A Large-scale Study of Security Vulnerability Support on Developer Q&A Websites

Triet Huynh Minh Le
The University of Adelaide
Adelaide, Australia
triet.h.le@adelaide.edu.au

Roland Croft
The University of Adelaide
Adelaide, Australia
Cyber Security Cooperative
Research Centre, Australia
roland.croft@adelaide.edu.au

David Hin
The University of Adelaide
Adelaide, Australia
Cyber Security Cooperative
Research Centre, Australia
david.hin@adelaide.edu.au

M. Ali Babar
The University of Adelaide
Adelaide, Australia
Cyber Security Cooperative
Research Centre, Australia
ali.babar@adelaide.edu.au

ABSTRACT

Context: Developers usually seek solutions to addressing Security Vulnerabilities (SVs) on developer Question and Answer (Q&A) websites. However, there is still little known about these SV-specific discussions on different Q&A sites. **Objective**: We present a large-scale empirical study to understand developers' SV discussions and how these discussions are being supported by Q&A sites. **Method**: We use topic modeling to uncover the topics of 71,329 curated SV posts from two large Q&A sites, namely Stack Overflow (SO) and Security StackExchange (SSE). We then analyze the popularity, difficulty, and level of expertise for each topic. We also perform a qualitative analysis to identify the types of solutions to SV-related questions. **Results**: We identify 13 main SV discussion topics. Many topics do not follow the distributions and trends in expert-based security sources, e.g., Common Weakness Enumeration (CWE) and Open Web Application Security Project (OWASP). We also discover that SV discussions attract more experts to answer than many other domains, but some difficult SV topics (e.g., Vulnerability Scanning Tools) still receive quite limited support from experts. Moreover, we identify seven key types of answers given to SV questions, in which SO often provides code and instructions, while SSE usually gives experience-based advice and explanations. **Conclusion**: Our findings provide support for researchers and practitioners to effectively acquire, share and leverage SV knowledge on Q&A sites.

CCS CONCEPTS

• **Security and privacy → Software security engineering**.

KEYWORDS

Security Vulnerability, Natural Language Processing, Topic Modeling, Mining Software Repositories, Empirical Study

ACM Reference Format:
Triet Huynh Minh Le, Roland Croft, David Hin, and M. Ali Babar. 2021. A Large-scale Study of Security Vulnerability Support on Developer Q&A Websites. In *Evaluation and Assessment in Software Engineering (EASE 2021), June 21–23, 2021, Trondheim, Norway.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3463274.3463331

1 INTRODUCTION

It is important to constantly track and resolve Security Vulnerabilities (SVs) to ensure the availability, confidentiality and integrity of software systems [12]. Developers can seek information for resolving SVs from sources verified by security experts such as Common Weakness Enumeration (CWE), National Vulnerability Database (NVD) and Open Web Application Security Project (OWASP). However, these expert-based SV sources do not provide any mechanisms for developers to promptly ask and answer questions about issues in implementing/understanding the reported SV solutions/concepts. On the other hand, developer Questions and Answer (Q&A) websites contain a plethora of such SV-related discussions. Stack Overflow (SO) and Security StackExchange[1] (SSE) contain some of the largest number of SV-related discussions among developer Q&A sites, with contributions from millions of users [21].

The literature has analyzed different aspects of discussions on Q&A sites, but there is still no investigation of how SO and SSE are supporting SV-related discussions. Specifically, the main concepts [35], the top languages/technologies and user demographics [6], as well as user perceptions and interactions [23] of general security discussions on SO have been studied. However, from our analysis (see section 3.2), only about 20% of the available SV posts on SO were investigated in the previous studies, limiting a thorough understanding about SV topics (developers' concerns when tackling SVs in practice) on Q&A sites. Moreover, the prior studies only focused on SO, and little insight has been given into the support of SV discussions on different Q&A sites that can affect the choice of a suitable site (e.g., SO vs. SSE) to discuss certain SV topics.

To fill these gaps, we conduct a large-scale empirical study using 71,329 SV posts on SO and SSE. Specifically, we use Latent Dirichlet Allocation (LDA) [7] topic modeling and qualitative analysis to answer the following four Research Questions (RQs) that measure the support of Q&A sites for different SV discussion topics:
RQ1: What are SV discussion topics on Q&A sites?
RQ2: What are the popular and difficult SV topics?
RQ3: What is the level of expertise for supporting SV questions?
RQ4: What types of answers are given to SV questions?
Our findings to these RQs can help raise developers' awareness of common SVs and enable them to seek solutions to such SVs more effectively on Q&A sites. We also identify the areas to which experts

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
EASE 2021, June 21–23, 2021, Trondheim, Norway
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-9053-8/21/06…$15.00
https://doi.org/10.1145/3463274.3463331

[1] https://security.stackexchange.com/

can contribute to assist the secure software engineering community. Furthermore, we release a large dataset of SV discussions on Q&A sites for replication and future work at [20].

2 RELATED WORK

2.1 Topic Modeling on Q&A Websites

Q&A websites such as SO and SSE contain a large number of discussion posts. LDA [7] has been frequently used to extract the taxonomy/topics of various software-related domains from such posts. In 2014, a seminal work of Barua et al. [5] discovered the topics of all SO posts. They also found that LDA could find more consistent topics than the tags on SO. Many subsequent studies have leveraged LDA to investigate discussions of specific domains, such as general security [35], concurrent computing [2], mobile computing [30], big data [3], machine learning [4] and deep learning [13]. Among the aforementioned studies, Yang et al. [35] is the closest to our work. However, our work is still fundamentally different from this previous study. Despite sharing a similar security context to Yang et al. [35], we focus specifically on the flaws of security implementation/features since exploitation of such flaws can disclose user's data and interrupt system operations. Moreover, we consider the content of both questions and answers of SV posts on two Q&A sites (SO and SSE) rather than just questions on SO as in [35]. This gives more in-depth insights into how different Q&A sites are supporting on-going SV discussions. Detailed discussion on these differences is given in section 5.1.

2.2 SV Analytics Using Open Sources

SV analytics have long been of interest to researchers. Shahzad et al. [33] conducted a large-scale study on the characteristics (e.g., risk metrics, exploitation, affected vendors and products) of reported SVs on NVD. Besides empirical study, there is another active research trend to build prediction models to analyze SVs. Bozorgi et al. [8] used Support Vector Machine to predict the probability and time-to-exploit of SVs. There have been many follow-up studies since then on developing learning-based models (e.g., [14, 22, 31]) to determine various properties of SVs using expert-based SV sources (e.g., CWE and NVD). A recent study [17] leveraged security mentions on social media (i.e., Twitter and Reddit) to forecast the SV-related activities on GitHub. Unlike the above studies, we focus on SV analytics on developer Q&A sites. Several studies (e.g., [27, 28]) analyzed SVs of different programming languages using code snippets on SO. Contrary to these studies, we do not limit our investigation to any specific programming language, and we consider every type of SV-related posts, not just the ones with code snippets.

3 RESEARCH METHOD

3.1 Research Questions

We investigated four RQs to study the support of Q&A websites for SV-related discussions. To answer these RQs, we retrieved 71,329 SV posts from a general Q&A website (SO) and a security-centric one (SSE) using both the tags and content of posts (see section 3.2).

RQ1: What are SV discussion topics on Q&A sites?

Motivation: To provide fine-grained information about the support of SO and SSE for different types of SV discussions, we first needed to identify the taxonomy of commonly discussed SV topics in RQ1. Our taxonomy does not aim to replace the existing ones provided by experts (e.g., CWE or OWASP), but rather helps to highlight the important aspects of SVs from developers' perspective.

