Android Uygulama Güvenliğine Saldırgan Yaklaşım

Offensive Approach to Mobile Application Security

Ahmet GÜREL
www.gurelahmet.com


MOBİL
PENTEST
EĞİTİMİ
Ahmet GÜREL
Security Consultant
Innovera

$whoami

- Blogger
- Canyoupwnme Ekip Üyesi
- Güvenlik Danışmanı / Sızma Testi Uzmanı

Eğitim Başlıkları

- Android Dünyasına Giriş
- Android Güvenlik Modeli
- Android Uygulamaları
- APK ve Paket İçeriği
- Android Tersine Mühendislik İşlemleri
- OWASP Mobil Top 10 Zafiyetleri
- Test Ortamının Kurulumu
- Android Uygulama Dosyaları
- Android Uygulama İzinleri
- SSL Pinning ve Bypass Yöntemleri
- Mobil Sızma Testi Araçları
- Mobil Sızma Testi Uygulamaları

Android Dünyasına Giriş

- Open Handset Alliance liderliğinde Google firması tarafından akıllı telefon ve tablet bilgisayarlar gibi mobil cihazlar için geliştirilmiş **Linux** tabanlı işletim sistemi.

- Android cihazlarda uygulamaların çalışabilmesi için .apk uzantılı dosyalar ile uygulamalar yüklenir ve cihazlara dağıtabilir.
-
- Günümüzde Native ve Hybrid uygulamalardan söz edilmekte.
- Native uygulama dediğimiz C++ veya Java dilini temel alan Android ile yazılan uygulamalar Native fakat HTML, CSS, JavaScript tabanlı kodu yazıp birçok platforma çıktı veren frameworkler ile geliştirildiğinde Hybrid uygulama olarak geçmektedir.

Neden Android ?

- Açık kaynak kodlu
- Linux tabanlı
- Kullanım yaygınlığı - telefonlar, tabletler, arabalar...
- Gelişmiş ve ücretsiz yazılım geliştirme ortamı sunması
- Açık uygulama marketi

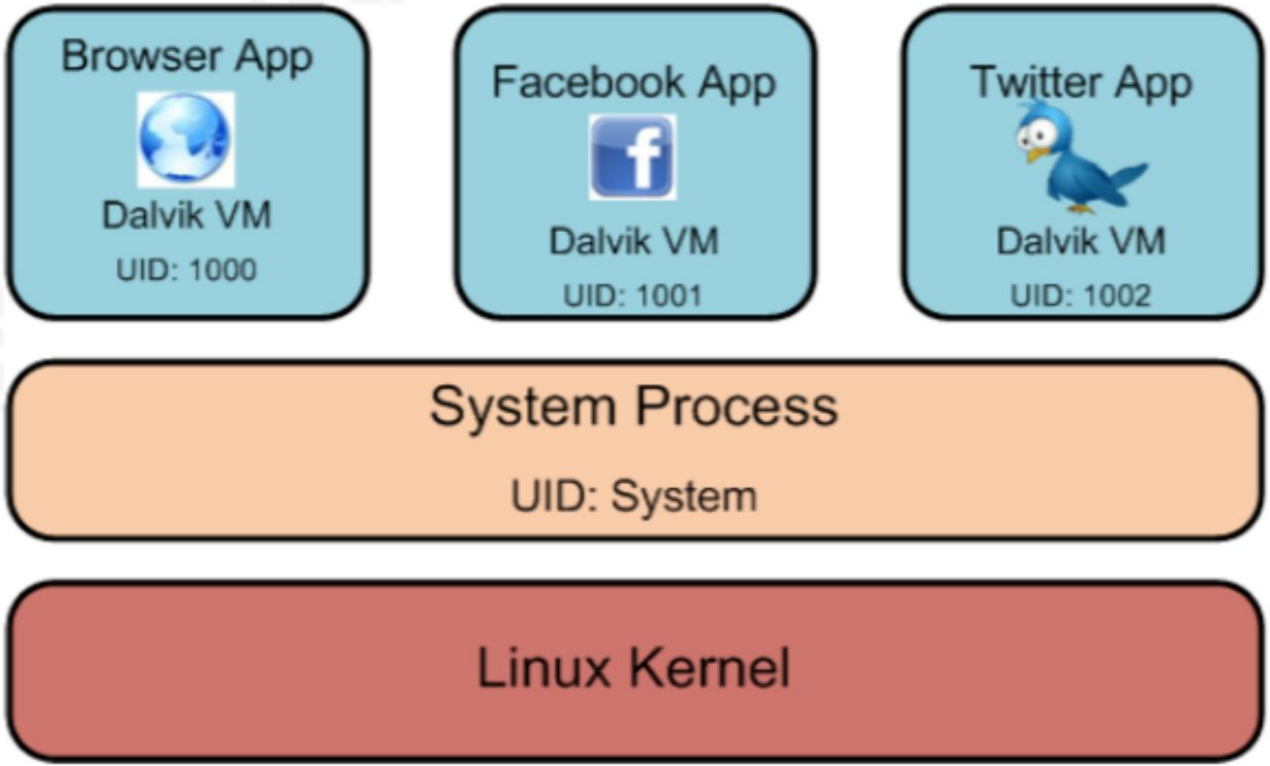
Android Kullanım Alanları

- Cep telefonları, tabletler, akıllı saatler vs….
- Arabalar, akıllı ev sistemleri
- Mobil bankacılık
- Internet of Things (IoT)

Android Güvenlik Modeli

- Linux güvenlik modeli baz alınmıştır (UID/GUID).
- Uygulama bazlı izinler kullanılmaktadır.
- Uygulama izinleri, **AndroidManifest.xml** dosyasında tanımlanmaktadır.
- Uygulama kurulumu için uygulamanın sertifika ile imzalanmış olması gerekmektedir.
- Her bir uygulama farklı bir DVM(Dalvik Virtual Machine) içerisinde çalışmaktadır.
- Sistem güvenliği açısından **kullanıcı** kilit rol oynamaktadır.
- Rootlanmamış bir cihaz için root erişimi mümkün değildir. “su” uygulaması sistemde bulunmaz.

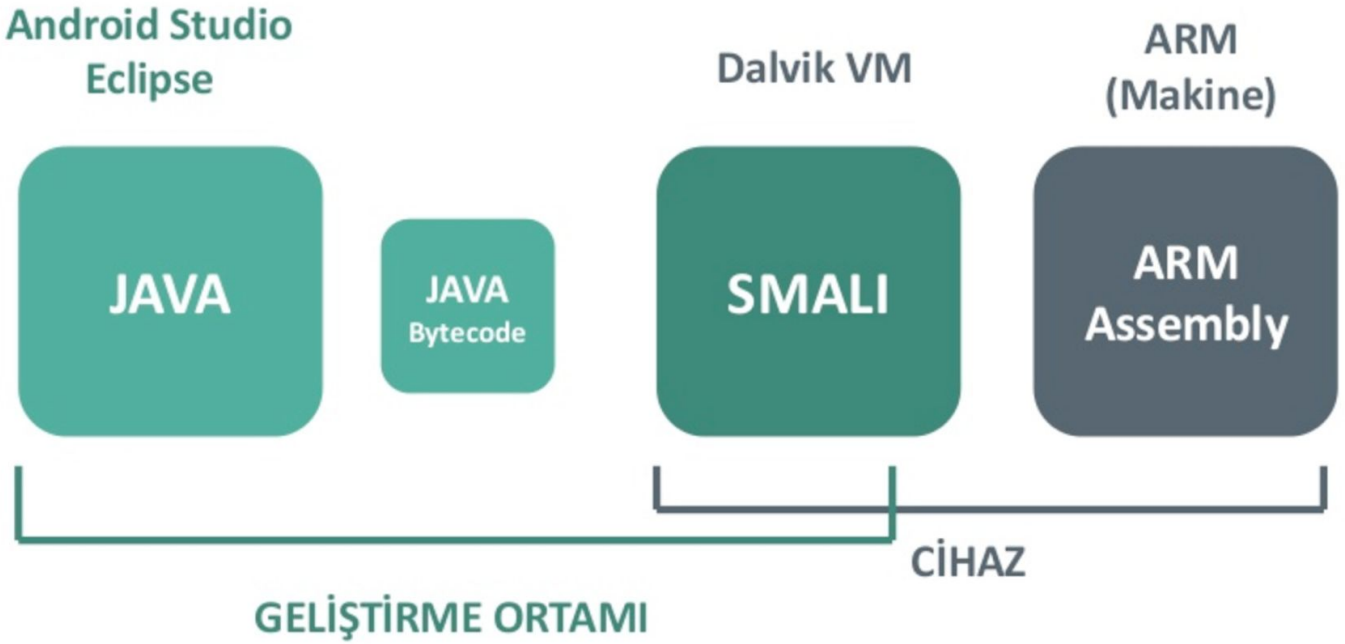
Android Güvenlik Modeli

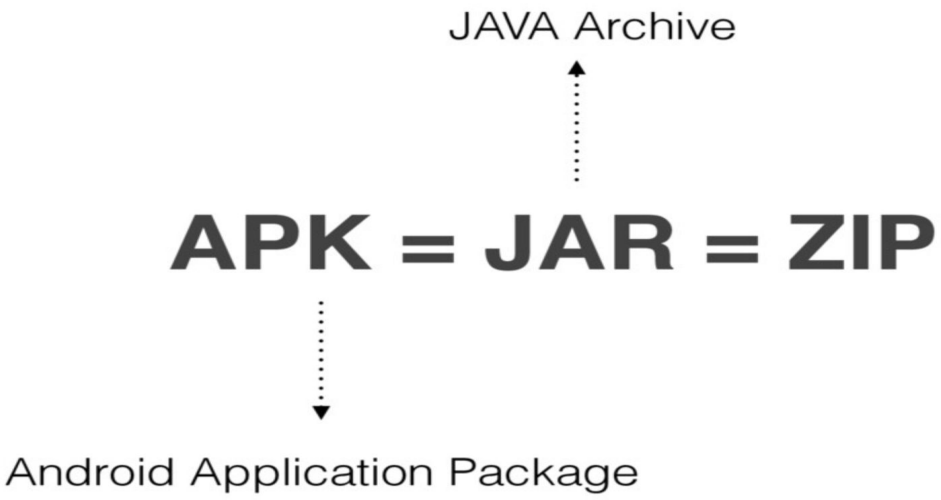


Android Uygulamaları

- Java + Android SDK ile geliştirilir.
- Android Dalvik VM ile çalıştırılır.
- JAVA **->** .class **->** .dex

Android Uygulamaları

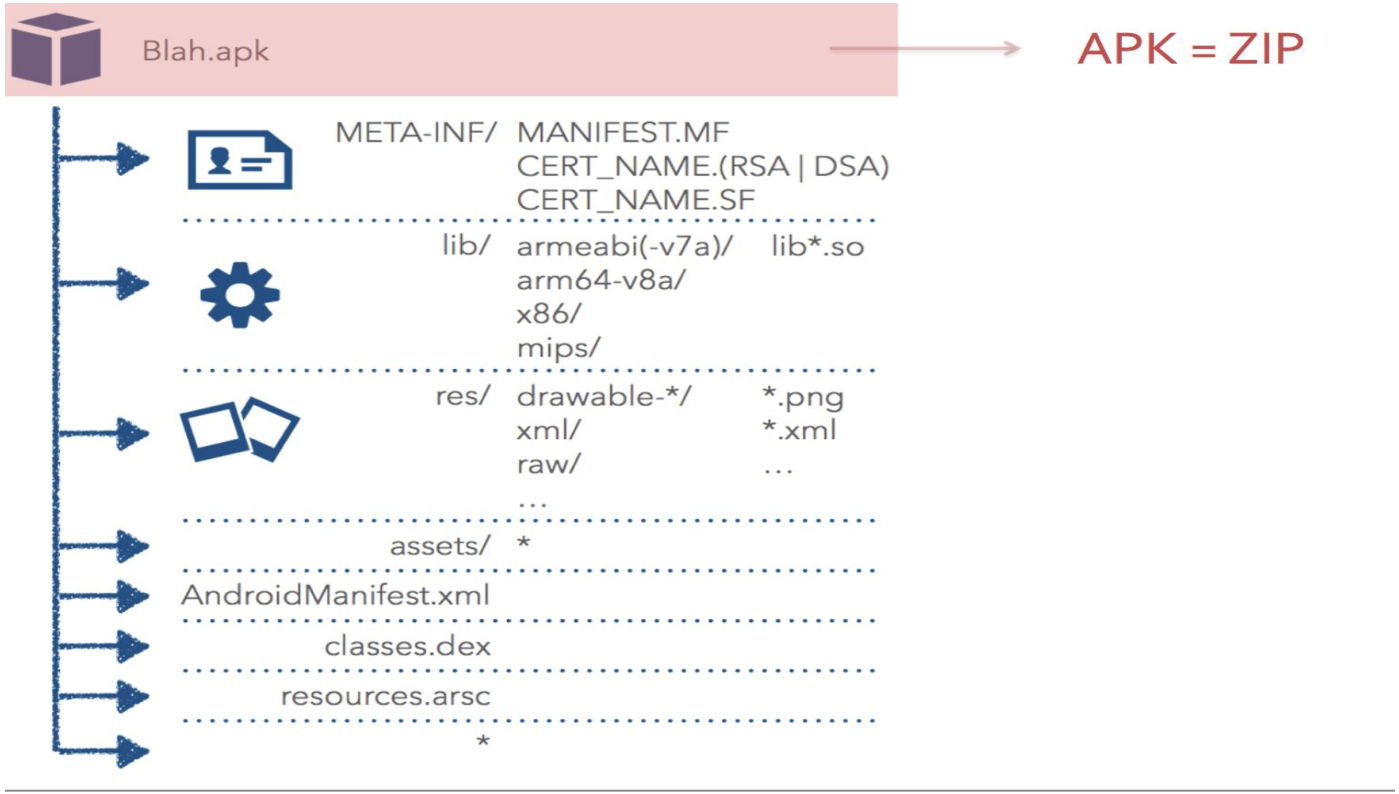



JAVA Archive
APK = JAR = ZIP
Android Application Package

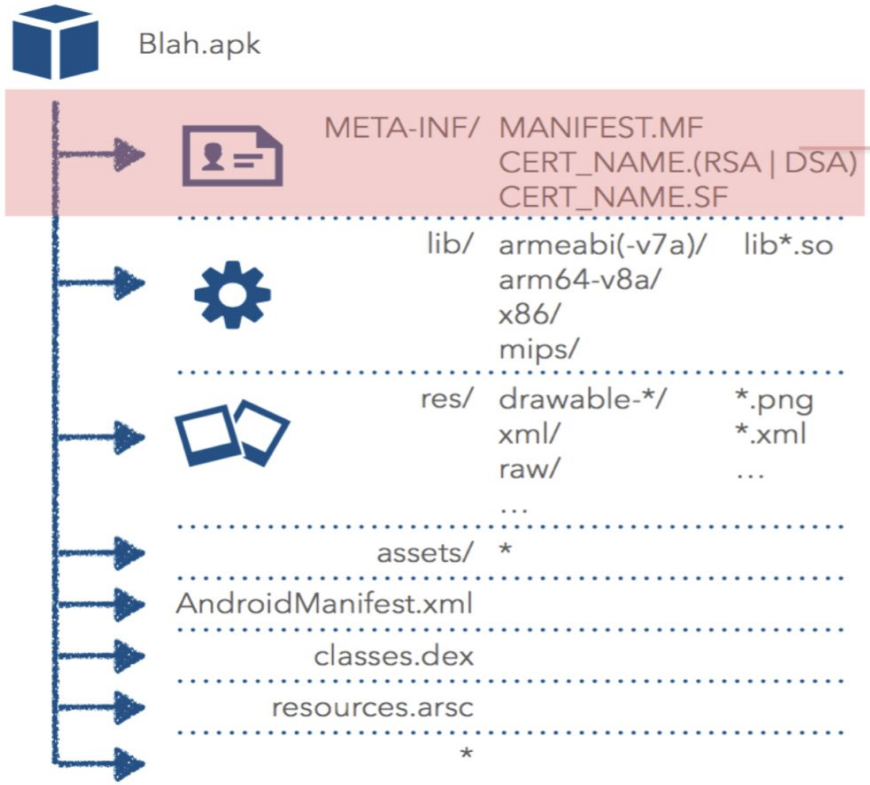
APK

- **Android Application Package File (APK)** dosyası zip dosya formatına sahip .apk uzantılı dosyalardır.
- APK dosyasının uzantısı .zip olarak değiştirildikten sonra WinZip, WinRAR gibi arşiv programları ile dosya içeriği görüntülenebilir.

Android Paket İçeriği

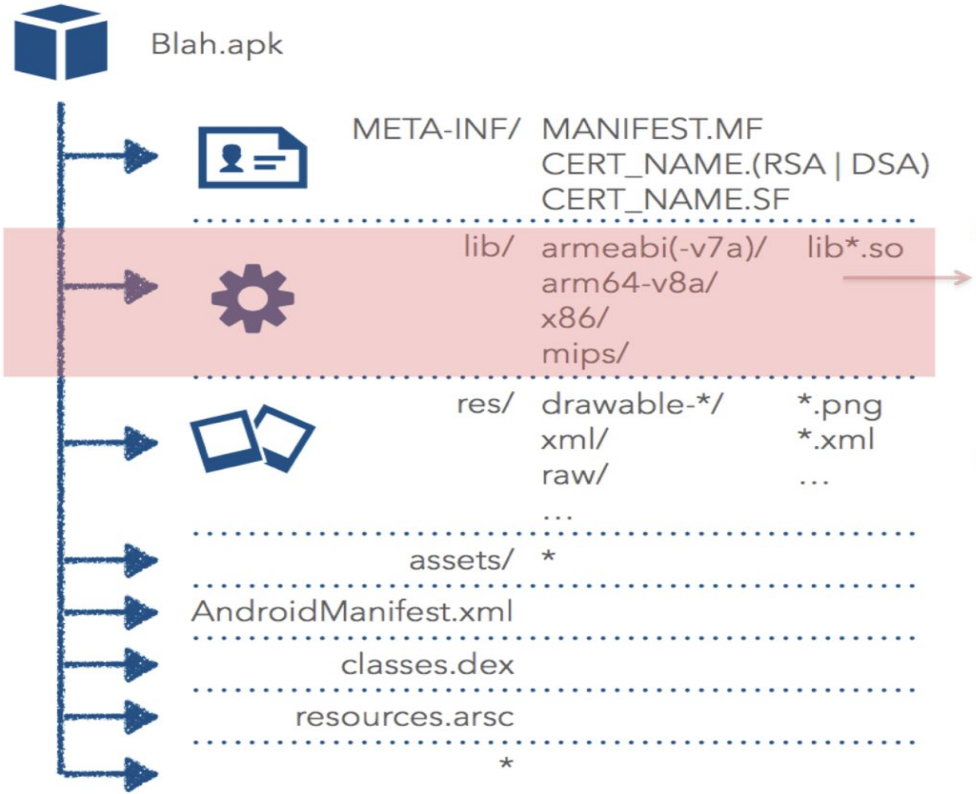


Android Paket İçeriği



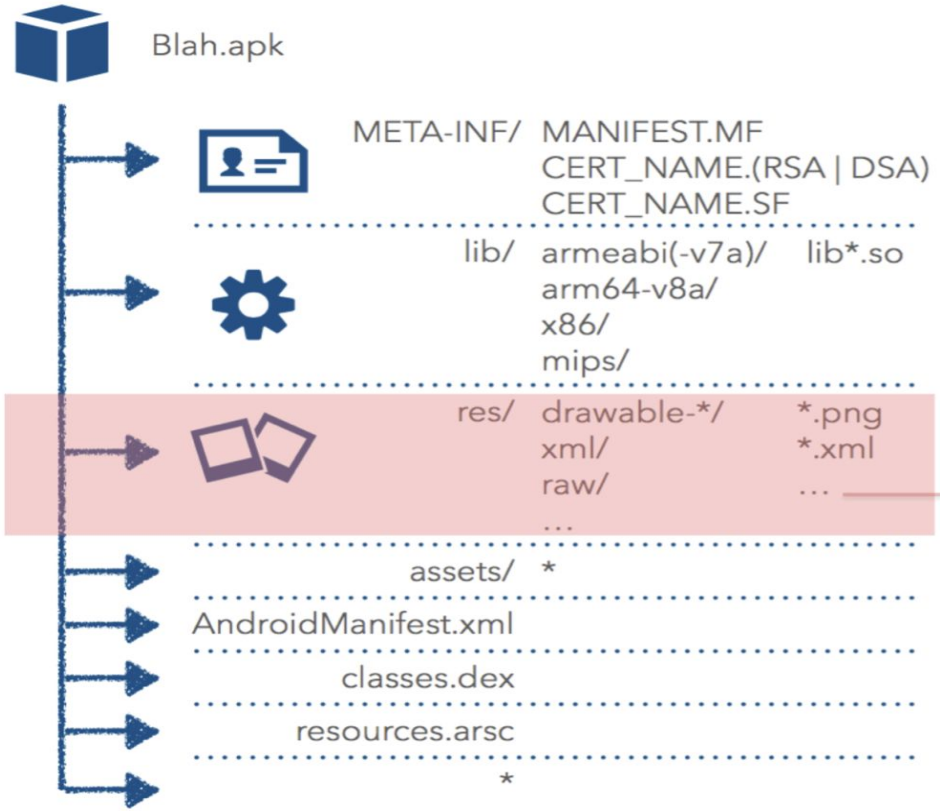
- Dosya imzaları (hash)
- Geliştirici Public Key'i

Android Paket İçeriği



- İşlemci mimarisine göre compile edilmiş native kütüphaneler (Native ELF dosyaları)
- JAR Dosyaları (kütüphaneler)

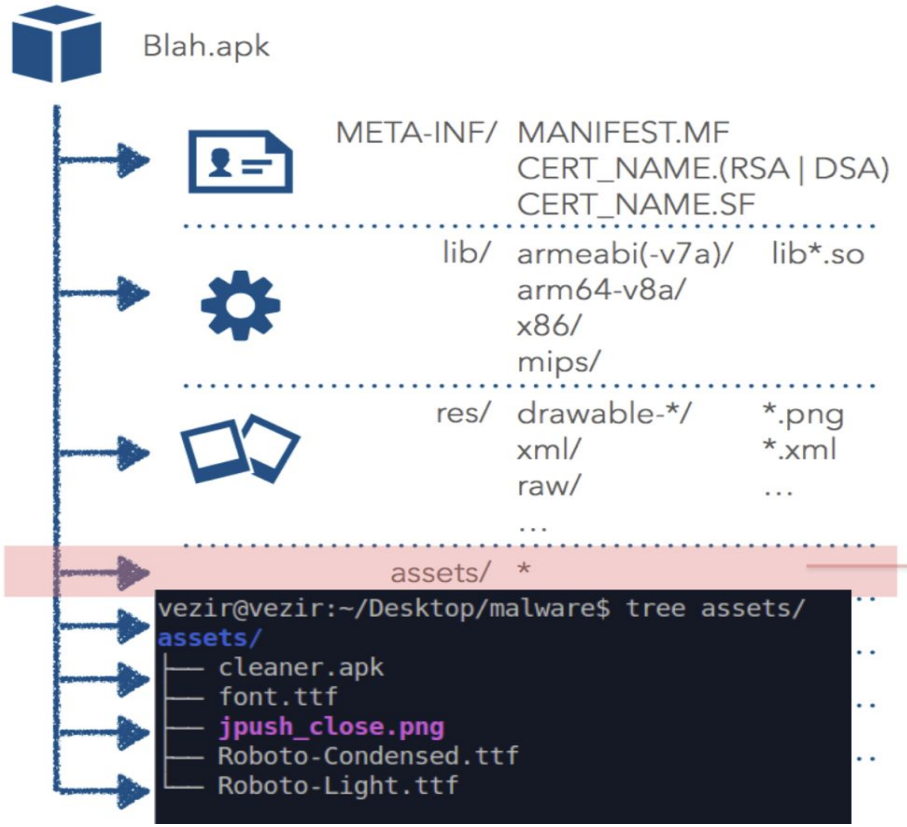
Android Paket İçeriği



- **anim:** Compile edilmiş animasyon dosyaları
- **drawable:** Resim dosyaları
- **layout:** UI/view tanımlamaları
- **values:** Diziler, renkler, style'lar, string'ler dimensions
- **xml:** Compile edilmiş XML dosyaları
- **raw:** Compile edilmemiş raw dosyalar

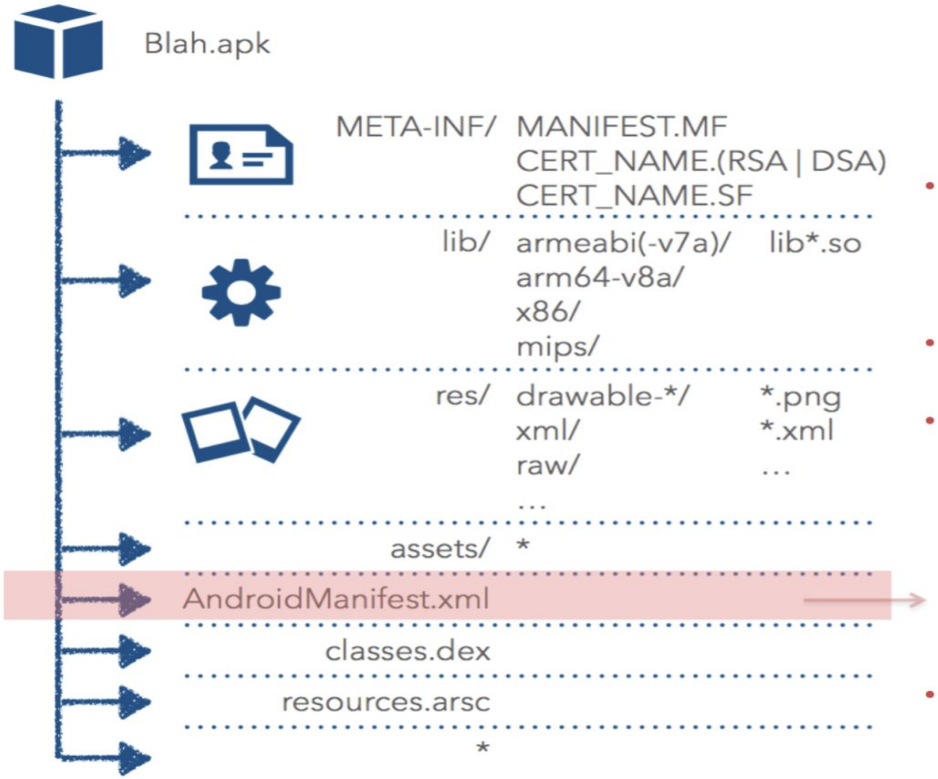
Compile işlemi AAPT (Android Asset Packaging Tool) tarafından yapılır

Android Paket İçeriği



- Çoğu zaman raw dosyalar bulunur.
- Resimler, fontlar, ses dosyaları
- Bazı malware'ler bu dizinde cihaza kurrmak üzere APK dosyaları saklarlar

