What Challenges Do Developers Face in Securing Their Mobile Applications? An Examination of Questions From Stack Overflow

Anonymous Author(s)

***Abstract*—The widespread prevalence of smartphones and tablets has led to society's reliance on mobile devices for accessing resources and services through the use of mobile applications (apps). As part of their functionality, many apps process, store, and transmit sensitive data, such as personal, financial, and health data, among others. Hence, app developers must take the necessary precautions to protect their apps from malicious attacks and other vulnerabilities. Though research into software security, such as malware and vulnerabilities, is widespread, little is known about the challenges developers face in troubleshooting their mobile apps' security-related issues in a practical setting. To this extent, this paper examines the mobile app security questions developers ask on Stack Overflow. Through a combination of quantitative and qualitative techniques, we show that Stack Overflow is a popular venue for developers to ask mobile app security questions, with most questions related to Android apps. Additionally, we present that questions typically fall into seven categories Secured Communications, Database, App Distribution Service, Encryption, Permissions, File-Specific, and General Security concerns. We envision findings from this study will better educate developers on potential areas of concern when implementing/maintaining their apps and empower the research/vendor community with constructing tools/techniques to help developers secure their apps.**

I. Introduction

Advancements in tools and technologies, such as integrated development environments (e.g., Xcode, Android Studio, etc.) and software frameworks/libraries (e.g., OkHTTP, Firebase, etc.) have made it possible for almost anyone to create mobile applications (apps) without extensive programming knowledge or software engineering skills [1], [2]. Indeed, this is evident by the extensive volume of apps available on app distribution service providers like the Google Play Store and Apple App Store; as of Q3 2022, there are over 3 million and 1 million apps on the Google Play Store and Apple App Store, respectively [3]. This ease of app creation and publishing has led developers and organizations to implement apps of varying functionality. These apps can range from simple timezone conversion apps to more sophisticated apps that allow users to conduct online financial transactions and health and fitness apps that provide personalized information based on tracked user data.

However, this ease of app development also leads to the development of poorly designed and implemented apps. This, in turn, can negatively impact the user experience [4] and even result in end-users installing unsafe apps that expose sensitive data or act as a vector for malware [5]. While there are best practices, tools, and techniques to help developers and project teams construct high-quality, secure software systems [6], one should remember that mobile apps, even though technically software systems, have different characteristics from non-mobile systems. For instance, unlike non-mobile systems, mobile apps need to be user-centric, energy and resource conscious, constrained in functionality, secure, and portable, among other characteristics [7], [8]. Therefore, not all software development practices and challenges that typically apply to non-mobile systems also apply to mobile apps. Hence, the need to study mobile apps.

Mobile app development and maintenance provides the research community with the opportunity to investigate developer practices, identify challenges and propose improvements in multiple areas [9]–[12]. One such crucial area is app security. The reliance of apps on sensitive user information for their functionality makes it imperative that the research community understand mobile app security vulnerabilities and shortcomings. To this extent, researchers have investigated app security through multiple types of studies, such as mining software repositories, reverse engineering distribution packages, and developer and end-user surveys [13]–[15]. While empirical studies of source code and other related artifacts allow researchers to conduct extensive, reproducible studies with generalizable results, they also pose challenges. These include difficulty identifying causal relationships, and understanding context and background, among others. That said, existing work that provides insight into app security from the viewpoint of developers is lacking– these studies are limited to a specific organization, domain, or the number of developers participating in the survey [16]–[19]. Hence, we lack a large, diverse, and representative view of the real-world challenges developers frequently encounter when securing their mobile apps. Furthermore, we need to assess the gaps in the existing research on mobile app security and comprehend their causes. Therefore, in this study, we mine the questions developers ask on Stack Overflow about mobile app security. Stack Overflow is one of the largest online programming-specific question-answer sites, with over 20 million questions and users [20].

A. Motivating Example

Consider the question the developer asks in Quote 1. This type of security question is specific to mobile apps– it deals with challenges using the device's fingerprint sensor on different devices (i.e., app portability). This security issue is not usually seen in non-mobile systems, where in order to increase the adoption of their app, the developer has to support multiple

devices while not compromising on security. However, since smartphones are manufactured by multiple vendors, with these vendors possibly using different variants of hardware components, app portability becomes challenging. Hence, while general research on software security is essential, we also need to understand the intricacies of mobile app security, especially the real-world challenges developers face in securing their apps while supporting a large and diverse user base.

SecurityException while checking if fingerprints are enrolled in Samsung Phones

"I am using a lockscreen with fingerprint in my app. While it works seamlessly with other phones having fingerprint sensor, samsung users are facing some SecurityException as I can see in my google console reports. I am having a hard time figuring it out as I have no samsung phones to test. Till now it has happened in Galaxy J7 and Grand Prime Plus"

Quote 1: An example of a developer asking the community for help in solving an issue in securing their mobile app [21].

B. Goal & Research Questions

As mobile app development continues to evolve and smartphone/tablet adoption grows, so do the security risks and issues associated with the apps. Hence, it is imperative to understand the challenges mobile app developers face in real-world settings. Therefore, the goal of this study is to *understand the trends and challenges around developer discussions on securing their mobile apps*. We envision that findings from this study will help researchers and educators understand the gaps between the academic viewpoint of mobile app security and the challenges developers face in a practical setting. Additionally, we envision app developers better understanding and planning for areas in their apps that are prone to security issues. Our study aims at answering the following research questions (RQs):

RQ1: How have mobile security discussions on Stack Overflow grown over the years? This research question aims to understand the extent to which developers turn to the community for help with securing their mobile apps. Therefore, through this research question, we measure the yearly growth of mobile security questions on Stack Overflow and examine the types and growth of tags developers associate with these questions. The findings from this research question form a starting point for understanding the trends and challenges of mobile app security in a real-world setting.

RQ2: What security challenges do mobile developers face? Through this research question, we aim to discover the specific challenges mobile app developer face when securing their apps. Hence, we utilize natural language processing techniques, such as n-grams and topic modeling, to present a granular view of the challenging topics in mobile app security.

C. Contribution

The main contributions from this work are as follows:

- This study provides preliminary yet promising findings that expand our awareness of real-world mobile app security challenges. Through our analysis of trends and challenges, we discuss how our findings align with existing work and takeaways for the community.
- We make our dataset of mobile app security questions and their metadata, publicly available for replication and extension purposes.

D. Paper Structure

This paper is organized as follows: In Section II, as part of our related work, we report on studies that examine security discussions on Stack Overflow. Section III provides details about our investigation methodology, while Section IV answers our research questions by reporting on the results of our experiments. Section V provides an in-depth discussion and takeaways from our findings. Finally, Section VI reports on the threats to the validity of our study before our conclusion and future work in Section VII.

II. Related Work

This section reports on related studies. However, before describing the set of related works, we want to emphasize that the research community has made tremendous strides in mining Stack Overflow discussions to study multiple aspects of programming and software engineering [22]–[24]. That said, the studies highlighted in this subsection focus on research that mine Stack Overflow discussions for challenges developers face in securing their software systems.

