



# The Hitchhacker's Guide to the Mobile Galaxy

WorkShop



BSides Munich 2023

# Your Captain

- Penetration tester at NVISO with focus on web and mobile
- Develops and teaches trainings on secure coding practices and pentesting basics
- Guest lecturer at FH Hagenberg (Austria) for *Mobile and Embedded OS* course



✉ [claudia.ully@nviso.eu](mailto:claudia.ully@nviso.eu)

[linkedin.com/in/claudia-ully](https://www.linkedin.com/in/claudia-ully)



# Our Journey



Adventures  
on Android



Meddling in  
the Middle



Big Bang  
of Basics



Incidents  
on iOS

# Travel Advisories

# DON'T PANIC

# Travel Advisories



**Buckle up!**

This course will be hands-on



**Don't be shy!**

Ask whenever anything is unclear, or you need help. Your neighbours can help, too!



**Keep exploring!**

The course will only scratch the surface

# Our Journey



Adventures  
on Android



Meddling in  
the Middle



Big Bang  
of Basics



Incidents  
on iOS

# Short Planetology

Android and iOS basics

# Chronology of the mobile universe

1973

First portable cell phone by Motorola



By: Rico Shen

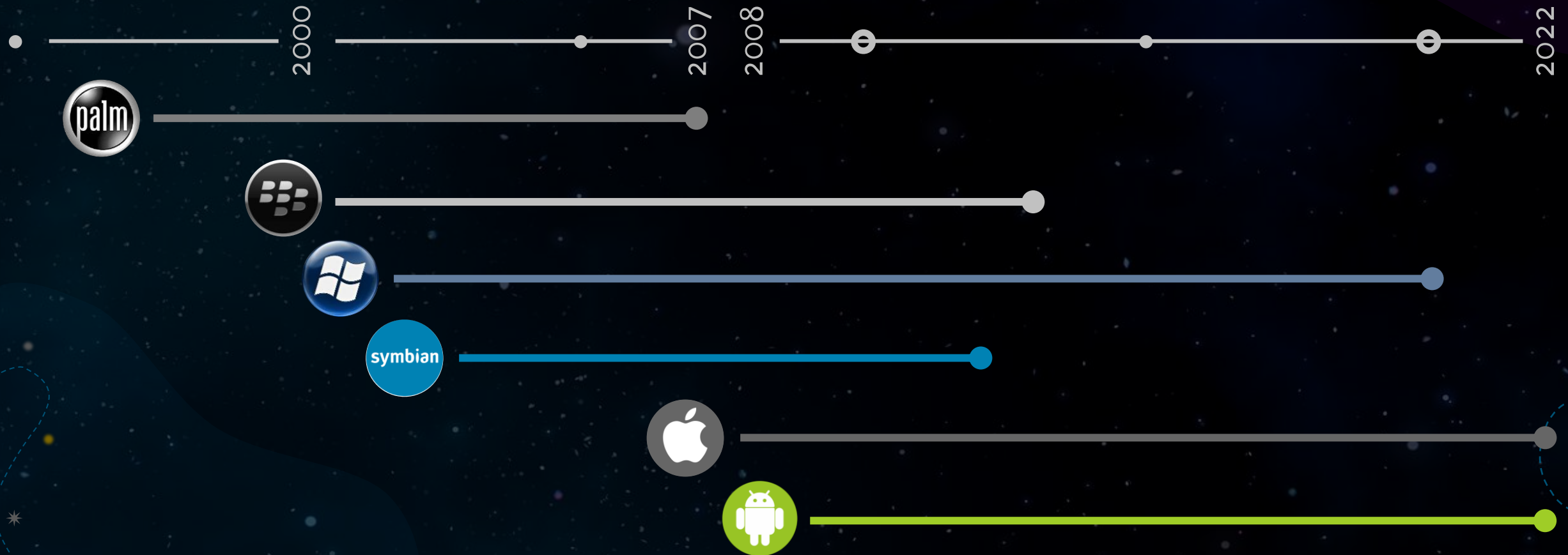
1994

First touchscreen phone by IBM

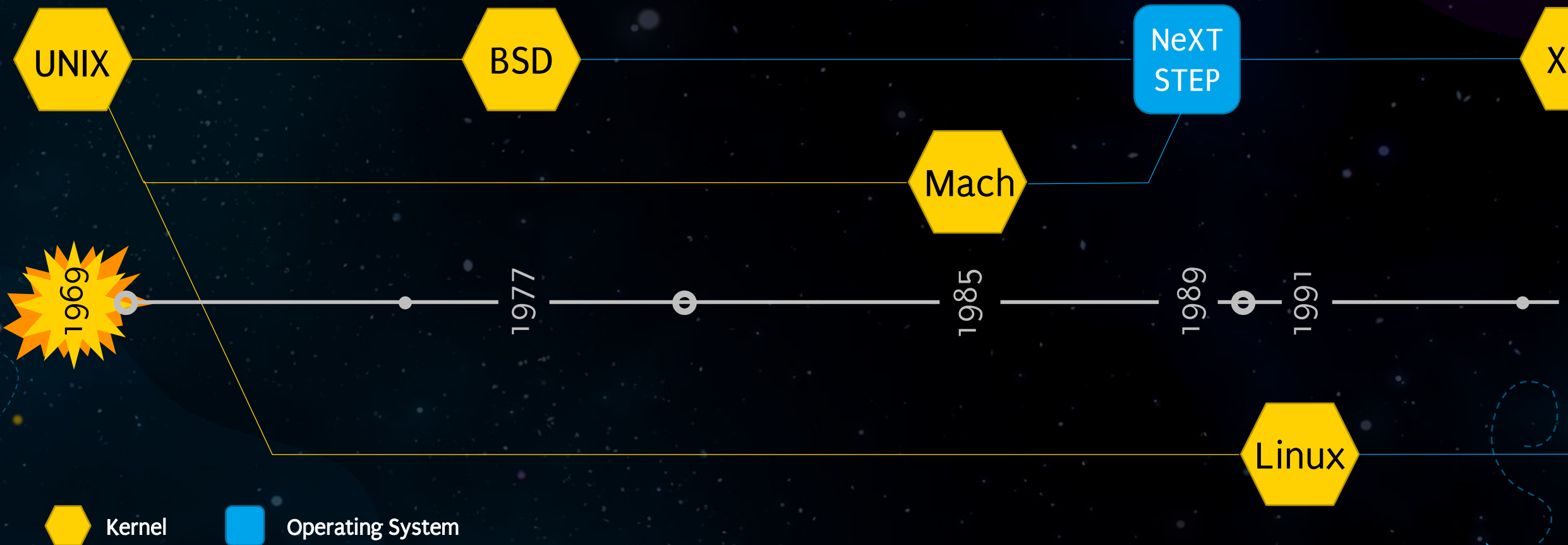




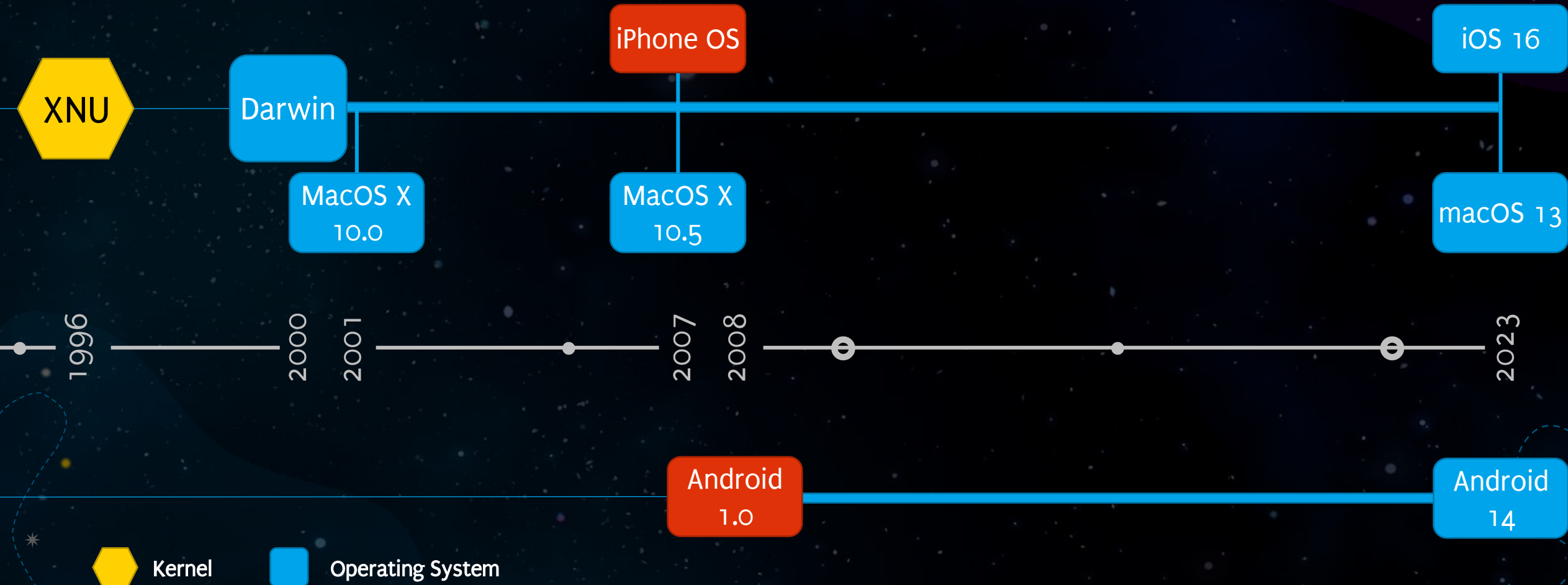
# Chronology of the mobile universe



# Evolution of Android and iOS



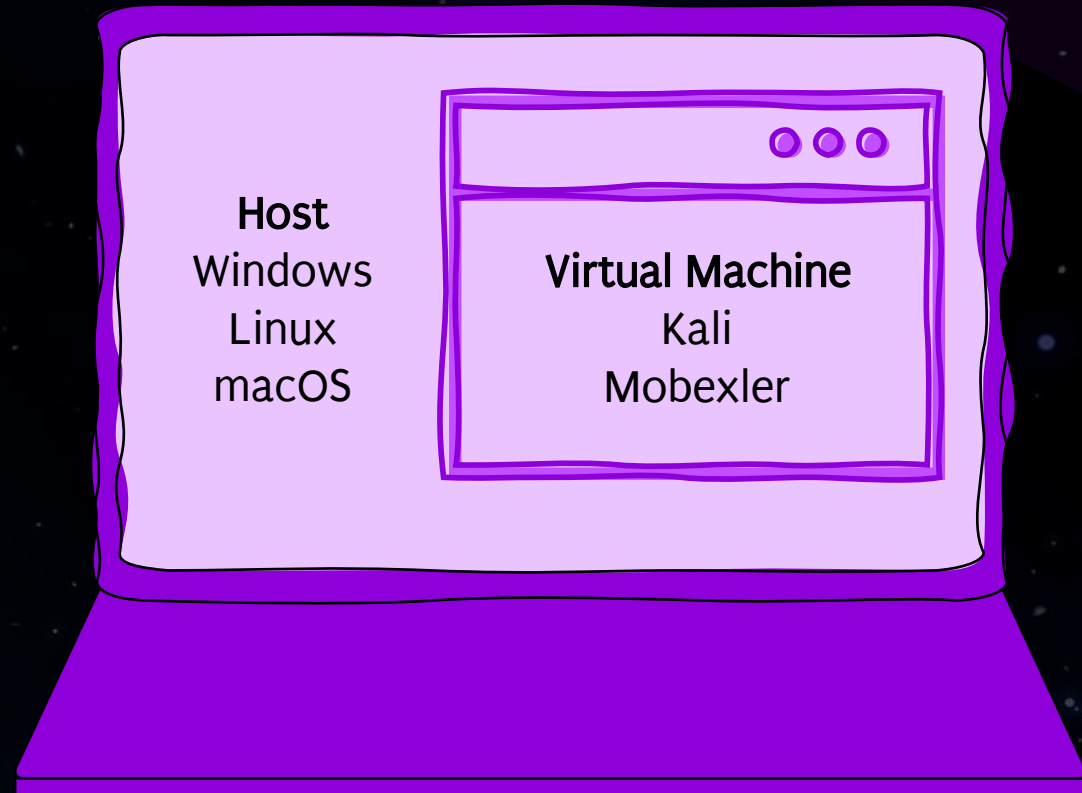
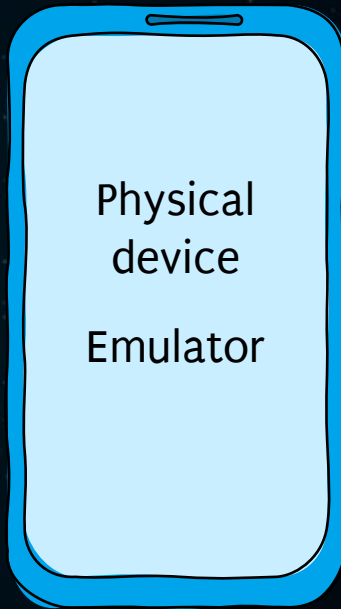
# Evolution of Android and iOS



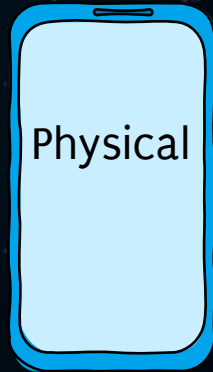
# Boarding the Spaceship

Setting everything up

# ASSEMBLING the parts



# ASSEMBLING the parts

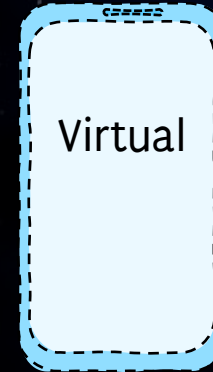


Physical

ARM processor

Not all devices are easily rootable

Cost factor

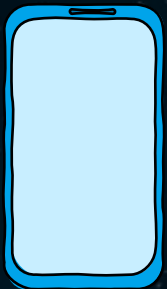


Virtual

Usually x86 processor;  
ARM processor can also be emulated  
but will be very slow

Missing features, e.g. biometrics, NFC

# ASSEMBLING the parts



android studio



Genymotion



CORELLIUM



# Grab Your Towels



The screenshot displays a Linux desktop environment. A terminal window is open, showing a list of tools categorized by type. The background features a dark blue theme with a white wolf logo and the text "MOBEXLER HACKERS | BY HACKERS".

**Mobexler**

- Favorites
- Recently Used
- All
- AndroidZone
- iOSZone
- SecZone
- Internet

**Androbugs framework**  
an Android vulnerability analysis system

**Android CLI**  
Use the command line

**Android Studio**  
Android Dev

**Burp Suite**  
Burp Suite Proxy Community version

**Bytecode-Viewer**  
Java Bytecode Viewer, GUI Java Decompiler

**DB Browser for Sqlite**  
DB Browser for SQLite is a light GUI editor for SQLite databases

**Drozer**  
comprehensive security audit and attack framework for Android

**Frida**  
Dynamic Instrumentation Toolkit

**Jadx**  
java code decompiler

**JD-GUI**  
Java Decompiler

**MARA Framework**  
Mobile Application Reverse engineering and Analysis Framework

**MobSF**  
Docker run for apk Static Analysis

**MobSF image Manager**  
Mobile Security Framework

**Objection**  
runtime mobile exploration toolkit

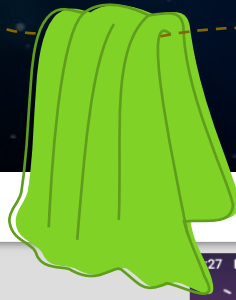
**Pidcat**  
Colorful log analysis tool

**MOBEXLER**  
HACKERS | BY HACKERS

Taskbar: Mobexler, [Icons], 1:50:42 pm



# Grab Your Towels



The screenshot displays the Genymotion web interface. On the left is a navigation sidebar with the following menu items: Administrator, Display (highlighted in red), Instance information, Configuration, Device content, Share, API reference, Device Logs, Shell, Documentation (with a dropdown arrow), and Support (with a dropdown arrow). The main area shows a virtual Android device screen with a space-themed wallpaper, a Google search bar at the top, and a dock at the bottom with icons for Phone, Messages, Gallery, and Camera. The device status bar at the top shows the time 27 and various system icons. On the right side of the interface, there is a vertical toolbar with various control icons for the virtual device, including volume, rotation, zoom, and power. At the bottom left of the interface, the text reads: © 2016 - 2023 Genymotion Cloud v13.1

# Grab Your Towels



All exercises, files and instructions are available at:

<http://bsides.hitchhacker.space>

Username: bsides  
Password: hitchhacker



# The bridge is yours

1. Launch Mobexler; PW: mobexler

2. Install missing tools:

- `sudo apt install apktool adb`
- `wget https://github.com/patrickfav/uber-apk-signer/releases/download/v1.3.0/uber-apk-signer-1.3.0.jar`
- `git clone https://github.com/Hamz-a/frida-android-helper`  
`cd frida-android-helper`  
`sudo python3 setup.py install`

# Our Journey



Big Bang  
of Basics



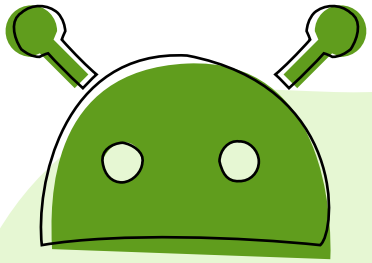
Adventures  
on Android



Incidents  
on iOS



Meddling in  
the Middle



# Getting Ready for Launch

Devices & rooting

# Android Studio

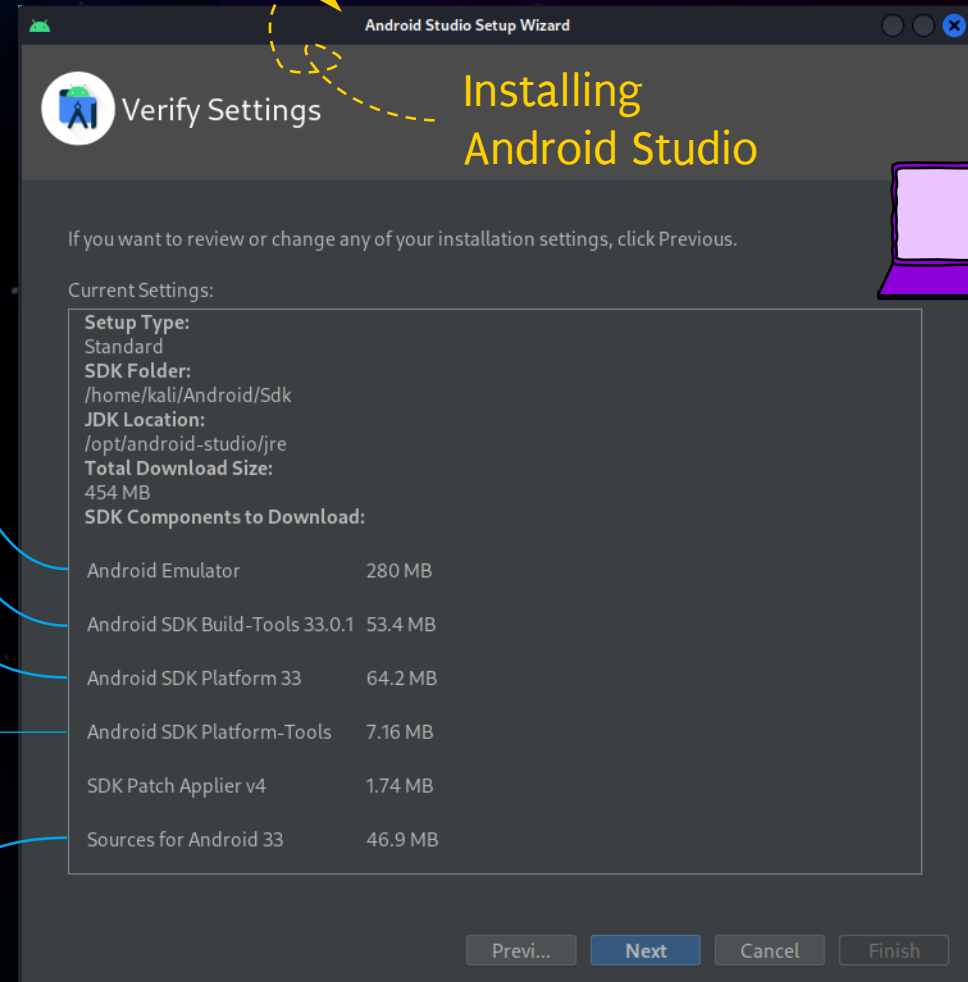
Emulates Android devices  
(Android Virtual Device, AVD)

Tools for building Android apps  
<sdk>/build-tools/

Tools and libraries for app development

Command line tools interfacing with the  
Android platform, e.g. adb and fastboot

Android source files



# Android Studio Emulator



Play Store available on emulated device, but no root access

Virtual Device Configuration

### Select Hardware

Choose a device definition

Category	Name	Play Store	Size	Resolution	Density
Phone	Pixel 5		6.0"	1080x23...	440dpi
Tablet	Pixel 4a		5.8"	1080x23...	440dpi
Wear OS	Pixel 4 XL		6.3"	1440x30...	560dpi
Desktop	Pixel 4	▶	5.7"	1080x22...	440dpi
TV	Pixel 3a XL		6.0"	1080x21...	400dpi
Automotive	Pixel 3a	▶	5.6"	1080x22...	440dpi
	Pixel 3 XL		6.3"	1440x29...	560dpi

New Hardware Profile Import Hardware Profiles

Root access on emulated device

Virtual Device Configuration

### System Image

Select a system image

Recommended x86 Images Other Images

Release Name	API Level	ABI	Target
<b>Tiramisu</b> ⬇	33	x86_64	Android 13.0 (Google A
Sv2 ⬇	32	x86_64	Android 12L (Google A
S ⬇	31	x86_64	Android 12.0 (Google A
R ⬇	30	x86	Android 11.0 (Google A
Q ⬇	29	x86	Android 10.0 (Google A
Pie ⬇	28	x86	Android 9.0 (Google A
Oreo ⬇	27	x86	Android 8.1 (Google A
Oreo ⬇	26	x86	Android 8.0 (Google A
Nougat ⬇	25	x86	Android 7.1.1 (Google A

**Tiramisu**

API Level  
**33**

Android  
**13.0**  
**Google Inc.**

System Image  
**x86\_64**

We recommend these images because they run the fastest and support Google APIs.