Method: Following the standard practice in [2–5, 13, 30, 35], RQ1 used Latent Dirichlet Allocation (LDA) [7] topic modeling technique (see section 3.3) to select SV discussion topics based on the titles, questions and answers of SV posts on both SO and SSE. LDA is commonly used since it can produce topic distribution (assigning multiple topics with varying relevance) for a post, providing more flexibility/scalability than manual coding. We also used the topic share metric [5] in Eq. (1) to compute the proportion (share_i) of each SV topic and their trends over time.

$$\text{share}_i = \frac{1}{N} \sum_{p \in D} \text{LDA}(p,\ \text{T}_i) \tag{1}$$

where p, D and N are a single SV post, the list of all SV posts and the number of such posts, respectively; T_i is the i^{th} topic and LDA is the trained LDA model.

RQ2: What are the popular and difficult SV topics?

Motivation: After the SV topics were identified, RQ2 identified the popular and difficult topics on Q&A websites. The results of RQ2 can aid the selection of a suitable (i.e., more popular and less difficult) Q&A site for respective SV topics.

Method: To quantify the topic popularity, we used four metrics from [2, 3, 30, 35], namely the average values of (*i*) views, (*ii*) scores (upvotes minus downvotes), (*iii*) favorites and (*iv*) comments. Intuitively, a more popular topic would attract more attention (views), interest (scores/favorites) and activities (comments) per post from users. We also obtained the geometric mean of the popularity metrics to produce a more consistent result across different topics. Geometric mean was used instead of arithmetic mean here since the metrics could have different units/scales. To measure the topic difficulty, we used the three metrics from [2, 3, 30, 35]: (*i*) percentage of getting accepted answers, (*ii*) median time (hours) to receive an accepted answer since posted, and (*iii*) average ratio of answers to views. A more difficult topic would, on average, have a lower number of accepted answers and ratio of answers to views, but a higher amount of time to obtain accepted answers. To achieve this, we took reciprocals of the difficulty metrics (*i*) and (*iii*) so that a more difficulty topic had a higher geometric mean of the metrics.

RQ3: What is the level of expertise to answer SV questions?

Motivation: RQ3 checked the expertise level available on Q&A websites to answer SV questions, especially the ones of difficult topics. The findings of RQ3 can shed light on which topic may require more attention from experts. Note that experts here are users who frequently contribute helpful (accepted) answers/knowledge.

Method: We measured both user's general and specific expertise for SV topics on Q&A sites. For the general expertise, we leveraged the commonly used metric, the reputation points [15, 27, 28], of users who got accepted answers since reputation is gained through one's active participation and appreciation from the Q&A community in different topics. Higher reputation received for a topic usually implies that the questions of that topic are of more interest to experts. Similar to [27], we did not normalize the reputation by user's participation time since reputation may not increase linearly, e.g., due to users leaving the sites. However, reputation is not specific to any topic; thus it does not reflect whether a user is experienced with a topic. Hence, we represented developers' specific expertise with the SV content in their answers on Q&A sites. This was inspired by Dey et al.'s findings that developers' expertise/knowledge could be expressed through their generated content [11]. We determined a user's expertise in SV topics using the topic distribution generated by LDA applied to the concatenation of all answers to SV questions given by that user. The specific expertise of an SV topic (see Eq. (2)) was then the total correlation between LDA outputs of the current topic in SV questions and the specific expertise of users who got the respective accepted answers. The correlation of LDA values could reveal the knowledge (SV topics) commonly used to answer questions of a certain (SV) topic [5].

$$\begin{gathered} Specific\_Expertise_i = \sum_{p \in D} \text{LDA}(Q(p),\ \text{T}_i) \odot \text{LDA}(K(U_{\text{Accept.}})) \\ K(U_{\text{Accept.}}) = A^1_{U_{\text{Accept.}}} + A^2_{U_{\text{Accept.}}} + \ldots + A^k_{U_{\text{Accept.}}} (k = \left|A_{U_{\text{Accept.}}}\right|) \end{gathered} \tag{2}$$

where D is the list SV posts and T_i is the i^{th} topic, while $Q(p)$ and $K(U_{\text{Accept.}})$ are the question content and SV knowledge of the user $U_{\text{Accept.}}$ who gave the accepted answer of the post p, respectively. $\odot$ is the topic-wise multiplication. $\left|A_{U_{\text{Accept.}}}\right|$ is all SV-related answers given by user $U_{\text{Accept.}}$. Note that we only considered posts with accepted answers to make it consistent with the general expertise. Specifically, for each question, we first extracted the user that gave the accepted answer ($U_{\text{Accept.}}$). We then gathered all answers, not necessarily accepted, of that user in SV posts ($\left|A_{U_{\text{Accept.}}}\right|$). Such answer list was the SV knowledge of $U_{\text{Accept.}}$ ($K(U_{\text{Accept.}})$). Finally, we computed the LDA topic-wise correlation between the topic T_i in the current SV question ($\text{LDA}(Q(p),\ \text{T}_i)$) and the user knowledge ($\text{LDA}(K(U_{\text{Accept.}}))$) to determine the specific expertise for post p.

RQ4: What types of answers are given to SV questions?

Motivation: RQ4 extended RQ2 in terms of the solution types given if an SV question is satisfactorily answered. We do not aim to provide solutions for every single SV. Rather, we analyze and compare the types of support for different SV topics on SO and SSE, which can guide developers to a suitable site depending on their needs (e.g., looking for certain artefacts). To the best of our knowledge, we are the first to study answer types of SVs on Q&A sites.

Method: We employed an open coding procedure [32] to inductively identify answer types. LDA is *not suitable* for this purpose since it relies on word co-occurrences to determine categories. In contrast, the same type of solutions may not share any similar words. In RQ4, we only considered the posts with accepted answer to ensure the high quality and relevance of the answers. We then used stratified sampling to randomly select 385 posts (95% confidence level with 5% margin error [10]) each from SO and SSE to categorize the answer types. Stratification ensured the proportion of each topic was maintained. Following [9], two of the authors first conducted a pilot study to assign initial codes to 30% of the selected posts and grouped similar codes into answer types. For example, the accepted answers of SO posts 32603582 (PostgreSQL code), 20763476 (MySQL code) and 12437165 (Android/Java code) were grouped into *Code Sample* category. Similarly to [34], we also allowed one post to have more than one answer type. Two same authors then independently assigned the identified categories to the remaining 70% of the posts. The Kappa inter-rater score (κ) [26] was 0.801 (strong agreement), showing the reliability of our coding. The third author involved to discuss and resolve the disagreements. We also correlated the answer types with the question types on Q&A sites [34].

3.2 Security Vulnerability Post Collection

To study the support of Q&A sites for SV discussions, we proposed a workflow (see Fig. 1) to obtain, to the best of our knowledge, the largest and most contemporary set of SV posts on both SO and SSE. We used *tag-based* and *content-based filtering* to retrieve SV posts based on their tags and content of other parts (i.e., title, body and answers), respectively. We considered a post to be related to SV when it mainly discussed a security flaw and/or exploitation/testing/fixing of such flaw to compromise a software system (e.g., SO post 29098142[2]). A post was not SV-related if it just asked how to implement/use a security feature (e.g., SO post 685855) without any explicit mention of a flaw. All the tags, keywords and posts collected were released at [20].

Tag-based filtering. We had a *vulnerability* tag on SSE but not on SO to obtain SV-related posts, and a *security* tag on SO used by [35] was too coarse-grained for the SV domain. Many posts with *security* tag did not explicitly mention SV (e.g., SO post 65983245 about privacy or SO post 66066267 about how to obtain security-relevant commits). Therefore, we used Common Weakness Enumeration (CWE), which contains various SV-related terms, to define relevant SV tags. However, the full CWE titles were usually long and uncommonly used in Q&A discussions. For example, the fully-qualified CWE name of SQL-injection (CWE-89) is "*Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')*", which appeared only nine times on SO and SSE. Therefore, we needed to extract shorter and more common terms from the full CWE titles. We adopted Part-of-Speech (POS) tagging for this purpose, in which we only considered consecutive (n-grams of) verbs, nouns and adjectives since most of them conveyed the main meaning of a title. For instance, we obtained the following 2-grams for CWE-89: *improper neutralization*, *special elements*, *elements used*, *sql command*, *sql injection*. We obtained 2,591 n-gram ($1 \leq \text{n} \leq 3$) terms that appeared at least once on either SO or SSE. To ensure the relevance of these terms, we manually removed the irrelevant terms without any specific SV context (e.g., *special elements*, *elements used* and *sql command* in the above example). We found 60 and 63 SV-related tags on SO and SSE that matched the above n-grams, respectively. We then obtained the initial set_{tag} of SV posts that had at least one of these selected tags.