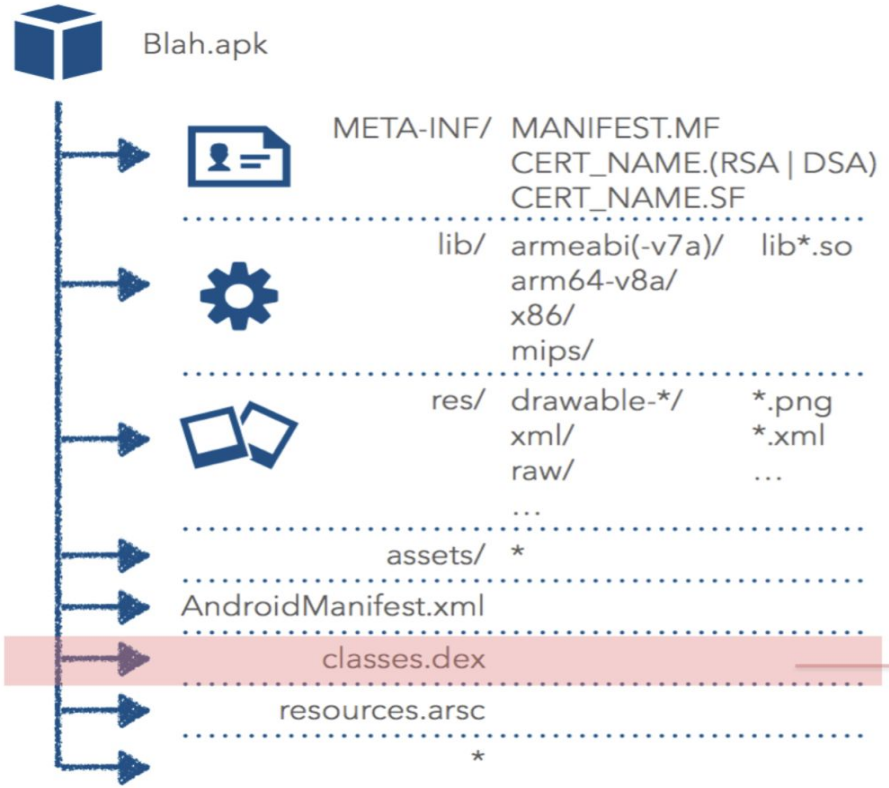
Android Paket İçeriği



- Uygulama meta-data'ları
  - Paket ismi
  - Versiyon bilgisi
  - …
- Uygulamanın talep ettiği izinler
- Uygulamada bulunan komponentler
  - Activity
  - Service
  - Bradcast Receiver
  - Content Provider

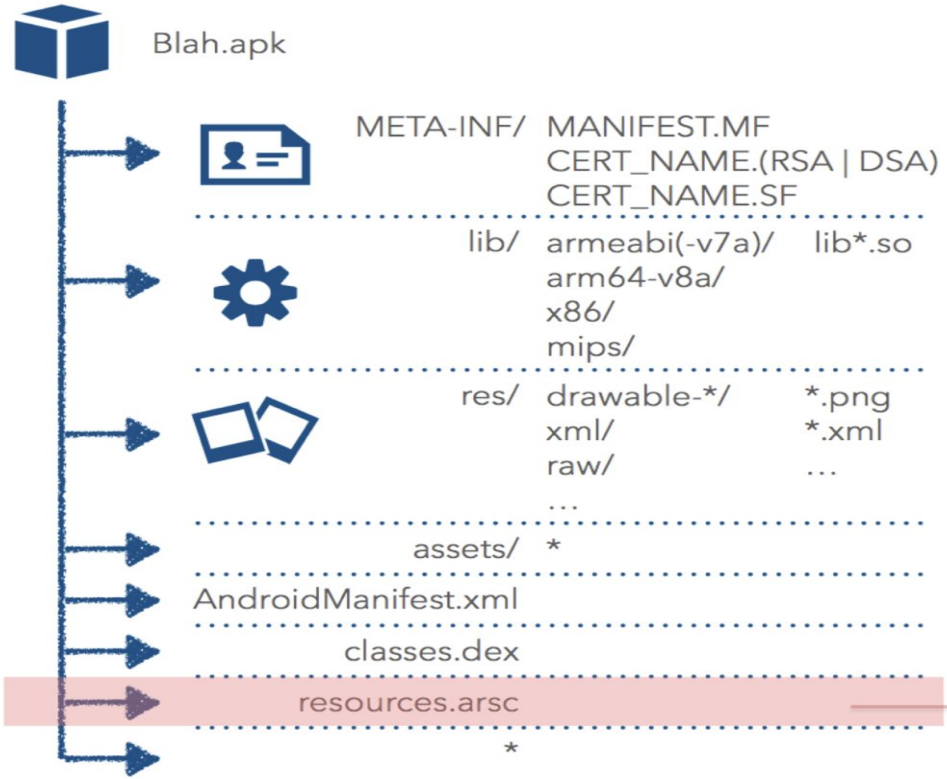
- Compile edilmiş olarak paket içerisinde yer alır.

Android Paket İçeriği



- DEX: Dalvik Executable
- Android'in EXE'si
- Dalvik VM üzerinde çalışır
- DEX: Dalvik VM için compile edilmiş class dosyaları

Android Paket İçeriği



- Compile edilmiş resource'lar
  - R.java
  - string.xml
  - ids.xml
  - layouts.xml

Kaynak Kod Dönüşümü - Decompile

Decompile
Dex -> JAR ->JAVA

dex2jar
enjarify

Java Decompiler
- JD-GUI
- JAD
- Jadx
- Procyon
- ...

Kaynak Kod Dönüşümü - Decompile

- **Dex2jar** aracı ile class dosyasına dönüştürülmüş olan Android uygulaması, **JD-GUI** aracı ile kaynak koduna (decompile) geri çevirilebilir.

- Decompile işleminin yetersiz olduğu durumlarda incelenecek uygulamayı Disassembling işleminden geçirmek gerekebilir.

Android Tersine Mühendislik

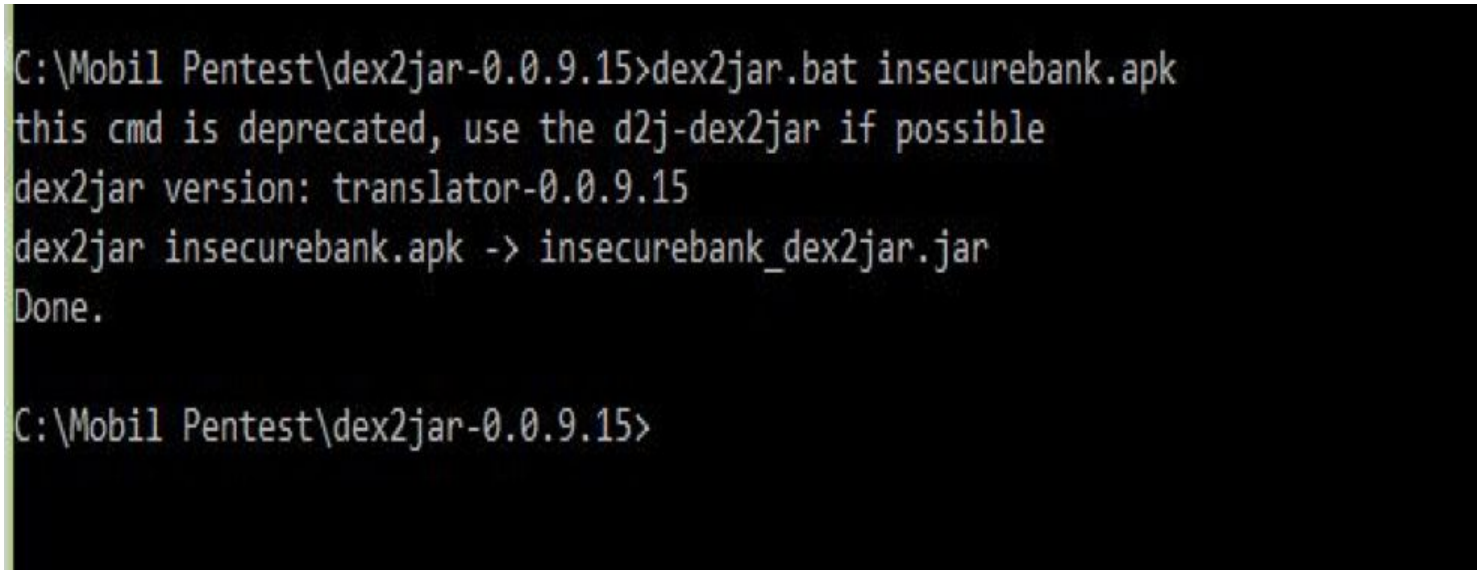
Dex2Jar

Adından da anlaşılacağı üzere dex dosyalarını jar dosyalarına çevirmektedir.

Resimde görüldüğü üzere apk dosyamızı jar haline getirdik.

Android Tersine Mühendislik

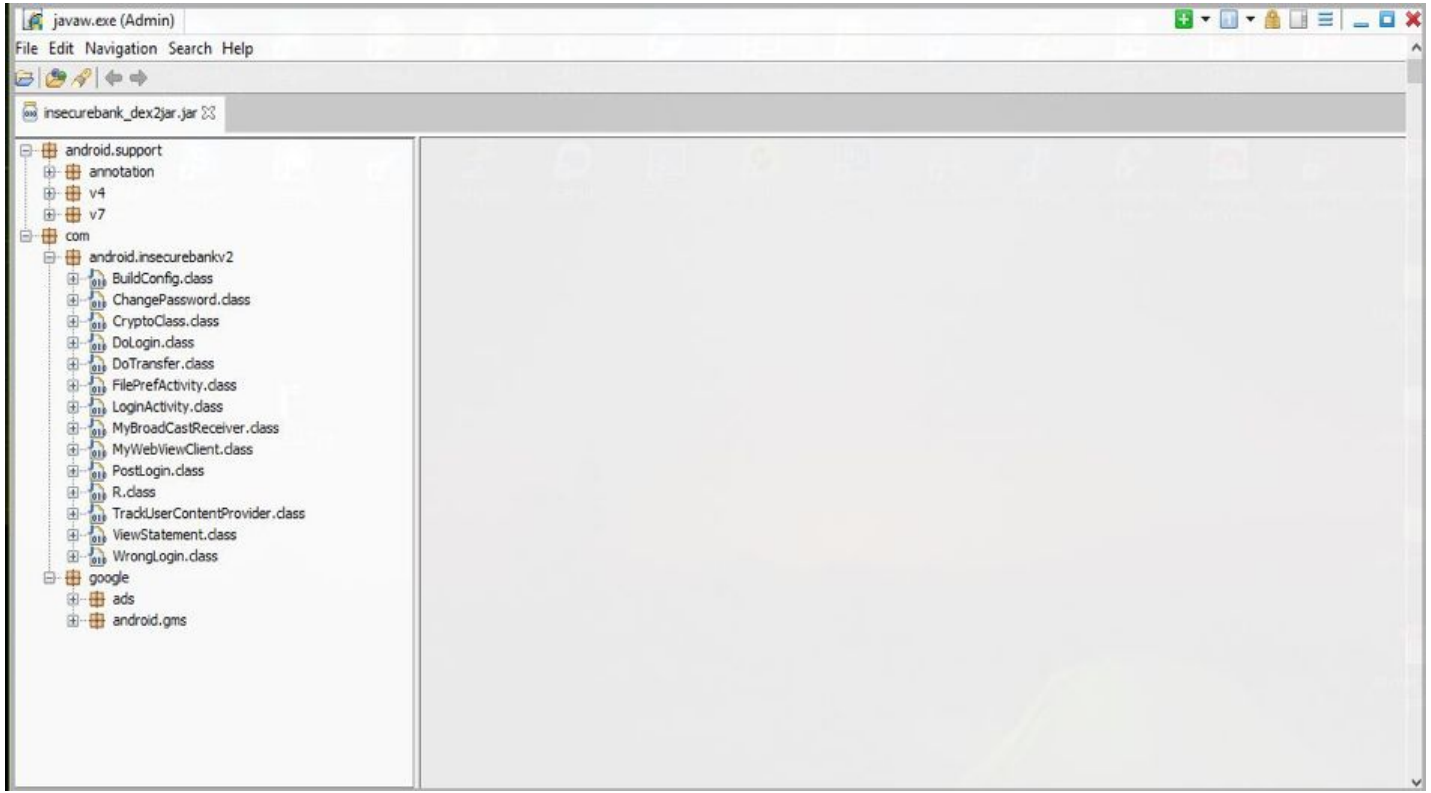
Dex2Jar



Android Tersine Mühendislik

JD-GUI

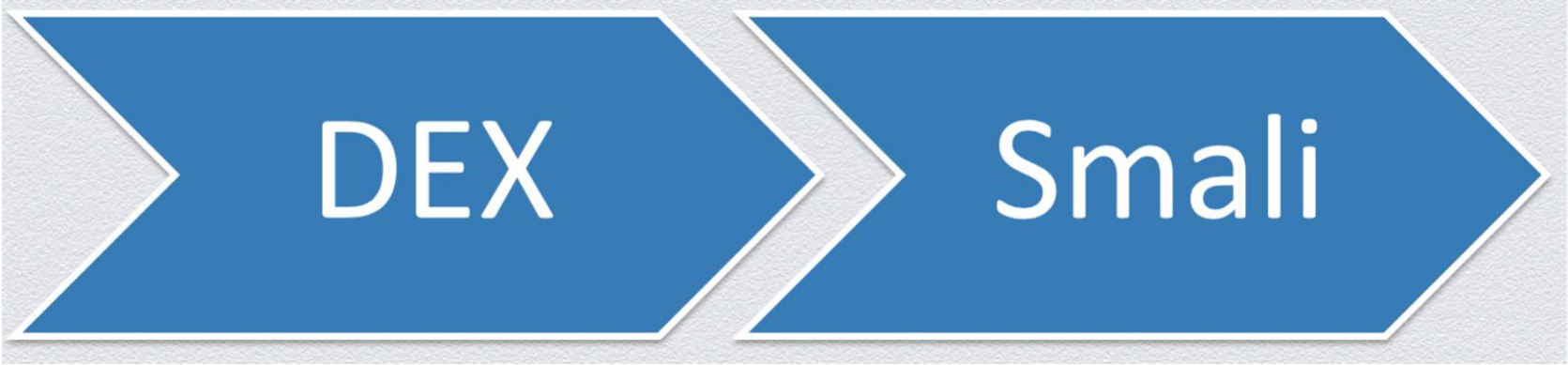
JAR haline getirdiğimiz dosyamızı görüntülemek için kullanacağız.



Kaynak Kod Dönüşümü - Decompile

- Decompile edilmiş JAR kodu tekrar compile edilerek çalıştırılabilir hale getirilemez.
- Decompile edilen kod yaklaşık koddur. %100 geri dönüşüm gerçekleştirilemez.
- Dex2jar çıktısından elde edilen JAR kodu çalıştırılamaz.
- Dalvik Bytecode, JAR koduna dönüştürülerek kolay okunabilir ve anlaşılabilir hale gelir.

Disassembling



Disassemble
Dex -> .smali

Dex Disassembler

- Baksmali
- Dedexer
- apktool

Android Tersine Mühendislik

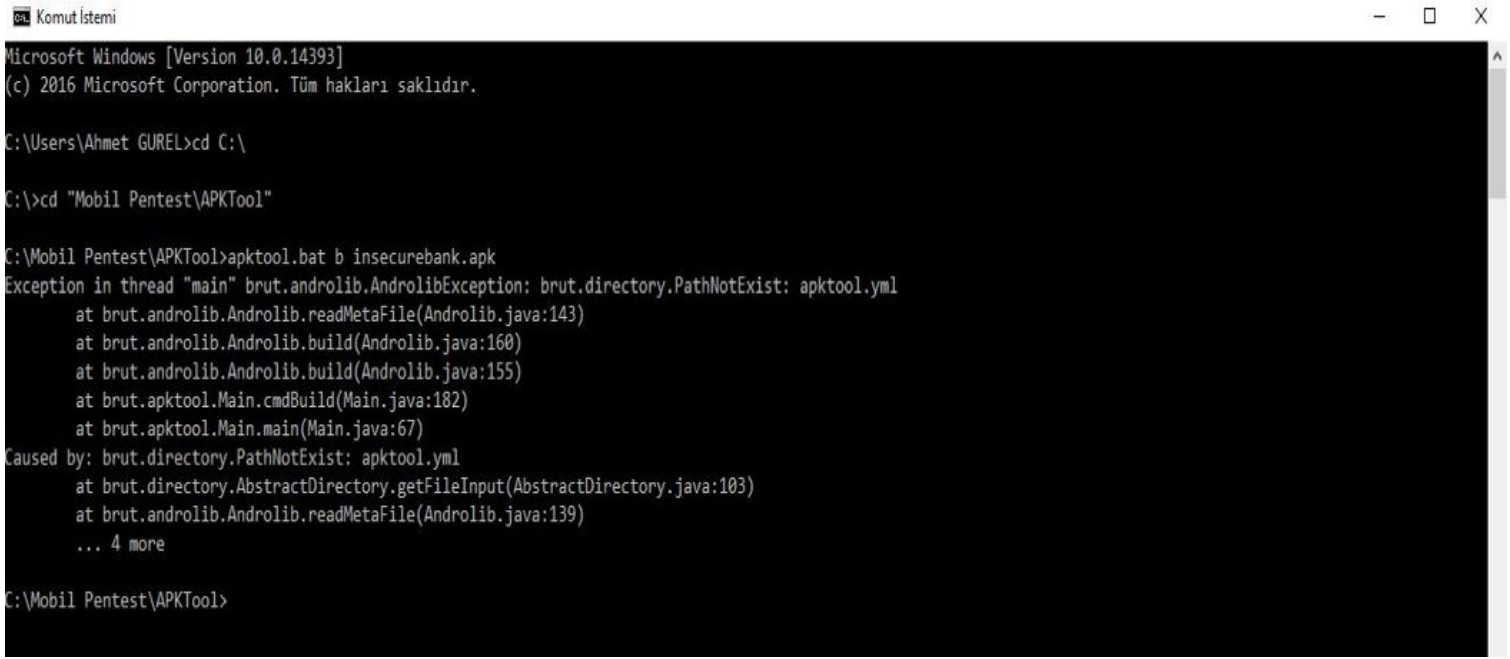
APKTool

APKTool apk dosyalarını decompile ederek smali kodlarına dönüştürür.

Kullanımı oldukça basit aşağıdaki resimde görüldüğü üzere b parametresi ile decompile etmekte.

Android Tersine Mühendislik

APKTool



Disassembling

Disassemble DEX **->** .smali

DEX dosyası okunabilir Dalvik Bytecode'a dönüştürülüyor.

.smali uzantılı Dalvik bytecode modifiye edilebilir.

Modifiye edilen Dalvik bytecode tekrar imzalanır,
paketlenir ve cihazda çalıştırılabilir.

Baksmali aracı ile disassemble işlemi yapılabilir.

Alınabilecek Önlemler / Anti Reversing

Kod Karmaşıklaştırma - Obfuscation

Progaurd

Dexgaurd

Java
obfuscators

Alınabilecek Önlemler / Anti Reversing

Kod Karmaşıklaştırma - Obfuscation

- Kullanılmayan sınıflar, metodlar temizlenir.
- Bytecode optimize edilir.
- Kullanılmayan instructionlar temizlenir.
- Geriye kalan sınıflar, metodlar, alanlar ve değişkenler anlamsız kısa isimlerle adlandırılır.
- **ProGuard, DexGuard** ile obfuscation önlemi alınabilir.
- **ALLATORI** ile analistleri çıldırtacak obfuscation önlemleri alınabilir.

Alınabilecek Önlemler / Anti Reversing

Kod Karmaşıklaştırma - Obfuscation

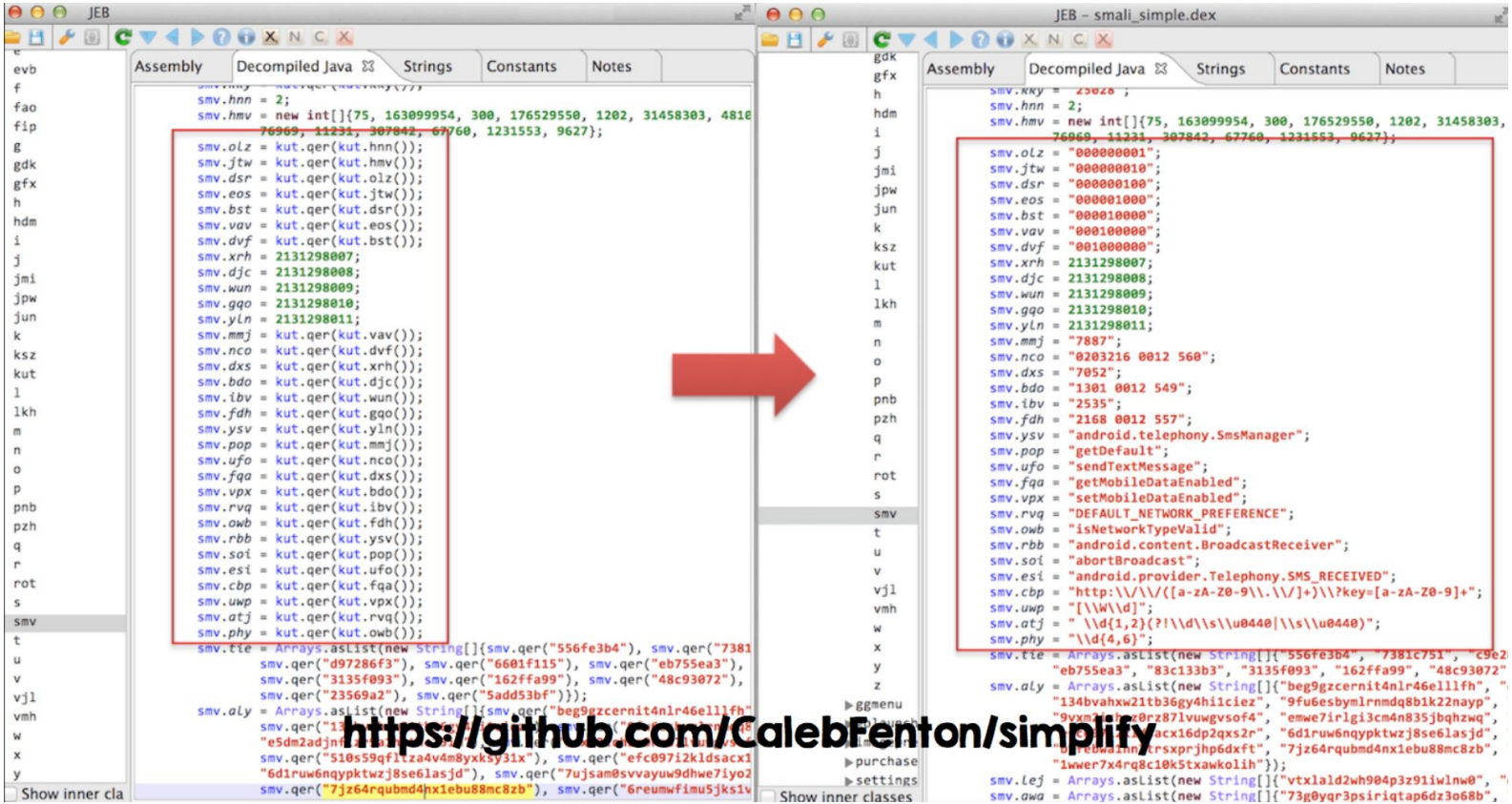
Alınabilecek Önlemler / Anti Reversing

Kod Karmaşıklaştırma - Obfuscation

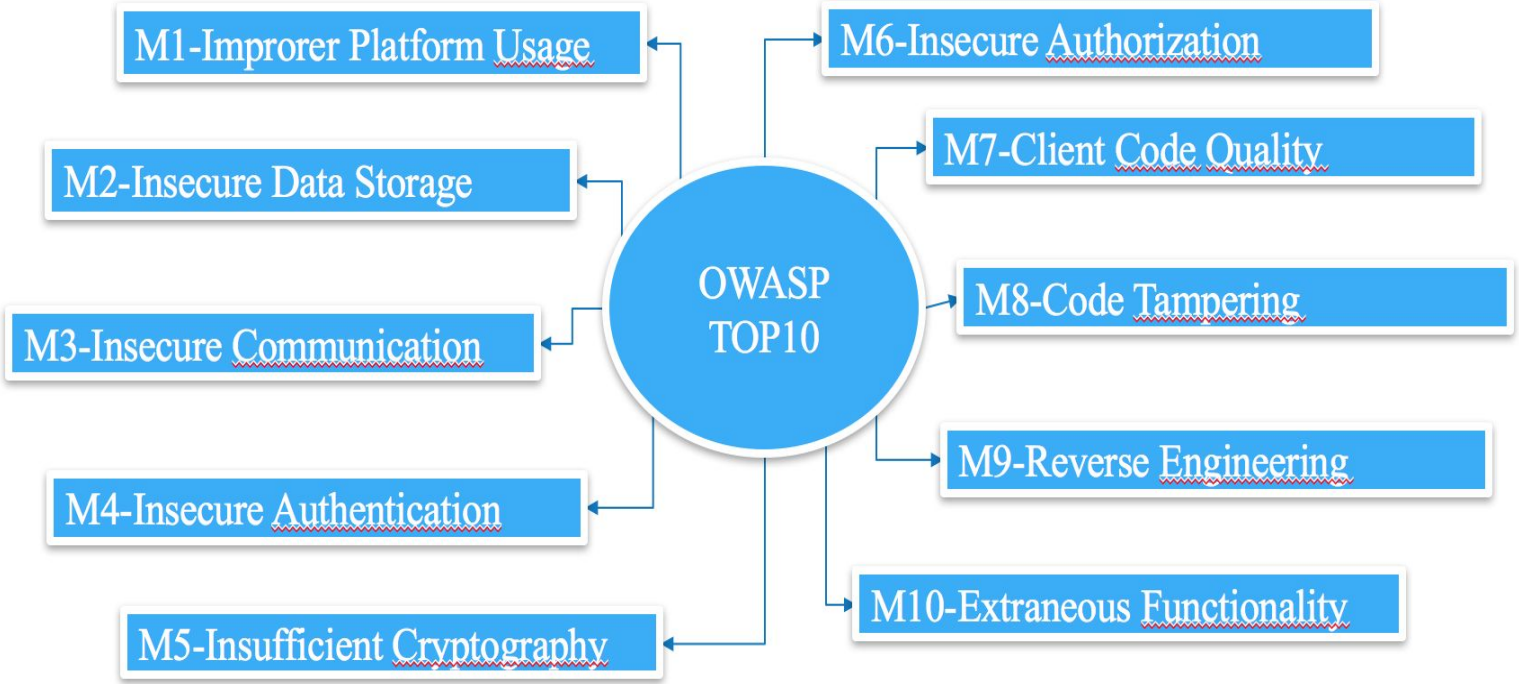
```
public class MyVehicleClass{

    private Motor        myMotor;
    private Tekerlek     myTekerler;
    private int          vitesSayisi;

    public int suratHesapla(int sure){

        ...

        return sonSurat;
    }
}
```

```
public class A{

    private B        a;
    private C        b;
    private int      c;

    public int a(int a){

        ...

        return c;
    }
}
```

Deobfuscation - Simplify



OWASP MOBILE TOP 10



https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

OWASP TOP10

M1– Improper Platform Usage (Hatalı Platform Kullanımı)

- Platform güvenlik kontrollerinin hatalı veya kötüye kullanılması
- Mobil uygulamalarda bu riski yaşamanın çeşitli yolları vardır :
- Yayımlanmış kılavuzlar ihlali
- Sözleşme veya yaygın bir uygulama ihlali
- Kasıtsız kötüye kullanma

OWASP TOP10

M2 – Insecure Data Storage (Güvensiz Veri Saklama)

- Bu kısım güvensiz veri depolama ve istenmeyen veri sızıntılarını kapsar.
- SQL databases
- Log files
- XML data stores ou manifest files;
- Binary data stores
- Cookie stores
- SD card
- Cloud synced.