Questions on API level?  
See the API level distribution chart.

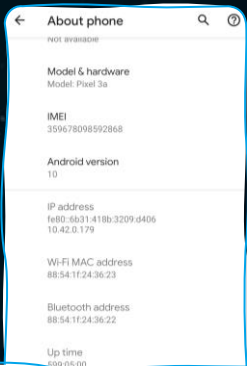
Previous Next Cancel Finish

# Rooting



Rooting steps differ from device type and Android version but usually require these high-level steps

Enabling developer options to enable USB debugging

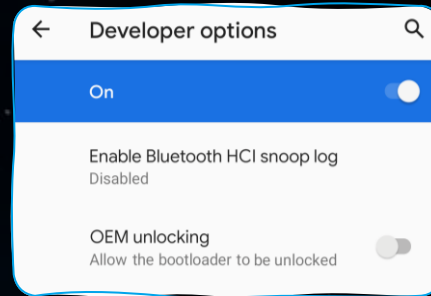


Build number  
QQ2A.200305.002

Send feedback about this device

Build **You are now a developer!**  
QQ2A.200305.002

Unlocking the bootloader



```
$> adb reboot bootloader  
$> fastboot flashing unlock
```

Flashing a patched image

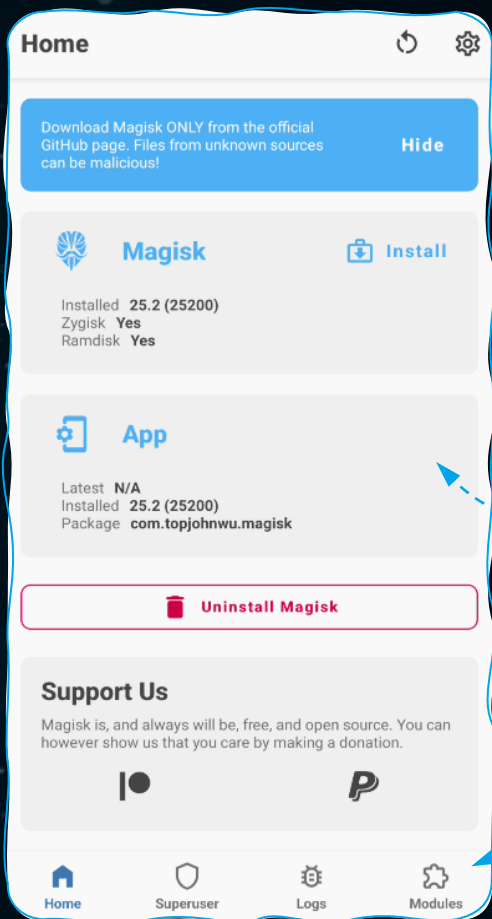
```
$> fastboot flash boot  
magisk_patched.img
```

Or sideloading a custom ROM

```
$> fastboot flash recovery  
lineage-recovery.img  
$> adb sideload lineageos.zip
```



# Magisk



## Magisk

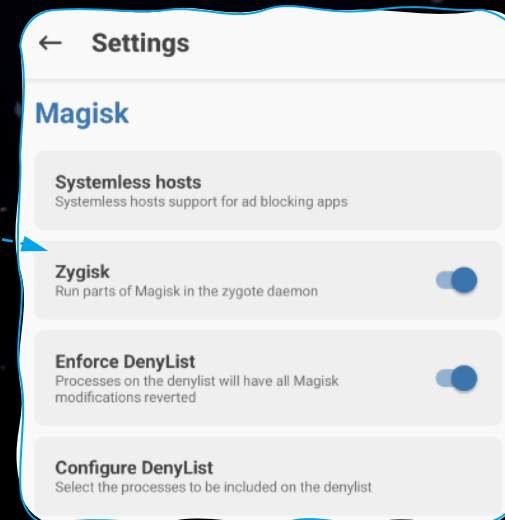
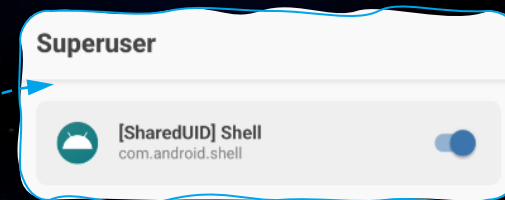
The Magic Mask for Android

<https://github.com/topjohnwu/Magisk>

Rooting without modifying the /system partition

Run inside Zygote process and unload it for certain apps (e.g. in case of root detection)

Installing additional modules

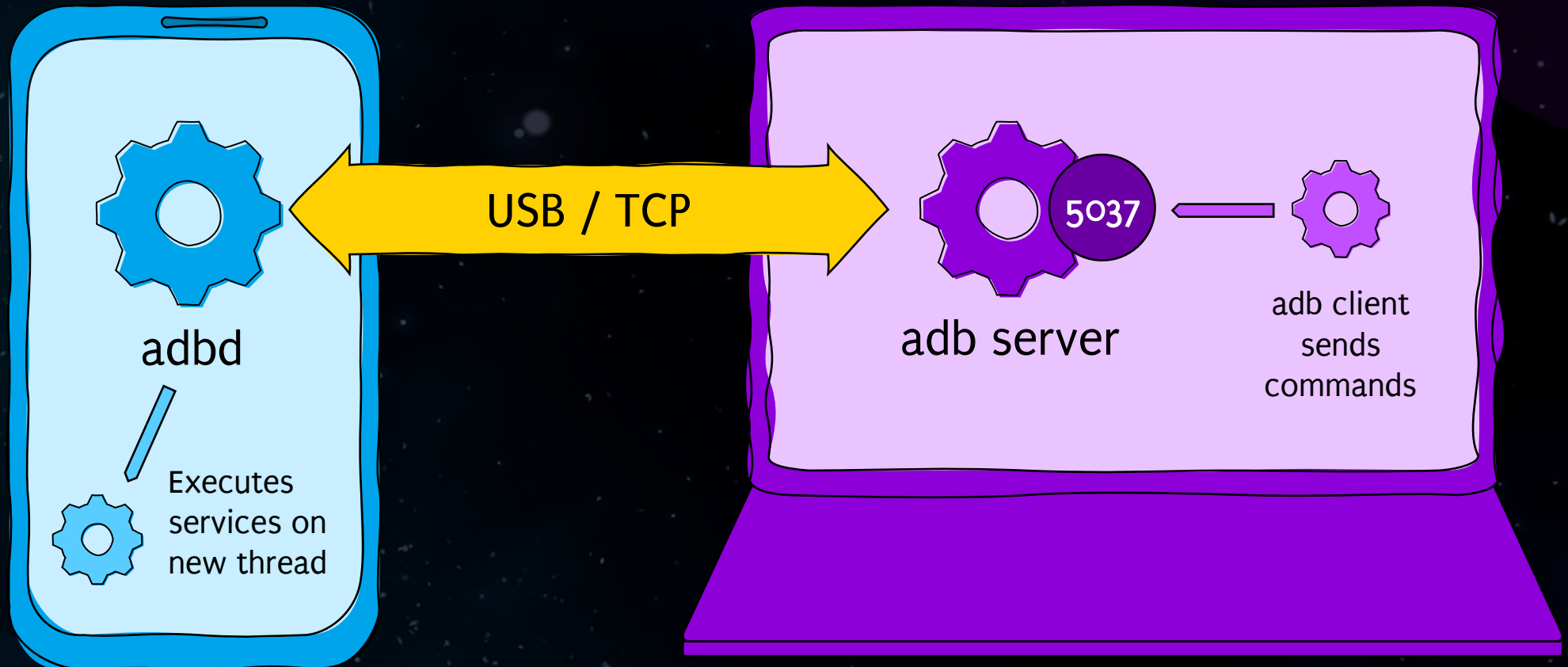




# First Contact

Reaching out

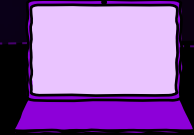
# Reaching out



# Reaching out



*command starts adb client, which first checks if server is running*



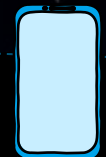
```
$> adb devices
```

```
* daemon not running; starting now at tcp:5037  
* daemon started successfully
```

```
$> netstat -ant
```

```
Active Internet connections (servers and established)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	127.0.0.1:5037	0.0.0.0:*	LISTEN



```
sargo:/ $ ps -A | grep adb
```

root	1310	1	10886352	2012 0	0 S	adb_root
shell	15209	1	11012400	7744 0	0 S	abdd

# Reaching out



List all available devices

Start an interactive shell on a device  
connected via USB

If there are multiple devices, we need to  
specify the serial number of the target device

Restart adb as root and start a root shell  
(only possible on rooted devices)

Connect to a device via TCP  
(useful for connecting to virtual devices)

```
$> adb devices
List of devices attached
A01X21CF00E    device
RF1DB6K177Y    device

$> adb shell
$> adb -s A01X21CF00E shell
sargo:/ $

$> adb root
$> adb shell
sargo:/ #

$> adb connect [IP]:5555
```

# Reaching out



Install an app package (apk)

```
$> adb install hitchhacker.apk
Performing Streamed Install
Success
```

List all installed packages, e.g. to find out package name of installed app

```
$> adb shell pm list packages | grep hitch
package:space.hitchhacker.guide
```

Find out install location of the app binary

```
$> adb shell pm path space.hitchhacker.guide
package:/data/app/~~ogBY1Czkk70W3C63cFVYdA==/
space.hitchhacker.guide-
uSilirCcYZWeKbnAasXZTg==/base.apk
```

Uninstall an app

```
$> adb uninstall space.hitchhacker.guide
Success
```

# Reaching out



Copy file to the phone

```
$> adb push file.txt /sdcard
file.txt: 1 file pushed, 0 skipped. 0.0 MB/s
(18 bytes in 0.099s)
```

Copy file from the phone

```
$> adb pull /sdcard/file.txt ~/Documents
/sdcard/file.txt: 1 file pulled, 0...ped. 0.0
MB/s (18 bytes in 0.008s)
```

View system log

```
$> adb logcat
```

Create backup of an app  
(deprecated!)

```
$> adb backup -f mybackup.ab -apk
-shared space.hitchhacker.guide
Now unlock your device and confirm the backup
operation...
```



## The bridge is yours

1. Activate adb access
2. Install the Hitchhacker app with “adb install” or via drag&drop
3. Open a shell on the device and find out the package name of the app
4. Browse to the app binary





# Rocket Science

Inside an APK

# InSide an APK



```
$> unzip hitchhacker.apk -d hitchhacker  
$> ls -la hitchhacker
```



assets



AndroidManifest.xml



META-INF



classes.dex



res



resources.arsc

# InSide an APK



assets

*Application assets, e.g. document templates*



META-INF

*Signature and certificate for app distribution*



res

*UI resource files, e.g. XML configuration files*



AndroidManifest.xml

*Binary file; app information*



classes.dex

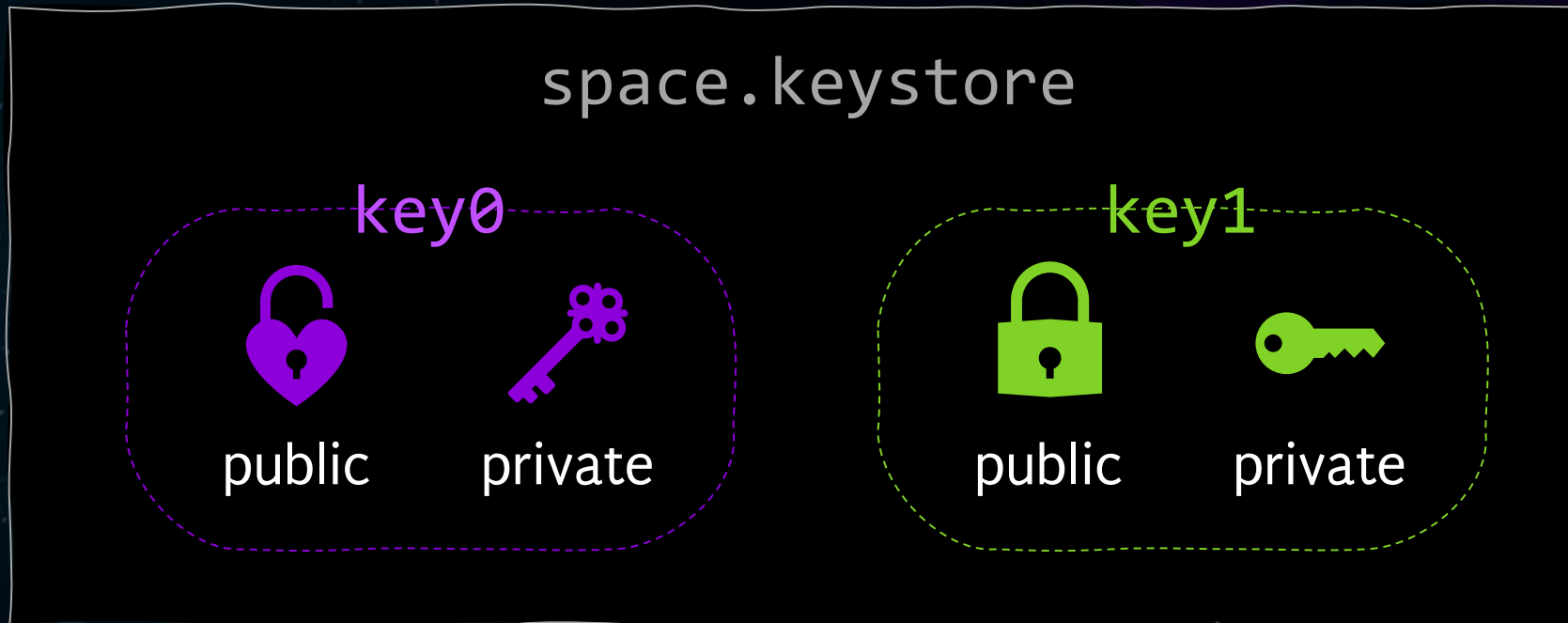
*Dalvik file: app code*



resources.arsc

*Binary file; describes app resources*

# Signing off - Keystore



# Signing off - Creating a keystore

```
$> keytool -genkey -v -keystore space.keystore -alias key0 -keyalg RSA -keysize 2048 -validity 10000
```

```
Enter keystore password:
```

```
Re-enter new password:
```

```
What is your first and last name?
```

```
[Unknown]: Arthur Dent
```

```
What is the name of your organizational unit?
```

```
[Unknown]: Hitchhacker
```

```
What is the name of your organization?
```

```
[Unknown]: Intergalactic Travellers
```

```
What is the name of your City or Locality?
```

```
[Unknown]: Heart of Gold
```

```
What is the name of your State or Province?
```

```
[Unknown]:
```

```
What is the two-letter country code for this unit?
```

```
[Unknown]:
```

```
Is CN=Arthur Dent, OU=Hitchhacker, O=Intergalactic Travellers, L=Heart of Gold, ST=Unknown, C=Unknown
```

```
[no]: yes
```

```
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days  
for: CN=Arthur Dent, OU=Hitchhacker, O=Intergalactic Travellers, L=Heart of Gold, ST=Unknown, C=Unknown  
[Storing space.keystore]
```

New Key Store

Key store path: /home/kali/Documents/spacekeystore.jks

Password: ..... Confirm: .....

Key

Alias: key0

Password: ..... Confirm: .....

Validity (years): 25

Certificate

First and Last Name: Arthur Dent

Organizational Unit: Hitchhacker

Organization: Intergalactic Travellers

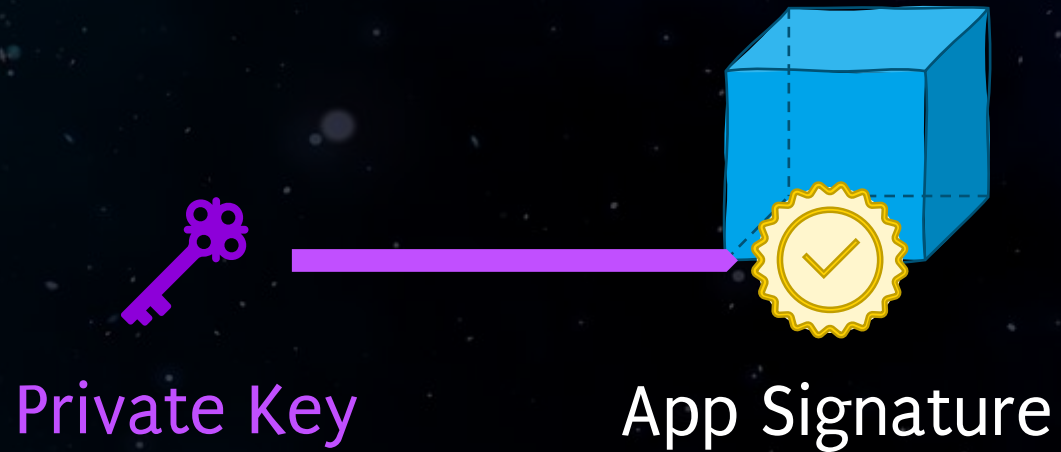
City or Locality: Heart of Gold

State or Province:

Country Code (XX):

OK Cancel

# Signing off - Signing an app



```
$> apksigner sign --ks spacekeystore.jks --ks-key-alias key0 universe.apk  
Keystore password for signer #1:
```

# Signing off - Signing an app

## Certificate for app signature



Meta data



Public Key

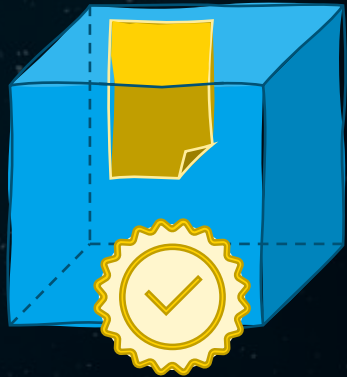
```
$> apksigner verify --print-certs universe.apk
Signer #1 certificate DN: L=Heart of Gold, O=Intergalactic
  Travellers, OU=Hitchhackers, CN=Ford Prefect
Signer #1 certificate SHA-256 digest:
  f2ea49fb4bed82a93207928e15df91dc21fba7d08bfac3010f
  f8d37ad85e7586
Signer #1 certificate SHA-1 digest:
  08ad6226f6e41c3f9220950d7df5f9bf04af7d19
Signer #1 certificate MD5 digest:
  827d01734d6141b91fa17ea32655cbd6
```

# Signing off - Verifying the signature



Certificate inside APK

META-INF/CERT.RSA



App Signature

Android Package Manager verifies upon installation if app signature (private key) and contained certificate (public key) match



# Signing off... for some else?



```
$> jarsigner -verify -verbose -certs NASA.apk
```

```
...
```

```
- Signed by "CN=NASA Ames Research Center, OU=Ames Research Center, O=NASA, L=Mountain View, ST=CA, C=US"
```

```
  Digest algorithm: SHA-256
```

```
  Signature algorithm: SHA256withRSA, 2048-bit key
```

*Android does not perform CA verification for application certificates - you can sign with whatever you want*

```
$> jarsigner -verify -verbose -certs NASA.apk
```

```
...
```

```
- Signed by "CN=Zaphod Beeblebrox, OU=Heart of Gold, O=Galaxy, L=Betelgeuse, ST=Galaxy, C=GA"
```

```
  Digest algorithm: SHA-256
```

```
  Signature algorithm: SHA256withRSA, 2048-bit key
```

# Signing off... for Some else?



A (malicious) app using the same app ID cannot be installed or overwrite files on the device if the certificate doesn't match the one of the app that is already installed



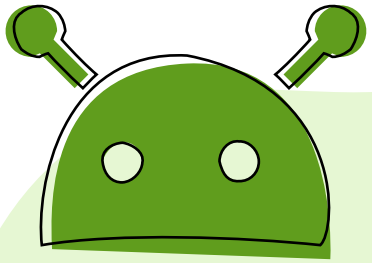
gov.nasa



An app using a different ID will be installed



space.nasa



# Scanning the Area

Static analysis

# Decompiling apps

```
i = 1;  
j = 1;  
while (true) {  
    *val++ = i+j;  
    j = i+(i=j);  
}
```

High-level language

*Compiler*

```
mov r0,#1  
mov r1,#1  
l:  
add r2,r0,r1  
str r2,[r3]  
add r3,#4  
mov r0,r1  
mov r1,r2  
b l
```

Assembler

*Assembler*

```
110100111010000000  
0000000000000001110  
100111010000000010  
000011000011101000  
010000111000100000  
000001000110101011  
000001100100000000  
110100101000001100  
110000111000100010  
100000000010011101  
000011010001101010
```

Machine code

# Decompiling apps

```
public static void  
main(String[] args) {  
    int a = 1;  
    int b = 2;  
    int c = a + b;  
}
```

Source code (.java)

*Compiler*

