

iOS Pentesting Series Part 2- Into The Battlefield..



Kishor balan · Following

6 min read · Aug 14, 2023

Listen

Share

More

Hola Peeps,

Recently, much to my surprise, I've managed to wrap up the Part-2 section of my iOS Pentesting series, all thanks to my incredibly efficient and intelligent self. Please note, just a heads up, I've kept the theory parts brief; if you're craving more in-depth knowledge, feel free to consult dear ChatGPT. xD

Btw, If you haven't gone through the Part-1 yet, here it is

[Start your first iOS Application Pentest with me.. \(Part- 1\)](#) | by Kishor balan | Medium

IPA file Anatomy

Rename the IPA to ZIP and extract it. Let's analyze some of the important components.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.2604]
(c) Microsoft Corporation. All rights reserved.

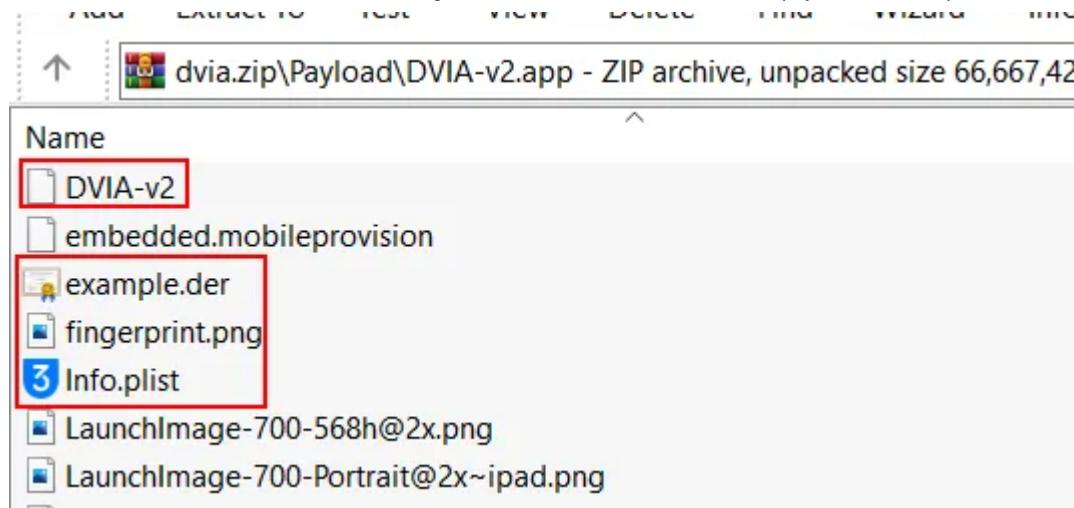
D:\Dvia>move dvia.ipa dvia.zip
    1 file(s) moved.

D:\Dvia>
```

↑ dvia.zip\Payload\DVIA-v2.app - ZIP archive, unpacked size 66,667,429 bytes

Name	Size
..	
_CodeSignature	106,060
AntiAntiHookingDebugging.storyboardc	14,252
AppIcons	435,129
ApplicationPatching.storyboardc	14,272
AttackingThirdPartyLibraries.storyboardc	12,069
Base.Iproj	23,721
BinaryProtection.storyboardc	16,058
BrokenCryptography.storyboardc	21,828
ClientSideInjection.storyboardc	13,505
Donate.storyboardc	26,361
DVIA_v2.momd	1,360
DVIASplash	1,385,240
ExcessivePermissions.storyboardc	27,484
Frameworks	54,634,122
Home.storyboardc	21,174
InsecureDataStorage.storyboardc	117,720
JailbreakDetection.storyboardc	7,634
LearniOSApplicationSecurity.storyboardc	12,100
META-INF	0
Model.momd	1,656

Frameworks: This is one of the important directory that we can take a look. This folder may contains extrernal frameworks that are implemented within the application (Eg: Frameworks for Jailbreak detection, SSL pinning ,etc)



DVIA-2 — The application's executable file

example.der — CA Certificate that may use for the SSL Pinning process or any other purposes.

Info.plist -The Info. plist file contains critical information about the configuration of an iOS mobile app

Test Cases:

1: Check for App Transport Security misconfiguration.

i) Read the Info.plist file using 3uTools or any other plist readers.

plist Editor **Info.plist**

Save Export Refresh

```

<key>DTXcodeBuild</key>
<string>9C40b</string>
<key>LSRequiresiPhoneOS</key>
<true/>
<key>MinimumOSVersion</key>
<string>10.0</string>
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
</dict>
<key>NSCameraUsageDescription</key>
<string>To demonstrate the misuse of Camera, please grant permission</string>

```

The boolean value is set TRUE for the “`NSAllowArbitraryLoads`” property, indicates that the App Transport Policy (ATS) is globally disabled. That means the app allows insecure HTTP communication.

Note: If Exception domains are defined, in that case, The insecure HTTP loads will applicable only for that domains.

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSEExceptionDomains</key>
    <dict>
        <key>example.com</key>
        <dict>
            <!-- Allow insecure HTTP connections -->
            <key>NSAllowsArbitraryLoads</key>
            <true/>
        </dict>
    </dict>
</dict>
```

In this example, the app is allowed to communicate with “example.com” using insecure HTTP connections, despite the general ATS requirements.