First, we report on Stack Overflow studies where the authors either investigate or report on observations related to mobile app security. A qualitative analysis of 269 Stack Overflow posts about privacy and permissions in health apps for Android and iOS by Tahaei et al. [25] groups posts into multiple themes, with most questions around challenges in accessing data or resources. A manual analysis of 450 Android-related posts by Beyer and Pinzger [26] shows that most questions are related to the user interface of an app. The authors encounter security questions in their dataset, but the volume is far less than the other topics. The authors also propose a technique to classify posts automatically. In a study of security-related code snippets in Android-related posts, Fischer et al. [27] construct a machine-learning model to classify code snippets as either secure or insecure. Additionally, the authors analyze 1.3 million Android apps and report that 15.4% of the apps contain security-related code snippets from Stack Overflow.

Next, we report on studies that examine general mobile app development discussions on Stack Overflow. In their general study of mobile app discussions, Rosen and Shihab [12] identify multiple discussion topics. The authors report that AIP-related questions are a common challenge developers face. However, it is interesting to note that security is not included in their set of topics. The authors also show an increase in developers posting mobile-related questions to Stack Overflow and that these questions are more difficult to answer. Similarly, Linares-Vásquez et al. [28] identify a set of topics associated with mobile discussions; once more, the authors do not list security as a primary topic concerning mobile app development. An

analysis of 1,568,377 posts by Fontão et al. [29] on developer emotions shows that sadness, anger, and joy are the most common emotions involved when developers discuss mobile app development challenges.

Finally, we report on studies that analyze general security-related discussions on Stack Overflow. While Yang et al.'s [30] study on security questions on Stack Overflow highlights that these discussions include mobile security, the authors do not solely focus on the challenges developers face around mobile security. In fact, the authors only identify one type of issue associated with mobile security– app access. Lopez et al. [31] perform a thematic analysis on 20 questions and their related comments. The authors note that questions are often about a specific technology, with developers utilizing comments for responding to clarifications. Some popular areas developers need help with include storing passwords, SQL injections, and OAuth. In a subsequent study, the authors perform a qualitative examination of 20 questions and show that developers knowingly admit that security is a complex area of software systems, with little information available. They also highlight that there are instances where not all provided answers are correct. A topic modeling analysis by Tahaei et al. [32] on privacy-related questions yields 15 topics, with most questions around access control. The authors also observe developers asking privacy-related questions for Android apps. Licorish and Nishatharan [33] perform a pilot study on code snippets on Stack Overflow and report that most of these snippets contain security faults. At a high level, the authors note that snippets containing security vulnerabilities can result in various security breaches. Bayati and Heidary [34] quantitatively examine questions related to information security and report that most of these questions are asked by developers based in the United States. Furthermore, while the authors report that the most popular tags are Security, Authentication, and SSL, they also encounter numerous posts tagged as Android. In their study, Hong et al. [35] present a technique to detect insecure code snippets in Stack Overflow. The technique supports multiple programming languages and reports a 91% precision and 93% recall. In a similar study, Chen et al. [36] utilize social coding properties, such as relationships between users and badges, in addition to code content to automatically detect insecure code snippets in Stack Overflow. In a comparison analysis, the authors show that their technique outperforms other insecure code snippet detection models.

A. Summary

The research community has shown that mining Stack Overflow is an invaluable resource for studying real-world software engineering practices and challenges. Included in these studies are research works on mobile app development and maintenance. While there are studies on mobile app security, they are limited in their analysis. In contrast, our work exclusively analyzes a larger and more recent dataset of mobile app security discussions. In short, our work extends the body of knowledge on mobile app development and examines, from multiple viewpoints, the challenges developers face in securing their apps in a practical setting.

III. Experiment Design

In this section, we provide details about the methodology for our study. Figure 1 shows a high-level overview of our experiment, which we describe in detail below. Furthermore, the dataset we generate in this study is available on our project website for replication and extension purposes [37].

A. Dataset

To obtain the Stack Overflow posts for this study, we utilized Stack Exchange Data Explorer[1]; the official source for obtaining Stack Overflow data. The Data Explorer interface accepts a T-SQL query, and the resulting output can be exported as a CSV file. The interface also displays the database schema.

To acquire the set of mobile security posts, we execute three queries. The first query obtains all questions, while the other two retrieves all accepted and non-accepted answers. We extract all posts from the inception of Stack Overflow (September 2008) to October 2022. Similar to prior research [38], our search process involves querying the tag and title of questions, as developers concisely state the main objective of the question in the title [12]. Listing 1 shows the query we utilize to extract questions, which is explained below.

```
SELECT DISTINCT *
FROM Posts p
WHERE p.Tags LIKE '%security%'
  AND
  (
    p.Tags LIKE '%<android%'
    OR p.Tags LIKE '%<ios%'
  )
  OR
  (
  LOWER (p.Title) LIKE '% security %'
    AND
    (
    LOWER (p.Title) LIKE '% android %'
      OR LOWER (p.Title) LIKE '% ios %'
    )
  )
ORDER BY p.Id
```

Listing 1: T-SQL to extract questions based on the question's title and tag.

1) Tag Query: The tag search retrieves questions with tags containing '%security%' plus '%<android%' or '%<ios%'. The percentage sign (%) surrounding the tag indicates a wildcard search. Further, to exclude false positives, the less-than sign (<) indicates that the tag must start with the word 'android' or 'ios'. For example, when searching questions for Android and iOS without the less-than sign, questions with tags such as 'kiosk' appear because they have the characters 'ios' in the word; these questions may not be completely associated with mobile security. On the other

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

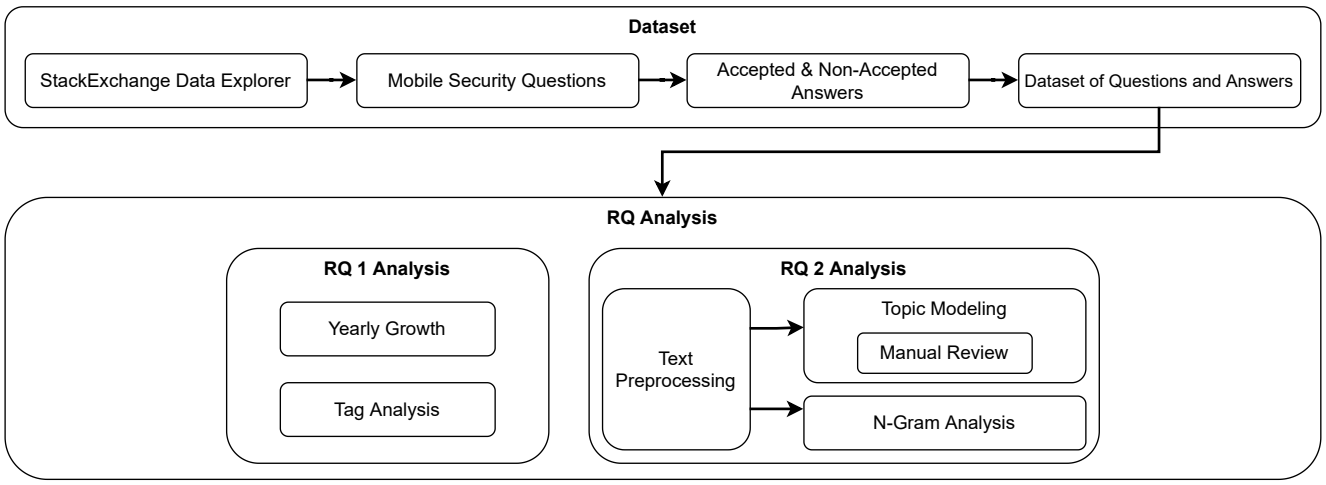

Fig. 1: Overview of our experiment design.

hand, by keeping the tag search open at the end of the word, such as '%<android%', the query can capture tags such as 'android-security'. Furthermore, a query constructed using only '%security%' and '%<android>%' would exclude questions with the 'android-security' tag, which has over 600 questions.