OWASP TOP10

M3 - Insecure Communication (Güvenli Olmayan İletişim)

- Güvenlik açısından yanlış SSL versiyonlarının kullanımı
- Hassas verilerin clear-text olarak gönderilmesi
- İletişimin sağlandığı kanallar arasında zayıf iletişimlerin kurulması

OWASP TOP10

M4 - Insecure Authentication (Güvensiz Doğrulama)

- Mobil uygulama, bir erişim belirteci sağlamadan bir arka uç API hizmet isteğini anonim olarak çalıştırabilirse, bu uygulama güvensiz kimlik doğrulama uygular.
- Mobil uygulama herhangi bir parolayı veya paylaşılan sırları cihazda yerel olarak saklarsa, güvensiz kimlik doğrulama sorunuyla karşılaşır.
- Mobil uygulama bir şifre girmeyi kolaylaştırmak için zayıf bir şifre politikası kullanıyorsa, güvensiz kimlik doğrulama uygular.

OWASP TOP10

M5 - Insufficient Cryptography (Yetersiz Şifreleme)

- Hassas kod bilgileri şifrelenir. Ancak yine de şifreleme yetersiz kalabilir.
- Yaygın hatalar:
- Zayıf şifreler
- Yanlış şifreleme
- Plaintext attack

OWASP TOP10

M6 - Insecure Authorization (Güvensiz Yetki)

- Yetkilerde hataları yakalama
- Örneğin; cihaz kayıt, kullanıcı tanımlama kimlik doğrulama sorunları farklıdır.
- Eğer uygulamada kullanıcı kimlik doğrulama yoksa bu kimlik doğrulama hatası değil başarısız yetkilendirme hatasıdır.

OWASP TOP10

M7 – Client Code Quality (İstemci Kod Kalite Sorunları)

- Mobil istemcideki kod düzeyinde uygulama hatasıdır.
- Sunucu tarafındaki kodlama hatalarından farklıdır.
- Buffer overflows, format string güvenlik açıkları ve çözümün mobil cihaz üzerinde çalışan bazı kodu tekrar yazmak olan çeşitli kod düzeyindeki hatalar gibi riskleri bu yakalar.

OWASP TOP10

M8 – Code Tampering (Kod Kurcalama)

- Uygulama mobil cihaza yüklendikten sonra kod ve veri kaynakları orada bulunur.
- Bir saldırgan doğrudan uygulamanın kodunu veya kullandığı sistem API'lerini değiştirebilir.
- Böylece saldırgan kişisel ya da parasal kazanç için yazılımın kullanım amacını yıkarak kötü amaçlı kullanması yöntemidir.

OWASP TOP10

M9 - Reverse Engineering (Tersine Mühendislik)

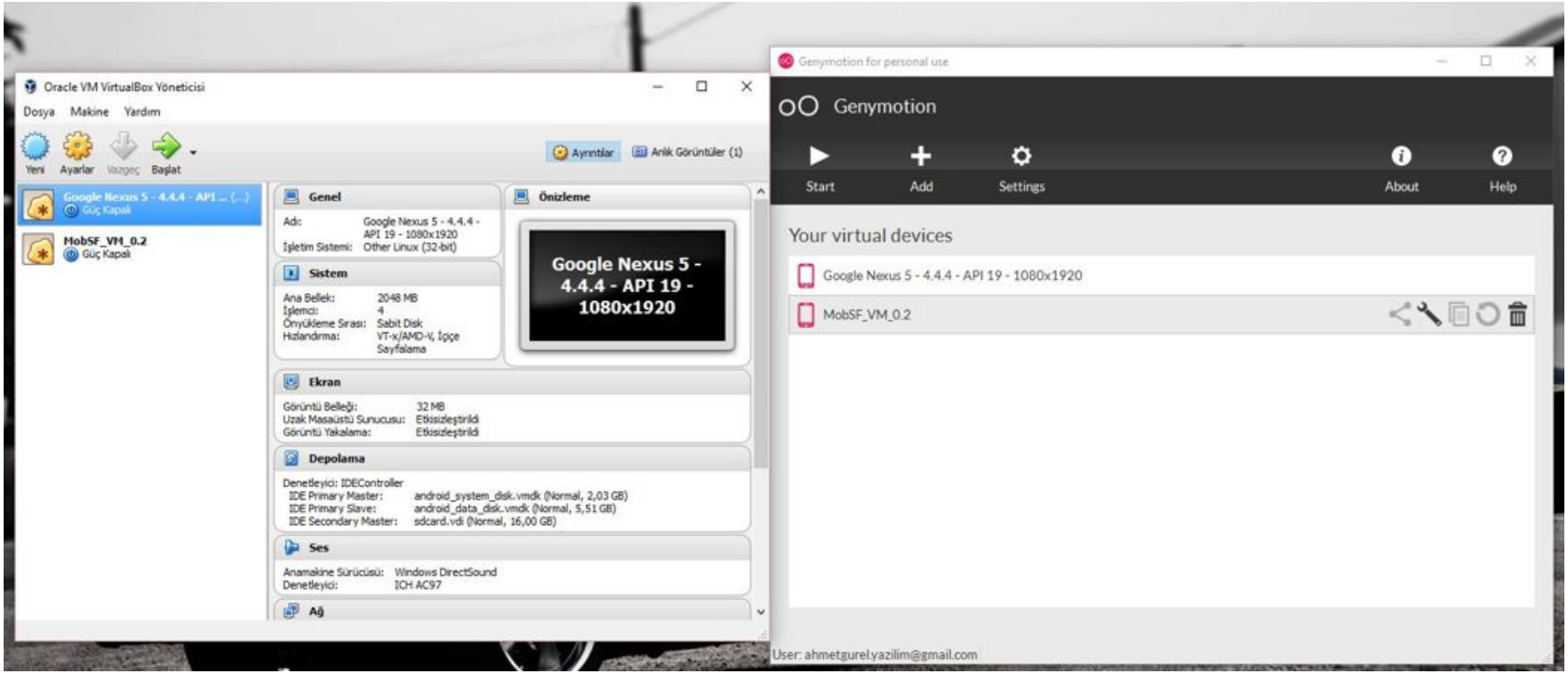
- Uygulamanın kaynak kodu, kütüphaneleri, algoritması ve diğer kaynakların tespitidir.
- Saldırgan doğabilecek açıkları, parolaları vb. bilgileri yararına kullanabilir.

OWASP TOP10

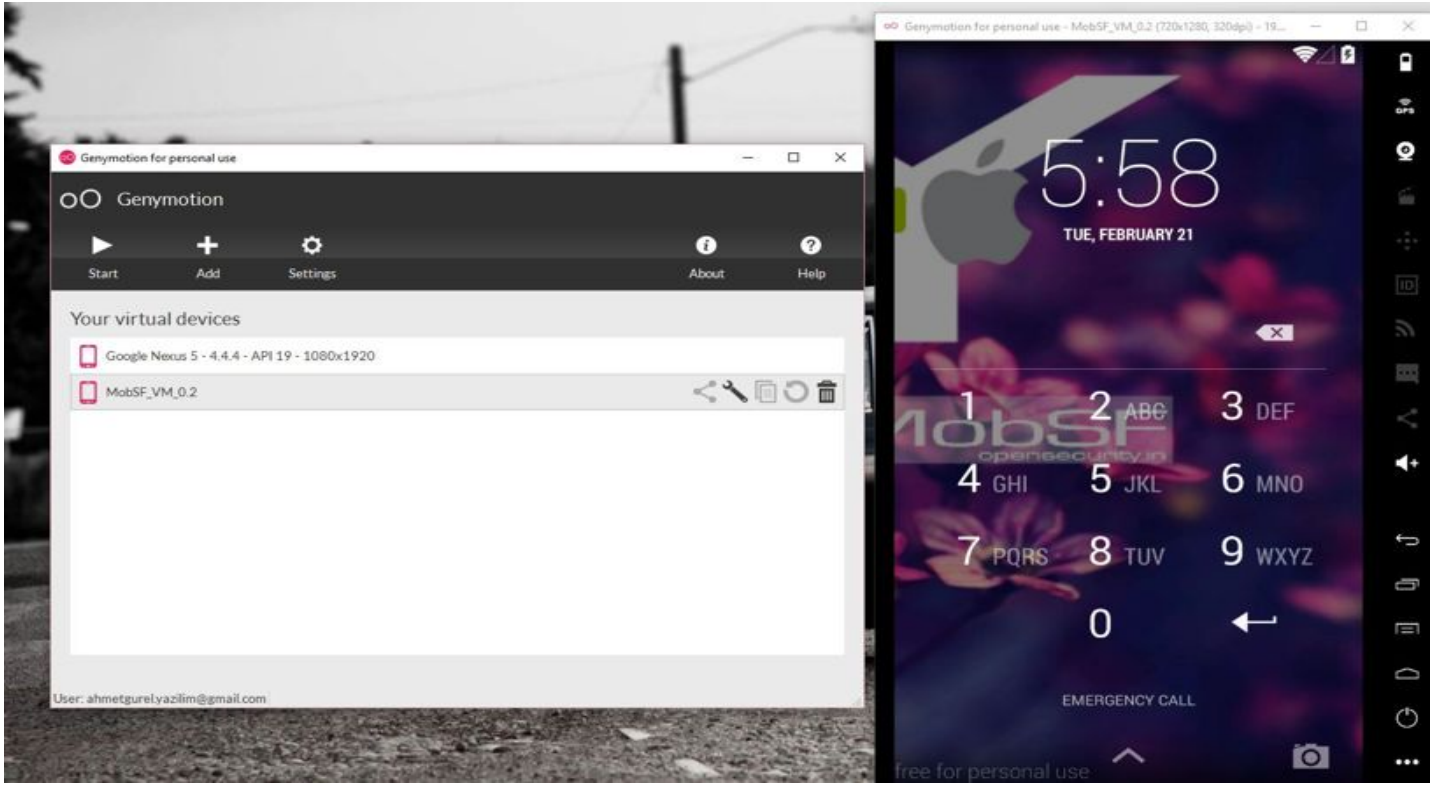
M10 - Extraneous Functionality (Gereksiz İşlevsellik)

- Genellikle geliştiricilerin arka kapı bırakması
- Örneğin, geliştirici bir uygulamada şifre unutmuş olabilir.
- Diğer bir örnek de test sırasında 2 faktörlü kimlik doğrulama devre dışı bırakmasıdır.

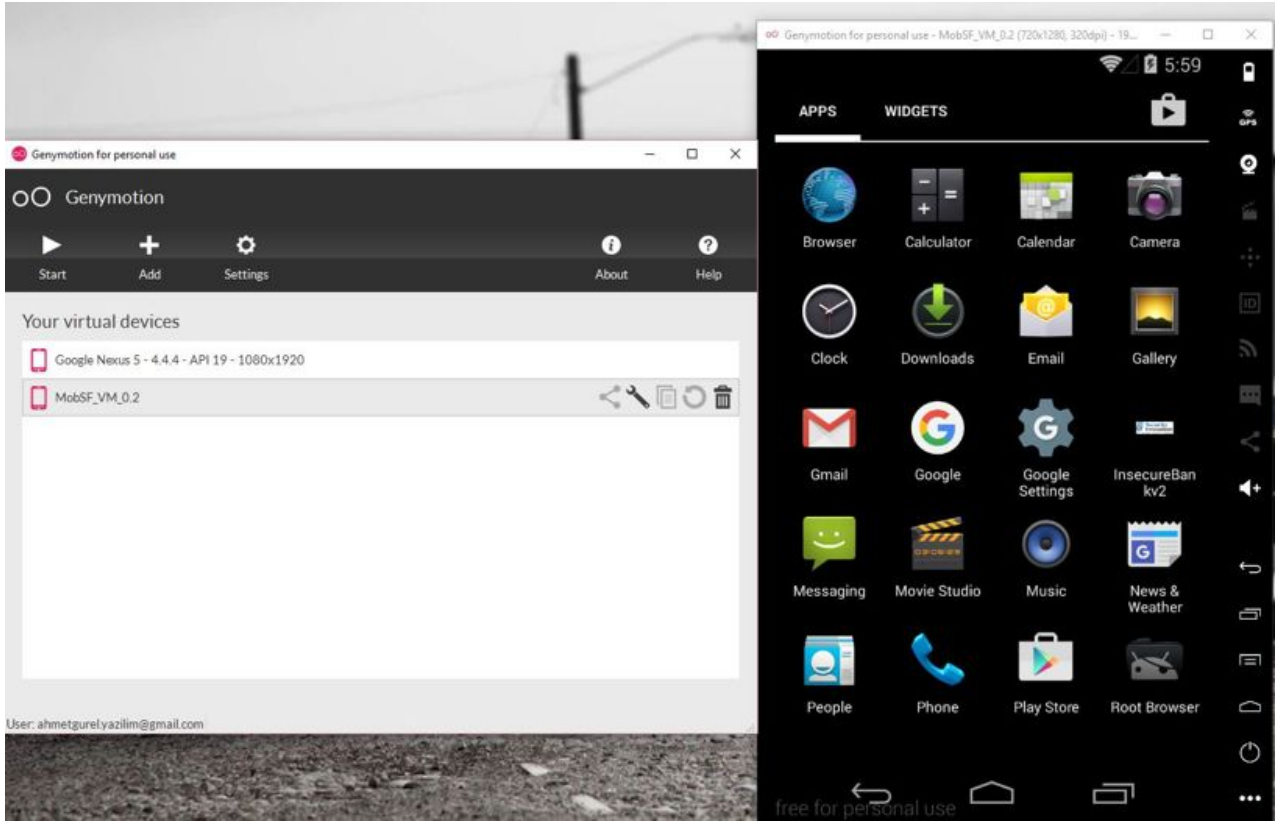
Ortam Kurulumu



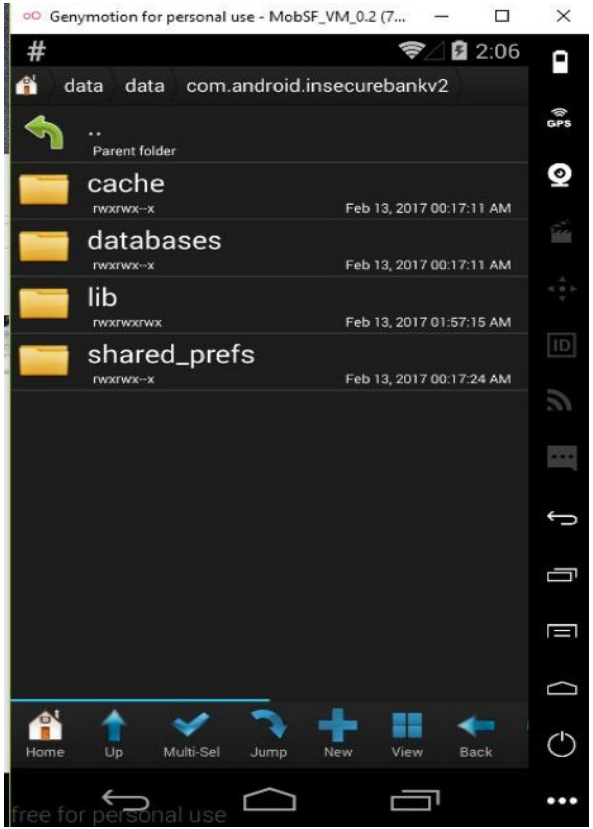
Ortam Kurulumu

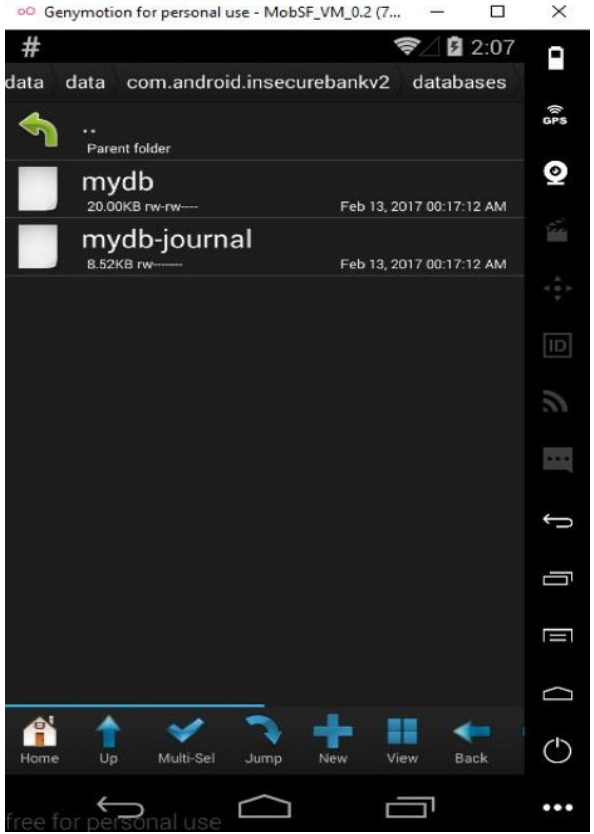


Ortam Kurulumu



Android Uygulama Dosyaları

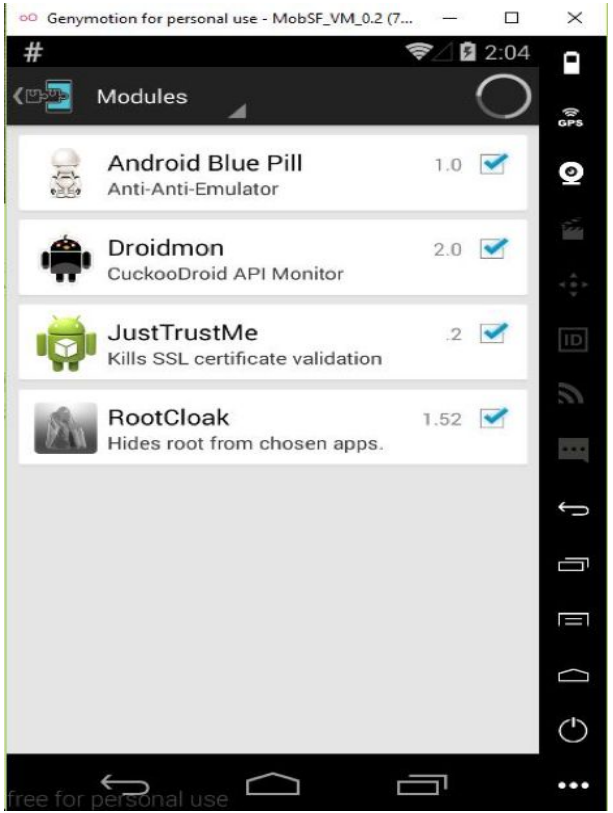




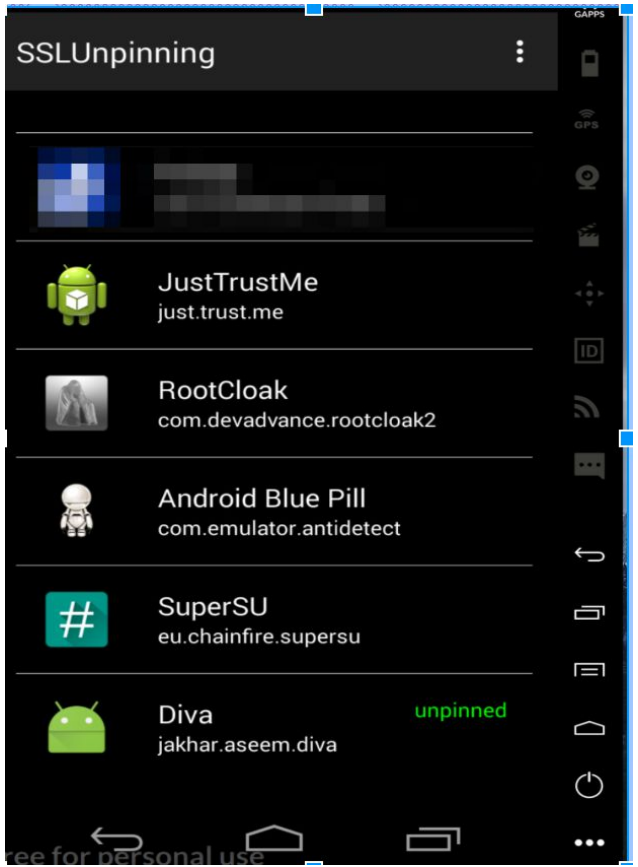
Xposed Modülleri

Xposed Modülleri Android cihazda uygulamaları özelleştirmek değiştirmek için kullanılır.
Uygulama geliştirilirken yazılan kontrollerin, izinlerin değiştirilmesine imkan verebiliyor.

Mesela yukarıda gördüğümüz RootCloak modülü bir uygulama cihaz root lumu diye kontrol edip, çalışmıyorsa bu kontrolü engelleme/atlatmaya yaramaktadır ve güvenlik testleri için önemli bir yer tutmaktadır.
Bunun gibi birçok modül bulunmaktadır.

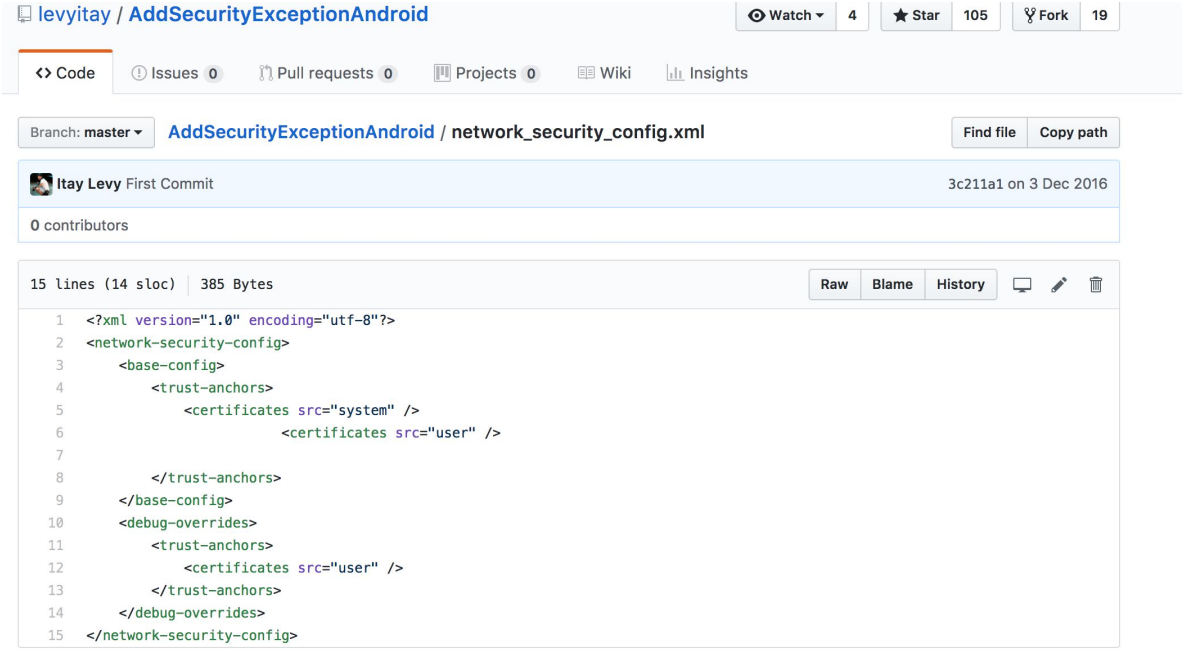


SSL Pinning Bypass



SSL Pinning Bypass

Android 7.0'da, Google, kullanıcıların Sertifika Yetkililerine (CA) güvenme biçiminde değişiklikler getirdi. Bu değişiklikler, üçüncü şahısların uygulamadan gelen ağ isteklerini dinlemelerini engeller.



Android Uygulama İzinleri

Value	Meaning
"`normal`"	The default value. A lower-risk permission that gives requesting applications access to isolated application-level features, with minimal risk to other applications, the system, or the user. The system automatically grants this type of permission to a requesting application at installation, without asking for the user's explicit approval (though the user always has the option to review these permissions before installing).
"`dangerous`"	A higher-risk permission that would give a requesting application access to private user data or control over the device that can negatively impact the user. Because this type of permission introduces potential risk, the system may not automatically grant it to the requesting application. For example, any dangerous permissions requested by an application may be displayed to the user and require confirmation before proceeding, or some other approach may be taken to avoid the user automatically allowing the use of such facilities.
"`signature`"	A permission that the system grants only if the requesting application is signed with the same certificate as the application that declared the permission. If the certificates match, the system automatically grants the permission without notifying the user or asking for the user's explicit approval.
"`signatureOrSystem`"	A permission that the system grants only to applications that are in the Android system image *or* that are signed with the same certificate as the application that declared the permission. Please avoid using this option, as the `signature` protection level should be sufficient for most needs and works regardless of exactly where applications are installed. The "`signatureOrSystem`" permission is used for certain special situations where multiple vendors have applications built into a system image and need to share specific features explicitly because they are being built together.

Android Uygulama İzinleri

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="[redacted]">
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

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

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

Mobil Sızma Testi Araçları

ADB (Android Debug Bridge)

"**Android Debug Bridge** (Android Ayıklama Köprüsü - **adb**), bir emulator kopyasının veya Android'li bir cihazının durumunu yönetmeye izin veren çok yönlü bir araç.

Mobil Sızma Testi Araçları

ADB (Android Debug Bridge)

- **ADB push :** Cihaza dosya gönderir. **adb push uygulama.apk /sdcard/uygulama.apk**
- **ADB pull :** Cihazdan dosya alır. **adb pull /system/app/uygulama.apk**
- **ADB install :** Cihaza uygulama yükler. **adb install uygulama.apk**
- **ADB shell :** Cihazın komut satırına (terminal) bağlanır. **adb shell**

Mobil Sızma Testi Araçları

ADB (Android Debug Bridge)

Mobil Sızma Testi Araçları

Andro Guard

AndroGuard python ile geliştirilen statik kod analizi yapan bir araçtır.

