Security and Emotion: Sentiment Analysis of Security Discussions on GitHub

Daniel Pletea, Bogdan Vasilescu, Alexander Serebrenik
Dept. of Mathematics and Computer Science
Eindhoven University of Technology
POB 513, 5600 MB Eindhoven, The Netherlands
d.pletea@student.tue.nl, {b.n.vasilescu, a.serebrenik}@tue.nl

ABSTRACT

Application security is becoming increasingly prevalent during software and especially web application development. Consequently, countermeasures are continuously being discussed and built into applications, with the goal of reducing the risk that unauthorized code will be able to access, steal, modify, or delete sensitive data. In this paper we gauged the presence and atmosphere surrounding security-related discussions on GitHub, as mined from discussions around commits and pull requests. First, we found that security-related discussions account for approximately 10% of all discussions on GitHub. Second, we found that more negative emotions are expressed in security-related discussions than in other discussions. These findings confirm the importance of properly training developers to address security concerns in their applications as well as the need to test applications thoroughly for security vulnerabilities in order to reduce frustration and improve overall project atmosphere.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*data mining*; K.6.5 [**Management of Computing and Information Systems**]: Security and protection

General Terms

Security, Human Factors

Keywords

Security, GitHub, sentiment analysis, mining challenge

1. INTRODUCTION

Application security is becoming increasingly prevalent during software and especially web application development. Security vulnerabilities are costlier than traditional bugs [9] and may lead to disclosure of sensitive, confidential, or personally identifiable data as well as legal ramifications [4]. Consequently, countermeasures are continuously being discussed and built into applications, with the goal of reducing the risk that unauthorized code will be able to access, steal, modify, or delete sensitive data. Moreover, despite their potentially huge impact, security concerns are often given only a side thought. This is happening because programmers are typically trained to write code that implements the required functionality without considering its security aspects [7].

In this paper we gauge the presence and atmosphere surrounding security-related discussions on GitHub. GitHub is the largest code host in the world, with more than 5M developers collaborating across 10M repositories. GitHub offers support for distributed version control (Git) and pull-based development, and allows developers to comment on commits or pull requests (e.g., as a means to perform code reviews).

Our contributions are two-fold. First, we assess the fraction of security-related discussions among commit or pull request discussions. Second, we explore expressions of emotion in security-related discussions and compare these to the general atmosphere of non-security-related discussions. We find that security-related discussions account for approximately 10% of all discussions on GitHub, and that they have a more negative tone than other discussions. Our findings confirm the importance of properly training developers to address security concerns in their applications as well as the need to test applications thoroughly for security vulnerabilities in order to reduce frustration and improve overall project atmosphere.

The rest of this paper is organised as follows. We discuss the related work in Section 2, followed by our methodology in Section 3 and results in Section 4. Finally, we conclude in Section 5.

2. RELATED WORK

The sentiment analysis of security comments from GitHub projects consists of two big stages: the detection of comments related to the security topic and the sentiment analysis of all the comments. Our approach to do identification of security-related comments and discussions can be seen as related to topic mining. While different topic mining techniques have been discussed in the literature [1, 10], the approach closest to one we use is due to Warten and Brussee [13]. The solution proposed is based on extracting keywords and clustering them. The resulted clusters represent different topics. Sentiment analysis has been based in the past on such classification techniques as Naive Bayes, Support Vector Machines and Maximum Entropy classifi-

MSR'14, May 31 – June 1, 2014, Hyderabad, India
ACM 978-1-4503-2863-0/14/05
http://dx.doi.org/10.1145/2597073.2597117

cation. A list of more than 20 sentiment analysis apis [6] summarizes the available sentiment analysis tools. Finally, GitHub itself has been subject of a number of recent studies [3, 11].

3. METHODOLOGY

3.1 Dataset

We analyzed the commits (60,658) and pull requests (54,892) from the software projects present in the MSR 2014 Mining Challenge Dataset [2]. The MSR14 dataset contains data for 90 GitHub projects (repositories) and their forks. Our analysis was restricted to the tables containing comments on commits and pull requests. We will use the term "discussion" to denote the collection of comments belonging to a commit or a pull request. We analysed both comments as well as discussions, because sentiment analysis tools are often sensitive to the amount of text available.