2) Title Query: The condition for querying titles is similar to the tag query. Our query retrieves questions where the title contains the word '% security %' plus '% android %' or '% ios %'. Note: We surround each query term with a white space to help exclude any false positives, similar to the less-than sign for the tag query.

We limit our query to Android and iOS, as these are the leading mobile operating systems [39], [40]. Furthermore, developers often include these terms when seeking help with a specific framework or library, which can be security-related.

B. RQ Analysis

Similar to prior research [38], we follow a mixed-method approach consisting of quantitative and qualitative investigations to answer our RQs. More specifically, our quantitative analysis involves using well-established statistical approaches and performing a topic modeling analysis. In our qualitative analysis, we manually examine a statistically significant sample of the data. The first RQ quantitatively analyzes the volume and growth of mobile security posts on Stack Overflow. In the second RQ, we utilize natural language processing (NLP) techniques to identify the areas with which mobile security developers seek assistance. In Section IV, as part of reporting our findings, we elaborate on our approach for each RQ.

IV. Results

In this section, we report on the findings of our experiments by answering our RQs. In the first RQ, we quantitatively examine and report on the volume of mobile app security-related questions. Next, the second RQ goes one step deeper and examines the security challenges these app developers face through quantitative and qualitative approaches. Due to space constraints, specific tables in the RQs show only the most frequently occurring instances; the complete set is online: [37].

A. *RQ1: How have mobile security discussions on Stack Overflow grown over the years?*

Motivation: This RQ provides insight into the extent to which mobile developers turn to the community for help with security challenges for their apps and how frequently they receive assistance. Furthermore, by analyzing the tags developers use, we gain a high-level understanding of the types and trends of common technologies and concepts developers find challenging.

Hence, this RQ is composed of two sub-RQs that examine the volume and growth of mobile security posts on Stack Overflow over time by analyzing specific attributes of the posts. $RQ_{1.1}$ investigates the volume of mobile security questions and their answers. $RQ_{1.2}$ investigates the type and growth of the common tags developers use for mobile security questions.

$RQ_{1.1}$: How have mobile security posts grown over time? Executing the queries described in Section III-A yields 5,786 questions related to mobile security, with a majority (i.e., 4,558 or 78.78%) of these questions receiving an answer. Looking at the answered questions, we observe that there are 2,371 (or 40.98%) questions with accepted answers and 2,187 (or 37.8%) questions with non-accepted answers. Finally, only 1,228 (or 21.22%) questions did not receive an answer.

Next, we look at the yearly breakdown of questions by examining the date when the question was posted. As shown in Figure 2 2016 witnessed the highest amount of mobile security questions (872, to be precise) posted by developers, followed by 2017 and 2015. Furthermore, the median number of yearly mobile security questions is 435, while the median number of questions receiving an answer in a year is 326.50. Furthermore, we also observe that each year shows more questions receiving an answer (green and orange bars) than questions receiving no response (blue bar). To understand the reason for the increase in questions in 2016, we reviewed the Title of the questions posted in 2015, 2016, and 2017. Our analysis shows

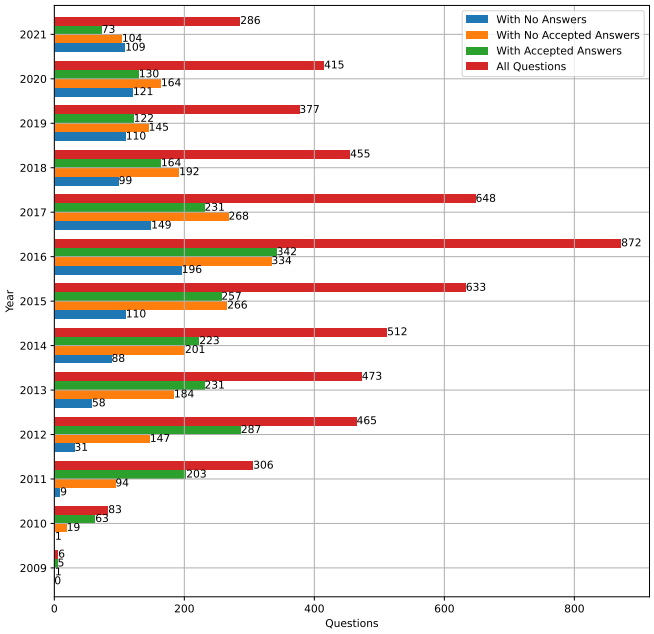

Fig. 2: Mobile security questions asked by developers each year.

TABLE I: Year-Over-Year growth of mobile security questions.

Year	Year-Over-Year Growth	
	All Questions	Answered Questions
2010	1283.33%	1266.67%
2011	268.67%	262.20%
2012	51.96%	46.13%
2013	1.72%	-4.38%
2014	8.25%	2.17%
2015	23.63%	23.35%
2016	37.76%	29.25%
2017	-25.69%	-26.18%
2018	-29.78%	-28.66%
2019	-17.14%	-25.00%
2020	10.08%	10.11%
2021	-31.08%	-39.80%

TABLE II: Top five occurring tags for mobile security questions.

Tag Name	Occurrence	Percentage
android	4,042	17.74%
security	3,630	15.94%
ios	1,613	7.08%
java	666	2.92%
android-security	633	2.78%
others	*12,195*	*53.54%*
Total	**22,779**	**100%**

a rise in questions around the Google Play Store, such as help resolving `X509TrustManager`[2] warnings. This aligns with Google Play blocking apps containing unsafe implementations `X509TrustManager` in May 2016 [41]. Additionally, iOS developers need help with `App Transport Security`[3] issues on iOS 9 or 10, which were released during this period and could explain the rise in questions on this topic.

Moving on, we look at the year-over-year growth of mobile security questions. From Table I, we see positive yearly growth of questions from the inception of Stack Overflow till 2016, after which there is a negative growth (i.e., the total number of questions posted in a specific year is less than the prior year), with the only exception being the year 2020. This shows that while developers need assistance with mobile security-related issues, there is a downward trend in the number of questions they are posting.

The final analysis for this sub-RQ examines the median time between a developer asking a question and receiving a response. We observe that it takes approximately 83.95 minutes (or 1.4 hours) for a mobile security question to receive its first answer. In contrast, in a study on refactoring questions on Stack Overflow, the authors observe a median time interval of 0.27 hours [38]. This comparison shows that answering mobile security questions requires developers with specialized knowledge, hence the relatively long wait for a response.

RQ$_{1.2}$*: What tags are utilized for mobile security questions?* This sub-RQ looks at the tags developers use to label mobile security questions. Tags help developers categorize and locate questions that are relevant to them. Stack Overflow maintains a predefined list of tags from which developers can select one or more to assign to a question. Our extracted dataset contains a total of 1,796 unique tags. From this set, we derive the number of times each tag occurs in the dataset. Table II shows the top 5 frequently occurring tags, with 'android' claiming first place with 4,042 (or 17.74%) instances, followed by 'security' with 3,630 (or 15.94%) instances, and 'ios' with 1,613 (or 7.08%) instances. Next, shown in Figure 3 is a visualization of the yearly growth of these top five frequently occurring tags. Similar to Figure 2, the spike in questions from 2015 to 2016 is also reflected in a spike of these tags, especially 'android'. Thus, at a high level, most questions in this period are related to Android security.