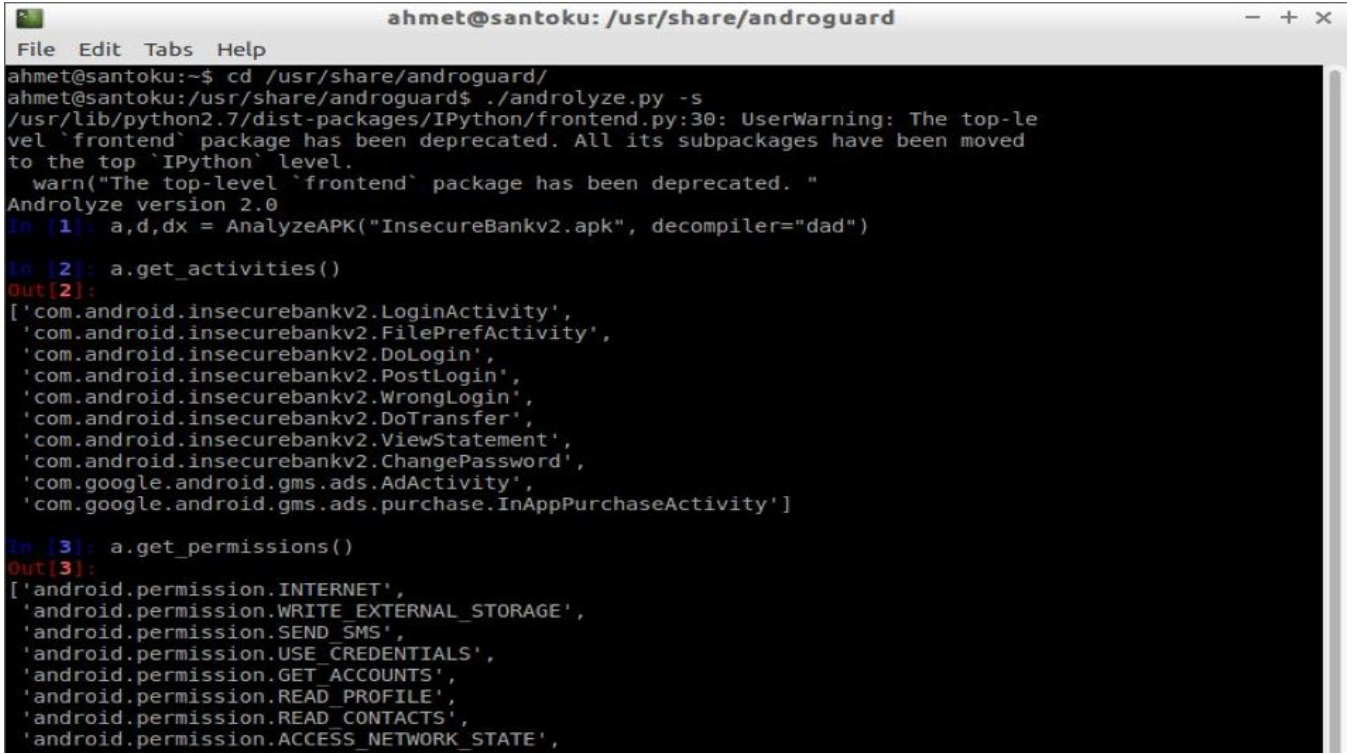
San Toku sanal makinasının iiçinde kurulu olarak gelmektedir.

AndroGuard Santoku üzerinde bu şekilde çalıştırılmaktadır. İlk olarak kurulu olduğu dizine gittik ve apk dosyamızı da oraya taşıdık. **./androlyze.py -s** ile çalıştırdık. İlk satırımıza

a,d,dx= AnalyzeAPK("Insecurebankv2.apk", decompiler="dad") komutunu yazarak apk dosyamızı göstererek decompile ediyoruz. Daha sonra programın parametreleri ile birçok analiz edebilmekteyiz. Resimde uygulamanın activitylerini ve izinlerini getirdik.

Mobil Sızma Testi Araçları

Andro Guard



Mobil Sızma Testi Araçları

Burp Suite

Burp gelişmiş bir proxy yazılımıdır.

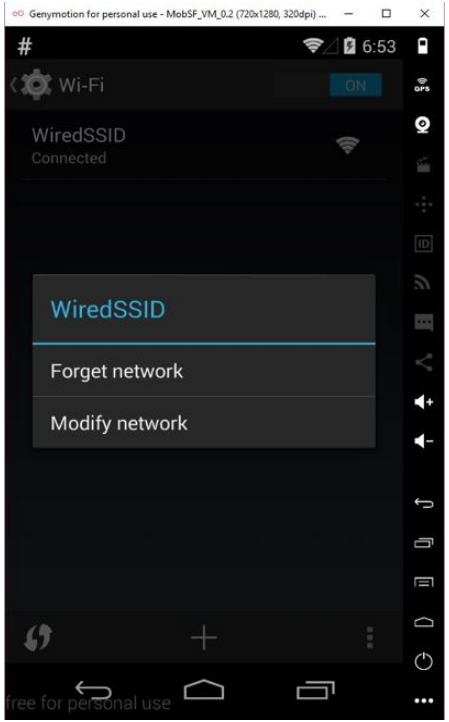
Bunun dışında birçok teste yardımcı olmakta ve imkan tanımaktadır.

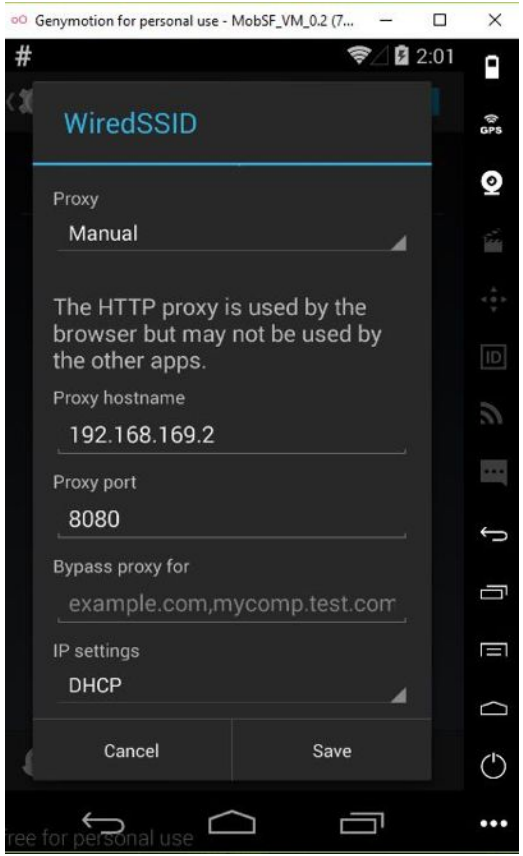
Web Testlerinin olmazsa olmazı Burp Mobil testlerimizde de o kadar önemli.

Şimdi Burp Suite Emulatorümüzden bağlanmayı bakalım beraber.

Mobil Sızma Testi Araçları

Burp Suite





Mobil Sızma Testi Araçları

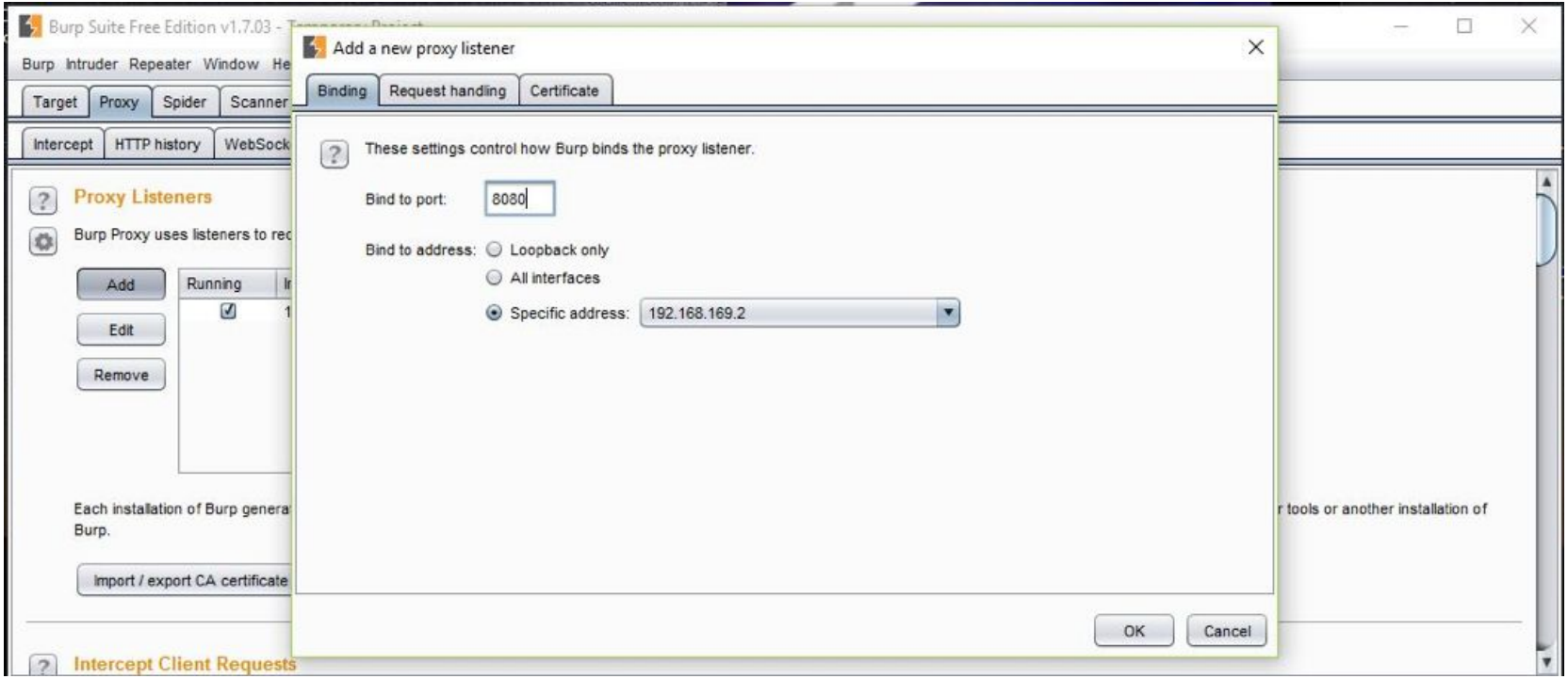
Burp Suite

Ayarlara (Settings) e girerek daha sonra Wi-Fi ye tıklayarak WiredSSID nin üzerine basılı tutarak Modify network diyerek Proxy belirliyoruz.

Burada IP adresi test yaptığınız makinenin IPsidir kendi ana makineniz ya da yukarıda bahsettiğim makinelari indirdiyseniz onun IP adresidir.

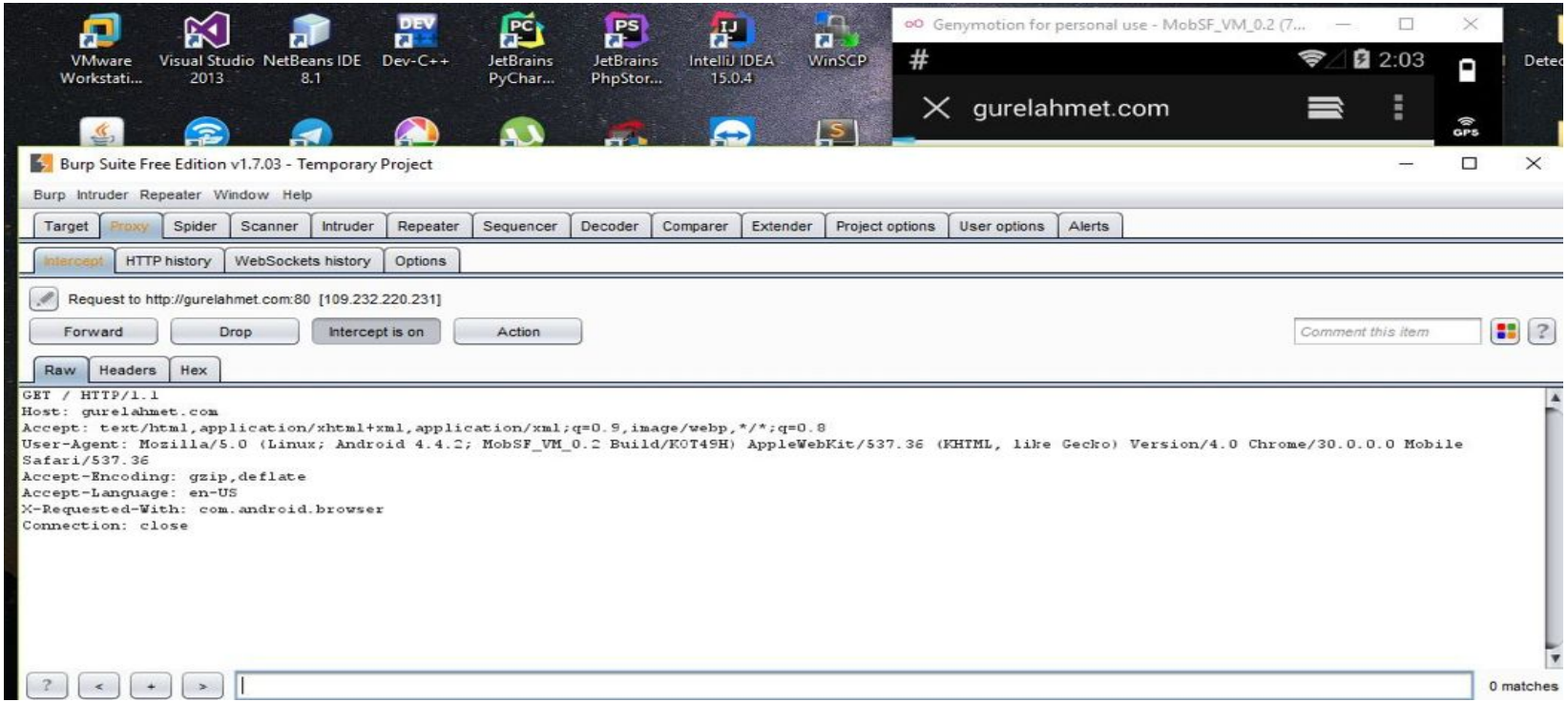
Mobil Sızma Testi Araçları

Burp Suite



Mobil Sızma Testi Araçları

Burp Suite



Mobil Sızma Testi Araçları

Sqlite Veritabanı incelemede Sqlite Browser ve Sqlite3 Kullanımı

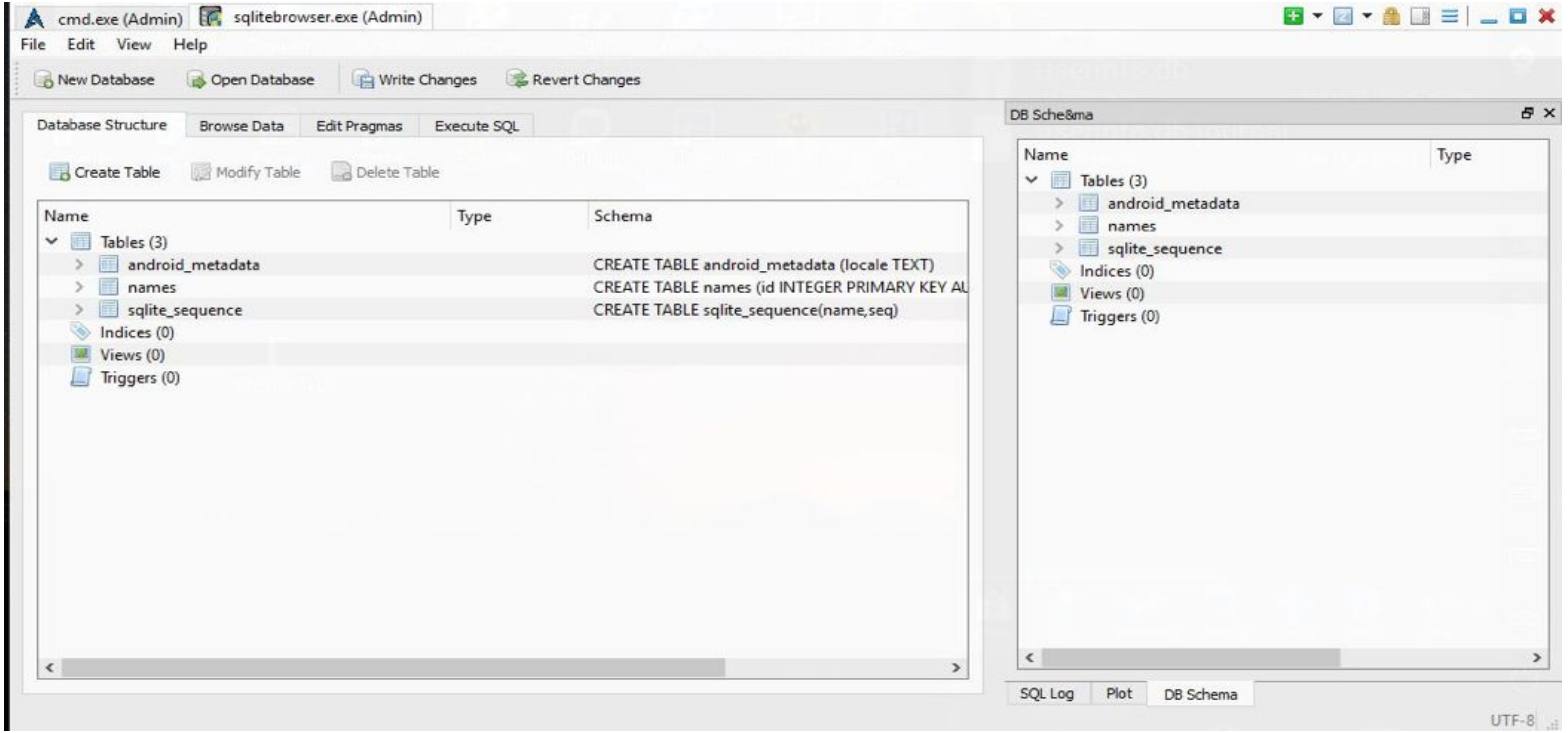
Uygulamayı cihazımıza aktardıktan sonra eğer dosyalar cihazda tutuluyorsa veritabanı dosyalarını ADB ile kendi bilgisayarımıza indirebiliriz.

Bu database dosyalarının içeriğini Sqlite Browser ile görüntüleyebiliriz.

Bunun dışında da Sqlite3 ile veritabanını seçerek sorgular yazıp bununla da görüntüleyebilmekteyiz.

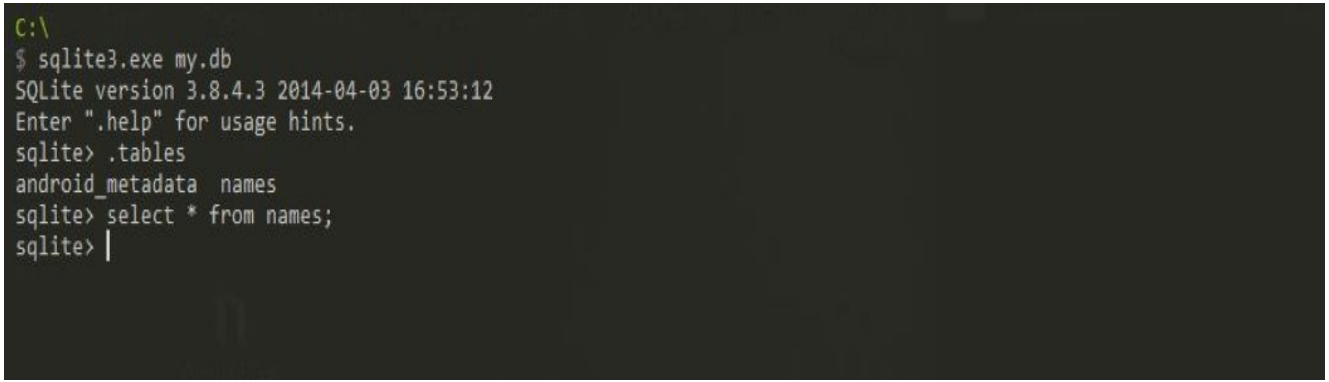
Mobil Sızma Testi Araçları

Sqlite Veritabanı incelemede Sqlite Browser ve Sqlite3 Kullanımı



Mobil Sızma Testi Araçları

Sqlite Veritabanı incelemede Sqlite Browser ve Sqlite3 Kullanımı



Mobil Sızma Testi Araçları

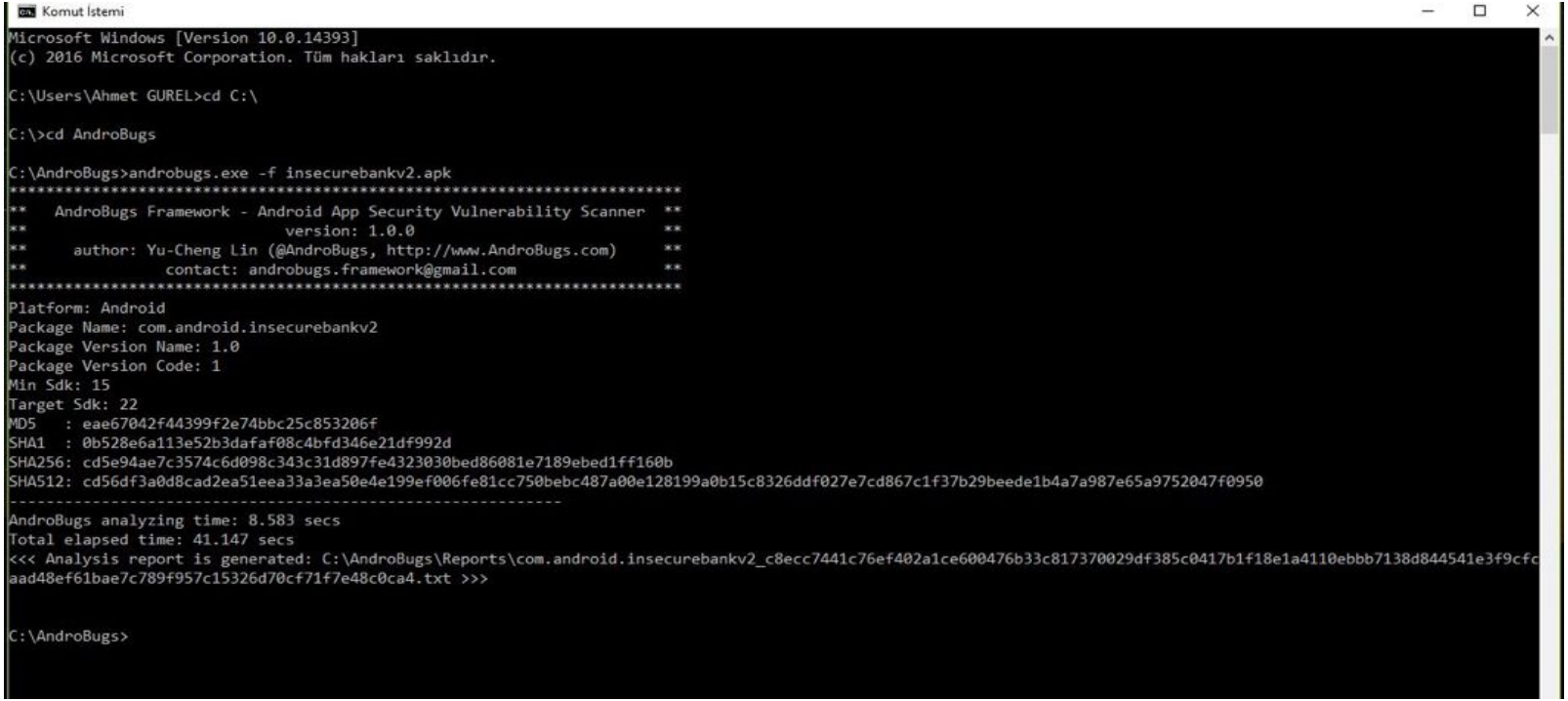
AndroBugs Framework

AndroBugs Framework, Android uygulamalarda güvenlik testi gerçekleştiren frameworklerden bir tanesidir.

Kullanımı oldukça basittir. Konsol üzerinden biz **androbugs -f apk_dosyasi** şeklinde kullanarak frameworkumuzu çalıştırdık.

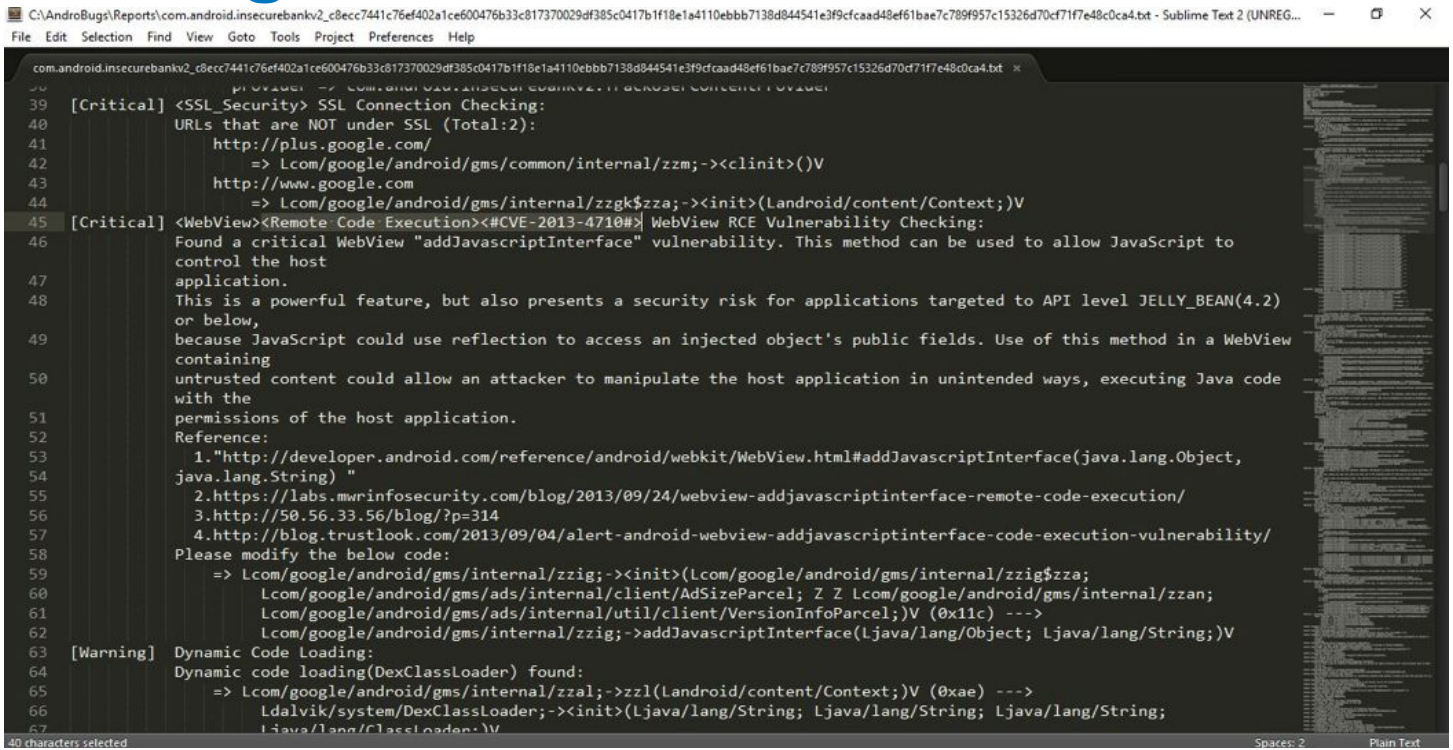
Bunun sonucunda kendi klasörünün altında Reports klasörünün altında detaylı rapor oluşturmaktadır.

Mobil Sızma Testi Araçları
AndroBugs Framework



Mobil Sızma Testi Araçları

AndroBugs Framework



Mobil Sızma Testi Araçları

Mobile Security Framework (MobSF)

AndroBugs gibi MobSF de mobil uygulama analizi yapan bir frameworktur.

Şu an en kullanışlı ve sağlam araç denebilir. Oldukça popüler ve güzel bir araçtır.