```
Code:  
stack=2, locals=4,  
args_size=1  
0: iconst_1  
1: istore_1  
2: iconst_2  
3: istore_2  
4: iload_1  
5: iload_2  
6: iadd  
7: istore_3  
8: return
```

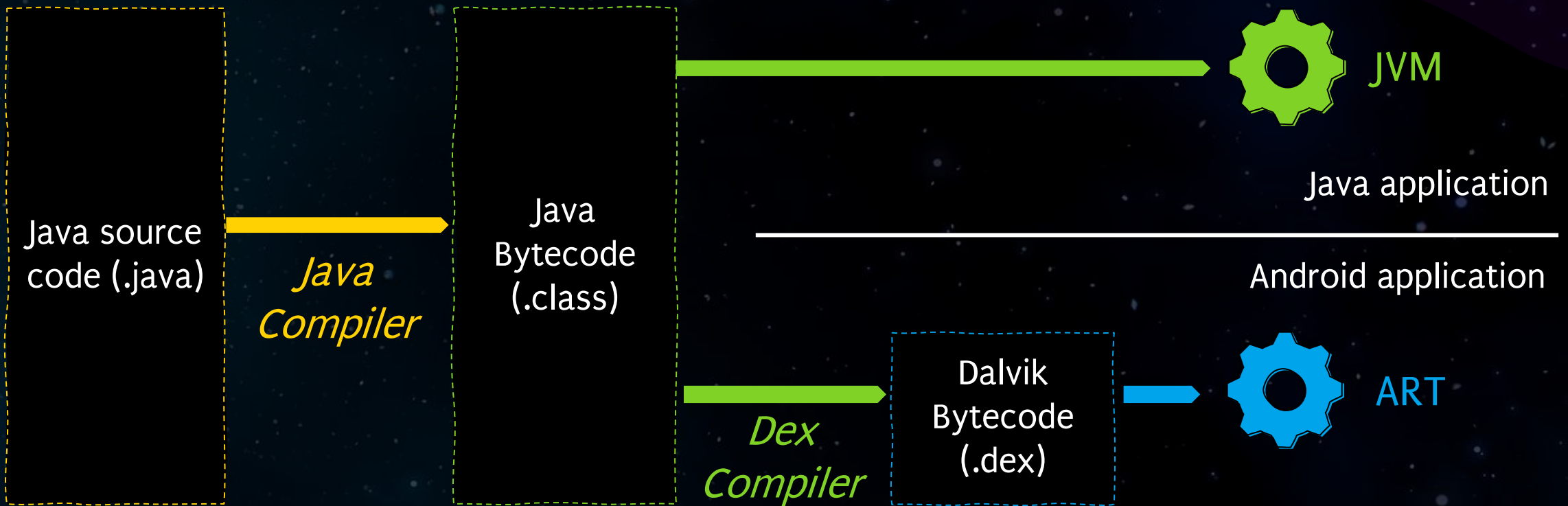
Byte code (.class)

*JVM*

```
110100111010000000  
0000000000000001110  
100111010000000010  
000011000011101000  
010000111000100000  
000001000110101011  
000001100100000000  
110100101000001100  
110000111000100010  
100000000010011101  
000011010001101010
```

Machine code

# Decompiling apps



# Decompiling APK files



```
if (isExternalStorageWritable()) {  
    File file = new File (Environment.getExternalStorageDirectory(), "password.txt");  
    // Log.d("Storage Directory", String.valueOf(Environment.getExternalStorageDirectory()));  
    String password = "L33tS3cr3t";  
    FileOutputStream fos;  
    try {  
        fos = new FileOutputStream(file);  
        fos.write(password.getBytes());  
        fos.close();  
    } catch (FileNotFoundException e) {  
        e.printStackTrace();  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}
```

*Original source code*

# Decompiling APK files



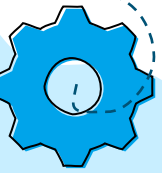
```
if (isExternalStorageWritable()) {
    File file = new File (Environment.getExternalStorageDirectory(), "password.txt");
    // Log.d("Storage Directory", String.valueOf(Environment.getExternalStorageDirectory()));
    String password = "L33tS3cr3t";
    FileOutputStream fos;
    try {
        fos = new FileOutputStream(file);
        fos.write(password.getBytes());
        fos.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

```
if (isExternalStorageWritable()) {
    try {
        FileOutputStream fileOutputStream = new FileOutputStream(new
            File(Environment.getExternalStorageDirectory(),
                "password.txt"));
        fileOutputStream.write("L33tS3cr3t".getBytes());
        fileOutputStream.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e2) {
        e2.printStackTrace();
    }
}
```

*By decompiling we can easily retrieve nearly the original source code*



# JADX-GUI



Package structure

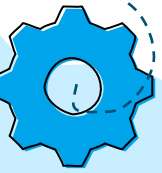
The screenshot shows the JADX-GUI interface. On the left, a package structure tree is visible for 'MSTG-Android-Java.apk', showing a hierarchy of folders like 'android.support.v4', 'androidx', 'com', 'net.sqlcipher', and 'sg.vp.owasp\_mobile'. The 'MyActivity' class is selected. The main window displays the decompiled Java code for 'MyActivity', including imports, package declarations, and class definitions. A search dialog box titled 'Text search: secret' is open, showing search results for the text 'secret' within the code. The search options include 'Case insensitive', 'Regex', and 'Active tab only'. The search results show a list of nodes with the text 'secret' highlighted in yellow.

Searching within classes

Decompiled Java code

<https://github.com/skylot/jadx>

# Bytecode Viewer



Bytecode Viewer 2.11.2 - <https://bytecodeviewer.com> | <https://the.bytecode.club> - @Konloch

Package structure

Decompiled Java code using up to three different decompilers

Searching within classes

```
1 package sg.vp.owasp_mobile.OMTG_Android;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import android.view.Menu;
6 import android.view.MenuItem;
7 import android.view.View;
8 import androidx.appcompat.app.AppCompatActivity;
9 import androidx.appcompat.widget.Toolbar;
10
11 public class MyActivity extends AppCompatActivity {
12     public static final String EXTRA_MESSAGE = "com.mycompany.myfirstapp.
13
14     public void OMTG_CODING_003_Best_Practice(View var1) {
15         this.startActivity(new Intent(this, OMTG_CODING_003_Best_Practice.
16     }
17
18     public void OMTG_CODING_003_SQL_Injection(View var1) {
19         this.startActivity(new Intent(this, OMTG_CODING_003_SQL_Injection.
20     }
21
22     public void OMTG_CODING_003_SQL_Injection_Content_Provider(View var1)
23         this.startActivity(new Intent(this, OMTG_CODING_003_SQL_Injection
24     }
25
26     public void OMTG_CODING_004_Code_Injection(View var1) {
27         this.startActivity(new Intent(this, OMTG_CODING_004_Code_Injection
28     }
29
30     public void OMTG_DATAST_001_BadEncryption(View var1) {
31         this.startActivity(new Intent(this, OMTG_DATAST_001_BadEncryption.
32     }
33
34     public void OMTG_DATAST_001_ExternalStorage(View var1) {
35         this.startActivity(new Intent(this, OMTG_DATAST_001_ExternalStorag
36     }
37
38     public void OMTG_DATAST_001_ExternalStorage(final View view) {
39         this.startActivity(new Intent((Context)this, (Class)OMTG_DATAST_0
```

<https://github.com/Konloch/bytecode-viewer>



# The bridge is yours

1. Demo: Using jadx-gui
2. Find the Hitchhacker app's original pin by using static analysis

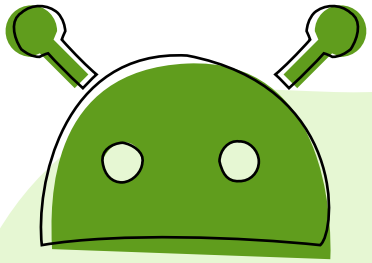
# Solution

```
File View Navigation Tools Help
hitchhacker3.apk
├── Source code
│   ├── android.support.v4
│   ├── androidx
│   ├── com
│   ├── kotlin
│   ├── kotlin.coroutines
│   ├── net.sqlcipher
│   ├── org
│   └── space.hitchhacker.guide
│       ├── comm
│       ├── data
│       ├── databinding
│       ├── ui
│       ├── App
│       ├── AppKt
│       ├── AppKt$database$2
│       ├── AppKt$encprefs$2
│       ├── AppKt$prefs$2
│       ├── BuildConfig
│       ├── LoginActivity
│       ├── MainActivity
│       ├── PinActivity
│       ├── R
│       └── UrlActivity
├── Resources
│   ├── APK signature
│   └── Summary
└── LoginActivity
    ├── 65     this.pinInput.get(index).clearFocus();
    ├── 66     this.pinInput.get(index + 1).requestFocus();
    └── }

    72     private final void clearPin() {
    73         Iterable $this$forEach$iv = this.pinInput;
    74         for (Object element$iv : $this$forEach$iv) {
    105             EditText it = (EditText) element$iv;
    73             it.getText().clear();
    74         }
    74         this.pinInput.get(3).clearFocus();
    75         this.pinInput.get(0).requestFocus();
    75     }

    82     private final boolean validatePin() {
    83         StringBuilder concat = new StringBuilder();
    84         Iterable $this$forEach$iv = this.pinInput;
    107         for (Object element$iv : $this$forEach$iv) {
    84             EditText it = (EditText) element$iv;
    84             concat.append(it.getText().toString());
    85         }
    85         String enteredPin = concat.toString();
    86         Intrinsic.checkNotNullExpressionValue(enteredPin, "concat.toString()");
    86         if (AppKt.getEncprefs().getEncSharedPrefsManualPin()) {
    87             boolean result = Intrinsic.areEqual(enteredPin, AppKt.getEncprefs().getEncSharedPrefsPin());
    94             return result;
    89         }
    89         int bigNumber = Integer.parseInt(enteredPin);
    90         int check1 = bigNumber >> 2;
    90         boolean result2 = true;
    91         int check2 = (Integer.parseInt(this.pinInput.get(3).getText().toString()) - Integer.parseInt(this.pinInput.get(1).getText().toString())) * Integer.parseInt(this.
    91         pinInput.get(2).getText().toString());
    92         if (!(check1 == 42 && check2 == 42)) {
    92             result2 = false;
    92         }
    92         return result2;
    92     }

    100     private final void launchMain() {
    101         Intent changeView = new Intent(this, MainActivity.class);
    102         startActivity(changeView);
    102     }
}
```



# The Cargo Bay

How data is stored on Android

# One app, multiple locations



/data/app/~~xyz/space.hitchhacker.guide



base.apk



oat



lib

/data/data/space.hitchhacker.guide



cache



files



code-cache



lib



databases



shared\_prefs

/storage/emulated/0/Android/data/  
space.hitchhacker.guide



files

# Getting an APK



```
$> adb shell pm path space.hitchhacker.guide
```

```
package:/data/app/~~gF7fUVcWUu59lRcvexS8tQ==/space.hitchhacker.guide-shjdiXYF300VtZfhqrAjQg==/base.apk
```

```
$> adb pull /data/app/~~gF7fUVcWUu59lRcvexS8tQ==/space.hitchhacker.guide-shjdiXYF300VtZfhqrAjQg==/base.apk
```

```
/data/app/~~gF7fUVcWUu59lRcvexS8tQ==/space.hitchhacker.guide-shjdiXYF300VtZfhqrAjQg==/base.apk: 1 file pulled, 0 skipped. 30.9 MB/s (5295949 bytes in 0.163s)
```

# Split APKs



/data/app/~~xyz/space.hitchhacker



base.apk



split\_config.arm64.apk



split\_config.nl.apk



split\_config.xxhdpi.apk

```
$> python3 patch-apk.py space.hitchhacker
```

```
Getting APK path(s) for package: space.hitchhacker
```

```
[+] APK path: /data/app/~~XqZX_gCpfQQ==/space.hitchhacker-XvabuQowwLc8uGKXUaqHPw==/base.apk
```

```
[+] APK path: /data/app/~~XqZX_gCpfQQ==/space.hitchhacker-XvuQowc8uGKXUaqHPw==/split_config.arm64_v8a.apk
```

```
...
```

```
Pulling APK file(s) from device.
```

```
[+] Pulling: space.hitchhacker-base.apk
```

```
[+] Pulling: space.hitchhacker-split_config.arm64_v8a.apk
```

```
...
```

```
[+] Pulling: space.hitchhacker-split_config.xxhdpi.apk
```

```
App bundle/split APK detected, rebuilding as a single APK.
```

<https://github.com/TheDauntless/patch-apk/tree/master>



# Shared Preferences



```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="UsernameKey">arthurdent</string>
  <string name="PasswordKey">acupoftea</string>
</map>
```

*Key-value pairs  
stored in XML*

Stored in clear  
text by default

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="ARCyFuEcW9/TcaBiAs+tOgZubZYGzw5i9NkTORKY764=">
AWu/DKgTl/TWIAesThkCZtqssJwcg8a1RICuP/UQem9iFhx08vDrgRGM8rMBrlqKvGoE
</string>
  <string name="__androidx_security_crypto_encrypted_prefs_key_keyset__">
12a901e172...001
</string>
</map>
```

Stored encrypted  
using androidx.  
security.crypto.  
EncryptedSharedP  
references

# Databases



```
sargo:/data/data/glaxy.hitchhacker.guide/databases # sqlite3 travelCompanions
SQLite version 3.32.2 2021-07-12 15:00:17
Enter ".help" for usage hints.
sqlite> .tables
Companions          android_metadata
sqlite> select * from Companions;
fordprefect|goosnargh
```

SQLite databases are not encrypted by default



We can use the external library SQLCipher to encrypt the entire database

# Databases



## Realm DB

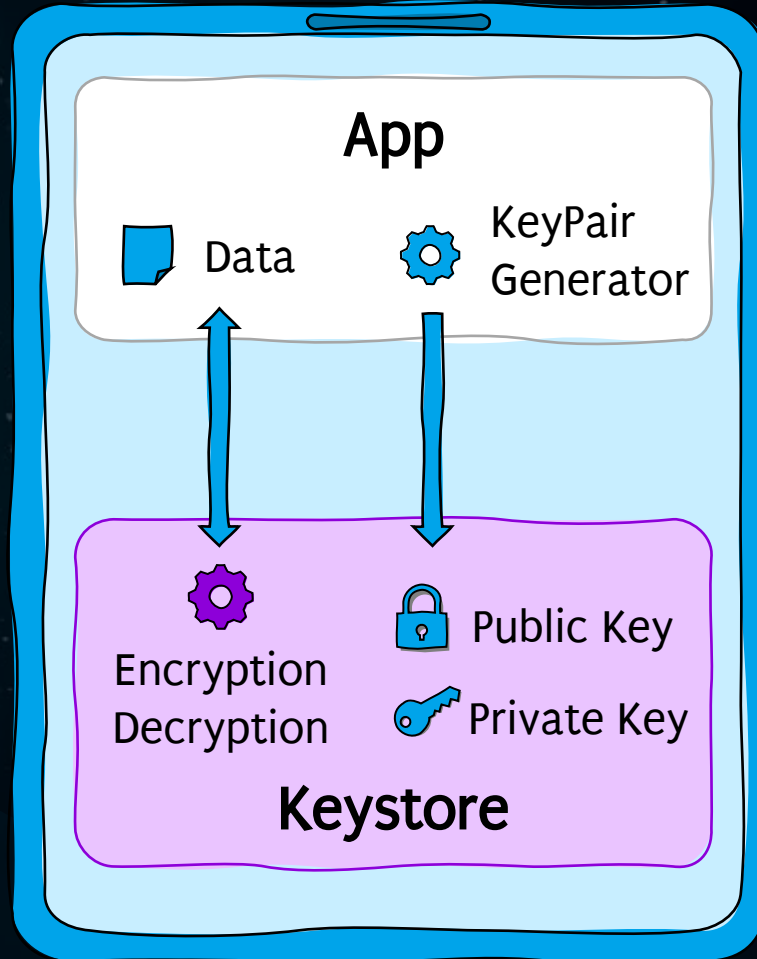
Not encrypted by default  
Encryption can be enabled  
through configuration



## Firebase DB

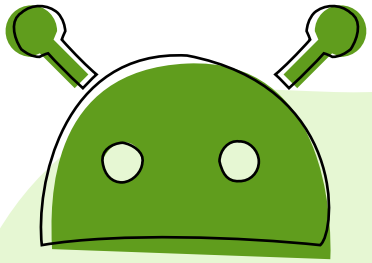
Cloud database,  
encrypted in transit, but not  
locally  
  
Main security concern: Proper  
authorization & access controls

# Keystore



Secure container for storing keys and performing cryptographic operations

- Software-based at `/data/misc/keystore/user_0`
- Hardware-based as secure area on main processor or separate microchip



# InSide the engine

Identifying the attack surface

# Android Manifest



*XML file defining information and components of the application*

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-feature android:name="android.hardware.location.gps" android:required="false"/>
  <application android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:icon="@mipmap/hitchhacker_guide" android:name="galaxy.hitchhacker_guide"
android:allowBackup="false" android:usesCleartextTraffic="true">
    <meta-data android:name="com.google.android.geo.API_KEY" android:value="AIzaSyB159r06yyIvPEBG37daQUaaial6RdryVc"/>
    <activity android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:name="galaxy.hitchhacker.MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>
    <provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
      android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true" />
    <service android:name="galaxy.hitchhacker.SpaceShipMonitor" android:permission="android.permission.BIND_JOB_SERVICE"
      android:enabled="@bool/enable_system_job_service_default" android:exported="true" android:directBootAware="false"/>
    <receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
      <intent-filter>
        <action android:name="galaxy.spaceship.signal.RECEIVE"/>
      </intent-filter>
    </receiver>
  </application>
</manifest>
```

*Exported components can be interacted with by other apps*

# Attack Surface



```
<application android:theme="@style/AppTheme_NoActionBar" android:Label="Hitchhacker
  android:icon="@mipmap/hitchhacker_guide" android:name="galaxy.hitchhacker,guide"
  android:allowBackup="false" android:usesCleartextTraffic="true">
  <activity android:theme="@style/AppTheme_NoActionBar" android:Label="Hitchhacker"
    android:name="galaxy.hitchhacker.MainActivity">
    <intent-filter>
      <action android:name="android.intent.action.MAIN"/>
      <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
  </activity>
  <provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
    android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true" />
  <service android:name="galaxy.hitchhacker.SpaceShipMonitor"
    android:permission="android.permission.BIND_JOB_SERVICE"
    android:enabled="@bool/enable_system_job_service_default" android:exported="true"
    android:directBootAware="false"/>
  <receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
    <intent-filter>
      <action android:name="galaxy.spaceship.signal.RECEIVE"/>
    </intent-filter>
  </receiver>
</application>
```

*Implicitly exported component through intent-filters*  
(as of Android 12, explicit exports are obligatory)

*Explicitly exported component*

# Permissions



*Permissions required  
by the app*

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

<intent-filter>
  <action android:name="android.intent.action.MAIN"/>
  <category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>
<provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
  android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true" />
<service android:name="galaxy.hitchhacker.SpaceShipMonitor" android:permission="android.permission.BIND_JOB_SERVICE"
  android:enabled="@bool/enable_system_job_service_default" android:exported="true" android:directBootAware="false"/>
<receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
  <intent-filter>
    <action android:name="galaxy.spaceship.signal.RECEIVE"/>
  </intent-filter>
</receiver>
</application>
</manifest>
```



# Activities



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

*Activities = Visible parts in an application*


```
<activity android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:name="galaxy.hitchhacker.MainActivity">
  <intent-filter>
    <action android:name="android.intent.action.MAIN"/>
    <category android:name="android.intent.category.LAUNCHER"/>
  </intent-filter>
</activity>
```

*Name hints at the class that implements the activity*


```
    android:enabled="@bool/enable_system_job_service_default" android:exported="true" android:directBootAware="false"/>
  <receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
    <intent-filter>
      <action android:name="galaxy.spaceship.signal.RECEIVE"/>
    </intent-filter>
  </receiver>
</application>
</manifest>
```

# Activities



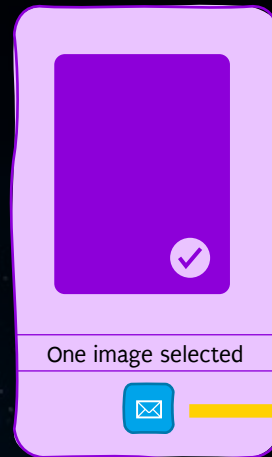
Inbox 

- Ford Prefect**  
Don't forget towel
- Marvin**  
Why bother at all?
- Trillian**  
Don't have time
- Zaphod**  
Wow


New Mail 

To:

Subject:



One image selected



# Activities



*Action name (optional)*

```
android:/ $ am start -a android.intent.action.MAIN  
-n galaxy.hitchhacker/.MainActivity
```

*package\_name/.activity\_name*

# Content Providers



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-feature android:name="android.hardware.location.gps" android:required="false"/>
  <application android:theme="@style/AppTheme_NoActionBar" android:Label="Hitchhacker" android:icon="@mipmap/hitchhacker_guide" android:allowBackup="false" android:usesCleartextTraffic="true">
    <meta-data android:name="com.google.android.geo.API_KEY" android:value="AIzaSyB159r06y1VPZBGa7daQUaaial6Rd" />
    <activity android:theme="@style/AppTheme_NoActionBar" android:Label="Hitchhacker" android:name="galaxy.hitchhacker.GuideBrowser">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

*Content Provider = Share data stored in app database between apps*

```
<provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
  android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true" />
```

*Not exported, i.e. not available to other apps*

# Content Providers



*Access using a URI*

```
sargo:/ $ content query --uri "content://galaxy.  
hitchhacker.provider/galaxies/1"
```

*Can be queried like a database:  
provider/database/entryId  
(name of the database can usually  
determined through source code analysis)*

# Services



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-feature android:name="android.hardware.location.gps" android:required="false"/>
  <application android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:icon="@mipmap/hitchhacker_guide" android:name="galaxy.hitchhacker_guide"
android:allowBackup="false" android:usesCleartextTraffic="true">
    <meta-data android:name="com.google.android.geo.API_KEY" android:value="AIzaSyB159r06yyIvPEBG37daQUaaial6R0" />
    <activity android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:name="galaxy.hitchhacker.MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>
    <provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
      android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true" />
  </application>
</manifest>
```

*Service = Perform tasks in  
the background*

```
<service android:name="galaxy.hitchhacker.SpaceShipMonitor" android:permission="android.permission.BIND_JOB_SERVICE"
android:enabled="@bool/enable_system_job_service_default" android:exported="true" android:directBootAware="false"/>
```

# Services



*package\_name/.service\_name*

```
sargo:/ $ am startservice  
-n galaxy.hitchhacker/.SpaceshipMonitor  
--es ship "Vogon Constructor Fleet"
```

*Name of a StringExtra  
expected by the service*

*Content of the StringExtra*

# Broadcast Receivers



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-feature android:name="android.hardware.location.gps" android:required="false"/>
  <application android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:icon="@mipmap/hitchhacker_guide" android:name="galaxy.hitchhacker_guide"
android:allowBackup="false" android:usesCleartextTraffic="true">
    <meta-data android:name="com.google.android.geo.API_KEY" android:value="AIzaSyB159r06yyIvPEBG37daQUaaial6R..."/>
    <activity android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:name="galaxy.hitchhacker.MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>
    <provider android:name="galaxy.hitchhacker.GuideBrowser" android:exported="false"
      android:authorities="galaxy.hitchhacker.provider" android:grantUriPermissions="true">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </provider>
  </application>
</manifest>
```

*Broadcast Receiver = Listen  
and handle incoming intents,  
e.g. notifications*

