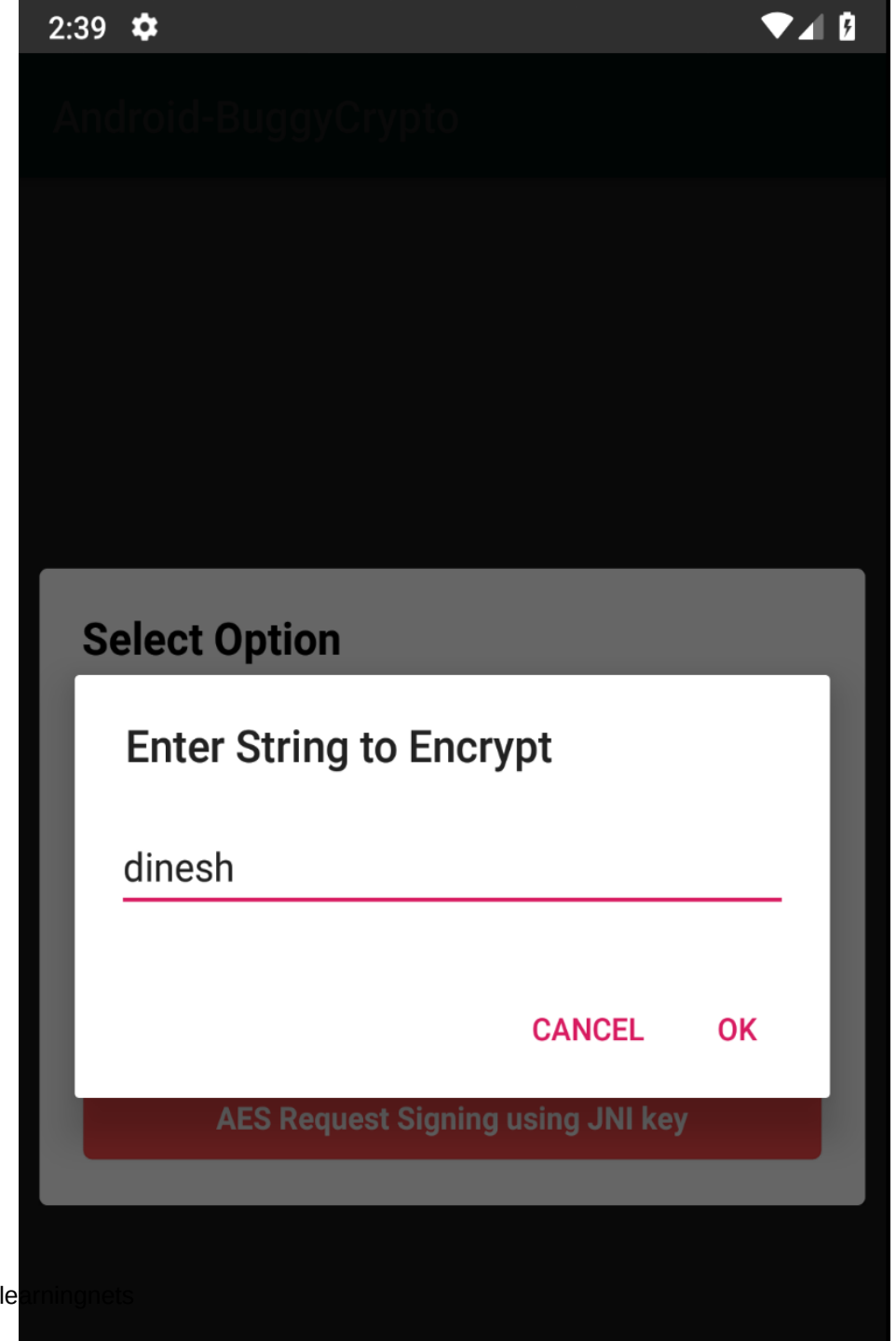
```
[➜  templates cat show_specific_function_info.js
var java_class_thread = null
var linebreak = " ---------> "

setTimeout(function() {
    Java.perform(function() {
        console.log("Entering code now!!!\n")
        var java_class_thread = Java.use("java.lang.Thread");
        var target_class_name = "<<class-name>>";
        var target_function_name = "<<function-name>>"
                var class_hook = Java.use(target_class_name);
                var class_all_function_hook_overloads = eval("class_hook."+target_function_name+".overloads");
                var function_hook_overloads_count = class_all_function_hook_overloads.length;
                var class_specified_function_hook = null
                console.log("Number of function hooks = "+function_hook_overloads_count.toString())

                for (var i = 0; i < function_hook_overloads_count; i++) {
                        class_specified_function_hook = eval('class_hook.'+target_function_name+'.overloads[i]')
                    class_specified_function_hook.implementation = function () {
                        var trace_flow = ''
                        var call_flow = ''
                        var arguments_list_all = ''
                        var argument_type = ''
                        var retval = ''
                        var ret_type = ''

                        for (var index = 0; index < arguments.length; index++) {
                                argument_type += ('argumentType' + index.toString() + " : " + String(typeof(arguments[index])) + ' '
)
                                arguments_list_all += ("argument" + index.toString() + ": " + String(arguments[index]))
                        }

                        call_flow = java_class_thread.currentThread().getStackTrace().slice(2,5).reverse().toString().replace(/,/g,l
inebreak);
                        trace_flow += call_flow + linebreak + class_specified_function_hook.methodName + '( ' + argument_type+ ') '

                        try {
                        retval = eval('this.'+target_function_name+'.apply(this, arguments)')
                    } catch (err) {
                        retval = null
                        console.log("Exception while handling retval!" + String(err))
                    }

                                ret_type = String(class_specified_function_hook.returnType['className'])
                        var retval_dump = "(" + ret_type + ') : ' + String(retval)

                        console.log("\n=======================")
                        console.log("argument type = "+argument_type)
                        console.log("argument value = "+arguments_list_all)
                        console.log("Return value = "+retval_dump)
                        console.log("Flow of data (reverse calling stack) = "+trace_flow)
                        console.log("=======================\n")

                        return retval;
                    }
                }

    });
}, 0);
```

