Vulnerably (Mis)Configured? Exploring 10 Years of Developers' Q&As on Stack Overflow

Richard May
Harz University of Applied Sciences
Wernigerode, Germany
rmay@hs-harz.de

Christian Biermann
msg systems ag
Hamburg, Germany
Harz University of Applied Sciences
Wernigerode, Germany
christian.biermann@msg.group

Xenia Marlene Zerweck
Harz University of Applied Sciences
Wernigerode, Germany
xenia.zerweck@gmx.de

Kai Ludwig
Landesamt für Geoinformation und Landesvermessung Niedersachsen
Hannover, Germany
Kai.Ludwig@lgln.niedersachsen.de

Jacob Krüger
Eindhoven University of Technology
Eindhoven, The Netherlands
j.kruger@tue.nl

Thomas Leich
Harz University of Applied Sciences
Wernigerode, Germany
tleich@hs-harz.de

ABSTRACT

The increasing number of attacks exploiting system vulnerabilities in recent years underpins the growing importance of security; especially for software comprising configuration options that may cause unintended vulnerabilities. So, not surprisingly, developers discuss secure software configurations extensively, for instance, via community-question-answering systems like Stack Overflow. In this exploratory study, we analyzed 651 Stack Overflow posts from 2013 until 2022 to investigate what vulnerabilities in the context of configuring software developers discuss. We employed a manual data analysis and automated topic modeling using Latent Dirichlet Allocation to identify and classify relevant topics and contexts. Our results show that vulnerabilities in the context of configuring receive more and more interest, with most posts discussing issues related to faulty security configurations and dependencies causing vulnerabilities that could be or have actually been exploited. Overall, we contribute insights into configuration and security issues that developers experience in the real world. Such insights help researchers and practitioners understand and resolve these issues, thereby guiding future improvements.

CCS CONCEPTS

• **Software and its engineering**; • **Security and privacy** → **Vulnerability management**;

KEYWORDS

variability, configuration, vulnerability management, security, Stack Overflow, community-question-answering system

VaMoS 2024, February 7 –9, 2024, Bern, Switzerland

ACM ISBN 979-8-4007-0877-0/24/02…$15.00
https://doi.org/10.1145/3634713.3634729

ACM Reference Format:
Richard May, Christian Biermann, Xenia Marlene Zerweck, Kai Ludwig, Jacob Krüger, and Thomas Leich. 2024. Vulnerably (Mis)Configured? Exploring 10 Years of Developers' Q&As on Stack Overflow. In *18th International Working Conference on Variability Modelling of Software-Intensive Systems (VaMoS 2024), February 7 –9, 2024, Bern, Switzerland*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3634713.3634729

1 INTRODUCTION

The continuous rise of attacks on modern software systems highlights the ever-growing importance of software security [44, 97]. An attack (e.g., cross-site scripting [87]) typically exploits a system vulnerability, and negatively affects the confidentiality, integrity, or availability of a system [38, 60]. Vulnerabilities can occur for a variety of reasons, such as bugs [95], code smells [19], or authentication issues [85]. Moreover, with software becoming increasingly configurable, the risks of vulnerabilities also grows [4, 49, 51], for example, due to unknown feature interactions [64], invalid settings [24], or misconfigurations [75]. So, the topic of securely configuring software systems to minimize the risks of potential vulnerabilities has become more and more important [24].

Developing secure software is a challenging task, especially when facing the additional complexity caused by configuration options enabling numerous customized system variants [24, 59, 67]. To tackle this challenge, various researchers study security aspects in the context of configurability [37, 51], such as secure network configurations [14, 89] or mobile apps [29, 83]. However, the actual vulnerabilities related to configuring have not been the focus of such research directions and have not been oriented towards practice [37, 52]. We argue that we require a better understanding of the practical challenges developers face when it comes to security and configurability. Improving this understanding can guide the research community in identifying and tackling practice-driven problems [42, 84].

Developers' voices and experiences can be found in Community-Question-Answering (CQA) systems, which provide a platform for developers to discuss topics and resolve problems associated to software engineering based on their shared experiences [6, 42, 43, 73]. The most prominent CQA system for developers is Stack Overflow,[1]

[1] https://stackoverflow.com/

which contains millions of discussions on diverse topics [10, 55], such as on vulnerabilities in the context of configurability. On Stack Overflow, developers highlight not only security or configuration problems, but consequently discuss also experiences, conceptual understandings, or implementation insights to help each other. As a result, Stack Overflow provides a valuable basis for understanding the complex dynamics and relationships between vulnerabilities and software configurations from the perspectives of actual software developers.

In this paper, we present an exploratory analysis of 651 Stack Overflow discussions (2013–2022) that are concerned with vulnerabilities in the context of configurability; more precisely in the context of configuring applications. We provide a comprehensive dive into the relevant topics, contexts, and main issues developers discuss. More specifically, our goal is to **contribute an overview of whether and what configurability-related vulnerability topics practitioners discuss**. To the best of our knowledge, there is no comparable work analyzing discussions from Stack Overflow or other CQA systems related to this area. In detail, we contribute the following in this paper:

- We provide an overview of the topics and their temporal evolution that are discussed on Stack Overflow in relation to vulnerabilities in the context of configurability.
- We discuss the analyzed posts (i.e., questions with their answers and comments), the topics, and their broader context to shed light into the connections between vulnerabilities and configurations.
- We publish an open-access repository with our analysis file to enable replications of our study.[2]

Our findings can support researchers and developers in understanding the importance of secure configurations and the impact incorrect configurations can have, evidenced by practice-oriented experiences contributed by a broad CQA community. In particular, our results can help strengthen the awareness for secure configurations, cross-dependencies, configuration-related vulnerabilities, and appropriate security mechanisms.

2 BACKGROUND

Community-Question-Answering Systems. CQA systems are platforms on which users can ask questions, write answers, and make comments on a variety of topics [55, 73, 93]. Such interactions lead to a collaboratively created collection of peer-assessed knowledge that includes different perspectives, opinions, and proposals for problem solutions. CQA systems operate based on the ideas of reputation and reward (e.g., upvotes) to encourage high-quality discussions [72, 91]. The most prominent CQA system for software development is Stack Overflow, which involves millions of developer discussions and code snippets, for instance, on software-design decisions, programming languages, databases, APIs, algorithms, code reviews, or bugs [10, 42, 55, 90].

The strengths of CQA systems are their potential diversity and timeliness. Specifically, such systems can gather experiences and opinions from people with various backgrounds and from all around the world, all contributing their unique insights and problems. This typically results in diverse solution proposals to concrete problems that are often practical, well-tested, and applicable. However, the quality of questions and especially answers can vary in terms of topicality and correctness [72, 91, 93]. Nevertheless, CQA systems offer a great data base to get in-depth insights into trending and relevant topics that developers care about [10, 43].

System Configuration. Modern software systems are usually characterized by their ability to offer a variety of variants based on a set of configurable features (i.e., their variability) [2]. Technically, developers implement configuration options for features in their system by using a variability mechanism, for instance, feature-oriented programming [8, 15] or the C preprocessor [1, 48, 79]. By enabling, disabling, or setting features (i.e., configuring), system variants can be created to adapt the system according to a stakeholder's requirements [2, 8, 30, 68]. To organize, implement, and document features along with their relationships, dependencies, and directives, configurable systems typically integrate the variability mechanism with a variability model [18]. Feature models are the de facto standard used on the conceptual level, while configuration files are used on the implementation level to define feature constraints, and thus allowable configurations [11, 35, 63, 71]. For instance, there can be cases where certain features are mutually exclusive and cannot exist simultaneously, while others require each other [2, 54, 86]. Unfortunately, an increasing number of configurable features often leads to non-trivial interactions between these, meaning that features can influence each other's functionality and potentially cause bugs. So, it is essential to carefully manage and verify configurations, especially if they can impair a system's viability or may induce vulnerabilities [1, 25, 51, 80].

