Mobile App Security Trends and Topics: An Examination of Questions From Stack Overflow

Timothy Huo
University of Hawaiʻi at Mānoa
thuo@hawaii.edu

Ana Catarina Araújo
University of Hawaiʻi at Mānoa
acoa@hawaii.edu

Jake Imanaka
University of Hawaiʻi at Mānoa
jimanaka@hawaii.edu

Anthony Peruma
University of Hawaiʻi at Mānoa
peruma@hawaii.edu

Rick Kazman
University of Hawaiʻi at Mānoa
kazman@hawaii.edu

Abstract

The widespread use of smartphones and tablets has made society heavily reliant on mobile applications (apps) for accessing various resources and services. These apps often handle sensitive personal, financial, and health data, making app security a critical concern for developers. While there is extensive research on software security topics like malware and vulnerabilities, less is known about the practical security challenges mobile app developers face and the guidance they seek. In this study, we mine Stack Overflow for questions on mobile app security, which we analyze using quantitative and qualitative techniques. The findings reveal that Stack Overflow is a major resource for developers seeking help with mobile app security, especially for Android apps, and identifies seven main categories of security questions: Secured Communications, Database, App Distribution Service, Encryption, Permissions, File-Specific, and General Security. Insights from this research can inform the development of tools, techniques, and resources by the research and vendor community to better support developers in securing their mobile apps.

Keywords: Stack Overflow, Android, iOS, Security

1. Introduction

Technology advancements, such as integrated development environments (e.g., Xcode, Android Studio, etc.), software frameworks/libraries (e.g., OkHTTP, Lottie, etc.), and platforms (e.g., Firebase, Supabase, etc.) have made it possible for almost anyone to create mobile applications (apps) without extensive programming knowledge (). This has led to a significant increase in the volume of apps available on app distribution service providers like the Google Play Store and Apple App Store, ranging from simple timezone conversion apps to more sophisticated apps for online financial transactions and health and fitness tracking (Statista, 2022). 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 (Khalid et al., 2015) and even result in end-users installing unsafe apps that expose sensitive data or act as a vector for malware (Qamar et al., 2019). While there are best practices, tools, and techniques to help developers and project teams construct high-quality, secure software systems (Pressman and Maxim, 2019), 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 energy and resource conscious (Carette et al., 2017), have access to fewer default permissions due to tighter sandboxing (Scoccia et al., 2019), are vetted through centralized app stores (Lin et al., 2021), rely more on inconsistent and untrusted network connectivity (Greenwood and Khan, 2014), and depend more heavily on external libraries for added functionality (Feal et al., 2021), among other characteristics (). Hence, it is essential to study mobile apps separately from other software systems to address their unique challenges. Therefore, in this study, we mine the questions developers ask about mobile app security on Stack Overflow– one of the largest online programming-specific question-answer sites, with over 20 million questions, answers, and users (StackExchange, 2022).

1.1. 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 and software libraries, 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 (StackOverflow, 2018a).

1.2. Goal & Research Questions

Grounded in theories of socio-technical systems and knowledge sharing in software development communities, the goal of this study is to *understand the key trends, topics, and challenges around developer discussions on securing their mobile apps*. We envision our findings helping researchers and educators better understand these trends and challenges in order to improve security training materials and tools tailored to the needs of app developers. Moreover, the results can aid developers in better anticipating and addressing potential vulnerabilities in their mobile applications by planning for security issues that commonly arise in practice. Below are the research questions (RQs) for this study along with the reasons for their inclusion:

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. 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 to present a granular view of the challenging topics in mobile app security.

2. Related Work

Previous studies have investigated mobile app security discussions on Stack Overflow. Tahaei et al. (2022) analyzed privacy and permission-related posts for Android and iOS health apps, while Beyer and Pinzger (2014) manually examined Android-related posts, finding that security questions were less common than other topics. Fischer et al. (2017) classified security-related code snippets in Android posts and analyzed their presence in Android apps. Studies have also investigated general security-related discussions on Stack Overflow. Yang et al. (2016) highlighted mobile security as a topic, while Lopez et al. (2018) performed thematic analyses on security questions and comments. Tahaei et al. (2020) conducted topic modeling on privacy-related questions, and Licorish and Nishatharan (2021) examined security faults in code snippets. Bayati and Heidary (2016) quantitatively analyzed information security questions, and Hong et al. (2021) and Chen et al. (2020) developed techniques to detect insecure code snippets in Stack Overflow. Rosen and Shihab (2016) and Linares-Vásquez et al. (2013) identified common topics, but security was not a primary focus. Similarly, Beddiar et al. (2020) analyzes and classifies Android API-related discussions on Stack Overflow, but does not address API security issues or topics. Ahmad et al. (2019) focuses on analyzing non-functional requirements (NFRs) for iOS app development based on Stack Overflow posts, but security is not mentioned as one of the NFRs analyzed in this study. A study conducted by Li et al. (2021) analyzed discussions among Android developers on Reddit. The study found that privacy concerns were not often discussed in relation to specific app design contexts. However, these concerns were actively debated when prompted by external privacy-enhancing restrictions.