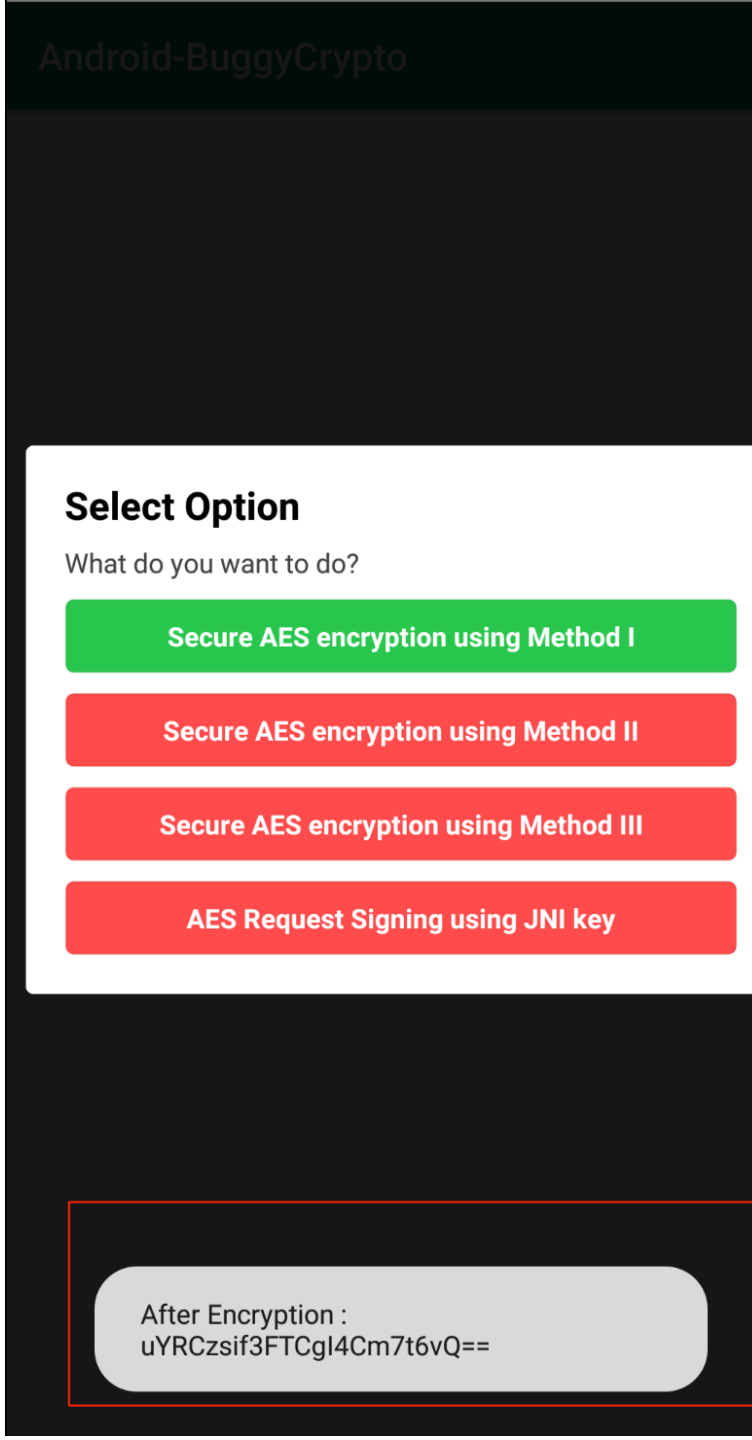
```
[➜ 3-showfunctioninformation cat show_specific_function_info_rootchecker.js
var java_class_thread = null
var linebreak = " ---------> "

setTimeout(function() {
    Java.perform(function() {
        console.log("Entering code now!!!\n")
        var java_class_thread = Java.use("java.lang.Thread");
        var target_class_name = "com.tiagorlampert.rootcheck.util.RootChecker";
        var target_function_name = "checkRootMethodTwo"
                var class_hook = Java.use(target_class_name);
                var class_all_function_hook_overloads = eval("class_hook."+target_function_name+".overloads");
                var function_hook_overloads_count = class_all_function_hook_overloads.length;
                var class_specified_function_hook = null
                console.log("Number of function hooks = "+function_hook_overloads_count.toString())

                for (var i = 0; i < function_hook_overloads_count; i++) {
                        class_specified_function_hook = eval('class_hook.'+target_function_name+'.overloads[i]')
                    class_specified_function_hook.implementation = function () {
                        var trace_flow = ''
                        var call_flow = ''
                        var arguments_list_all = ''
                        var argument_type = ''
                        var retval = ''
                        var ret_type = ''

                        for (var index = 0; index < arguments.length; index++) {
                                argument_type += ('argumentType' + index.toString() + " : " + String(typeof(arguments[index])) + ' '
)
                                arguments_list_all += ("argument" + index.toString() + ": " + String(arguments[index]))
                        }

                        call_flow = java_class_thread.currentThread().getStackTrace().slice(2,5).reverse().toString().replace(/,/g,l
inebreak);
                        trace_flow += call_flow + linebreak + class_specified_function_hook.methodName + '( ' + argument_type+ ') '

                        try {
                        retval = eval('this.'+target_function_name+'.apply(this, arguments)')
                    } catch (err) {
                        retval = null
                        console.log("Exception while handling retval!" + String(err))
                    }

                                ret_type = String(class_specified_function_hook.returnType['className'])
                        var retval_dump = "(" + ret_type + ') : ' + String(retval)

                        console.log("\n=======================")
                        console.log("argument type = "+argument_type)
                        console.log("argument value = "+arguments_list_all)
                        console.log("Return value = "+retval_dump)
                        console.log("Flow of data (reverse calling stack) = "+trace_flow)
                        console.log("=======================\n")

                        return retval;
                    }
                }
    });
}, 0);
```

```
➜ 3-showfunctioninformation frida -U -l show_specific_function_info_rootchecker
.js com.tiagorlampert.rootchecker

     ____
    / _  |   Frida 12.6.23 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at http://www.frida.re/docs/home/

[Android Device::com.tiagorlampert.rootchecker]-> Entering code now!!!

Number of function hooks = 1

=========================
argument type =
argument value =
Return value = (boolean) : true
Flow of data (reverse calling stack) = com.tiagorlampert.rootchecker.MainActivit
y$onCreate$1.onClick(MainActivity.kt:34) ----------> com.tiagorlampert.rootchecke
r.MainActivity.checkRoot(MainActivity.kt:56) ----------> com.tiagorlampert.rootch
eck.util.RootChecker.checkRootMethodTwo(Native Method) ----------> checkRootMetho
dTwo( )
=========================
```