Even though developers utilize various tags for their mobile security questions, these tags can be grouped into specific categories. Similar to [38], we manually examined a statistically significant sample of frequently utilized tags and grouped related tags into specific categories. To this extent, we analyzed 413 distinct tags, which equates to a confidence level of 99% and a confidence interval of 5%. As part of the manual review, each author's annotation was reviewed by another to mitigate bias. Conflicts were resolved by discussion. As shown in Table III, our analysis yields 10 categories. Most of the tags are part of the Security Concepts/Features category, which includes general security concepts like 'oauth' and more mobile-specific features such as 'android-securityexception'. The next highest category is Framework/Library/API, which includes mobile and non-mobile technologies developers utilize in building apps, such as 'cordova' and 'angularjs'. Additionally, developers also use general Programming/SE tags such as 'debugging' and

[2]https://developer.android.com/reference/javax/net/ssl/X509TrustManager

[3]https://developer.apple.com/documentation/bundleresources/information_property_list/nsapptransportsecurity

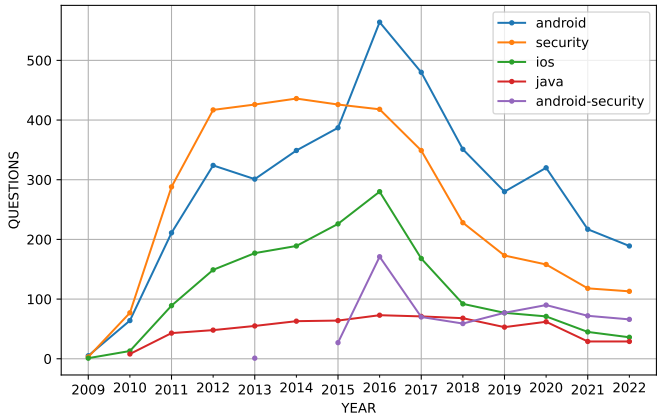

Fig. 3: Yearly growth of popular tags associated with mobile security questions.

TABLE III: Breakdown of the volume of instances for each tag category.

Tag Category	Count	Percentage
Security Concepts/Features	154	37.29%
Framework/Library/API	101	24.46%
Programming/Software Engineering Concepts	42	10.17%
Tools	29	7.02%
Operating System	25	6.05%
General Technology Concept	19	4.60%
Programming Language	12	2.91%
Hardware	12	2.91%
Storage	10	2.42%
General Mobile App Functionality	9	2.18%

'web-services'. We also observe developers using tags related to mobile and non-mobile app development tools such as 'xcode' and 'eclipse'. Similarly, developers also use Operating System tags like 'android' and 'macos'. The General Technology Concept category includes tags like 'browser' and 'screenshot', while the Programming Language category includes languages like 'java' and 'swift'. Developers also refer to hardware, which includes mobile devices, such as 'ipad' and 'samsung-mobile'. Finally, in the last two categories, Storage includes database technologies like 'sqlite', while General Mobile App Functionality includes features specific to mobile apps like 'in-app-billing' and 'push-notification'.

Summary for RQ1. While Stack Overflow is a popular venue to seek assistance with mobile security problems, there is a decrease in the yearly rate of questions developers ask. Most questions asked by developers are related to the security of Android apps, and developers use a variety of tags for their questions, most of which are security-related. Finally, significant changes by Google and Apple to the ecosystem usually result in a surge of questions.

B. *RQ2: What security challenges do mobile developers face?*

Motivation: The prior RQ shows Stack Overflow is a popular venue for asking mobile security questions and the types of tags developers utilize to categorize their questions. In this RQ, we utilize NLP techniques to gain a deeper understanding of mobile app developers' security challenges. To this extent, we analyze the body of the question. The question's body is written in natural language and hence will provide more insight, compared to tags, on the typical pain points developers face in securing their apps. Prior to analyzing the questions' body, we normalize the text (i.e., text preprocessing). Our activities include converting all characters to lowercase, removing code snippets, formatting tags, digits, and special characters, and expanding contractions. Additionally, we also remove stopwords, both standard and custom words.

Similar to RQ_1, this RQ incorporates two sub-RQs. $RQ_{2.1}$ examines the common words in questions, while $RQ_{2.2}$ groups related questions based on their primary topic.

***$RQ_{2.1}$**: What are the common words developers use when asking mobile security questions?*

Similar to [38], this sub-RQ performs an n-gram analysis. Specially, we examine the common unigrams, bigrams, and trigrams occurring in the body of questions. Unigrams are single words; bigrams are pairs of consecutive words occurring in a sentence; and trigrams are three consecutively occurring words. Unlike unigrams, bigrams and trigrams provide more context around the words. Table IV, shows the seven common unigrams, bigrams, and trigrams in our dataset. The occurrence of the unigram 'android' and Google service-based words is not surprising given our RQ_1 analysis showing a high prevalence of Android questions. However, these n-grams help us narrow down the problematic areas, more specifically, related to the Google Play Store and its services. The occurrence of the 'public_key' and 'private_key' bigrams shows that encryption is another pain point for mobile app developers. We also observe app permissions and credentials as other areas of concern. Additionally, developers face security-related problems with external resources, as seen by the bigrams 'web_service' and 'firebase_database' and the unigram 'server', which also explains the 'username_password' bigram. Finally, the trigram 'via_apps_infoplist' shows that iOS app developers likely need help resolving `App Transport Security` issues.

***$RQ_{2.2}$**: What are the primary topics associated with mobile security questions asked by developers?*

In this sub-RQ, we further examine the rationale for mobile developers to seek assistance in securing their apps by performing a topic modeling analysis on the body of the question using the Latent Dirichlet Allocation (LDA) algorithm [42]. This type of analysis is frequently utilized in similar studies (e.g., [12], [38], [43], [44]). Our approach involves running multiple execution cycles of the LDA analysis. We start with two topics and then increase the number of topics by one in each subsequent cycle until we have 50 topics, with each cycle having 150 passes and iterations. We determine the optimal model by calculating the topic coherence [45] and manually inspecting the output [46] of each model. Our analysis determined that the most optimum model is the seven-topic model. Since the LDA algorithm does not provide a name for the generated

TABLE IV: Top seven frequently occurring unigrams, bigrams, and trigrams in questions.

Unigram	Count	Percentage	Bigram	Count	Percentage	Trigram	Count	Percentage
android	4,293	1.63%	google_play	333	0.93%	google_play_store	54	1.03%
user	3,388	1.28%	public_key	230	0.64%	firebase_realtime_database	28	0.54%
data	2,733	1.04%	private_key	217	0.61%	google_play_console	24	0.46%
application	2,523	0.96%	web_service	211	0.59%	google_play_services	22	0.42%
server	2,463	0.93%	api_key	172	0.48%	developing_android_application	21	0.40%
key	2,222	0.84%	username_password	164	0.46%	via_apps_infoplist	19	0.36%
file	2,005	0.76%	firebase_database	119	0.33%	missing_insufficient_permissions	15	0.29%
others	24,4098	92.56%	*others*	*34,280*	*95.96%*	*others*	*5,037*	*96.50%*

TABLE V: Volume of questions for each LDA topic and reviewed by the authors, and a partial set of associated words.