2: Check for hardcoded secrets/URLs from the IPA file using Grep or ack commands.

- i) Change the file extension from .ipa to .zip
- ii) Extract the zip
- iii) Get inside the app package

Find the hardcoded secrets.

```
(kali㉿kali) - [~/Payload]
└─$ ack "Secret"
└── PayloadFlow.app/Info.plist
69:      <key>APISecretKey</key>

(kali㉿kali) - [~/Payload]
└─$
```

Obatin the secret from that particular file:

```
</dict>
<key>CFBundlePackageType</key>
<key>APISecretKey</key>
<string>TTYHSJJJKI44NNABALSKSKSHTRRR</string>
<string>APPL</string>
<key>UISupportedDevices</key>
```

[Open in app ↗](#)



Unified must be installed on the iOS device for bypassing the signature checks.).

		All (9)	Updates (0)
<input type="checkbox"/> Info	<input type="checkbox"/> Import & Install ipa	<input type="checkbox"/> Backup	<input type="checkbox"/> Uninstall
<input type="checkbox"/> Apps (9)	<input type="checkbox"/> App Name	<input type="checkbox"/> Type	
<input type="checkbox"/> Photos	<input type="checkbox"/>	Other	
<input type="checkbox"/> Music	<input type="checkbox"/>	Other	
<input type="checkbox"/> Ringtones	<input type="checkbox"/>	Buy	
<input type="checkbox"/> Videos	<input type="checkbox"/>		

Note: If the testing IOS application is shared through Testflight, then we can directly install from it.

Insecure Storage

Checking the sensitive data in local storage

1: SharedPreference file

i) Connect to the target application using Objection and obtain its working directory using the below command.

ii) Navigate to the application's Library folder

```
[tab] for command suggestions
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # env
Name Path
-----
BundlePath /private/var/containers/Bundle/Application/73335472-A448-4367-B0E5-1607F438F81D/DamnVulnerableIOSApp.app
CachesDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches
DocumentDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Documents
LibraryDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # cd /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches
```

iii) Get into the SharedPreference folder

```
/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ls
NSFileType Perms NSfileProtection Read Write Owner Group Size Creation Name
-----
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-11 07:08:47 +0000 Caches
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-13 12:42:57 +0000 Saved Application State
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-13 12:42:57 +0000 HTTPStorages
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-13 12:44:51 +0000 SplashBoard
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-13 12:42:57 +0000 Private_Documents
Directory 493 n/a True True mobile (501) mobile (501) 96.0 B 2023-08-11 07:08:47 +0000 Preferences

Readable: True Writable: True
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # cd Preferences
/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Preferences
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ls
NSFileType Perms NSfileProtection Read Write Owner Group Size Creation Name
-----
Regular 384 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 405.0 B 2023-08-13 13:55:02 +0000 com.highaltitudehacks.dvia.plist
```

iv) Read the Preference file using below command.

```
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # cd Preferences
/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Preferences
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ios plist cat com.highaltitudehacks.dvia.plist
{
    WebDatabaseDirectory = "/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches";
    WebKitLocalStorageDatabasePathPreferenceKey = "/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches";
    WebKitOfflineWebApplicationCacheEnabled = 1;
    WebKitShrinksStandaloneImagesToFit = 1;
}
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # exit
```

(Sorry, No sensitive data is found in my current target application)

2: Local DB files

i) Connect the target application with objection

ii) Find the application path in which the cache files gets stored sing the below command and navigate to that path

```
[tab] for command suggestions
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # env
Name Path
BundlePath /private/var/containers/Bundle/Application/73335472-A448-4367-B0E5-1607F438F81D/DamnVulnerableIOSApp.app
CachesDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches
DocumentDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Documents
LibraryDirectory /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # cd /var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches
/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # cd com.highaltitudehacks.dvia
/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches/com.highaltitudehacks.dvia
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ls
NSFileType Perms NSFileProtection Read Write Owner Group Size Creation Name
----- -----
Directory 493 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 128.0 B 2023-08-13 13:54:59 +0000 com.apple.metal
Directory 493 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 64.0 B 2023-08-13 13:54:59 +0000 com.apple.metalfe
Regular 420 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 48.0 KiB 2023-08-13 12:42:57 +0000 Cache.db
Regular 420 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 0.0 B 2023-08-13 12:42:57 +0000 Cache.db-wal
Regular 420 CompleteUntilFirstUserAuthentication True True mobile (501) mobile (501) 32.0 KiB 2023-08-13 12:42:57 +0000 Cache.db-shm
```