REWIND

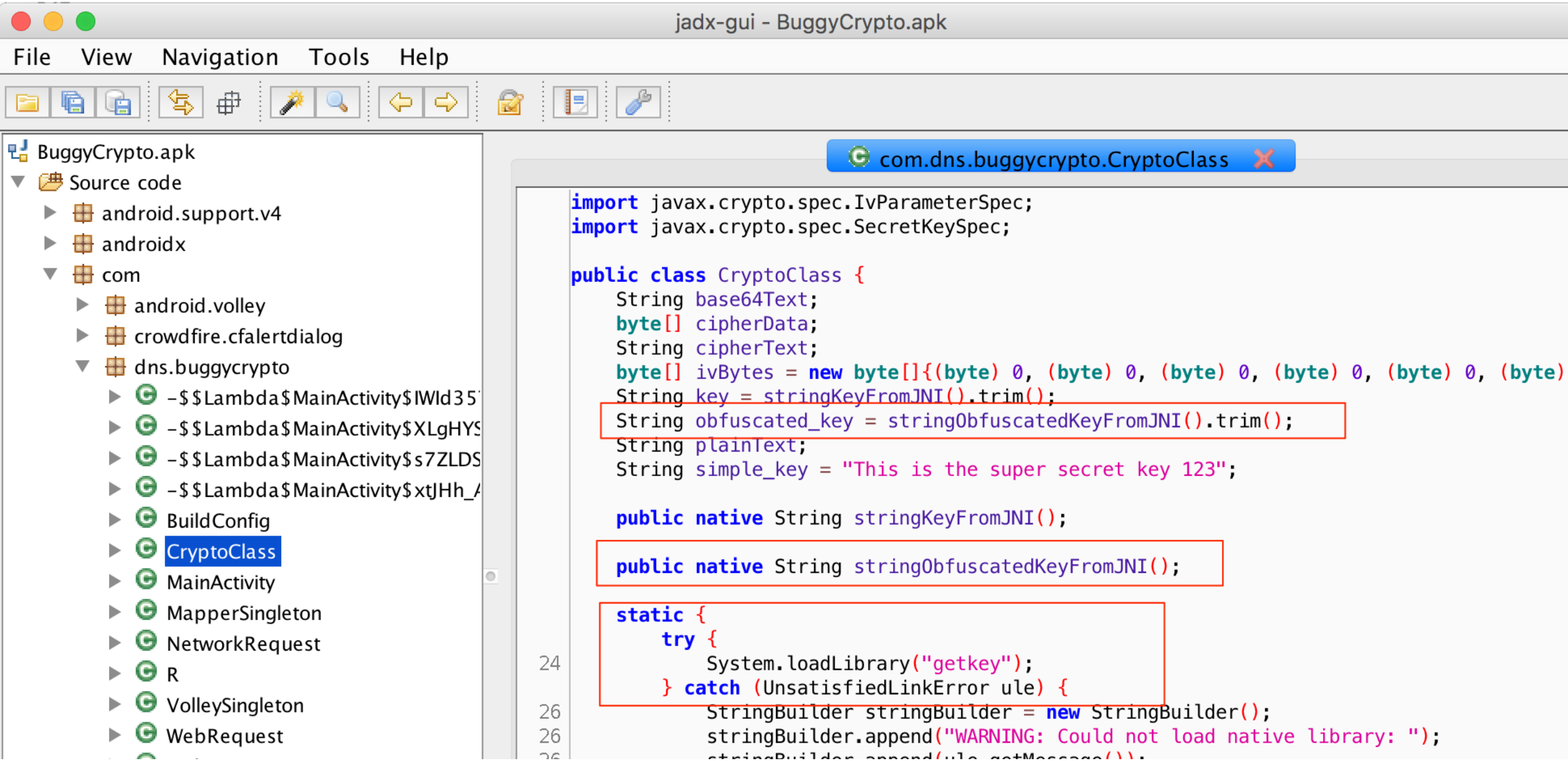






com.dns.buggycrypto.CryptoClass

```
105         this.cipherData = aes256decrypt(this.ivBytes, this.key.getBytes(str), Base64.decode(theString.
106         this.plainText = new String(this.cipherData, str);
107         return this.plainText;
        }

116     public String aesEncryptedString(String theString) throws UnsupportedEncodingException, InvalidKey
            String str = "UTF-8";
117         byte[] keyBytes = this.key.getBytes(str);
118         this.plainText = theString;
119         this.cipherData = aes256encrypt(this.ivBytes, keyBytes, this.plainText.getBytes(str));
120         this.cipherText = Base64.encodeToString(this.cipherData, 0);
121         return this.cipherText;
        }

126     public String simpleAesEncryptedString(String theString) throws UnsupportedEncodingException, Inva
            String str = "UTF-8";
127         byte[] keyBytes = this.simple_key.getBytes(str);
127         this.plainText = theString;
128         this.cipherData = aes256encrypt(this.ivBytes, keyBytes, this.plainText.getBytes(str));
129         this.cipherText = Base64.encodeToString(this.cipherData, 0);
130         return this.cipherText;
131     }

137     public String obfuscatedAesEncryptedString(String theString) throws UnsupportedEncodingException,
            String str = "UTF-8";
138         byte[] keyBytes = this.obfuscated_key.getBytes(str);
139         this.plainText = theString;
140         this.cipherData = aes256encrypt(this.ivBytes, keyBytes, this.plainText.getBytes(str));
141         this.cipherText = Base64.encodeToString(this.cipherData, 0);
142         return this.cipherText;
        }
}
```



What do we hook?