3.2 Identification of Security-Related Comments and Discussions

Most related work is centered around *clustering* the input, which consists of a set of entities (e.g., news items, text). Instead of first clustering and then identifying which cluster can be associated with application security, we followed a *keywords-based* approach.

To construct a set of relevant keywords, we followed a iterative process. First, we manually selected a number of keywords based on our experience with application security as well as the literature [5], including such words as *security, ssl, encryption, authentication, authorization, encrypt, decrypt, audit, integrity, repudiation, confidentiality, privacy, ldap, dsa.* Next, we determined for each seed in this set the corresponding Stack Overflow tag (if any), and enlarged our set of keywords with co-occurring tags from Stack Overflow. To ensure relevancy of the tags collected from Stack Overflow, we (1) only looked at the top 25 co-occurring tags; (2) excluded tags representing programming languages [12]; (3) scored candidate tags depending on the number of Stack Overflow questions tagged with both the original seed and the candidate tag. Using other Stack Exchange websites (e.g. Information Security) to collect additional relevant keywords is considered as future work.

The resulting keywords were reviewed and corrected manually. Finally, we performed Porter stemming on the set of keywords. For completion, our final list of keywords was: *access policy, access role, access-policy, access-role, accesspolicy, accessrole, aes, audit, authentic, authority, authoriz, biometric, black list, black-list, blacklist, blacklist, cbc, certificate, checksum, cipher, clearance, confidentiality, cookie, crc, credential, crypt, csrf, decode, defensive programming, defensive-programming, delegation, denial of service, denial-of-service, diffie-hellman, dmz, dotfuscator, dsa, ecdsa, encode, escrow, exploit, firewall, forge, forgery, gss api, gss-api, gssapi, hack, hash, hmac, honey pot, honeypot, honeypot, inject, integrity, kerberos, ldap, login, malware, md5, nonce, nss, oauth, obfuscat, open auth, openauth, openauth, openid, owasp, password, pbkdf2, pgp, phishing, pki, privacy, private key, private-key, privatekey, privilege, public key, public-key, publickey, rbac, rc4, repudiation, rfc 2898, rfc-2898, rfc2898, rijndael, rootkit, rsa, salt, saml, sanitiz, secur, sha, shell code, shell-code, shellcode, shibboleth, signature, signed, signing, sing sign-on, single sign on, single-sign-on, smart assembly, smart-assembly, smartassembly, snif, spam, spnego, spoofing, spyware, ssl, sso, steganography, tampering, trojan, trust, violat, virus, white list, white-list, whitelist, x509, xss.*

We labeled comments as security-related if a full-word search for the three-letter keywords (e.g., sha or sso) or a substring search for all the other keywords returned at least one hit. We treated the three-letter keywords differently (i.e., using full-word search) in order to reduce false positives. Similarly, we labeled discussions as security-related if at least one comment was labeled security-related.

3.3 Sentiment Analysis

To perform sentiment analysis, we used the Natural Language Text Processing (NLTK) tool [8]. Given an input text, NLTK outputs the probabilities that the text is *neutral*, *negative* or *positive* as well as an aggregate label (one of *neutral*, *negative* or *positive*) summarising the three scores. The probabilities for *negative* and *positive* will add up to 1, while *neutral* is standalone. If *neutral* is greater than 0.5 then the label will be *neutral*. Otherwise, the label will be *negative* or *positive*, whichever has the greater probability. The tool was trained on movie reviews and uses two classifiers, a Naive Bayes Classifier and a Hierarchical Classifier.

To overcome the NLTK API limitation of 5000 requests per day per IP, we used one virtual machine in Amazon EC2 and multiple IPs, resulting in approximately 16 hours of processing time.

4. QUESTIONS AND RESULTS

4.1 How many comments and discussions are security related?

The statistics for the security comments and discussions in the commits table is shown in Table 1. It can be seen that the number of security related comments is around 4% in both tables and the number of discussions is around 10%. These percentages translate into quantities of comments and discussions which are big enough in order to derive relevant conclusions for the MSR 2014 Mining Challenge Dataset. The conclusions can in the future be probed on other datasets.

