Stuck in the Permissions With You: Developer & End-User Perspectives on App Permissions & Their Privacy Ramifications

Mohammad Tahaei
mohammad.tahaei@nokia-bell-labs.com
Nokia Bell Labs
United Kingdom

Ruba Abu-Salma
ruba.abu-salma@kcl.ac.uk
King's College London
United Kingdom

Awais Rashid
awais.rashid@bristol.ac.uk
University of Bristol
United Kingdom

ABSTRACT

While the literature on permissions from the end-user perspective is rich, there is a lack of empirical research on why developers request permissions, their conceptualization of permissions, and how their perspectives compare with end-users' perspectives. Our study aims to address these gaps using a mixed-methods approach.

Through interviews with 19 app developers and a survey of 309 Android and iOS end-users, we found that both groups shared similar concerns about unnecessary permissions breaking trust, damaging the app's reputation, and potentially allowing access to sensitive data. We also found that developer participants sometimes requested multiple permissions due to confusion about the scope of certain permissions or third-party library requirements. Additionally, most end-user participants believed they were responsible for granting a permission request, and it was their choice to do so, a belief shared by many developer participants. Our findings have implications for improving the permission ecosystem for both developers and end-users.

CCS CONCEPTS

• **Security and privacy → Software and application security**; **Human and societal aspects of security and privacy**; **Usability in security and privacy**; • **Human-centered computing → Human computer interaction (HCI)**; **HCI design and evaluation methods**; • **Software and its engineering**;

KEYWORDS

smartphone permissions, privacy, developers, usable privacy, usable security, programming, empirical software engineering

ACM Reference Format:

Mohammad Tahaei, Ruba Abu-Salma, and Awais Rashid. 2023. Stuck in the Permissions With You: Developer & End-User Perspectives on App Permissions & Their Privacy Ramifications. In *CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 24 pages.

CHI '23, April 23–28, 2023, Hamburg, Germany

1 INTRODUCTION

Permissions are the primary mechanism for protecting data and resources on smartphones, including end-users' location, contacts, and photos. Developers use permissions to ask for data and resources needed for particular app features to function, and end-users, on the other hand, decide whether they want to grant or deny such requests.[1] Developers play a critical role in choosing which app permissions to include or exclude and, consequently, in the app ecosystem. However, despite the rich literature on permissions, there have been no empirical studies *with* developers about their views, understanding, and practices with regard to permissions and how these compare to end-users' attitudes toward permissions.

End-users primarily grant or deny permission requests based on how permissions are related to the app's functionality and end-users' expectations [21, 73, 97]. Some end-users also consider the privacy ramifications of permissions and make contextualized decisions [97]. However, developers' decision-making processes with regard to permissions and how developers' perspectives on permissions compare with end-users' perspectives have not been studied. Besides the lack of research on developers' understanding of permissions, the recurring pattern of unused and excessive permissions requested by apps over the past ten years [23, 51, 95] as well as end-users' privacy concerns about permissions (e.g., data leaks to third parties [22, 66, 95]) emphasize the need for understanding how developers decide on which permissions to include or exclude in their apps and what challenges they face when integrating permissions.

In this study, we shed light on the decision-making processes of (and the challenges faced by) developers with regard to permissions. We also augment our findings with a follow-up study with end-users to give in-depth insights into the permission ecosystem from the perspectives of two primary stakeholders of the app ecosystem, developers and end-users. Our research questions (RQs) are:

RQ1: How do developers decide on what permissions to request?
RQ2: What is developers' understanding of permissions and their privacy ramifications?
RQ3: What are end-users' attitudes toward permissions, & how do these compare with developers' perspectives of permissions?

We conducted a mixed-method study with developers and end-users to answer our RQs. We interviewed 19 developers about their practices with regard to permissions, decision-making processes, and challenges when integrating permissions (RQ1 & RQ2). Based on these findings, we then designed and conducted a survey with 309 Android and iOS end-users to explore their views on permissions and compare them with developer participant views (RQ3).

[1] Throughout the paper, we refer to *developers* as those who develop smartphone apps and *end-users* as those who use smartphone apps.

We found that developer participants often viewed permissions as an access control mechanism, a perspective shared by some of our end-user participants. Some end-user participants viewed permissions as a legal guard against the misuse of their data, a view not shared by developer participants who rarely brought up any legal ramifications of permissions. We also found that app features and functionality were the primary reasons for developer participants to include permissions. However, they rarely removed permissions. Further, they were aware of the poor end-user experience of requesting too many permissions, and they highly regarded the trust and reputation that came with respecting end-users. Our results also show that some developer participants needed clarification on the scope of permissions (mainly when a resource could be accessed with different permissions, e.g., fine location vs. coarse location). Due to this confusion, they sometimes added multiple permissions to avoid crashes or unexpected consequences. Additionally, our developer participants included third-party libraries not only for providing functionality but also for helping them manage permissions, which sometimes resulted in apps requesting permissions that developer participants were not expecting.