- package_name = com.dns.buggycrypto
- target_class_name = com.dns.buggycrypto.CryptoClass
- target_function_name = stringObfuscatedKeyFromJNI()

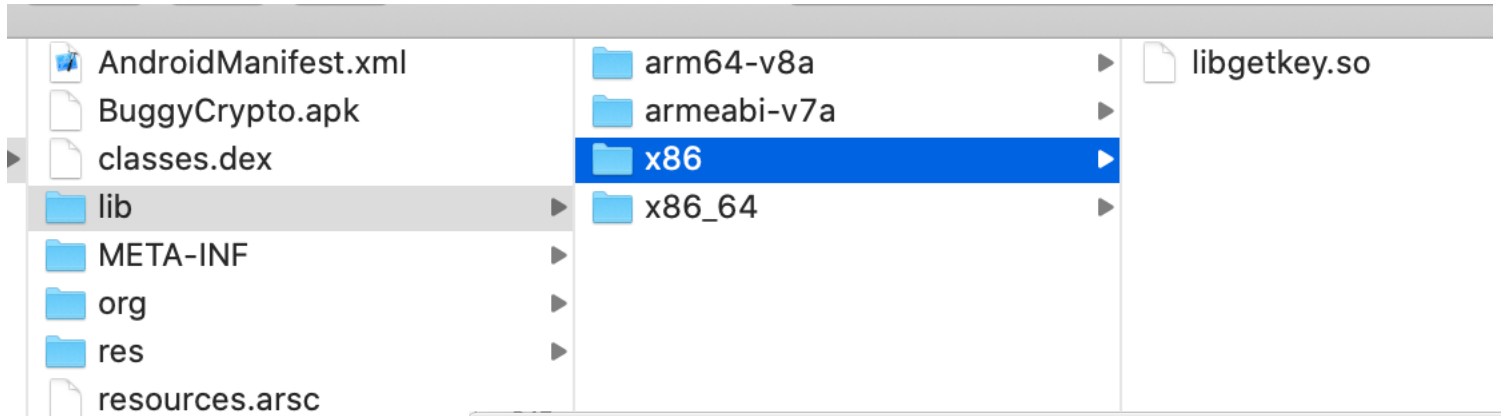
/home/mobile/Desktop/vulnapps/frida_labs/buggycrypto_method3/show-return-value-buggycrypto-method.js

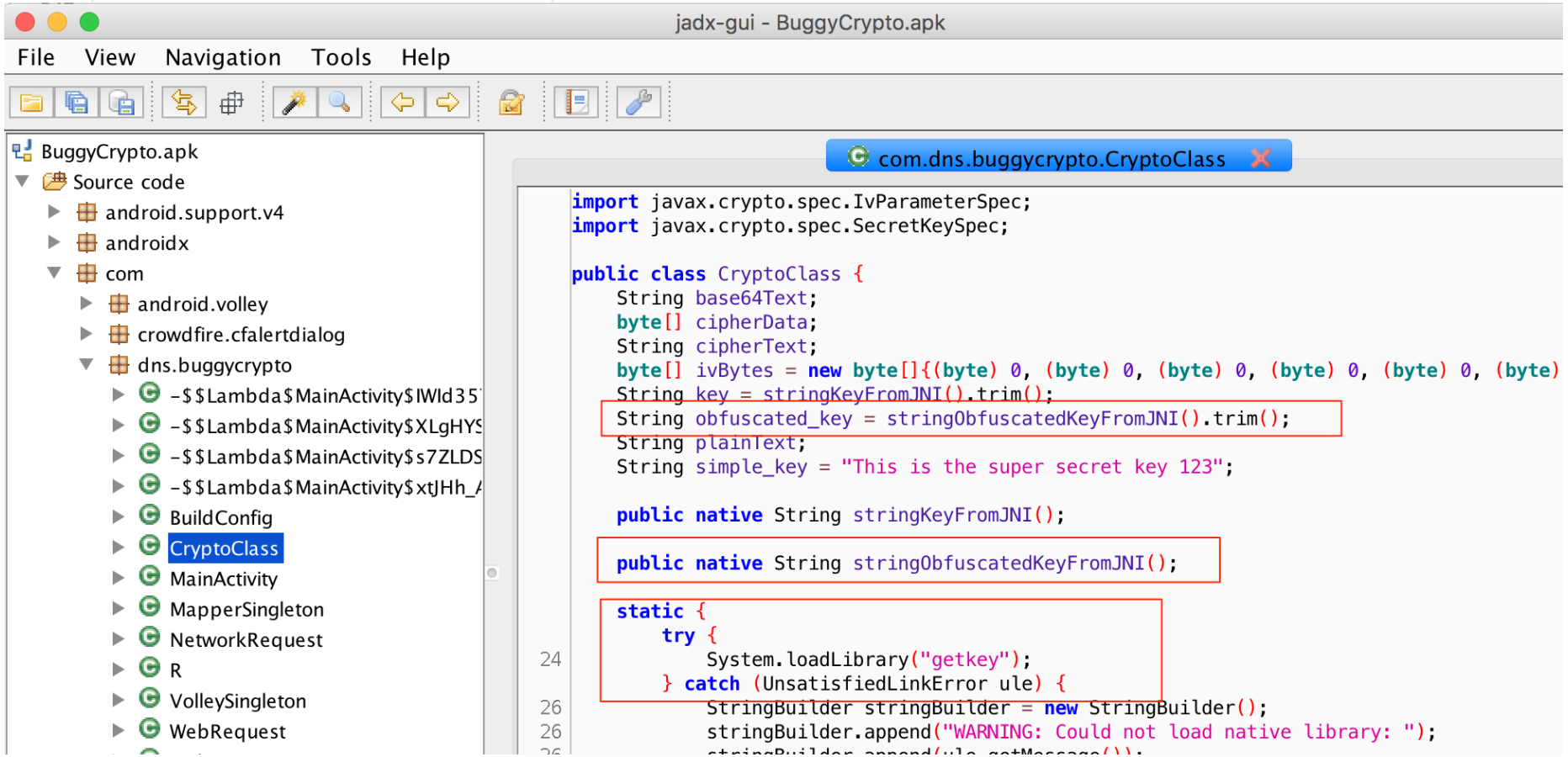
```
console.log("View return value from stringObfuscatedKeyFromJNI");

Java.perform(function () {
var MainActivity = Java.use("com.dns.buggycrypto.CryptoClass");


MainActivity.stringObfuscatedKeyFromJNI.implementation = function() {
    console.log();
    console.log("com.dns.buggycrypto.CryptoClass.stringObfuscatedKeyFromJNI() called");
    var returnValue = this.stringObfuscatedKeyFromJNI();
        console.log("\nOriginal Return Value stringObfuscatedKeyFromJNI =",returnValue);
        return returnValue;
};

});
console.log("Done!")
```

```
[➜  buggycrypto_method3 frida -U --no-pause com.dns.buggycrypto -l show-return-value-buggycrypto-method.js
     ____
    / _  |   Frida 12.10.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://www.frida.re/docs/home/
Attaching...
View return value from stringObfuscatedKeyFromJNI
Done!
[CustomPhone1::com.dns.buggycrypto]->
com.dns.buggycrypto.CryptoClass.stringObfuscatedKeyFromJNI() called

Original Return Value stringObfuscatedKeyFromJNI = th0ush4ltn0tde0bfusc4tethis0kpls
[CustomPhone1::com.dns.buggycrypto]->
```