MobSF i indirdikten sonra Windows, Linux ve OSX e kurabilirsiniz. Kurduktan sonra localhostunuzda tarayıcıda çalışmakta ve apk dosyasını seçerek direk çalışmakta. Oldukça basit bir kullanımı vardır.

Adres olarak **127.0.0.1:8000** adresinde çalışmaktadır.

Mobil Sızma Testi Araçları

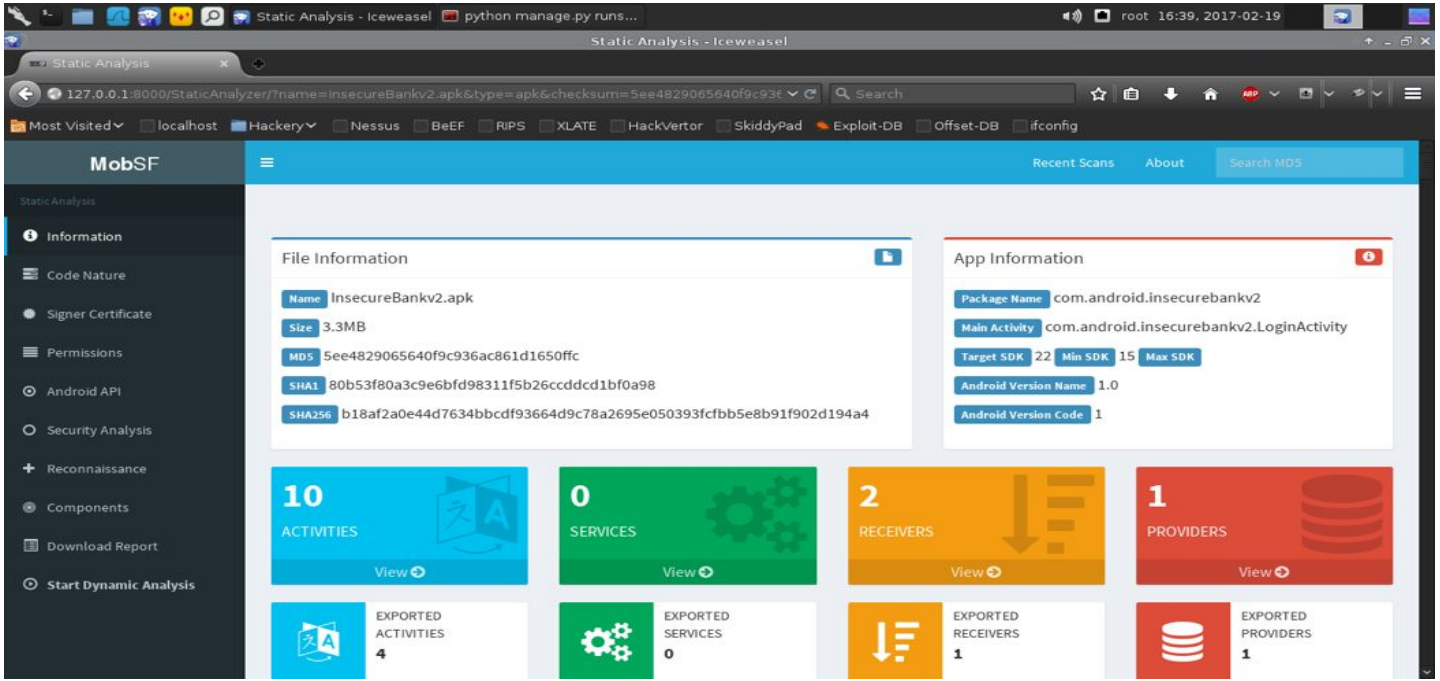
Mobile Security Framework (MobSF)

Mobil Sızma Testi Araçları

Mobile Security Framework (MobSF)

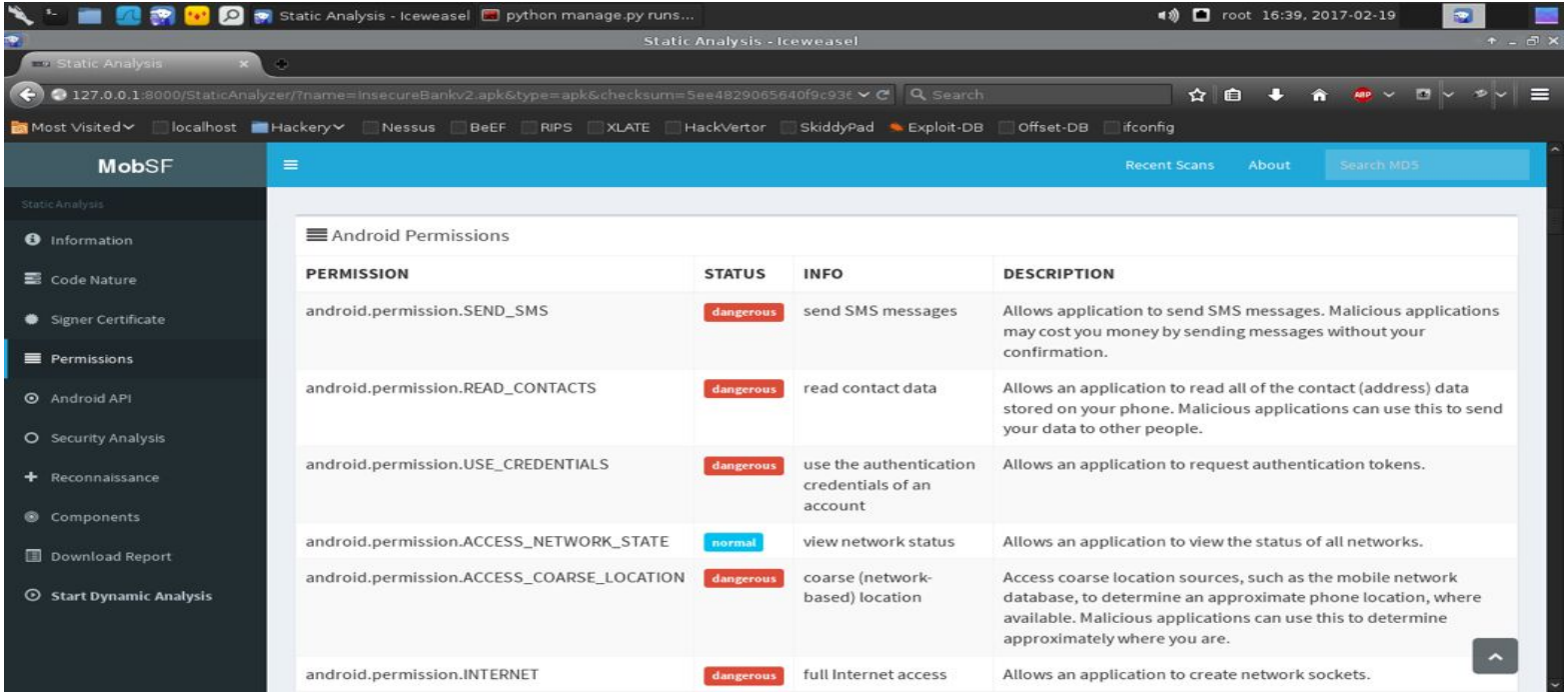
Mobil Sızma Testi Araçları

Mobile Security Framework (MobSF)



Mobil Sızma Testi Araçları

Mobile Security Framework (MobSF)



Mobil Sızma Testi Araçları

Drozer

Drozerda mobil testlerde kullanılan dinamik analiz yapan bir frameworktur.

Uygulama çalışırken test etme imkanı verir.

Diğer AndroBugs ve MobSF de ise statik analizi yaptık fakat uygulama çalışmıyordu. Drozer'da uygulama çalışırken testlerimizi gerçekleştiriyoruz.

Drozer'ı kendi test pc nize kurduktan sonra aynı zamanda emulatordeki mobil cihaza da yükleyerek birbirleri ile haberleşmesini sağlıyoruz.

Mobil Sızma Testi Araçları

Drozer

Kendi test pc nize drozeri kurduktan sonra agent.apk yı emulatore atmayı unutmayınız.

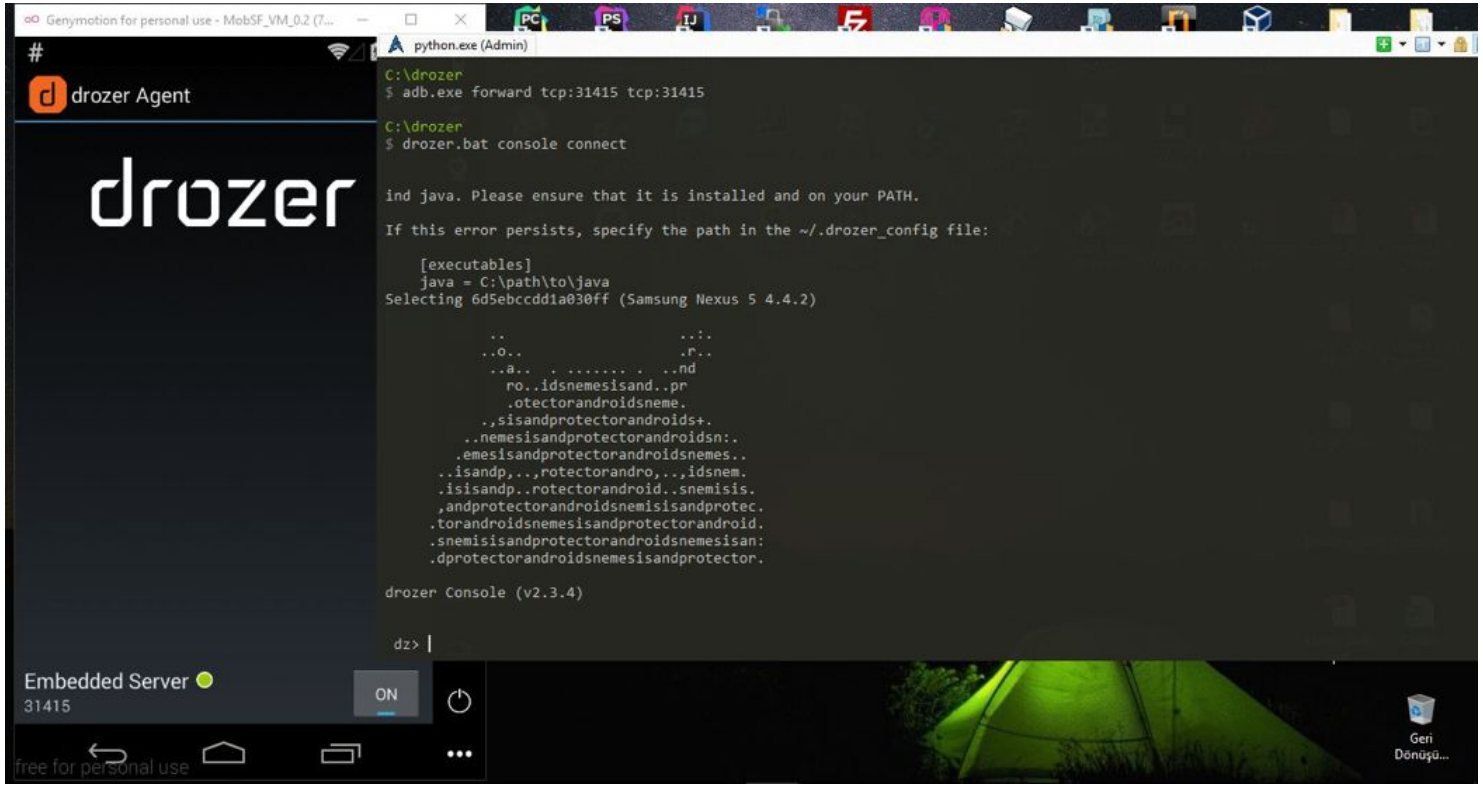
Bunu ister sürükle bırak ile istersenizde **adb. install agent.apk** komutu ile yapabilirsiniz.

Yükledikten sonra agent.apk yı emulatorde açarak off dan on'a alınız.

Daha sonra kurulum sayfasında gösterdiği gibi **adb forward tcp:31415 tcp:31415** komutunu verip **drozer console connect** diyerek drozerin komut satırına düşebilirsiniz.

Mobil Sızma Testi Araçları

Drozer



Mobil Sızma Testi Araçları

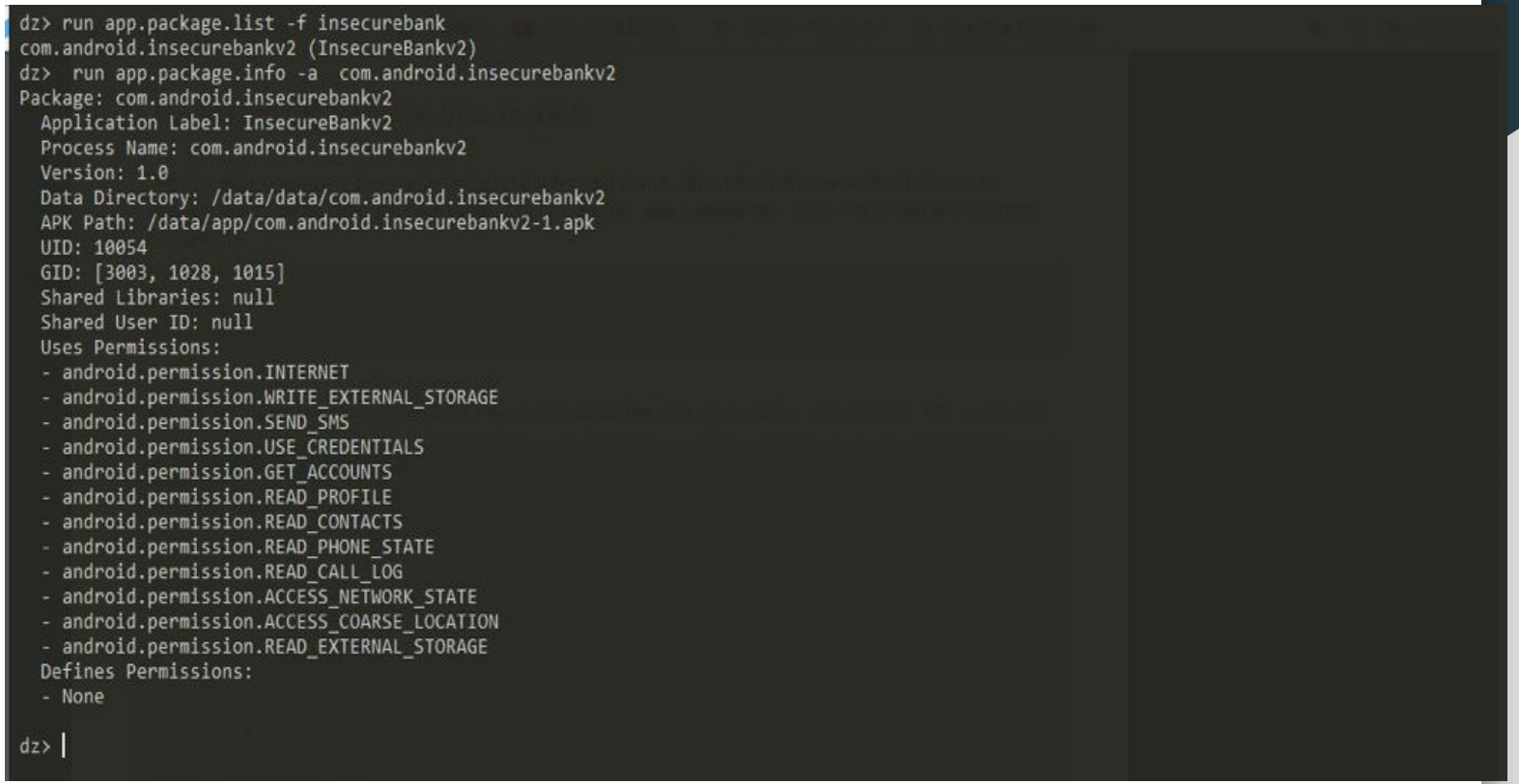
Drozer

run app.package.list -f insecurebank komutu ile kurulu paketler arasında adı insecurebank olan paketi arıyoruz.

```
          ..                    ..:.
         ..o..                  .r..
          ..a..  . ....... .  ..nd
            ro..idsnemesisand..pr
            .otectorandroidsneme.
         .,sisandprotectorandroids+.
       ..nemesisandprotectorandroidsn:.
      .emesisandprotectorandroidsnemes..
    ..isandp,..,rotectorandro,..,idsnem.
   .isisandp..rotectorandroid..snemisis.
   ,andprotectorandroidsnemisisandprotec.
  .torandroidsnemesisandprotectorandroid.
  .snemisisandprotectorandroidsnemesisan:
  .dprotectorandroidsnemesisandprotector.

drozer Console (v2.3.4)
dz>
dz> run app.package.list -f insecurebank
com.android.insecurebankv2 (InsecureBankv2)
dz>  run app.package.info -a  com.android.insecurebankv2
Package: com.android.insecurebankv2
  Application Label: InsecureBankv2
  Process Name: com.android.insecurebankv2
  Version: 1.0
  Data Directory: /data/data/com.android.insecurebankv2
  APK Path: /data/app/com.android.insecurebankv2-1.apk
  UID: 10054
  GID: [3003, 1028, 1015]
  Shared Libraries: null
  Shared User ID: null
  Uses Permissions:
  - android.permission.INTERNET
  - android.permission.WRITE_EXTERNAL_STORAGE
```

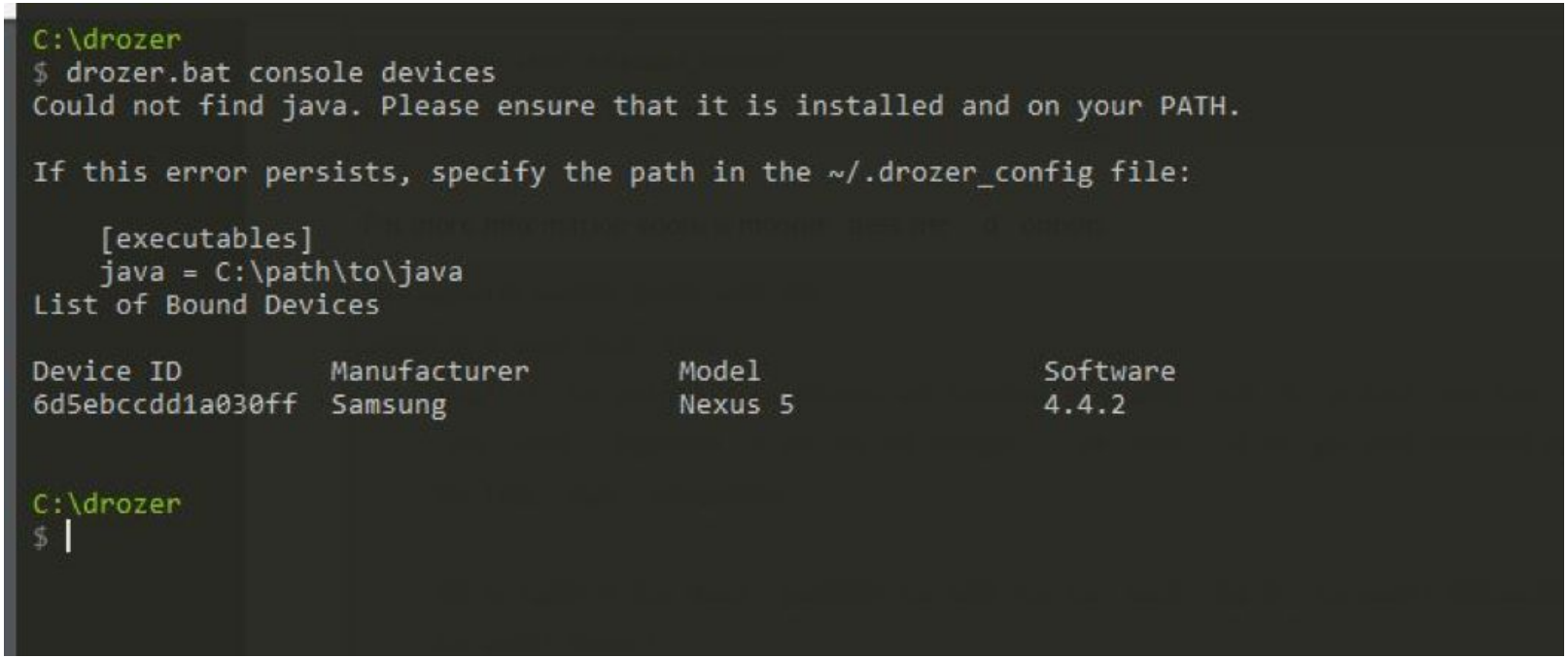
Mobil Sızma Testi Araçları

Drozer



Mobil Sızma Testi Araçları

Drozer



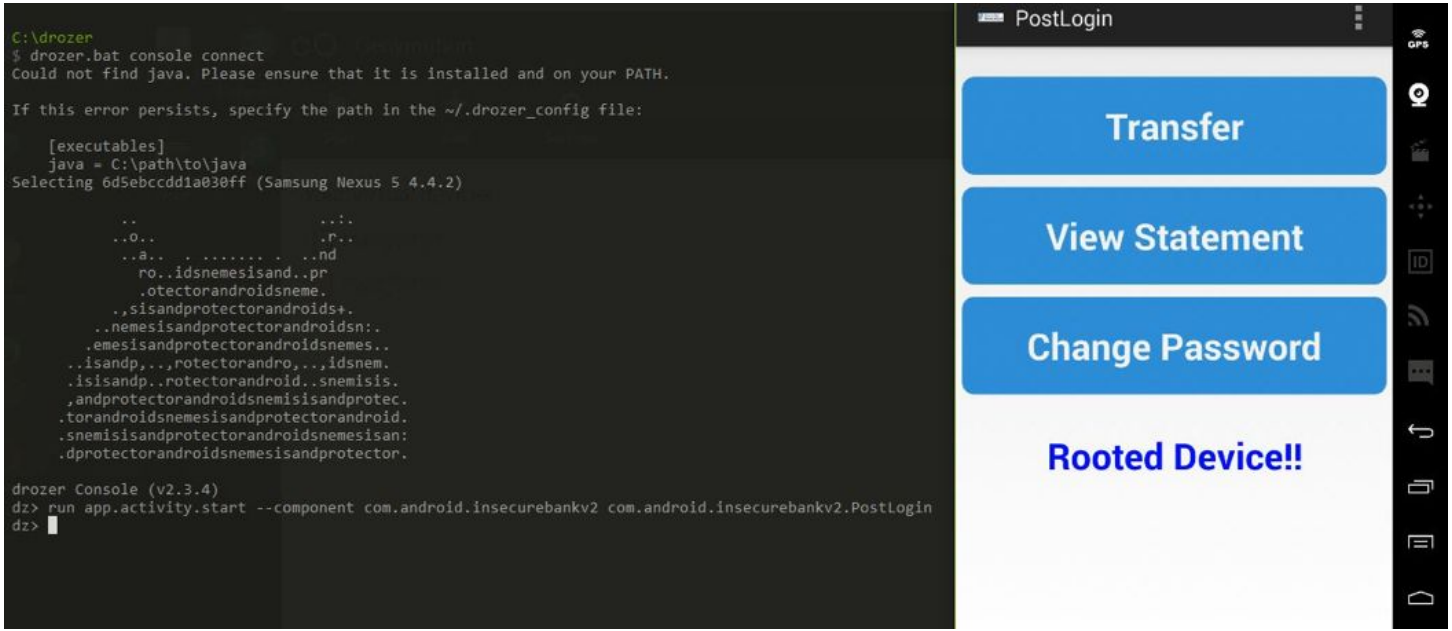
Mobil Sızma Testi Araçları

Drozer

```
C:\drozer
$ drozer.bat exploit list
Could not find java. Please ensure that it is installed and on your PATH.

If this error persists, specify the path in the ~/.drozer_config file:

    [executables]
    java = C:\path\to\java
exploit.remote.browser.addjavascriptinterface   WebView addJavascriptInterface Remote Code Execution (CVE-2012-6636)

exploit.remote.browser.knoxsmdm                 Abuse the New enrolment/UniversalMDMApplication application in Samsung Knox suite to

                                                install rogue drozer agent

exploit.remote.browser.normalize                Webkit Node Normalize (CVE-2010-1759)

exploit.remote.browser.useafterfree             Webkit Use After Free Exploit (Black Hat 2010)

exploit.remote.dos.remotewipe_browserdelivery   Invoke a USSD code that performs a remote wipe on Samsung Galaxy SIII (Ekoparty 2012)

exploit.remote.fileformat.polarisviewerbof_browserdelivery
                                                Deliver Polaris Viewer 4 exploit files over browser (Mobile Pwn2Own 2012)

exploit.remote.fileformat.polarisviewerbof_generate
                                                Generate Polaris Viewer 4 exploit DOCX (Mobile Pwn2Own 2012)

exploit.remote.socialengineering.unknownsources
                                                Deliver the Rogue drozer Agent over browser and hold thumbs the user will install it

exploit.usb.socialengineering.usbdebugging      Install a Rogue drozer Agent on a connected device that has USB debugging enabled
```

Mobil Sızma Testi Uygulamaları

Activity Bypass / InsecureBankv2 Login

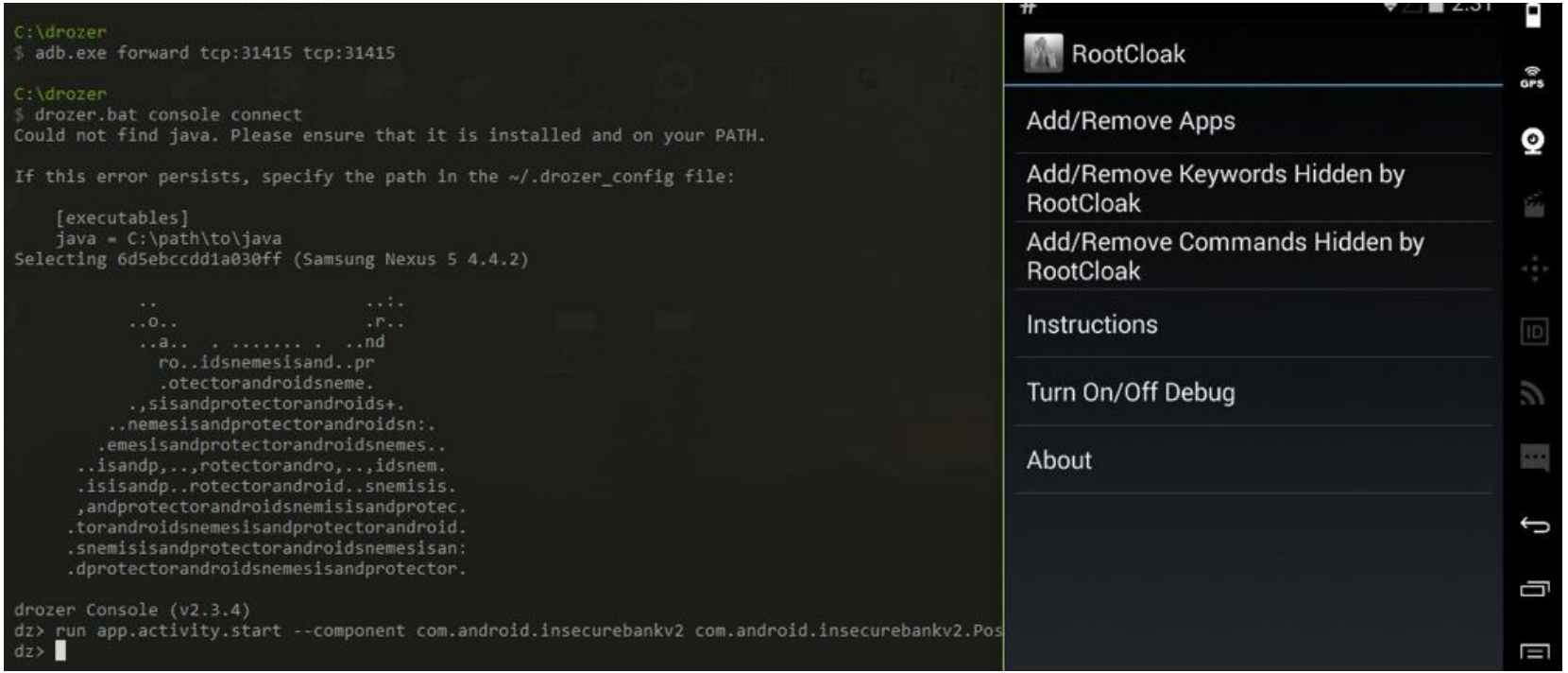


run app.activity.start –component com.android.insecurebankv2 com.android.insecurebankv2.PostLogin

adb shell am start -n com.android.insecurebankv2/com.android.insecurebankv2.PostLogin