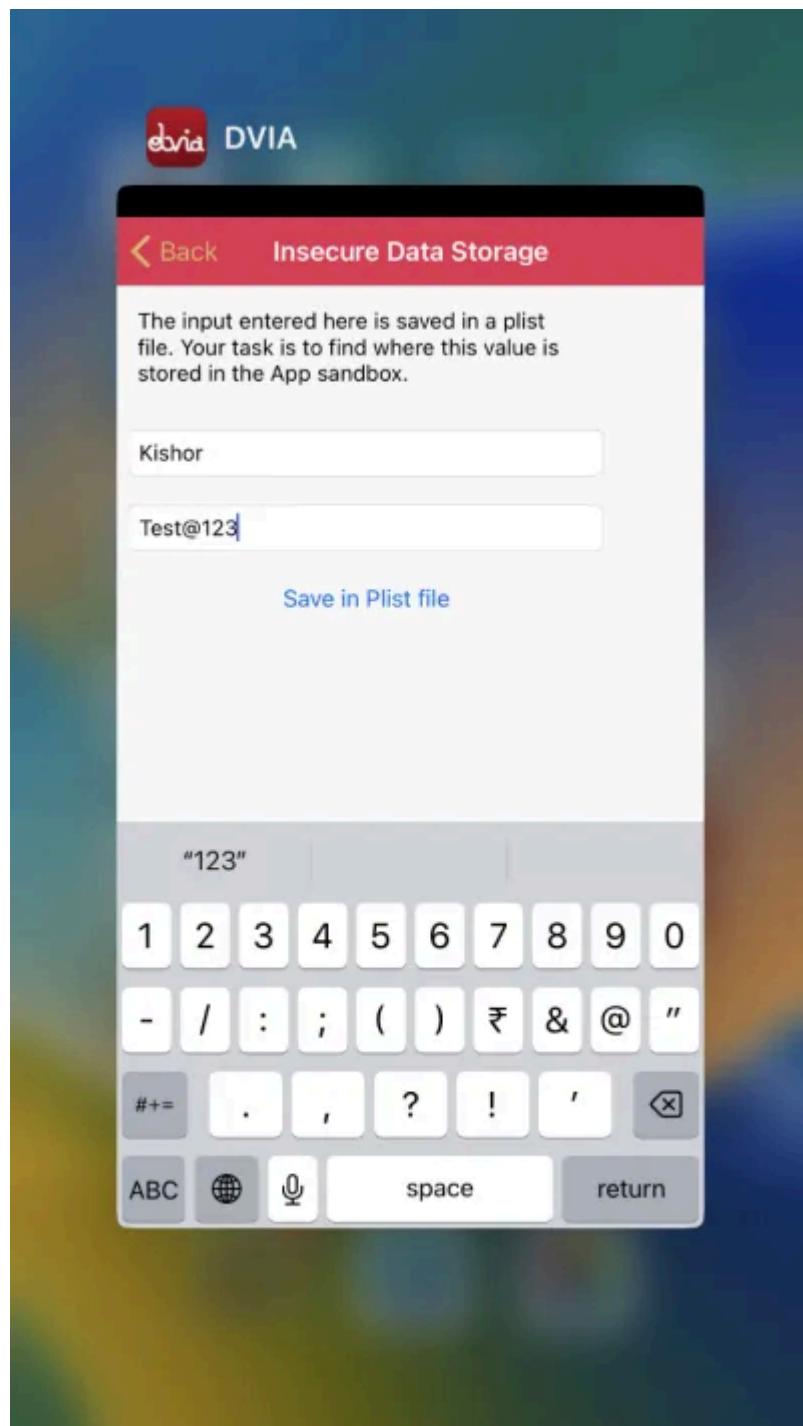
iii) Copy the “Cache.db” to your local machine with the scp command

```
C:\Users\kishor>scp root@192.168.0.101:/var/mobile/Containers/Data/Application/8722EAC6-06EC-4665-A854-24DBCB5D1FD6/Library/Caches/com.highaltitudehacks.dvia/Cache.db .
Password for root@iPhone8-VM-fnz:
Cache.db

C:\Users\kishor>
```

iv) Explore the db file using any DB browsers and look for the sensitive data

3: Background screencaching: (Lol i guess i dont wanna spit out a deep explanation for this, Its just the thing when the app being sent to the background state. If the app screen is visible, then it is vulnerable)



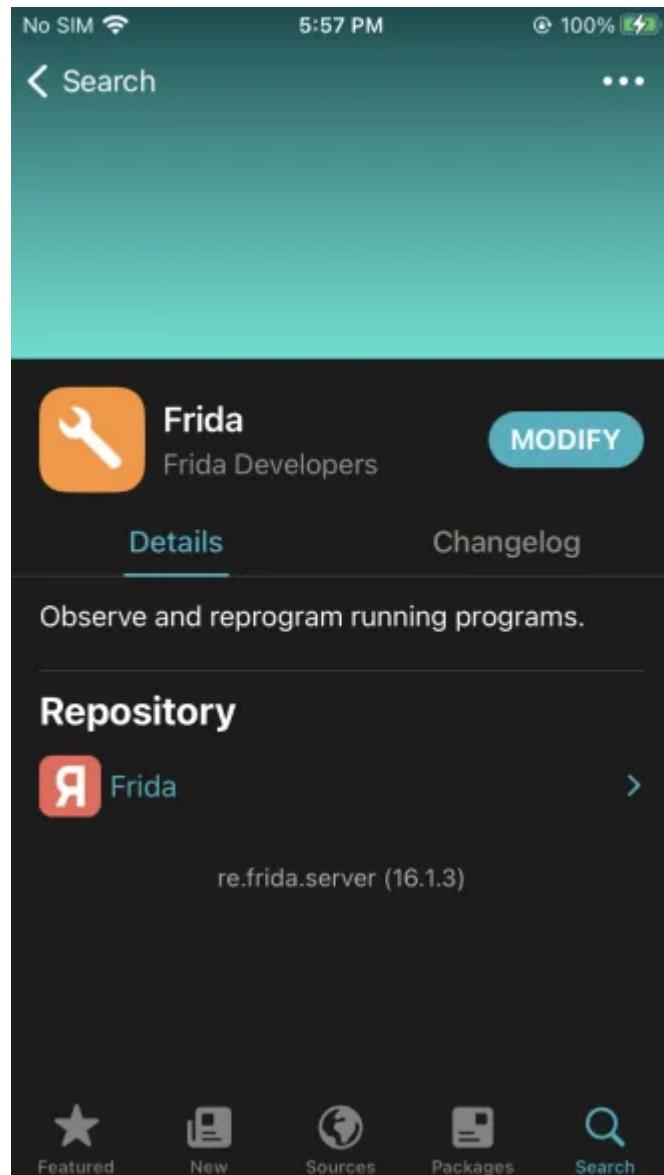
4: Checking sensitive data in Heap memory:

sensitive data such as login credentials / OTP/ security PIN codes/ Session Tokens / PII data

Tool: Fridummp

[GitHub – Nightbringer21/fridump: A universal memory dumper using Frida](#)

- i) Make sure the frida tweak is installed and running in your Cydia/Sileo



ii) Execute the fridump command as follows:

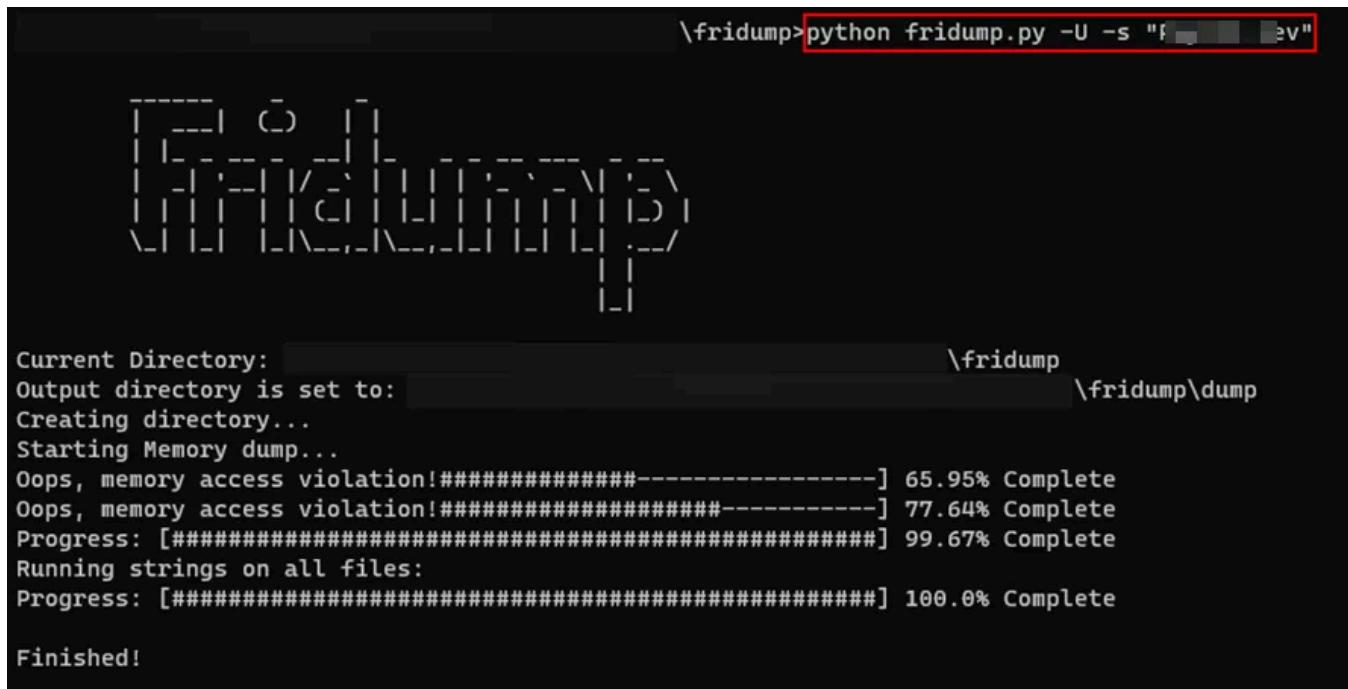
```
python3 fridump.py -U -s "App name"
```

```
\fridump>python fridump.py -U -s "f...ev"
[REDACTED]
[REDACTED]
[REDACTED]
[REDACTED]
[REDACTED]

Current Directory: \fridump
Output directory is set to: \fridump\dump
Creating directory...
Starting Memory dump...
Oops, memory access violation!#####-----] 65.95% Complete
Oops, memory access violation!#####-----] 77.64% Complete
Progress: [#####-----] 99.67% Complete
Running strings on all files:
Progress: [#####-----] 100.0% Complete

Finished!
```