ALTERNATE APPROACH

- Exploiting JNI





- `Process.enumerateModules()`: enumerates modules loaded right now, returning an array of Module objects.

Module

Objects returned by e.g. `Module.load()` and `Process.enumerateModules()`.

- `enumerateExports()`: enumerates exports of module, returning an array of objects containing the following properties:
  - `type`: string specifying either `function` or `variable`
  - `name`: export name as a string
  - `address`: absolute address as a `NativePointer`

- `Process.findModuleByAddress(address)`, `Process.getModuleByAddress(address)`, `Process.findModuleByName(name)`, `Process.getModuleByName(name)`: returns a Module whose *address* or *name* matches the one specified. In the event that no such module could be found, the *find*-prefixed functions return *null* whilst the *get*-prefixed functions throw an exception.

Interceptor

- `Interceptor.attach(target, callbacks[, data])`: intercept calls to function at `target`. This is a `NativePointer` specifying the address of the function you would like to intercept calls to. Note that on 32-bit ARM this address must have its least significant bit set to 0 for ARM functions, and 1 for Thumb functions. Frida takes care of this detail for you if you get the address from a Frida API (for example `Module.getExportByName()`).
  The `callbacks` argument is an object containing one or more of:
  - `onEnter: function (args)`: callback function given one argument `args` that can be used to read or write arguments as an array of `NativePointer` objects.
  - `onLeave: function (retval)`: callback function given one argument `retval` that is a `NativePointer`-derived object containing the raw return value. You may call `retval.replace(1337)` to replace the return value with the integer `1337`, or `retval.replace(ptr("0x1234"))` to replace with a pointer. Note that this object is recycled across *onLeave* calls, so do not store and use it outside your callback. Make a deep copy if you need to store the contained value, e.g.: `ptr(retval.toString())`.

```
//frida -U --no-pause com.dns.buggycrypto -l sniff_jni_data_method3.js
console.log("Architecture Hooked -> " + Process.arch);
setTimeout(function() {
    if (Java.available) {
        console.log("Sniff specified JNI function\n");
        var soLibName = "libgetkey"; //change-me
        var targetFunctionName = "
            Java_com_dns_buggycrypto_CryptoClass_stringObfuscatedKeyFromJNI"; //change-me
        console.log("Current target = "+targetFunctionName+" inside "+soLibName);
        Java.perform(function() {
            Process.enumerateModules().filter(function(m) {
                    //console.log(m["path"]);
                    return m["path"].toLowerCase().indexOf(soLibName) != -1;
                })
                .forEach(function(mod) {
                    mod.enumerateExports().forEach(function(exp) {
                        // console.log(exp.name);
                        //replace below with "JAVA" to hookall
                        if (exp.name.indexOf(targetFunctionName) != -1) {
                            console.log(exp.name + " was found at " + exp.address);
                            // console.log("exp.address = " + exp.address);
                            // console.log(mod[1]);
                            Interceptor.attach(exp.address, {

                                onEnter: function(args) {
                                    console.log("inside attach");
                                    // // Verify argument types if script crashes
```