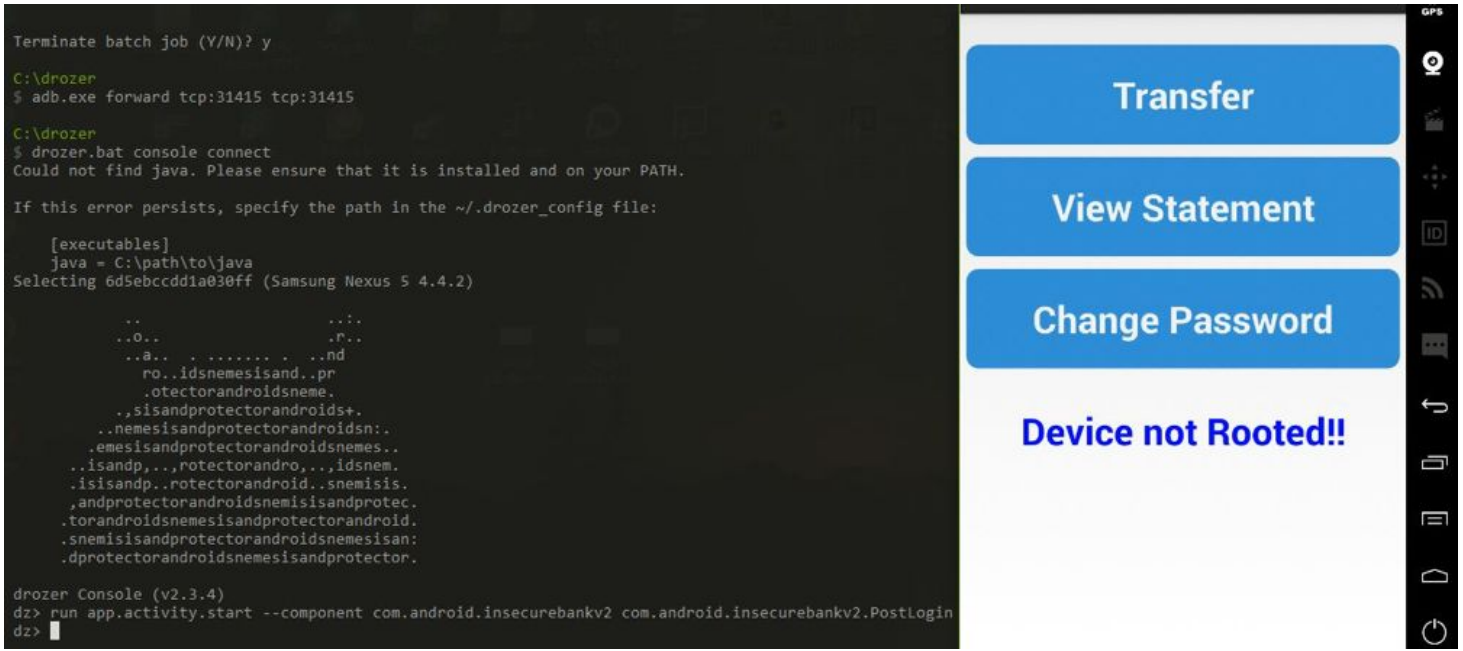
Mobil Sızma Testi Uygulamaları

Root Detection Bypass



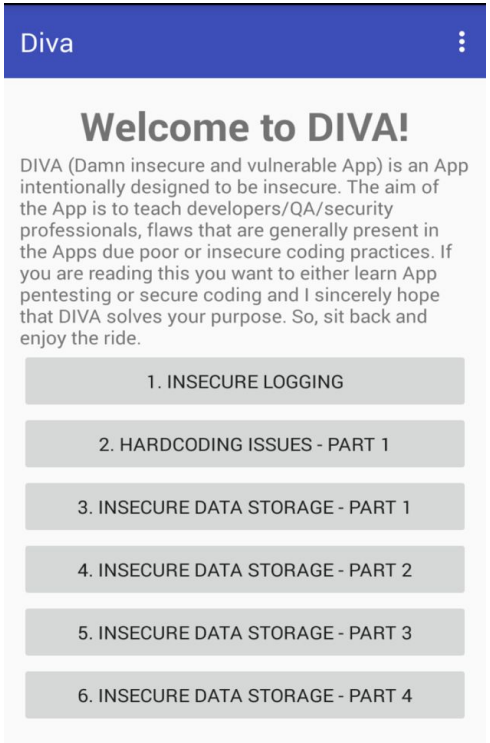
Mobil Sızma Testi Uygulamaları

Root Detection Bypass



Mobil Sızma Testi Uygulamaları

DIVA (Damn Insecure and Vulnerable App)



Mobil Sızma Testi Uygulamaları

DIVA Insecure Logging

diva-beta.apk › AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:"http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0" package="jakhar.aseem.diva" platformBuildVersionCode="23" platformBuildVersionName="6.0-2166767">
    <uses-sdk android:minSdkVersion="15" android:targetSdkVersion="23" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application android:theme="@style/AppTheme" android:label="@string/app_name" android:icon="@mipmap/ic_launcher" android:debuggable="true" android:allowBackup="true" android:supportsRtl="true">
        <activity android:theme="@style/AppTheme_NoActionBar" android:label="@string/app_name" android:name="jakhar.aseem.diva.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:label="@string/d1" android:name="jakhar.aseem.diva.LogActivity" />
        <activity android:label="@string/d2" android:name="jakhar.aseem.diva.HardcodeActivity" />
        <activity android:label="@string/d3" android:name="jakhar.aseem.diva.InsecureDataStorage1Activity" />
        <activity android:label="@string/d4" android:name="jakhar.aseem.diva.InsecureDataStorage2Activity" />
        <activity android:label="@string/d5" android:name="jakhar.aseem.diva.InsecureDataStorage3Activity" />
        <activity android:label="@string/d6" android:name="jakhar.aseem.diva.InsecureDataStorage4Activity" />
        <activity android:label="@string/d7" android:name="jakhar.aseem.diva.SQLInjectionActivity" />
        <activity android:label="@string/d8" android:name="jakhar.aseem.diva.InputValidation2URISchemeActivity" />
        <activity android:label="@string/d9" android:name="jakhar.aseem.diva.AccessControl1Activity" />
        <activity android:label="@string/apic_label" android:name="jakhar.aseem.diva.APICredsActivity">
            <intent-filter>
                <action android:name="jakhar.aseem.diva.action.VIEW_CREDS" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:label="@string/d10" android:name="jakhar.aseem.diva.AccessControl2Activity" />
        <activity android:label="@string/apic2_label" android:name="jakhar.aseem.diva.APICreds2Activity">
            <intent-filter>
                <action android:name="jakhar.aseem.diva.action.VIEW_CREDS2" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <provider android:name="jakhar.aseem.diva.NotesProvider" android:enabled="true" android:exported="true" android:authorities="jakhar.aseem.diva.provider.notesprovider" />
        <activity android:label="@string/d11" android:name="jakhar.aseem.diva.AccessControl3Activity" />
        <activity android:label="@string/d12" android:name="jakhar.aseem.diva.Hardcode2Activity" />
        <activity android:label="@string/pnotes" android:name="jakhar.aseem.diva.AccessControl3NotesActivity" />
        <activity android:label="@string/d13" android:name="jakhar.aseem.diva.InputValidation3Activity" />
    </application>
</manifest>
```

Mobil Sızma Testi Uygulamaları

DIVA Insecure Logging

Dıva uygulamasının ilk örneği güvensiz log kayıtlarından kaynaklanan güvenlik açığıdır. Kullanıcıdan inputta kredi kartı bilgisinin girilmesi isteniyor.

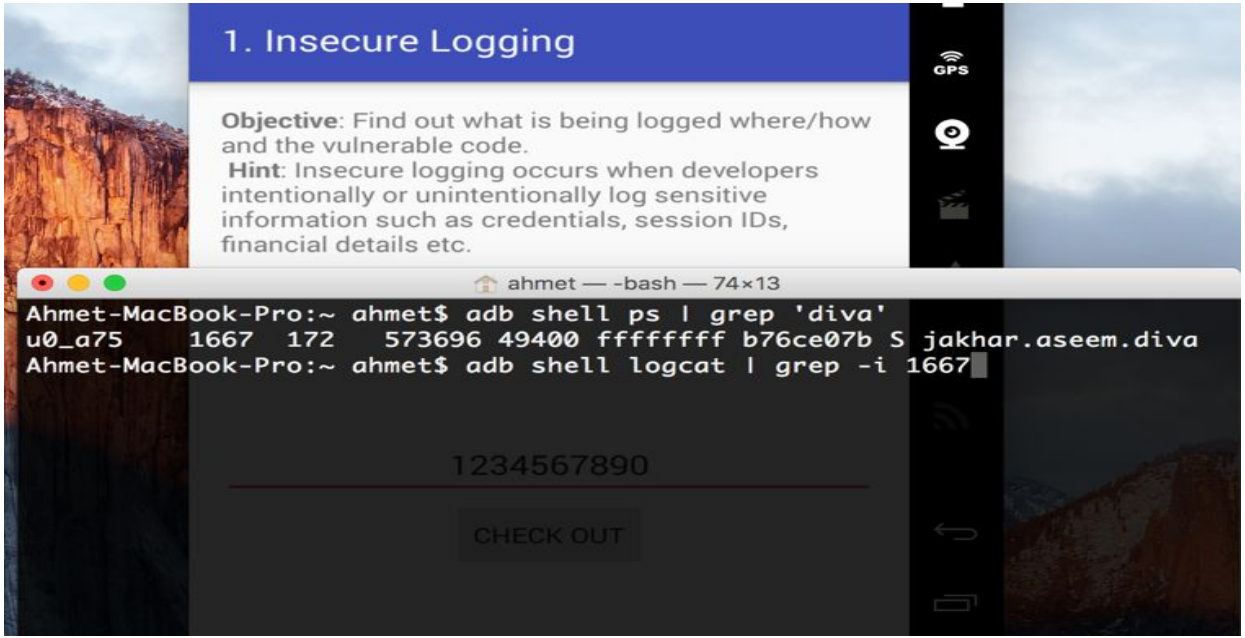
Fakat kaynak kodda bu işlem yapılarken loglama açık bırakılmış yani yazılımın loglarına düşmekte.

Bu logları cihaza ad bile bağlantı kurup logcat ile incelenebilmektedir.
adb shell ps | grep 'diva' diyerek cihazda çalışan süreçlerden diva uygulamanının pid numarasına ulaşabiliyoruz.

Daha sonra **adb shell logcat | grep -i pid** ile logları görüntüleyebilmekteyiz.

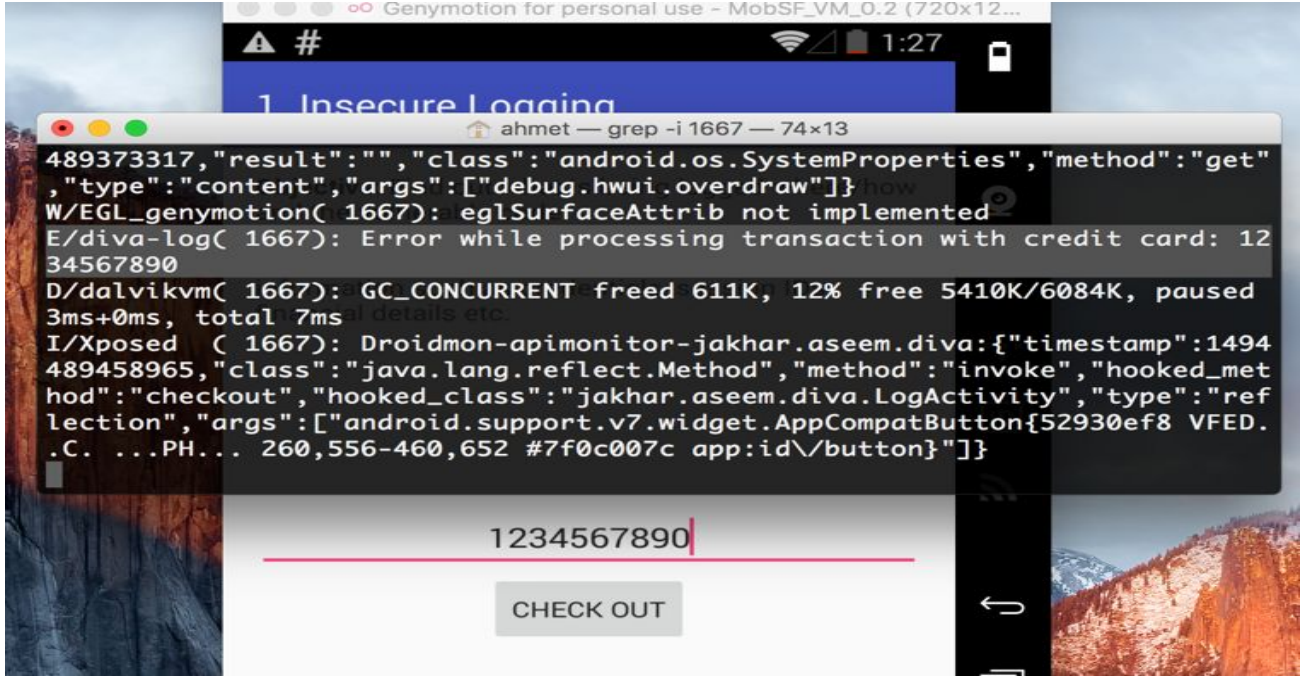
Mobil Sızma Testi Uygulamaları

DIVA Insecure Logging



Mobil Sızma Testi Uygulamaları

DIVA Insecure Logging



Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues - Part 1

Bu aşamada ise yazılımcı tarafından kaynak kodda kullanılan sabitlerden kaynaklanan sorunlara değinilmiş.

Input var ve doğru değeri girdiğimizde başarılı girmediğimizde yeniden deneyin tarzı bir mesaj veriyor.

Bu kodlanırken en temel sevıyede mesaja eşitse true değilse false mantığı kullanılmış ve eşitse diye kontrol edilirken cleartext olarak doğru değer kaynak kodda bulunmakta.

Elimizdeki apk dosyasını decompiler ettiğimizde değere ulaşabilmekteyiz.

Mobil Sızma Testi Uygulamaları

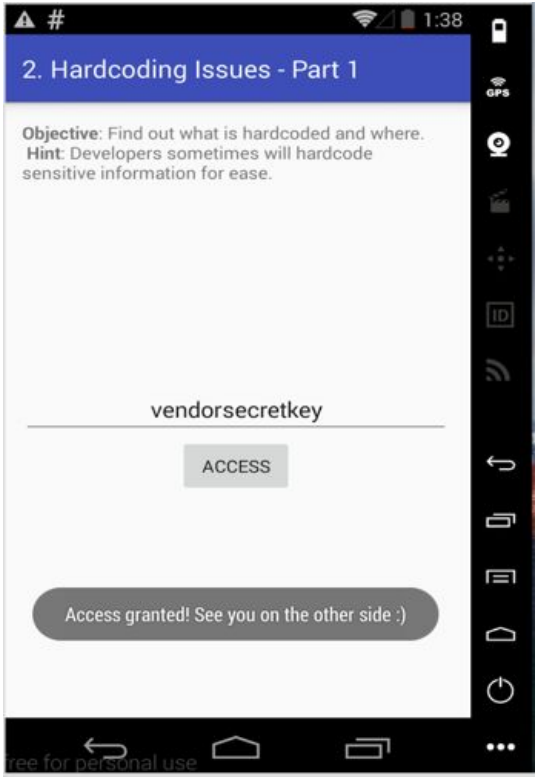
DIVA Hardcoding Issues - Part 1

</> diva-beta.apk > jakhar > aseem > diva > HardcodeActivity.java

```
package jakhar.aseem.diva;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class HardcodeActivity extends AppCompatActivity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView((int) C0200R.layout.activity_hardcode);
    }

    public void access(View view) {
        if (((EditText) findViewById(C0200R.id.hcKey)).getText().toString().equals("vendorsecretkey")) {
            Toast.makeText(this, "Access granted! See you on the other side :)", 0).show();
        } else {
            Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
        }
    }
}
```

Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues - Part 1



Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues - Part 1

Bu bölümde güvensiz veri saklamadan kaynaklanan güvenlik açıklıklarına değinilmiştir.

Açılan activityde önümüze gelen inputlara username,password girmemizi istemekte girdiğimiz bu değerleri güvenli saklanamamasından dolayı saldırganlar tarafından erişilebilir durumdadır.

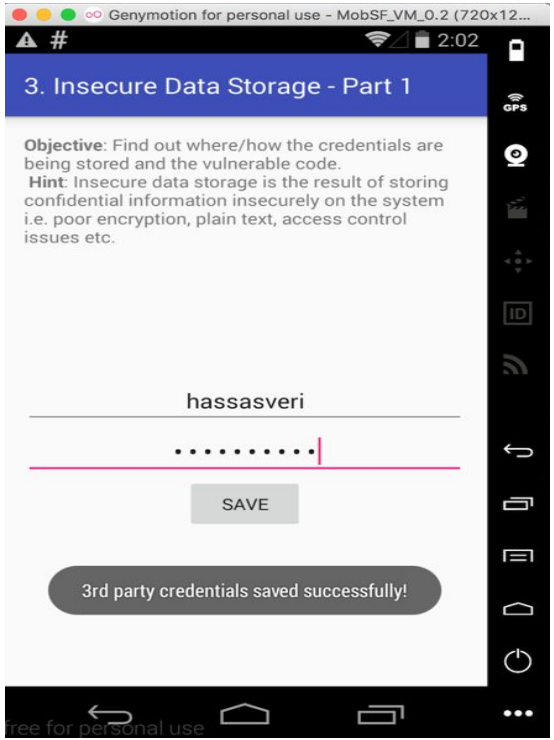
Uygulama /data/data dizinin altında genellikle kendi dizinini oluşturur.

Bu klasörün altında sharedpreferences ve databases klasörleri bulunabilir.

Eğer bunlara hassas veri yazılıyor ise bunlar kontrol edilmelidir.

Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues - Part 1



Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues - Part 1

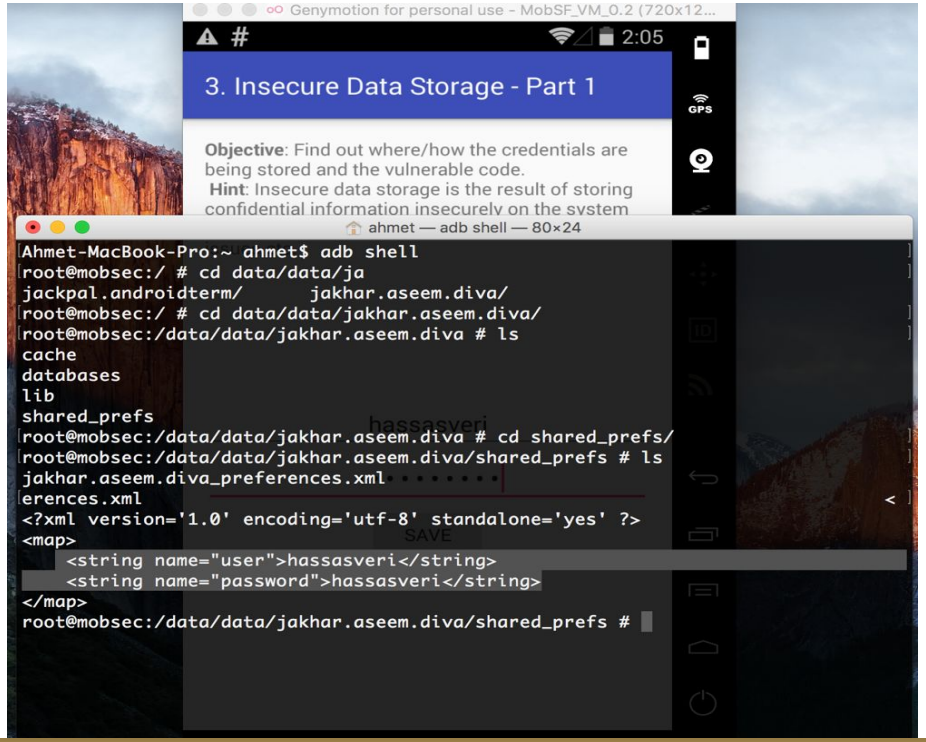
Yukarıdaki resimde gördüğünüz gibi hassasveri adında username ve password girdik.

Girdiğimiz bu verileri yazılımcı **/data/data/jakhar.aseem.diva/shared_prefs/jakhar.aseem.diva_prefences.xml** dosyasında tutmaktadır.

Adb ile cihazda shell alarak cihaz üzerinde bu dosyaya giderek görüntülediğimizde girdiğimiz username ve passworda ulaşmaktayız.

Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 1



Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage- Part 2

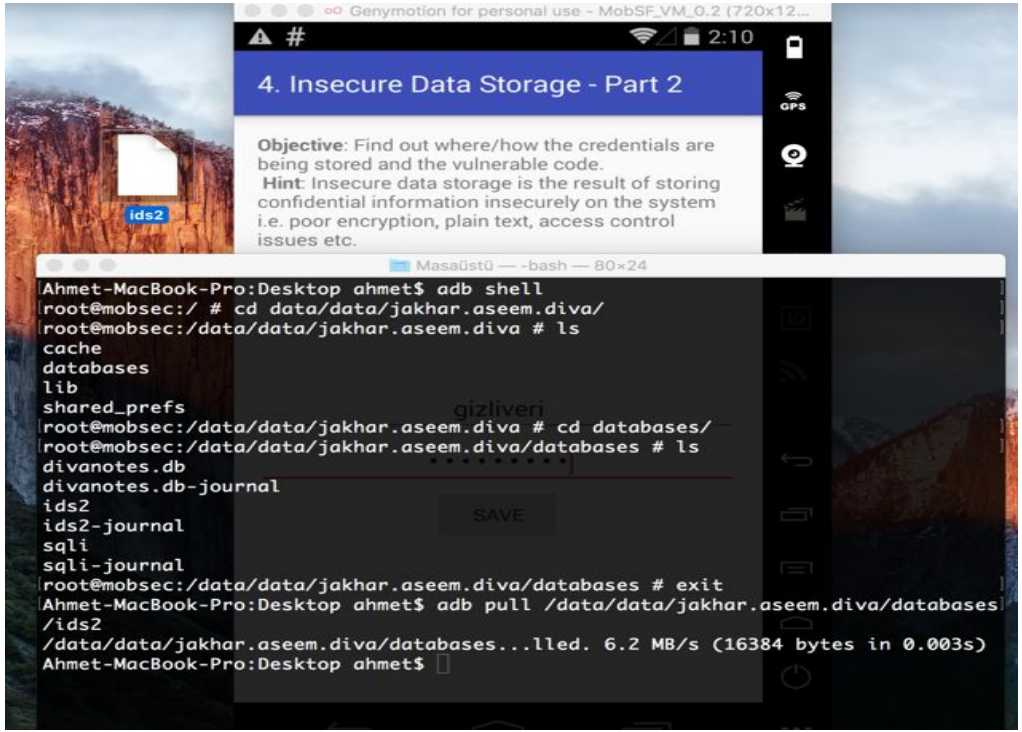
Bu bölümdede güvensiz veri saklama yöntemlerinden kaynaklanan zafiyet bulunmaktadır.

Bu sefer girdiğimiz bilgiler veritabanına kaydolmaktadır.

Fakat kaydolduğu yer cihazın içinde **/data/data/jakhar.aseem.diva/databases** dizininin altındadır.

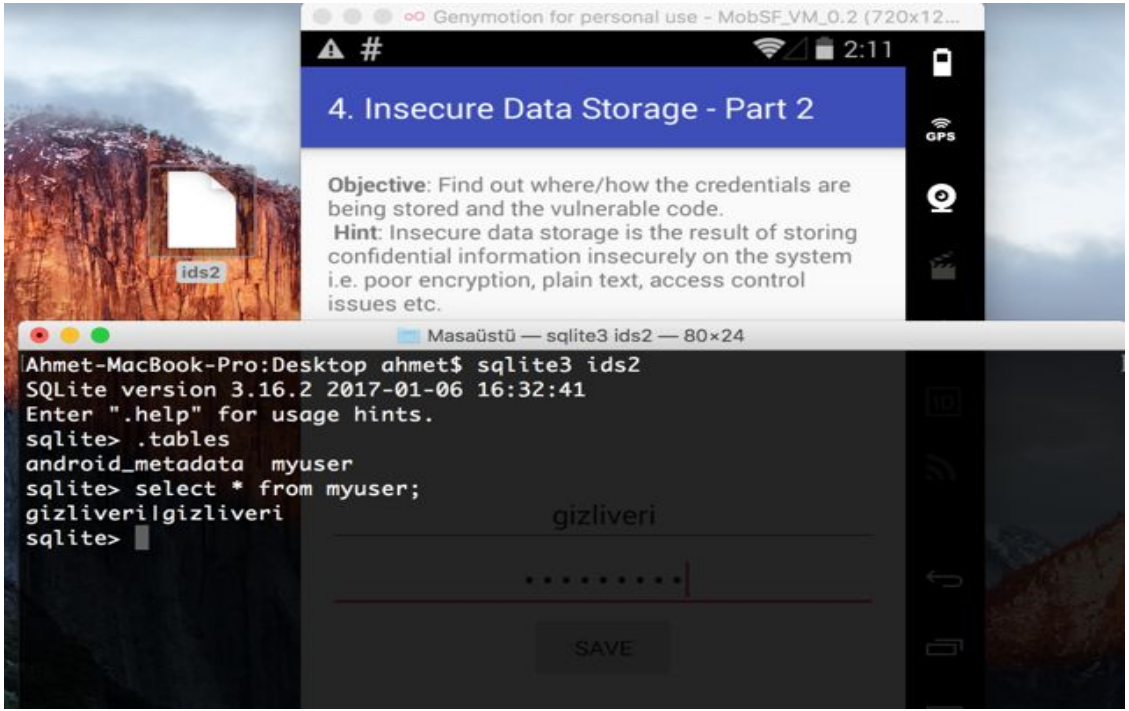
Mobil Sızma Testi Uygulamaları

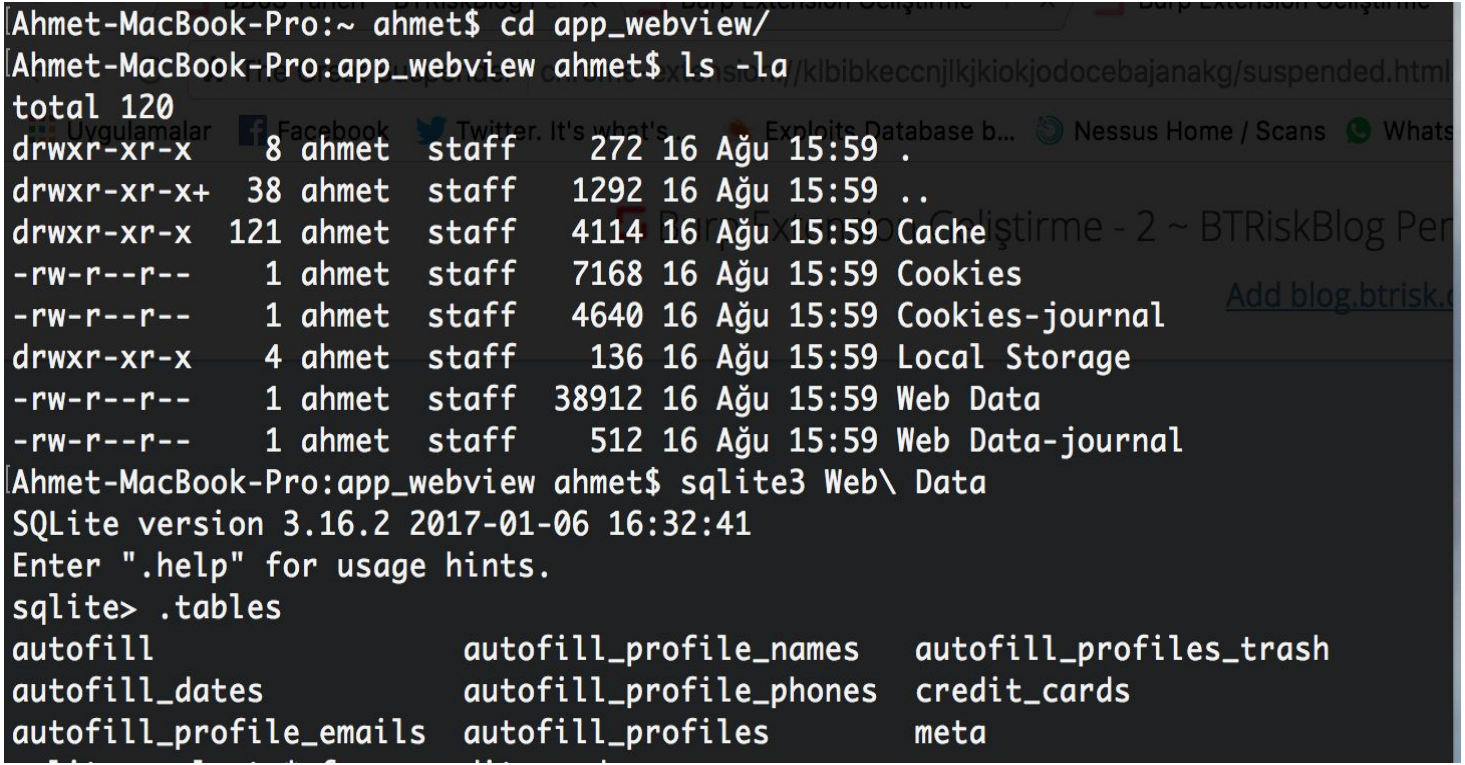
DIVA Insecure Data Storage- Part 2



Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 2




Ahmet-MacBook-Pro:~ ahmet$ cd app_webview/
Ahmet-MacBook-Pro:app_webview ahmet$ ls -la
total 120
drwxr-xr-x   8 ahmet staff   272 16 Ağu 15:59 .
drwxr-xr-x+ 38 ahmet staff   1292 16 Ağu 15:59 ..
drwxr-xr-x 121 ahmet staff   4114 16 Ağu 15:59 Cache
-rw-r--r--   1 ahmet staff   7168 16 Ağu 15:59 Cookies
-rw-r--r--   1 ahmet staff   4640 16 Ağu 15:59 Cookies-journal
drwxr-xr-x   4 ahmet staff   136 16 Ağu 15:59 Local Storage
-rw-r--r--   1 ahmet staff 38912 16 Ağu 15:59 Web Data
-rw-r--r--   1 ahmet staff   512 16 Ağu 15:59 Web Data-journal
Ahmet-MacBook-Pro:app_webview ahmet$ sqlite3 Web\ Data
SQLite version 3.16.2 2017-01-06 16:32:41
Enter ".help" for usage hints.
sqlite> .tables
autofill   autofill_profile_names   autofill_profiles_trash
autofill_dates   autofill_profile_phones credit_cards
autofill_profile_emails autofill_profiles   meta

Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 3

Bu kısımdada yine güvensiz veri saklama sorunlarından birine değinilmiş.

Uygulama bulunduğu klasöre gecici bir dosya oluşturarak hassas verileri buraya kaydetmekte.

Bu tip tespitler için her zaman uygulamanın klasörü,databases ve shared prefences dizinleri incelenmelidir.

Bunun dışında apk dosyası decompiler edilerek kaynak kodu incelenmeli ve mutla Android Manifest.xml dosyasındaki izinler ve diğer bilgiler analiz edilmelidir.

Mobil Sızma Testi Uygulamaları

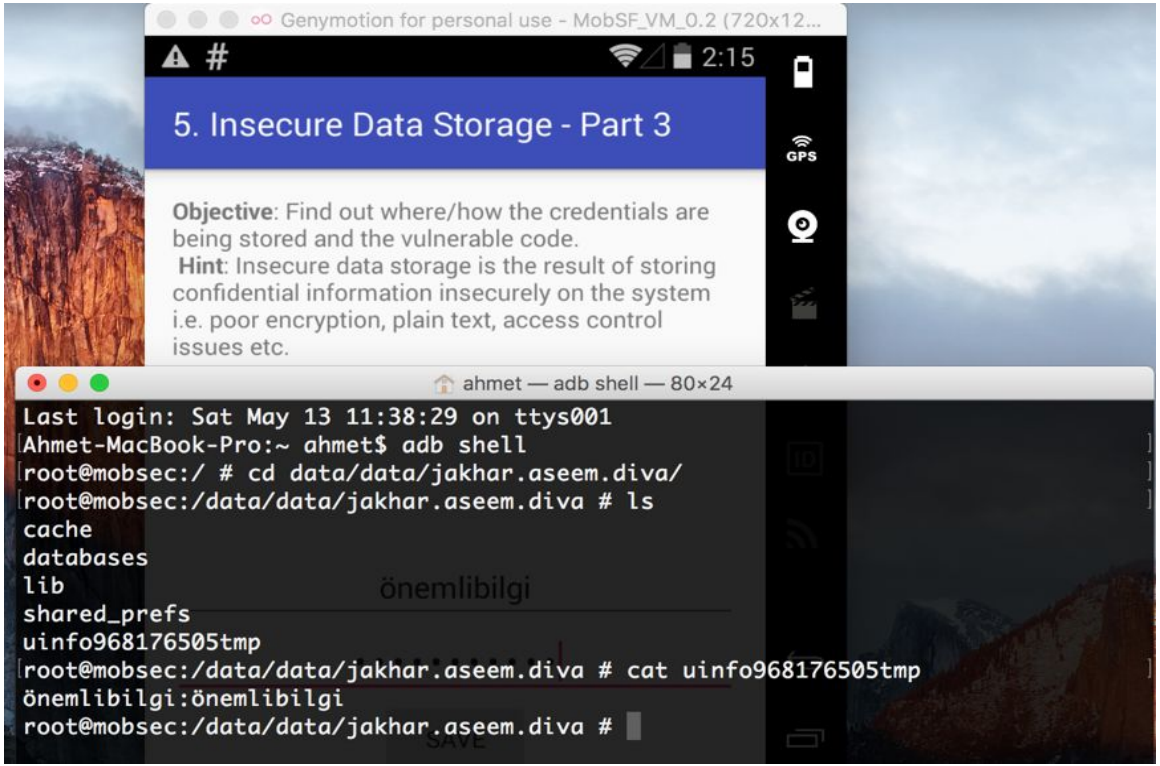
DIVA Insecure Data Storage- Part 3