Security and Vulnerability Management. To obtain a reliable and high-quality system, it is vital to ensure its security [31, 58]. However, ensuring security is challenging, since software is usually diverse and has to comply with a wide range of security policies, requirements, and regulations to minimize potential threats. Such threats are incidents that can negatively impact a system, such as an unintended failure of a mitigation technique or unauthorized access [32, 33, 66]. Threats are closely related to vulnerabilities, which are system weaknesses that could be exploited by potential attackers [32, 66], for instance, through cross-site scripting [87] or SQL injections [28]. The actual likelihood of such events as well as their potential consequences is called a risk [32, 34]. There are a variety of data sources (e.g., vulnerability databases) that classify and rate (e.g., CVE[3]) incidents and their risks to provide an overview of known vulnerabilities [36, 56].

To ensure the security of a system, developers must implement mitigation techniques, strategies to identify threats, risk as well as vulnerability analyses, and appropriate counter measures, such as authentication [33, 66]. This is far from trivial, since inappropriate counter measures can cause vulnerabilities, too; especially when different system variants with individual configurations may cause unintended feature interactions, bugs, or misconfigurations that can lead to fatal security failures [64, 75, 88]. So, comprehensive security management must take system configurations into account to minimize the occurrence of vulnerabilities and ensure the security (i.e, confidentiality, integrity, and availability) of a system [33, 65, 69].

[2] https://doi.org/10.5281/zenodo.10245332

[3] https://www.cve.org/

3 METHODOLOGY

In this section, we detail our methodology including our research goal and questions, study design, and conduct.

3.1 Goal and Research Questions

We aimed to identify developers' discussions related to vulnerabilities and configuring to understand the respective problems they face in practice. To this end, we defined three Research Questions (RQs):

RQ_1 **To what extent are vulnerabilities in the context of configurability discussed on Stack Overflow?**
First, we aimed to assess whether the Stack Overflow community is interested in the intersection of the two areas and how this has evolved over time.

RQ_2 **What are the main topics in the intersection of security and configurability discussed on Stack Overflow?**
Second, we aimed to classify what topics are discussed by, and thus important to, the community.

RQ_3 **What are the contexts of the discussions and their main topics on vulnerabilities in the context of configuring?**
Lastly, we aimed to elicit the concrete contexts of the discussions to shed light into the causes, vulnerabilities, and attacks related to security and configuring.

By addressing these research questions, we aim to provide an overview of the challenges developers face when working on the security of configurable systems, contributing insights for practitioners and researchers to learn about and advance this intersection.

3.2 Study Design

We performed an exploratory study of Stack Overflow discussions [43] by mining data from Google's BigQuery Stack Overflow data set.[4] Specifically, we conducted a manual data analysis and automated topic modeling using the Latent Dirichlet Allocation (LDA) method [13] as we show in Figure 1. Our study design follows a similar method as conducting a systematic literature review [39], which improves transparency, replicability, and validity—and has been employed successfully for other topics in the past [42].

Search String. First, we created a search string to identify relevant data for addressing our research questions:

```
("secur*") AND
("vulnerabilit*" OR "weakness*" OR "breach*" OR
"exposure*" OR "CVE*" OR "CWE*") AND
("config*")
```

The string is based on the two topics relevant for our study: *vulnerabilities* and *configuring* (including appropriate wildcards). Regarding configuration, we only include terms directly related to the word stem *config* to limit the number of posts found. In the context of vulnerabilities, we included frequently used synonyms (e.g., *breach*) and important terms (e.g., *CVE*). We included *security* as an explicit term to ensure that the discussions we retrieve are connected to security, since some other terms may be used in different contexts as well (e.g., *weakness*). Please note that we did not add terms weakly linked to some security issues (e.g., configuration errors), since these do not necessarily cause security vulnerabilities or related issues. Since such other issues are out of our scope, and we assume that our search terms would still occur if vulnerabilities arise, we argue that we cover the relevant discussions with our search string.

Data Set. We applied the search string as an SQL query on the Google BigQuery Stack Overflow data set. BigQuery provides a web-based console that allows to (automatically) execute queries and analyses of diverse public data sets [20]. The Stack Overflow data set includes 20 different fields, covering data regarding questions, answers, and users. Thus, it is an appropriate data set for our purpose and eases the conduct of our study, because we do not need to crawl or manually search through Stack Overflow itself.

Selection Criteria. To identify relevant posts, we defined three Inclusion Criteria (IC):

IC_1 The post has been created in the last decade (2013–2022).
IC_2 The post is still available on the Stack Overflow website.
IC_3 The post is directly connected to a vulnerability-related issue in the context of configuring.

By restricting the time period, we ensured that our analysis reflects on more recent, and thus more relevant, topics (IC_1). We argue that issues older than 10 years are usually outdated and of minor relevance for understanding current practices, or have become widely known in security engineering. Since the BigQuery Stack Overflow data set comprises some deleted posts, we manually checked for each post whether it was still available on Stack Overflow itself (IC_2) and whether it is actually connected to our research goal (IC_3). Deleted posts usually have no added value, since they may be removed due to irrelevance or inappropriate content.

Data Extraction and Analysis. The first two authors defined 13 categories of data for the data extraction and analysis. They identified nine post attributes from the BigQuey data set as relevant, namely **ID**, **creation year**, **title**, **body**, **tags**, **upvotes and downvotes**, **favorite points**, as well as **view and answer count**. While this data sufficed to answer RQ_1, we defined four more categories to synthesize the titles, bodies, and tags of posts to tackle RQ_2 and RQ_3:

System to specify the technological environment or platform in which a discussed vulnerability issue occurred (e.g., software, framework, operating system).

Development topic to define the general subject of the post (e.g., web development, mobile development).

Configuration context to classify how configuring connects to a post, which can be one of the following:

- *Conceptual understanding* focusing on theoretical aspects without any implementation details, such as valid configurations through variability modeling [61].
- *Evolutionary behavior* referring to configuration issues during the evolution of systems, such as insecure version updates of software [8].
- *Issue solution* considering troubleshooting and resolving problems without providing concrete implementation details, such as configuration errors [70].
- *Implementation* dealing with the actual code of a system or its features, such as a configuration file [70].
- *Tools and software* referring to concrete tools for developing, testing, or other tasks, such as FeatureIDE [53].

[4] https://cloud.google.com/blog/topics/public-datasets/google-bigquery-public-datasets-now-include-stack-overflow-q-a?hl=en, accessed August 1st, 2023.

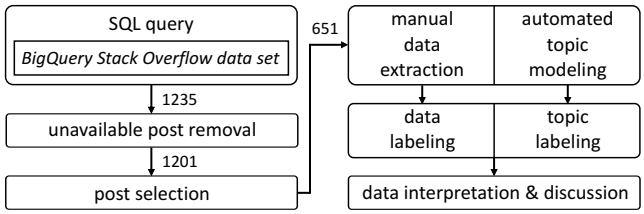

Figure 1: Overview of our methodology, with numbers representing the amount of posts after each step.

Security context to classify how vulnerability-related issues connect to the post, which can be one of the following:

- *Threat or risk* focusing on issues that threaten a system and may lead to a vulnerability, such as the possibility of unauthorized access [32].
- *Vulnerability or attack* referring to a weakness and/or associated exploit, such as remote code execution [94].
- *Measure* including techniques or protocols to mitigate risks of an exploits, such as encryption like AES [82].
- *Policy* describing a guideline for secure development and deployment, such as regular risk assessments [66].

Besides our manual data extraction and analysis, we used topic modeling on titles and bodies following the guidelines of Agrawal et al. [5] to tackle RQ_3. Specifically, we relied on LDA to discover what *configuration topics* the posts address. LDA is a generative probabilistic model in which a topic is represented as the probability that each of its terms occurs in a given text [13], with a posts' title (for questions) and body representing the texts. Since LDA has been widely used in software-engineering research (cf. Section 6) and similar studies to ours [10], we argue that it is suitable for our work.