Content-based filtering. As recommended by some recent studies [16, 21], tag-based filtering was not sufficient for selecting posts due to wrong tags (e.g., non SV-post 38539393 on SO with *stackoverflow* tag) or general tags (e.g., SV post 15029849 on SO with only *php* tag). Therefore, as depicted in Fig. 1, we customized content-based filtering, which was based on keyword matching, to refine the set_{tag} obtained from the tag-based filtering step and select missing SV posts that were not associated with SV tags. First, we presented the up-to-date list of 643 SV keywords for matching [20].

[2] stackoverflow.com/questions/29098142 (postid: 29098142). SSE format is security.stackexchange.com/questions/postid. Posts in our paper follow these formats.

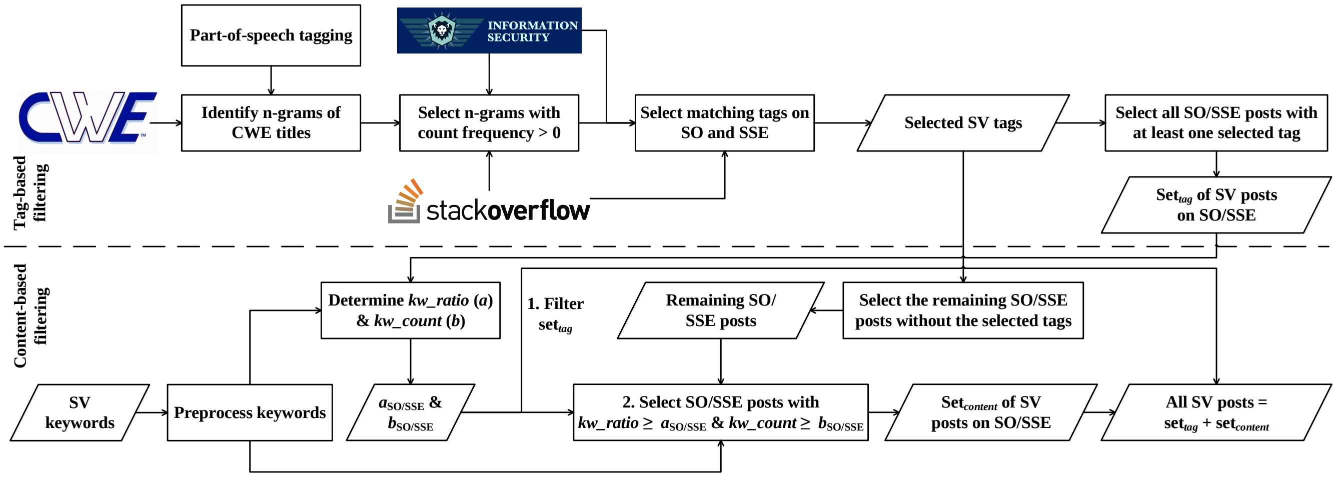

Part-of-speech tagging
INFORMATION SECURITY
Identify n-grams of CWE titles
Select n-grams with count frequency > 0
Select matching tags on SO and SSE
Selected SV tags
Select all SO/SSE posts with at least one selected tag
Set_tag of SV posts on SO/SSE
Tag-based filtering
Content-based filtering
Determine kw_ratio (a) & kw_count (b)
1. Filter set_tag
Remaining SO/ SSE posts
Select the remaining SO/SSE posts without the selected tags
SV keywords
Preprocess keywords
a_SO/SSE & b_SO/SSE
2. Select SO/SSE posts with kw_ratio ≥ a_SO/SSE & kw_count ≥ b_SO/SSE
Set_content of SV posts on SO/SSE
All SV posts = set_tag + set_content

Figure 1: Workflow of retrieving posts related to SV on Q&A websites using tag-based and content-based filtering heuristics.

Table 1: Content-based thresholds ($a_{SO/SSE}$ & $b_{SO/SSE}$) for the two steps of the content-based filtering as shown in Fig. 1.

Threshold	Stack Overflow (SO)		Security StackExchange (SSE)	
	Step 1	Step 2	Step 1	Step 2
a	1	3	2	3
b	0.011	0.017	0.017	0.025

Table 2: The obtained SV posts using our tag-based and content-based filtering heuristics.

	Stack Overflow (SO)	Security StackExchange (SSE)	SO + SSE
Set$_{tag}$	46,212	9,677	55,889
Set$_{content}$	12,660	2,780	15,440
Total	58,872	12,457	71,329

These keywords were preprocessed with stemming and augmented with American/British spellings, space/hyphen to better handle various types of (mis-)spellings/plurality. For instance, we considered the following variants: *input(-)sanitization/sanit/sanitisation/sanitis* for "*input sanitization*". Similar to [16, 21], we also performed exact matching for three-character keywords and subword matching for longer ones to reduce false positives. Subsequently, for each set$_{tag}$ (SO and SSE) obtained in the tag-based filtering step, we computed two content-based metrics (see Eq. (3)) [16, 21]: *kw_count* and *kw_ratio*, denoting the count and appearance proportion of SV keywords in a post, respectively. *Kw_count* ensured diverse SV-related content in a post, while *kw_ratio* increased the confidence that these relevant words did not appear by chance.

$$kw\_count_p = \left|SV\_KWs_p\right|, \quad kw\_ratio_p = \frac{\left|SV\_KWs_p\right|}{\left|Words_p\right|} \tag{3}$$

where $\left|SV\_KWs_p\right|$ and $\left|Words_p\right|$ are the numbers of SV keywords and total number of words in post p, respectively.

Based on the post content and human inspection, the thresholds $a_{SO/SSE}$ and $b_{SO/SSE}$ for filtering set$_{tag}$ (step 1) as well as selecting extra posts based on their content (step 2) were found, as given in Table 1. Using these thresholds, we obtained set$_{tag}$ and set$_{content}$ of SV posts on SO and SSE, respectively, as shown in Fig. 1.

SV datasets and validation. As of June 30, 2020, we retrieved 20,062,329 and 58,912 posts from SO and SSE, respectively, using Stack Exchange Data Explorer. We then applied the tag-based and content-based filtering steps in Fig. 1 and obtained *71,329* SV posts (see Table 2) in total including 55,883 and 15,436 ones for set$_{tag}$ and set$_{content}$, respectively. We manually validated four different sets of SV posts, i.e., set$_{tag}$ and set$_{content}$ for SO and SSE, respectively. Specifically, we randomly sampled 385 posts (significant size [10]) in each set for two authors to examine independently.

For set$_{tag}$, we disagreed on 7/770 cases and only two posts were not related to SV. The main issue was still the incorrect tag assignment (e.g., SSE post 175264 was about dll injection but tagged with *malware*[3]), though this issue had been significantly reduced by the content-based filtering. For set$_{content}$, the relevance of the posts was very high as there was no discrepant case.