```
<receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
  <intent-filter>
    <action android:name="galaxy.spaceship.signal.RECEIVE"/>
  </intent-filter>
</receiver>
```



# Broadcast Receivers



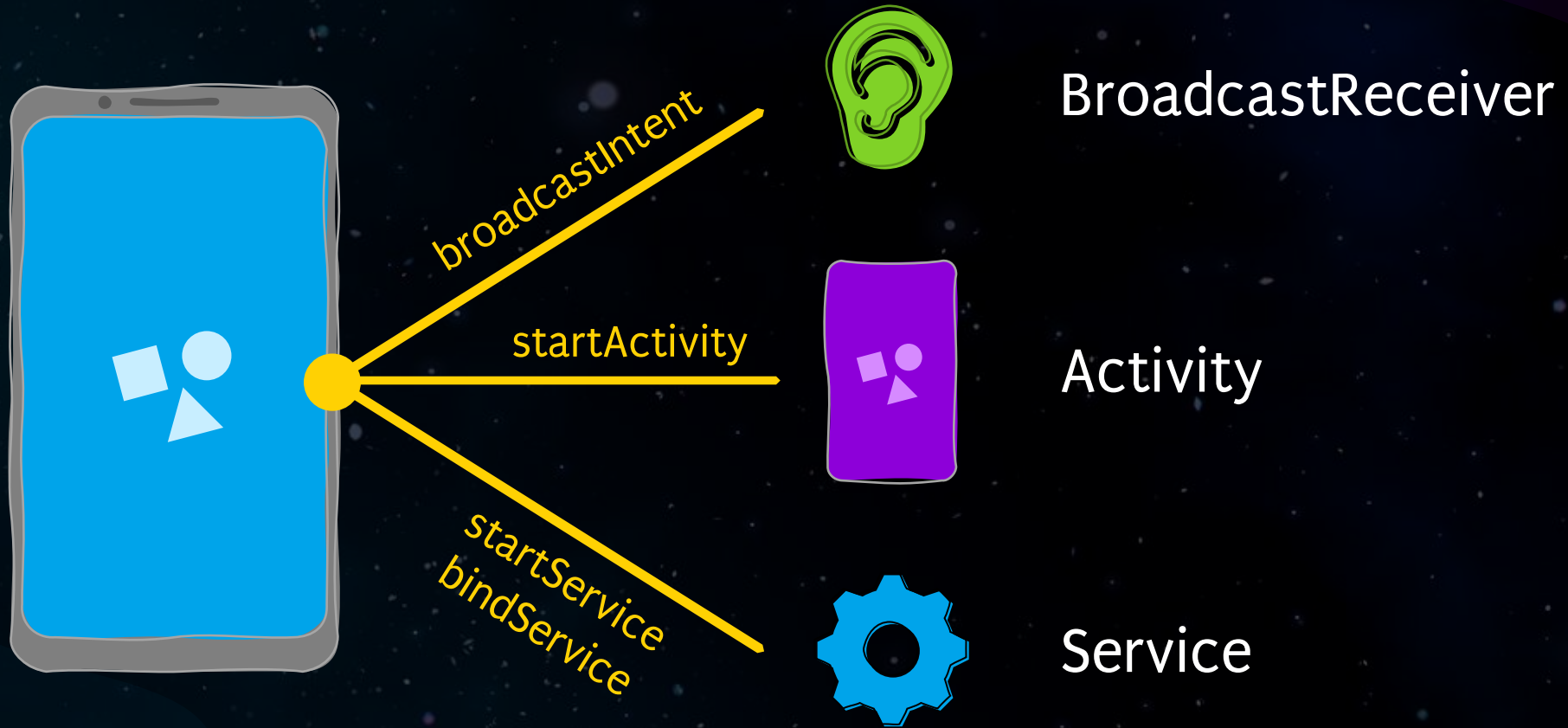
*package\_name.receiver\_name*

```
sargo:/ $ am broadcast  
-n galaxy.hitchhacker/.SpaceshipAlert  
--es ship "Vogon Constructor Fleet"
```

*Name of a StringExtra  
expected by the receiver*

*Content of the StringExtra*

# Intents



# Intents



## Explicit Intent

Supplies class name or target app's package name that is to handle the intent

► Sent to specified app

## Implicit Intent

Declares an action to perform without specifying a certain component

► Sent to all apps

```
Intent showDetail = new Intent(this, DetailActivity.class);  
startActivity(showDetail);
```

```
Intent showUrl = new Intent(Intent.ACTION_VIEW, Uri.parse(url));  
startActivity(showUrl);
```

# Explicit Intents - Examples



Calling an activity within the same app

```
Intent showDetail = new Intent(this, DetailActivity.class);  
startActivity(showDetail);
```

Calling an activity that is part of another app

```
ComponentName cn = new ComponentName("space.hitchhacker.other",  
    "space.hitchhacker.other.DetailActivity");  
Intent showDetails = new Intent();  
showDetails.setComponent(cn);  
showDetails.setAction("space.hitchhacker.SHOW_DETAILS")  
startActivity(showDetails);
```

# Implicit Intents - Examples



Calling an app that can display URLs (e.g. browser)

```
Intent showUrl = new Intent(Intent.ACTION_VIEW,  
    Uri.parse("https://hitchhacker.space"));  
startActivity(showUrl);
```

Calling an app for making a phone call

```
Intent callNumber = new Intent(Intent.ACTION_DIAL,  
    Uri.parse("tel:123456"));  
startActivity(callNumber);
```

# Sending data with intents

```
Intent emailIntent = new Intent(Intent.ACTION_SEND);  
emailIntent.putExtra(Intent.EXTRA_EMAIL, new String[] {"arthur@hitchhacker.space"});  
emailIntent.putExtra(Intent.EXTRA_SUBJECT, "Packing List");  
emailIntent.putExtra(Intent.EXTRA_TEXT, "Towel");  
emailIntent.putExtra("Custom", "Something else");  
startActivity(emailIntent)
```

Data contained in an intent can be accessed by the receiving component

```
Intent intent = getIntent();  
String subject = intent.getStringExtra(Intent.EXTRA_SUBJECT);  
Bundle allExtras = intent.getExtras();
```

# Intent Filters



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="97" android:versionName="2.05" android:installLocation="auto" android:compileSdkVersion="30"
android:compileSdkVersionCodename="11" package="galaxy.hitchhacker" platformBuildVersionCode="30" platformBuildVersionName="11">
  <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="30"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-feature android:name="android.hardware.location.gps" android:required="false"/>
  <application android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:icon="@mipmap/hitchhacker_guide" android:name="galaxy.hitchhacker_guide"
android:allowBackup="false" android:usesCleartextTraffic="true">
    <meta-data android:name="com.google.android.geo.API_KEY" android:value="AIzaSyB159r06yyIvPEBG37daQUaaial6RdryVc"/>
    <activity android:theme="@style/AppTheme_NoActionBar" android:label="Hitchhacker" android:name="galaxy.hitchhacker.MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
      <receiver android:name="galaxy.hitchhacker.SpaceShipAlert" android:exported="true">
        <intent-filter>
          <action android:name="galaxy.spaceship.signal.RECEIVE"/>
        </intent-filter>
      </receiver>
    </application>
  </manifest>
```

*Intent filters specify which types of implicit intents the component handles*

# Intent Filters - Examples



## Default intents

Handles ACTION\_SEND intents that contain text data; starts the ReceiveText activity within the app

```
<activity android:name=".ReceiveText">
  <intent-filter>
    <action android:name="android.intent.action.SEND"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:mimeType="text/plain"/>
  </intent-filter>
</activity>
```

## Custom intents

Handles the custom SHIPLAUNCH intent; starts the LaunchShip activity within the app

```
<activity android:name=".LaunchShip" >
  <intent-filter>
    <action android:name="space.hitchhacker.SHIPLAUNCH" />
    <category android:name="android.intent.category.DEFAULT" />
  </intent-filter>
</activity>
```



# Intent Filters - Examples



## Launcher

Intent filter that handles when the app is started from the Launcher

```
<intent-filter>  
  <action android:name="android.intent.action.MAIN"/>  
  <category android:name="android.intent.category.LAUNCHER"/>  
  <category android:name="android.intent.category.DEFAULT"/>  
</intent-filter>
```

## Deep Links

Intent filter that can handle incoming links that begin with space://hitchhacker

```
<intent-filter>  
  <action android:name="android.intent.action.VIEW" />  
  <category android:name="android.intent.category.DEFAULT" />  
  <category android:name="android.intent.category.BROWSABLE" />  
  <data android:scheme="space" android:host="hitchhacker" />  
</intent-filter>
```

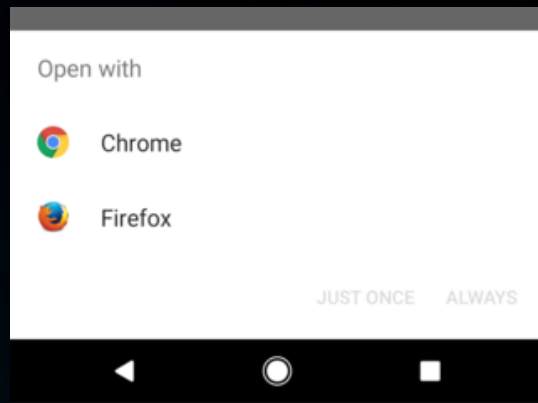
# Intent Filters - Handling



If no app exists that can handle the intent, an exception occurs

```
try {  
    startActivity(intent);  
} catch (ActivityNotFoundException e) {  
    // Do something  
}
```

An app can register for any implicit intents; if more than one app exists that can handle the intent, the app picker is shown



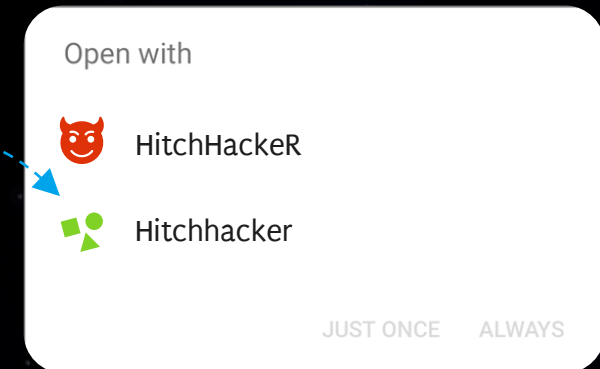
# Deep Links



```
<intent-filter>  
  <action android:name="android.intent.action.VIEW" />  
  <category android:name="android.intent.category.DEFAULT" />  
  <category android:name="android.intent.category.BROWSABLE" />  
  <data android:scheme="space" android:host="hitchhacker" />  
</intent-filter>
```

App will open when custom URL scheme  
space://hitchhacker is invoked

Dialog appears if another app has  
registered the same scheme



# App Links



```
<intent-filter>
  <action android:name="android.intent.action.VIEW" />
  <category android:name="android.intent.category.DEFAULT" />
  <category android:name="android.intent.category.BROWSABLE" />
  <data android:scheme="https" android:host="hitchhacker.space"/>
</intent-filter>
```

App Link is tied to a website

```
[{
  "relation": ["delegate_permission/common.handle_all_urls"],
  "target": {
    "namespace": "android_app",
    "package_name": "space.hitchhacker",
    "sha256_cert_fingerprints": ["14:6D:E9:83:---SNIP---:A0:83:42:E6:1D:BE:A8:8A:E5"]
  }
}]
```

Handling app is specified in Digital Asset Links file at <https://hitchhacker.space/.well-known/assetlinks.json>



# The bridge is yours

1. Analyze the Hitchhacker's app directories (using adb) and find out which data is stored where
2. Analyze the Android Manifest and find a way to:
  - Bypass the pin activity
  - ★ Read the contents of the diary

# Solution

## 1. Storage

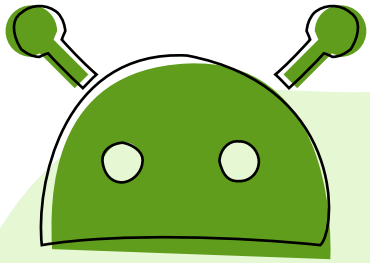
- Search terms: `/storage/emulated/0/Android/data/space.hitchhacker.guide/files/my_files/search.txt`
- Diary entries: `/data/data/space.hitchhacker.guide/databases/diary.db`
- Username: `/data/data/space.hitchhacker.guide/shared_prefs/shared_prefs.xml`
- Pin code: `/data/data/space.hitchhacker.guide/shared_prefs/secret_shared_prefs.xml`

## 2. Bypass pin activity

```
$> adb shell am start -a android.intent.action.MAIN -n space.hitchhacker.guide/.PinActivity
```

## 3. Read content of the diary

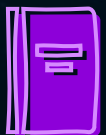
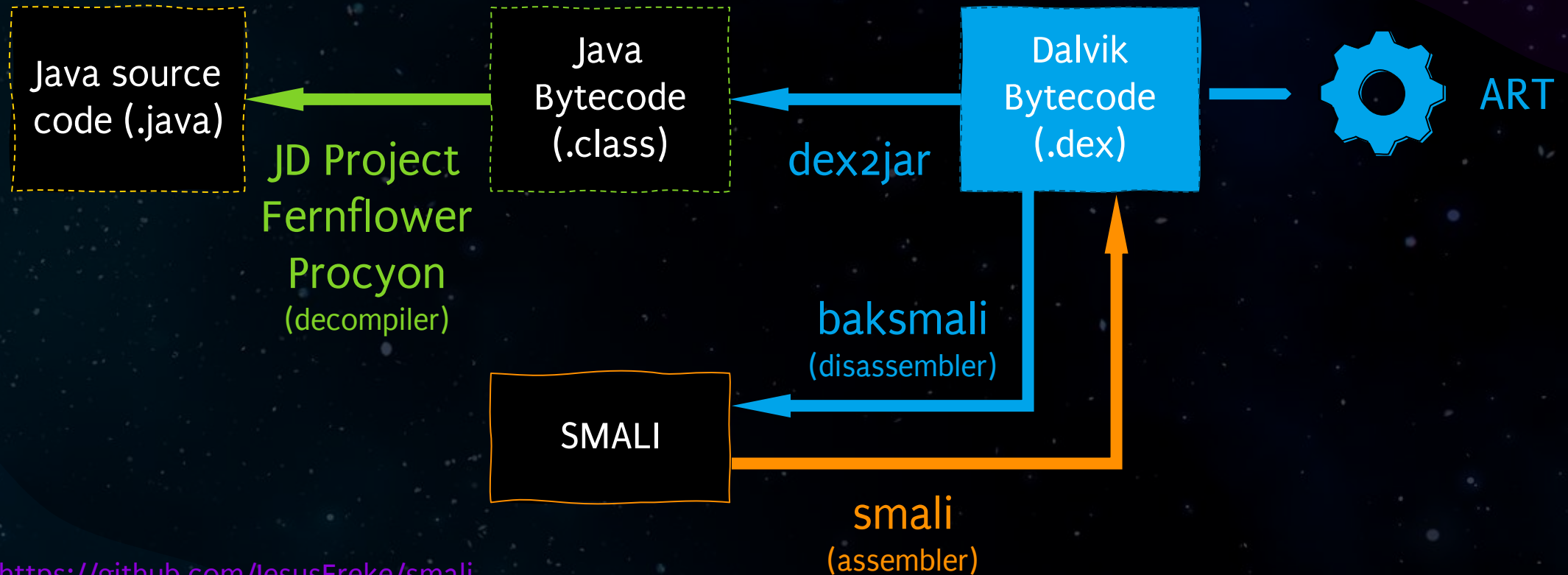
```
$> adb shell content query -uri "content://space.hitchhacker.guide.contentprovider/diary/"
```



# Journey to Deep Space

Reverse engineering

# Take apart and rebuild



<https://github.com/JesusFreke/smali>  
<https://github.com/pxb1988/dex2jar>  
<http://java-decompiler.github.io/>



# Take apart and rebuild



Disassemble

```
$> apktool d hitchhacker.apk
I: Using Apktool 2.5.0-dirty on app-debug.apk
I: Loading resource table...
...
```

Reassemble

```
$> apktool b hitchhacker
I: Using Apktool 2.5.0-dirty
I: Checking whether sources has changed...
...
```

<https://ibotpeaches.github.io/Apktool/>

# The Android Babel fish - Smali



```
$> apktool d hitchhacker.apk
```



assets



AndroidManifest.xml



lib



apktool.yml



original



res



smali

# The Android Babel fish - Smali



```
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView tv =
        (TextView)findViewById(R.id.message);
        tv.setText("Don't Panic");
    }
}
```



```
.class public Lgalaxy/hitchhiker/dontpanic/MainActivity;
[...]
.line 14
const v0, 0x7f0800e2
invoke-virtual {p0, v0},
    Lgalaxy/hitchhiker/dontpanic/MainActivity;->
    findViewById(I)Landroid/view/View;
move-result-object v0
check-cast v0, Landroid/widget/TextView;

.line 15
.local v0, "tv":Landroid/widget/TextView;
const-string v1, "Don't Panic"
invoke-virtual {v0, v1}, Landroid/widget/TextView;->
    setText(Ljava/lang/CharSequence;)V

.line 16
return-void
.end method
```

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;
const-string v1, "Life, The Universe, and Everything"

invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z
move-result v2

if-eqz v2, :cond_0
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getAnswer(
    Ljava/lang/String;)Ljava/lang/Integer;
goto :goto_0

:cond_0
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;
    ->getNoAnswer()Ljava/lang/String;

:goto_0
return-void
```

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;
const-string v1, "Life, The Universe, and Everything"

invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z
move-result v2

if-eqz v2, :cond_0
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getAnswer(
    Ljava/lang/String;)Ljava/lang/Integer;
goto :goto_0

:cond_0
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;
    ->getNoAnswer()Ljava/lang/String;

:goto_0
return-void
```

Simple data type  
marked by an  
uppercase letter

V	void
Z	Boolean
B	byte
C	char
I	int
F	float
S	short
J	long

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

```
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

```
if-eqz v2, :cond_0  
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getAnswer(  
    Ljava/lang/String;)Ljava/lang/Integer;  
goto :goto_0
```

```
:cond_0  
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
    ->getNoAnswer()Ljava/lang/String;
```

```
:goto_0  
return-void
```

Complex data types are  
denoted as:  
**L<fully qualified name>;**

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

```
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

Data stored in local registers vX

```
if-eqz v2, :cond_0  
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getAnswer(  
    Ljava/lang/String;)Ljava/lang/Integer;  
goto :goto_0
```

```
:cond_0  
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
    ->getNoAnswer()Ljava/lang/String;
```

Parameter register pX  
p0 = "this"

```
:goto_0  
return-void
```

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

Directive for  
method call

Fully qualified  
class name

```
invoke-direct {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

```
if-eqz v2, :cond_0
```

```
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
->getAnswer(Ljava/lang/String;)Ljava/lang/Integer;
```

Called  
method

Type of argument  
passed to method

Type of  
return value

```
goto :goto_0
```

```
:cond_0  
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
->getNoAnswer()Ljava/lang/String;
```

```
:goto_0  
return-void
```



# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

Registers with values

passed to the method call

```
if-eqz v2, :cond_0
```

```
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;
```

```
->getAnswer(Ljava/lang/String;)Ljava/lang/Integer;
```

```
goto :goto_0
```

```
:cond_0
```

```
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;
```

```
->getNoAnswer()Ljava/lang/String;
```

```
:goto_0
```

```
return-void
```

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

```
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

```
if-eqz v2, :cond_0  
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getAnswer(  
    Ljava/lang/String;)Ljava/lang/Integer;  
goto :goto_0
```

```
:cond_0  
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
    ->getNoAnswer()Ljava/lang/String;
```

```
:goto_0  
return-void
```

Return value of  
method call is  
stored in register

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

```
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

Private method call

```
if-eqz v2, :cond_0
```

```
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
    ->getAnswer(Ljava/lang/String;)Ljava/lang/Integer;
```

```
goto :goto_0
```

Public method call

```
:cond_0
```

```
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
    ->getNoAnswer()Ljava/lang/String;
```

```
:goto_0  
return-void
```

# Smali crash course



```
sget-object v0, Ljava/lang/Boolean;->TRUE:Ljava/lang/Boolean;  
const-string v1, "Life, The Universe, and Everything"
```

```
invoke-virtual {v0}, Ljava/lang/Boolean;->booleanValue()Z  
move-result v2
```

```
if-eqz v2, :cond_0
```

Conditional jump

```
invoke-direct {p0, v1}, Lgalaxy/hitchhiker/dontpanic/MainActivity;->getNoAnswer(Ljava/lang/String;)Ljava/lang/Integer;  
goto :goto_0
```

```
:cond_0
```

```
invoke-virtual {p0}, Lgalaxy/hitchhiker/dontpanic/MainActivity;  
->getNoAnswer()Ljava/lang/String;
```


```
:goto_0  
return-void
```

if-eq v1, v2	v1 == v2
if-ne v1, v2	v1 != v2
if-lt v1, v2	v1 < v2
if-gt v1, v2	v1 > v2
if-le v1, v2	v1 <= v2
if-ge v1, v2	v1 >= v2
if-eqz v2	v2 == 0
if-nez v2	v2 != 0
if-ltz v2	v2 < 0
if-gtz v2	v2 > 0
if-lez v2	v2 <= 0
if-gez v2	v2 >= 0

# VSCoDe & Smalise



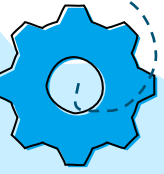
```
File Edit Selection View Go Run Terminal Help
EXPLORER
BASE
  > kotlin
  > META-INF
  > original
  > res
  > smali
  > smali_classes2
smali_classes3 / space / hitchhacker / guide
  BuildConfig.smali
  MainActivity.smali
  unknown
  AndroidManifest.xml
  ! apktool.yml
MainActivity.smali
41
42 # virtual methods
43 .method protected onCreate(Landroid/os/Bundle;V
44   .locals 2
45   .param p1, "savedInstanceState" # Landroid/os/Bundle;
46
47   .line 9
48   invoke-super {p0, p1}, Landroidx/appcompat/app/AppCompatActivity;->onCreate(Landroid/os/Bundle;)
49
50   .line 10
51   const v0, 0x7f0b001c
52
53   invoke-virtual {p0, v0}, Lspace/hitchhacker/guide/MainActivity;->setContentView(I)V
54
55   .line 11
56   const v0, 0x7f080107
57
58   invoke-virtual {p0, v0}, Lspace/hitchhacker/guide/MainActivity;->findViewById(I)Landroid/view/View;
59
60   move-result-object v0
61
62   const-string v1, "findViewById(R.id.message)"
63
64   invoke-static {v0, v1}, Lkotlin/jvm/internal/Intrinsics;->checkNotNullExpressionValue(Ljava/lang/Object;)V
65
66   check-cast v0, Landroid/widget/TextView;
67
```



**Smalise** v0.0.12  
Loyie King | 63,741 | ★★★★★ (4)  
Language support for Smali (Davilk bytecode)

Provides syntax highlighting and helps finding code references

# Modify and reassemble



Reassemble the APK