Table 1: Identification of security-related comments and discussions results

Type		Comments	Discussions
Commits	Security	2689 (4.43%)	1809 (9.84%)
	Total	60658	18380
Pull Requests	Security	1932 (3.51%)	1158 (12.06%)
	Total	54892	9602

4.2 Are the security comments or discussions different (sentiment-wise) than the rest of the comments or discussions?

It can be seen in Table 2 and Table 3 that in both types of discussions (around commits and pull requests), the fraction of negative discussions is higher for security-related discussions than for non-security-related discussions: 72.52% vs. 54.2% for commits and 81.00% vs. 69.58% for pull requests.

The aggregation level makes little difference. At comment rather than discussion level, the fraction of negative

Table 2: Commits Sentiment Analysis Statistics

Type		Negative	Neutral	Positive
Discussions	Security	72.52%	10.88%	16.58%
	Rest	54.28%	20.37%	25.33%
Comments	Security	55.59%	23.42%	20.97%
	Rest	46.94%	26.58%	26.47%

Table 3: Pull Requests Sentiment Analysis Statistics

Type		Negative	Neutral	Positive
Discussions	Security	81.00%	5.52%	13.47%
	Rest	69.58%	11.98%	18.42%
Comments	Security	59.83%	19.09%	21.06%
	Rest	50.16%	26.12%	23.70%

comments is higher for security-related comments than for non-security-related comments: 55.59% vs. 46.94% for commits and 59.83% vs. 50.16% for pull requests. The small differences between the percentages can be explained by the fact that comments are smaller than discussions. Sentiment analysis tools are sensitive to the amount of text available. Being smaller, comments lack context and may contain less sentiment. In conclusion comments tend to be more neutral and lack sentiment.

χ^2 tests on contingency tables with the absolute values confirm that the topic of discussion (in this case security) impacts the sentiments expressed in that discussion (in all four combinations of commits/pull requests vs. comments/ discussions $p < 0.0001$).

To formalise the differences between security-related and non-security-related comments and discussions, we performed Wilcoxon signed-rank tests. The Wilcoxon tests took as input two vectors. One vector contained the $negative * (1 - neutral)$ values of the security-related entities, while the second one the $negative*(1-neutral)$ values of the non-security entities (same type as the first vector). The entities are of type comment or discussion. In all four cases (Figure 1) we were able to reject the null hypothesis and accept the alternative hypothesis, with effect sizes upwards of 50,000: security-related entities are more negative than the rest of the entities ($p < 0.0001$).

4.3 Case Study

To gain more insight in these results, we performed a case study on 30 security-related commit discussions. Based on their security scores (the number of keywords found in each discussion), we randomly selected 10 discussions from the top 10%, 10 discussions from the middle 10% and 10 discussions from the bottom 10% of all security-related discussions. The 30 discussions were then shuffled and analyzed (both their relevance as security-related discussions as well as the dominant emotion as compared to the NLTK results) without knowing their security scores and their sentiment analysis results.

Analysis of the discussions yielded a number of observations. First, we noted a significant number of false positives (discussions mislabeled as security-related) among the middle tier: 6/10 false positives and lower tier: 7/10 false positives. In most cases, the discussions had been labeled as security-related due to a single keyword being present in them, while this keyword was in fact (part of) the username of developers contributing to the discussion, referenced in the text. On the other hand, in the top tier most discussions had been correctly classified (2/10 false positive).

Table 4: Case study results (sentiments labeled on a 5-star scale).