3.3 Study Conduct

Conducting our study involved two tasks (cf. Figure 1): (1) a manual data extraction and analysis as well as (2) topic modeling, which we carried out in parallel.

Manual Data Extraction and Analysis. We executed our SQL query on August 1st, 2023, which returned 1,235 posts. Then, the first two authors investigated 30 posts together to derive a uniform understanding and data-extraction process. Subsequently, each of the two analyzed half of the posts. Each case of ambiguity or confusion was discussed among both authors. In this process, they found that 34 posts were already deleted on Stack Overflow and classified 550 posts as out of scope according to our inclusion criteria. So, the two authors manually extracted data for 651 posts. For the data extraction, the two authors used a collaborative spreadsheet in which they documented all 13 categories of data as free-text (e.g., system) or pre-defined categories (e.g., configuration context). For this purpose, they used a combination of open coding to identify relevant data and open-card-sorting [96] to synthesize recurring categories from feasible free-text data (e.g., topics). The spreadsheet served as basis for all authors to analyze and check the data.

Topic Modeling. For the topic modeling, the first author implemented a Python script using the libraries `NLTK`, `stop_words`, and `gensim`. The script first pre-processed the 651 posts' titles and bodies to reduce their complexity. In particular, the script removed unnecessary HTML tags, tokenized the text, performed lemmatization (using the Penn Treebank tagset), and vectorized the text (i.e., converting the text into numerical values based on term frequency-inverse document frequency). Then, the algorithm applied LDA on the results. To this end, we experimented to find the optimal number of topics k according to the coherence value by varying k from four to 20 and performing between 100 and 500 (steps of 100) iterations. Our experimentation implied a range from seven to 11 topics with 200 iterations to lead to a feasible coherence of around 0.6 (i.e., a "good" score [74]). Then, we picked nine topics as our k, since it resulted in the highest coherence, and defined the hyper-parameters ($\alpha = k$, $\beta = 0.01$). Finally, the first two authors manually categorized the topics identified by LDA via open-card sorting, which they added to the spreadsheet. To validate the topics, we randomly picked 100 posts and the first two authors reviewed these together. Moreover, the third author performed a complete review of the whole topic-modeling process.

4 RESULTS AND DISCUSSION

In this section, we present and discuss the results for each of our research questions individually.

4.1 RQ_1: General Relevance

To tackle RQ_1, we analyzed the post attributes to understand to what extent developers on Stack Overflow discuss the intersection of vulnerabilities and configuring.

Post Attributes. We display the yearly distribution of posts in our data set in Figure 2. As we can see, the total number of posts per year increased continuously with a few exceptions. In 2013 and 2014, we can see a constant number of 18 posts, while this number raised to 58 posts in 2015. The following two years, the number of posts grew further to 65 (2016) and 70 (2017). Interestingly, in 2018, there is a large outlier with 122 posts, which we analyze in more detail in Section 4.3. For the last four years, the number of posts remained more or less constant. More specifically, compared to 2017, there was an increase to 77 posts in 2019, a decrease to 70 posts in 2020, and a slight increase again via 74 (2021) to 79 (2022) posts. Note that, at the time of our data collection, the BigQuery Stack Overflow data set contained only data until September 2022, implying that the final number of posts for 2022 is likely higher.

Considering the answer status, a majority of 491 (75.42 %) out of 651 posts have at least one answer to the involved question (160 questions are unanswered). Out of the 491 answered posts, 446 contain one or two answers. Only two posts comprise 10 or more answers (`72401421`, `71901632`). Most posts have been viewed between 100 and 2,500 times, with 24 posts having more than 10,000 views. Despite the views, the majority of 516 (79.26 %) posts got no favorite points. In contrast, 127 posts have between one to five points and nine posts have more than five points. The question ratings vary between zero (289 posts) and five (305 posts). Only 47 questions got a higher rating than five points, including a question with 251 points (`72401421`). Furthermore, we found 23 posts that have negative points. Surprisingly, 2022 includes the most views (464,868), most favorite points (116), and highest question score

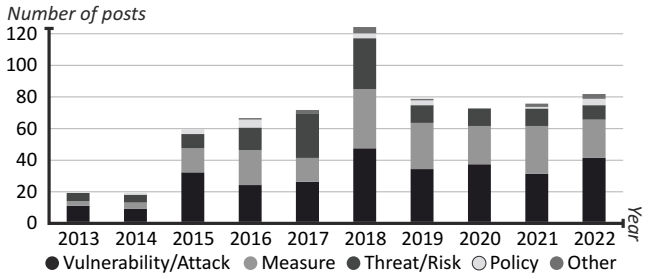

Figure 2: Chronological distribution of the posts we analyzed.

(451), even though the data for that year is not even completely covered in our data.

Discussion. We found that the interest in vulnerability-related issues in the context of configuring has steadily increased, especially between 2013 and 2018. Except due to one peak in 2018, the number of posts per year has mostly grown—indicating a smaller, albeit visible, interest of the community. This observation is underscored by 75.42 % of the questions receiving at least one answer, which implies a certain level of expertise on the topic and interest of developers in helping others facing problems in it. Our data matches similar studies on other security concerns highlighting that security is a recurring topic on Stack Overflow and that developers are aware of that fact [17, 77]. While the relevance increases only slowly and stays almost constant after 2018, we argue that our data supports the assumption that developers experience and actively discuss vulnerabilities related to configuring.

Outliers like the year 2018 or single posts with a particularly high number of answers may relate to prominent or recurring vulnerabilities and systems, which we further analyze in Section 4.3. The recent peak of views, favorite points, and question ratings in 2022 could be related to the COVID-19 pandemic, which has led to an increased use of Stack Overflow [40]. Similarly, the availability of large language models like ChatGPT or GitHub Copilot may also impact user activity on Stack Overflow in the future. So, it is important to replicate our findings in the future and connect them to insights on the use of such tools. While such tools may decrease the number of posts on the topic that are posted on Stack Overflow, it may become even more relevant to study those. In fact, recommendations made by such tools are still error-prone and may cause problems themselves [26], which, in turn, may be discussed on Stack Overflow. Overall, we argue that the intersection of vulnerabilities and configuring apparently impacts developers in practice, and Stack Overflow represents a data set with hundreds of relevant posts.

RQ_1: General Relevance

Vulnerability-related issues that are connected to configuring are a visible and constantly growing topic on Stack Overflow, and thus arguably highly relevant in practice, too.

4.2 RQ_2: Main Topics

To tackle RQ_2, we identified and classified the affected systems and development domains to discover main topics within the discussions. Note that posts may relate to multiple topics, but we assigned only the dominant one (i.e., the one we identified as most important in the question of a post). We display the results in Figure 3, in which the gray boxes in the background aggregate configuration topics (cf. Section 4.3) into their respective development topics.

Results. We categorized the domains of the posts into five main development topics. Most posts (261) refer to web development, with further considerable amounts of posts being concerned with server and storage development (196) or general system and software development (117). Mobile development is the topic in 68 posts, while only nine posts cover operating systems (OSs).

We identified around 50 different systems or technologies within these topics, showcasing a high diversity of areas in which vulnerabilities and configuring are intertwined. Moreover, please note that we could not even determine all systems and technologies, since they were sometimes not described. Overall, the systems and technologies involve various areas, ranging from programming languages (e.g., C#) over development framework (e.g., Spring framework) to content-management systems (e.g., Wordpress). All of these share that they build on variability mechanisms to configure their behavior, or are used to create configurable software. Referring to web development, we identified that the majority of issues occurred in the context of Spring (59), including Spring Boot and Spring Security. Another considerable amount of posts refers to .NET platforms (44) or Ruby on Rails (22). Interestingly, other highly-configurable systems, including content-management systems like Wordpress (4) or enterprise-resource-planning systems like SAP ERP (2) occur rarely. Regarding mobile development, most posts are related to Android (34) or Apache Cordova (14) whereas others like iOS (6) are rarely mentioned. Posts on operating systems refer mainly to Linux distributions (6), particularly Red Hat Enterprise Linux. For server and storage development, we found that posts related to Spring (Cloud) and a specific database issue within it are mentioned most often (27). Other posts refer to storage media like traditional SQL databases (23) or storage environments like cloud services, for instance, Microsoft Azure (13) or Amazon Web Services (7). Moreover, 14 posts are connected to Apache Tomcat web servers.