Topic	Questions			Topic Representative Words
	Total Count	Total Percentage	Manually Reviewed	
General Security	3,120	53.92%	343	application, android, ios, implement, login, credentials, password, server, device, secure, api, token, client, access, user
Secured Communications	841	14.54%	265	certificate, error, ssl, http, transport, load, ats, https, network, domain, tls, cert, connect, url, webview
Databases	672	11.61%	245	database, firebase, rule, create, read, write, delete, firestore, db, query, data, add, update, id, uid, field
App Distribution Service	392	6.77%	195	google, play, version, apk, apps, vulnerability, warning, console, cordova, developer, store, error, policy, email, package, rejected
Encryption	328	5.67%	178	keys, keystore, encryption, private, string, public, encrypt, encrypted, algorithm, secret, keychain, decrypt, rsa, cipher, aes
Permissions	253	4.37%	153	permission, granted, denied, request, manifest, infoplist, securityexception, exception, intent, broadcast, sms
File Specific	180	3.11%	123	project, build, gradle, run, config, path, uri, file, folder, directory, image, videos, studio, ipa, apk
Total	*5,786*	*100.00%*	*1,502*	–

topics, the authors manually examined the distribution of words across the topics to determine the topic names. Additionally, to understand the distribution of questions among these topics, we assigned each question to its dominant topic. Furthermore, to help us contextualize the topic assignments, we reviewed 1,502 questions, a stratified statistically significant (95% confidence level and 5% confidence interval) sample of questions (in each topic). Table V shows a breakdown of the seven topics and the words typically associated with each topic. Questions on "General Security" practices and concerns occur the most, followed by more specialized topics like "Secured Communications", "Database", and so on. Based on our manual review, the subsections below describe each topic; additionally, we include a representative example for each topic.

Secured Communications. While mobile apps provide users with the flexibility of accessing various online services from their smartphone/tablet, it is essential that these apps provide the user with a secure environment to access these services. Mobile apps should secure the data stored locally on the device and securely communicate it to and from the device while preserving its integrity. Mobile apps can provide a secure connection using technologies such as SSL, TLS, and HTTPS that utilize digital certificates for identity verification.

Our examination of these questions shows that both Android and iOS app developers need help in this area. A common pain point for iOS app developers is with `App Transport Security`, such as either including or excluding specific domains. We also see app developers struggling with certificates, such as generating and accepting self-signed certificates. For example, in Quote 2, an iOS app developer faces an issue with their app accepting a self-signed server certificate. Additionally, we see questions about making and debugging SSL connections.

How to use NSURLConnection to connect with SSL for an untrusted cert?

"I have a simple code to connect to a SSL webpage, however, it gives an: untrusted server certificate. Is there a way to set it to accept connections anyway (just like in a browser you can press accept) or a way to bypass it?"

Quote 2: An example of an iOS secured communications question for assistance with NSURLConnection and SSL [47].

Database. A database forms an integral part of many mobile apps that require a mechanism to store and retrieve data, usually for offline functionality or caching to improve performance. This data can be in the form of user information and app data. While storing the data is important, what is equally important is securing the data to ensure that it is not accessed or compromised by unauthorized individuals or entities.

Our analysis of questions associated with this topic shows access control as a common problem for app developers. This includes issues related to database authentication and permissions to perform CRUD operations. Furthermore, developers face most of these security pain points when utilizing Firebase/Firestore. For example, in Quote 3, an app developer seeks help with providing an authenticated user read/write access to a Firebase database.

Firebase error: Permission denied. Unable to read/write from Firebase Database

"I need to give read/write access only to authenticated users, but it seems like Firebase is not recognizing that the user is authenticated. After I sign in the user with email and password, I am assuming user is authenticated and should be allowed to read/write from database. However permission is denied. Please help, how to authenticate a registered user to access database in Firebase."

Quote 3: An example of an app developer encountering database read/write permission issues [48].

App Distribution Service. These are services like the Google Play Store and Apple App Store that play a crucial role in giving app developers a platform to publish and make their apps available for end-users to discover and download. However, to protect end-user data and privacy from malicious, fraudulent, and unsafe apps (i.e., apps with security vulnerabilities), most app stores enforce security policies that apps must adhere to before they are made available to end-users.

Examination of questions shows that developers seek assistance resolving rejections and warnings from app stores like the Google Play Store. These warnings are due to developers having security vulnerabilities in their apps (e.g., JavaScript Interface Injection Vulnerability). Most vulnerabilities are associated with third-party libraries the app uses, but there are also instances of developers incorrectly using the APIs of the mobile operating system. Developers are unsure as to how to fix these vulnerability issues and usually include a code snippet, log trace, and build script with their question, an example of which is provided in Quote 4.

Google Play Security Alert - Your app is using an unsafe implementation of the HostnameVerifier

"Recently one of my app got a security alert from Google Play: 'You app is using an unsafe implementation of the HostnameVerifier'. And refer a link to Google Play Help Center article for details regarding to fixing and deadline of vulnerability. Below is my code. Anyone can explain with example what changes should I do to fix this warning?"

Quote 4: An example of a developer seeking help in resolving a Google Play Store security policy warning for their app [49].

Encryption. While a database provides a medium for an app to store its data, sensitive information must be encoded (i.e., encrypted) to only be readable by authorized entities. Furthermore, multiple encryption algorithms are available for developers to utilize in their apps to secure data transmitted over networks or stored on the device.

While our analysis of the questions reveals developers struggling with data encryption and decryption (e.g., Quote 5), we also observe that a common pain point for Android and iOS app developers is the generation and storage of keys (usually private keys). Furthermore, app developers need help using various encryption algorithms (e.g., RAS, AES, and DES) to secure data. We also observe Android developers needing help retrieving keys from the KeyStore.

How to encrypt and decrypt file in Android?

"I want to encrypt file and store it in SD card. I want to decrypt that encrypted file and store it in SD card again. I have tried to encrypt file by opening it as file stream and encrypt it but it is not working. I want idea how to do this."

Quote 5: An example of an Android app developer requesting help with data encryption and decryption [50].

Permission. To further protect user data and privacy, mobile operating systems require apps to specify the type of permissions they require to access certain information and resources. Failure to do so results in the app being denied access to these features, limiting its functionality and possibly causing a crash.

Our analysis of these questions shows that most developers need help resolving permission-related exceptions (e.g., `java.lang.SecurityException`) their apps throw when end-users interact with it. We observe most exceptions occurring due to developers not being aware they need to request specific permissions for the app's action. We also observe developers facing issues due to bad configurations, such as missing or incorrect entries in the AndroidManifest file and needing help defining custom permissions for Android apps. For example, in Quote 6, the developer is running into an exception due to a missing permission.

Android Security Exception while accessing contacts

"This is totally weird and I've searched through the forums. In my main class I have a button onClick will launch the contacts application as shown below. When I click the button, the contacts list is shown but as soon as I tap on a contact a security exception is thrown. I've checked the manifest and have tried all combinations of placing the uses-permission tag, within the application, activity etc. But nothing works. Any help will be greatly appreciated."

Quote 6: An example of a developer running into a permission exception due to missing permission [51].

File-Specific. Mobile app developers utilize multiple file types, in addition to the source code files, to construct apps. These include resource, build, data, certificate, and downloaded files, among others. Understanding how to integrate and manage these files can be challenging for developers. Some issues we observe include help with resolving zip path traversal vulnerabilities, resolving security/permission exceptions when downloading files, making configurations to correct or improve the app's security policy, and securing files. For example, in Quote 7, the developer needs help understanding how to secure the app's resource files. Finally, it is interesting to note that even though some of the challenges in this topic are also related to other topics (e.g., encryption and permissions), the association

of files with app security is still a substantial challenge that deserves a topic of its own.