iii) By executing the previous step, the command will create a folder in the current directory called dump and a file named ‘strings.txt’, which contains all the memory strings. Open the ‘strings.txt’ file and it is observed that user-sensitive data are not cleared or replaced from memory.



```
strings.txt
File Edit View
Referer https://paymer 555
_charset_=UTF-8&sttoken=60-
d8b3fd[REDACTED]32f474c1cd83b7ae31b03fc4113&stnonce=
60-3a6ac6d0a7887183de8084ef1d4ebe6a&pan=[REDACTED]&expirymonth=06&expiryyear=
2029&securitycode=[REDACTED]&process=&fraudcontroltransactionid=Wei[REDACTED]y84Gv%
3BAmNgFI2WhscAuzYLCQtV4g%3D%3D%3B3ZcCfCKAgXMWXxNtKWsrgIxS%
2B0s7o5ju6ThKyUn6jtyAUz8y9j311tbUEzT2B8gVmgtVcb9PdvgQ4wWtzViZC5tSN98ND0sbfffEINMXhHn
ukM8H7qsui0Y%2BAKwxeESB%2BoQrd2K0zwf9Vqllfuvcn29JSMlkGbF0CB8AvYsc%2F19rc7evmd%
2Byd59H1nvk%2BuRKqwQOME4p9RSb%2B2R%
2B97EABFJfUeaHDCwMbZIVdGiefZKkN6GXQcYCKoJkqMb6MxAUrUeaZoDtJZm9H3PMs%2FFLPjW%
2FPB11t1p06VAH06fw4BDKQBK9NqsuloUPAH%
```

5: Sensitive Data in the Keychain:

Keychain services provide APIs (Application Programming Interfaces) that allow developers to securely store, retrieve, and manage sensitive data in a way that's isolated from the rest of the app's data. And some applications may store this data in plaintext format.

i) After Login process, Attach the target application with the objection and execute the following command:

```
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # e
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ios keychain dump
Note: You may be asked to authenticate using the devices passcode or touchID
Save the output by adding `--json keychain.json` to this command
Dumping the iOS keychain...
Created Accessible ACL Type Account Service
-----
2023-08-13 13:32:56 +0000 WhenUnlocked None Password keychainValue com.highaltitudehacks.dvia
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] #
```

					Data
					Passw0rd@1234

6) Insecure logging of Sensitive data: There are some dev dudes who logs everything including user data during the app development process and some of them would forget to remove the log functions during the app releasing process. So in that cases, We gotta grab those snacks ;)

- i) Identify the sensitive input fields in the target application.
- ii) Submit the data via the input fields
- iii) Collect the App logs and obtain the data (You can use “Real Time logger” utility provided by 3uTools