```
$> apktool b hitchhacker
I: Checking whether sources has changed...
...
```

Create a key pair for signing

```
$> keytool -genkey -v -keystore foo.keystore -alias keyalias -
keyalg RSA -keysize 2048 -validity 10000
Enter keystore password:
...
```

Sign the APK

```
$> jarsigner -sigalg SHA1withRSA -digestalg SHA1 -keystore
foo.keystore hitchhacker.apk keyalias
Enter Passphrase for keystore:
...
```

Align the APK

```
$> zipalign -v 4 hitchhacker.apk hitchhacker-aligned.apk
Verifying alignment of hitchhacker-aligned.apk (4)...
```

# But we like being lazy



Sign and zipalign the APK using the integrated debug keystore

Sign the APK with your own keystore

```
$> java -jar uber-apk-signer.jar --apks hitchhacker.apk
```

```
$> java -jar uber-apk-signer.jar --apks hitchhacker.apk --ks  
foo.keystore --ksAlias keyalias
```

<https://github.com/patrickfav/uber-apk-signer>



# The bridge is yours

Patch the Hitchhacker app so you can log in without knowing the pin code

```
Decompile:apktool d -r hitchhacker.apk
```

```
Recompile:apktool b --use-aapt2 -o fixed_hitchhacker.apk  
hitchhacker
```

```
Sign: java -jar uber-apk-signer.jar --apks  
fixed_hitchhacker.apk
```

- ★ Install and launch the Vagon Construction application. Patch the app to get around the login restrictions



# Solution

## 1. Patch Hitchhacker app

```
$> java -jar /usr/local/bin/apktool.jar b --use-aapt2 -o  
fixed_hitchhacker.apk hitchhacker  
$> java -jar uber-apk-signer.jar --apks  
fixed_hitchhacker.apk  
$> adb install fixed_hitchhacker-aligned-debugSigned.apk
```

```
mul-int/2addr v2, v1  
const/16 v1, 0x2a  
if-ne v0, v1, :cond_2  
if-ne v2, v1, :cond_2  
move v0, v4  
goto :goto_1  
:cond_2  
const/4 v0, 0x0  
:goto_1  
return v0  
.end method
```

```
mul-int/2addr v2, v1  
const/16 v1, 0x2a  
if-ne v0, v1, :cond_2  
if-ne v2, v1, :cond_2  
move v0, v4  
goto :goto_1  
:cond_2  
const/4 v0, 0x1  
:goto_1  
return v0  
.end method
```

## 2. Patch Vagon Construction app

```
if(!isVagonOperator()) {...}
```

```
.line 29  
invoke-direct {p0}, Leu/nviso/vagonconstruction/MainActivity;->isVagonOperator()Z  
move-result p1  
if-nez p1, :cond_33  
.line 30  
new-instance p1, Landroidx/appcompat/app/AlertDialog$Builder;  
const v0, 0x7f100002  
invoke-direct {p1, p0, v0}, Landroidx/appcompat/app/AlertDialog$Builder;-><init>(Lan  
invoke-virtual {p1}, Landroidx/appcompat/app/AlertDialog$Builder;->create()Landroidx
```

```
.end method  
.method protected onCreate(Landroid/os/Bundle;)V  
.locals 3  
.line 26  
invoke-super {p0, p1}, Landroidx/appcompat/app/AppCompatActivity;~  
const p1, 0x7f0b001e  
.line 27  
invoke-virtual {p0, p1}, Leu/nviso/vagonconstruction/MainActivity;  
.line 29  
invoke-direct {p0}, Leu/nviso/vagonconstruction/MainActivity;->isV  
move-result p1  
if-eqz p1, :cond_0  
.line 30  
new-instance p1, Landroidx/appcompat/app/AlertDialog$Builder;
```

# Our Journey



Adventures  
on Android



Meddling in  
the Middle

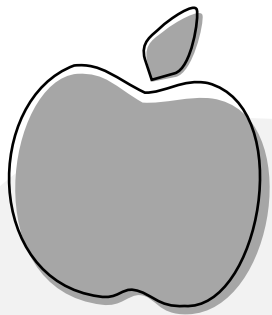


Big Bang  
of Basics



Incidents  
on iOS

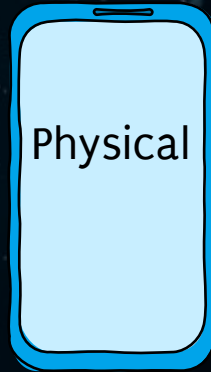




# Boarding the SpaceShip

Devices & jailbreaking

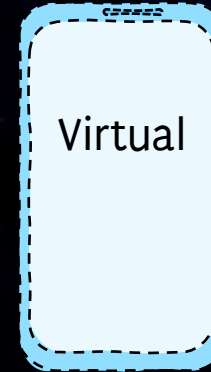
# Getting a testing device



Physical

Usually only older iOS versions can be jailbroken

Cost factor

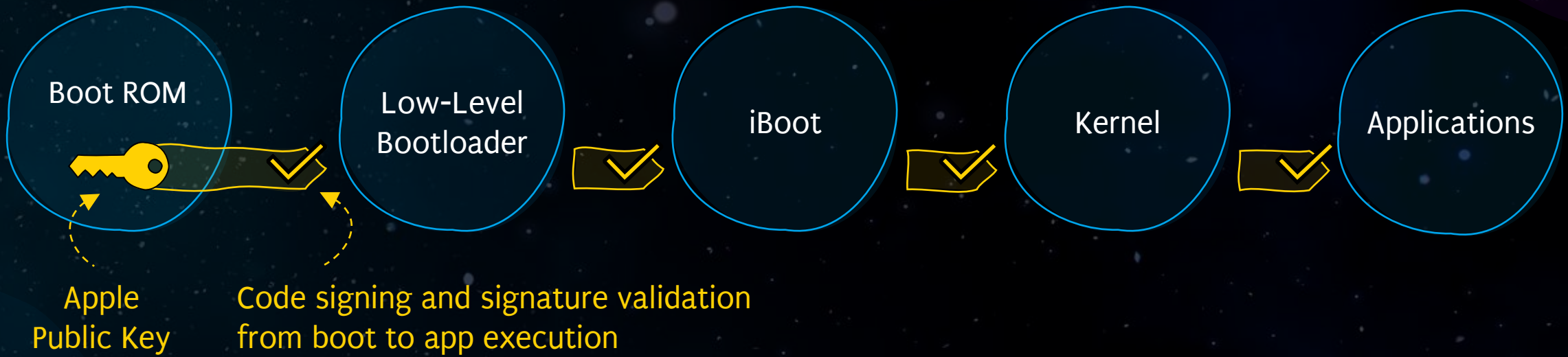


Virtual

iOS simulator in Xcode cannot run apps from App Store

Limited other options due to Apple's closed environment

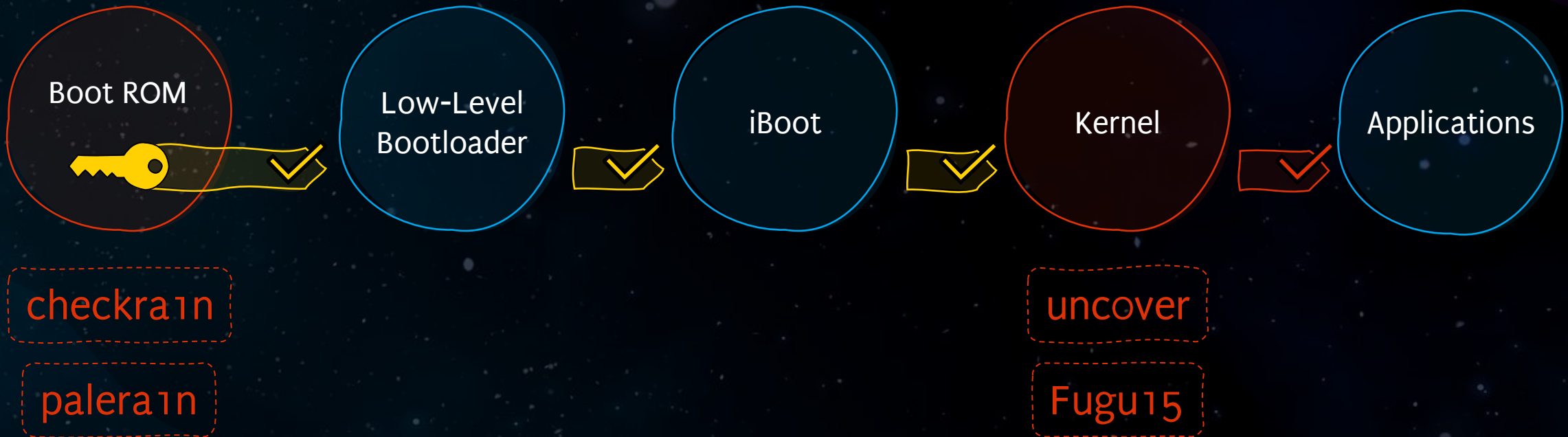
# Jailbreaking



# Jailbreaking



A jailbreak nullifies one or more signature validation checks



# Jailbreaking



checkra1n

iPhone 5s - iPhone X  
iOS 12.3 - 14.8

<https://checkra.in/>



palera1n

iPhone 5s - iPhone X  
iOS 15.0 - 16.x

[https://github.com/  
palera1n/palera1n](https://github.com/palera1n/palera1n)



uncover

iOS 11.0 - 14.8

<https://uncover.dev/>

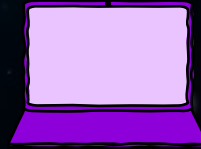


Fugu15

iOS 15.0 - 15.4.1

[https://github.com/  
pinauten/Fugu15](https://github.com/pinauten/Fugu15)

# Jailbreaking



Host needed  
for jailbreak



Jailbreak  
without host

Reboots  
jailbroken



Tethered

Untethered

Reboots  
non-jailbroken

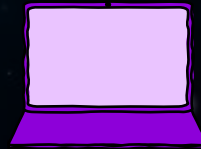


Semi-Tethered

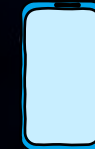
Semi-Untethered



# Jailbreaking



Host needed  
for jailbreak



Jailbreak  
without host

Reboots  
jailbroken



**Tethered**

palera1n

**Untethered**

Reboots  
non-jailbroken



**Semi-Tethered**

checkra1n

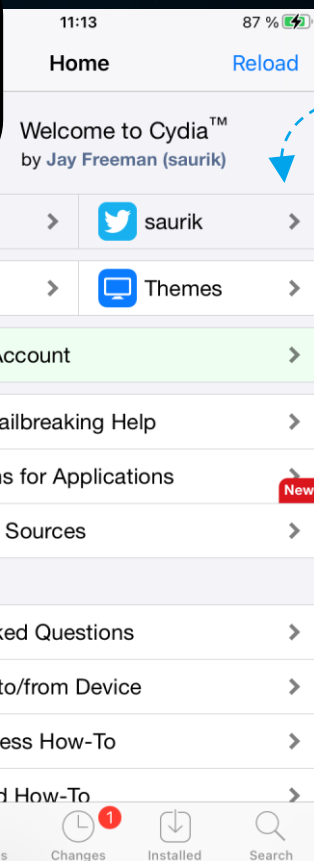
palera1n

**Semi-Untethered**

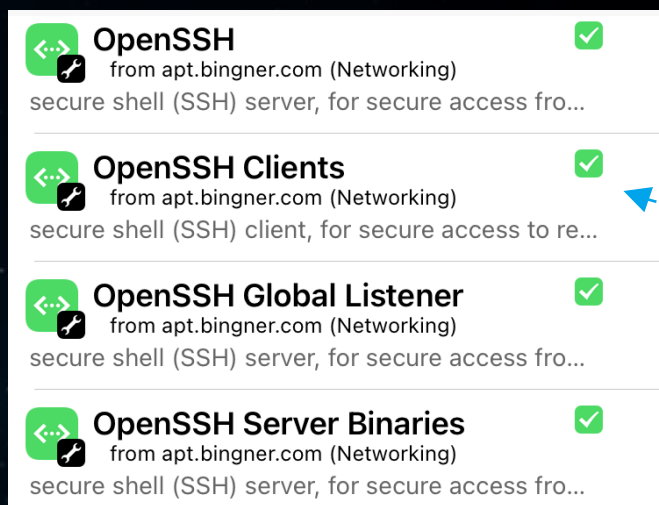
Fugu15

uncover

# Gear Up

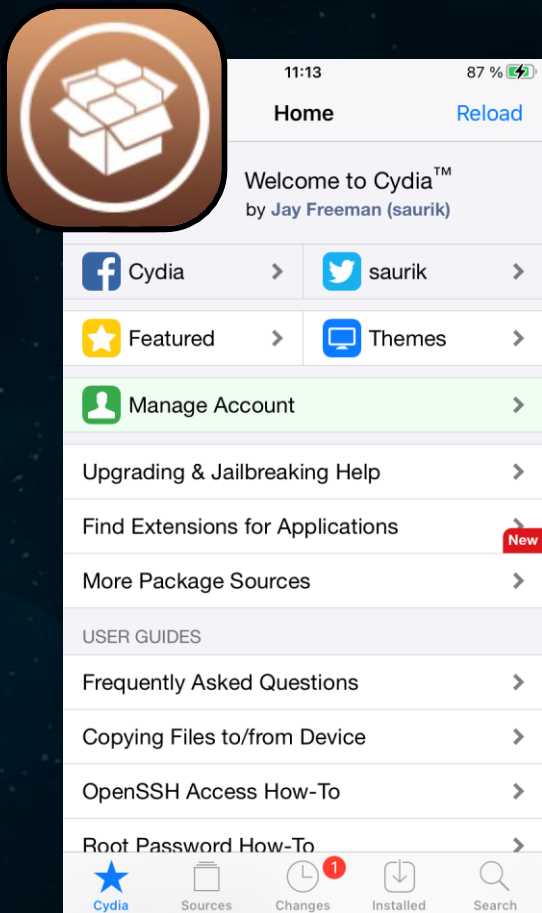


Cydia App Store is included in most jailbreaks and allows us to install useful tools for pentesting



Most jailbreaks install an SSH server on the device. If not, you can install it via Cydia

# Gear Up



**Frida** – [build.frida.re](https://build.frida.re)

Frida server to observe and reprogram running apps



**BigBoss tools** – [apt.thebigboss.org](https://apt.thebigboss.org)

Useful tools, e.g. zip, unzip, sqlite3, wget, OpenSSH etc.



**Filza** – [cydia.akemi.ai](https://cydia.akemi.ai)

File manager and IPA installer



**AppSync Unified** – [cydia.akemi.ai](https://cydia.akemi.ai)

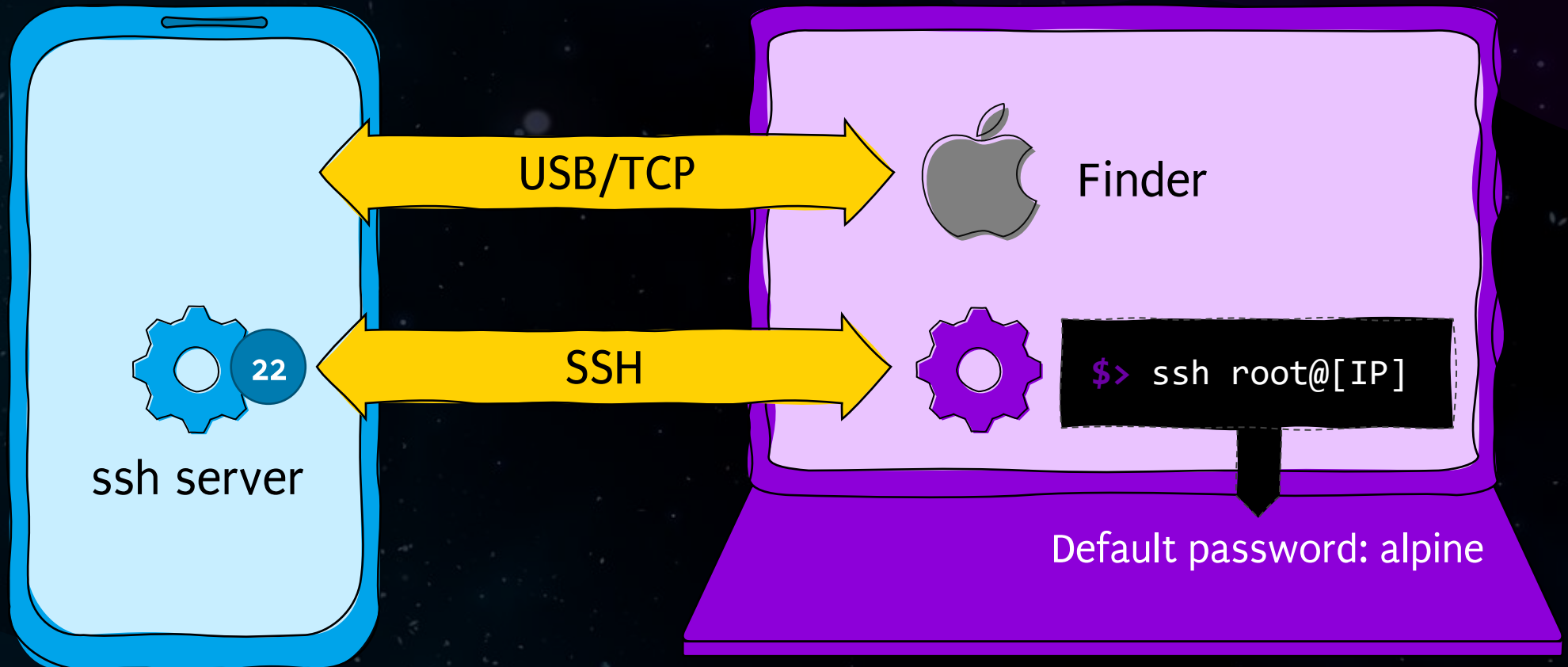
Disable signature requirements for installing apps



**iHide** – [repo.kc57.com](https://repo.kc57.com)

Bypass jailbreak detection

# Reaching out




# Exploring the planet




*the content of the IPA*


`/private/var/containers/Bundle/Application/[appID]`

 Payload

`/var/mobile/Containers/Data/Application/[appID]`

 Documents

 Library

 SystemData

 tmp

# Exploring the planet

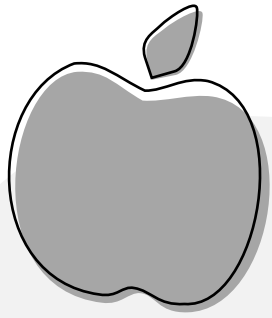


```
iPhone:~ root# ipainstaller -l  
com.apple.TestFlight  
com.swiftkey.SwiftKeyApp  
space.hitchhacker.Hello
```

IPA Installer Console can be installed via Cydia

```
...  
iPhone:~ root# ipainstaller -i space.hitchhacker.Hello  
Identifier: space.hitchhacker.Hello  
Version: 1  
Short Version: 1.0  
Name: Hello  
Display Name: Hello  
Bundle: /private/var/containers/Bundle/Application/6A745F9C-D991-43E4-9C4A-0F380CD18D9B  
Application: /private/var/containers/Bundle/Application/6A745F9C-D991-43E4-9C4A-0F380CD18D9B/Hello.app  
Data: /private/var/mobile/Containers/Data/Application/BEA183DA-19AF-48D8-A5C8-F1F22C7CF48E
```

Locates local files related to the application



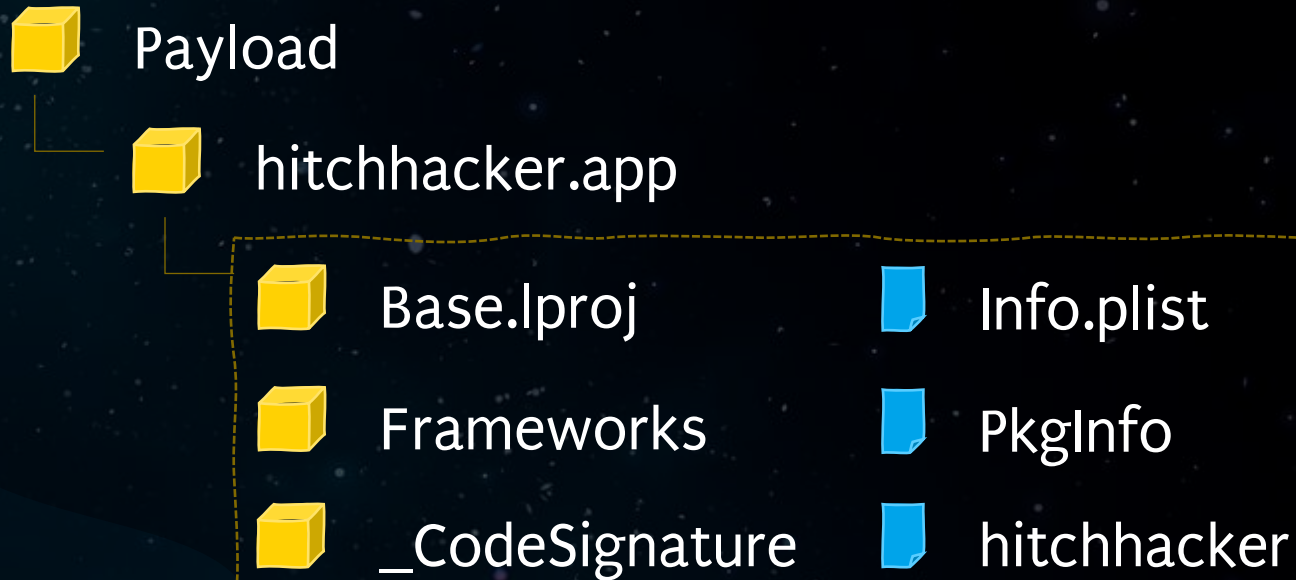
# Rocket Science

Inside an IPA



# InSide an IPA