Our SV dataset was only 20% overlapping with the existing security dataset [35], implying that there were significant differences in the nature of the two studies. Note that we followed the settings in [35] to retrieve the updated security posts from the same SO data we used in our study. We also reported the top tags of SV posts (see Table 3) and compared them with the ones of security posts [35] and a subset of all the posts containing an equal number of posts to the SV posts on SO and SSE. SV posts were associated with many SV-related tags (e.g., *memory-leaks*, *malware*, *segmentation-fault*, *xss*, *exploit* and *penetration-test*). Conversely, security posts were tagged with general terms that may not explicitly discuss security flaws such as *encryption*, *authentication* and *passwords*. The tags of general posts were mostly programming languages on SO and general security terms on SSE. These findings highlight the importance of obtaining SV-specific posts instead of reusing the security posts to study the support of Q&A sites for SV-related discussions.

[3] This post was short yet contained many SV keywords (e.g., "*injection*" and "*hijack*"), resulting in high kw_count and kw_ratio of the content-based filtering.

Table 3: Top-5 tags of SV, security and general posts on SO and SSE (in parentheses).

No.	SV posts	Security posts	General posts
1	memory-leaks (malware)	security (encryption)	javascript (encryption)
2	segmentation-fault (web-application)	encryption (tls)	java (tls)
3	php (xss)	php (authentication)	python (authentication)
4	c (exploit)	java (passwords)	c# (passwords)
5	security (penetration-test)	cryptography (web-application)	php (certificates)

3.3 Topic Modeling with LDA

Following the common practice of the existing work (e.g., [3, 5, 35]), we extracted the topics of the identified SV-related posts on both SO and SSE using Latent Dirichlet Allocation (LDA) [7].

Preprocessing of SV posts. Following the previous practices of [2, 35], we first removed the HTML tags and code snippets in each post as these elements were not informative for topic modeling. We also converted the text to lowercase, removed punctuations, and then eliminated stop words and performed stemming (reducing a word to its root form) to avoid irrelevant and multi-form words.

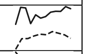
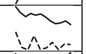
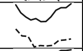
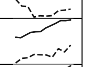
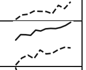
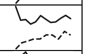
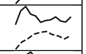
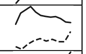
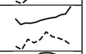
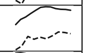
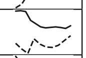
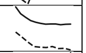
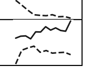
Topic modeling with LDA. We applied LDA to the title, question body and all answers of each Q&A post. Regarding the number of topics (k) of LDA, we examined an inclusive range from 2 to 80, with an increment of one topic at a time. As suggested in [2, 3, 5, 30], alongside k, we also tried different values of α ($1/k$ or $50/k$) and β (0.01 or same as α) hyperparameters to optimize the performance of LDA. α controls the sparsity of the topic-distribution per post and β determines the sparsity of the word-distribution per topic. For each tuple of (k, α and β), we ran LDA with 1,000 iterations, then evaluated the coherence metric [29] of the identified topics. Coherence metric has been recommended by many previous studies (e.g., [1, 37]) to select the optimal number of LDA topics since it usually highly correlates with human understandability. Topic coherence is the average correlation between pairs of words that appear in the same topic. The higher value of the coherence metric means the more coherent content of the posts within a same topic. To avoid insignificant topics like [5], we only considered topics with a probability of at least 0.1 in a post. We manually read the top-20 most frequent words and 15 random posts of each topic per site (SO/SSE) obtained by the trained LDA models to label the name of that topic as done in [2, 3]. The LDA model with most relevant set of topics would be used for answering the four RQs.

4 RESULTS

4.1 RQ1: What are SV Discussion Topics on Q&A Sites?

Following the procedure in section 3.3, we identified *13* SV topics (see Table 4) on SO and SSE using the optimal LDA model with $\alpha = \beta = 0.08$. We found LDA models having from 11 to 17 topics produced similar coherence metrics. Three of the authors manually examined these cases, as in [1]. Duplicate and/or platform-specific topics (e.g., web and mobile) appeared from 14 topics, making the taxonomy less generalizable. 11 and 12 topics also had high-level topics (e.g., combining XSS and CSRF). Thus, 13 was chosen as the optimal number of SV topics. All the terms/posts of each SV topic can be found at [20]. We describe each topic hereafter with example SO/SSE posts. We examined 15 random posts per topic per site. If we identified some common patterns of discussions (e.g., attack vectors or assets) on a site, we would extract another 15 random posts of the respective site to confirm our observations. If a pattern was no longer evident in the latter 15 posts, we would not report it.

Table 4: SV topics on SO and SSE identified by LDA along with their proportions and trends over time. Notes: The topic proportions on SSE are in parenthesis. The trends of SO are the top solid sparklines, while the trends of SSE are the bottom dashed sparklines. Unit of proportion: %.

Topic Name	Proportion	Trend
Malwares (T1)	1.39 (8.18)	
SQL Injection (T2)	11.0 (4.17)	
Vulnerability Scanning Tools (T3)	5.42 (3.15)	
Cross-site Request Forgery (CSRF) (T4)	9.49 (5.09)	
File-related Vulnerabilities (T5)	2.88 (3.24)	
Synchronization Errors (T6)	3.79 (0.47)	
Encryption Errors (T7)	1.82 (7.81)	
Resource Leaks (T8)	10.6 (0.42)	
Network Attacks (T9)	1.37 (8.79)	
Memory Allocation Errors (T10)	**21.6** (2.82)	
Cross-site Scripting (XSS) (T11)	7.73 (8.09)	
Vulnerability Theory (T12)	10.7 (**33.7**)	
Brute-force/Timing Attacks (T13)	1.08 (1.28)	

Malwares (T1). This topic referred to the detection and removal of malicious software. T1 posts on SO were usually about malwares in content management systems such as Wordpress or Joomla (e.g., post 16397854: "*How to remove wp-stats malware in wordpress*" or post 11464297: "*How to remove .htaccess virus*'). In contrast, SSE often discussed malwares/viruses coming from storage devices such as SSD (e.g., post 227115: "*Can viruses of one ssd transfer to another ssd?*") or USB (e.g., post 173804: "*Can Windows 10 bootable USB drive get infected while trying to reinstall Windows?*").

SQL Injection (T2). This topic concerned tactics to properly sanitize malicious inputs that could modify SQL commands and pose threats (e.g., stealing or changing data) to databases in various programming languages (e.g., PHP, Java, C#). A commonly discussed tactic was to use prepared statements, which also helped increase the efficiency of query processing. For example, developers asked questions like "*How to parameterize complex oledb queries?*" (SO post 9650292) or "*How to make this code safe from SQL injection and use bind parameters*" (SSE post 138385).

Vulnerability Scanning Tools (T3). This topic was about issues related to tools for automated detection/assessment of potential SVs in an application. Discussions of T3 mentioned different tools, and OWASP ZAP was a commonly discussed one. For example, post 62570277 on SO discussed "*Jenkins-zap installation failed*", while post 126851 on SSE asked "*How do I turn off automated testing in OWASP ZAP?*" One possible explanation is that OWASP ZAP is a free and easy-to-use tool for detecting and assessing SVs that appear in the well-known top-10 OWASP list for web applications.

Cross-site Request Forgery (CSRF) (T4). This topic contained discussions on proper setup and configuration of web application frameworks to prevent CSRF SVs. These SVs could be exploited to send requests to perform unauthorized actions from an end user that a web application trusts. Discussions covered various issues in implementing different CSRF prevention techniques recommended by OWASP.[4] Some commonly discussed techniques were anti-CSRF token (e.g., SO post 59664094: "*Why Laravel 4 CSRF token is not working?*"), double submit cookie (e.g., SSE post 203996: "*What is double submit cookie? And how it is used in the prevention of CSRF attack?*"), and SameSite cookie attribute (e.g., SO post 41841880: "*What is the benefit of blocking cookie for clicked link? (SameSite=strict)*").