Discussion. The results show a great diversity in the development topics and systems discussed. Web development is the most prominent topic, indicating its central role in the development community, but also its high risks for configuration-related vulnerabilities. This observation is in line with the fact that server and storage systems are also frequently discussed. Existing research underpins these topics as particularly critical by finding that internet technologies and data storages are the main targets of cyber attacks [3, 9, 23]. All of this evidences that functional and secure configuring becomes even more important the more a system is exposed to the internet and the more (sensitive) data it processes or stores. The importance of other topics (e.g., operating systems) is not diminished by this prominence, but the extent to which vulnerabilities are discussed decreases as the dependence on the internet decreases. Interestingly, most posts occur in the context of the Spring Framework (181 posts), including Spring Boot, Spring Security, and Spring Cloud.

Surprisingly, content-management systems, which are typically configurable by definition (e.g., Wordpress), are only discussed in a small number of posts. Since there is a lot of research, especially in the context of security for Wordpress [16, 81], this underrepresentation raises some questions. Similarly, the small number

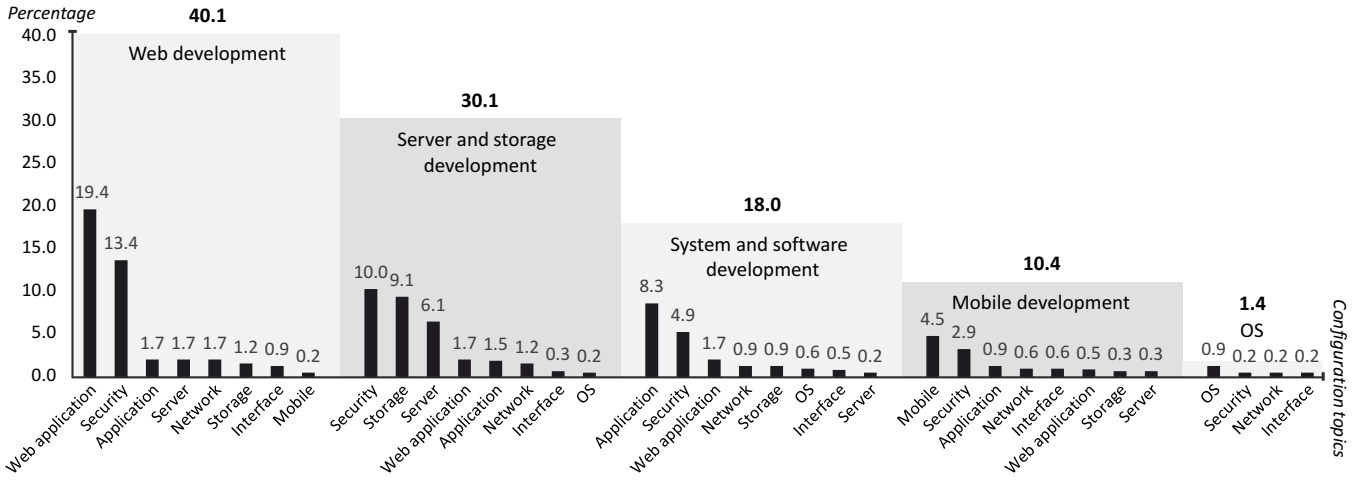

Figure 3: Distribution of configuration topics (small bars) and their classification in development topics (boxes).

of posts on operating systems, which are mainly related to Linux distributions, is also surprising. The Linux kernel [21] is known to be highly configurable and studies on configuration-related bugs within the kernel have been performed [1, 2]. Since the actual reasons for these under-representations are unclear (e.g., use of other discussion platforms, missing awareness for security issues and configuring, sufficient security education), they pose interesting future work. Regarding mobile systems, Android-related apps are discussed particularly often. In contrast to iOS, these apps must be able to run on a variety of mobile devices or browsers [7], which requires extensive configurability. From our analysis, its seems that the Stack Overflow community seems aware of the effects of cross-platform configurations and insufficient vulnerability management.

Overall, the posts we analyzed often deal not only with one system, but with the interactions of several ones. To keep this study's scope manageable, we have focused only on the dominant systems. However, it is very likely that there are a number of cross-system or cross-technology configurations that lead to misconfigurations (e.g., due to unconsidered dependencies) or unintended feature interactions. Further research is needed in this context to understand the relationships between vulnerabilities and such configurations.

RQ$_2$: Topics

The posts cover diverse topics, with web development as well as server and storage development being most prominent. So, most discussions on vulnerabilities and configuring revolve around systems with network and storage capabilities. Spring and .NET systems are discussed frequently, while systems known for high configurability (e.g., Wordpress, Linux) are discussed surprisingly rarely.

4.3 RQ$_3$: Contexts

To tackle RQ$_3$, we investigated the contexts of the vulnerabilities and configurability discussed in the posts. More specifically, we next detail common security issues related to configuring by exploring these two contexts and then discuss the results of our topic modeling. Note that we again considered only the most dominant context even if a post involved multiple ones.

Security Contexts. Following the five categories we defined for the security context (cf. Section 3.2), most posts (279) refer to an attack or a vulnerability that could be exploited by an attack. Note that vulnerabilities are usually closely related and difficult to distinguish from each other even if their context is described in great detail within a post. Still, we could generally differentiate whether a post was concerned with whether a *vulnerability* that may be exploited exists, or how such a vulnerability could be exploited through an *attack*. For example, the following two posts refer to Cross-Site Scripting (XSS), each from one of the two perspectives:

```
Vulnerability                                                ID:71733286
"Checkmarx is giving XSS vulnerability for [the] following
method in my Controller class."
```

```
Attack                                                       ID:46407680
"[Should I] store JWTs on cookies to get protection against
XSS attacks?"
```

Also, we found that 203 posts primarily discuss security measures to protect a system or to mitigate the risk of an exploit. These posts usually relate to authentication (70) or authorization (19) issues:

```
Measure                                                      ID:68064690
"After I configure authentication [I got] the following
error [...] after I log in through chrome."
```

Another 135 posts revolve around various security threats and risks. These range from dependency issues (53) that could lead to vulnerabilities to insecure system or security configurations (98):

```
Threat or Risk                                               ID:15414791
"I recognize that it's possible for a plugin to override
a dependency of the scheduler causing it to do something
nefarious, or to reflect into the common assembly and get
at internals."
```

In addition, we identified 20 posts that focus on policies to facilitate security risk management, including vulnerability management. These posts are quite diverse, involving general violations of security policies (6) or even updating or creating new policies (4), for instance, in the context of secure authentication:

```
Policy ID:54891501
"[By] default the Group Policy is updated every two hours,
so the changes might not be persistent. Also, most likely
you will breach your company security rules."
```

In total, 103 posts name specific vulnerabilities or attacks. Typically, these posts are highly diverse and do not fall within our security contexts. Across all posts, most vulnerabilities relate to XSS (21), Cross-Site-Request-Forgery (CSRF, 21), Log4j (13), directory-traversal (12), SSL via POODLE (6), or SQL injections (4). Others refer to file injections (5), for instance, via CSV, XML, or JSON. Concrete vulnerabilities based on CVE-identifiers are mentioned in 23 cases, for example, CVE-2015-3192, CVE-2014-3625, and CVE-2014-3578 (`39273479`).