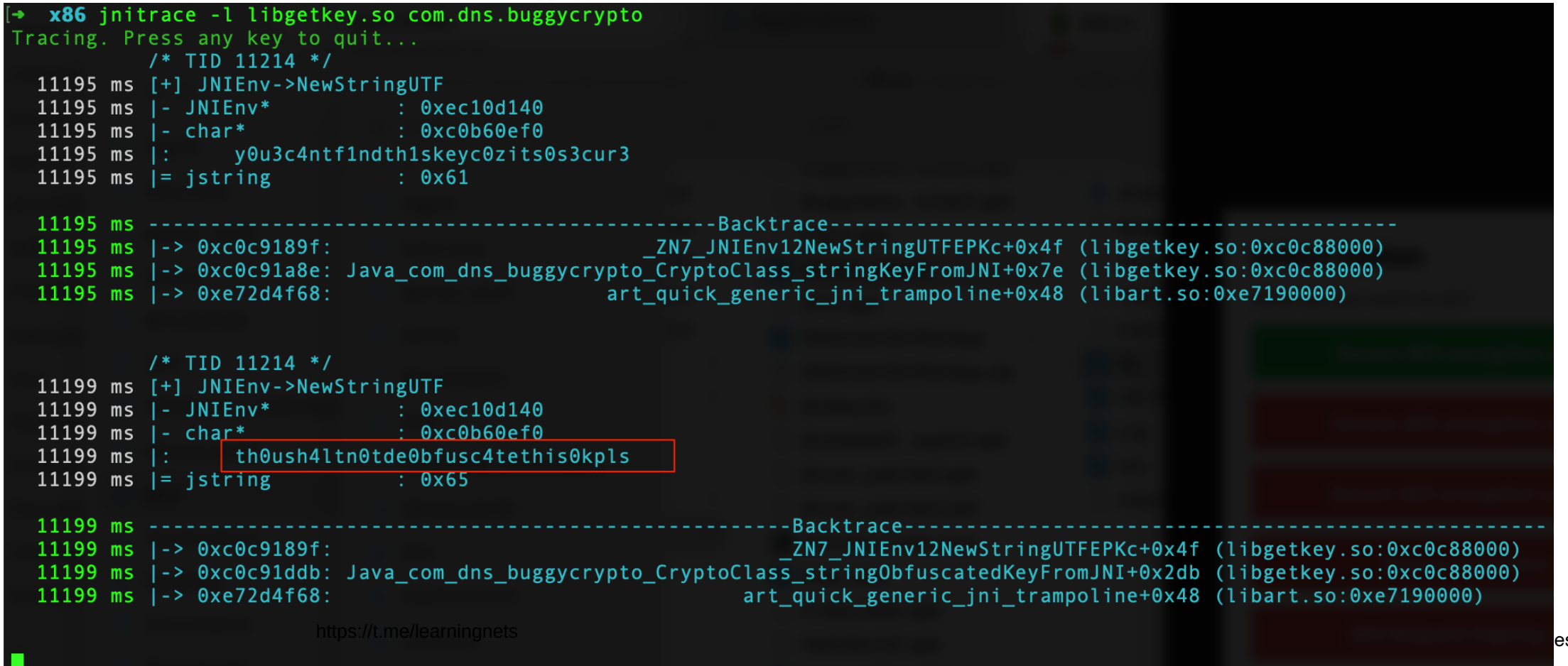
```
Interceptor.attach(exp.address, {

    onEnter: function(args) {
        console.log("inside attach");
    // // Verify argument types if script crashes

    //      var String = Java.use("java.lang.String");
    //      var stringArg = Java.cast(ptr(args[2]), String);
    //      console.log("Value of argument2 = " + stringArg);
    //       // console.log("[*] Argument 2 = ", JSON.stringify(
    stringArg.toString(), null,'\t'));

    //      var intArg = parseInt(args[3]);
    //      console.log("Value of argument3 = " + intArg);

     },
    onLeave: function(retValue) {
        // Verify retValue datatype if script crashes
        console.log("inside onleave");
        var String = Java.use("java.lang.String");
        var returnValueString = Java.cast(ptr(retValue), String);
        console.log("Return value = " + returnValueString);
     }
});
```

```
➜  exploit_jni_working frida -U --no-pause com.dns.buggycrypto -l sniff_jni_data
_method3.js
     ____
    / _  |   Frida 12.10.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://www.frida.re/docs/home/
Attaching...
Architecture Hooked -> ia32
Sniff specified JNI function

Current target = Java_com_dns_buggycrypto_CryptoClass_stringObfuscatedKeyFromJNI
 inside libgetkey
Java_com_dns_buggycrypto_CryptoClass_stringObfuscatedKeyFromJNI was found at 0xc
f18eb00
[CustomPhone1::com.dns.buggycrypto]-> inside attach
inside onleave
Return value = th0ush4ltn0tde0bfusc4tethis0kpls
```

/home/mobile/Desktop/vulnapps/frida_labs/buggycrypto_method3/sniff_jni_data_method3.js

JNI SNIFFING MADE EASY

- Jnitrace
  - https://github.com/chame1eon/jnitrace



Frida – NonRooted Android

- Run `adb shell getprop ro.product.cpu.abi` to get the version
  - In our case it is x86 -> frida-server-<version>-android-arm.xz
- `apktool d diva-android-debug.apk -o diva`
- Download frida-gadget-<version>-android-x86.so.xz from https://github.com/frida/frida/releases
- `unxz frida-gadget-<version>-android-x86.so.xz`
- `mkdir diva/lib/`

Frida – NonRooted Android

- `mkdir diva/lib/x86`
- `cp frida-gadget-12.6.11-android-x86.so diva/lib/x86/libfrida-gadget.so`
- Inject System.loadLibrary("frida-gadget") in MainActivity
  - Add below code under “#direct methods“:

```
.method static constructor <clinit>()V
    .locals 1
    .prologue
    const-string v0, "frida-gadget"
    invoke-static {v0}, Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
    return-void
.end method
```

Frida – NonRooted Android

- To AndroidManifest.xml add
  - <uses-permission android:name="android.permission.INTERNET" />

- `apktool b diva/ -o updated_diva.apk`

- `keytool -keystore dnskey.jks -genkey -keyalg RSA -keysize 2048 -validity 30 -alias dnskey`

- `jarsigner -keystore dnskey.jks updated_diva.apk dnskey`

Frida – NonRooted Android

- `pip install frida`
- If Error -> Use -> sudo -H pip install frida --ignore-installed six
- `sudo pip install frida-tools`
- `easy_install pyasn1`
- Run `` `frida-ps -Uai` `` to confirm if Frida works