Sec. relevance	Discussion (Commit ID)	# sec. keywords	Sec. relevance (human)	Sentiment (tool) neutral (%)	Sentiment (tool) neg (%)	Sentiment (tool) pos (%)	Sentiment (tool) result	Sentiment (human)
High	535033	6	Yes	16.5	42.9	57.0	pos	neg(*)
	256855	4	Yes	17.1	84.2	15.7	neg	neg(*)
	455971	6	Yes	19.1	84.3	15.6	neg	neutral
	131473	5	Yes	21.4	45.8	54.2	pos	neg(*****)
	253685	4	No	20.4	59.1	40.8	neg	pos(*)
	370765	5	Yes	20.0	65.0	34.9	neg	pos(***)
	59082	4	No	19.8	76.4	23.5	neg	neg(*)
	157981	11	Yes	23.9	58.8	41.1	neg	neg(***)
	391963	9	Yes	16.7	71.9	28.0	neg	pos(****)
	272987	4	Yes	22.4	41.6	58.3	pos	neg(*)
Medium	15128	1	No	20.6	71.3	28.6	neg	neutral
	396099	1	No	18.8	74.0	26.0	neg	neg(****)
	132779	1	No	30.6	76.4	23.5	neg	neutral
	295686	1	No	23.9	70.7	29.3	neg	pos(*)
	541007	1	Partial	37.7	71.7	28.2	neg	neg(*)
	199287	1	Partial	18.9	76.4	23.5	neg	neg(*)
	461318	1	Yes	15.0	75.0	24.9	neg	neg(*)
	509384	1	Partial	33.4	67.3	32.7	neg	neutral
	338681	1	No	29.9	75.5	24.4	neg	neg(*)
	511734	1	No	17.6	79.4	20.5	neg	pos(***)
Low	364215	1	No	41.4	44.1	55.8	pos	neg(*)
	274571	1	Partial	30.1	46.5	53.4	pos	neg(**)
	47639	1	Yes	19.3	38.6	61.3	pos	pos(*****)
	277765	1	No	27.0	45.2	54.7	pos	pos(*)
	6491	1	No	37.6	29.6	70.4	pos	neutral
	130367	1	No	15.4	43.6	56.3	pos	pos(*)
	189623	1	No	57.9	35.8	64.1	neutral	pos(***)
	41379	1	Partial	30.9	26.1	73.8	pos	pos(***)
	456580	1	No	26.6	46.6	53.3	pos	pos(***)
	52122	1	No	17.6	46.3	53.6	pos	pos(*****)

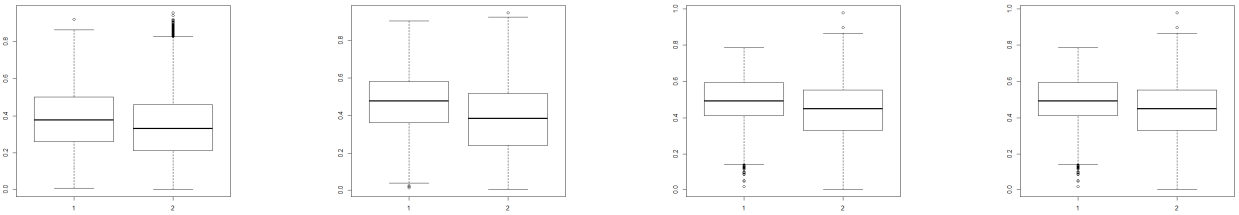
Second, we observed a mixture of agreement and disagreement between the emotion labels computed by NLTK and the ones resulted from manual review. Examples of disagreement include the discussion around commit 535033 (labeled by us as *negative*, by NLTK as *positive*):

> I don't think "cookie_secret" should be a first class option. Unless, that is, we call it something more generic like "secret_token" which is something various things can hook into. I don't see the purpose of having both a "cookie_secret" and a "session" ...

Similarly, we labeled the discussion around commit 391963 as *positive*, although NLTK disagreed:

> Why use md5 here and not sha1? As far as I know, because of its many issues, md5 has been considered bad practice for the last few years [...]
>
> Totally agree - it has broken JMS DI Extra Bundle.

However, even in cases of disagreement, we noted that the NLTK results were mostly bipolar, having both strong *negative* and strong *positive* components. This suggests that while the quantitative analysis resulted in the conclusion security-related discussions tend to be *more negative* than



(a) Commit comments **(b) Commit discussions** **(c) Pull request comments** **(d) Pull request discussions**

Figure 1: Negativity for security-related comments/discussions (1) is higher than for other comments/discussions (2).

non-security-related discussions, perhaps a more accurate interpretation of those results is security-related discussions tend to be *more emotional* than non-security-related discussions.