The research community has found that mining Stack Overflow is valuable for studying real-world software engineering practices, including mobile app development and security. Our work extends knowledge of app development challenges by analyzing a large, recent dataset of mobile app security discussions.

3. Method

In this section, we provide details about the methodology for our study. Furthermore, the artifacts from this study are available for download [1].

3.1. Dataset

To obtain the Stack Overflow posts for this study, we used the Stack Exchange Data Explorer[2]. We ran three queries to collect mobile security posts: one for all questions and two for accepted and non-accepted answers. We extracted posts from the inception of Stack Overflow (September 2008) to December 31, 2022. Similar to prior research (Peruma et al., 2021), our search process involves querying the tag and title of questions, as developers concisely state the main objective of the question in the title (Rosen and Shihab, 2016). We refrained from searching keywords in the question body as these may not necessarily indicate relevance to mobile security.

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 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.

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 limited our query to Android and iOS, as these are the leading mobile operating systems (StatCounter, 2022). Furthermore, developers often include these terms when seeking help with a specific framework or library, which can be security-related.

[1] https://zenodo.org/doi/10.5281/zenodo.13753125
[2] https://data.stackexchange.com/

3.2. RQ Analysis

Similar to prior research (Peruma et al., 2021), 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. In Section 4, we elaborate on the specific techniques we utilize to answer each RQ.

4. Results

This section reports on the findings of our study by answering our RQs.

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

RQ 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 questions expanded over the years?*

Executing the queries described in Section 3.1 yields 5,759 questions related to mobile security, with a majority (i.e., 4,586 or 79.63%) of these questions receiving an answer. Looking at the answered questions, we observe that there are 2,387 (or 41.45%) questions with accepted answers and 3,372 (or 58.55%) questions that do not have an accepted answer. Finally, only 1,173 (or 20.37%) questions did not receive an answer.

Next, we examine the yearly breakdown of questions by examining the date when the question was posted. As shown in Table 1, 2016 witnessed the highest amount of mobile security questions (872, to be precise) posted by

Table 1: Mobile security questions asked each year.

Year	Total Questions	Questions With		
		No Answers	Non-Accepted Answers	Accepted Answers
2022	253	118	83	52
2021	272	94	101	77
2020	414	119	164	131
2019	375	106	147	122
2018	455	99	191	165
2017	647	149	267	231
2016	872	195	336	341
2015	631	108	267	256
2014	510	87	200	223
2013	473	58	184	231
2012	464	31	146	287
2011	304	8	93	203
2010	83	1	19	63
2009	6	0	1	5

Table 2: Top five occurring tags for mobile security questions.

Tag Name	Occurrence	Percentage
android	4,022	17.72%
security	3,612	15.92%
ios	1,609	7.09%
java	660	2.91%
android-security	623	2.75%
others	*12,168*	*53.62%*
Total	**22,694**	**100%**

developers, followed by 2017 and 2015. Furthermore, the median number of yearly mobile security questions is 434.50, while the median number of answers in a year is 535. Moreover, we also observe that for each year, more questions receive an answer than questions that do not receive an answer.

Moving on, 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 a rise in questions around the Google Play Store, such as help resolving `X509TrustManager`[3] warnings. This aligns with Google Play blocking apps containing unsafe implementations `X509TrustManager` in May 2016 (Axway, 2016). Additionally, iOS developers needed help with `App Transport Security`[4] issues on iOS 9 or 10, which were released during this period and could explain the rise in questions on this topic.

Finally, we examine the median time between a developer asking a question and receiving a response. We observe that it takes approximately 87.1 minutes (or 1.45 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 (Peruma et al., 2021). 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 1,793 unique tags. From this set, we derive the number of times each tag occurs in the dataset. Table 2 shows the top 5 frequently occurring tags, with 'android' claiming first place with 4,022 (or 17.72%) instances, followed by 'security' with 3,612 (or 15.92%) instances, and 'ios' with 1,609 (or 7.09%) instances.

