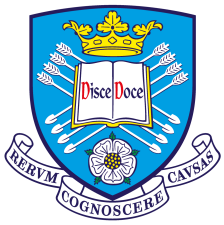

The University Of Sheffield.

Finding Security Issues in (Open Source) Software Repositories

Zer Jun Eng

supervised by

Dr. Achim Brucker

This report is submitted in partial fulfilment of the requirement for the degree of MEng Software Enginnering by Zer Jun Eng

COM3610

21st September 2018

Contents

Chapter 1

Introduction

1.1 Background

Free/Libre and Open Source Software (**FLOSS**) is a type of software which license allows the users to inspect, use, modify and redistribute the software's source code [2]. Since the introduction of Git, and later the Git repositories hosting site such as GitHub, many users have started to make their softwares open source by storing them as public repositories on GitHub. As a result, the participation of global communities into **FLOSS** projects has started to grow and different contributions were made to improve the softwares quality, which included fixing the software vulnerabilities [3].

Building a secure software is expensive, difficult, and time-consuming. In **FLOSS** projects, it is necessary to know when and how a security vulnerability is fixed. Therefore, having a list of changelogs or informative git commit messages that record the fixed security vulnerabilities is helpful. However, Arora and Telang [1] stated that some open source developers believe that public disclosure of security vulnerabilities patch is dangerous, and thus vulnerability fixing commits are not commonly identified in some open source software repositories to prevent malicious exploits. In this case, a repository mining tool that investigate vulnerability patterns and identify vulnerable software components can be developed to reduce the time and cost required to mitigate the vulnerabilities.

1.2 Objectives

- Develop a repository mining tool that searches through the commit history to find the actual code commits which addressed the vulnerabilities.
- Mine commit messages from a list of software repositories to collect the vulnerability patterns. The patterns should be represented using regular expressions.
- Extend the project and the repository mining tool by using the identified patterns to determine whether an application is affected by potential vulnerabilities.

1.3 Challenges

- **Data**: There are a lot of popular open source repositories available on GitHub. However, it is challenging to find a set of sample repositories that can produce accurate and consistent results.
- **Time**: Large repository such as Linux which have more than 780,000 commits in total [4] could be extremely time-consuming for the repository mining tool to complete the search.
- **Evaluation**: After mining a list of commit messages that contain the identified patterns, it might not be be intuitive for the tool to determine which one is the real patch containing the fixed security vulnerabilities.

1.4 Report Structure

Chapter 2 reviews a range of academic articles, theories and previous implementations that is related to this project, as well as investigating the techniques and tools to be used.

Chapter 3 is a list of detailed requirements and a thorough analysis for design, implementation and testing stage.

Chapter 4 is a comparison between different design concepts, where the advantages and disadvantages of difference approach are stated. The chosen design is justified with suitable diagrams provided including wireframes and UML.

Chapter 5 describes the implementation process by highlighting novel aspects to the algorithms used. Testing are performed by following a suitable model to evaluate the implementation.

Chapter 6 presents all the results along with critical discussions about the main findings, and outlines the possible improvements that could be made for future work.

Chapter 7 summarises the main points of previous chapters and emphasise the results found.

1.5 Relationship to Degree Programme

This project focusses on researching real-world software security problem by deploying a repository mining tool to open source software repositories with the purpose of studying the patterns of different security vulnerabilities patch. This relates to the 'Software Engineering' degree as it requires a good understanding in version control system and it aims to improve softwares quality by reducing the time and effort needed to locate and fix security vulnerabilities in the source code.

Chapter 2

Analysis

The purpose of this chapter is to discuss the problems to be solved and consider some of the core decisions to be made before starting the implementation.

2.1 Problems

As mentioned in **Section 1.2**, the repository mining tool must be able to detect commits that contain distinct patterns such as *fix*, *patch*, *vulnerability* etc. After extracting a possible list of commits, it should perform an evaluation process to identify the actual commits that fixed security vulnerability. This could be hard because not all open source software repositories are using the same programming language. Hence, it could might be difficult to determine the actual lines of code that addressed the vulnerabilities.

2.2 Tools

- PyGithub is a Python library build to access the GitHub API [6].
- GitPython is a Python library build to interact with Git repositories using a combination of python and git command implementation [5].
- Secbench Mining Tool is a repository mining tool build by The Quasar

Research Group to mine vulnerability patterns from GitHub repositories [7].

2.3 Plan of Action

- **Week 1**: Discuss about the problems encountered when writing the description stage with supervisor. Ask supervisor for feedback if available.
- **Week 2**: Should have finished the draft of description stage by the start of this week. Discuss the draft with supervisor to check for mistakes. Research work should be started during this week.
- **Week 3**: Should have found at least 5 sources, which is related to the project. Plan the outline and start writing the introductory section for literature review.
- **Week 4**: Perform a source analysis and note down the sentences that is related to this project. Started the research for suitable techniques and tools to be used in this project.
- **Week 5**: Organise the sources and start writing the literature review.
- **Week 6**: Finish the literature review and start writing the requirements and analysis.
- **Week 7**: Finish the requirements and analysis. Proof read the document and write the abstract.
- **Week 8**: Submit the first draft of survey and analysis to supervisor to seek early feedback.
- **Week 9**: Amend the document based on the feedback.
- **Week 10**: Ask the supervisor about any final changes.
- **Week 11**: Survey and analysis stage should be completed and ready to submit. Discuss with supervisor about the work to do during the holiday.

Bibliography

[1] A. Arora and R. Telang, 'Economics of software vulnerability disclosure', *IEEE security & privacy*, vol. 3, no. 1, pp. 20–25, 14th Feb. 2005, ISSN: 1540-7993. DOI: 10.1109/MSP.2005.12.

[2] K. Crowston, K. Wei, J. Howison and A. Wiggins, 'Free/libre open-source software development: What we know and what we do not know', *ACM Computing Surveys (CSUR)*, vol. 44, no. 2, p. 7, 1st Feb. 2012, ISSN: 0360-0300. DOI: 10.1145/2089125.2089127.

[3] L. Dabbish, C. Stuart, J. Tsay and J. Herbsleb, 'Social coding in github: Transparency and collaboration in an open software repository', in *Proceedings of the ACM 2012 conference on computer supported cooperative work*, ACM, 11th Feb. 2012, pp. 1277–1286. DOI: 10.1145/2145204.2145396.

[4] *Github - linux kernel source tree*. [Online]. Available: https://github.com/torvalds/linux (visited on 20/09/2018).

[5] *Gitpython*. [Online]. Available: https://github.com/gitpython-developers/GitPython (visited on 20/09/2018).

[6] V. Jacques, *Pygithub*, PyGithub. [Online]. Available: https://github.com/PyGithub/PyGithub (visited on 20/09/2018).

[7] *Secbench mining tool*, The Quasar Research Group. [Online]. Available: https://github.com/TQRG/secbench-mining-tool (visited on 20/09/2018).