Configuration Contexts. Following the five categories we defined for the configuration context, we found that most posts (248) discuss an issue solution, such as missing configurations or configuration errors, without providing concrete implementations:

```
Issue Solution ID:49442855
"Please suggest the configs that we can use directly on
Tomcat."
```

In contrast, 183 posts are related to implementation issues, such as successfully configuring security measures (e.g., authentication mechanisms), or parts of a systems (e.g., modules):

```
Implementation ID:39267531
"How do I configure this module and make minimal changes to
make sure [the] whole application is secured from CSRF."
```

We found that 117 posts discuss a conceptual understanding of configuration-related issues. These are usually very general or hypothetical to get a basic understanding of the topic. In some cases, these posts are also based on small case studies or scenarios to better explain a specific use case and the associated problem:

```
Conceptual Understanding ID:29845208
"In this scenario, a security flaw shows up in the wild
and the server is reconfigured to reject the vulnerable
protocol."
```

Another 84 posts refer to evolutionary behavior. These posts are diverse, including, for instance, emerging vulnerabilities caused by deprecated versions, software dependencies between versions, or updates intended to prevent security issues:

```
Evolutionary Behavior ID:63171516
"After updating npm to the latest, I ran npm audit and got
two vulnerabilities for the dot-prop package dependency."
```

We identified 19 posts related to tools and software. Interestingly, we identified no issues for tools used to model configurable systems and their features, such as FeatureIDE [53]. All posts refer to tools used to mitigate security issues, for example, the OWASP vulnerability scanner or SonarQube security scanner:

```
Tools and software ID:54695037
"I've tried including an obvious SQL Injection vulnera-
bility but its still not detected. [...] So I'm missing
some extra configuration or plugin?"
```

Topic Modeling. Lastly, we employed LDA on our data to extract the nine most discussed topics on configuring. The topics we identified via LDA partly overlap with our manual analysis results (cf. Section 4.2), which improves our confidence in the validity of our study. We illustrate which configuration topics unveiled by the LDA (small bars inside the boxes) are covered by which development topic we elicited manually in Figure 3 (boxes).

The most discussed configuration issues refer directly to *security* configurations (204), such as authorization, authentication, or login configurations. Interestingly, a major topic is the correct security configuration for the Spring framework (102), especially Spring Security (e.g., `43999961`). A total of 150 posts deal with problems arising from configuring *web applications*. Not surprisingly, these posts are very common within our homonymous topic of web applications, and discuss the configuring of the features and dependencies of such applications, for instance, regarding SSL (e.g., `22764324`) or browsers like Google Chrome and Mozilla Firefox (e.g., `46933820`). LDA identified 81 posts dealing with *application* configurations, which are not (primarily) web applications. Issues related to this topic mainly refer to the configuration of features and their dependencies, for instance, configuring in the context of npm (e.g., `72408347`) or of WPF applications (e.g., `28506734`). In 75 posts, the configuration of *storages* (e.g., cloud) or a storage medium (e.g., database) is discussed as the cause of a vulnerability-related issue. Some of these posts focus on misconfigurations (e.g., `52047261`), while others refer to insecure repositories leading to sensitive information disclosure (e.g., `71901632`). Moreover, LDA elicited 54 posts related to *server* configurations. These discussions deal with various issues, such as SSL (e.g., `50042299`) or the features and dependencies of web servers like Apache Tomcat (e.g., `64230009`) or nginx (e.g., `34257611`).

For *network* and *mobile* configurations, we found 30 posts each. The systems in the context of network configurations are diverse, including, for instance, proxy configurations (e.g., `55786273`) or IP configurations (e.g., `52050742`). Mobile configurations span similarly diverse topics, involving insecure mobile policy configurations (e.g., `34383655`) or issues related to a configuration leading to app-store rejections (e.g., `67212589`). Only a few posts deal with configuring *interfaces* (16) or *operating systems* (11). Interface configurations typically relate to API configurations, for instance, to authenticate data transfer (e.g., `20849217`). Operating-system configuration posts often refer to feature dependencies or system drivers, for example, of Linux distributions, or the use of programming languages like Python (e.g., `47013091`).

Discussion. Our results show that configuration-related security issues are highly diverse and may lead to exploits, highlighting the need for thorough security checks. In particular, considering the evolutionary changes to features and configurations is important, since these can cause varying and potentially unexpected behavior. Consequently, as systems change and evolve, configurations and dependencies must be adjusted to ensure a system's security. Tools for modeling the variability of a configurable system (e.g., feature models) can help address this problem, but we did not identify any mentioning of such in the posts. This may be due to developers' unawareness for such concepts, unfit tool support for some domains and programming languages, or that it was just not mentioned. However, exploring these causes and how to securely evolve configurations is subject to future work.

In our data, we noticed that many vulnerabilities seem to arise in conjunction with feature dependencies. This emphasizes the need

for secure dependency management, for instance, within configuration files or package managers. Closely related, we noticed that developers often seem to rely on third-party libraries and components. Therefore, it is important to consider such external dependencies and their relations to internal features to ensure the security of configurations. Specifically, these external libraries should be updated and checked regularly to avoid vulnerabilities.

Vulnerabilities and attacks are logically interrelated, which is why they are often addressed together. This becomes even more challenging in the context of configuring, which is why we argue that developers need a comprehensive understanding of security to protect against threats and assess the impact of configuration options. To emphasize, we argue that a better understanding of security-related concepts (e.g., vulnerability, attack, threat), their differences, and their analyses should be integrated into development processes via a security-engineering phase [50, 57, 58]. Ideally, this phase takes place in the initial analysis of relevant configurations prior to a release, which are assessed for severity as part of a comprehensive risk-management process. To introduce such a phase, practitioners can rely on industry standards like the ISO/IEC 27000 series [32].

We found that XSS and CSRF attacks are mentioned more often in the Stack Overflow posts we analyzed. There are various reasons why these two attacks are particularly relevant for configurable systems. For instance, misconfigurations can lead to inadequate input validation and missing or broken security headers, allowing malicious scripts or requests to be executed. Furthermore, user sessions may not be managed correctly regarding authentication and authorization, which allows CSRF attacks to successfully make unauthorized requests on behalf of the user. These attacks are primarily relevant for web applications, which we argue are an important subject for future work.

The various domains we identified (e.g. web application, data storages, mobile apps) emphasize the diversity of systems for which secure configuring is relevant. This poses challenges for practitioners and researchers to address the unique configuration challenges within such a domain. While the diverse domains already pose their own challenges, this situation becomes even more complicated seeing the increasing number of systems and configuration options that interact with each other. Particularly, that several posts seemed to address multiple configurations or systems indicates that modern systems often involve a combination of configurations, namely cross-configurations with their dependencies. For instance, the configurations of a web application (e.g., Spring) and its interacting web server (e.g., Apache Tomcat) typically contain dependencies. These may cause vulnerabilities in the communication between the systems that can enable attacks on both. Such cases challenge researchers to design novel techniques for analyzing and assessing the security of interconnected configurable systems. Accordingly, developers need a holistic view to ensure that the overall system is secure.

RQ$_3$: Security and Configuration in Context

The posts we identified focus on solving security issues, configurations of web application, as well as vulnerabilities and measures at implementation level. Moreover, the configuration and security issues are diverse and cause various challenges, for instance, regarding cross-dependencies, evolution, and verification.

5 THREATS TO VALIDITY

Internal Validity. We may have incorrectly interpreted or classified posts during our manual analysis. The posts are very diverse in terms of detail and terms used, which is the result of analyzing discussions of diverse users (e.g., different nationalities, educational backgrounds, language skills). This is a threat for any qualitative text analysis like ours. We aimed to mitigate this threat by involving multiple researchers in the analysis, validating the classification, and performing LDA to complement the manual reading. This also helped us minimize the threat of misinterpreting the suggested LDA topics in the topic modeling. Furthermore, we were quite strict in selecting posts to ensure sufficient quality, making sure that all inclusion criteria were met and that there was indeed a clear connection to vulnerabilities and configurations (i.e., based on question title, body, and tags). This, in turn, also means that we may have excluded some relevant posts and that some topics (e.g., Spring) occur more dominant than they actually are. We aimed to focus on posts clearly connected to our research goal to ensure relevance.

