Security Expert Recommender in Software Engineering

Shahab Bayati
ISOM Department, Business School, University of Auckland
Auckland, New Zealand
s.bayati@auckland.ac.nz

ABSTRACT

Software engineering is a complex filed with diverse specialties. By the growth of Internet based applications, information security plays an important role in software development process. Finding expert software engineers who have expertise in information security requires too much effort. Stack Overflow is the largest social Q&A Website in the field of software engineering. Stack Overflow contains developers' posts and answers in different software engineering areas including information security. Security related posts are asked in conjunction with various technologies, programming languages, tools and frameworks. In this paper, the content and metadata of Stack Overflow is analysed to find experts in diverse software engineering security related concepts using information security ontology.

Keywords

Information Security, Software Engineering, Stack Overflow, Recommender System, Expert Recommendation, Ontology

1. INTRODUCTION

Developing a software application is a complex socio-technical process. Software engineering (SE) depends on variety of technologies, frameworks, programming languages (PL) and skills. One of the most important issues in software development process are security related features and bugs [1, 2]. Security bugs in software development may lead to undesirable defects in software applications. By the growth of Internet and mobile applications, value of information security techniques gets more obvious. Studies in SE shows the priority and high impact of security bugs in software development life cycle (SDLC) [3]. Many of the software engineering teams do not pay enough attention to secure programming and security issues which may have hard side effects. Lack of security experts in development team is one of the reasons for security defects. By focusing on the importance of information security in SDLC process, this study proposed an expert recommender system for security issues in software projects.

Stack Overflow (SO) as the largest social Q&A platform of software engineering communications is highly used by developers [4]. It is a common practice in software engineering community to seek for the problem solutions in SO. Experts answer developers' questions in a shortest possible time [4].

Different studies in SE and MSR (mining software repositories) analysed SO data with different perspectives which illustrate the value of this online platform [5-7]. Along with its social Q&A features SO provides reputation management through gamification approach, voting, badges and tags [8]. SO collected data about 5 Million users since 2008 with more than 10 million posted questions. Its Alexa.com rank is less than 50 which is very impressive for a domain specific platform. In this study SO data and metadata is used to find security experts based on the project requirements.

Although some valuable studies have already done in the field of expert recommendation in software engineering [9, 10], however, the main contributions of this study include focusing on finding information security expert using SO data which is based on novel filtering and evaluation factors, and using information security ontology to relate SO tags with project security requirements. These contributions are proposed through a framework in the rest of this study. To have a big picture of this paper, after introduction section related works are presented. Then research method is elaborated. Proposed framework details and recommendation techniques are illustrated in the next section. Finally, the last discussions and conclusion are presented.

2. RELATED WORKS

2.1. Stack Overflow Mining

A classification mining research is done on unanswered questions in SO Q&A Website [6]. Reputation factors of SO users are mined in [8]. In a text analysis research on SO questions, topic modeling analysis (LDA) is used to identify the relation among questions concepts, types and codes [5]. A paper focused on analyzing expertise of developer who attends on SO to answer the questions. They checked the user participation in GitHub projects by technical term analysis and its relation to tags in SO [11]. SO is mined to recommend the code example to the JQuery developers [4]. Mobile programming related posts on SO is mined by LDA text analysis approach in [12].

2.2. Expert Recommendation

Developer interactions to IDE log is used to define the developers' expertise on a specific entity in SE project [13]. A recommender system applies usage expertise to find the experts in code base repository [9]. Some studies focused on finding experts for bug triaging purpose [14, 15]. Latent semantic indexing is used to find a list of expert developers for unassigned issues [10]. Experts are recommended for mentoring newbie developers in open source [16]. Social network analysis and user's reputation are used to find experts in Q&A Website [17].

3. RESEARCH APPROACH

Our data analysis on SO data shows the lack of posts and experts in the area of information security. Moreover, by considering the importance of information security defects on software projects

ICSE '16 Companion, May 14-22, 2016, Austin, TX, USA
ACM 978-1-4503-4205-6/16/05.
DOI: http://dx.doi.org/10.1145/2889160.2892648

[18, 19] the needs for security experts in software projects are obvious. Currently based on our analysis overall response time for each post in SO is about 20-30 minutes which is ten times higher for information security related posts. Stack Overflow data is imported to our system through SEDE (Stack Exchange Data Explorer).

Information security ontology in OWL (Web Ontology Language) format is used to retrieve security based terms [20]. These terms are basically used in this study to find experts based on used tags in SO. This approach is used to find security related issues in [21]. Project coordinators and mangers can input their security requirement explicitly and implicitly. In the implicit way list of project security requirements and issue reports are entered to the system. Security term frequencies are extracted from these documents and matched to the similar concept and node in information security ontology. Then the queries are built based on semantically related tags in SO to find experts. Experts are extracted based on their answers to the posts related to tags and votes they get from other users and list of badges they gained for related tags. It is also possible for the system users to enter their security requirements as a list of terms from information security glossary.

