Web Android Static Analysis Tools

1. Team Members:

Hung-Yi Chen, Zhenhua He, Yifan Jiang, Hao Yuan
Youzhi Luo (Product Owner), Xin Zhang (Scrum Master)

2. Links

GitHub Link: https://github.com/xintamu/CSCE606Cloudroid
Poster Presentation Link: https://youtu.be/pQN8GSe2eqs
Video Demo Link: https://youtu.be/QDUueA273LI

3. Two Paragraph Summary

The objective of this project was to develop a scalable Android Analysis framework where the majority of computation is completed by leveraging the cloud resources. In our efforts to achieve this goal, a broad range of novel techniques are employed in this project, such as Django, Vuejs frontend, FlowDroid, Kubernetes, and Amazon AWS. Detailed implementations and references are listed in the following.

To be able to analyze uploaded Android applications, the FlowDriod, which is a novel and highly precise static analysis for Android applications, were integrated with a dockerized web service. Next, a Kuberbetes cluster was employed to determine which applications or workloads should be running. The use of Kubernetes enables developers to scale up their service when needed and run more instances based on external requests. We have developed a Django application along with a Vuejs frontend to handle the communication between users and the FlowDroid. Users can upload their Android applications via the Django application. Next, the Django application communicates with the FlowDriod service to initiate the analysis. To improve user experience, we manage to incorporate two more functions, one being the user guide where detailed documentations are provided, the other one being user comments where users' feedback can be collected and viewed.

4. User Stories

4.1. Enrich the user instructions (50 points)

The user guide allows users to quickly grasp how to get started and proceed. Thus, it is of importance to guide users with detailed instructions. In the documentation, a list of descriptions associated with FlowDriod, Kubernetes, Django Server, and Vue.js frontend is provided. The way we organize those sections have been described in the poster. The core service is achieved by FlowDroid. While there are several different static analysis tools, it is advantageous to use FlowDroid because FlowDriod is an open-source service that can analyze a complete APK file without requiring additional configuration. Instead of interfacing directly with FlowDroid, what users need to do is simply to upload the APK file and run the Cloudroid.

4.2. Improve the error-handling (20 points)

For large scale software, one of the most challenging tasks is to properly handle unexpected errors and bugs. We thus added a comment function to enable uses to provide feedbacks. Moreover, by leveraging those submitted comments, we can develop more testing case to cope with various scenarios. While we realize that only limited situations can be examined with existing test cases, our expectation is that users' feedback can facilitate the update and improvement of the software performance.

4.3. Vue.js Frontend (10 points)

We employed the Vue.js to build the web application. Vue.js is an open-source model-view-viewmodel framework for building user interfaces and single-page applications. Moreover, we are able to achieve separate control between the frontend and the backend. Techniques used for frontend development includes webpack, es-linter to achieve better code quality. Users are required to fill the email address when uploading the apk. To retrieve their analysis report, it is critical to guarantee that the input email address is active and correct. To handle cases where users happen to provide wrong email address, we developed an email checking function. Specifically, the software system will first send a verification email to the user. After confirmation, the user can proceed to upload the apk.

4.4. Django server (10 points)

The Django backend handles the user submissions and read operation of analysis report. A series of operations, such as admin site, traffic control, and permission policy, has been implemented as well. For the consideration of robustness, flexibility, and scalability, we used Django and Celery, combined with RabbiMQ. After uploading the apk, a post-signal will be triggered and a certain number of Celery workers will work on the task and execute concurrently. We met with our customer and leaned about the overall objective of this project.

5. Iteration Summary

5.1. Iteration0 (10 points)

We met with our customer and learned about the objective of this project.

5.2. Iteration1 (20 points)

Set up Kubernetes

Learned about FlowDroid

5.3. Iteration2 (50 points)

Learned about Django frameworks

Worked on the Django backend

5.4. Iteration3 (70 points)

Learned about Vue.js frameworks

Worked on the Vue.js Frontend

5.5. Iteration4 (100 points)

Completed frontend and backend communication

6. Customer meeting dates

5.1 February 24th: learned about the overall objective of this project.

5.2 March 17th: Discussed with the previous product owner.

5.3 April 18th: Discusses with TA about unsolved difficulties.

5.4 May 5th: Final demo of the entire project

7. BDD/TDD testing

Since this project is not completed based on Ruby on Rails. Majority of our work lies in learning about new techniques and configuring various frameworks. Components that can be tested includes the Django server and Flowdroid service. The goal of those tastings is to ensure that these modules can function as expected. The poster presentation and final demo incorporate all aspects of this project. It would make more sense to demonstrate those pieces via recorded videos.