External Validity. We relied solely on Stack Overflow as our data source, which, however, is currently the largest CQA system related to software-engineering topics. Unfortunately, we could not retrieve all of Stack Overflow's posts, because the Google BigQuery data set covers data only until September 2022. However, the BigQuery data set is the largest external data set of Stack Overflow posts and using it avoids technical problems with crawling or searching the website. Furthermore, our search string does likely not cover all potential posts related to vulnerabilities and configurations. For these reasons, our sample of posts does not represent all concerns of developers around the world. Nonetheless, we have obtained a larger number of posts on our topic to derive valuable and reliable insights. This is supported by the fact that various trends have developed as expected (e.g., increasing number of vulnerabilities) and that we discovered recurring patterns (e.g., increased likelihood of internet-related vulnerabilities).

6 RELATED WORK

Several existing studies have been concerned with analyzing topics and trends on Stack Overflow. For instance, Barua et al. [10] proposed a statistical method and associated quantitative analysis to discover general topics and trends of developer discussions on Stack Overflow. Others have focused on more specific domains or (sub-)topics. For example, there are studies that analyze discussions on mobile development issues [46], docker development [27], or Android APIs [12]. Regarding security topics, most recent studies focus much more on rather broad issues. For instance, Yang et al. [92] and Lopez et al. [47] investigated and classified questions with the tag "security" to derive trends and challenges. Tahaei et al. [78] conducted a study on privacy-related questions. In addition, there are a few studies focusing on specific use cases, such as security practices for microservices [62] or eHealth applications [76]. Other studies related to security are based on source code included in the questions and answers, for instance, related to the security of Android applications [22] or vulnerabilities in Java code [45]. Lastly, there are two studies by Krüger [41, 42] that focused on CQA posts related to variability. Precisely, the author focused on what developers discuss in the context of separation of concerns

and related to software product lines. While all of such studies are connected to our work and we investigated their methodologies to define our own, they cover broader or different topics and posts, for instance, on identifying all possible questions, general security issues, or a specific domain like microservices. So, the related work covers a different body of knowledge, which is either out of our study's scope or does not cover our topics of interest. In summary, we contribute complementary and different insights that have not been addressed by the related work.

7 CONCLUSION

In this paper, we reported an exploratory study on configuration-related vulnerability issues discussed on Stack Overflow. Precisely, we contribute insights into security issues of configuring that the developers are confronted with in practice. We conducted a manual data analysis as well as an automated topic modeling to derive and validate our findings. The main findings of our study are:

- Vulnerabilities in the context of configuring are of growing interest in the developer community of Stack Overflow.
- Web, server, and storage development are particularly critical areas for dealing with configuration-related security issues.
- Developers discuss primarily solutions for security or web application issues, configurations, and vulnerability countermeasures at implementation rather than at concept level.
- The discussed issues are highly diverse and imply numerous opportunities for new solutions and research, including challenges surrounding cross-configurations, cross-dependencies, system evolution, and verification while also taking a secure engineering (i.e., configuration) process into account.

Through our study, we provide both an understanding of the relevance of secure configurations and an awareness of the impact of configurations on security. Throughout the paper, we identified and motivated various directions for future research. For instance, we are interested in comparing the security issues we found to entries in security data sources (e.g., the National Vulnerability Database), discussion on other CQA systems (e.g., Quora), and issues or bug reports on development platforms (e.g., GitHub) to get further insights into the contexts between vulnerabilities and configuring. Lastly, we see the clear need to enhance development methods for engineering secure configurable systems.

REFERENCES

[1] I. Abal, C. Brabrand, and A. Wasowski. 2014. 42 variability bugs in the Linux kernel: A qualitative analysis. In *Automated Software Engineering Conference*. ACM, 421–432.

[2] I. Abal, J. Melo, Ş. Stănciulescu, C. Brabrand, M. Ribeiro, and A. Wąsowski. 2018. Variability bugs in highly configurable systems: A qualitative analysis. *ACM Transactions on Software Engineering and Methodology* 26, 3 (2018), 1–34.

[3] M. Abomhara and G. M. Køien. 2015. Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks. *Journal of Cyber Security and Mobility* (2015), 65–88.

[4] M. Acher, G. Bécan, B. Combemale, B. Baudry, and J.-M. Jézéquel. 2015. Product lines can jeopardize their trade secrets. In *Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 930–933.

[5] A. Agrawal, W. Fu, and T. Menzies. 2018. What is wrong with topic modeling? And how to fix it using search-based software engineering. *Information and Software Technology* 98 (2018), 74–88.

[6] A. Ahmad, C. Feng, S. Ge, and A. Yousif. 2018. A survey on mining Stack Overflow: Question and answering (Q&A) community. *Data Technologies and Applications* 52, 2 (2018), 190–247.

[7] A. Albakri, H. Fatima, M. Mohammed, A. Ahmed, A. Ali, A. Ali, and N. M. Elzein. 2022. Survey on reverse-engineering tools for android mobile devices. *Mathematical Problems in Engineering* 2022 (2022), 1–7.

[8] S. Apel, D. Batory, C. Kästner, and G. Saake. 2013. *Feature-oriented software product lines*. Springer.

[9] A. Bamrara. 2015. Evaluating database security and cyber attacks: A relational approach. *The Journal of Internet Banking and Commerce* 20, 2 (2015).

[10] A. Barua, S. W. Thomas, and A. E. Hassan. 2014. What are developers talking about? An analysis of topics and trends in Stack Overflow. *Empirical Software Engineering* 19 (2014), 619–654.

[11] R. Beckett, A. Gupta, R. Mahajan, and D. Walker. 2017. A general approach to network configuration verification. In *Conference of the Special Interest Group on Data Communication*. ACM, 155–168.

[12] C. Beddiar, I. E. Khelili, N. Bounour, and A.-D. Seriai. 2020. Classification of android APIs posts: An analysis of developer's discussions on Stack Overflow. In *International Conference on Advanced Aspects of Software Engineering*. IEEE, 1–5.

[13] D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3, 1 (2003), 993–1022.

[14] D. Bringhenti, G. Marchetto, R. Sisto, and F. Valenza. 2023. Automation for network security configuration: State of the art and research Trends. *Comput. Surveys* (2023).

[15] M. Calder, M. Kolberg, E. H. Magill, and S. Reiff-Marganiec. 2003. Feature interaction: A critical review and considered Forecast. *Computer Networks* 41, 1 (2003), 115–141.

[16] I. Cernica and N. Popescu. 2019. Security evaluation of wordpress backup plugins. In *Conference on Control Systems and Computer Science*. IEEE, 312–316.

[17] R. Croft, Y. Xie, M. Zahedi, M. A. Babar, and C. Treude. 2022. An empirical study of developers' discussions about security challenges of different programming languages. *Empirical Software Engineering* 27 (2022), 1–52.

[18] Krzysztof Czarnecki, Paul Grünbacher, Rick Rabiser, Klaus Schmid, and Andrzej Wąsowski. 2012. Cool features and tough decisions: A comparison of variability modeling approaches. In *Working Conference on Variability Modelling of Software-Intensive Systems*. ACM, 173–182.

[19] A. A. Elkhail and T. Cerny. 2019. On relating code smells to security vulnerabilities. In *BigDataSecurity*. IEEE, 7–12.

[20] S. Fernandes and J. Bernardino. 2015. What is BigQuery?. In *International Database Engineering and Applications Symposium*. ACM, 202–203.