Like developer participants, our end-user participants mostly granted permissions because of features, and some granted permissions because they trusted the app. 31.4% of our end-user participants did not see any harm in permissions, and those who saw the harm in granting permissions were worried about unintended use of their data, such as selling their data to third parties. 57.9% of our end-user participants had never removed already granted permissions because they believed they had no reason to do it, did not know how to do it, or had not thought about doing so previously.

2 RELATED WORK

2.1 App Permissions

Based on analyses of Android app reviews [39, 57, 58], one of the main privacy concerns of end-users relates to app permissions and what access apps have to data and resources on their smartphones. However, when a permission request is contextualized, matches end-users' expectations, and is perceived by end-users as necessary for an app to function, end-users are likely to grant the permission request [21, 50, 54, 73, 97], which is rooted in the theory of Contextual Integrity [97]. For example, a end-user may understand why service providers need to gather certain data about end-users as long as providers do not misuse and protect end-user data [50]. Therefore, knowing why a developer includes or excludes a permission request can shed light on a primary privacy concern of end-users, permission requests.

Unused and excessive permissions have been a recurring pattern in apps over the past ten years [23, 24, 51]. Studying developer artifacts suggests that developers may leave unused or unnecessary permissions in their code due to updating the list of permissions requested in different app development stages (e.g., a developer may include a permission request during the testing stage and forget to remove it later during the production stage) [23, 71], copy-pasting code from the Internet, lack of access to comprehensive documentation, confusion about permissions' naming, and not fully understanding the scope and use cases of permissions [23]. On the positive side, developers can be nudged toward reducing the number of permissions they include by knowing that apps with similar functionalities ask for fewer permissions. Such a nudge in the form of a warning message that pops up when developers submit their apps to the Google Play Store has improved developers' decisions and reduced permissions [60, 61]. However, using unnecessary permissions or forgetting to remove unused permissions remains a problem [15, 51, 71]. These permissions can stay in an app, shifting the responsibility to end-users to review them at some point later in time (if granted), which is not a typical behavior among end-users [73]. Apps with unnecessary permissions can cause end-user frustration and have privacy ramifications for them [23, 27].

Despite the importance of permission systems in the app privacy ecosystem and the plethora of studies with end-users, the *only* study exploring issues faced by developers when working with permissions had explicitly focused on iOS permission descriptions in 2014, when these descriptions were in their early stages, using a survey [88]. Developers had a mixed understanding of why descriptions were needed and were unsure how to write them, and some were also unaware of such descriptions. The permission systems have changed significantly since the study was conducted; for example, Android pushed for run-time permissions instead of install-time permissions (ask at install time vs. ask when needed) [33].

While there is a rich literature on permissions, there have been no empirical studies *with* developers about permissions. Our study bridges the gap between how developers, on the one hand, and how end-users, on the other hand, understand permissions. We provide empirical insights into developers' understanding and decision-making processes with regard to app permissions using interviews. Drawing on the interview findings, we design and conduct a survey of end-users to capture their perspectives on permissions and compare them with developer perspectives.

2.2 Empirical Privacy Studies With Developers

A strand of empirical software engineering research has studied the support developers need to build privacy-friendly apps and perform privacy-related tasks, such as finding privacy issues in code [46], deciding on personalizing ads [79], and building apps that are compliant with privacy laws [3, 81]. In the case of making child-directed apps compliant with privacy laws like the California Consumer Privacy Act (CCPA) [29], developers often try to satisfy app store requirements instead of laws, and they often rely on app stores and operating systems to detect privacy-related issues [3, 86].

Third-party libraries are often confusing for developers because of the libraries' unclear data collection practices and complicated configurations [3]. These libraries could be unclear about their purposes and permission requests, causing developers to send unnecessary sensitive data about end-users to third parties without developers' knowledge [65, 66]. Studies of privacy posts on developer forums such as Stack Overflow, Reddit, and iPhoneDevSDK have shown that managing permissions is a complex task for developers [47, 74, 80, 87], due in part to developers not understanding the scope of specific permissions or, more broadly, the app ecosystem [47]. On the other hand, experienced developers recognize the direct relationship between asking for permissions and end-users trusting an app, recommending fellow developers to ask for permissions only when necessary, with clear permission descriptions [74].

In this work, we contribute to this body of research by providing first-hand qualitative insights into developers' understanding of and decision-making processes with regard to permissions using in-depth interviews.

3 METHOD

To answer our RQs (§1), we first conducted 19 interviews with developers to understand their thought processes with regard to permissions (RQ1 & RQ2). Our interview findings informed the design of a survey study we conducted with 309 smartphone end-users. The survey aimed to explore end-users' perspectives on permissions and compare them with those of developers (RQ3). The research ethics committee at the University of Bristol (faculty of engineering) reviewed and approved our study.