```
public void saveCredentials(View paramView)
{
  EditText localEditText1 = (EditText)findViewById(2131493006);
  EditText localEditText2 = (EditText)findViewById(2131493007);
  File localFile1 = new File(getApplicationInfo().dataDir);
  try
  {
    File localFile2 = File.createTempFile("uinfo", "tmp", localFile1);
    localFile2.setReadable(true);
    localFile2.setWritable(true);
    FileWriter localFileWriter = new FileWriter(localFile2);
    localFileWriter.write(localEditText1.getText().toString() + ":" + localEditText2.getText().toString() + "\n");
    localFileWriter.close();
    Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
    return;
  }
```

Yukarıdaki resimde gördüğünüz kodda uinfo adında gecici bir dosya oluşturmakta ve alınan değerler buna yazılmaktadır.

Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 3



Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 4

Bu kısımdada yine girilen bilgiler cihaz içinde güvensiz şekilde tutulmaktadır.

Verilerimizi girip save diyoruz. Bu sefer sdcard'da bir dosya oluşturup ona kaydedilmektedir.

Mobil Sızma Testi Uygulamaları

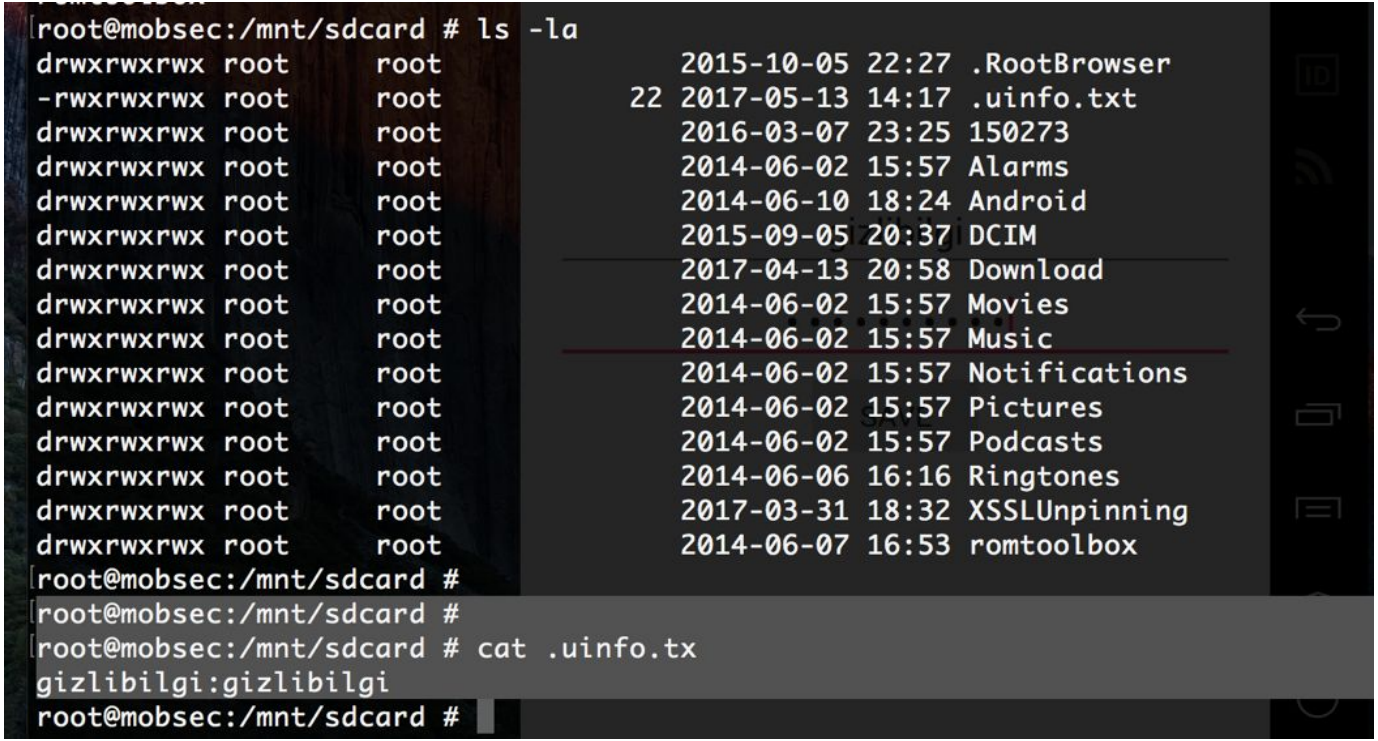
DIVA Insecure Data Storage - Part 4

diva-beta.apk > jakhar > aseem > diva > InsecureDataStorage4Activity.java

```
package jakhar.aseem.diva;

import android.os.Bundle;
import android.os.Environment;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import java.io.File;
import java.io.FileWriter;

public class InsecureDataStorage4Activity extends AppCompatActivity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView((int) C0200R.layout.activity_insecure_data_storage4);
    }

    public void saveCredentials(View view) {
        EditText usr = (EditText) findViewById(C0200R.id.ids4Usr);
        EditText pwd = (EditText) findViewById(C0200R.id.ids4Pwd);
        try {
            File uinfo = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/.uinfo.txt");
            uinfo.setReadable(true);
            uinfo.setWritable(true);
            FileWriter fw = new FileWriter(uinfo);
            fw.write(usr.getText().toString() + ":" + pwd.getText().toString() + "\n");
            fw.close();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (Exception e) {
            Toast.makeText(this, "File error occurred", 0).show();
            Log.d("Diva", "File error: " + e.getMessage());
        }
    }
}
```

Mobil Sızma Testi Uygulamaları

DIVA Insecure Data Storage - Part 4



Mobil Sızma Testi Uygulamaları

DIVA Input Validation Issues – Part 1

Bu kısımda input kontrol eksikliğinden kaynaklanan zafiyetlere değinilmiştir.

Buradaki input girilen değeri aramaktadır.

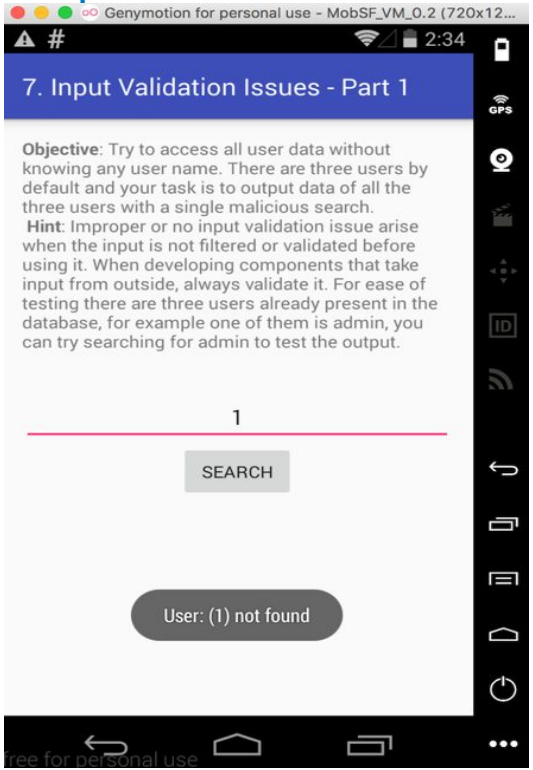
Varsa ekrana getirmekte yoksa bulunamadı olarak çıktı vermekte.

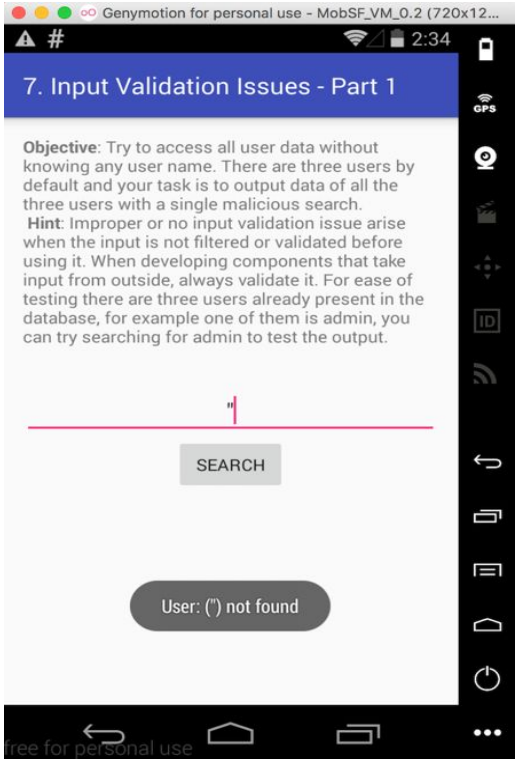
Sql injection denemesi yapmak için “ tırnak kullanıyoruz fakat bize user bulunamadı olarak çıktı vermekte.

‘ tırnak denediğimizde ise her hangi bir user bulunamadı çıktısı vermemekte boş döndürmektedir.

Mobil Sızma Testi Uygulamaları

DIVA Input Validation Issues – Part 1





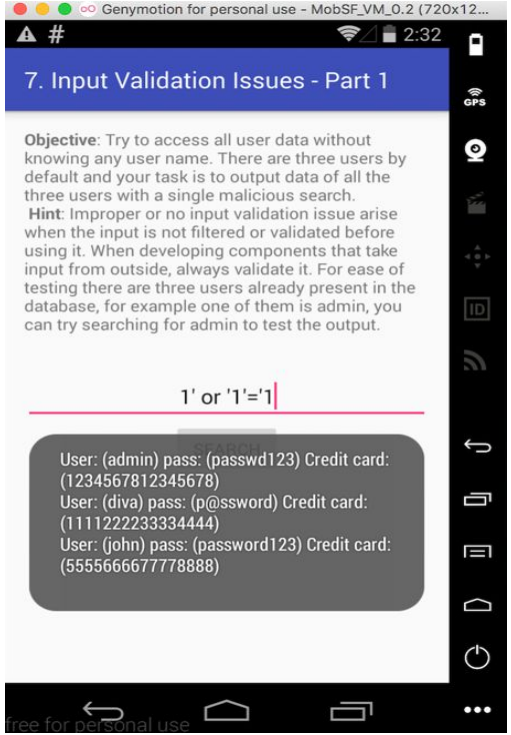
Mobil Sızma Testi Uygulamaları

DIVA Input Validation Issues – Part 1