File-related Vulnerabilities (T5). Discussions of this topic were about SVs in files that could be exploited to gain unauthorized access. The common SV types were Path/Directory Traversal via Symlink (e.g., SSE post 165860: "*Symlink file name - possible exploit?*"), XML External Entity (XXE) Injection (e.g., SO post 51860873: "*Is SAXParserFactory susceptible to XXE attacks?*"), and Unrestricted File Upload (e.g., SSE post 111935: "*Exploiting a PHP server with a .jpg file upload*"). These SVs usually occurred for Linux-based systems, suggesting that Linux is more popular for servers.

Synchronization Errors (T6). This topic involved SVs produced through errors in synchronization logic (usually related to threads), which could slow down system performance. Some common SV types being discussed were deadlocks (e.g., SO post 38960765: "*How to avoid dead lock due to multiple oledb command for same table in ssis*") and race conditions (e.g., SSE post 163209: "*What's the meaning of 'the some sort of race condition' here?*").

Encryption Errors (T7). This topic included cryptographic issues leading to falsified authentication or retrieval of sensitive data, e.g., Man-in-the-middle (MITM) attack. Many posts discussed public/private keys for encryption/decryption, especially using SSL/TLS certificates to defend against MITM attacks (attempts to steal information sent between browsers and servers). Some example discussions are post 23406005 on SO ("*Man In Middle Attack for HTTPS*") or post 105773 on SSE ("*How is it that SSL/TLS is so secure against password stealing?*"). This may imply that many developers are still not familiar with these certificates in practice.

Resource Leaks (T8). This topic considered SVs arising from improper releases of unused memory which could deplete resources and decrease system performance. Many discussions of T8 were about memory leaks in mobile app development. Issues were usually related to Android (e.g., SO post 58180755: "*Deal with Activity Destroying and Memory leaks in Android*") or IOS (e.g., SO post 47564784: "*iOS dismissing a view controller doesn't release memory*").

Network Attacks (T9). This topic discussed attacks carried out over an online computer network, e.g., Denial of Service (DoS) and IP/ARP Spoofing, and potential mitigations. These network attacks directly affected the availability of a system. For instance, SSE post 86440 discussed "*VPN protection against DDoS*" or SO post 31659468 asked "*How to prevent ARP spoofing attack in college?*".

Memory Allocation Errors (T10). T10 and T8 were related to memory issues, but T10 did not consider memory release. Rather, this topic focused on SVs caused by accessing or using memory outside of what allocated that could be exploited to access restricted memory location or crash an application. In this topic, segmentation faults (e.g., SO post 31260018: "*Segmentation fault removal duplicate elements in unsorted linked list*") and buffer overflows (e.g., SSE post 190714: "*buffer overflow 64 bit issue*") were commonly discussed.

Cross-site Scripting (XSS) (T11). This topic mentioned tactics to properly neutralize user inputs to a web page to prevent XSS attacks. These attacks could exploit users' trust in web servers/pages to trick them to execute malicious scripts and perform unwanted actions. XSS (T11) and CSRF (T4) are both client-side SVs, but XSS is more dangerous since it can bypass all countermeasures of T4.[4] On SO and SSE, discussions covered all three types of XSS: (*i*) reflected XSS (e.g., SSE post 57268: "*How does the anchor tag (<a>) let you do an Reflected XSS?*"), (*ii*) stored/persistent XSS (e.g., SO post 54771897: "*How to defend against stored XSS inside a JSP attribute value in a form*"), and (*iii*) DOM-based XSS (e.g., SO post 44673283: "*DOM XSS detection using javascript(source and sink detection)*").

Vulnerability Theory (T12). This topic focused on theoretical/social aspects and best practices in the SV life cycle. Many posts compared different SV-related terminologies, e.g., SSE post 103018 asked about "*In CIA triad of information security, what's the difference between confidentiality and availability?*" or SO post 402936 discussed "*Bugs versus vulnerabilities?*". Several other posts asked about internal SV reporting process (e.g., SO post 3018198: "*How best to present a security vulnerability to a web development team in your own company?*") or public SV disclosure policy (e.g., SSE post: "*How to properly disclose a security vulnerability anonymously?*").

Brute-force/Timing Attacks (T13). T13 and T7 both exploited cryptographic flaws, but these two topics used different attack vectors/methods. T7 focused on MITM attacks, while T13 was about attacks making excessive attempts or capturing the timing of a process to gain unauthorized access. Some example posts of T13 are SO post 3009988 ("*What's the big deal with brute force on hashes like MD5*") or SSE post 9192 ("*Timing attacks on password hashes*").

Proportion and Evolution of SV Topics. We analyzed the proportion (share metric in Eq. (1)) and the evolution trend of SV topics from their inception on SO (2008) and SSE (2010) to 2020 (see Table 4). The topic patterns and dynamics of SO were different from those of SSE. Specifically, Memory Allocation Errors (T10) had the greatest number of posts on SO, while Vulnerability Theory (T12) had the largest proportion on SSE. Apart from XSS (T11) and Brute-force/Timing Attacks (T13), topics with many posts in one source were not common in the other source. Moreover, we discovered three consistent topic trends on both SO and SSE: Malwares (T1) (↗), CSRF (T4) (↗), File-related SVs (T5) (↗) and Vulnerability Theory (T12) (↘). Among them, CSRF had the fastest changing pace. These trends were confirmed significant with p-values < 0.05 using Mann-Kendall non-parametric trend test [24].

[4] https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

Table 5: General expertise in terms of average reputation of each topic on SO and SSE (in parentheses). Notes: The values were normalized by the max and min values of each category. T8 on SSE was excluded since it did not have any accepted answer.

General expertise	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13
Reputation	**0.00** (0.16)	0.93 (0.05)	**0.01** (**0.00**)	0.28 (0.18)	0.43 (0.14)	0.88 (0.04)	0.51 (0.72)	0.49 (–)	0.07 (0.24)	0.62 (0.34)	0.84 (0.22)	0.59 (0.29)	**1.00** (**1.00**)

4.2 RQ2: What are the Popular and Difficult SV Topics on Q&A Sites?

As shown in Fig. 2, the popularity and difficulty of 13 identified SV topics were different between SO and SSE. For conciseness, we only report the geometric means of the popularity and difficulty metrics in this section. The complete values of individual metrics (see section 3.1) can be found at [20].

Topic Popularity. Brute-force/Timing attacks (T13) and Vulnerability Theory (T12) were the top-2 most popular topics. Despite being the most popular topic, T13 only had 1.1% and 1.3% posts on SO and SSE, respectively. Conversely, Memory Allocation Errors (T10) had the most posts on SO (RQ2), but T10 was only the second least popular topic. We found no significant correlation between the topic popularity and share metric with Kendall's Tau correlation test [18] at 95% confidence level. These findings suggest that share metric does not necessarily reflect the topic popularity since it does not consider user's activities on Q&A sites.

Topic Difficulty. The most difficult topics were not popular or associated with many posts, i.e., Vulnerability Scanning Tools (T3) and Network Attacks (T9) on SO as well as T3, Synchronization Errors (T6) and Memory Allocation Errors (T10) on SSE. The high difficulty of T3 on both sites was potentially caused by the low familiarity with a wide array of vendors and tools available for SV detection and assessment [19]. Some topics with many posts (high share metric) like Memory Allocation Errors (T10) and SQL Injection (T2) were the two easiest ones on SO despite being significantly more difficult on SSE. On the contrary, Malwares (T1) and Network Attacks (T9) were more popular yet easier on SSE. These numbers suggest that it may be better to ask the topics T2, T8 (only a few posts on SSE) and T10 on SO to obtain answers faster, while asking T1 and T9 on SSE would be more optimal. However, the topic difficulty did not correlate with either the topic popularity or share metric on both SO and SSE, confirmed using Kendall's Tau test [18] with a confidence level of 95%. With the same confidence level, no significant differences in terms of average topic-wise popularity and difficulty between SO and SSE were recorded using non-parametric Mann-Whitney U-test [25].