```
$> unzip hitchhacker.ipa -d hitchhacker  
$> ls -la hitchhacker
```

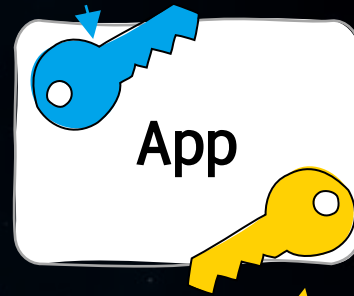




# Signing off



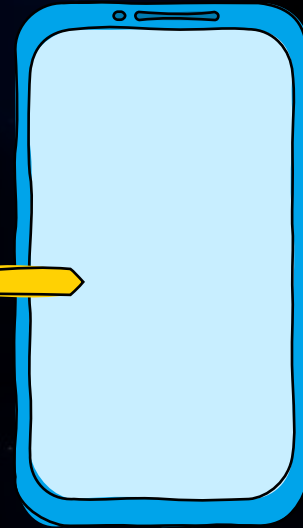
Developers build and sign app



Apple approves the app and signs it with their private key



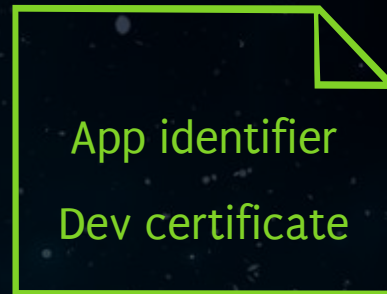
Upon installation, iOS checks the app signatures and refuses app in case of invalid signatures



# Signing off



For testing purposes apps can be distributed with a provisioning profile instead of being signed with Apple's private key



## Ad-Hoc Certificate

- ▶ Free developer account
  - Valid for 7 days
  - For up to 100 devices
- ▶ Paid developer account (\$99/year)
  - Valid for 1 year
  - For up to 100 devices

## Enterprise Certificate

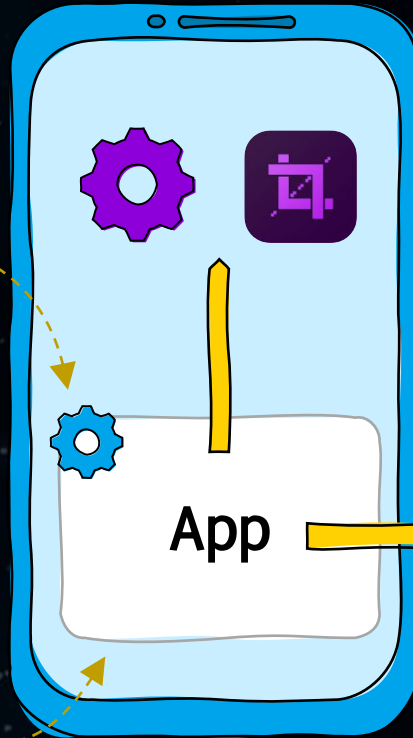
- \$299/year for companies with more than 100 employees
- Valid for 3 years
- Unlimited devices

# Obtaining an IPA



Apps are encrypted, so we need to obtain the IPA when the app is decrypted while running

We need a jailbroken device

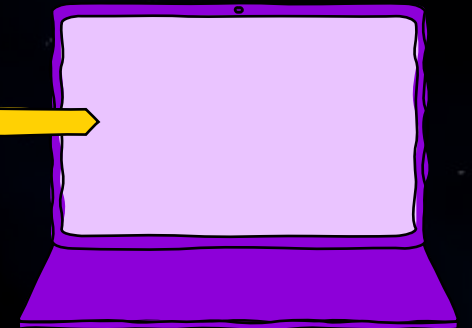


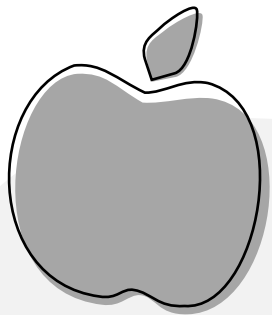
## Dump on phone

<https://github.com/JohnCoates/flexdecrypt>  
<https://onejailbreak.com/blog/crackerxi/>

## Dump to host

<https://github.com/AloneMonkey/frida-ios-dump>





# The Cargo Bay

How data is stored on iOS

# File protection

Data object with  
file content

```
do {  
  try data.write(  
    to: fileURL,  
    options: .completeFileProtection  
  )  
} catch {  
  // Handle errors  
}
```

Protection level that determines  
encryption of file

# Protection levels

`.noFileProtection`

Unencrypted  
Reading and writing always possible

`.completeFileProtection`

Encrypted  
Reading and writing only possible if device is unlocked

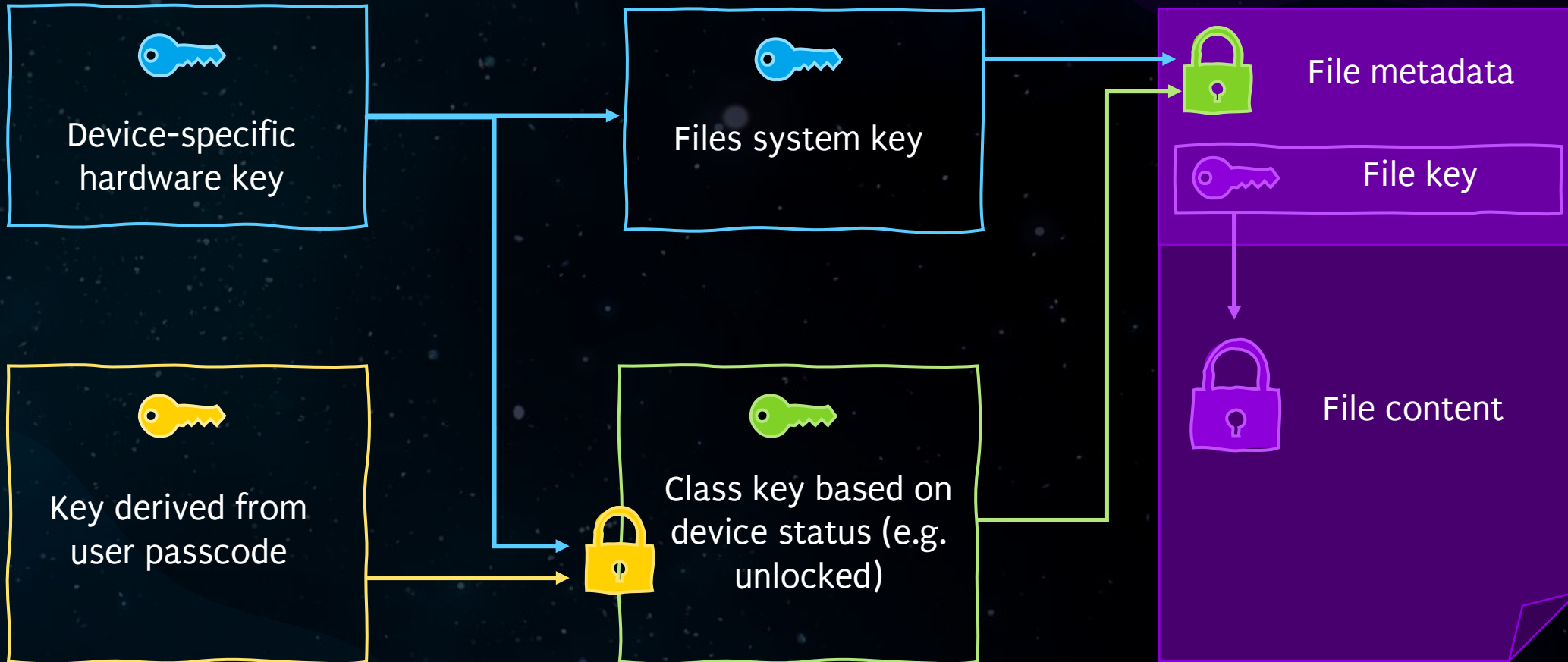
`.completeFileProtectionUnlessOpen`

Encrypted  
Reading and writing only possible if device is unlocked;  
files remains available for background use

`.completeFileProtectionUntilFirst  
UserAuthentication` (default)

Encrypted  
Reading and writing only possible if device has been  
unlocked once after a reboot

# File encryption



# NSUserDefaults

To store small pieces of information

```
// Write
NSUserDefaults.standard.set("42", forKey: "UltimateAnswer")

// Read
NSUserDefaults.standard.string(forKey: "UltimateAnswer")
```



# plist files



```
iPhone:/private/var/.../Library/Safari root# file *.plist
FrequentlyVisitedSitesBannedURLStore.plist: XML 1.0 document, ASCII text
PasswordBreachStore.plist: Apple binary property list
```

Binary or  
XML data

# plist files



```
$> plistutil -i PasswordBreachStore.plist -o Store.out
```

Convert binary  
plist to XML

```
$> head Store.out
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>KeychainPersistentIdentifierCanaryValue</key>
  <string>6F30D983-A91B-4337-9D78-5FE72480B153</string>
  <key>Version</key>
  <integer>1</integer>
  <key>KeychainPersistentIdentifierCanaryPersistentIdentifier</key>
  <data>
```

Key-value  
pairs

# Databases



SQLite



Realm



Firebase



Couchbase Lite

———— *same as for Android* ————

NoSQL database

Not encrypted by default

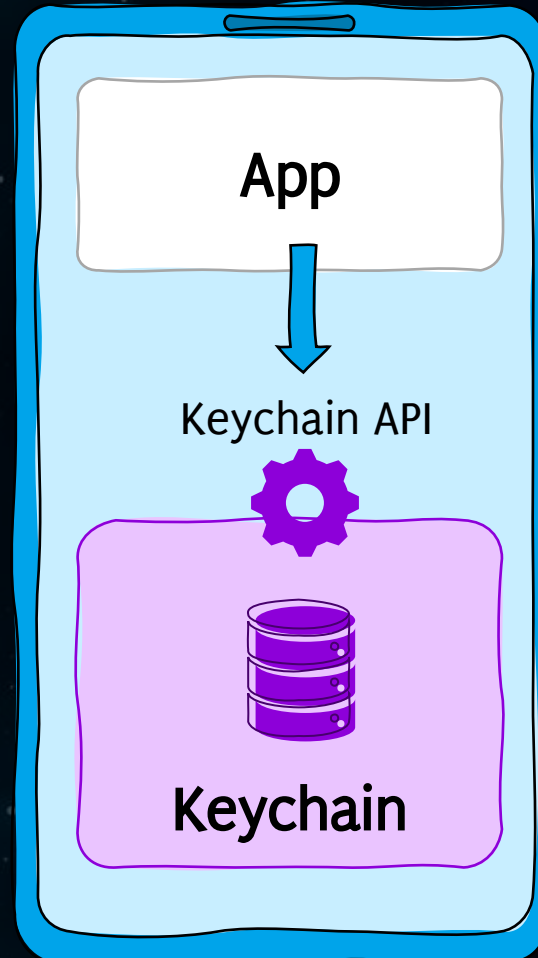
Encryption can be enabled  
through configuration

# Keychain



Encrypted SQLite  
database

Single Keychain  
for all apps



Data can be stored with an Accessibility attribute that determines when the data becomes accessible

- Always
- After unlocking the device once after restart
- When device is unlocked
- Only when passcode is set for device

# Data access

```
let username = "arthur"
let password = "acupoftea".data(using: .utf8)!
let query: [String: Any] = [
    kSecClass as String: kSecClassGenericPassword,
    kSecAttrAccount as String: username,
    kSecValueData as String: password,
    kSecAttrAccessible as String: kSecAttrAccessibleWhenUnlocked
]
let status = SecItemAdd(query as CFDictionary, nil)
guard status == errSecSuccess else { throw KeychainError.unhandledError(status: status) }
```

Data can be stored with an Accessibility attribute which specifies when access to the data item is possible

# Data access



**Always** (deprecated)

Always accessible

**AfterFirstUnlock**

Accessible if device has been unlocked once after a reboot

**WhenUnlocked**

Accessible if the device is currently unlocked

**WhenPasscodeSetThisDeviceOnly**

Accessible if the device is currently unlocked and the user has set up a passcode for the device

**...ThisDeviceOnly**

For a backup, data will be encrypted with a device-specific key; thus it cannot be restored on a different device

# Access control



```
let access = SecAccessControlCreateWithFlags(nil,  
      kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly,  
      .userPresence,  
      nil)
```

Access control instance defines accessibility and authentication requirements of a keychain item

```
let query: [String: Any] = [kSecClass as String: kSecClassInternetPassword,  
      kSecAttrAccount as String: account,  
      kSecAttrServer as String: server,  
      kSecAttrAccessControl as String: access as Any,  
      kSecUseAuthenticationContext as String: context,  
      kSecValueData as String: password]
```

# Access control



**userPresence**

Require biometric authentication or fall back to passcode if biometry is not available

**biometryAny**

Require biometric authentication with any enrolled biometrics (Touch or Face ID), i.e. also biometric features that were enrolled after the creation of the keychain item

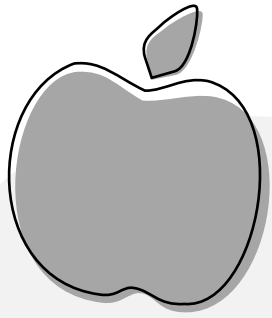
**biometryCurrentSet**

Require biometric authentication with *currently enrolled* biometrics (Touch or Face ID). If biometric features are added or removed, the item becomes invalid

**devicePasscode**

Require device passcode





# Static analysis

Opening the binary

# Creating iOS apps



The original language

Highly dynamic

Still used, but less and less

RE friendly



New kid on the block

Compiles to normal C

Used more and more

Difficult to RE

# Hopper



Hopper Disassembler v4

File Edit Find Modify Navigate Scripts Debug Window Help

Document \*Hello\_dec

Search

Labels Strs Procs Bookmarks Bkpts

Tags

Address	Type	Name
0x1000075b	P	partial apply forwarder for reabs
0x1000075c	P	lazy protocol witness table acces
0x10000762	P	Hello.HelloApp.init() -> Hello.He
0x10000762	P	static Hello.HelloApp.\$main() ->
0x10000765	P	lazy protocol witness table acces
0x1000076b	P	protocol witness for SwiftUI.App
0x1000076d	P	protocol witness for SwiftUI.App
0x1000076e	P	<b>_main</b>
0x1000076f	P	associated type witness table ac
0x10000772	P	type metadata accessor for Helle
0x10000773	P	sub_100007734
0x10000774	P	SwiftUI.WindowGroup.init(conten
0x10000775	P	static SwiftUI.SceneBuilder.build
0x10000776	P	static (extension in SwiftUI):Swif
0x10000776	P	static (extension in SwiftUI):Swif
0x10000777	P	static (extension in SwiftUI):Swif
0x10000777	P	static (extension in SwiftUI):Swif
0x10000778	P	SwiftUI.LocalizedStringKey.init(s
0x10000779	P	static (extension in SwiftUI):Swif
0x10000779	P	static SwiftUI.Edge.Set.all.getter
0x1000077a	P	static SwiftUI.Font.largeTitle.get
0x1000077b	P	SwiftUI.Text.font(SwiftUI.Font?)
0x1000077c	P	SwiftUI.Text.init(_: SwiftUI.Locali
0x1000077c	P	static (extension in SwiftUI):Swif
0x1000077d	P	static (extension in SwiftUI):Swif
0x1000077e	P	static (extension in SwiftUI):Swif
0x1000077e	P	static (extension in SwiftUI):Swif
0x1000077f	P	(extension in SwiftUI):SwiftUI.Vie
0x1000077f	P	Swift.String.init(_builtinStringLit

```
int main() {
    static Hello.HelloApp.$main();
    return 0x0;
}

; ----- BEGINNING OF PROCEDURE -----
; Variables:
;   saved_fp: 0

_main:
00000001000075e4 stp    fp, lr, [sp, #-0x10]!
00000001000076e8 mov    fp, sp
00000001000076ec bl     _$5Hello0A3AppV5$mainyyFZ ; static Hello.HelloApp.$main() ->
00000001000076f0 mov    w0, #0x0
00000001000076f4 ldp   fp, lr, [sp], #0x10
00000001000076f8 ret

; ----- BEGINNING OF PROCEDURE -----
; Variables:
;   saved_fp: 0

_$5Hello0A3AppV7SwiftUI080AA4BodyAdEP_AD5ScenePWT: // associated type witness table a
00000001000076fc stp    fp, lr, [sp, #-0x10]!
0000000100007709 mov    fp, sp
000000010000770d adrp   x1, #0x100007000 ; 0x100007d78@PAGE
0000000100007711 add    x1, x1, #0xd78 ; 0x100007d78@PAGEOFF, argument "d
0000000100007715 mov    w8, #0x1
0000000100007719 mov    x2, x8 ; argument "index" for method imp_
000000010000771d bl     imp__stubs__swift_getOpaqueTypeConformance ; swift_getOpaqueTypeConformance
0000000100007721 ldp   fp, lr, [sp], #0x10
0000000100007725 ret
```

File Information

Path: /home/kali/Documents/Hello\_dec

Loader: Mach-O

CPU: aarch64

Branch always stops procedures

CPU Syntax Variant: Generic

Calling Convention: AAPCS

Address Information

Type: Procedure

Prolog Heuristic: Not a procedure prolog

Prolog Mode: Use Heuristic

Navigation History

0x1000076e (\_main + 0x8)

0x10000764 (\_\$5Hello0A3AppV5\$mainyyFZ)

Clear Navigation Stack

Graphic Views

Type: Entropy

From: 0 Cur. Pos.

To: 111840 Cur. Pos.

Section Segment Whole File

Address 0x1000076e4, Segment \_\_TEXT, \_main + 0, Section \_\_text, file offset 0x76e4 - Alt+Double-Click to follow link in a new pane

# Ghidra



The screenshot displays the Ghidra CodeBrowser interface for a file named 'Hello/Hello\_enc'. The interface is divided into several panes:

- Program Trees:** Shows the project structure with folders for Hello\_enc, \_TEXT, \_DATA\_CONST, \_DATA, \_LINKEDIT, and EXTERNAL.
- Symbol Tree:** Lists symbols including Imports, Exports, Functions, entry, FUN\_100007, Labels, Classes, and Namespaces.
- Data Type Manager:** Shows data types such as BuiltInTypes, Hello\_enc, and mac\_osx.
- Listing:** Displays assembly code for the 'entry' function. The code includes instructions like 'stp x29,x30,[sp,#local\_10]!', 'mov x29,sp', 'bl \_\$SHello0A3AppV5\$mainyFZ', 'mov w0,#0x0', and 'ldp x29=>local\_10,x30,[sp],#0x10'. It also shows function pointers and return values.
- Decompile:** Shows the decompiled C code for the 'entry' function: 

```
undefined8 entry(void)
{
    _$SHello0A3AppV5$mainyFZ();
    return 0;
}
```
- Console - Scripting:** An empty area for running scripts.

The status bar at the bottom indicates the current instruction: '1000076e4 entry stp x29,x30,[sp,#-0x10]!'.

# Decompiled ObjC / C



Cf Decompile: addKey:withValue: - (Playground)