A 3rd party application reads my asset folder

"A 3rd party security application reads into my application. They probably read my asset folder. How is this possible? I thought that the sandbox model prevented from external access to the internal data structure of an app?"

Quote 7: An example of a developer needing help with protecting/securing files that are part of the app [52].

General Security. While the other topics mentioned above are related to specific areas of mobile app security, these are not the only security concerns mobile app developers face. In this topic cluster, we encounter security challenges that do not occur frequently enough to be represented in separate topics. This topic includes a number of questions on authentication and access control, such as storing credentials and incorporating OAuth-based authentication with sites like Facebook and Google. We also see questions on validating and sanitizing user input to prevent client-side injections, protecting artifacts owned or utilized by the app on the device, and working with the lock screen of the device.

Interestingly, the questions are not limited to the functionality of the app. We also see developers seeking help to obfuscate the app's source code to protect hardcoded sensitive data, such as passwords. Additionally, we observe questions on the app's distribution package file (i.e., apk and ipa), such as signature generation and verification, protecting from piracy, and preventing reverse engineering, as shown in Quote 8.

How to avoid reverse engineering of an APK file

"I am developing a payment processing app for Android, and I want to prevent a hacker from accessing any resources, assets or source code from the APK file. Now my questions are: How can I completely prevent reverse engineering of an Android APK? Is this possible? How can I protect all the app's resources, assets and source code so that hackers can't hack the APK file in any way? Is there a way to make hacking more tough or even impossible? What more can I do to protect the source code in my APK file?"

Quote 8: An example of a developer seeking help with securing the app's distribution package file [53].

Summary for RQ2. When it comes to mobile security, we show that developers turn to Stack Overflow for assistance for specific reasons, such as Secured Communications, Database, App Distribution Service, Encryption, Permissions, and File-Specific. Additionally, developers need help with other General Security practices/concerns, including code obfuscation and securing app distribution packages.

V. Discussion

As an exploratory study, our research aims to understand the extent to which developers turn to the community for help addressing security problems with their mobile apps and the typical challenges they require assistance with. Our findings not only expand the body of knowledge in mobile app development and maintenance but also highlight areas for further research. In this section, we discuss how our work extends and aligns with existing research on mobile app security. We also discuss the implications of our findings as a series of takeaways.

This study shows that mobile app developers face a variety of challenges related to securing their apps and turn to the developer community for help. In fact, our study confirms and extends findings by Yang et al.'s [30] study, showing that Stack Overflow is a popular venue for developers to discuss software security-related issues. However, while Yang et al.'s study is not specific to a particular computing environment, our work focuses purely on mobile apps. In fact, in our RQ2 findings, we show that some of the topics that Yang et al. identify as being web security (e.g., injection vulnerabilities, SSL, certificates, etc.) are also security challenges that mobile app developers face. Other similarities we observe include databases, encryption, credentials, and access control.

The sudden spike in question we observe in RQ1 aligns with the behavior reported by Linares-Vásquez et al. [54], where the authors observe API changes in the Android SDK cause an increase of questions. In our study, we witnessed a spike in iOS questions with the release of the App Transport Security API and Android questions related to X509TrustManager.

Our work also confirms findings from multiple empirical studies on Android apps. Our observation that permissions are an issue for app developers aligns with findings by Scoccia et al. [55], which show that mobile apps frequently contain permission-related problems introduced by novice and experienced developers. Likewise, work by Gao et al. [56] shows that Android apps typically contain vulnerabilities related to SSL, permissions, encryption, and KeyStore, among others. Additionally, the authors show that third-party libraries are the main contributors to vulnerabilities; this aligns with our findings showing developers needing assistance with resolving warnings and rejections from App Distribution Services due to vulnerable third-party libraries the app utilizes. Findings by Iwaya et al. [57] on mobile health apps also show security concerns related to permissions, disclosure of credentials, unencrypted user data, etc., usually caused by insecure programming. Likewise, Chen et al. [58] show that mobile banking apps are also subject to security issues like storing, encrypting, and transmitting data, as well as third-party library vulnerabilities. While most studies focus on Android apps, Rahkema and Pfahl [59] show that iOS app developers are also prone to using third-party libraries with vulnerabilities; again, this is in alignment with our findings.

Next, we discuss how the findings from our RQs support the community (i.e., developers/project teams, tool/IDE vendors, researchers, and educators) through a series of takeaways.

Takeaway 1 - ***Proactively keeping app dependencies up to date through automation.*** Even though developers may follow secure coding practices when implementing their app [60], vulnerabilities contained in third-party libraries the app utilizes negatively impact the app's overall security [61]. To this extent, developers should proactively keep their app dependencies up to date to ensure that their apps utilize patched/fixed vulnerabilities present in these dependencies. Further, using recent versions of libraries will help developers incorporate up-to-date industry standards and regulations, such as GDPR and HIPAA. Finally, IDE vendors should work with the research community to integrate automated library upgrading techniques/tools [62], [63] into their products to allow developers to make changes during implementation. Project teams should also incorporate these tools with their build process and version control system [64] for immediate notifications of library upgrade opportunities.

Takeaway 2 - ***Code reviews should accompany the usage of security tools.*** While tools and techniques exist to detect app vulnerabilities [14], these tools should not be treated as a one-stop solution. For instance, as discussed in RQ2, developers can hardcode sensitive details in configuration files or the source code (such as authentication keys), which can be retrieved by reverse engineering the app [65]. Furthermore, developers tend to document sub-optimal (shortcuts/workarounds) implementation decisions in their code via comments, known as "self-admitted technical debt" [66]. These sub-optimal decisions may relate to insecure programming that code reviews may identify that code quality tools might miss. Code reviews also aid in the detection of security issues early in the development process, reducing the chance of releasing vulnerable apps to end-users.

Takeaway 3 - ***Mobile security focus areas.*** Our findings represent the practical challenges developers face in securing their apps and provide educators, researchers, and project teams with insight into specific mobile security focus areas. These findings can improve existing training material and courses to ensure developers have a more in-depth understanding of how and why it is crucial to secure apps and the repercussions of app vulnerabilities. The findings also provide the research community with opportunities to build or improve code quality tools and techniques to support app developers, such as expanding the catalog of mobile security smells. Developers and project teams can utilize our findings to understand and plan for problematic areas in securing their apps. Furthermore, the importance of training developers on app security is highlighted by Aljedaani et al. [16]– the authors show that most mobile health app developers complain of a lack of security knowledge and a lack of security experts on their teams.

VI. Threats To Validity

Even though there are other online question-and-answer sites and forums, we limit our analysis to only Stack Overflow, the largest technology/programming-related question-and-answer site with an extensive user base [20]. Furthermore, our data extraction queries are not limited to a specific date range. Therefore, the findings we present in our study are a snapshot of mobile app security topics and trends at this point in time, providing future studies with a platform for examining the evolution of mobile app security.

While our data extraction query is limited to Android and iOS questions, these two mobile operating systems make up the largest market share. However, there can be instances of security issues specific to other mobile operating systems that we did not capture in our study. Furthermore, to mitigate false positives questions in our dataset, we query the title and tag of the questions and include the term 'security' in our query, ensuring that we capture questions specifically related to mobile security. Our analysis is constrained to analyzing only the most recent version of a question, as is the case with similar studies.