Factors that might affect the accuracy of the sentiment analysis are the fact that the tool was trained on movie reviews and the fact that in the pull request discussions some pieces of code and/or variable names might be included. Since best practices dictate that variables should have meaningful names, the discussions might end up with a lot of words like 'success' or 'error', which in turn will alter the results of the sentiment analysis.

5. CONCLUSIONS

In this paper we mined emotions from security-related discussions around commits and pull requests on GitHub. We found that security-related discussions account for approximately 10% of all discussions on GitHub, and they encompass more negative emotions than other discussions. These findings confirm the anecdotal evidence that implementing application security can often lead to frustration and anger among developers, and is a source of tension to the overall project atmosphere.

However, although supported both by statistical testing and a case study, these results should be taken with a grain of salt. First, our results could have been affected by the unbalance in sample sizes for security-related vs. non-security-related discussions. Second, our keywords-based approach to do identification of security-related comments and discussions did generate false positives (discussions mislabeled as security-related), especially in the lower confidence tiers. Third, we have chosen to perform sentiment analysis using a single tool (NLTK), while alternatives exist. It is not clear to which extent different tools would yield different results. Finally, due to limitations in the way the Challenge dataset was constructed, comments longer than 256 characters were truncated. The missing comment parts could have affected the sentiment levels mined.

6. REFERENCES

[1] C. Clifton, R. Cooley, J. Rennie, *TopCat: data mining for topic identification in a text corpus*, Knowledge and Data Engineering, IEEE Transactions on, vol.16, no.8, pp.949-964, Aug. 2004

[2] G. Gousios, *The GHTorrent dataset and tool suite*, In MSR, pp.233-236, IEEE, 2013

[3] G. Gousios, B. Vasilescu, A. Serebrenik, A. Zaidman, *Lean GHTorrent: GitHub data on demand*, In 11th MSR, IEEE, 2014

[4] M. Howard, S. Lipner, *The Security Development Lifecycle*, Microsoft Press, May 2006

[5] R. Kissel, *Glossary of Key Information Security Terms*, NIST Interagency/Internal Report (NISTIR) - 7298rev2, 5 Jun. 2013

[6] *List of 20+ Sentiment Analysis APIs*, `http://blog.mashape.com/post/48757031167/list-of-20-sentiment-analysis-apis`, Accessed on February 4, 2014

[7] D. Mitropoulos, G. Gousios, D. Spinellis, *Measuring the Occurrence of Security-Related Bugs through Software Evolution*, Informatics (PCI), 2012 16th Panhellenic Conference on, pp. 117-122, 5-7 Oct. 2012

[8] Python NLTK Demos and Natural Language Text Processing APIs, `http://text-processing.com/`, Accessed on February 4, 2014

[9] *Security within a development lifecycle*, `http://www.blackhat.com/presentations/bh-europe-04/bh-eu-04-elio.pdf`, Accessed on February 4, 2014

[10] S. Sista, R. Schwartz, T. R. Leek, J. Makhoul, *An algorithm for unsupervised topic discovery from broadcast news stories*, In HLT '02. Morgan Kaufmann Publishers Inc., pp. 110-114, 2002

[11] B. Vasilescu, V. Filkov, A. Serebrenik, *StackOverflow and GitHub: associations between software development and crowdsourced knowledge*, In 2013 ASE/IEEE International Conference on Social Computing, pp.188-195, 2013.

[12] B. Vasilescu, A. Serebrenik, M. G. J. van den Brand, *The Babel of software development: Linguistic diversity in Open Source*, In SocInfo (LNCS 8238), pp.391-404, Springer, 2013

[13] C. Wartena, R. Brussee, *Topic Detection by Clustering Keywords*, Database and Expert Systems Application, 2008. DEXA '08. 19th International Workshop on, pp.54-58, 1-5 Sept. 2008