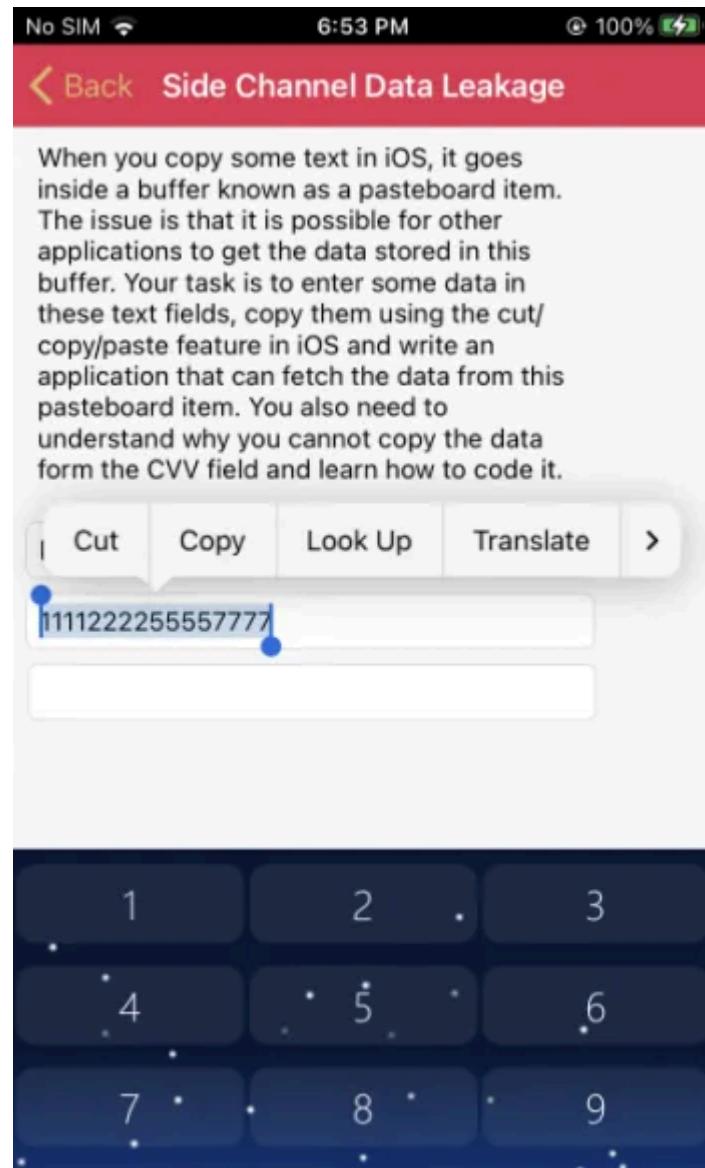
The screenshot shows the 3uTools Realtime Log interface. At the top left, there's a search bar labeled "Search results". Below it is a button labeled "Realtime Log" with the sub-instruction "Real-time display of log info...". A red arrow points to the "Export" button in the toolbar below the search bar. The main window displays a list of log entries. The first few entries are as follows:

```
(0/300) Pwr/0 Range:0 Critical:0 PriorityConfiguration:0 CombinedType:0 {com.apple.sharengd-Central} (470*100*300) AP,0 AD:0(30/300) AS:0 RAS:0 DMN:1 FG:0 ADVBF:0(0ms/0ms) Range:0 Critical:0 pwrAsrt:0 CT:0 AgentType:0 | Jun 22 10:32:49 iPhone-7--AK bluetooth[1464] <Notice>: Returning scan parameters: Main:30.00ms/300.00ms LP:0.00ms/300.00ms(unsupported) SC:0.00ms/0.00ms/non-concurrent(unsupported) Dm:(m:30/300) Cond:0:1:0 Sc:1:0 (passive) Dupfilter:Enabled minScanLevel:4 HD:Yes MP:No Jun 22 10:32:49 iPhone-7--AK bluetooth[1464] <Notice>: needToRestart=0 Jun 22 10:32:49 iPhone-7--AK kernel(AppleBCMWLanCore)[0] <Notice>: 7 b 20MHz n CCA 17% duration: 237 congest: 41 ts: 226374 Jun 22 10:32:49 iPhone-7--AK locationd[1487] <Notice>: MotionCoprocessor,startTime,709102969.896073,motionType,1,youthType,0,youthTypeReason,0 Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:TX(14:A7:2B:41:4A:3C) AC<SU MS NB NRS NA CM EX TF FFP MRET FLE> BE<0 0 0 0 0 0 0 0 0 0> (5001ms) Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:TX(14:A7:2B:41:4A:3C) AC<SU MS NB NRS NA CM EX TF FFP MRET FLE> BK<1 0 0 0 0 0 0 0 0 0> (5001ms) Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:TX(14:A7:2B:41:4A:3C) AC<SU MS NB NRS NA CM EX TF FFP MRET FLE> VI<0 0 0 0 0 0 0 0 0 0> (5001ms) Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:TX(14:A7:2B:41:4A:3C) AC<SU MS NB NRS NA CM EX TF FFP MRET FLE> VO<0 0 0 0 0 0 0 0 0 0> (5001ms) Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:L3 Control VO TX(14:A7:2B:41:4A:3C) Success=0 NoACK=0 Expired=0 OtherErr=0 Jun 22 10:32:51 iPhone-7--AK kernel([080211Family][0]) <Notice>: LQM-WiFi:TX(FF:FF:FF:FF:FF:FF) AC<SU MS NB NRS NA CM EX TF FFP MRET FLE> BE<0 0 0 0 0 0 0 0 0 0> (5002ms) Jun 22 10:32:51 iPhone-7--AK wifid(WiFiPolicy)[1451] <Notice>: _WiFiDeviceProcessLqmTxStatsEvent: LQM-TX: Success: 2(100.0%) Failure:0(0.0%) Retries:0(0.0%) NoACK:0(0.0%) Drops:0(0.0%) NoBuff:0(0.0%) NoRes:0(0.0%) ChipErr:0(0.0%) Expired:0(0.0%) ForcedExpiry:0(0.0%) Free:0(0.0%)
```

Side Channel Leakage:

1: Clipboard buffer cache

- i) Find the sensitive input fields in the target application
 - ii) Try to copy the sensitive data



iii) Now attach the target application with Objection and execute the command “ios pasteboard monitor”

```
C:\Users\kishor>objection -g DVIA explore
Using USB device `Apple iPhone`
Agent injected and responds ok!

[object]注入(v1.11.0)

Runtime Mobile Exploration
by: @leonjza from @sensepost

[tab] for command suggestions
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ios pasteboard monitor
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # (agent) [pasteboard-monitor] Data: 1111222255557777
```

We can see the objection could retrieve the copied card data from clipboard cache.

And that's a wrap for today, folks! But hey, don't fret, there's still a bunch of exciting stuff in our toolkit waiting to be unleashed — you know, the whole shebang like setting up proxies, bypassing JailBreak detection and SSL pinning, Bypassing biometrics and more. Buckle up, because in my next installment, I'll dive into all of these juicy topics.

Oh, the exhaustion is real! I'm starting to think the next part might just have to come with its very own YouTube video... I mean, who needs all that typing, am I right?

Bella Ciao..

IOS

Pentesting

Cybersecurity

Mobile Security

Hacking



Following



Written by Kishor balan

418 Followers

Security Analyst || eWPTXv2 || ejPT

More from Kishor balan

```

on="1.0" encoding="utf-8"?>
<security-config>
<config>
  main includeSubdomains="true">example.com</domain>
  <pin expiration="2018-01-01">
    <pin digest="SHA-256">7HIpactkIAq2Y49orF00QKurWxmmSFZhBCoQYcRhJ3
    <!-- backup pin -->
    <pin digest="SHA-256">fwza0LRMXouZHRC8Ei+4PyuldPDcf3UKg0/04cDM1c
  </pin>
</config>
<security-config>

```

 Kishor balan

It's all about Bypassing Android SSL Pinning and Intercepting Proxy Unaware applications.