In RQs that involve a manual review of data, we utilize a statistically significant random sample. Even so, our review sample may not contain important data items. Furthermore, we mitigate reviewer bias by performing peer reviews on our annotations. Since our review process involves resolving all conflicts through discussion, we do not calculate inter-annotator agreements. Finally, similar to prior studies, we use standard statistical measures and algorithms (e.g., LDA); however, there are other techniques we can utilize (e.g., HLTA).

VII. Conclusion & Future Work

Mobile apps play an essential part in people's lives, from simple tasks, such as sending an email, to more complex tasks, such as online banking. With most apps interacting with sensitive user information to achieve their desired functionality, developers must take the necessary precautions to secure their apps from issues that make them vulnerable to malicious attacks and data leaks. However, securing mobile apps can be challenging for developers and may require them to seek help from the community. In this study, we perform a mixed-methods examination of questions related to mobile app security that developers ask on Stack Overflow. We show that Stack Overflow is a popular venue for seeking help with app security, especially when ecosystem vendors make changes impacting common features, and developers typically receive a response in less than an hour. We also show that most questions are related to Android apps. Finally, a topic modeling analysis shows questions falling into seven categories– Secured Communications, Database, App Distribution Service, Encryption, Permissions, File-Specific, and General Security concerns. Our future work involves determining the types of questions that are challenging to answer and those that are popular among the community. We also plan on examining the users that ask and answer mobile security questions to determine if specific users are responsible for assisting the community. We also plan to reach out to mobile app developers to validate the findings of this study; this would provide a more comprehensive and accurate picture of the current state of mobile app security.

REFERENCES

[1] G. Allen, *Android for Absolute Beginners: Getting Started with Mobile Apps Development Using the Android Java SDK*. Apress, 2021.

[2] A. Sahar and C. Clayton, *iOS 15 Programming for Beginners: Kickstart your mobile app development journey by building iOS apps with Swift 5.5 and Xcode 13*. Packt Publishing, 2021.

[3] "Biggest app stores in the world 2022 — statista." https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/.

[4] H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan, "What do mobile app users complain about?," *IEEE Software*, vol. 32, no. 3, pp. 70–77, 2015.

[5] A. Qamar, A. Karim, and V. Chang, "Mobile malware attacks: Review, taxonomy & future directions," *Future Generation Computer Systems*, vol. 97, pp. 887–909, 2019.

[6] R. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 2019.

[7] A. Aldayel and K. Alnafjan, "Challenges and best practices for mobile application development: Review paper," in *Proceedings of the International Conference on Compute and Data Analysis*, ICCDA '17, (New York, NY, USA), p. 41–48, Association for Computing Machinery, 2017.

[8] H. K. Flora, X. Wang, and S. V. Chande, "An investigation on the characteristics of mobile applications: A survey study," *International Journal of Modern Education and Computer Science*, vol. 6, no. 11, pp. 21–27, 2014.

[9] L. Li, T. F. Bissyandé, M. Papadakis, S. Rasthofer, A. Bartel, D. Octeau, J. Klein, and L. Traon, "Static analysis of android apps: A systematic literature review," *Information and Software Technology*, vol. 88, pp. 67–95, 2017.

[10] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman, "A survey of app store analysis for software engineering," *IEEE Transactions on Software Engineering*, vol. 43, no. 9, pp. 817–847, 2017.

[11] P. Kong, L. Li, J. Gao, K. Liu, T. F. Bissyandé, and J. Klein, "Automated testing of android apps: A systematic literature review," *IEEE Transactions on Reliability*, vol. 68, no. 1, pp. 45–66, 2019.

[12] C. Rosen and E. Shihab, "What are mobile developers asking about? a large scale study using stack overflow," *Empirical Software Engineering*, vol. 21, pp. 1192–1223, Jun 2016.

[13] M. Xu, C. Song, Y. Ji, M.-W. Shih, K. Lu, C. Zheng, R. Duan, Y. Jang, B. Lee, C. Qian, S. Lee, and T. Kim, "Toward engineering a secure android ecosystem: A survey of existing techniques," *ACM Comput. Surv.*, vol. 49, aug 2016.

[14] F. Ebrahimi, M. Tushev, and A. Mahmoud, "Mobile app privacy in software engineering research: A systematic mapping study," *Information and Software Technology*, vol. 133, p. 106466, 2021.

[15] A. Peruma, J. Palmerino, and D. E. Krutz, "Investigating user perception and comprehension of android permission models," in *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, MOBILESoft '18, (New York, NY, USA), p. 56–66, Association for Computing Machinery, 2018.

[16] B. Aljedaani, A. Ahmad, M. Zahedi, and M. A. Babar, "An empirical study on developing secure mobile health apps: The developers' perspective," in *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*, pp. 208–217, 2020.

[17] D. van der Linden, P. Anthonysamy, B. Nuseibeh, T. T. Tun, M. Petre, M. Levine, J. Towse, and A. Rashid, "Schrödinger's security: Opening the box on app developers' security rationale," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pp. 149–160, 2020.

[18] C. Weir, A. Rashid, and J. Noble, "How to improve the security skills of mobile app developers: comparing and contrasting expert views," 2016.

[19] R. Balebako, A. Marsh, J. Lin, J. Hong, and L. Cranor, "The privacy and security behaviors of smartphone app developers," 01 2014.

[20] https://stackexchange.com/sites?view=list#users.

[21] "android - securityexception while checking if fingerprints are enrolled in samsung phones - stack overflow." https://stackoverflow.com/questions/48542621/securityexception-while-checking-if-fingerprints-are-enrolled-in-samsung-phones.

[22] S. Meldrum, S. A. Licorish, and B. T. R. Savarimuthu, "Crowdsourced knowledge on stack overflow: A systematic mapping study," in *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, EASE'17, (New York, NY, USA), p. 180–185, Association for Computing Machinery, 2017.

[23] A. Ahmad, C. Feng, S. Ge, and A. Yousif, "A survey on mining stack overflow: question and answering (q&a) community," *Data Technologies and Applications*, 2018.

[24] S. Meldrum, S. A. Licorish, and B. T. R. Savarimuthu, "Exploring research interest in stack overflow–a systematic mapping study and quality evaluation," *arXiv preprint arXiv:2010.12282*, 2020.

[25] M. Tahaei, J. Bernd, and A. Rashid, "Privacy, permissions, and the health app ecosystem: A stack overflow exploration," in *Proceedings of the 2022 European Symposium on Usable Security*, EuroUSEC '22, (New York, NY, USA), p. 117–130, Association for Computing Machinery, 2022.

[26] S. Beyer and M. Pinzger, "A manual categorization of android app development issues on stack overflow," in *2014 IEEE International Conference on Software Maintenance and Evolution*, pp. 531–535, IEEE, 2014.

[27] F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl, "Stack overflow considered harmful? the impact of copy&paste on android application security," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 121–136, 2017.

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

[29] A. Fontão, O. M. Ekwoge, R. Santos, and A. C. Dias-Neto, "Facing up the primary emotions in mobile software ecosystems from developer experience," in *Proceedings of the 2nd Workshop on Social, Human, and Economic Aspects of Software*, WASHES '17, (New York, NY, USA), p. 5–11, Association for Computing Machinery, 2017.

[30] X.-L. Yang, D. Lo, X. Xia, Z.-Y. Wan, and J.-L. Sun, "What security questions do developers ask? a large-scale study of stack overflow posts," *Journal of Computer Science and Technology*, vol. 31, no. 5, pp. 910–924, 2016.