3.1 Interview Study With Developers

3.1.1 Developer Participant Recruitment. We advertised our interview study on LinkedIn [48], Twitter [91], and two freelancing websites, namely Freelancer [28] and Upwork [92]. We also used Prolific [63], a crowdsourcing platform, to find additional developer participants. On Prolific, we invited interested participants who stated they had programming skills and worked in the computer and information technology industry. All these approaches and platforms were used in the literature to recruit developers [37, 81, 85, 94]. We recruited developer participants between April 2022 and May 2022.

Screening Survey. To ensure that all developer participants had a background in programming, app development, and permissions, we asked all interested candidates to fill out a short survey describing their role in their last software development job, years of experience in software development, and app development. We also included six programming questions, suggested by Danilova et al. [18], to screen out participants who did not have basic programming knowledge. The first four programming questions were about the definition of a compiler function, the possible value of a Boolean variable, the website that programmers frequently visited, and the definition of a recursive function. Two additional questions assessed participants' understanding of a short pseudocode snippet. The screening survey can be found in Appendix A.

It took interested candidates five minutes on average (SD=2) to complete the screening survey. After screening out those who did not pass the programming questions, we ensured that those who passed had at least two years of software development experience to set a baseline for participants' experience with permissions. In total, 67 developer participants met our criteria to be invited to our interview study; 19 participated. The interview transcripts are unavailable online due to privacy and ethical considerations; however, they can be accessed upon request for research purposes.

3.1.2 Developer Participant Demographics. We interviewed 19 developer participants via audio calls; eight were from Upwork, six from Freelancer, three from LinkedIn, and two from Prolific. On average, they were thirty years old (SD=6), and they had eight years of experience in software development (SD=5), four years of experience in Android development (SD=3), and three years of experience in iOS development (SD=3). Fourteen self-identified as male, three as female, and two preferred not to describe their gender. Ten were located in Asia, three in Europe, two in Africa, two in North America, and one in South America. They all had experience working with permissions. Table 3 in Appendix B summarizes developer participants' demographics.

App Development Experience. Participants developed different apps, including healthcare, utility, e-commerce, educational, social media, gaming, smart homes, and finance. Four participants developed apps specifically targeted at adults and four at kids; the rest did not have age limits for their apps.

When developing apps, participants reported that they collected different types of data mainly depending on what their apps required to function correctly, including end-users' phone numbers, email addresses, countries of residence, locations at specific times, photos, videos, contacts, messages, device IP addresses and tokens (for fingerprinting), purchase history, shopping preferences, and data for debugging, analytics, and measuring performance. Developer participants also used different libraries when developing apps. The most common ones were Firebase, Google Analytics, Flurry, Retrofit, and other libraries for image processing, QR code scanning, data visualization, and crash report generation.

3.1.3 Study Procedure. After obtaining our developer participants' consent and briefly explaining the study (we sent participants the consent form and additional information about the study ahead of the interviews), we asked participants about their job background and experience in app development. We then explored their understanding of permissions, decision-making processes, whether they updated (by adding or removing) permissions post-development, and their confusing and challenging experiences with permissions.

Based on their platform expertise, we showed participants the 15 most commonly used permissions in Android, iOS, or both, taken from Kollnig et al. [43] and asked them to describe what they thought those permissions did. We also discussed what data they considered sensitive, what privacy within the context of app permissions meant to them, whether they were aware of any privacy laws, and, if so, how they complied with them when choosing permissions. Through these discussions, we explored participants' understanding of permissions' harms and privacy ramifications. Interviews ended with questions about participants' ideas or recommendations for improving permissions, as well as any other missing thoughts participants might have.

The interviews took, on average, 44 minutes (SD=10). We reached data saturation; the interviewer frequently compared notes on the topics arising and discussed with another author whether or not to continue interviews after every few. Each developer participant received £30 for their time.[2] All interviews were conducted in English. The interview guide can be found in Appendix C.

3.1.4 Data Analysis. All interviews were audio recorded and transcribed using professional privacy-compliant transcription services. We then imported the interview transcripts to NVivo for analysis [40]. Two authors inductively coded all 19 transcripts using thematic analysis [55, 68]. To develop an initial codebook, they first

[2]Although our developer participants did not reside in the U.K. (Table 3), as per the guidelines and recommendations provided by our institution's Research Ethics Committee, we needed to comply with domestic labor laws in the U.K. Hence, we paid our developer participants about three times the U.K. national minimum wage (£9.5 was the minimum at the time of recruitment; developer participants received £30).

independently open-coded two transcripts (the same ones) and developed their own codebooks. In multiple sessions, they discussed their codebooks, merged them into one, and resolved disagreements. They also sought additional feedback and input from a third author on their codebook, notes, and preliminary findings. The codebook-building procedure was iteratively followed until the codebook was stable, signaling that code saturation was achieved; new codes stopped emerging. Code saturation occurred after 11 iterations—each iteration involved both authors independently coding one transcript (the same transcript but different from transcripts used in other iterations), meeting to resolve disagreements, and refining the codebook. Both authors coded two additional transcripts after the 11th iteration using the final codebook to ensure the codebook was stable and no changes were made. Codes were not mutually exclusive, and a quote might appear in multiple codes.