4.3 RQ3: What is the Level of Expertise to Answer SV Questions on Q&A Sites?

General Expertise. The average general expertise (reputation) of the accepted answerers in SV posts was 1.3 to 5.8 times higher than those of generic posts [5], general security [35], mobile development [30], concurrency [2], machine learning [4] and deep learning [13]. The higher reputation values were confirmed with p-values < 0.05 (significance level) using non-parametric Mann-Whitney U-test [25]. However, the average percentage of the same users who got accepted answers on both SO and SSE was quite small across topics, i.e., 1% to 18%, implying not much SV knowledge sharing between the two sites. The average topic-wise reputation on SO was higher than that of SSE with a p-value of 0.001 (Mann-Whitney U-test). This might be because SO users could engage in many more posts of different topics (not only security). Table 5 reports the general expertise of 13 SV topics. On SO, Brute-force/Timing Attacks (T13), SQL Injection (T2), Synchronization Errors (T6) and XSS (T11) were the topics that experts focused on the most. On SSE, T13 again and Encryption Errors (T7) were the topics of interest for experts. In contrast, Malwares (T1), Vulnerability Scanning Tools (T3) and Network Attacks (T9) on SO did not attract as much attention from experts. On SSE, T3 was also of the least interest to experts. Overall, experts on Q&A sites tended to favor the SV topics with high popularity and low difficulty, confirmed with p-values < 0.05 using Kendall's Tau correlation test [18].

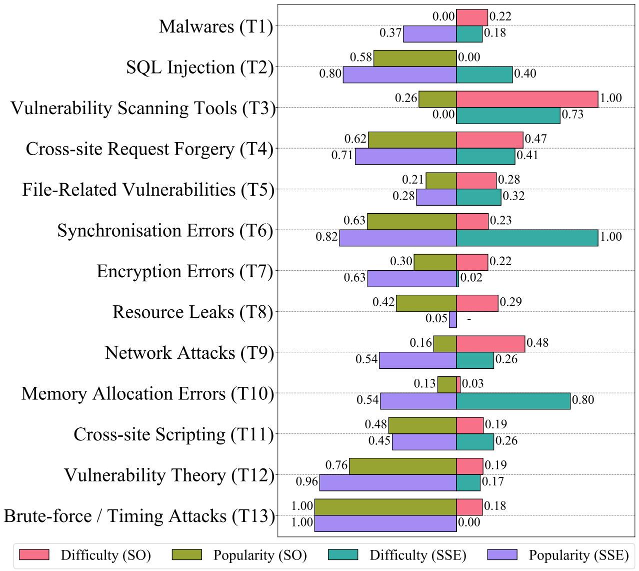
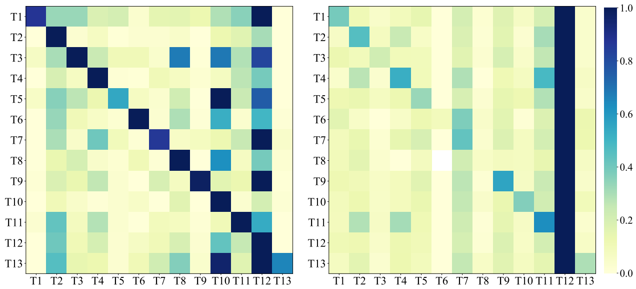


Figure 2: Popularity and difficulty of 13 SV topics on SO and SSE. Notes: The values were normalized by the max and min values of each category. Difficulty of T8 on SSE was excluded since it did not have any accepted answer.

Specific Expertise. Fig. 3 shows the correlation between the pairs of question SV topics and answerers' SV topics (see Eq. (2)). The most frequent answerers' SV topic was Vulnerability Theory (T12). On SSE, frequent answerers for T12 could answer every question topic. On SO, besides T12, users specialized in Memory-related Errors (T8 and T10) also answered the questions of other SV topics. These patterns might be because of the prevalence (RQ2) of topics T8 and T10 on SO as well as T12 on SSE. Conversely, Malwares (T1),



(a) Specific Expertise: Question topics (y-axis) vs. Answerers' topics (x-axis) (SO)

(b) Specific Expertise: Question topics (y-axis) vs. Answerers' topics (x-axis) (SSE)

Figure 3: Topic correlations between SV questions & answerers' SV specific knowledge on SO (a) & SSE (b). Notes: Light to dark color shows weak to strong correlation. Each cell was normalized by max and min values of each question topic.

Network Attacks (T9) and Brute-force/Timing Attacks (T13) on SO as well as Synchronization Errors (T6), Resource Leaks (T8) and T13 on SSE had unique answerers (i.e., users who usually answered questions of only one topic in the SV domain). Furthermore, on SO, most answerers were relevant for each SV topic (dark color on the diagonal in Fig. 3a), but it was not always the case on SSE (see Fig. 3b). Such results suggest that it may be easier to find relevant answerers for different SV topics on SO than on SSE.

4.4 RQ4: What Types of Answers are Given to SV Questions on Q&A Sites?

Our open coding process in RQ4 identified *seven* answer categories of SV discussions on Q&A sites, as shown in Table 6. Some answer types provided experience (DC/Co and Er) or language/platform-specific support (AT, ES and CS), which is hardly found on expert-based security sites (e.g., CWE or OWASP). We correlated such answer types with the question categories of Treude et al. [34]. We found reasonable matches between answer and question types, e.g., (dis)agreeing (DC/Co) with a decision (Decision Help), explaining (Ex) a concept (Conceptual), and providing different solutions to resolve unexpected situations (Discrepancy). Discrepancy and Error were also among the most frequent question types, supporting that our posts were about issues/errors in addressing SVs.

Site-wise answer types. According to Table 6, Action to Take (AT) and External Sources (ES) were the most common answer types on SO and SSE, respectively; whereas, Self-Answer (SA) was the least frequent one on both sites. We also noticed that both sites usually referred to external sources (ES). The most common sources included Wikipedia, other posts (e.g., related answers), GitHub issues/commits, product documentation (e.g., PHP, MySQL and Android) and SV sources (e.g., CVE (Common Vulnerabilities and Exposures), NVD, CWE, OWASP, CVE Details[5] and Exploit-DB[6]). Note that some provided links were unavailable or no longer maintained (e.g., CVE Details). Overall, the answers to SV questions on SO frequently provided detailed instructions (AT) and/or code samples (CS), while SSE tended to share more experience (DC/Co and Ex) to help.

Topic-wise answer types. We extend the site-wise findings to individual topics to enable developers to select the respective site (SO vs. SSE) based on their preferable SV solution types, as shown in Table 7. Specifically, SO highlighted steps (AT) to fix Malwares (T1), Memory Allocation Errors (T10) and XSS (T11) as well as provided code snippets for SQL Injection (T2). On the other hand, SSE gave more relevant sources and explanations for such topics. One may argue that these different answer types were because of the different question types between SO and SSE, but we did not find any such significant differences for these topics. Instead, the fact that SO and SSE had quite different accepted answerers, as shown in RQ3, probably led to such different solution types. There were four topics, namely SV Scanning Tools (T3), Synchronization & Encryption Errors (T6 & T7), Brute-force/Timing Attacks (T13), sharing similar top solutions on both SO and SSE. The remaining topics (T4, T5, T8 and T9) were mostly answered with explanation (Ex) or external sources (ES) on both SO and SSE.