[31] T. Lopez, T. T. Tun, A. Bandara, M. Levine, B. Nuseibeh, and H. Sharp, "An investigation of security conversations in stack overflow: Perceptions of security and community involvement," in *Proceedings of the 1st International Workshop on Security Awareness from Design to Deployment*, SEAD '18, (New York, NY, USA), p. 26–32, Association for Computing Machinery, 2018.

[32] M. Tahaei, K. Vaniea, and N. Saphra, "Understanding privacy-related questions on stack overflow," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, (New York, NY, USA), p. 1–14, Association for Computing Machinery, 2020.

[33] S. A. Licorish and T. Nishatharan, "Contextual profiling of stack overflow java code security vulnerabilities initial insights from a pilot study," in *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 1060–1068, 2021.

[34] S. Bayati and M. Heidary, "Information security in software engineering, analysis of developers communications about security in social q&a website," in *Intelligence and Security Informatics: 11th Pacific Asia Workshop. PAISI 2016, Auckland, New Zealand, April 19, 2016, Proceedings 11*, pp. 193–202, Springer, 2016.

[35] H. Hong, S. Woo, and H. Lee, "Dicos: Discovering insecure code snippets from stack overflow posts by leveraging user discussions," in *Annual Computer Security Applications Conference*, ACSAC '21, (New York, NY, USA), p. 194–206, Association for Computing Machinery, 2021.

[36] L. Chen, S. Hou, Y. Ye, T. Bourlai, S. Xu, and L. Zhao, "Itrustso: An intelligent system for automatic detection of insecure code snippets in stack overflow," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '19, (New York, NY, USA), p. 1097–1104, Association for Computing Machinery, 2020.

[37] https://sites.google.com/view/mobilesoft2023-stackoverflow.

[38] A. Peruma, S. Simmons, E. A. AlOmar, C. D. Newman, M. W. Mkaouer, and A. Ouni, "How do i refactor this? an empirical study on refactoring trends and topics in stack overflow," *Empirical Software Engineering*, vol. 27, p. 11, Oct 2021.

[39] "Mobile & tablet operating system market share worldwide — statcounter global stats." https://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/#monthly-200901-202209-bar.

[40] "Global mobile os market share 2022 — statista." https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/.

[41] "Google security alert: Unsafe implementation of the interface x509trustmanager." https://blog.axway.com/

learning-center/software-development/api-development/google-security-alert-unsafe-implementation-of-the-interface-x509trustmanager.

[42] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, p. 993–1022, Mar. 2003.

[43] A. Barua, S. W. Thomas, and A. E. Hassan, "What are developers talking about? an analysis of topics and trends in stack overflow," *Empirical Software Engineering*, vol. 19, no. 3, pp. 619–654, 2014.

[44] I. K. Villanes, S. M. Ascate, J. Gomes, and A. C. Dias-Neto, "What are software engineers asking about android testing on stack overflow?," in *Proceedings of the 31st Brazilian Symposium on Software Engineering*, pp. 104–113, 2017.

[45] M. Röder, A. Both, and A. Hinneburg, "Exploring the space of topic coherence measures," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, (New York, NY, USA), pp. 399–408, ACM, 2015.

[46] C. Sievert and K. Shirley, "Ldavis: A method for visualizing and interpreting topics," in *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pp. 63–70, 2014.

[47] "ios - how to use nsurlconnection to connect with ssl for an untrusted cert? - stack overflow." https://stackoverflow.com/questions/933331/how-to-use-nsurlconnection-to-connect-with-ssl-for-an-untrusted-cert.

[48] "android - firebase error: Permission denied. unable to read/write from firebase database - stack overflow." https://stackoverflow.com/questions/39584090/firebase-error-permission-denied-unable-to-read-write-from-firebase-database.

[49] "android - google play security alert - your app is using an unsafe implementation of the hostnameverifier - stack overflow." https://stackoverflow.com/questions/40928435/google-play-security-alert-your-app-is-using-an-unsafe-implementation-of-the-h.

[50] "security - how to encrypt and decrypt file in android? - stack overflow." https://stackoverflow.com/questions/4275311/how-to-encrypt-and-decrypt-file-in-android.

[51] "Android security exception while accessing contacts - stack overflow." https://stackoverflow.com/questions/10443615/android-security-exception-while-accessing-contacts.

[52] https://stackoverflow.com/questions/48770008/a-3rd-party-application-reads-my-asset-folder.

[53] https://stackoverflow.com/questions/13854425/how-to-avoid-reverse-engineering-of-an-apk-file.

[54] M. Linares-Vásquez, G. Bavota, M. Di Penta, R. Oliveto, and D. Poshyvanyk, "How do api changes trigger stack overflow discussions? a study on the android sdk," in *Proceedings of the 22nd International Conference on Program Comprehension*, ICPC 2014, (New York, NY, USA), p. 83–94, Association for Computing Machinery, 2014.

[55] G. L. Scoccia, A. Peruma, V. Pujols, I. Malavolta, and D. E. Krutz, "Permission issues in open-source android apps: An exploratory study," in *2019 19th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, pp. 238–249, 2019.

[56] J. Gao, L. Li, P. Kong, T. F. Bissyandé, and J. Klein, "Understanding the evolution of android app vulnerabilities," *IEEE Transactions on Reliability*, vol. 70, no. 1, pp. 212–230, 2021.

[57] L. H. Iwaya, M. A. Babar, A. Rashid, and C. Wijayarathna, "On the privacy of mental health apps," *Empirical Software Engineering*, vol. 28, no. 1, pp. 1–42, 2023.

[58] S. Chen, L. Fan, G. Meng, T. Su, M. Xue, Y. Xue, Y. Liu, and L. Xu, "An empirical assessment of security risks of global android banking apps," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pp. 1310–1322, 2020.

[59] K. Rahkema and D. Pfahl, "Quality analysis of ios applications with focus on maintainability and security," in *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 602–606, 2022.

[60] S. M. Dye and K. Scarfone, "A standard for developing secure mobile applications," *Computer Standards & Interfaces*, vol. 36, no. 3, pp. 524–530, 2014.

[61] T. Watanabe, M. Akiyama, F. Kanei, E. Shioji, Y. Takata, B. Sun, Y. Ishi, T. Shibahara, T. Yagi, and T. Mori, "Understanding the origins of mobile app vulnerabilities: A large-scale measurement study of free and paid apps," in *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, pp. 14–24, 2017.

[62] P. T. Nguyen, J. Di Rocco, R. Rubei, C. Di Sipio, and D. Di Ruscio, "Deeplib: Machine translation techniques to recommend upgrades for third-party libraries," *Expert Systems with Applications*, vol. 202, p. 117267, 2022.

[63] D. C. Nguyen, E. Derr, M. Backes, and S. Bugiel, "Up2dep: Android tool support to fix insecure code dependencies," in *Annual Computer Security Applications Conference*, pp. 263–276, 2020.

[64] "About dependabot version updates - github docs." https://docs.github.com/en/code-security/dependabot/dependabot-version-updates/about-dependabot-version-updates.

[65] C. Zuo, Z. Lin, and Y. Zhang, "Why does your data leak? uncovering the data leakage in cloud from mobile apps," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1296–1310, 2019.

[66] A. Potdar and E. Shihab, "An exploratory study on self-admitted technical debt," in *2014 IEEE International Conference on Software Maintenance and Evolution*, pp. 91–100, 2014.