```
1
2 /* Function Stack Size: 0x20 bytes */
3
4 void VulnerableVault::addKey:withValue:(ID param_1,SEL param_2,ID param_3,ID param_4)
5
6 {
7     id value;
8     id pvVar1;
9     id value_00;
10    CFDictionaryRef attributes;
11
12    value = _objc_retain((id)param_3);
13    pvVar1 = (id)_objc_msgSend(param_4,"dataUsingEncoding:",4);
14    pvVar1 = _objc_retainAutoreleasedReturnValue(pvVar1);
15    value_00 = (id)prepareDict:(param_1,(SEL)"prepareDict:",(ID)value);
16    attributes = (CFDictionaryRef)_objc_retainAutoreleasedReturnValue(value_00);
17    _objc_release(value);
18    _objc_msgSend(attributes,"setObject:forKey:",pvVar1,* (undefined8 *)PTR_kSecValueData_1000104a8);
19    _objc_msgSend(attributes,"setObject:forKey:",&cf_VulnerableVaultService,
20                *(undefined8 *)PTR_kSecAttrService_100010490);
21    _SecItemAdd(attributes,(CTypeRef *)0x0);
22    _objc_release(attributes);
23    _objc_release(pvVar1);
24    return;
25 }
26
```

# Our Journey



Adventures  
on Android



Meddling in  
the Middle



Big Bang  
of Basics



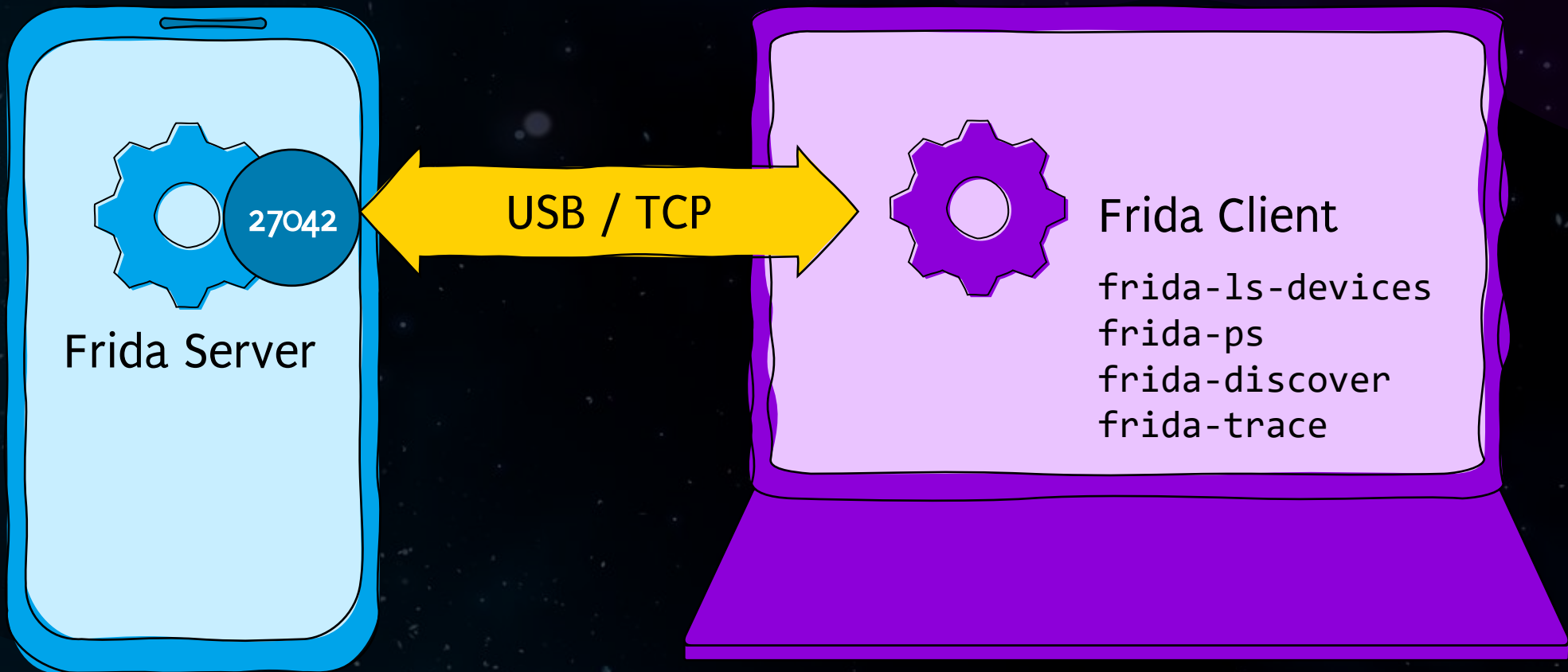
Incidents  
on iOS



# Igniting the Infinite Improbability Drive

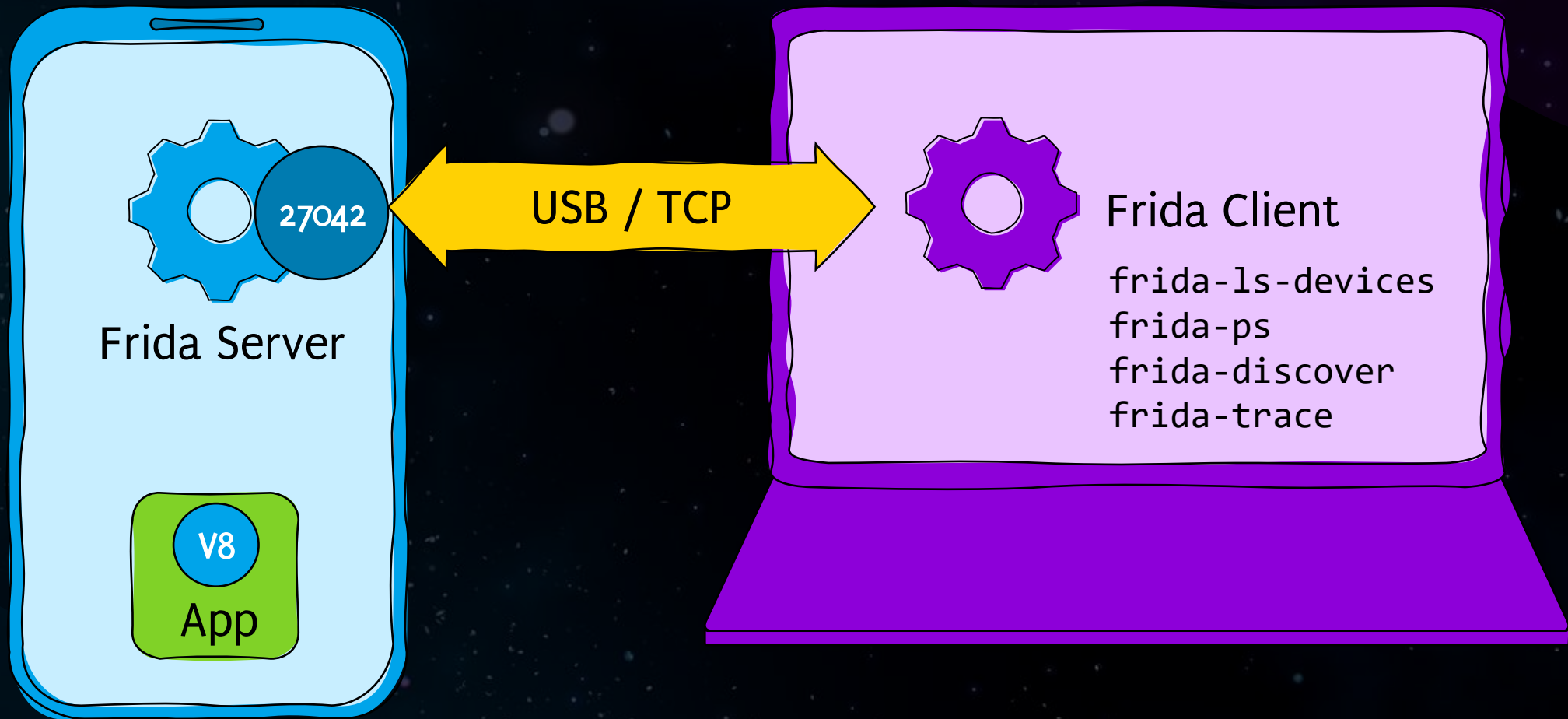
Hooking with Frida

# Frida

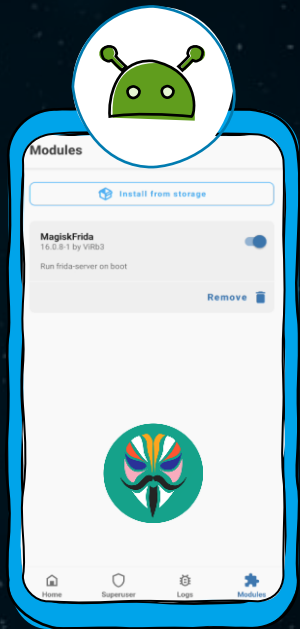
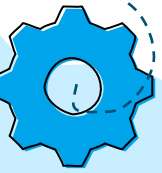




# Frida



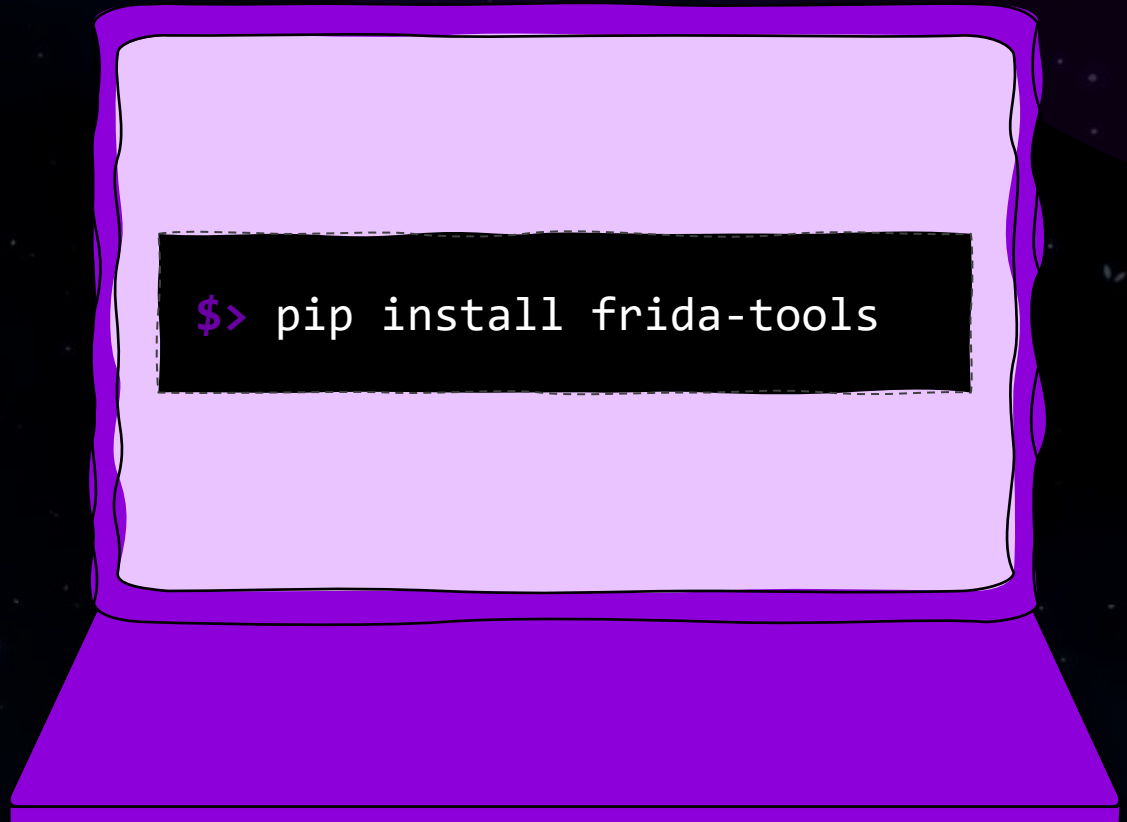
# Frida



[github.com/ViRb3/  
magisk-frida](https://github.com/ViRb3/magisk-frida)

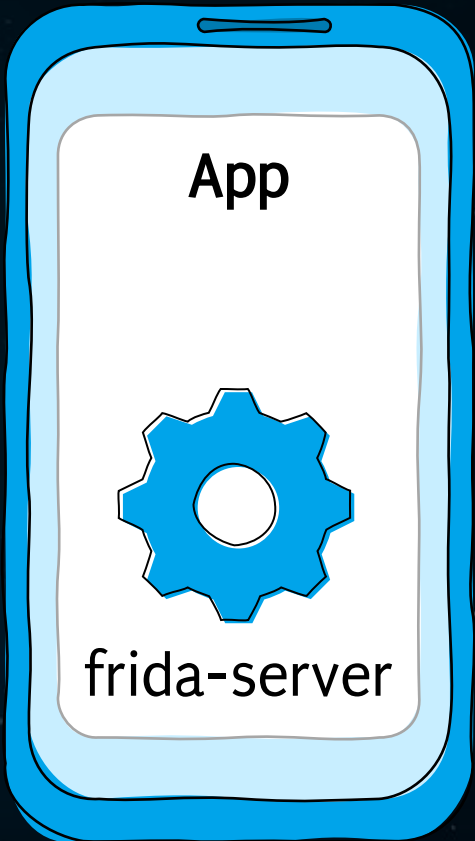


[build.frida.re](https://build.frida.re)



<https://github.com/frida/frida/>

# what to do without root?



On non-rooted/jailbroken devices, we can compile frida-agent.so (Android) / FridaGadget.dylib (iOS) into the app

```
$> objection patchapk --source ./hitchhacker.apk  
  
$> objection patchipa --source "HitchHacker.ipa"  
--codesign-signature XXXXXXXXXXXXXXXX
```

# Frida



```
$> frida-ps -Ua
```

```
PID  Name      Identifier
---  -
946  Calendar  com.apple.mobilecal
985  Poems     space.hitchhacker.guide
```

*Connect via USB*

*List running user-land applications*

```
$> frida -Uf space.hitchhacker.guide
```

```
|_/_/|
|(_|_|
|>_|_|
|_/_/|_|
```

Frida 16.0.2 - 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 iOS Device (id=d7070753257d3c069acf97f642f2de6917061448)

```
[iOS Device::space.hitchhacker.guide]->
```

*Interactive console*

# Frida for Android - Inspect



```
$> [Pixel 3a::space.hitchhacker.guide]-> Java.enumerateLoadedClassesSync()
```

```
[  
  "android.icu.text.UnicodeSet$ComparisonStyle",  
  "android.icu.text.CharsetRecog_mbc$IteratedChar",  
  "android.icu.impl.duration.impl.XMLRecordWriter",  
  ...  
]
```

*List all loaded classes*

```
$> [Pixel 3a::space.hitchhacker.guide]-> Java.enumerateMethods("space.hitchhacker.guide.  
  LoginActivity!*")
```

```
[  
  {  
    "classes": [  
      {  
        "methods": [  
          "$r8$lambda$ABD1GrxfYNKMF77fryH3xXrhFCo",  
          "$init",  
          "launchMain",  
          "validatePin",  
          ...  
        ]  
      }  
    ]  
  }  
]
```

*List all methods using wildcards:  
\*class!\*method\**

# Frida for Android - Scripts



*Our script*

```
$> frida -U -l theanswer.js -f space.hitchhacker.guide
```

*App package name  
that we want to attack*

# Frida Scripts - Analyzing static variables



```
package space.hitchhacker;
public class TheAnswer
{
    public static String question;
    public static String getAnswer()
    {
        String answer = calculate(this.question);
        return answer;
    }
}
```

*We want to know the value of the static variable "question"*

# Frida Scripts - Analyzing static variables



```
package space.hitchhacker;
public class TheAnswer
{
    public static String question;
    public static String getAnswer()
    {
        String answer = calculate(this.question);
        return answer;
    }
}
```

```
function myFunction() {

    // Here goes the hooking code

}
Java.perform(myFunction);
```



# Frida Scripts - Analyzing static variables



```
package space.hitchhacker;
public class TheAnswer
{
    public static String question;
    public static String getAnswer()
    {
        String answer = calculate(this.question);
        return answer;
    }
}
```

```
Java.perform(() => {
    // Here goes the hooking code
});
```

# Frida Scripts - Analyzing static variables



```
package space.hitchhacker;
public class TheAnswer
{
    public static String question;
    public static String getAnswer()
    {
        String answer = calculate(this.question);
        return answer;
    }
}
```

*Fully qualified name of the class to be analyzed*

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    // ...TODO...
}
Java.perform(myFunction);
```

# Frida Scripts - Analyzing static variables



```
package space.hitchhacker;
public class TheAnswer
{
    public static String question;
    public static String getAnswer()
    {
        String answer = calculate(this.question);
        return
    }
}
```

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    console.log("Question: " + answerClass.question.value)
}
Java.perform(myFunction);
```

*We're logging the value of the "question" variable*

# Frida Scripts - Analyzing arguments



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

*We want to know which "question" is passed to the "getAnswer" method*

# Frida Scripts - Analyzing arguments



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

*Method name and arguments whose implementation we want to analyze*

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");
    answerClass.getAnswer.implementation = function(question) {
        console.log("Question: " + question);
    }
}
Java.perform(myFunction);
```

*We're logging the value of "question"*

# Frida Scripts - Analyzing arguments



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return
    }
}
```

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.implementation = function(question) {
        console.log("Question: " + question);
        this.getAnswer(question);
    }
}
Java.perform(myFunction);
```

*We have to call the original method  
so it is executed*

# Frida Scripts - Modifying arguments



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

*We want to get the answer for a certain question*

# Frida Scripts - Modifying arguments



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.implementation = function(question) {
        this.getAnswer("My custom question");
    }
}
Java.perform(myFunction);
```

*Calling the function with modified arguments*



# Frida Scripts - Analyzing return values



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

*We want to know which "answer" is returned by the method*

# Frida Scripts - Analyzing return values



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return
    }
}
```

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.implementation = function(question) {
        var answer = this.getAnswer(question);
        console.log("Answer: " + answer);
        return answer;
    }
}
Java.perform(myFunction);
```

*We're logging the return value of the function and then return it normally*

# Frida Scripts - Modifying return values



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

*We want to set a certain value to be returned as "answer"*

# Frida Scripts - Modifying return values



```
package space.hitchhacker;
public class TheAnswer
{
    public String question;
    public String getAnswer(String question)
    {
        String answer = calculate(question);
        return answer;
    }
}
```

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");
    answerClass.getAnswer.implementation = function(question) {
        return "42";
    }
}
Java.perform(myFunction);
```

*The answer is always 42*

# Frida Scripts - Method overload



```
package space.hitchhacker;
public class TheAnswer {

    public String getAnswer(String question) {
        String answer = calculate(question);
        return answer;
    }

    public String getAnswer(int numquestion) {
        String answer = calculate(numquestion);
        return answer;
    }

    public String getAnswer(String[] questionarray) {
        String answer = calculate(questionarray);
        return answer;
    }

}
```

*Methods have the same name but expect different arguments*

# Frida Scripts - Method overload



```
package space.hitchhacker;
public class TheAnswer {

    public String getAnswer(String question) {
        String answer = calculate(question);
        return answer;
    }

    public String getAnswer(int numquestion) {
        String answer = calculate(numquestion);
        return answer;
    }

    public String getAnswer(String[] questionarray) {
        String answer = calculate(questionarray);
        return answer;
    }
}
```

```
function myFunction() {
    var answerClass =
        Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.overload("java.lang.String
    ").implementation = function(question) {
        // ...
    }
}
Java.perform(myFunction);
```

# Frida Scripts - Method overload



```
package space.hitchhacker;
public class TheAnswer {

    public String getAnswer(String question) {
        String answer = calculate(question);
        return answer;
    }

    public String getAnswer(int numquestion) {
        String answer = calculate(numquestion);
        return answer;
    }

    public String getAnswer(String[] questionarray) {
        String answer = calculate(questionarray);
        return answer;
    }
}
```

```
function myFunction() {
    var answerClass =
        Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.overload("int").implementation = function(question) {
        // ...
    }
}
Java.perform(myFunction);
```

# Frida Scripts - Method overload



```
package space.hitchhacker;
public class TheAnswer {

    public String getAnswer(String question) {
        String answer = calculate(question);
        return answer;
    }

    public String getAnswer(int numquestion) {
        String answer = calculate(numquestion);
        return answer;
    }

    public String getAnswer(String[] questionarray) {
        String answer = calculate(questionarray);
        return answer;
    }
}
```

```
function myFunction() {
    var answerClass =
        Java.use("space.hitchhacker.TheAnswer");

    answerClass.getAnswer.overload(
        "[Ljava.lang.String").implementation =
        function(question) {
            // ...
        }
    Java.perform(myFunction);
}
```



# Frida Scripts - Method overload



## Java

int

byte

short

long

float

double

char

<Object>  
(z.B. String)

## Frida

int

byte

short

long

float

double

char

<package>.<Object>  
(z.B. java.lang.String)

## Java

int[]

byte[]

short[]

long[]

float[]

double[]

char[]

<Object>[]  
(z.B. String[])

## Frida

[I

[B

[S

[J

[F

[D

[C

[L<package>.<Object>  
(z.B. [Ljava.lang.String)

# Frida Scripts - static inner classes



```
package space.hitchhacker.guide;
public class TheAnswer {

    static class TheStaticAnswer {

        public static String getAnswer() {
            return 'Static 42';
        }

    }

}
```

*We can access them via  
<OuterClass>\$<StaticClass> and then call  
methods directly*

```
function myFunction() {
    var staticAnswerClass =
        Java.use("space.hitchhacker.TheAnswer$TheStaticAnswer");

    console.log(staticAnswerClass.getAnswer())
}

Java.perform(myFunction);
```

# Frida Scripts - Non-static inner classes



```
package space.hitchhacker.guide;
public class TheAnswer {

    class TheExactAnswer {

        public String getAnswer() {
            return '42';
        }
    }
}
```

*We can access them via `<OuterClass>$<InnerClass>` and have to create a new instance with `$new` before calling a method*

```
function myFunction() {
    var answerClass = Java.use("space.hitchhacker.TheAnswer");

    var exactAnswerClass =
        Java.use("space.hitchhacker.TheAnswer$TheExactAnswer");

    var answerInstance = answerClass.$new();
    var exactAnswerInstance = exactAnswerClass.$new(answerInstance)

    console.log(exactAnswerInstance.getAnswer())
}
Java.perform(myFunction);
```

# Frida - jadx to the rescue



```
File View Navigation Tools Help
app-release.apk
Source code
  android.support.v4
  androidx
  com
  kotlin
  kotlin.coroutines
  net.sqlcipher
  org
  space.hitchhacker.guide
    comm
    data
    databinding
    ui
    App
    AppKt
    AppKt$database$2
    AppKt$encprefs$2
    AppKt$prefs$2
    AppKt$users$2
    BuildConfig
    LoginActivity
    MainActivity
    PinActivity
    R
    UrlActivity
Resources
  APK signature
  Summary

LoginActivity
51     } else if (i2 != 3) {
52     } else {
53         if (validatePin()) {
54             launchMain();
55             return;
56         }
57         clearPin();
58         Toast.makeText(this, (int) R.string.failed_login, 0).show();
59     }
60 }
61
62 private final void changeFocus(int i) {
63     this.pinInput.get(i).clearFocus();
64     this.pinInput.get(i + 1).requestFocus();
65 }
66
67
68 private final void clearPin() {
69     for (EditText editText : this.pinInput) {
70         editText.getText().clear();
71     }
72     this.pinInput.get(3).clearFocus();
73     this.pinInput.get(0).requestFocus();
74 }
75
76
77 private final boolean validatePin() {
78     StringBuilder sb = new StringBuilder();
79     for (EditText editText : this.pinInput) {
80         sb.append(editText.getText().toString());
81     }
82     String sb2 = sb.toString();
83     Intrinsic.checkNotNull(sb2);
84     if (AppKt.getEncPrefs().getEncSharedPrefsPin().contains(sb2)) {
85         return Intrinsic.areEqual(Integer.parseInt(this.pinInput.get(3).getText().toString()) - Integer.parseInt(this.pinInput.get(1).getText().toString()), Integer.parseInt(this.pinInput.get(2).getText().toString()));
86     }
87     return Integer.parseInt(this.pinInput.get(3).getText().toString()) - Integer.parseInt(this.pinInput.get(1).getText().toString()) == Integer.parseInt(this.pinInput.get(2).getText().toString());
88 }
89
90 private final void launchMain() {
91     startActivity(new Intent(this, MainActivity.class));
92 }
93
94 }
```

Issues: 23 warnings Code Smali

Context menu options:

- Undo
- Can't Redo
- Cut
- Copy
- Paste
- Delete
- Select All
- Folding
- Line Wrap
- Find Usage (x)
- Go to declaration (d)
- Comment (;)
- Search comments (Ctrl + ;)
- Rename (r)
- Copy as frida snippet (f)**