Even though developers utilize various tags for their mobile security questions, these tags can be grouped into categories. Similar to (Peruma et al., 2021), we manually examined a statistically significant sample of frequently utilized tags and grouped related tags into categories. To this extent, three authors analyzed 413 distinct tags, which equates to a confidence level of 99% and a confidence interval of 5%. Each author independently evaluated and grouped related tags into specific categories, resolving any disagreements through discussions and agreeing on the finalized set of categories. As shown in Table 3, 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/Software Engineering Concepts* tags such as 'debugging' and '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

[3]https://developer.android.com/reference/javax/net/ssl/X509TrustManager
[4]https://developer.apple.com/documentation/bundleresources/information_property_list/nsapptransportsecurity

Table 3: 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%

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. Stack Overflow remains a well-known platform for developers seeking help with mobile security issues. Most of the questions are about the security of Android applications and they are tagged with various security-related labels. Furthermore, significant changes made by Google and Apple to the mobile ecosystem may cause a sudden surge in the number of questions.

4.2. RQ2: What security challenges do mobile developers face?

RQ 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 natural language processing techniques to better understand the security challenges faced by mobile app developers. To do this, we analyze the actual content of the questions asked. By focusing on the body of the questions, which is written in natural language, we can gain more insight into the difficulties that developers encounter when securing their apps, as compared to just looking at the tags used.

Prior to analyzing the question body, we normalize the text (i.e., we do 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. Next, we perform a topic modeling analysis on the body of the question using the Latent Dirichlet Allocation (LDA) algorithm (Blei et al., 2003). This type of analysis is frequently utilized in similar studies (e.g.,). 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 (Röder et al., 2015) and manually inspecting the output (Sievert and Shirley, 2014) 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 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 1,499 questions, a stratified statistically significant (95% confidence level and 5% confidence interval) sample of questions (in each topic). Table 4 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 and include a suitable 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. 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.

Table 4: Volume of questions for each LDA topic and reviewed by the authors, and a partial set of associated words.

Topic	Questions			Topic Representative Words
	Count	Percentage	Manually Reviewed	
General Security	3,100	53.83%	342	application, android, ios, implement, login, credentials, password, server, device, secure, api, token, client, access, user
Secured Communications	843	14.64%	265	certificate, error, ssl, http, transport, load, ats, https, network, domain, tls, cert, connect, url, webview
Databases	668	11.60%	245	database, firebase, rule, create, read, write, delete, firestore, db, query, data, add, update, id, uid, field
App Distribution Service	386	6.70%	193	google, play, version, apk, apps, vulnerability, warning, console, cordova, developer, store, error, policy, email, package, rejected
Encryption	329	5.71%	178	keys, keystore, encryption, private, string, public, encrypt, encrypted, algorithm, secret, keychain, decrypt, rsa, cipher, aes
Permissions	253	4.39%	153	permission, granted, denied, request, manifest, infoplist, securityexception, exception, intent, broadcast, sms
File Specific	180	3.13%	123	project, build, gradle, run, config, path, uri, file, folder, directory, image, videos, studio, ipa, apk
Total	*5,759*	*100.00%*	*1,499*	–

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 (StackOverflow, 2009).

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 data is essential, securing it is equally important. Unauthorized access to user data can result in serious consequences such as theft of personal information, identity theft, or other fraudulent activities.

Our analysis of questions associated with this topic shows access control as a common area of concern 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 StackOverflow, 2016a.

App Distribution Service. These are services, including 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?"

```
HttpsURLConnection.setDefaultHostnameVerifier(new
HostnameVerifier(){
        public boolean verify(String arg0, SSLSession arg1) {
                return true;}});
```

Quote 4: An example of a developer seeking help in resolving a Google Play Store security policy warning for their app (StackOverflow, 2016b).

Encryption. While a database provides a medium for an app to store its data, it is highly recommended to encrypt sensitive information to ensure it can only be accessed by authorized entities (OWASP, 2023). 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). Further, 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 (StackOverflow, 2010).

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 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 (StackOverflow, 2012a).

File-Specific. Mobile app development is a complex process that involves the use of various file types, such as resources, build, data, certificates, and downloaded files, among others. While creating an app, developers need to integrate and manage these files effectively. However, it can be challenging for developers to understand how to integrate and manage these files.

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, 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 (StackOverflow, 2018b).

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.

Further, 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 (StackOverflow, 2012b).

Summary for RQ2. When it comes to mobile security, our findings 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.

