Yu-Shiang Dang
pulorsok@gmail.com
Kaohsiung, Taiwan
Timezone: Taiwan (GMT+8)

New Rule Generation Technique & Make Quark Everywhere Among Security Open Source Projects

GSoC Project proposal for Quark-Engine

Personal Details and Contact Information

- **Github Username:** pulorsok
- **Email:** pulorsok@gmail.com
- **University:** National Kaohsiung University of Science and Technology
- **Time-zone:** Taiwan (GMT+8)
- **Address:** No. 1, Ln. 87, Wannian E. Rd., Douliou City, Yunlin County 640106, Taiwan (R.O.C.)
- **Slack Username:** Shaun Dang (Yu-Shiang Dang)
- **Primary Spoken Language:** Mandarin Chinese
- **GitHub Profile:** https://github.com/pulorsok

Synopsis

Quark-Engine is a rule-based android malware detection tool, However, as a contributor of both Quark and its side projects (quark-rule-generate, ruleviewer), we know that Quark currently faces a problem of insufficient number of detection rules. Although Quark-Engine has a side project for automatically generating rules, the performance of generating detection rules is still low, and the multi-process technique used in this project still has problems. Thus, my first object of this proposal is to develop and implement a new strategy for the rule generation technique to improve the efficiency of rule generation and solve the multi-process issues.

Furthermore, the Quark team is now executing a strategy called “Quark Everywhere” which is to expand the influence of Quark among security open source projects. Thus, my second object of this proposal is to help Quark-Engine integrate with other security open-source projects and improve user experiences of Quark-related functionalities to the integrated projects (APKLab, Jadx).

Benefits to the Community

Two benefits from developing and implementing new rule generation technique:

1. Much faster to find the important rules. In other words, it made Quark practical and easy to use for malware analyzers.
2. Reduce the efforts of manual validation for filtering garbage rules.

Two benefits from executing the strategy of “Make Quark Everywhere Among Security Open Source Projects”:

1. To expand the influences of Quark to other security open source projects.
2. To attract more users and contributors, making Quark a healthier open source project.

Current Status of the Project

I have initiated a side project of Quark-Engine to automatically analyze Android APK and generate detection rules of Quark. The project name is called “quark-rule-generate”. However, the rule generation process is terribly slow despite that we applied the multi-process technique to raise the performance.

I also helped the team to integrate Quark to several open source projects (including Jadx, APKLab). Also, the team submitted a proposal to MobSF for Quark integration and got accepted. However, some works are still there to be done (e.g. Fix issues to complete the integration to Jadx, improve user experience of Quark functionalities in Jadx and APKLab, complete integration to MobSF)

Goals

New Rule Generation Technique

- Implement a new technique for rule generation.
- Solve CPU idle problem when applying multi-process technique.

Make Quark Everywhere Among Security Open Source Projects

- Integrate Quark-Engine to MobSF (A mobile application pen-testing, malware analysis and security assessment framework.)
- Improve user experience of Quark functionality in Jadx and APKLab.
- Fix issues to complete the integration to Jadx.
- Add call graphs feature of Quark to APKLab.

Deliverables

New Rule Generation Technique

- Implement a feature that calculates statistics of native APIs.
- Implement a feature that divides the least used native APIs into a number of groups
- Implement a feature that schedules a group of APIs to idle CPU processes to pairing APIs, and reschedules when it finishes.
- A performance report for the new technique

Make Quark Everywhere Among Security Open Source Projects

- A successful Quark integration to MobSF.
- Implement a tree of call graphs feature of Quark in APKLab.
- Discuss practical approaches about how to improve user experiences of using Quark functionalities in APKLab with APKLab members and implement it.
- Show error dialogue when the Quark process fails in Jadx to improve user experiences.
- Show more detail of Quark's summary report in Jadx (e.g. the function where the malicious behaviour is executed) to improve user experiences.

Expected Results

New Rule Generation Technique

The goal of the new rule generation technique including:

- Reduce at least 50% of analysis time.
- The number of validated rules generated by the new strategy must be no less than 90% of the old strategy.
- Solved CPU idle problem when applying multi-process technique.

Make Quark Everywhere Among Security Open Source Projects

The goal of Quark Everywhere:

- Increase visibility of Quark and the number of Quark users/contributors.
- Create synergy between quark and open source projects.
- Grow Quark community.

Approach

New Rule Generation Technique

The approach of the new rule generation technique including 4 steps:

1. Extract native APIs used in the targeted apk.
2. Calculate statistics of native API usage and find all pairs from the top N% least used native APIs. (The reason we find pairs from the least used APIs is that we have observed that APIs used in the highest frequency are probably not helpful. For example, the API, toString() is everywhere, but it doesn't help! So we think, if we can start from the least used one might be a smart way to find useful rules in the early stage of rule generation.)
3. We adjust to find the best N that helps to generate important rules with least computation resources.
4. To solve the CPU idle problem, we divide the top N% native APIs into a number of groups and scheduling a group of APIs to idle CPU processes to pairing APIs, and rescheduling when it finishes.

Make Quark Everywhere Among Security Open Source Projects

The approach of implementing Quark to MobSF including implementation of 2 features:

1. Analyze APK using Quark and show the summary report of Quark.
2. Highlight the source code of the function and key APIs when the user clicks on the potential malware behaviour on the summary report.

The approach of add call graph (feature of Quark) and improve user experiences of Quark functionalities in APKLab including implementation of 4 features:

1. Show call graph of Quark on vscode panel.
2. User clicks on a parent function that he/she wants to trace.
3. Highlight the function and key APIs in the source code.
4. Discuss practical approaches about how to improve user experiences of using Quark functionalities in APKLab with APKLab members and implement it.

The approach of improve user experiences of Quark functionalities in Jadx including implementation of 2 features:

1. Show error dialogue when the process fails in Jadx.
2. Show more detail for the summary report (e.g. the function where malware behaviour executed)

Timeline

Period	Task
After proposal submission April 15 - May 18	- Discuss, implements and adjust the automatic rule generation technique with Quark-Engine members.
May 18 - May 25	- Improve user experience of Quark functionalities in Jadx. - Solve quark integration issues and complete the integration to Jadx.
May 25 - June 1	- Refactor the old rule generation technique. - Implement test unit for quark-rule-generate.
June 1 - June 15	- Implement new rule generation technique.
June 15 - Jun 22	- Testing and comparing the efficiency between new techniques and original one.
June 22 - June 29	- Familiarize with the MobSF code base, design UI/UX, functionality for Quark integration.
June 29 - July 13	- Implement Quark integration to MobSF.
July 13 - July 27	- Testing the integration and submitting PR to MobSF. - Solve the integration issues from MobSF's feedback.
July 27 - Aug 10	- Implement call graph (feature of Quark) to APKLab integration.
Aug 10 - Aug 17	- For Documentation.

A buffer of two weeks has been kept for any unpredictable delay.

About Me

My name is Yu-Shiang Dang, I am a master's student from National Kaohsiung University of Science and Technology Taiwan. I am interested in sharpening my skills in cybersecurity and the building of open-source community.

Before applying to GSoC I have already contributed to Quark-Engine for a while. I even initialized a side project called quark-rule-generate which is an automatically rule-generated system that tried to help Quark generate more detection rules automatically. Also, I have implemented the integration of Quark to APKLab and Jadx.

I think Quark-Engine is a highly potential security tool for malware detection, it has grown very fast in the past year, and I will keep contributing and collaborating with them even after the GSoC.