[21] D. Fernandez-Amoros, R. Heradio, C. Mayr-Dorn, and A. Egyed. 2022. Scalable sampling of highly-configurable systems: Generating random instances of the Linux kernel. In *Automated Software Engineering Conference*. 1–12.

[22] F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl. 2017. Stack Overflow considered harmful? The impact of copy&paste on Android application security. In *Symposium on Security and Privacy*. IEEE, 121–136.

[23] A. M. Gamundani and L. M. Nekare. 2018. A review of new trends in cyber attacks: A zoom into distributed database systems. In *IST-Africa*. IEEE, 1–9.

[24] P. Gazzillo. 2020. Inferring and Securing Software Configurations Using Automated Reasoning. In *Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 1517–1520.

[25] P. B. Gutgarts and A. Temin. 2010. Security-critical versus safety-critical software. In *International Symposium on Technologies for Homeland Security*. IEEE.

[26] A. Haleem, M. Javaid, and R. P. Singh. 2022. An era of ChatGPT as a significant futuristic support tool: A study on features, abilities, and challenges. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations* 2, 4 (2022), 100089.

[27] M. U. Haque, L. H. Iwaya, and M. A. Babar. 2020. Challenges in docker development: A large-scale study using Stack Overflow. In *Empirical Software Engineering and Measurement*. ACM, 1–11.

[28] M. Humayun, M. Niazi, N. Z. Jhanjhi, M. Alshayeb, and S. Mahmood. 2020. Cyber security threats and vulnerabilities: A systematic mapping study. *Arabian Journal for Science and Engineering* 45, 4 (2020).

[29] M. Hussain, A. Al-Haiqi, A. A. Zaidan, B. Bahaa Zaidan, M. Kiah, S. Iqbal, S. Iqbal, and M. Abdulnabi. 2018. A security framework for mHealth apps on Android platform. *Computers & Security* 75 (2018), 191–217.

[30] M. S. Iqbal, R. Krishna, M. A. Javidian, B. Ray, and P. Jamshidi. 2022. Unicorn: Reasoning about configurable system performance through the lens of causality. In *European Conference on Computer Systems*. ACM, 199–217.

[31] ISO/IEC 25010 2011. *Systems and software engineering – SQuaRE - System and software quality*. Standard. ISO.

[32] ISO/IEC 27000 2018. *Information technology – Security techniques – Information security management systems*. Standard. ISO.

[33] ISO/IEC 27001 2013. *Information Security Management Systems – Requirements*. Standard. ISO.

[34] ISO/IEC 27005 2022. *Information security, cybersecurity and privacy protection – Guidance on managing information security risks*. Standard. ISO.

[35] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. 1990. *Feature-oriented domain analysis feasibility study*. Technical Report CMU/SEI-90-TR-21. Carnegie Mellon University.

[36] A. Kenner, S. Dassow, C. Lausberger, J. Krüger, and T. Leich. 2020. Using variability modeling to support security evaluations: Virtualizing the right attack scenarios.