As the list of experts in semantically related tags is extracted, we can filter them based on other conditions to have a more appropriate list of experts. For each user in SO the most important items which represent her reputation in a fields are (votes, answered questions count) and for general expert selection are (last access date, total reputation, total votes, Total profile views). Because different projects use different technologies (ASP, JSP, Web Services, Mobile …) and PLs (C#, Java, VB, Python …) experts should be filtered by their reputation in other related tags to these technologies and PLs. In this study we just do the second round of filtering based on the PLs. In parallel as an optional item to our system we filter the list of experts based on their locations. It is an item for projects which require local experts. A threshold level is settable for the minimum reputation requirement. This list of experts send to ranker for sorting based on the ranking formula. Equation 1 shows the ranking formula

$$\boldsymbol{ExpRank}(\boldsymbol{ex}) = \frac{\mathbf{1}}{\boldsymbol{SecRank}(\boldsymbol{ex})} + \frac{\mathbf{1}}{\boldsymbol{PLRank}(\boldsymbol{ex})} + \boldsymbol{LocFunc}(\boldsymbol{ex}) \qquad \text{Eq 1}$$

In equation 1, *SecRank(ex)* shows the rank of the expert in the expert list based on the security terms and up-votes and other factors related to security. Also semantic similarity is applied. Equation 2 shows this evaluation formula.

$$\boldsymbol{SecRank}(\boldsymbol{ex}) = \frac{\sum_{i-1}^{N} \frac{\boldsymbol{VoteRankSecTerm}(\boldsymbol{tg}(\boldsymbol{i}))}{\boldsymbol{SemSim}(\boldsymbol{tm}(\boldsymbol{i}),\boldsymbol{tg}(\boldsymbol{i}))} + \frac{\boldsymbol{AnswRankSecTerm}(\boldsymbol{tg}(\boldsymbol{i}))}{\boldsymbol{SemSim}(\boldsymbol{tm}(\boldsymbol{i}),\boldsymbol{tg}(\boldsymbol{i}))}}{\boldsymbol{N} \times \mathbf{2}} \qquad \text{Eq 2}$$

In equation 2, *N* is the number of selected tags and *tg* and *tm* represent tag and term. *SemSim(tm(i), tg(i))* evaluates the semantic similarity between the used tag and the requested term the result has a value between 0-1 and 1 represents the highest possible similarity. We used the semantic similarity function of [22]. Also in equation 1 the same process applies for *PLRank(ex)* just in this case we do not need the value for Semantic similarity and it is always 1. For *LocFunc(ex),* it can get three different values. For same city it has value of 1, else if it is just in same country value of 0.5 and 0 for the rests. The results is sorted based on *ExpRank* values and represented to the project organizers. Just to keep the content of database updated, refresher automatically imports newly generated records to related tables.

4. PROPOSED FRAMEWORK

In this section based on the presented methodology the proposed framework is shown in figure 1. Also algorithm 1 presents the flow of process in recommendation phase. This framework has a R-DBMS database to store SO data. It contains Query analyser to extract security terms from requirements and security Ontology in OWL for similarity matching. Also it has a Recommender engine to find experts and a Ranker to list and sort experts.

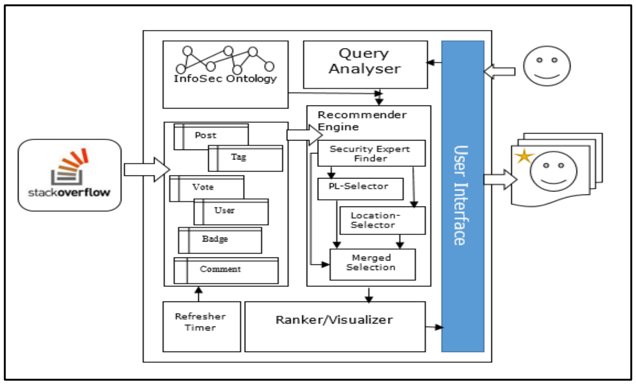

Figure 1: Proposed framework

```
Algorithm 1: Expert Recommendation
Inputs:
        Trms (List of Requested Security Terms), PL, Location, Td
        (Threshold)
Output:
        RankedExps (List of Experts)
For t in Trms
        SecExperts.Add(FindExpertInTag(t,Td))
For exp in SecExperts
        if IsExpertIn(exp,PL)
                PLExpert.Add(exp)
        if IsInLocation(exp,Location)
                LocExperts.Add(exp)
FilteredExperts=Merge(PLexperts,LocExperts)
For exp in FilteredExperts
        RankedExps=ExpRank(exp)
```

5. CONCLUSION, EXPECTATIONS AND FUTURE WORKS

This study presented security expert recommender system in software engineering. To find security experts, stack overflow data in analyzed. Security ontology and glossary is used to find the experts who answered to the posts with similar tags in SO with the highest answer count and vote value. Also to find the right information security experts for the requested software engineering application the project programming language tags are used. Moreover, expert location in her profile is used for the project which needs an expert in their local area. All of these are proposed through a framework.