The two authors then independently coded all 19 transcripts using the final stable codebook and measured inter-rater reliability using NVivo. The average Cohen's kappa coefficient across all codes was .71, which is considered a substantial agreement [44]. Both authors resolved the remaining disagreements through discussions. The qualitative findings in §4, labeled with ***Developer Participant Perspectives***, are based on the final codebook and resolved disagreements. Due to the qualitative nature of our interview study, we do not report frequencies of occurrences in §4. We instead use qualifiers (e.g., few, some, several, many, all) to avoid overgeneralization [52]. However, Table 4 in Appendix D includes the codebook with frequencies for interested readers.

3.2 Survey Study With End-Users

3.2.1 End-User Participant Recruitment. We recruited 309[3] end-user participants using Prolific [63], a typical crowdsourcing platform for recruiting participants for empirical privacy studies [20]. We decided to recruit participants residing in the U.K. because (1) privacy is a cultural and contextual topic, and people's interpretation of what privacy means can vary based on where they are located [14, 69, 70, 96];[4] (2) Prolific offers representative samples only for two countries, the U.S. and the U.K. [64]; (3) we had limited resources and budget; and (4) a simple translation from English into other languages could have caused different interpretations of our questions. To avoid this, we would have needed to perform a validated translation, instead of a simple translation, for consistency (as stated in [70]), requiring additional resources. Thus, we ran our survey in English.

We used Prolific's prescreening feature to target participants residing in the U.K. (for the reasons stated above) and using Android or iOS because these two had over 99% of the market share of all smartphone operating systems [76]. We also used a gender-balanced sample provided by Prolific because it could generate an almost similar sample to a representative sample with a lower cost [89].

We ran the survey in August 2022 and paid participants £1.84, slightly above the minimum hourly wage in the U.K. (£9.50, all survey participants were located in the U.K.). It took participants, on average, 11 minutes (SD=5) to complete the survey. Our anonymized survey dataset can be found at [PUBLIC LINK TO BE ADDED].

3.2.2 End-User Participant Demographics. Our end-user participants were all located in the U.K. On average, they were 37 years old (SD=13). 50.2% self-identified as male and 49.8% as female. 38.3% were employed full-time, 14.2% were employed part-time, 9.5% were not in paid work, 7.4% were unemployed (and job seeking), 1.2% were due to start a job within the next month, and 29.4% did not have an updated employment status on Prolific. On average, our end-user participants spent 5 hours (SD=3) on their smartphones daily and had a smartphone for 11 years (SD=4). 51.1% were Android end-users, and 48.9% were iOS end-users.

3.2.3 Study Procedure. The survey consisted of open-ended and closed-ended questions inspired by our interview study. The open-ended questions also explored end-user participants' understanding of permissions and their perceptions of the harms associated with permissions, as discussed with developer participants. The close-ended questions assessed whether the assumptions our developer participants made about end-users were accurate (e.g., end-users did not care about or were not aware of permissions, see §4.5), end-users' opinions of the privacy ramifications of commonly used permissions in Android and iOS (e.g., photos, locations, and camera) [43], how end-users interacted with permissions, and how they felt about permissions. Where possible and relevant, we also adapted questions from prior surveys [1, 11, 19, 38, 41, 42, 89]. Our survey instrument can be found in Appendix E.

3.2.4 Data Analysis. We did a descriptive analysis of responses to all closed-ended (multiple-choice & Likert) questions. For the open-ended questions, two authors collaboratively analyzed the qualitative responses using Miro [56] boards. We copied all responses into Miro using sticky notes and constructed groups based on the similarities across responses (i.e., building affinity diagrams [13, 45]). Some notes ended up in multiple groups, and groups were not mutually exclusive. The subsections labeled with ***End-User Participant Perspectives*** in §4 are based on these analyses.

3.3 Limitations

Our samples do not necessarily represent all developers and end-users. However, we recruited our developer participants using different channels and platforms to reduce sampling bias. Nonetheless, most participants in our sample self-identified as male (consistent with the gender-biased software development profession; over 90% of software developers are male [75]) and were from Asia. Hence, our findings are not generalizable to all developers. However, they provide novel insights into developers' understanding and practices with regard to permissions and their privacy ramifications, which is the main objective of our qualitative research. Similarly, our survey targeted a Western country: the U.K. Future research may use our public data to compare the results obtained from end-user participants residing in other countries with our findings. Furthermore, our study used a mixed-methods design: an interview-based study and a survey; each was appropriate for the targeted participant population (developers and end-users, respectively). Therefore, our results should be interpreted with these limitations in mind.

[3] Prolific's minimum size of a representative sample of the U.K. is 300 [64].

[4] We did not consider this for our interview study with developers because our sample was small, and our developer participants developed apps for end-users across the globe and not necessarily for end-users in the developers' home countries.