# Frida - jadx to the rescue



```
let LoginActivity = Java.use("space.hitchhacker.guide.LoginActivity");
LoginActivity.validatePin.implementation = function(){
  console.log('validatePin is called');
  let ret = this.validatePin();
  console.log('validatePin ret value is ' + ret);
  return ret;
};
```

*Generated JavaScript code  
Don't forget to wrap it in Java.perform()*



# Frida Scripts for iOS

*Get references to class  
and method*


```
var SC = ObjC.classes["SharedCredentialController"]
var pC = SC["- passwordCredential"];
Interceptor.attach(pC.implementation,
{
  onEnter: function(args)
  {
    console.log(args[0]);
  },
  onLeave: function(retval)
  {
    console.log(new ObjC.Object(retval).toString());
    retval.replace(1);
  }
});
```

*Use Interceptor.attach*

*Implement onEnter / onLeave  
functions (both are optional)*

# Beg, Borrow, Steal, Create



 <https://learnfrida.info/>



[github.com/mobilesecurity/RMS-  
Runtime-Mobile-Security](https://github.com/mobilesecurity/RMS-Runtime-Mobile-Security)



[github.com/MobSF/Mobile-  
Security-Framework-MobSF](https://github.com/MobSF/Mobile-Security-Framework-MobSF)



[github.com/sensepost/objection](https://github.com/sensepost/objection)



[github.com/ChiChou/grapefruit](https://github.com/ChiChou/grapefruit)



[github.com/  
FrenchYeti/dexcalibur](https://github.com/FrenchYeti/dexcalibur)



[codeshare.frida.re](https://codeshare.frida.re)



[github.com/nccgroup/house](https://github.com/nccgroup/house)



# The bridge is yours

Demo: Setting up Frida





# The bridge is yours

Bypass the login pin of the Hitchhacker app

Launch the Vogon Construction App

Disable the access restrictions

- ★ Obtain the SQLite password
- ★ Obtain the deactivation password
- ★ Obtain all users and their passwords

# Solution

## Hitchhacker: Pin bypass

```
Java.perform(() => {  
    var myClass = Java.use("space.hitchhacker.guide.LoginActivity");  
  
    myClass.validatePin.implementation = function() {  
        console.log("Returning true for pin");  
        return true;  
    }  
});
```

# Solution

## Vogon Construction: Disable access restriction

```
Java.perform(() => {  
    var myClass = Java.use("eu.nviso.vogonconstruction.MainActivity");  
  
    myClass.isVogonOperator.implementation = function() {  
        console.log("Returning true for vogon operator");  
        return true;  
    }  
});
```

# Solution

## Vogon Construction: Obtain SQLite password

```
Java.perform(() => {  
    let AnonymousClass1 = Java.use("eu.nviso.vogonconstruction.UserProvider$1");  
    AnonymousClass1.toString.implementation = function(){  
        console.log('toString is called');  
        let ret = this.toString();  
        console.log('toString ret value is ' + ret);  
        return ret;  
    };  
});
```

# Solution

## Vogon Construction: Obtain deactivation password

```
Java.perform(() => {  
    let ControlPanel = Java.use("eu.nviso.vogonconstruction.ControlPanel");  
    ControlPanel.generateNewPassword.implementation = function(){  
        console.log('generateNewPassword is called');  
        let ret = this.generateNewPassword();  
        console.log('generateNewPassword ret value is ' + ret);  
        return ret;  
    };  
});
```

# Solution

## Vogon Construction: Obtain all users and their passwords

```
Java.perform(() => {
    let UserProvider = Java.use("eu.nviso.vogonconstruction.UserProvider");
    UserProvider.query.implementation = function(uri, strArr, str, strArr2, str2){
        let ret = this.query(uri, strArr, str, strArr2, str2);
        while(ret.moveToNext()) {
            console.log("User: " + ret.getString(ret.getColumnIndexOrThrow("name")));
            console.log("Password: " + ret.getString(ret.getColumnIndexOrThrow("password")));
            console.log("---");
        }
        return ret;
    };
});
```

# Igniting the Infinite Improbability Drive Again

Hooking with Objection





# Objection for Android



*Calling default scripts for common scenarios*

```
# android sslpinning disable
```

```
(agent) Custom TrustManager ready, overriding SSLContext.init()
```

```
(agent) Found com.android.org.conscrypt.TrustManagerImpl, overriding  
TrustManagerImpl.verifyChain()
```

```
(agent) Found com.android.org.conscrypt.TrustManagerImpl, overriding  
TrustManagerImpl.checkTrustedRecursive()
```

```
(agent) Registering job 523285. Type: android-sslpinning-disable
```

```
# android root disable
```

```
(agent) Registering job 040751. Type: root-detection-disable
```

*May or may not work...*

# Objection for Android



```
# android hooking list activities
```

*List all activities*

```
com.google.android.gms.ads.AdActivity  
com.google.android.gms.common.api.GoogleApiActivity  
org.proxydroid.AppManager  
org.proxydroid.BypassListActivity  
org.proxydroid.FileChooser  
org.proxydroid.ProxyDroid
```

*Launch any activity (including non-exported ones)*

```
# android intent launch_activity org.proxydroid.BypassListActivity  
Starting activity org.proxydroid.BypassListActivity...  
(agent) Activity successfully asked to start.
```

```
# android hooking search classes proxy
```

*Search for interesting classes*

```
$Proxy0  
$Proxy2  
[Landroid.net.IpConfiguration$ProxySettings;  
[Landroid.net.ProxyInfo;  
...
```

# Objection for Android



## List all methods

```
# android hooking list class_methods org.proxydroid.ProxyDroid
private android.widget.LinearLayout org.proxydroid.ProxyDroid.getLayout(android.view.ViewParent)
private boolean org.proxydroid.ProxyDroid.isEmpty(java.lang.String,java.lang.String)
private boolean org.proxydroid.ProxyDroid.serviceStart()
...
```

## Monitor function usage

```
# android hooking watch class_method org.proxydroid.ProxyDroid.serviceStart
(agent) Attempting to watch class org.proxydroid.ProxyDroid and method serviceStart.
(agent) Hooking org.proxydroid.ProxyDroid.serviceStart()
(agent) Registering job 459083. Type: watch-method for: org.proxydroid.ProxyDroid.serviceStart
```

```
# android hooking set return_value org.proxydroid.ProxyDroid.serviceStart true
(agent) Attempting to modify return value for class org.proxydroid.ProxyDroid and method serviceStart.
(agent) Hooking org.proxydroid.ProxyDroid.serviceStart()
(agent) Registering job 785400. Type: set-return for: org.proxydroid.ProxyDroid.serviceStart
```

## Modify basic return values



# Objection for iOS



*Default scripts for common scenarios*

```
# com.apple.mobilecal on (iPhone: 15.7) [net] # ios sslpinning disable
(agent) Hooking common framework methods
(agent) Found NSURLSession based classes. Hooking known pinning methods.
(agent) Hooking lower level SSL methods
(agent) Hooking lower level TLS methods
(agent) Hooking BoringSSL methods
(agent) Registering job 440948. Type: ios-sslpinning-disable
# com.apple.mobilecal on (iPhone: 15.7) [net] # ios jailbreak disable
(agent) Registering job 492880. Type: ios-jailbreak-disable
```

*May or may not work...*

# Objection for iOS



*Dump the content of the keychain for the current app*

```
com.apple.mobilecal on (iPhone: 14.7.1) [usb] # ios keychain dump
Created Accessible ACL Type Account Service Data
-----
2022-04-29 08:37:09 +0000 AfterFirstUnlock None kSecClassKey
(Key data not displayed)
2022-04-29 08:36:06 +0000 AfterFirstUnlock None Password arthur@space
com.apple.gs.beta.auth.com.apple.account.AppleIDAuthentication.token
AA---SNIP---9E=
```

```
com.apple.mobilecal on (iPhone: 14.7.1) [usb] # ios keychain dump --json keychain.json
Note: You may be asked to authenticate using the devices passcode or TouchID
Dumping the iOS keychain...
Writing keychain as json to keychain.json...
Dumped keychain to: keychain.json
```

*Save the dump to a json file on the host*

# Objection for iOS



```
# ios hooking search classes Security
```

```
NSFileSecurity
```

```
__NSPlaceholderFileSecurity
```

```
__NSFileSecurity
```

```
SecuritydXPCCallback
```

```
SecuritydXPCCClient
```

```
UISUISecurityContext
```

```
AFSecurityConnection
```

```
...
```

*Search for interesting classes*

*List class methods*

```
# ios hooking list class_methods AFSecurityConnection
```

```
- _connection
```

```
- setInternalAuthSessionToken:completion:
```

```
- _processData:usingProcedure:completion:
```

```
- processDataMap:usingProcedure:completion:
```

```
- initWithInstanceContext:
```

```
...
```

# Objection for iOS



## *Monitor function usage*

```
# ios hooking watch method "-[AFSecurityConnection setInternalAuthSessionToken:completion:]"
(agent) Found selector at 0x18ea9b758 as -[AFSecurityConnection setInternalAuthSessionToken:completion:]
(agent) Registering job 328546. Type: watch-method for: -[AFSecurityConnection
setInternalAuthSessionToken:completion:]
com.apple.mobilecal on (iPhone: 15.7) [net] #
```

```
# ios hooking set return_value "-[AFSecurityConnection setInternalAuthSessionToken:completion:]" false
(agent) Found selector at 0x18ea9b758 as -[AFSecurityConnection setInternalAuthSessionToken:completion:]
(agent) Registering job 545668. Type: set-method-return for: -[AFSecurityConnection
setInternalAuthSessionToken:completion:]
com.apple.mobilecal on (iPhone: 15.7) [net] #
```

## *Modify basic return values*





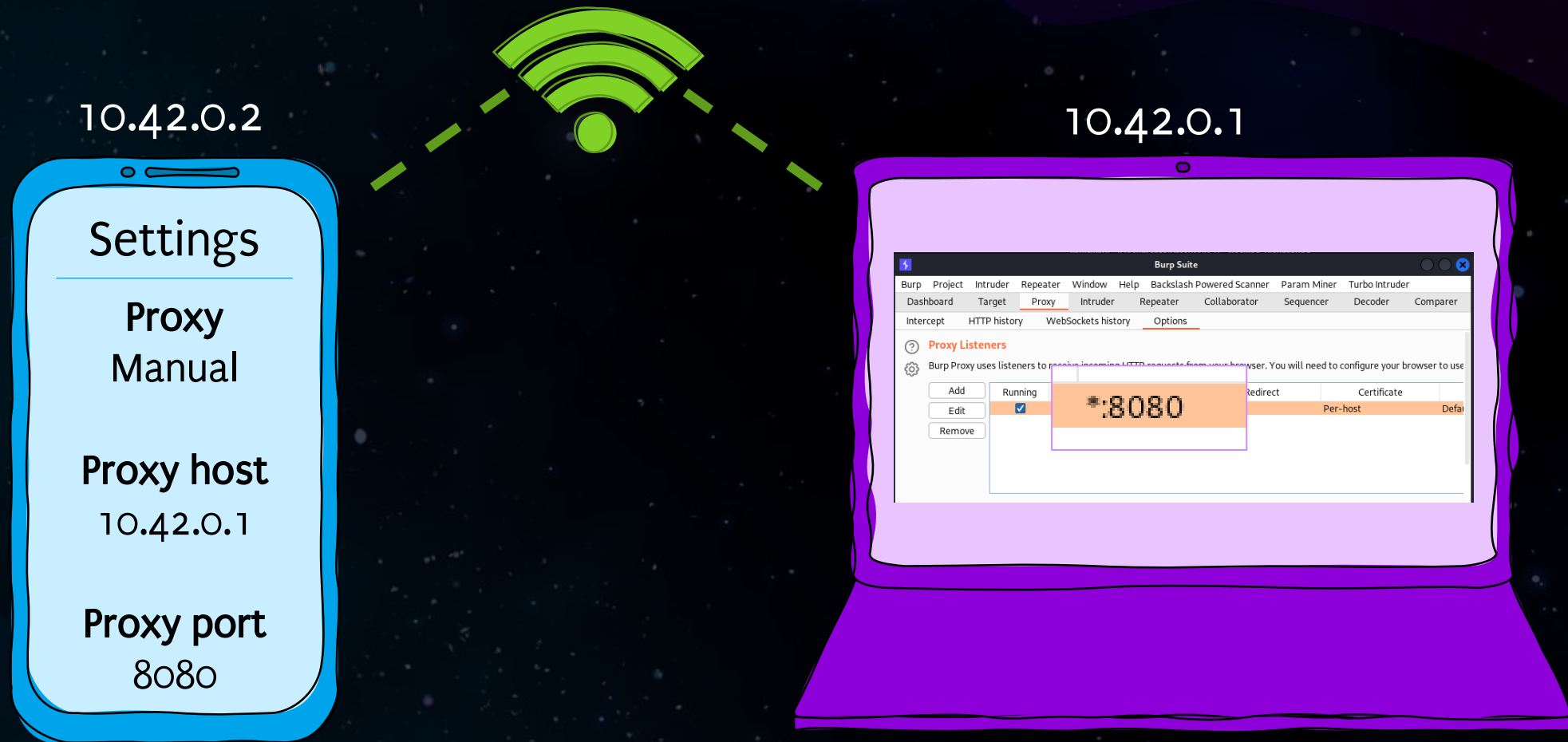
# The bridge is yours

1. Launch the Hitchhacker app with Objection
2. List all activities and launch and launch one of them
3. Identify the methods of the LoginActivity and modify the return value of the appropriate method to log in without knowing the pin

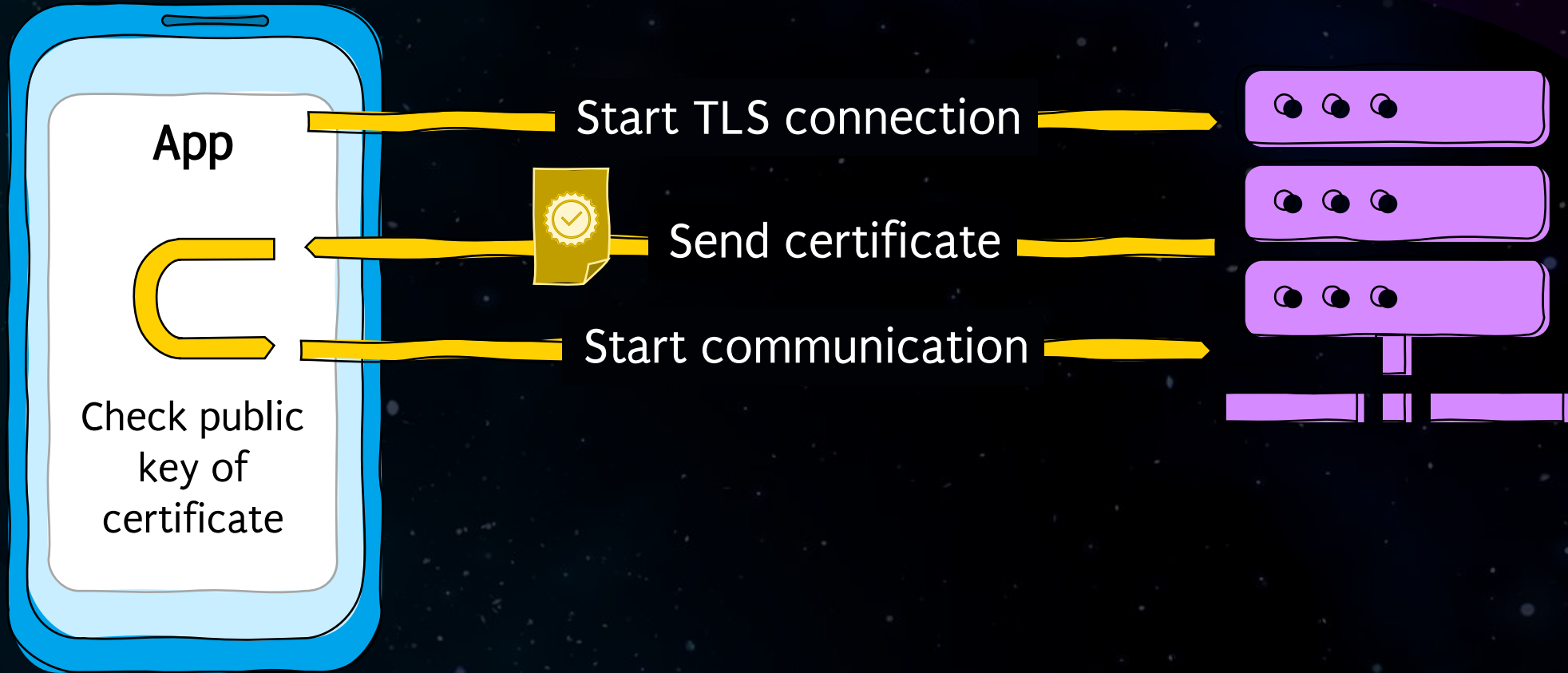
# Back to Earth

Adventures we didn't have time for

# Establishing a Man in the Middle



# Bypassing SSL pinning



# Frameworks



Xamarin



Unity



React Native



Ionic

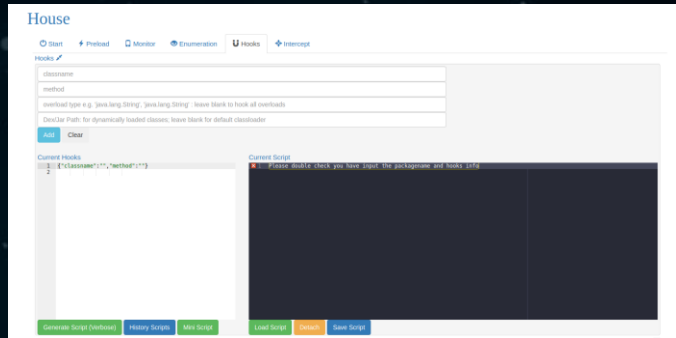
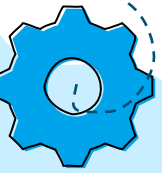


Flutter

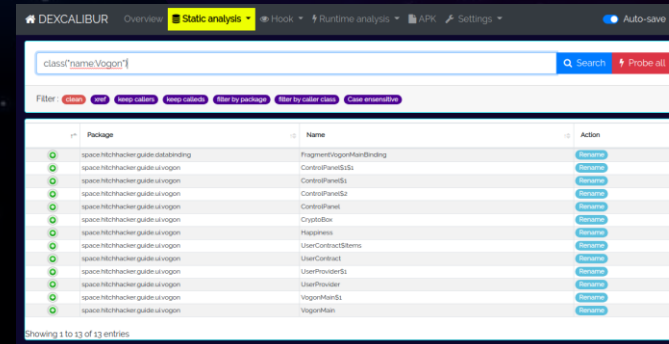


Apache Cordova

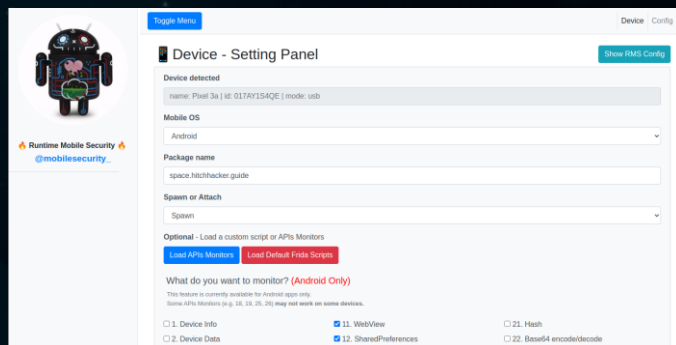
# Further tools



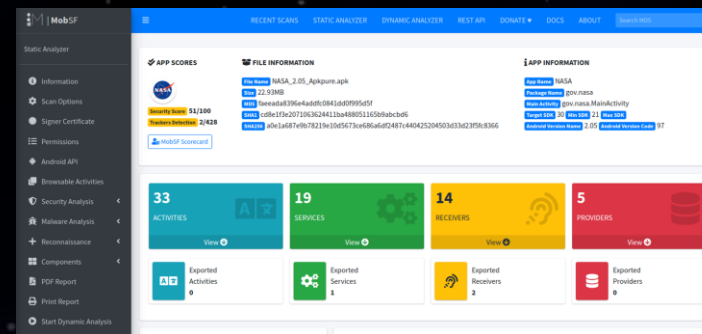
<https://github.com/nccgroup/house>



<https://github.com/FrenchYeti/dexcalibur>



<https://github.com/mobilesecurity/RMS-Runtime-Mobile-Security>



<https://github.com/MobSF/Mobile-Security-Framework-MobSF>

# MASVS

<https://mas.owasp.org/>

## MASVS

Mobile Application Security  
Verification Standard

## MASTG

Mobile Application Security  
Testing Guide

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
2.1	MSTG-STORAGE-1	System credential storage facilities need to be used to store sensitive data, such as PII, user credentials or cryptographic keys.	✓	✓	✓	Test Case	Test Case	Pass	✓
2.2	MSTG-STORAGE-2	No sensitive data should be stored outside of the app container or system credential storage facilities.	✓	✓	✓	Test Case	Test Case	Pass	✓
2.3	MSTG-STORAGE-3	No sensitive data is written to application logs.	✓	✓	✓	Test Case	Test Case	Fail	✗
2.4	MSTG-STORAGE-4	No sensitive data is shared with third parties unless it is a necessary part of the architecture.	✓	✓	✓	Test Case	Test Case	N/A	✗
2.5	MSTG-STORAGE-5	The keyboard cache is disabled on text inputs that process sensitive data.	✓	✓	✓	Test Case	Test Case	Pass	✓
2.6	MSTG-STORAGE-6	No sensitive data is exposed via IPC mechanisms.	✓	✓	✓	Test Case	Test Case	Fail	✗
2.7	MSTG-STORAGE-7	No sensitive data, such as passwords or pins, is exposed through the user interface.	✓	✓	✓	Test Case	Test Case	Fail	✗
2.8	MSTG-STORAGE-8	No sensitive data is included in backups generated by the mobile operating system.	✓	✓	✓	Test Case	Test Case	Fail	✗

Carlos Holguera  
Jeroen Willemsen



Sven Schleier  
Bernhard Mueller

Carlos Holguera  
Jeroen Willemsen



So Long,  
and Thanks for All  
the Fish