Hola H3ckers,

6 min read · Nov 27, 2022

 243  2



...

```

[ios] 100 command suggestions
com.highaltitudehacks.dvia on (iPhone: 16.4.1) [usb] # ios hooking generate simple JailbreakDetectionVC
var target = ObjC.classes.JailbreakDetectionVC;

Interceptor.attach(target['- isJailbroken'].implementation, {
  onEnter: function (args) {
    console.log('Entering - isJailbroken!');
  },
  onLeave: function (retval) {
    console.log('Leaving - isJailbroken!');
  },
});

Interceptor.attach(target['- readArticleTapped:'].implementation, {
  onEnter: function (args) {
    console.log('Entering - readArticleTapped:');
  },
  onLeave: function (retval) {
    console.log('Leaving - readArticleTapped:');
  },
});

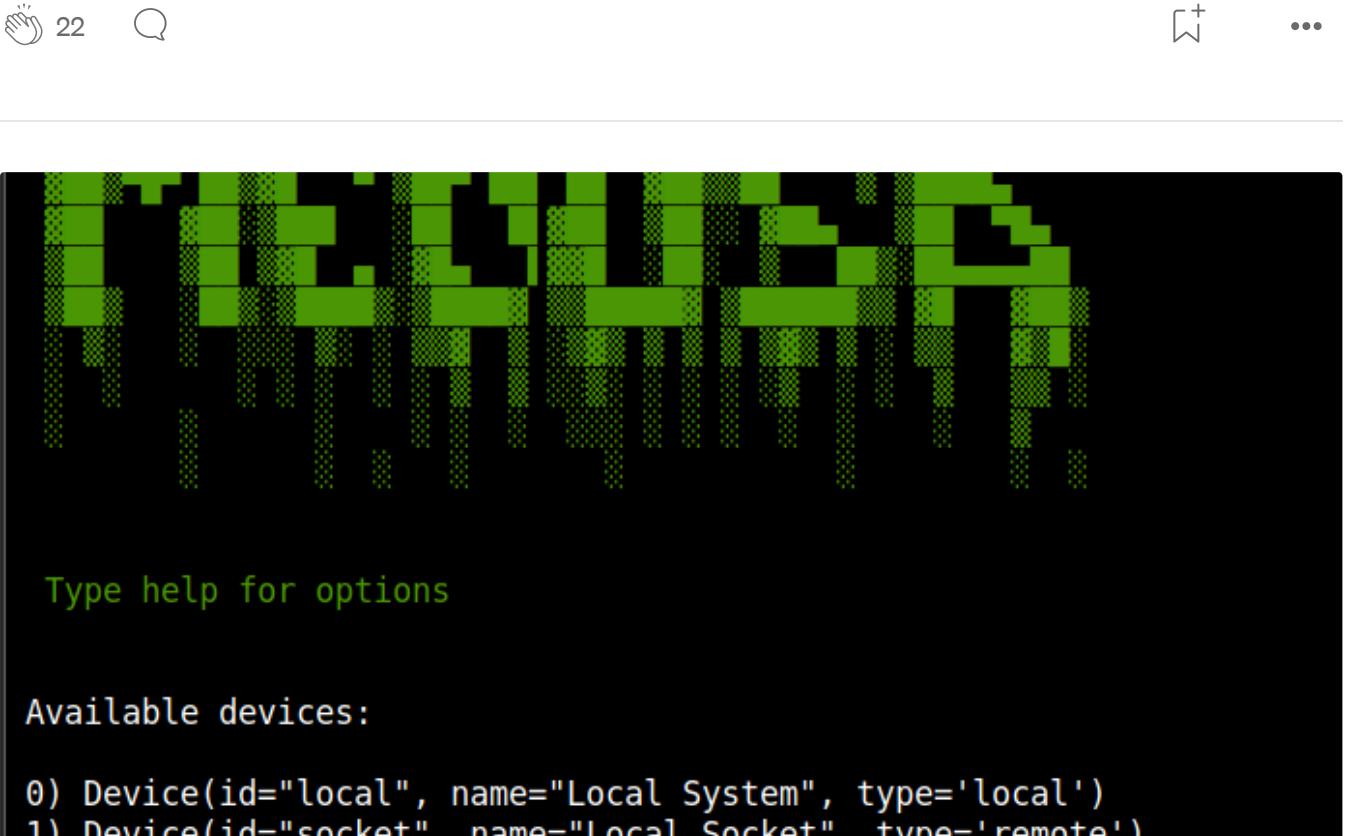
```

 Kishor balan

iOS Pentesting Series Part 3- The Ceasefire

Hola mates,

7 min read · Aug 19, 2023

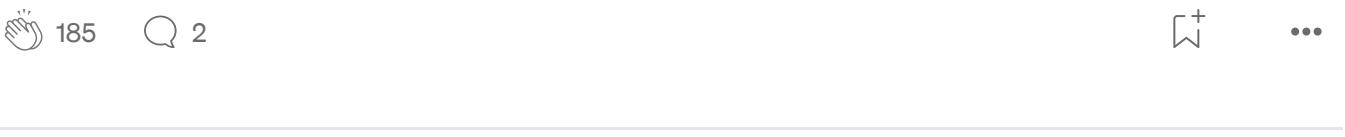


 Kishor balan

My fav 7 methods for Bypassing Android Root detection

Hola H3ck3rs.