In *Working Conference on Variability Modelling of Software-Intensive Systems*. ACM, 1–9.
[37] A. Kenner, R. May, J. Krüger, G. Saake, and T. Leich. 2021. Safety, security, and configurable software systems: A systematic mapping study. In *Systems and Software Product Line Conference*. ACM, 148–159.
[38] R. A. Khan, S. U. Khan, H. U. Khan, and M. Ilyas. 2021. Systematic mapping study on security approaches in secure software engineering. *IEEE Access* 9 (2021), 19139–19160.
[39] B. A. Kitchenham, D. Budgen, and O. P. Brereton. 2015. *Evidence-based software engineering and systematic reviews*. CRC Press.
[40] V. Klotzman, F. Farmahinifarahani, and C. Lopes. 2021. Public software development activity during the pandemic. In *Empirical Software Engineering and Measurement*. 1–12.
[41] J. Krüger. 2018. Separation of concerns: Experiences of the crowd. In *Symposium On Applied Computing*. ACM, 2076–2077.
[42] J. Krüger. 2019. Are you talking about software product lines? An analysis of developer communities. In *Working Conference on Variability Modelling of Software-Intensive Systems*. ACM, 1–9.
[43] J. Krüger, I. Schröter, A. Kenner, and T. Leich. 2017. Empirical studies in question-answering systems: A discussion. In *International Workshop on Conducting Empirical Studies in Industry*. IEEE, 23–26.
[44] Y. Li and Q. Liu. 2021. A comprehensive review study of cyber-attacks and cyber security: Emerging trends and recent developments. *Energy Reports* 7 (2021), 8176–8186.
[45] S. A. Licorish and T. Nishatharan. 2021. Contextual profiling of Stack Overflow Java code cecurity vulnerabilities initial insights from a pilot study. In *International Conference on Software Quality, Reliability and Security*. IEEE, 1060–1068.
[46] M. Linares-Vásquez, B. Dit, and D. Poshyvanyk. 2013. An exploratory analysis of mobile development issues using Stack Overflow. In *Mining Software Repositories Conference*. IEEE, 93–96.
[47] T. Lopez, T. T. Tun, A. Bandara, M. Levine, B. Nuseibeh, and H. Sharp. 2018. An investigation of security conversations in Stack Overflow: Perceptions of security and community involvement. In *International Conference on Information and Communication Technologies for Sustainability*. ACM, 26–32.
[48] K. Ludwig, J. Krüger, and T. Leich. 2019. Covert and phantom features in annotations: Do they impact variability analysis?. In *Systems and Software Product Line Conference*. ACM, 218–230.
[49] R. May. 2022. Security and configurable storage systems in Industry 4.0 environments: A systematic literature study. In *Open Conference Proceedings*, Vol. 2. 151–156.
[50] R. May, C. Biermann, A. Kenner, J. Krüger, and T. Leich. 2023. A product-line-engineering framework for secure enterprise-resource-planning systems. In *International Conference on ENTERprise Information Systems*. Elsevier, 1–8.
[51] R. May, C. Biermann, J. Krüger, G. Saake, and T. Leich. 2022. A systematic mapping study of security concepts for configurable data storages. In *Systems and Software Product Line Conference*. ACM, 108–119.
[52] R. May, J. Gautam, C. Sharma, C. Biermann, and T. Leich. 2023. A systematic mapping study on security in configurable safety-critical systems based on product-line concepts. In *International Conference on Software Technologies*. SciTePress, 217–224.
[53] J. Meinicke, T. Thüm, R. Schröter, F. Benduhn, T. Leich, and G. Saake. 2017. *Mastering software variability with FeatureIDE*. Springer.
[54] J. Meinicke, C.-P. Wong, C. Kästner, T. Thüm, and G. Saake. 2016. On essential configuration complexity: Measuring interactions in highly-configurable systems. In *Conference on Automated Software Engineering*. ACM, 483–494.
[55] S. Meldrum, S. A. Licorish, and B. T. R. Savarimuthu. 2017. Crowdsourced knowledge on Stack Overflow: A systematic mapping study. In *Conference on Evaluation and Assessment in Software Engineering*. ACM, 180–185.
[56] P. Mell, K. Scarfone, and S. Romanosky. 2006. Common vulnerability scoring system. *IEEE Security & Privacy* 4, 6 (2006), 85–89.
[57] D. Mellado, E. Fernández-Medina, and M. Piattini. 2008. Towards security requirements management for software product lines: A security domain requirements engineering process. *Computer Standards & Interfaces* 30, 6 (2008), 361–371.
[58] D. Mellado, E. Fernández-Medina, and M. Piattini. 2010. Security requirements engineering framework for software product lines. *Information and Software Technology* 52, 10 (2010), 1094–1117.
[59] N. Meng, S. Nagy, D. Yao, W. Zhuang, and G. A. Argoty. 2018. Secure coding practices in Java: challenges and vulnerabilities. In *International Conference on Software Engineering*. ACM, 372–383.
[60] O. Mesa, R. Vieira, M. Viana, V. H. S. Durelli, E. Cirilo, M. Kalinowski, and C. Lucena. 2018. Understanding vulnerabilities in pPugin-based web systems: An exploratory study of wordpress. In *Systems and Software Product Line Conference*. ACM, 149–159.
[61] S. Nadi, T. Berger, C. Kästner, and K. Czarnecki. 2014. Mining configuration constraints: Static analyses and empirical results. In *International Conference on Software Engineering*. IEEE, 140–151.
[62] A. R. Nasab, M. Shahin, S. A. H. Raviz, P. Liang, A. Mashmool, and V. Lenarduzzi. 2023. An empirical study of security practices for microservices systems. *Journal of Systems and Software* 198 (2023), 111563.
[63] D. Nešić, J. Krüger, S. Stănciulescu, and T. Berger. 2019. Principles of feature modeling. In *Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 62–73.
[64] A. Nhlabatsi, R. Laney, and B. Nuseibeh. 2008. Feature interaction: The security threat from within software systems. *Progress in Informatics* 5, 75 (2008), 1.
[65] NIST SP 800-154 2016. *Guide to data-centric system threat modeling*. Standard. National Institute of Standards and Technology.
[66] NIST SP 800-30r1 2012. *Guide for conducting risk assessments*. Standard. National Institute of Standards and Technology.
[67] N. Onumah, S. Attwood, and R. Kharel. 2020. Towards secure application development: A cyber security centred holistic approach. In *International Symposium on Communication Systems, Networks and Digital Signal Processing*. IEEE, 1–6.
[68] K. Pohl, G. Böckle, and F. Van Der Linden. 2005. *Software product line engineering: Foundations, principles, and techniques*. Springer.
[69] S. Samonas and D. Coss. 2014. The CIA strikes back: Redefining confidentiality, integrity and availability in security. *Journal of Information System Security* 10, 3 (2014).
[70] M. Santolucito, E. Zhai, R. Dhodapkar, A. Shim, and R. Piskac. 2017. Synthesizing configuration file specifications with association rule learning. *Proceedings of the ACM on Programming Languages* 1 (2017), 1–20.
[71] I. Schaefer, R. Rabiser, D. Clarke, L. Bettini, D. Benavides, G. Botterweck, A. Pathak, S. Trujillo, and K. Villela. 2012. Software diversity: State of the art and perspectives. *International Journal on Software Tools for Technology Transfer* 14 (2012), 477–495.
[72] S. Sengupta and C. Haythornthwaite. 2020. Learning with comments: An analysis of comments and community on Stack Overflow. In *Hawaii International Conference on System Sciences*. IEEE, 2898–2907.
[73] I. Srba and M. Bieliková. 2016. A comprehensive survey and classification of approaches for community question answering. *ACM Transactions on the Web* 10, 3 (2016), 1–63.
[74] K. Stevens, P. Kegelmeyer, D. Andrzejewski, and D. Buttler. 2012. Exploring topic coherence over many models and many topics. In *Conference on Empirical Methods in Natural Language Processing*. ACL, 952–961.
[75] R. Sulatycki and E. B. Fernandez. 2015. Two threat patterns that exploit security misconfiguration and sensitive data exposure vulnerabilities. In *European Conference on Pattern Language of Programs*. IEEE, 1–11.
[76] M. Tahaei, J. Bernd, and A. Rashid. 2022. Privacy, permissions, and the health app ecosystem: A stack overflow exploration. In *European Symposium on Usable Security*. ACM, 117–130.
[77] M. Tahaei, T. Li, and K. Vaniea. 2022. Understanding privacy-related advice on Stack Overflow. *Proceedings on Privacy Enhancing Technologies* 2022, 2 (2022), 114–131.
[78] M. Tahaei, K. Vaniea, and N. Saphra. 2020. Understanding Privacy-related questions on Stack Overflow. In *Conference on Human Factors in Computing Systems*. ACM, 1–14.
[79] R. Tartler, D. Lohmann, J. Sincero, and W. Schröder-Preikschat. 2011. Feature consistency in compile-time-configurable system software: Facing the Linux 10,000 feature problem. In *European Conference on Computer Systems*. ACM, 47–60.
[80] T. Thüm, S. Apel, C. Kästner, I. Schaefer, and G. Saake. 2014. A classification and survey of analysis strategies for software product lines. *Computing Surveys* 47, 1 (2014), 1–45.
[81] H. Trunde and E. Weippl. 2015. Wordpress security: An analysis based on publicly available exploits. In *International Conference on Information Integration and Web-based Applications & Services*. ACM, 1–7.
[82] Á. J. Varela-Vaca, D. G. Rosado, L. E. Sánchez, M. T. Gómez-López, R. M. Gasca, and E. Fernandez-Medina. 2021. CARMEN: A framework for the verification and diagnosis of the specification of security requirements in cyber-physical systems. *Computers in Industry* 132 (2021), 1–14.
[83] D. Vecchiato, M. Vieira, and E. Martins. 2016. The perils of Android security configuration. *Computer* 49, 06 (2016), 15–21.
[84] I. von Nostitz-Wallwitz, J. Krüger, J. Siegmund, and T. Leich. 2018. Knowledge transfer from research to industry: A survey on program comprehension. In *International Conference on Software Engineering*. ACM, 300–301.
[85] R. Wang, Y. Zhou, S. Chen, S. Qadeer, D. Evans, and Y. Gurevich. 2013. Explicating {SDKs}: uncovering assumptions underlying secure authentication and authorization. In *Security*. USENIX, 399–314.
[86] W. Wang, S. Jian, Y. Tan, Q. Wu, and C. Huang. 2022. Representation learning-based network intrusion detection system by capturing explicit and implicit feature interactions. *Computers & Security* 112 (2022), 102537.
[87] S. J. Y. Weamie. 2022. Cross-site scripting attacks and defensive techniques: A comprehensive survey. *International Journal of Communications, Network and System Sciences* 15, 8 (2022), 126–148.
[88] Y. Wei, X. Sun, L. Bo, S. Cao, X. Xia, and B. Li. 2021. A comprehensive study on security bug characteristics. *Journal of Software: Evolution and Process* 33, 10 (2021), e2376.

[89] J. Xu and G. Russello. 2022. Automated security-focused network configuration management: State of the art, challenges, and future directions. In *nternational Conference on Software Engineering and Applications*. IEEE, 409–420.
[90] D. Yang, A. Hussain, and C. V. Lopes. 2016. From query to usable code: An analysis of Stack Overflow code snippets. In *Mining Software Repositories Conference*. ACM, 391–402.
[91] J. Yang, C. Hauff, A. Bozzon, and G.-J. Houben. 2014. Asking the right Question in collaborative Q&A systems. In *Conference on Hypertext and Social Media*. ACM, 179–189.
[92] X.-L. Yang, D. Lo, X. Xia, Z.-Y. Wan, and J.-L. Sun. 2016. What security questions do developers ask? A large-scale study of Stack Overflow posts. *Journal of Computer Science and Technology* 31 (2016), 910–924.
[93] S. Yuan, Y. Zhang, J. Tang, W. Hall, and J. B. Cabotà. 2020. Expert finding in community question answering: A review. *Artificial Intelligence Review* 53 (2020), 843–874.
[94] Y. Zheng and X. Zhang. 2013. Path sensitive static analysis of web applications for remote code execution vulnerability detection. In *International Conference on Software Engineering*. IEEE, 652–661.
[95] Y. Zhou and A. Sharma. 2017. Automated identification of security issues from commit messages and bug reports. In *Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 914–919.
[96] T. Zimmermann. 2016. Card-sorting: From text to themes. In *Perspectives on Data Science for Software Engineering*. Elsevier, 137–141.
[97] M. Zwilling, G. Klien, D. Lesjak, Ł. Wiechetek, F. Cetin, and H. N. Basim. 2022. Cyber security awareness, knowledge and behavior: A comparative study. *Journal of Computer Information Systems* 62, 1 (2022), 82–97.