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

27th September 2018

# Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name : Zer Jun Eng

Date : 27th 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 [3]. 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 [4].

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

- Identify the security patterns of the most popular security issues in OWASP Top Ten Project [9]. The patterns should be expressed using regular expressions.
- Develop a repository mining tool to search through the commit history of a repository and find a list of commit messages that match the patterns.
- Extend the mining tool to run an evaluation process which checks the code difference in the commits found, where the added lines represent the fix for a vulnerability and the deleted lines represent the vulnerability.

## 1.3 Challenges

- **Data**: There are a large numbers of open source repositories available on GitHub. However, it is challenging to find a set of sample repositories that can produce accurate and consistent results.
- **Evaluation**: After mining a list of commit messages that contain the identified patterns, the evaluation process might not correctly locate the lines of code that addressed the security vulnerability.
- **Time**: Large repository such as Linux which has more than 780,000 commits in total [5] could be extremely time-consuming for the repository mining tool to complete the search and evaluation process.

## 1.4 Report Structure

**Chapter 2** reviews a range of academic articles, theories and previous studies 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 in the 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

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 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 [8].
- GitPython is a Python library build to interact with Git repositories using a combination of python and git command implementation [6].
- Secbench Mining Tool is a repository mining tool build by The Quasar Research Group to mine vulnerability patterns from GitHub repositor-

ies [12].

## 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.

# Chapter 3

# Literature Review

This chapter will explore the previous and existing works related to this project, as well as reviewing the security issues that might exist in an open source Git repository.

## 3.1 Software Security

There are currently two approaches to the license distribution of software: open source and closed source. The users of closed source software are limited to accept the level of security provided by their chosen vendor. In contrast, open source softwares provide more flexibility and freedom over the security option to their users [10], where the users can decide to wait for a patch from the vendor or collaborate with the community to develop their own. In this project, it is worth investigating that whether the security issues exis

Hoepman and Jacobs [7] suggested that open source softwares will have better security and reliability than closed source softwares through the power of open data and crowdsourcing. Conversely, Schryen [11] has shown that open source and closed source softwares do not have significant difference in terms of security vulnerabilities in his experiment. However, Cowan [2] stated that both statements are not totally true, which he indicated that the security of a software is largely determined by the developers but not the source availability model.

## 3.2 Security Issues in Open Source Sotware Repositories

### 3.2.1 OWASP

## 3.3 foo

| | | |
|---|---|---|
| cell1 | cell2 | cell3 |
| cell4 | cell5 | cell6 |
| cell7 | cell8 | cell9 |

**Table 3.1:** cell table

According to **Table 2.1**, 3.1

# 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] C. Cowan, 'Software security for open-source systems', *IEEE Security & Privacy*, vol. 99, no. 1, pp. 38–45, 19th Feb. 2003, ISSN: 1540-7993. DOI: 10.1109/MSECP.2003.1176994.

[3] 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.

[4] 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.

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

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

[7] J.-H. Hoepman and B. Jacobs, 'Increased security through open source', *Communications of the ACM*, vol. 50, no. 1, pp. 79–83, 1st Jan. 2007, ISSN: 0001-0782. DOI: 10.1145/1188913.1188921.

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

[9] *Owasp top ten 2017 project*, The Open Web Application Security Project (OWASP), 20th Oct. 2017. [Online]. Available: `https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project` (visited on 26/09/2018).

[10] C. Payne, 'On the security of open source software', *Information Systems Journal*, vol. 12, no. 1, pp. 61–78, 8th Feb. 2002, ISSN: 1350-1917. DOI: `10.1046/j.1365-2575.2002.00118.x`.

[11] G. Schryen, 'Is open source security a myth?', *Communications of the ACM*, vol. 54, no. 5, pp. 130–140, 1st May 2011, ISSN: 0001-0782. DOI: `10.1145/1941487.1941516`.

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