We did not run a survey with developers. First, unlike an interview-based study, a survey would have prevented us from generating in-depth insights into developers' understanding and practices. Second, due to privacy and ethical considerations, we could not recruit participants by harvesting their email addresses from software development platforms and forums (e.g., GitHub and Google Play) or sending unsolicited emails to developers to take part in our study (for sampling issues, see [83, 84, 93]). For future work, researchers with access to a large pool of developers could run a survey with developers to collect more data and compare their data with our publicly available survey data. Besides, we did not collect data about our developer participants' apps (e.g., magnitude of the app's end-user base and popularity, as well as the category of the app) or development mode (e.g., employed by an agency to develop apps customized for clients, working in a company on a single app, or working as a freelancer), which could have helped contextualize our findings. We suggest that future researchers gather this data as part of their screening or demographics survey.

Although we did not aim in this study to explore developers' privacy compliance processes, we asked our developer participants whether they had made changes to their permission requests in order to make their apps compliant with privacy laws. They all noted that they did not make any changes (§4.4). A recent study has found that child-directed app developers either assume that their apps are compliant as long as they have not been rejected by app stores or outsource most compliance decisions to auditing services [3]. Future research may want to investigate to what extent end-users expect developers to be aware of privacy laws and cater to legal necessities, and how different privacy laws may impact the permission requests, especially since some of our end-user participants (19%) perceived requesting permissions as a privacy compliance mechanism (see §4.2 for details).

4 FINDINGS

Figure 1 shows a summary of our findings. In each subsection, we first outline our findings from the interview study with developer participants (RQ1 & RQ2, ***Developer Participant Perspectives***). We then integrate the findings from the survey with end-user participants in these subsections (RQ3, ***End-User Participant Perspectives***), where appropriate. §4.6 & §4.7 cover topics specific to developer participants.

4.1 Commonly-Used Permissions

Developer Participant Perspectives. Developer participants mostly requested access to end-users' location, camera, photo gallery, Internet, Bluetooth, data storage, contacts, messages, and microphone. They understood what most Android and iOS permissions did (e.g., Internet or access to storage permission). However, some **could not differentiate** between permissions with similar scopes. Some Android developers were **confused** about the network and Wi-Fi permissions; they equated both with learning about the current connection. Similarly, many iOS developers were unsure whether they could access all or specific photos using PhotoLibrary and PhotoLibraryAdd (see §4.6 for the privacy ramifications of this confusion). Similar confusions occurred when permissions provided access to foreground or background services (e.g., location).

End-User Participant Perspectives. The location permission request, with 98.1% of end-user participants seeing it in the apps they had frequently used in the past year, was the most requested permission from our end-users' perspectives (Appendix F, Figure 7). While the location permission is not the most commonly used in Android and iOS (with evidence from an analysis of apps [43]), end-user participants might remember the location permission request the most because they were sensitive about it or because their commonly used apps often required this permission.

The camera (82.5%), photos (81.6%), microphone (64.1%), and contacts (54.7%) permissions were next in line for commonly seen permission requests by our end-user participants. Other permissions, such as the Internet and network state that did not require a run-time permission request from the end-user, were mentioned less often (19.7% and 19.1%, respectively).

4.2 Why Do Permissions Exist? What Do They Do?

Developer Participant Perspectives. Most developer participants used technical terms revolving around the concept of **access control**, which is not far from why apps request permissions (e.g., iOS describes permissions as "control[ing] access to information in apps on iPhone." [6]). Our developer participants used the following concepts to explain permissions: a **switch** used to enable access to specific restricted resources; a **bridge** between developers and end-users; a **medium** to protect end-users and their devices from malicious activities; and an end-user giving **consent** to accessing a resource they owned.

End-User Participant Perspectives. Most end-user participants correctly identified what permissions did: granting access to specific resources or data on their smartphone (87.7%)—similar to the access control concept from the developer study. Regarding apps that end-user participants had used in the past year, at least 70% of end-user participants agreed that they were **aware** of permissions used by these apps, the number of permission requests was **in line with their expectations**, and they **understood** why certain permissions were requested (Figure 2). Unlike our observation, the findings of a recent comprehension survey study with end-users have shown that most end-user participants could not understand permission requests' scope [73]. We speculate that this discrepancy may be attributed to the studies' methods: our data was self-reported, whereas Shen et al.'s data was based on a knowledge survey. Therefore, although most of our end-user participants reported that they were aware of permissions, this does not mean that they fully understood the permissions' scopes. Further, end-user participants used lay descriptions, compared to developer participants, to explain why apps requested permissions, as detailed below:

- To help with app's **functionality** (35%): in line with our developer participants' reasons for granting permissions and prior research with end-users [97], our end-user participants' predominant understanding of permissions was to provide some functionality, such as storing files, taking photos, and finding a location;