5 DISCUSSION

5.1 SV Discussion Topics on Q&A Sites vs. Existing Security Taxonomies

SV-specific topics and their support on Q&A sites. Compared to Yang et al.'s taxonomy [35], we found related topics: T2, T3, T5, T10, T11 and T13, but we still had the following important differences. Firstly, our topics were emphasized more on security flaws, e.g., issues with encryption/decryption algorithms (T7) than how to implement/use them as in [35]. Secondly, we identified SV-specific topics previously unreported in [35]: Malwares (T1), CSRF (T4), Synchronization Errors (T6), Resource Leaks (T8), Network Attacks (T9) and Vulnerability Theory (T12). These SV topics show the necessity of focusing on SV-specific posts instead of general security ones. Thirdly, unlike [35], we did not consider language-dependent topics (i.e., PHP, Flash, Javascript, Java and ASP.NET), helping our topics be more generalizable (e.g., XSS can occur in both PHP and ASP.NET). Mansooreh et al. [36] also devised a security taxonomy for GitHub issues; however, they focused on security features and implementation instead of any specific SV types. Note that we studied SV posts on both SO and SSE, while the existing studies only used one source of data (SO), enhancing the generalizability of our study. Specifically, we shed light on the differences between SV discussion topics on SO and those on SSE in terms of their proportions (RQ1), popularity/difficulty (RQ2), level of expertise (RQ3) and types of answers (RQ4). Our findings can be leveraged to select suitable site (i.e., more popular/experts, less difficult or having certain answer types) for asking different SV questions.

Disconnection between SV discussions and expert-based SV sources. Two authors manually mapped 13 SV topics with CWEs.[7] We found that only seven of them were overlapping with the two well-known expert-based SV taxonomies: top-25 CWE and top-10 OWASP.[8] The overlapping topics were T2 (SQL Injection), T4 (CSRF), T5 (i.e., Path-traversal and Unrestricted File Upload), T7 (i.e., Improper Certificate Validation), T8 (Resource Leaks), T10 (Memory Allocation Errors) and T11 (XSS). There was no CWE for T3 and T12 since they mainly discussed SV scanning tools and/or

[5] www.cvedetails.com
[6] www.exploit-db.com

[7] Due to limited space, we put our CWE mappings in the reproduction package [20].
[8] https://cwe.mitre.org/top25/ & https://owasp.org/www-project-top-ten/

Table 6: Answer types of SV discussions identified on Q&A websites. Note: An answer can have more than one solution type.

Answer type of SV discussions	Description & Example Posts	Top-3 related question types [34]	Proportion (%) on SO & SSE (in parenthesis)
(Dis-)Confirmation (DC/Co)	Confirm/agree or refute/disagree with a major point or concept made by the asker (e.g., SO post 16155188 or SSE post 31306)	Decision Help, How-to, Conceptual	11.5 (23.7)
Explanation (Ex)	Explain concepts, definitions and "why" to take certain actions (e.g., SO post 53446941 or SSE post 157240)	Decision Help, Conceptual, Discrepancy	14.6 (27.1)
Error (Er)	Point out an error in the source code or another attachment of the initial question (e.g., SO post 29750534 or SSE post 159907)	Discrepancy, Error, How-to	13.0 (2.3)
Action to Take (AT)	Describe step/action(s) ("how-to") to solve a problem (e.g., SO post 22860382 or SSE post 180053)	How-to, Discrepancy, Decision Help	**22.5** (15.4)
External Source (ES)	Provide reference/link to external source(s) (e.g., SO post 445177 or SSE post 107498)	Decision Help, How-to, Discrepancy	18.1 (**28.7**)
Code Sample (CS)	Provide an explicit example of code snippet (e.g., SO post 20763476 or SSE post 36804)	Discrepancy, How-to, Error	16.6 (2.0)
Self-Answer (SA)	Answer given by the same user who submitted the question (e.g., SO post 55784402 or SSE post 100761)	Discrepancy, Error, How-to	3.7 (1.0)

Table 7: Top-1 answer types of 13 SV topics on SO & SSE (in parenthesis). Note: T8 on SSE was excluded since it did not have any accepted answer.

Topic	Top-1 Answer Type	Topic	Top-1 Answer Type
T1	AT/ES (Ex)	T8	ES (–)
T2	CS (ES)	T9	DC/Co/ES/SA (Ex)
T3	ES (ES)	T10	AT (Ex)
T4	ES (DC/Co)	T11	AT (DC/Co)
T5	Ex (ES)	T12	Ex (ES)
T6	Ex/ES (DC/Co/ES)	T13	Ex/ES (Ex)
T7	Ex/ES (Ex)	–	–

socio-technical issues, respectively. Using keyword matching, we further found that only 159 and 71 out of a total of 839 CWEs were mentioned on SO and SSE, respectively; and only 20 and two CWEs appeared more than 100 times on SO and SSE, respectively. Moreover, the fast increase of CSRF (T2) in RQ1 is noteworthy given that this SV type has been removed from the top-10 OWASP since 2013. We observed that many developers were aware of CSRF prevention techniques, but it was not always easy to apply these techniques and/or use built-in CSRF protection of a web framework (e.g., Spring Security) in practice. These results imply that expert-based sources do not always provide details on how to use/configure and implement/debug the reported SV prevention measures/tools in different use cases. The observed strong disconnection in the SV patterns between expert-based sources and discussions on Q&A sites strengthens our motivation to study developers' real-life concerns in addressing SVs.

5.2 Implications of Our Study

Researchers. Different types of SVs are commonly discussed on Q&A sites (RQ1), especially the prevalent, popular and increasing ones like Brute-force/Timing Attacks, Memory/File-related SVs, Malwares and CSRF. Researchers should develop robust detection, assessment and fixing methods for these SV types. Moreover, abundant off-the-shelf testing/scanning tools with complex configurations and different versions have made Vulnerability Scanning Tools one of the most difficult SV topics on Q&A sites (RQ2). This motivates in-depth comparative study to investigate the effectiveness of available tools in different scenarios, especially for the SV types reported in this work. Further, researchers can develop techniques to search relevant users from multiple developer Q&A sites (e.g., SO and SSE) rather than just SO to increase the cross-site knowledge sharing for difficult SV topics with low level of expertise (RQ3).

Practitioners. Many of the SV topics identified in RQ1 have been induced by malicious inputs (i.e., SQL Injection, File-related SVs, CSRF and XSS). Hence, checking for proper input validation and neutralization should be top priorities in security testing. We have also observed that OWASP ZAP has been commonly used to support automated testing for these SV types. However, understanding, using and integrating SV scanning tools are still quite challenging (RQ2/RQ3). This suggests that tool developers/maintainers should further improve the functionalities, documentation and tutorials of their tools, potentially by leveraging SV-related questions/answers on Q&A sites. Moreover, practitioners should not expect to find information about zero-day SVs on SO and SSE. Instead, they can discuss how to identify/fix known SVs on SO and ask theoretical/social questions about SVs on SSE (RQ4). Some links in answers have been also found obsolete in RQ4; thus, practitioners should test and not always trust given solutions on Q&A sites.

5.3 Threats to Validity

Our data collection is the first threat. We might have missed some SV posts, but we followed standard techniques in the literature. It is hard to guarantee 100% relevance of the retrieved posts without exhaustive manual validation, which is nearly impossible with more than 70k posts. However, this threat was greatly reduced since the selected posts were carefully checked by three of the authors.

The identified taxonomies can be another concern. Topic modeling with LDA has been shown effective for processing large amount of textual posts, but there is still subjectivity in labeling the topics. We mitigated this threat by manually examining at least 30 posts per topic and cross-checking with three of the authors. We also performed a similar manual checking for the answer types in RQ4.

The generalizability of our study may be a threat as well. The patterns we found may not be the same for other Q&A sites and domains. However, the reported patterns for SV discussions on SO and SSE were at least confirmed significant using statistical tests with p-values < 0.05. We also released our code and data at [20] for replication and extension to other domains.

