Android Security & Exploitation

Aditya Gupta (@adi1391)

Founder, Attify (http://attify.com)

adi@attify.com

Certifications : http://securitytube-training.com
Pentester Academy : http://PentesterAcademy.com

Check the complete course

- Tons of more exercises, techniques, hands-on examples and exploitation .

- Check out the **complete training** at

- http://securitytube-training.com/online-courses/android-security-for-pentesters/index.html

- pentesteracademy.com

Damn Insecure and Vulnerable App

- DIVA - Vulnerable Android application
- Created by Aseem Jakhar (@aseemjakhar) of Payatu Technologies
- Good starting point to explore Android Security
- Other vulnerable apps include - GoatDroid, InsecureBank, Intrepidus Learner

Damn Insecure and Vulnerable App

DIVA Android

What is DIVA?

DIVA (Damn insecure and vulnerable App) is an App intentionally designed to be insecure. We are releasing the Android version of Diva. We thought it would be a nice way to start the year by contributing something to the security community. The aim of the App is to teach developers/QA/security professionals, flaws that are generally present in the Apps due poor or insecure coding practices. If you are reading this, you want to either learn App pentesting or secure coding and I sincerely hope that DIVA solves your purpose. So, sit back and enjoy the ride.

Why name it Diva?

No offense to anyone, but I was bored with the name DV* and decided to name it more fancy :)

Who can use Diva?

The idea originated, from a developer's perspective. The Android security training for developers becomes slightly boring with lot of theory and not much hands-on. SO, I created DIVA for our Android developer training. Diva gamifies secure development learning. With that said, it is an excellent learning tool for aspiring Android penetration testers and security professionals as it gives an insight into app vulnerabilities including the source code. To sum it up:

- Android App developers
- Android Penetration testers
- Security professionals

Download **binary** from http://goo.gl/sPvimd

Download **source** from https://github.com/payatu/diva-android

Challenges in DIVA

1. Insecure Logging
2. Hardcoding Issues
3. Insecure Data Storage
4. Input Validation vulnerabilities
5. Access Control Issues

Challenge 1 : InsecureLogging

- Video 9 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**

- Android maintains a centralised logcat

- Typically used by developers for debugging purposes

- Accessible via ADB or to other applications (<4.1)

- ```
  adb shell ps | grep -i 'diva'
  adb shell logcat | grep [pid]
  ```
  ```

Challenge 2 : HardCoding Issues - I

- Video 10 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**

- Reverse the Android application using jadx or dex2jar

- Analyse the application source code

- Find the "Secret key"

Challenge 3 : Insecure Data Storage I

- Video 8 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**
- Check the application local data storage
- Uses Shared Preferences for Data storage
- Used by a lot of developers to store "sensitive data"
- **`/data/data/[package-name]/shared_prefs`**
- Check the xml file if you can find the sensitive information

Challenge 4 : Insecure Data Storage II

- Video 8 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**

- Check the application local data storage

- Uses database for Data storage

- **`/data/data/[package-name]/databases`**

- database with the name **`ids2`**

Challenge 5 : Insecure Data Storage III

- Video 8 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**

- Check the application local data storage

- Uses temporary files for Data storage

- **`/data/data/[package-name]/`**

- New file created with the name **`uinfo-*tmp`**

Challenge 6 : Insecure Data Storage IV

- Video 8 of "Android Security for Pentesters" at **SecurityTube-Training** and **PentesterAcademy**
- Uses external data storage
- `/mnt/sdcard`
- New file created with the name `.uinfo.txt`

Challenge 7 : Input Validation Issues I

- Classical case of SQL Injection vulnerability
- Lack of input validation
- Taking user input and executing the SQL query
- 1' or '1' ='1'-- makes it true always

Challenge 8 : Input Validation Issues 2

- Uses the user input to show the browser content
- Does it allow file:// ?
- If yes, can you read local data storage using this ?
- How about external storage ?

Challenge 9 : Access Control Issues I

- API creds are accessible to the user
- We want to access it from outside the application
- Check out the AndroidManifest.xml file
- Defines an intent filter with the name
  `jakhar.aseem.diva.action.VIEW_CREDS`
- Can use am to invoke this intent-filter
  `adb shell am start -a jakhar.aseem.diva.action.VIEW_CREDS`

Challenge 10 : Access Control Issues 2

- Previous way does not work - brings us to a passcode screen
- Takes another argument chk_pin of the type boolean

```
public void viewAPICredentials(View view) {
    RadioButton rbregnow = (RadioButton) findViewById(R.id.aci2rbregnow);
    Intent i = new Intent();
    boolean chk_pin = rbregnow.isChecked();
    i.setAction("jakhar.aseem.diva.action.VIEW_CREDS2");
    i.putExtra(getString(R.string.chk_pin), chk_pin);
    if (i.resolveActivity(getPackageManager()) != null) {
        startActivity(i);
        return;
    }
    Toast.makeText(this, "Error while getting Tveeter API details", 0).show();
    Log.e("Diva-aci1", "Couldn't resolve the Intent VIEW_CREDS2 to our activity");
}
```

```
super.onCreate(savedInstanceState);
setContentView((int) R.layout.activity_apicreds2);
TextView apicview = (TextView) findViewById(R.id.apic2TextView);
EditText pintext = (EditText) findViewById(R.id.aci2pinText);
Button vbutton = (Button) findViewById(R.id.aci2button);
if (getIntent().getBooleanExtra(getString(R.string.chk_pin), true)) {
    apicview.setText("Register yourself at http://payatu.com to get your PIN and then login with that PIN!");
    pintext.setVisibility(0);
    vbutton.setVisibility(0);
    return;
}
apicview.setText("TVEETER API Key: secrettveeterapikey\nAPI User name: diva2\nAPI Password: p@ssword2");
}
```

Challenge 10 : Access Control Issues 2

- Using Drozer :

```
run app.activity.info -a jakhar.aseem.diva

run app.activity.start --component jakhar.aseem.diva
jakhar.aseem.diva.APICreds2Activity --extra boolean chk_pin
false
```

Challenge 11 : Access Control Issues 3

- Asks us to set a pin for the notes
- Pretty similar to what we did with the "Catch Notes" application in the **Android Security for Pentesters course**
- Uses a content provider
- Will use Drozer to exploit this vulnerability

Challenge 12 : Hardcoding Issues - 2

- JNI is used to perform validation in this case
- Look inside the lib folder
- Pull out the .so file
- Run objdump and readelf in order to understand more about it
- Check out .rodata section (segment to store constant data)

Challenge 12 : Hardcoding Issues - 2

- Read more about the ELF structure here - http://wiki.osdev.org/ELF

- `objdump -s -j .rodata *.so`
  - -s : show the full content
  - -j : specifying the segment name
- `readelf -x .rodata *.so`
  - -x : specifying the section name

Challenge 13 : Input Validation Issues - 3

- Left as a challenge :)

- Check out the **complete training** at
  - securitytube-training.net
  - pentesteracademy.com

- Reach out to me at adi@attify.com or @adi1391