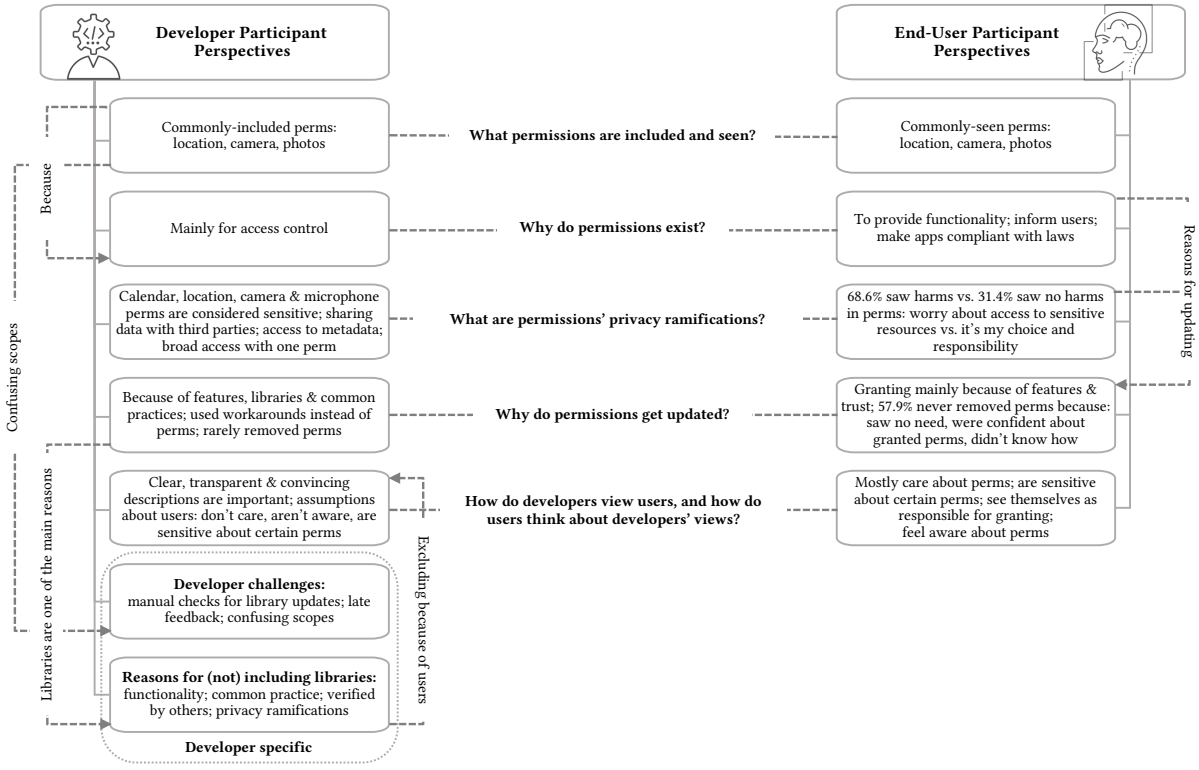

Figure 1: An overview of our findings from the interview with developer participants and the survey with end-user participants.

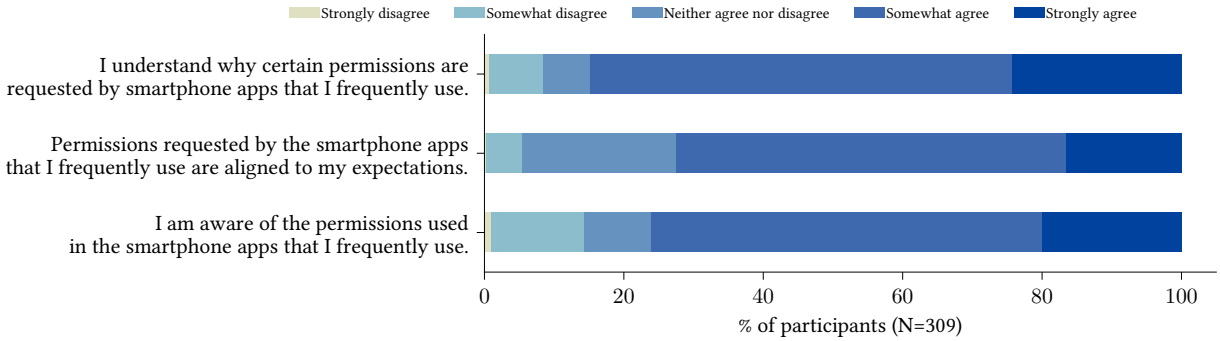

Figure 2: End-user participants' answers to "Thinking about the five smartphone apps that you have frequently used in the past year, to what extent do you agree or disagree with the following statements?"