6 CONCLUSIONS AND FUTURE WORK

Through a large-scale study of 71,329 posts on SO and SSE, we have revealed the support of SV-focused discussions on Q&A sites. Using LDA, we devised 13 commonly discussed SV topics on Q&A sites. Among these topics, we discovered the popular (e.g., Brute-force/Timing Attacks) and difficult (e.g., Vulnerability Scanning Tools) ones. The expertise for SV topics was high, but the knowledge sharing between the sites was still limited, and some topics (e.g., Vulnerability Scanning Tools) required more attention from experts. We also identified seven answer types for SV questions, in which SO offered more code-based/step-by-step solutions, while SSE provided more explanatory/experience-based replies. Overall, Q&A sites do support SV discussions, but there is still a fair disconnection between SO and SSE. More effort is required to motivate cross-site engagement to better support (difficult) SV topics.

In the future, we aim to investigate SV topics on other Q&A sites. We also plan to correlate the findings of SV discussions on Q&A sites with SV detection and fixing activities on version control systems such as GitHub.

ACKNOWLEDGMENTS

The work was supported by the Cyber Security Research Centre Limited whose activities are partially funded by the Australian Government's Cooperative Research Centres Programme.

REFERENCES

[1] Ahmad Abdellatif, Diego Costa, Khaled Badran, Rabe Abdalkareem, and Emad Shihab. 2020. Challenges in chatbot development: A study of stack overflow posts. In *Proceedings of the 17th International Conference on Mining Software Repositories*. 174–185.

[2] Syed Ahmed and Mehdi Bagherzadeh. 2018. What do concurrency developers ask about? a large-scale study using stack overflow. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. 1–10.

[3] Mehdi Bagherzadeh and Raffi Khatchadourian. 2019. Going big: a large-scale study on what big data developers ask. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 432–442.

[4] Abdul Ali Bangash, Hareem Sahar, Shaiful Chowdhury, Alexander William Wong, Abram Hindle, and Karim Ali. 2019. What do developers know about machine learning: a study of ML discussions on StackOverflow. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 260–264.

[5] Anton Barua, Stephen W Thomas, and Ahmed E Hassan. 2014. What are developers talking about? an analysis of topics and trends in stack overflow. *Empirical Software Engineering* 19, 3 (2014), 619–654.

[6] Shahab Bayati and Marzieh Heidary. 2016. Information Security in Software Engineering, Analysis of Developers Communications About Security in Social Q&A Website. In *Pacific-Asia Workshop on Intelligence and Security Informatics*. Springer, 193–202.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine learning research* 3, Jan (2003), 993–1022.

[8] Mehran Bozorgi, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2010. Beyond heuristics: learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 105–114.

[9] Zhenpeng Chen, Yanbin Cao, Yuanqiang Liu, Haoyu Wang, Tao Xie, and Xuanzhe Liu. 2020. A comprehensive study on challenges in deploying deep learning based software. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 750–762.

[10] William G Cochran. 2007. *Sampling techniques*. John Wiley & Sons.

[11] Tapajit Dey, Andrey Karnauch, and Audris Mockus. 2021. Representation of developer expertise in open source software. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE.

[12] Seyed Mohammad Ghaffarian and Hamid Reza Shahriari. 2017. Software vulnerability analysis and discovery using machine-learning and data-mining techniques: A survey. *ACM Computing Surveys (CSUR)* 50, 4 (2017), 1–36.

[13] Junxiao Han, Emad Shihab, Zhiyuan Wan, Shuiguang Deng, and Xin Xia. 2020. What do programmers discuss about deep learning frameworks. *Empirical Software Engineering* 25, 4 (2020), 2694–2747.

[14] Zhuobing Han, Xiaohong Li, Zhenchang Xing, Hongtao Liu, and Zhiyong Feng. 2017. Learning to predict severity of software vulnerability using only vulnerability description. In *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 125–136.

[15] Benjamin V Hanrahan, Gregorio Convertino, and Les Nelson. 2012. Modeling problem difficulty and expertise in stackoverflow. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion*. 91–94.

[16] Mubin Ul Haque, Leonardo Horn Iwaya, and M Ali Babar. 2020. Challenges in Docker Development: A Large-scale Study Using Stack Overflow. In *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 1–11.

[17] Sameera Horawalavithana, Abhishek Bhattacharjee, Renhao Liu, Nazim Choudhury, Lawrence O. Hall, and Adriana Iamnitchi. 2019. Mentions of Security Vulnerabilities on Reddit, Twitter and GitHub. In *IEEE/WIC/ACM International Conference on Web Intelligence*. 200–207.

[18] William R Knight. 1966. A computer method for calculating Kendall's tau with ungrouped data. *J. Amer. Statist. Assoc.* 61, 314 (1966), 436–439.

[19] Kyriakos Kritikos, Kostas Magoutis, Manos Papoutsakis, and Sotiris Ioannidis. 2019. A survey on vulnerability assessment tools and databases for cloud-based web applications. *Array* 3 (2019), 100011.

[20] Triet H M Le, Roland Croft, David Hin, and M Ali Babar. 2021. Reproduction package. https://github.com/lhmtriet/SV_Empirical_Study

[21] Triet H M Le, David Hin, Roland Croft, and M Ali Babar. 2020. PUMiner: Mining Security Posts from Developer Q&A Websites with PU Learning. In *Proceedings of the 17th International Conference on Mining Software Repositories*. 350–361.

[22] Triet H M Le, Bushra Sabir, and M Ali Babar. 2019. Automated software vulnerability assessment with concept drift. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 371–382.

[23] Tamara Lopez, Thein Tun, Arosha Bandara, Levine Mark, Bashar Nuseibeh, and Helen Sharp. 2019. An anatomy of security conversations in stack overflow. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 31–40.

[24] Henry B Mann. 1945. Nonparametric tests against trend. *Econometrica: Journal of the Econometric Society* (1945), 245–259.

[25] Henry B Mann and Donald R Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics* (1947), 50–60.

[26] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica* 22, 3 (2012), 276–282.

[27] Na Meng, Stefan Nagy, Danfeng Yao, Wenjie Zhuang, and Gustavo Arango Argoty. 2018. Secure coding practices in java: Challenges and vulnerabilities. In *Proceedings of the 40th International Conference on Software Engineering*. 372–383.

[28] Akond Rahman, Effat Farhana, and Nasif Imtiaz. 2019. Snakes in paradise?: Insecure python-related coding practices in stack overflow. In *IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 200–204.

[29] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*. 399–408.

[30] Christoffer Rosen and Emad Shihab. 2016. What are mobile developers asking about? a large scale study using stack overflow. *Empirical Software Engineering* 21, 3 (2016), 1192–1223.

[31] Sefa Eren Sahin and Ayse Tosun. 2019. A conceptual replication on predicting the severity of software vulnerabilities. In *Proceedings of the Evaluation and Assessment on Software Engineering*. 244–250.

[32] Carolyn B. Seaman. 1999. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering* 25, 4 (1999), 557–572.

[33] Muhammad Shahzad, Muhammad Zubair Shafiq, and Alex X Liu. 2012. A large scale exploratory analysis of software vulnerability life cycles. In *2012 34th International Conference on Software Engineering (ICSE)*. IEEE, 771–781.

[34] Christoph Treude, Ohad Barzilay, and Margaret-Anne Storey. 2011. How do programmers ask and answer questions on the web?(NIER track). In *Proceedings of the 33rd international conference on software engineering*. 804–807.

[35] Xin-Li Yang, David Lo, Xin Xia, Zhi-Yuan Wan, and Jian-Ling Sun. 2016. What security questions do developers ask? a large-scale study of stack overflow posts. *Journal of Computer Science and Technology* 31, 5 (2016), 910–924.

[36] Mansooreh Zahedi, M Ali Babar, and Christoph Treude. 2018. An empirical study of security issues posted in open source projects. In *Proceedings of the 51st Hawaii International Conference on System Sciences*.

[37] Mansooreh Zahedi, Roshan N Rajapakse, and M Ali Babar. 2020. Mining Questions Asked about Continuous Software Engineering: A Case Study of Stack Overflow. In *Proceedings of the Evaluation and Assessment in Software Engineering*. 41–50.