5. Threats To Validity

While there are other online question-and-answer sites and forums, we limit our analysis to Stack Overflow. While extensive, it may not fully represent the broader landscape of mobile app development and security, which could impact the generalizability of our results. Further, our data extraction query is limited to Android and iOS questions and includes the term 'security' in the title and tag. While these two mobile operating systems make up the largest market share, there can be instances of security issues specific to other mobile operating systems that we did not capture in our study. Our analysis is also 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. Finally, our study presents a snapshot of current mobile app security topics and trends, providing future studies with a platform for examining the evolution of mobile app security.

6. Conclusion & Future Work

This exploratory study confirms and extends findings from previous research, showing that Stack Overflow is a popular platform for mobile app developers to seek help with various security challenges. Our findings align with the study by Yang et al. (2016) on general software security issues, identifying similar topics such as injection vulnerabilities, SSL, certificates, databases, encryption, credentials, and access control. We also observe a sudden spike in questions related to API changes, as reported by Linares-Vásquez et al. (2014). Our results confirm that permissions are a common issue for app developers, as shown by Scoccia et al. (2019). Furthermore, our study aligns with Rahkema and Pfahl (2022) findings on iOS app developers using vulnerable third-party libraries. This study contributes to the theory of knowledge sharing in software development communities by revealing how mobile app security knowledge is collaboratively constructed and disseminated on Stack Overflow.

The findings of this study highlight practical challenges developers face in securing their apps, providing educators, researchers, and project teams with insights into specific mobile security focus areas. These insights can be used to improve training materials, courses, and code quality tools, as well as to help developers and project teams plan for problematic areas in securing their apps. Given the increasing use of Generative AI technologies, such as Large Language

Models (LLMs), our future work will explore the extent to which developers rely on LLMs instead of traditional resources like Stack Overflow for help with app security.

References

Ahmad, A., Feng, C., Li, K., Asim, S. M., & Sun, T. (2019). Toward empirically investigating non-functional requirements of ios developers on stack overflow. *IEEE Access*, *7*, 61145–61169. https://doi.org/10.1109/ACCESS.2019.2914429

Aldayel, A., & Alnafjan, K. (2017). Challenges and best practices for mobile application development: Review paper. *Proceedings of the International Conference on Compute and Data Analysis*. https://doi.org/10.1145/3093241.3093245

Allen, G. (2021). *Android for absolute beginners: Getting started with mobile apps development using the android java sdk*. Apress.

Axway. (2016). https://blog.axway.com/learning-center/software-development/api-development/google-security-alert-unsafe-implemen

Bayati, S., & Heidary, M. (2016). Information security in software engineering, analysis of developers communications about security in social q&a website. *Intelligence and Security Informatics: 11th Pacific Asia Workshop. PAISI 2016, Auckland, New Zealand, April 19, 2016, Proceedings 11*.

Beddiar, C., Khelili, I. E., Bounour, N., & Seriai, A.-D. (2020). Classification of android apis posts: An analysis of developer's discussions on stack overflow. *2020 International Conference on Advanced Aspects of Software Engineering (ICAASE)*.

Beyer, S., & Pinzger, M. (2014). A manual categorization of android app development issues on stack overflow. *2014 IEEE International Conference on Software Maintenance and Evolution*, 531–535.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*(null), 993–1022.

Carette, A., Younes, M. A. A., Hecht, G., Moha, N., & Rouvoy, R. (2017). Investigating the energy impact of android smells. *IEEE 24th International Conference on Software Analysis, Evolution and Reengineering*.

Chen, L., Hou, S., Ye, Y., Bourlai, T., Xu, S., & Zhao, L. (2020). Itrustso: An intelligent system for automatic detection of insecure code snippets in stack overflow. *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*.

Feal, Á., Gamba, J., Tapiador, J., Wijesekera, P., Reardon, J., Egelman, S., & Vallina-Rodriguez, N. (2021). Don't accept candy from strangers: An analysis of third-party mobile sdks. *Data Protection and Privacy, Volume 13: Data Protection and Artificial Intelligence*, *13*, 1.

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

Flora, H. K., Wang, X., & Chande, S. V. (2014). An investigation on the characteristics of mobile applications: A survey study. *International Journal of Modern Education and Computer Science*, *6*(11), 21–27.

Greenwood, D. S. J. S. G., & Khan, Z. L. L. (2014). Smv-hunter: Largescale, automated detection of ssl/tls man-in-the-middle vulnerabilities in android apps. *Network and Distributed System Security Symposium (NDSS). Internet Society, San Diego, CA*, 1–14.