- To **inform** end-users (21%): this theme covered topics around informing and notifying end-user participants of the reasoning behind the permission request and giving them the option to accept or deny the request. Participants believed that the control they received from the permission request could limit access to their data or stop random apps from being installed on their smartphones;
- To make apps **compliant** with laws (19%): while a few developer participants discussed compliance with privacy and data protection laws, some end-user participants viewed permissions as a way for apps to comply with laws. They thought permissions helped developers feel protected against laws when requesting end-user data. They viewed permission requests as part of legal practices that developers must follow—using a smartphone's resources without asking for permission could result in legal consequences for developers;
- To **protect** end-users (18%): similar to our developer participants, some end-user participants also saw the protection of resources and their data as one of the purposes of permissions. They believed permissions could protect their personal data from unwanted access and protect their privacy; and

- To **collect** end-users' data (13%): some end-user participants associated permissions with privacy-unfriendly practices. They thought apps requested permissions to collect data, track end-users, or target end-users with ads.

4.3 Privacy Ramifications of Permissions

Developer Participant Perspectives. Most developer participants agreed that the **calendar** and all **location-related** permissions, especially when apps were **running in the background**, could invade end-user privacy because they enabled tracking and learning end-users' location at a specific time. While our end-user participants did not explicitly mention background and foreground services as a concern, prior literature suggests that background activity can influence end-users' decision-making processes with regard to granting or denying certain permission requests [73].

Many developer participants expressed privacy concerns about the **location** permission because end-users could be tracked. It could also enable apps to **access metadata** and learn end-users' behavior by knowing, for example, whether the end-user was online or offline, as well as deduce the end-user location based on their network or Wi-Fi connection.

Several developer participants were also concerned about the **camera** and **microphone** permissions because they believed they allowed developers to capture end-users' surroundings, listen to all end-users' conversations, and record end-users without notification or feedback. Some were also concerned about the **contacts** permission because it could enable developers to sell end-users' contacts to third parties and generate profit. Although some developer participants recognized the benefits of accessing end-users' biometrics for facilitating authentication, they expressed concerns that the phone camera could capture unneeded data about end-users and their surroundings. Most developer participants agreed that the **Internet** permission was necessary for almost all apps to function correctly. However, it could invade end-user privacy if data was sent to **third parties** without end-users' knowledge or consent. It could also allow malicious actors on the Internet to track end-users or access their smartphones by exploiting unknown vulnerabilities (e.g., by using picture metadata or MAC address to locate end-users [66]).

Some developer participants also mentioned that Android had deprecated the **storage read** and **storage write** permissions due to security concerns (e.g., apps could access different account passwords stored in the same shared storage) and privacy concerns (e.g., apps should not share storage space; they should have their own private one). Some developer participants were surprised to see that granting permissions could give them **access to sensitive end-user data** and, hence, wanted to see **nuanced permissions** for additional end-user privacy:

> Some permissions are extremely open. Like, suddenly gives you a lot of access, which through the years, Android has fixed them actually. Like, when you wanted to read an SMS, you had to get SMS permission which suddenly allowed you to read all the SMS data on the phone, which they no longer allowed you to include that permission. Or, we have the get accounts permission, which again, if you want to introduce—add an account for your application to the device accounts—you have to add that permission. But that permission, at the same time, allows you to read all the other accounts that are defined on the device. (P5)

Many developer participants thought that **security** testing, **security** measures, and data **security could provide privacy**: "Nowadays, all the apps use stored data they have on a cloud system or something, so if I, as a developer, don't do my job right handling the security or permissions to access the accounts may leak the end-users later." (P8) Some developer participants explained methods to **protect end-user privacy**: testing for security vulnerabilities, using multi-factor authentication, minimizing data collection, and giving end-users the option to limit data collection. For data protection and privacy reasons, few developer participants processed (or wished they could process) their data locally by **avoiding sending data to servers**.

End-User Participant Perspectives. Like developer participants, end-user participants viewed the location, photos, and storage permissions as sensitive with privacy ramifications (a lot or above: 68%, 62.8%, and 61.8%, respectively). While the calendar permission came up as a privacy-sensitive permission request in the developer interview study, our end-user participants were not very concerned about it (a little or less: 48.2%, see Appendix F, Figure 4 for details).

In a series of Likert questions (Appendix F, Figure 5), we found that end-user participants were overall **worried** about permission requests. Many associated permission requests with tracking and monitoring (somewhat agree or above: 80.6%), would think twice if the app requested too many permissions (somewhat agree or above: 66%), and thought that apps with permissions used their data for **unintended purposes** (somewhat agree or above: 77%).

Possible Harms With Permissions. 68.6% of end-user participants said they could see possible harm in granting permission requests (Table 1):

- Access to **protected** resources (62%): some end-user participants (18%) were worried that apps could access their sensitive data (e.g., photos, location, and contacts). Some were also apprehensive about the unintended use cases of the permissions that they granted, such as security issues (18%), selling their data to third parties (12%), and being unsure about the data practices of developers (3%); and
- **Depending** on the app (6%): similar to the trust element discussed in both our developer and end-user studies, some end-user participants believed that the associated harms with permissions correlated with the developer's and app's reputation, echoing findings from prior work that brand reputation impacts end-users' decisions with regard to permissions [73].