This research has some limitations and can be improved in different ways. As it is mentioned in the content other related technological factors can be used for filtering and finding experts. In this study the weights for ranking parameters is set equal which needs optimization. In this study the value for all posts, answers and tags are evaluated equally which is not in reality. The time effect on experts' skills is not captured. All of these limitations can be covered in future works.

6. REFERENCES

[1] M. Gegick, P. Rotella, and T. Xie, "Identifying security bug reports via text mining: An industrial case study," in Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on, 2010, pp. 11-20.

[2] K. Tsipenyuk, B. Chess, and G. McGraw, "Seven pernicious kingdoms: A taxonomy of software security errors," Security & Privacy, IEEE, vol. 3, pp. 81-84, 2005.

[3] K. R. Van Wyk and G. McGraw, "Bridging the gap between software development and information security," Security & Privacy, IEEE, vol. 3, pp. 75-79, 2005.

[4] A. Zagalsky, O. Barzilay, and A. Yehudai, "Example overflow: Using social media for code recommendation," in Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering, 2012, pp. 38-42.

[5] M. Allamanis and C. Sutton, "Why, when, and what: analyzing stack overflow questions by topic, type, and code," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 53-56.

[6] M. Asaduzzaman, A. S. Mashiyat, C. K. Roy, and K. A. Schneider, "Answering questions about unanswered questions of stack overflow," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 97-100.

[7] R. Stevens, J. Ganz, V. Filkov, P. Devanbu, and H. Chen, "Asking for (and about) permissions used by android apps," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 31-40.

[8] A. Bosu, C. S. Corley, D. Heaton, D. Chatterji, J. C. Carver, and N. A. Kraft, "Building reputation in stackoverflow: an empirical investigation," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 89-92.

[9] D. Schuler and T. Zimmermann, "Mining usage expertise from version archives," in Proceedings of the 2008 international working conference on Mining software repositories, 2008, pp. 121-124.

[10] H. Kagdi, M. Gethers, D. Poshyvanyk, and M. Hammad, "Assigning change requests to software developers," Journal of Software: Evolution and Process, vol. 24, pp. 3-33, 2012.

[11] R. Venkataramani, A. Gupta, A. Asadullah, B. Muddu, and V. Bhat, "Discovery of technical expertise from open source code repositories," in Proceedings of the 22nd international conference on World Wide Web companion, 2013, pp. 97-98.

[12] M. Linares-Vásquez, B. Dit, and D. Poshyvanyk, "An exploratory analysis of mobile development issues using stack overflow," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 93-96.

[13] R. Robbes and D. Röthlisberger, "Using developer interaction data to compare expertise metrics," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 297-300.

[14] R. Shokripour, J. Anvik, Z. M. Kasirun, and S. Zamani, "Why so complicated? simple term filtering and weighting for location-based bug report assignment recommendation," in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 2-11.

[15] H. Naguib, N. Narayan, B. Brugge, and D. Helal, "Bug report assignee recommendation using activity profiles," in Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on, 2013, pp. 22-30.

[16] I. Steinmacher, I. S. Wiese, and M. A. Gerosa, "Recommending mentors to software project newcomers," in Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop on, 2012, pp. 63-67.

[17] D.-R. Liu, Y.-H. Chen, W.-C. Kao, and H.-W. Wang, "Integrating expert profile, reputation and link analysis for expert finding in question-answering websites," Information Processing & Management, vol. 49, pp. 312-329, 2013.

[18] M. Ohira, Y. Kashiwa, Y. Yamatani, H. Yoshiyuki, Y. Maeda, N. Limsettho, et al., "A Dataset of High Impact Bugs: Manually-Classified Issue Reports," 2014.

[19] S. Zaman, B. Adams, and A. E. Hassan, "Security versus performance bugs: a case study on firefox," in Proceedings of the 8th working conference on mining software repositories, 2011, pp. 93-102.

[20] V. Raskin, C. F. Hempelmann, K. E. Triezenberg, and S. Nirenburg, "Ontology in information security: a useful theoretical foundation and methodological tool," in Proceedings of the 2001 workshop on New security paradigms, 2001, pp. 53-59.

[21] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: sentiment analysis of security discussions on GitHub," in Proceedings of the 11th Working Conference on Mining Software Repositories, 2014, pp. 348-351.

[22] J. Cardoso and A. Sheth, "Semantic e-workflow composition," Journal of Intelligent Information Systems, vol. 21, pp. 191-225, 2003.