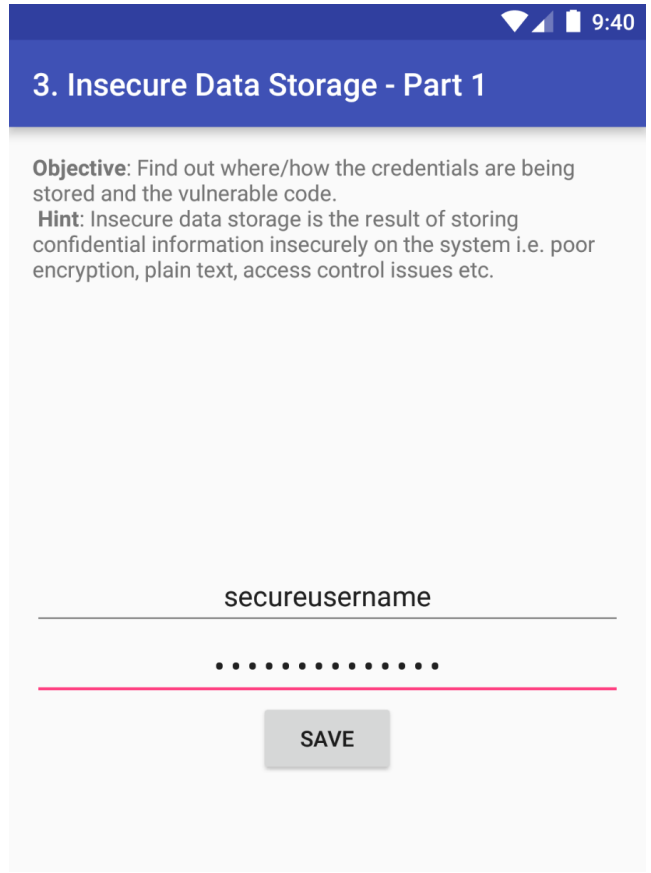
Frida – NonRooted Android

- Launch Application
  - `"adb logcat"` should say "Frida: Listening on TCP port 27042"
  - Screen will be blank

- On Mac run
  - `frida-ps -U`
  - `frida -U Gadget`
  - For function tracing - `frida-trace -U -i open Gadget`

