CS678 – Topics in Smartphone Security & Reliability | Project Stage A – MobileAudit

Submitted by – Kush Borikar (kb97)

Problem: Android apps often make insecure traffic requests, are over-privileged, and are implemented insecurely. Because malicious apps can use these vulnerabilities to steal sensitive data, take over devices, and carry out other malicious actions, there is a serious risk to the security and privacy of users.

Overview: Unnecessary/Over-privileged Permissions: Users can manage which apps have access to their data and device features with the help of Android permissions. An app will request permission to access specific features or data when you install it. And a subset of these are Dangerous Permissions - Dangerous permissions are permissions that can give an app access to sensitive data or device features. These permissions are not granted by default, and users will need to explicitly grant them to the app when they install it. Apps requesting access to data or device features unrelated to their main functions are known as unnecessary/over-privileged permission requests.

Unrestricted Exported Components: An Android applications' exported components are those that other apps can access. This covers broadcast receivers, content providers, activities, and services. Features like a photo picker or a file sharing service that are intended to be shared with other apps are usually implemented using exported components. But in some cases, over exported components are parts, such as Activities and Services, that are meant to be used only within the app but are unintentionally left accessible for use by other apps.

Insecure Request Traffic: HTTP requests are the messages that apps send to web servers to request data. The requests can be used to fetch web pages, images, videos, or other types of data. This HTTP request is also known as insecure unencrypted traffic since it lacks Transport layer security (TLS) or encryption in simple terms. On various exposed Wi-Fi networks, data is transmitted in plain text without TLS, making it vulnerable to interception and reading by hostile parties.

Relevance : Over privilege increases the app's ability to gather private user data without the user's knowledge, such as contacts, location, and camera access and many more sensitive data elements. It goes against the least privilege principle. Unrestricted access to components may make it possible for others malicious apps present on the device to initiate processes or gain unauthorized access to data from other apps. Additionally, insecure requests make it simple to obtain sensitive user information and credentials, making attacks like session hijacking feasible. Additionally, HTTP does not provide integrity checks, allowing hackers to alter and manipulate data while it is in transit, also known as a 'man-in-the-middle attack'. It shows inadequate security practices on the part of developers and poses a serious risk, given that malicious apps search for any opening to exploit. Overall, these are concerning issues that may make the user a victim of malicious actions and privacy violations.

Tool: MobileAudit is an open-source static analysis toolkit for auditing Android applications. It works by decompiling the app's bytecode and scanning it for issues such as excessive permissions, data leakage, and the use of hardware identifiers. MobileAudit generates a detailed report highlighting vulnerabilities to enable users/developers with transparency into an app behavior. My analysis using the tool uncovered several concerning findings:

Findings: My findings include issues analyzed in 5 popular android apps, each from a unique category – Health, Browsing, Streaming, Shopping and File Sharing. Please refer to the Resource Files to view a detailed report of each APK scan.

- **AirDroid** had 17 exported components, 28 dangerous permissions, and only 1 HTTP request.
- **Pedometer** had 12 exported components, 5 dangerous permissions, and 1038 HTTP requests.
- **XBrowser** had 2 exported components, 3 dangerous permissions, and 42 HTTP requests.
- **Crunchyroll** had 7 exported components, 3 dangerous permissions, and 1130 HTTP requests.
- **FiveBelow** had 21 exported components, 4 dangerous permissions, and 587 HTTP requests.

Overall, the findings revealed that they all have a large attack surface, with an average of 12 exported components per app. Additionally, three out of 5 apps make a large number of HTTP requests, with an average of 560 requests per app. This shows that applications may be vulnerable to a range of serious network attacks. The five apps request an average of

31 permissions each with all apps requesting a minimum of 3 Dangerous permissions, and these permissions have no direct relation to the apps functionality, like – ‘android.permission.ACCESS_FINE_LOCATION’ and ‘android.permission.READ_PHONE_STATE’. It is important to point out that AirDroid requests access to an astounding number of 28 Dangerous permissions. This suggests that the apps may be collecting more data than is necessary or that they may be sending data to third-party servers without the user's consent.

Implications: These findings demonstrate the widespread potential for abuse of access to private data and insecure implementation by developers. To reduce the risks associated with over privilege, unrestricted/unsafe exposure, and unencrypted traffic in apps, it is recommended that users closely review permissions before installing and install apps from trusted developers only. Developers need to limit component access scopes, encrypt network transmission, and adhere to the “least privilege principle” to minimize attack surfaces. Lastly, app markets can mandate transport layer security, impose more strict screening procedures around permissions justification, and offer transparency regarding data handling and implementation procedures.

Replication Steps:

- **Download & Installation URL for 'MobileAudit'** : https://github.com/mpast/mobileAudit | Version - 3.0.0

- **Steps to Install & Run :**

1. Install Docker and Docker Compose on your machine if not already installed.
2. Clone the 'MobileAudit' Github repository to your desired directory --
   `$ git clone https://github.com/mpast/mobileAudit`
3. Navigate to the repository directory from the terminal -- $ `cd MobileAudit`
4. Build the Docker image using the command -- $ `docker-compose build`
5. Once the image is built, start the container using command -- $ `docker-compose up`
6. Open your browser and go to http://localhost:8888 to access the MobileAudit dashboard.
7. Once you can access the dashboard, you can click on the 'New App' button to start a scan.
8. Enter the name and description of the app that you want to scan and click on 'Create'.
9. The previous step will take you to the next page where you upload the APK of the app.
10. Once the APK is uploaded, the tool will scan the APK file and generate a report with all its findings.
11. Once the scan is completed, the report can be analyzed on the webpage itself or downloaded.
12. To stop the container, use command -- $ `docker-compose down`
13. To start the container again later -- $ `docker-compose up`

(Note : The installation steps mentioned above for this tool are platform independent and are not specific to Windows, Linux, or Mac)

(I faced a 502 Bad Gateway error, and an OSError write error while running the tool – I updated the 'Dockerfile' in the 'mobileAudit' directory by adding environment variables – 'ENV UWSGI_PARAMS=--enable-threads' 'ENV UWSGI_PARAMS=--buffer-size=65535' | I also ran a command in the 'mobileAudit' directory – $ docker run --name nginx -d -v /root/nginx/conf:/etc/nginx/conf.d -p 443:443 -p 80:80 nginx)

- **APK Details :**

1. **AirDroid** – Version - 4.3.2.0 | 50M + Downloads
   - ✓ SHA1 Checksum - c4acd6250b5a290094a1821dc07b54f58aad02f5
2. **Step Counter - Pedometer** - Version: 1.3.6 (94) | 50M + Downloads
   - ✓ SHA1 Checksum - 6119fe1abf4051c267042d973e98ddedc2251ee1
3. **XBrowser - Mini & Super-fast** - Version: 4.2.1 (781) | 10M + Downloads
   - ✓ SHA1 Checksum - ba2c97895f308bce56f935d6f0aab469ef0fce1f
4. **Crunchyroll** - Version: 3.42.1 (691) | 100M + Downloads
   - ✓ SHA1 Checksum - 2cd9a38857cdf91b7e595b90597ddc0f9df7e04f
5. **FiveBelow** - Version: 5.50.63.00 (5506300) | 1M + Downloads
   - ✓ SHA1 Checksum - aa2cc8b27705ebb9966abf8b9394d445029d5044