Hong, H., Woo, S., & Lee, H. (2021). Dicos: Discovering insecure code snippets from stack overflow posts by leveraging user discussions. *Annual Computer Security Applications Conference*. https://doi.org/10.1145/3485832.3488026

Khalid, H., Shihab, E., Nagappan, M., & Hassan, A. E. (2015). What do mobile app users complain about? *IEEE Software*. https://doi.org/10.1109/MS.2014.50

Li, T., Louie, E., Dabbish, L., & Hong, J. I. (2021). How developers talk about personal data and what it means for user privacy: A case study of a developer forum on reddit. *Proc. ACM Hum.-Comput. Interact.*, *4*(CSCW3). https://doi.org/10.1145/3432919

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

Lin, F., Wang, H., Wang, L., & Liu, X. (2021). A longitudinal study of removed apps in ios app store. *Proceedings of the Web Conference 2021*.

Linares-Vásquez, M., Bavota, G., Di Penta, M., Oliveto, R., & Poshyvanyk, D. (2014). How do api changes trigger stack overflow discussions? a study on the android sdk. *Proceedings of the 22nd International Conference on Program Comprehension*.

Linares-Vásquez, M., Dit, B., & Poshyvanyk, D. (2013). An exploratory analysis of mobile development issues using stack overflow. *2013 10th Working Conference on Mining Software Repositories (MSR)*, 93–96. https://doi.org/10.1109/MSR.2013.6624014

Lopez, T., Tun, T. T., Bandara, A., Levine, M., Nuseibeh, B., & Sharp, H. (2018). An investigation of security conversations in stack overflow: Perceptions of security and community involvement. *Proceedings of the 1st International Workshop on Security Awareness from Design to Deployment*.

OWASP. (2023). https://owasp.org/www-project-mobile-top-10/2023-risks/m9-insecure-data-storage

Peruma, A., Simmons, S., AlOmar, E. A., Newman, C. D., Mkaouer, M. W., & Ouni, A. (2021). How do i refactor this? an empirical study on refactoring trends and topics in stack overflow. *Empirical Software Engineering*, *27*(1), 11. https://doi.org/10.1007/s10664-021-10045-x

Pressman, R., & Maxim, B. (2019). *Software engineering: A practitioner's approach*. McGraw-Hill Education.

Qamar, A., Karim, A., & Chang, V. (2019). Mobile malware attacks: Review, taxonomy & future directions. *Future Generation Computer Systems*, *97*, 887–909.

Rahkema, K., & Pfahl, D. (2022). Quality analysis of ios applications with focus on maintainability and security. *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*.

Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*.

Rosen, C., & Shihab, E. (2016). What are mobile developers asking about? a large scale study using stack overflow. *Empirical Software Engineering*, *21*(3), 1192–1223. https://doi.org/10.1007/s10664-015-9379-3

Sahar, A., & Clayton, C. (2021). *Ios 15 programming for beginners: Kickstart your mobile app development journey by building ios apps with swift 5.5 and xcode 13*. Packt Publishing.

Scoccia, G. L., Peruma, A., Pujols, V., Malavolta, I., & Krutz, D. E. (2019). Permission issues in open-source android apps: An exploratory study. *2019 19th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, 238–249. https://doi.org/10.1109/SCAM.2019.00034

Sievert, C., & Shirley, K. (2014). Ldavis: A method for visualizing and interpreting topics. *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, 63–70.

StackExchange. (2022). https://stackexchange.com/sites?view=list%5C#users

StackOverflow. (2009). https://stackoverflow.com/q/933331

StackOverflow. (2010). https://stackoverflow.com/q/4275311

StackOverflow. (2012a). https://stackoverflow.com/q/10443615

StackOverflow. (2012b). https://stackoverflow.com/q/13854425

StackOverflow. (2016a). https://stackoverflow.com/q/39584090

StackOverflow. (2016b). https://stackoverflow.com/q/40928435

StackOverflow. (2018a). https://stackoverflow.com/q/48542621

StackOverflow. (2018b). https://stackoverflow.com/q/48770008

StatCounter. (2022). https://gs.statcounter.com/os-market-share/mobile-ta

Statista. (2022). https://www.statista.com/statistics/276623/number-of-app

Tahaei, M., Bernd, J., & Rashid, A. (2022). Privacy, permissions, and the health app ecosystem: A stack overflow exploration. *Proceedings of the 2022 European Symposium on Usable Security*.

Tahaei, M., Vaniea, K., & Saphra, N. (2020). Understanding privacy-related questions on stack overflow. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*.

Yang, X.-L., Lo, D., Xia, X., Wan, Z.-Y., & Sun, J.-L. (2016). What security questions do developers ask? a large-scale study of stack overflow posts. *Journal of Computer Science and Technology*, *31*(5), 910–924.