6 min read · Oct 16, 2022



iPhone 7 32GB Black
PC Charging(2W) 4%

iOS Version	13.3.1 (17D50)
Jailbroken	Yes Install AFC2
Activated	Yes
Product Type	iPhone9,1 (A1779)
Sales Model	MNCE2 J/A
IMEI	[REDACTED]
Serial No.	G7X
ECID	0[REDACTED]026
Verify UDID	Yes
UDID	[REDACTED]B11B
View Verification Report	
View iDevice Details	

Hard Disk Capacity 11.65 GB / 29.79 GB

System Apps Photos Media UDisk Used Free

Reboot Turn Off Refresh

Kishor balan

Start your first iOS Application Pentest with me.. (Part- 1)

Hola Heckers,

6 min read · Jan 14, 2023

240 3



...

See all from Kishor balan

Recommended from Medium



 DianaOpanga

A beginners guide to using Frida to bypass root detection.

This is a simple use case to bypass root detection using Frida. For this example we have created a sample APK that has implemented root...

3 min read · Nov 28, 2023



...



 Chinmay Talad

Pass eJPT with Tryhackme Challenge Room

Below is the list of free Tryhackme rooms which will help you to pass the exam, The lists contains both walkthroughs and CTF challenges, I...

2 min read · Nov 22, 2023



...

Lists



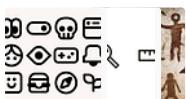
Tech & Tools

16 stories · 212 saves



Apple's Vision Pro

7 stories · 65 saves



Icon Design

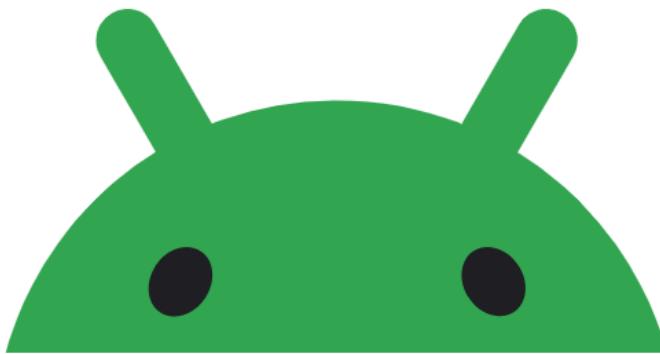
36 stories · 284 saves



Productivity

240 stories · 407 saves

Android



Sandeep Vishwakarma in InfoSec Write-ups

A step-by-step Android penetration testing guide for beginners

Greetings fellow hackers, my name is Sandy, Security Analyst and Bug bounty hunter.

18 min read · Nov 3, 2023

 678 4

...

 Pawan Jaiswal

Android Penetration Testing with Frida: A Comprehensive Guide with Examples

As mobile devices become increasingly integral to our daily lives, securing Android applications against potential vulnerabilities is of...

4 min read · Jan 27, 2024

 19

...

 Prince Varghese

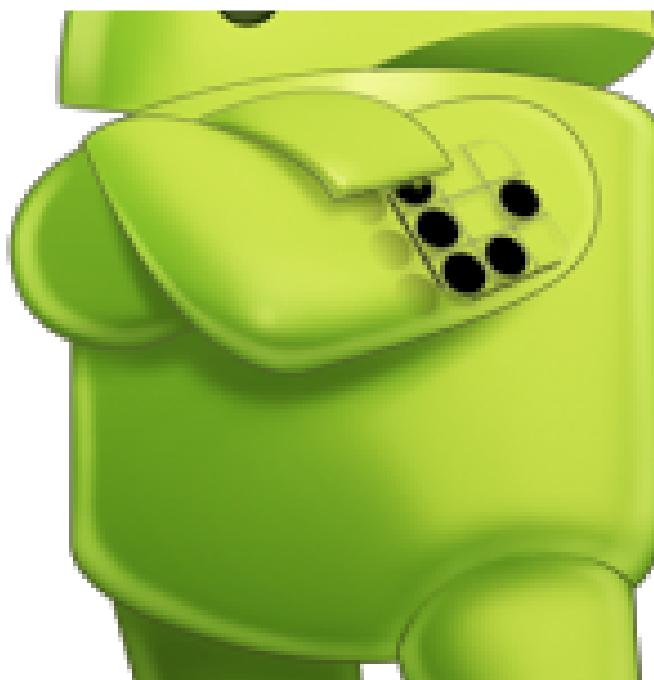
Task Hijacking in Android

Hello friends, Welcome back to my blog. Today we are going to explore Task Hijacking vulnerability in Android.

7 min read · Nov 25, 2023

 44

...

 Hacker's Dump

Diving Deep: A Comprehensive Guide to Android Penetration Testing—Part 1

Introduction: Navigating the Android Abyss

7 min read · Nov 16, 2023



...

See more recommendations