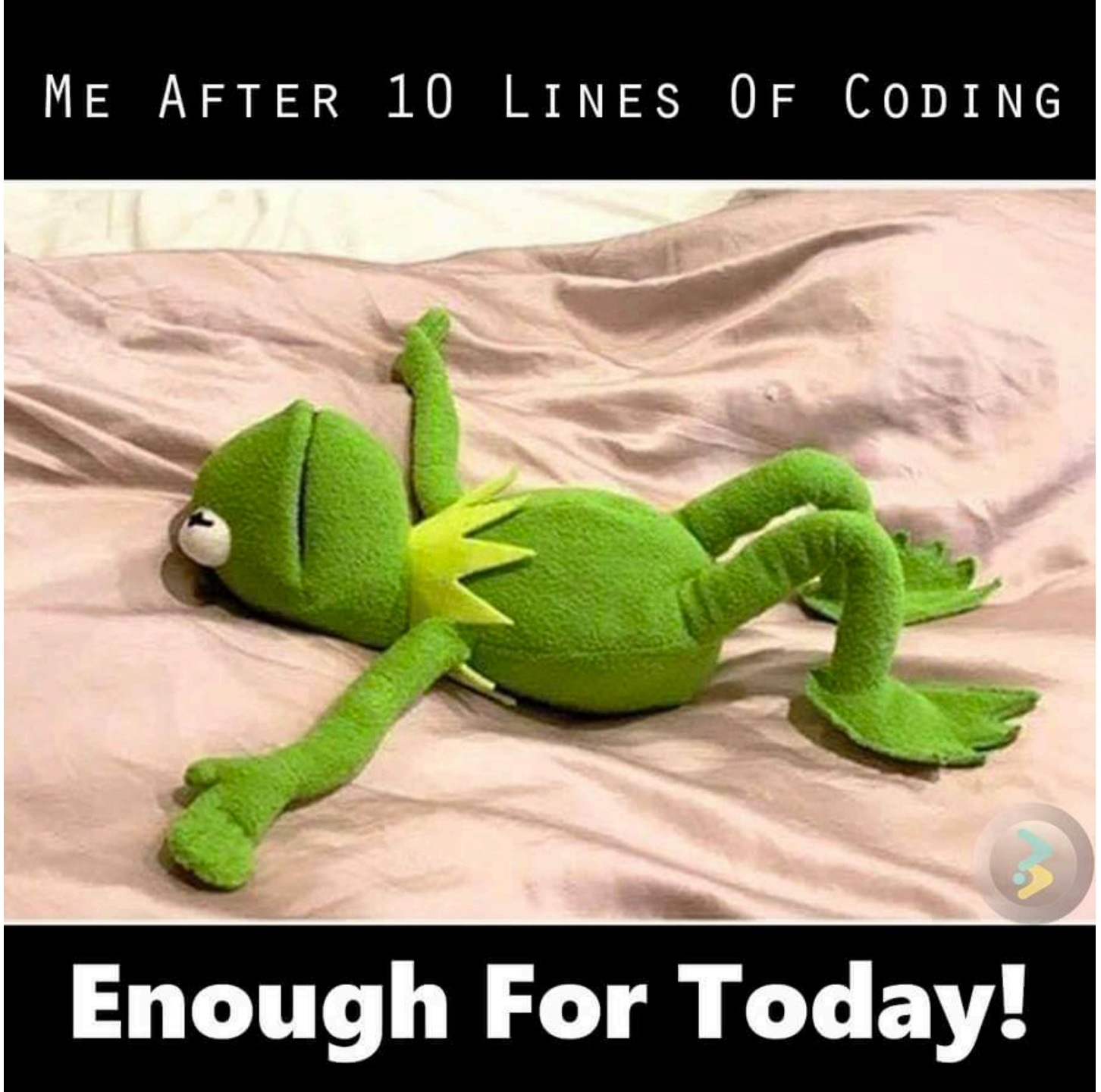
Frida – NonRooted Android

- `frida -U Gadget`
- For function tracing - frida-trace -U -i open Gadget




```
➜  ~ frida-trace -U -i open Gadget
Instrumenting functions...
open: Loaded handler at "/Users/dns/__handlers__/libc.so/open.js"
Started tracing 1 function. Press Ctrl+C to stop.
           /* TID 0x1648 */
 28181 ms  open(path="/data/misc/profiles/cur/0/jakhar.aseem.diva/primary.prof"
 oflag=0xa0002)
 28181 ms  open(path="/data/misc/profiles/cur/0/jakhar.aseem.diva/primary.pr
 oflag=0xa0001)
 66190 ms  open(path="/data/misc/profiles/cur/0/jakhar.aseem.diva/primary.pr
 oflag=0xa0002)
           /* TID 0x163a */
 69366 ms  open(path="/data/user/0/jakhar.aseem.diva/shared_prefs/jakhar.asee
iva_preferences.xml", oflag=0x241)
           /* TID 0x1656 */
 69427 ms  open(path="/proc/cpuinfo", oflag=0x0)
```

```
➜  ~ adb shell
r/0/jakhar.aseem.diva/shared_prefs/jakhar.aseem.diva_preferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="password">securepassword</string>
    <string name="user">secureusername</string>
</map>
vbox86p:/ #
```



github.com/sensepost/objection

objection - Runtime Mobile Exploration

`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.

twitter @leonjza | pypi package 1.9.5 | build passing | Black Hat Arsenal Europe 2017 | Black Hat Arsenal USA 2019

- 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...

Screenshots are available in the wiki.

```
def patch_android_apk(source: str, architecture: str, pause: bool, skip_cleanup: bool = True,
                      enable_debug: bool = True, gadget_version: str = None, skip_resources: bool = False,
                      network_security_config: bool = False, target_class: str = None,
                      use_aapt2: bool = False, gadget_config: str = None, script_source: str = None,
                      ignore_nativelibs: bool = True) -> None:
    """
        Patches an Android APK by extracting, patching SMALI, repackaging
        and signing a new APK.

        :param source:
        :param architecture:
        :param pause:
        :param skip_cleanup:
        :param enable_debug:
        :param gadget_version:
        :param skip_resources:
        :param network_security_config:
        :param target_class:
        :param use_aapt2:
        :param gadget_config:
        :param script_source:

        :return:
    """
```

Frida – NonRooted Android - Automated

```
objection patchapk -s <<apk-name>
-enable-debug
```

HOMEWORK

- Backdoor the /home/mobile/Desktop/vulnapps/Snapchat.apk file with Frida Gadget **using Objection**

- Ensure that you are able to connect to the Snapchat application using objection

Solution

```
objection patchapk -s
Snapchat.apk
```

```
mobile@ubuntu:~/Desktop/vulnapps$ objection patchapk -s Snapchat.apk
No architecture specified. Determining it using `adb`...
Detected target device architecture as: x86
Using latest Github gadget version: 12.11.0
Patcher will be using Gadget version: 12.11.0
Detected apktool version as: 2.4.1
Running apktool empty-framework-dir...
I: Removing 1.apk framework file...
Unpacking Snapchat.apk
App already has android.permission.INTERNET
Target class not specified, searching for launchable activity instead...
Smali not found in smali directory. This might be a multidex APK. Searching...
Found smali at: /tmp/tmpv4vpi8zx.apktemp/smali_classes2/com/snapchat/android/LandingPageActivity.smali
Reading smali from: /tmp/tmpv4vpi8zx.apktemp/smali_classes2/com/snapchat/android/LandingPageActivity.smali
Injecting loadLibrary call at line: 480
Attempting to fix the constructors .locals count
Current locals value is 0, updating to 1:
Writing patched smali back to: /tmp/tmpv4vpi8zx.apktemp/smali_classes2/com/snapchat/android/LandingPageActivity.sm
ali
Creating library path: /tmp/tmpv4vpi8zx.apktemp/lib/x86
Copying Frida gadget to libs path...
Rebuilding the APK with the frida-gadget loaded...

Built new APK with injected loadLibrary and frida-gadget
Signing new APK.
Signed the new APK
Performing zipalign
Zipalign completed
Copying final apk from /tmp/tmpv4vpi8zx.apktemp.aligned.objection.apk to Snapchat.objection.apk
ry...
Cleaning up temp files...
mobile@ubuntu:~/Desktop/vulnapps$
mobile@ubuntu:~/Desktop/vulnapps$ ls | grep Snapchat.objection.apk
Snapchat.objection.apk
mobile@ubuntu:~/Desktop/vulnapps$ adb install Snapchat.objection.apk
Success
```

```
mobile@ubuntu:~/Desktop/vulnapps$ frida-ps -Uai
 PID  Name     Identifier
----  -------  --------------
6976  Gadget   re.frida.Gadget
mobile@ubuntu:~/Desktop/vulnapps$ objection --gadget Gadget explore
Using USB device `CustomPhone1`
Agent injected and responds ok!

     _     _           _   _
 ___| |_|_|___ ___ ___| |_|_|___ ___
| . | . | | -_|  _|  _|  _| | . |   |
|___|___| |___|___|_| |_| |_|___|_|_|
      |___|(object)inject(ion) v1.9.5

     Runtime Mobile Exploration
        by: @leonjza from @sensepost

[tab] for command suggestions
com.snapchat.android on (Android: 10) [usb] #
```

Bonus Slides - Game Hacking

- CheatEngine for Android Games
  - Memory Editor - https://github.com/aktsk/apk-medit
  - Target APK - GreenWallGame.apk
    - `adb shell`
    - `pm list packages # to check <target-package-name>`
    - `run-as com.bulsy.greenwall`
    - `cp /data/local/tmp/medit ./medit`
    - `./medit`

Steps

- Inside medit
  - `Find 3`
  - Splash something and lose 1 life
  - `Filter 2`
  - `Patch 94`
  - `Exit`