```
public void search(View paramView)
{
  EditText localEditText = (EditText)findViewById(2131493017);
  try
  {
    Cursor localCursor = this.mDB.rawQuery("SELECT * FROM sqliuser WHERE user = '" + localEditText.getText().toString() + "'", null);
    StringBuilder localStringBuilder = new StringBuilder("");
    if ((localCursor != null) && (localCursor.getCount() > 0))
    {
      localCursor.moveToFirst();
      do
        localStringBuilder.append("User: (" + localCursor.getString(0) + ") pass: (" + localCursor.getString(1) + ") Credit card: (" +
      while (localCursor.moveToNext());
    }
    while (true)
    {
      Toast.makeText(this, localStringBuilder.toString(), 0).show();
      return;
      localStringBuilder.append("User: (" + localEditText.getText().toString() + ") not found");
    }
  }
```

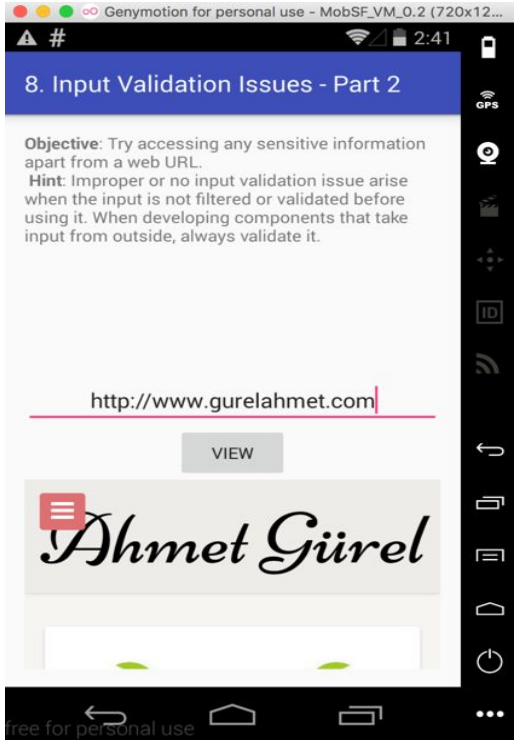
Mobil Sızma Testi Uygulamaları

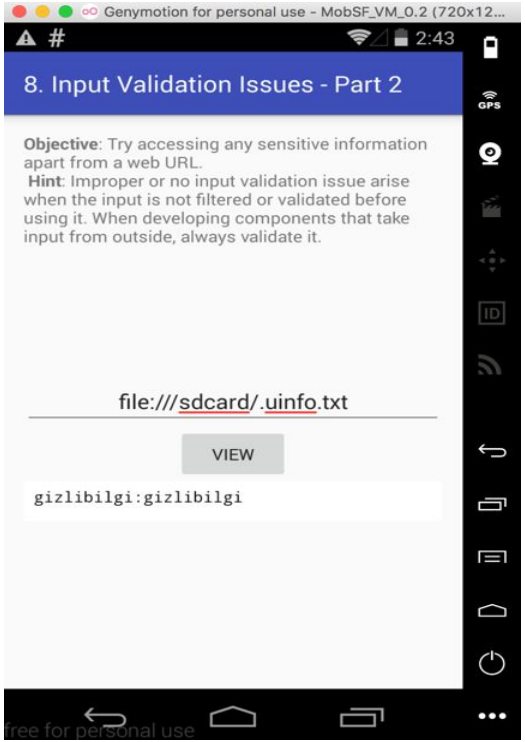
DIVA Input Validation Issues – Part 1



Mobil Sızma Testi Uygulamaları

DIVA Input Validation Issues – Part 2





Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 1

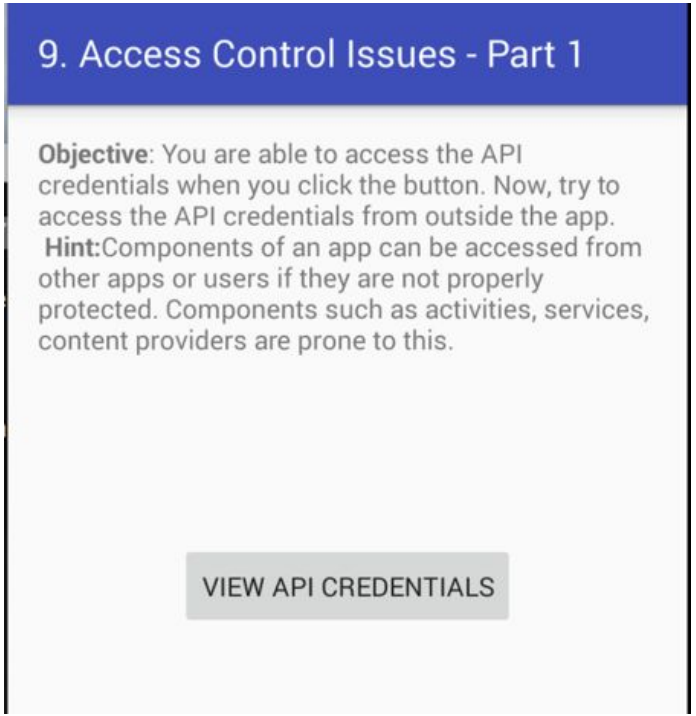
Erişim control sorunları başlıklı bu bölümde activitylerin AndroidManifest.xml dosyasında gerekli şekilde konfigurasyonu ve izinleri ayarlanamadığında activityler dışarıdan butonlara tıklanmadan açılabilmektedir.

Drozer ve adb gibi araçlarla bu işlemler yapılabilmektedir.

Bu kısımda View Apı Credentials a tıkladığımızda yeni bir activity açılarak api bilgilerini getiriyor.
Buraya tıklamadan dışarıdan komut ile activity başlatılabilmektedir.

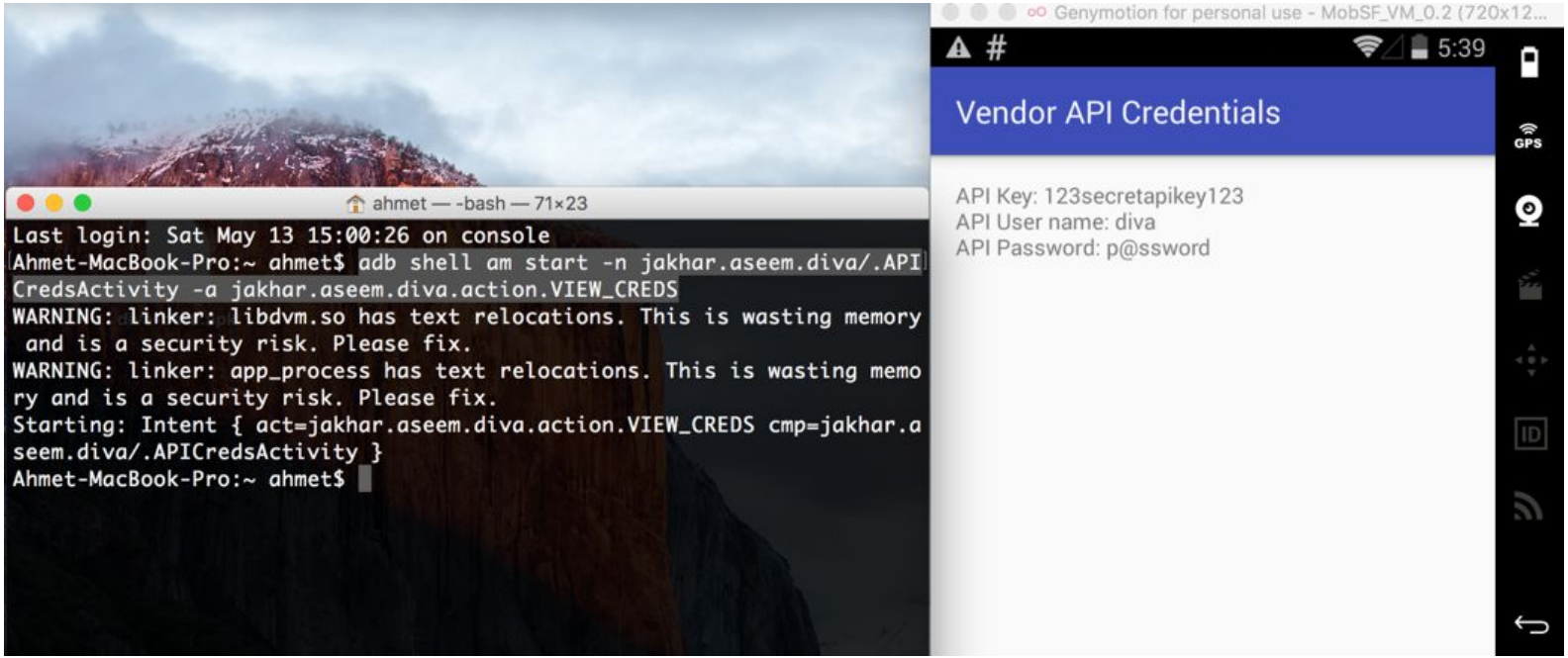
Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 1



Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 1



adb shell am start –n jakhar.aseem.diva/.API CredsActivity –a jakhar.aseem.diva.action.VIEW_CREDS

Mobil Sızma Testi Uygulamaları

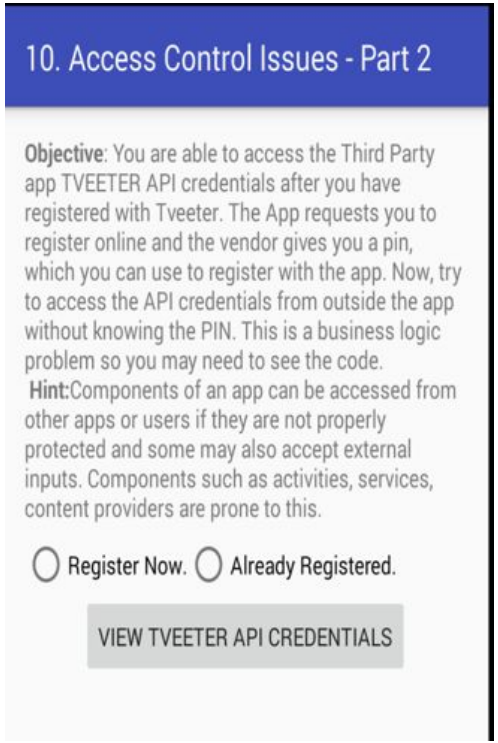
DIVA Acces Control Issues – Part 2

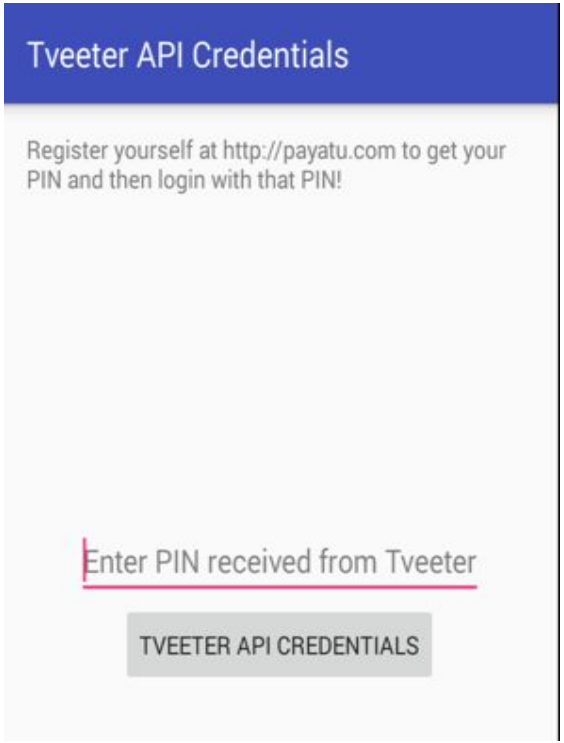
Erişim control sorunlarının ikinci kısmıda illk kısmına benzer sadece bu sefer uygulama açıldığında önümüze iki seçenek suruyor Kayıt ol ve Zaten Kayıtlı Kullanıcıyım tarzında kayıt ol seçeneğine tıklandığı zaman PIN sormakta.

Fakat Zaten kayıtlı kullanıcıyım sekmesinde ise direk activity açılmakta. Bizim amacımız zaten kayıtlı kullanıcıyım butonuna basmadan bu activity'i dışarıdan çalıştırmak.

Mobil Sızma Testi Uygulamaları

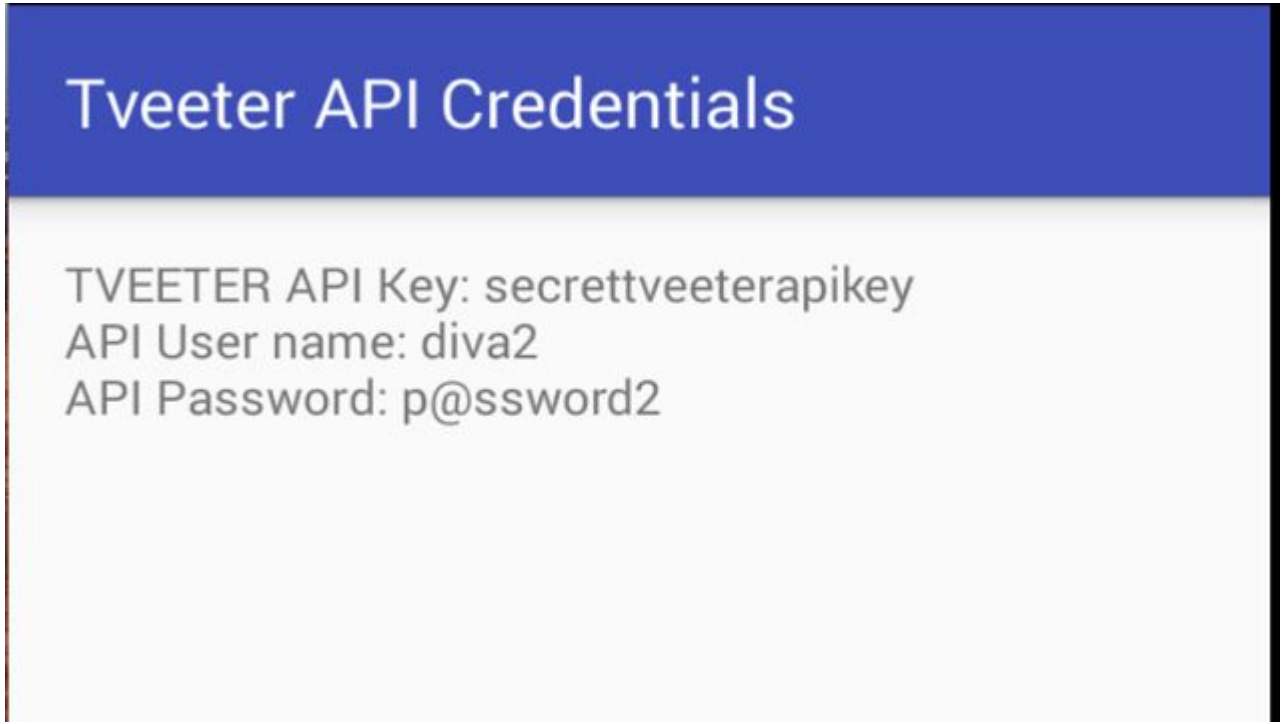
DIVA Acces Control Issues – Part 2





Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 2



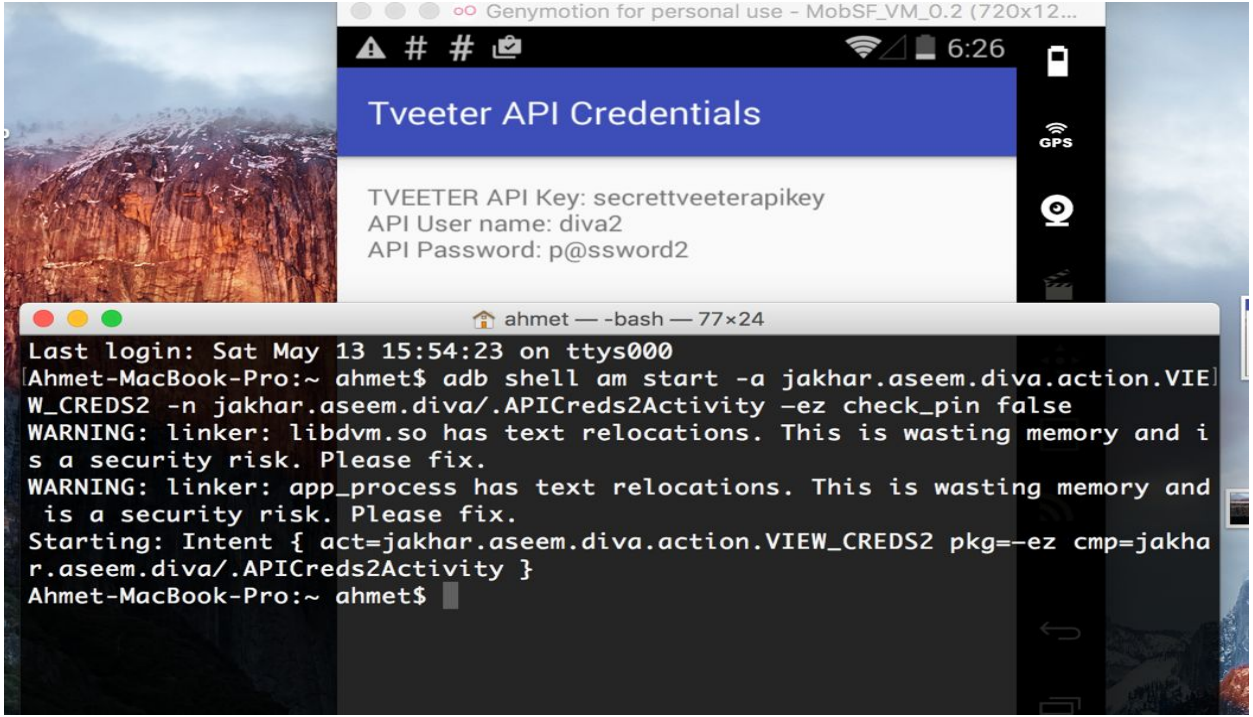
Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 2

Kaynak kod incelendiği zaman check_pin true gibi bir kod satırı görülmekte ve biz bunu adb ile activity'i başlatırken –ez check_pin false parametresini ekleyerek activity'i dışarıdan tetikleyip hiç bir butona basmadan çalıştırılabilmektedir.

Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 2

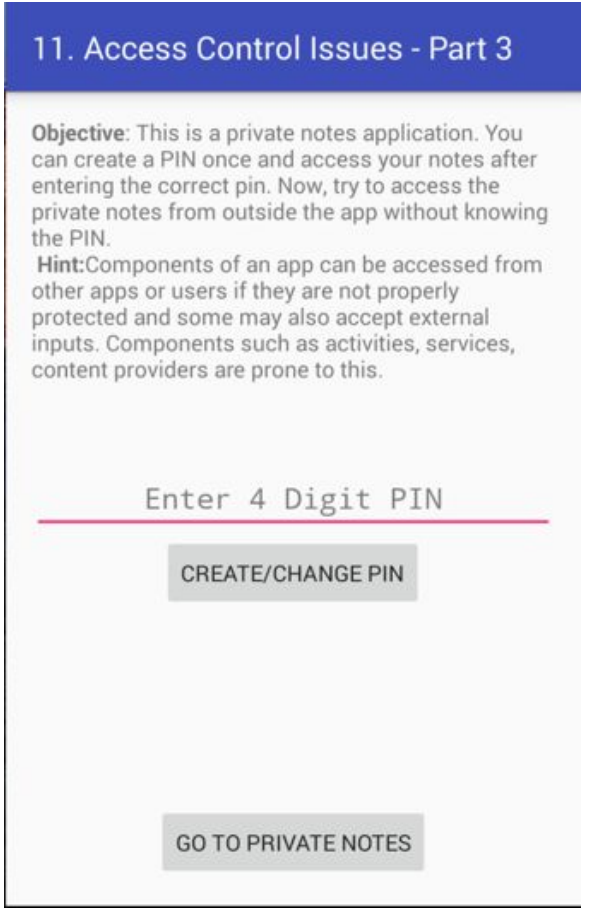


Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 3

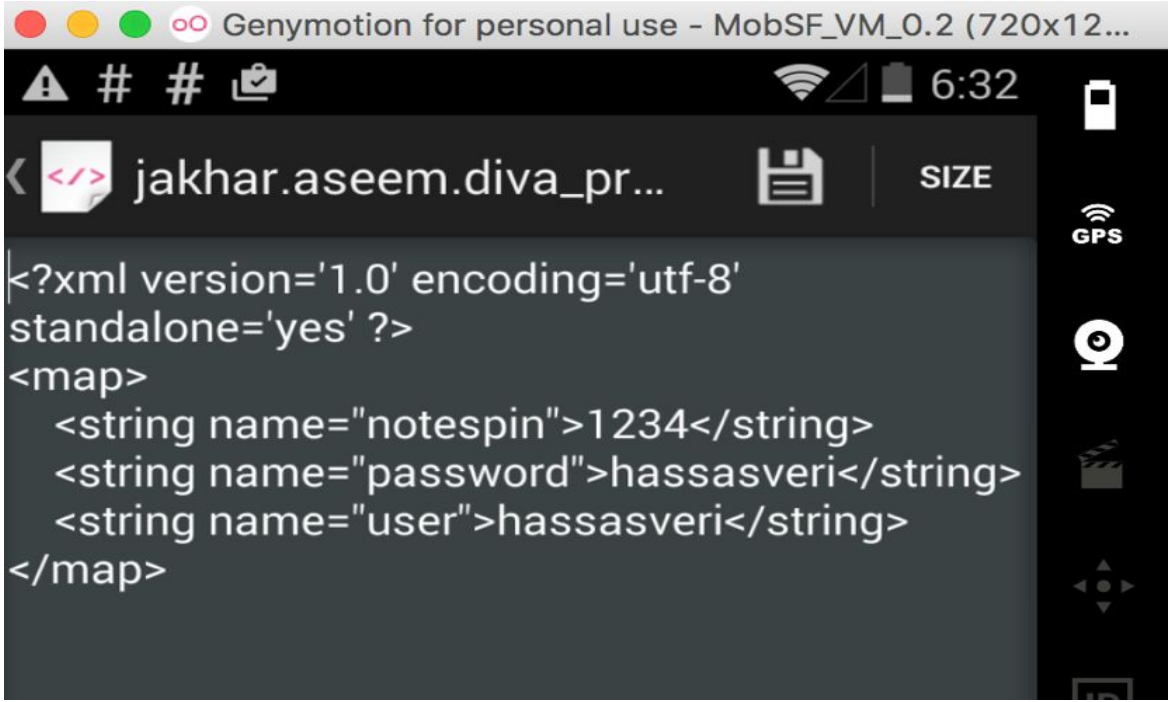
Bu kısımda yine erişim control sorunlarına değinilmiş.
Kullanıcıdan bir pin kodu girmesi istenmektedir.

Girilen PIN kodu sharedpreferences dizinin altında xml dosyasına yazılmaktadır. Buradan PIN koduna ulaşabilmekteyiz.



Mobil Sızma Testi Uygulamaları

DIVA Acces Control Issues – Part 3



Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues – Part 2

Bu aşama daha önce değindiğimiz kaynak kodda bulunan parola ve doğrulama değerlerinden kaynaklanan güvenlik açıklıklarıdır.

Mobil Sızma Testi Uygulamaları

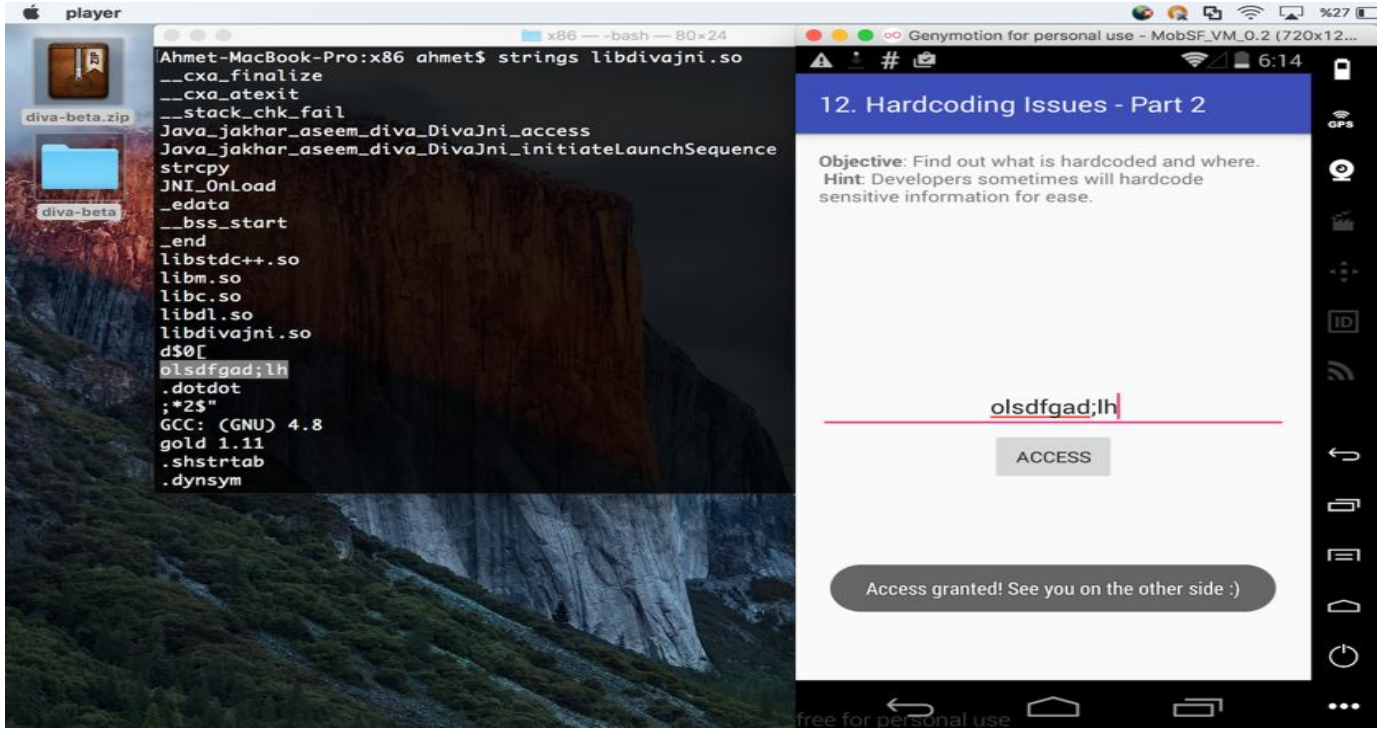
DIVA Hardcoding Issues – Part 2

```
33  #include <jni.h>
34  #include <string.h>
35  #include "divajni.h"
36
37  #define VENDORKEY   "olsdfgad;lh"
38  #define CODE        ".dotdot"
39  #define CODESIZEMAX 20
40  /*
41   * Verify the key for access
42   *
43   * @param  jkey   The key input by user
44   *
45   * @return 1 if jkey is valid, 0 otherwise. In other words
46   @         if the user key matches our key return 1, else return 0.
47   */
48  JNIEXPORT jint JNICALL Java_jakhar_aseem_diva_DivaJni_access(JNIEnv * env, jobject jobj, jstring jkey) {
49
50      const char * key = (*env)->GetStringUTFChars(env, jkey, 0);
51
52      return ((strncmp(VENDORKEY, key, strlen(VENDORKEY)))?0:1);
53  }
54
```

Kaynak kodda bulunan bu key apk decompiler işlemlerinden sonra elde edilebilir durumdadır.

Mobil Sızma Testi Uygulamaları

DIVA Hardcoding Issues – Part 2



Mobil Sızma Testi Uygulamaları

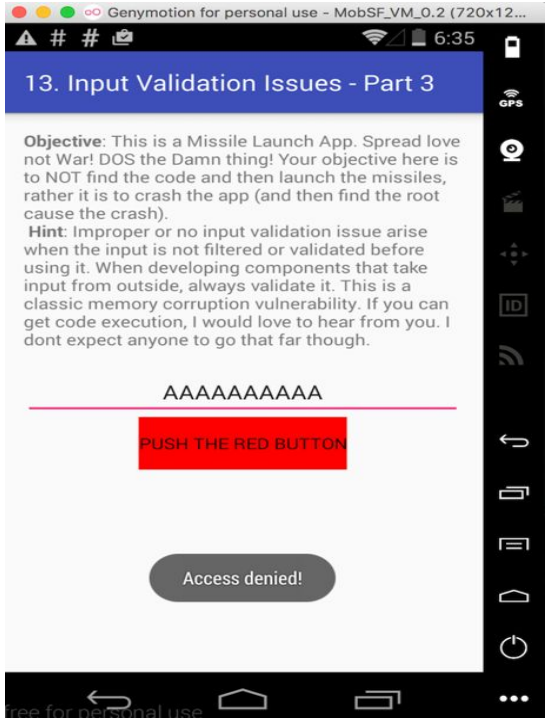
DIVA Input Validation Issues – Part 3

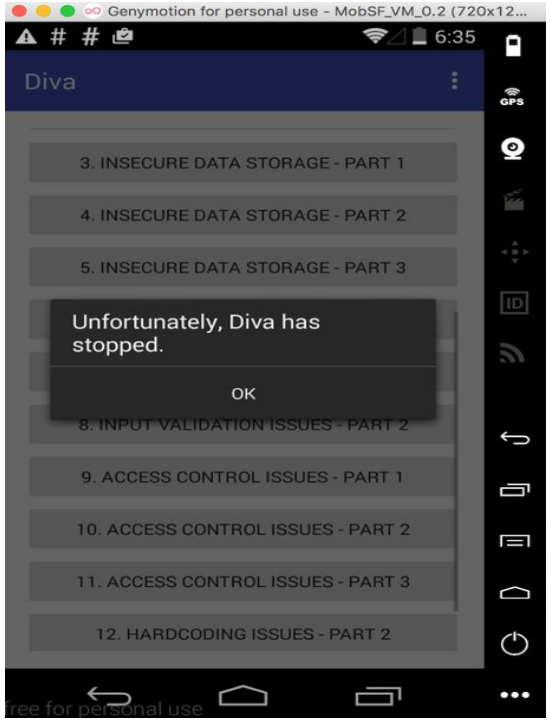
Input kontrol eksikliğinden kaynaklanan zafiyetlerin üçüncü kısmı bu sefer input uzunluğu kontrol edilmemiş.

10 karakter girildiğinde herhangi bir sorun vermemektedir fakat 40 adet karakter girildiğinde uygulama crash olmaktadır.

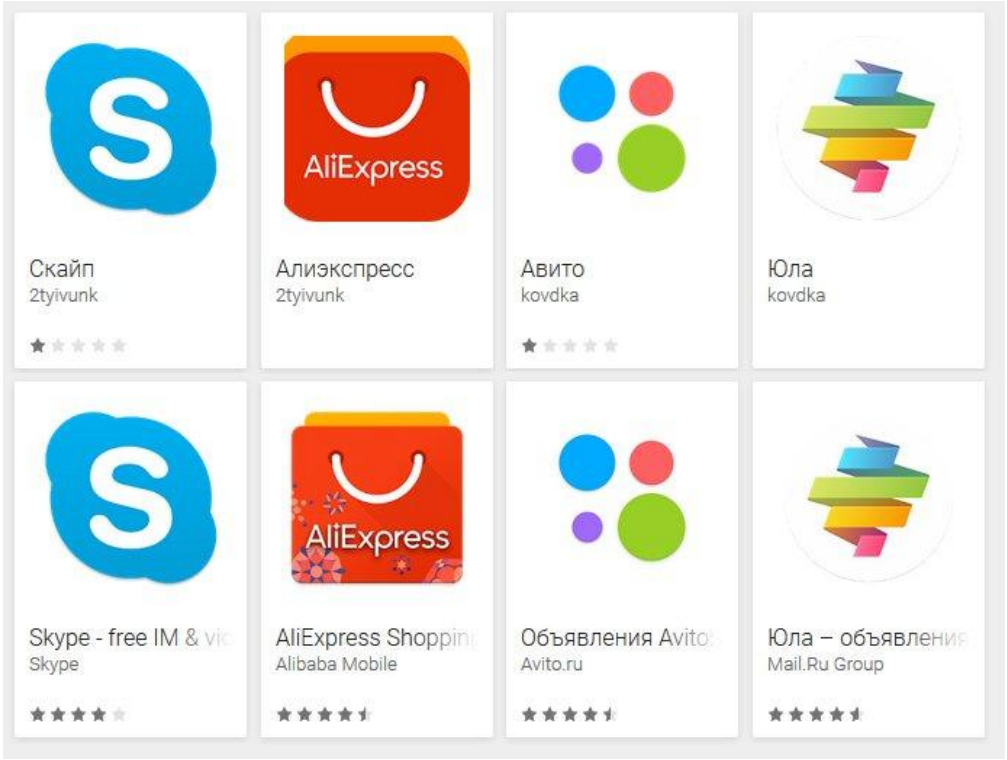
Mobil Sızma Testi Uygulamaları

DIVA Input Validation Issues – Part 3

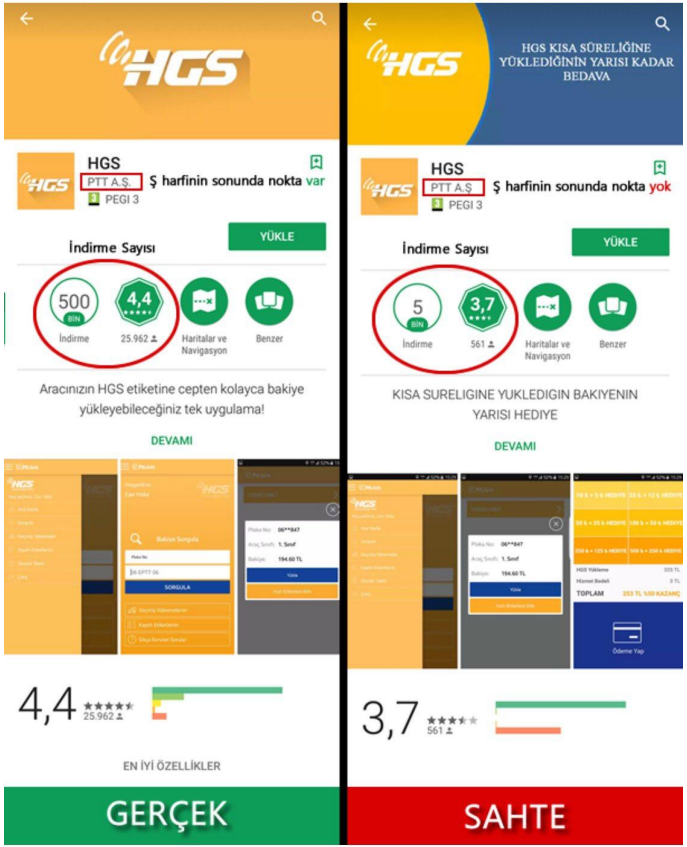




Google Play Uygulamaları Gerçek Olmayabilir !



Google Play Uygulamaları Gerçek Olmayabilir !



Kontrol Listesi (Check List)

1. Network Trafiği
2. SSL Pinning Kontrolu / Bypass
3. Web Servis Kontrolleri
4. Yanlış Kimlik Doğrulaması
5. Oturum Yönetimi Kontrolü
6. IDOR
7. MongoDB/NoSQL Injection
8. Yönetim Panel Login Brute Force
9. Uygulama Komponent Analizi (Drozer)
10. File Verification
11. Uygulama Yetkilendirmeleri ve Kriptografik Kontroller
12. Lokal Veri Analizi
13. Veri Sızıntısı Android Backup
14. Tersine Mühendislik Hardcoded Credential Kontrolü
15. Patching/Modification
16. QRCode Backdoor
17. Subdomain Keşif
18. Email Keşif
19. Github vb. Siteler ile Hassas Veri Kontrolü

info@gurelahmet.com