Permissions Are Harmless. Below, we lay out the reasons end-user participants did not see any harm in granting permissions (31.4%, Table 1):

- End-users are given a **choice** (11%): giving end-users the option to grant or deny permissions made some incorrectly believe that this choice left them with no harm. If they were to use certain functionality, there would be no reason to consider other ramifications of granting permissions;

Table 1: End-User participants' perspectives on harms associated with permissions.

Possible Harms With Permissions		Permissions Are Harmless	
Access to protected resources	62%	End-users are given a choice	11%
Access to sensitive data	18%	No or minimal harm	8%
Security issues	18%	End-users' responsibility	5%
Selling data to third parties	12%	Trusting the developer	3%
Unsure about data practices of developers	3%	Trusting the law	2%
Depending on the app	6%	Privacy resignation	2%

- **No or minimal** harm (8%): on the other hand, some end-user participants incorrectly believed that permissions were often innocuous and that nothing bad was related to them. Few also thought that if there were no physical harm in granting permission, there would be no other harm;
- End-users' **responsibility** (5%): similar to what some developer participants thought that granting permissions was an end-user choice and responsibility, few end-user participants also believed that it was their responsibility to choose the required permissions. Therefore, they thought no harm could happen because they could choose and assert their control over the app;
- **Trusting the developer** (3%): trust came up as a factor like those who saw harm with apps. Depending on the app, how much end-user participants trusted the developer, and whether an app was from a reputed source, a few end-user participants saw no harm in permissions;
- **Trusting the law** (2%): while trusting the developer was a reason to believe an app was harmless, few end-user participants believed laws should protect them against any possible harm; and
- **Privacy resignation** (2%): few end-user participants gave up on their privacy and said that their privacy would be invaded regardless of their decision. Few also said they had nothing to hide, so there was nothing to worry about.

4.4 Reasons for Adding, Updating, or Removing Permissions

Developer Participant Perspectives. The reasons developer participants added, updated, and removed permission requests were:

Features & Requirements. All developer participants agreed that app **features and requirements** were the primary reason for adding permissions. Based on what the client requested or the app required, developers added permissions. For example, a developer participant added the location permission because their client needed to keep track of parcels. Some developer participants acknowledged that adding **too many permissions** without a specific functionality or purpose could be **user-unfriendly**. Therefore, they kept their permission requests proportionate to the app's features.

Libraries. To function correctly, some **libraries** required permissions, which was the second primary reason for developer participants to include permissions. We discuss the reasons for including or excluding libraries by developer participants in §4.7.

Common Practice. Many developer participants added permissions because of their everyday practices. The main reason was **reusing** a component in several projects because the apps they developed were similar, or projects **shared features**. For example, having a map in different apps required location permission, or most apps needed to connect to a server; hence, including Internet permission was considered standard practice. We also found that some developer participants copied and pasted code from the Internet, echoing prior work's findings [23], or reused their own written code; both of these practices may leave unused or unnecessary permissions in the app.

Operating Systems & App Stores. Many developer participants updated their permissions to **satisfy operating system and app store requirements**. They changed the permissions they used due to changes in operating system permissions. Similarly, some received rejections from the Google Play Store or the Apple App Store after submitting their apps for review, making them change the permissions they initially added. We also explicitly asked developer participants about any changes they had to make to their app permissions to comply with *privacy laws*. There was a mix-up among some developer participants over privacy laws and Apple's or Google's terms of service and privacy policies. However, all developer participants noted that they did not change their apps and permissions to specifically comply with privacy laws, such as the General Data Protection Regulation (GDPR) [90], CCPA, or COPPA (However, investigating how developers comply with privacy laws was not the focus of our study. Therefore, we did not ask follow-up questions about this topic, see §3.3).

App Development Life Cycle. Similar to what prior work has found [23, 71], some of our developer participants updated their permissions because of the **app development life cycle**. Some developer participants, for example, updated permissions due to requirement changes or app revisions. However, only a few developer participants mentioned they changed their permissions during testing as opposed to what prior work has suggested [23]. The difference could be due to the different research methods employed; Felt et al. [23] analyzed Android apps (i.e., artifact analysis). We instead used direct retrospective interviews. Our developer participants might have changed their permissions during the testing and production stages but did not recall doing so during the interviews.

Code or App Crashes. Some developer participants added permissions because they **received errors or warnings** from the development tool. As the developers' first point of contact, development tools can play a significant role in how developers write

code and what they need to add to their code. Designing useful and enhanced compiler error messages shows promise and can assist novice developers in programming [10]. Therefore, lessons learned from prior studies could be used as a stepping stone to nudging developers toward permissions' privacy ramifications.

Workarounds. Instead of adding permissions, few developer participants found other ways to access data **without asking end-users** or requesting permissions. For accessing location, for example, a few developer participants used the IP address instead of the location permission: "The user will not give permission to disclose their location to the application . . . we can use other options to specify the regions specify the location. So, we tried to get the location from the IP." (P2)

One developer participant used the photo gallery permission instead of the camera permission because the gallery permission was viewed as an easier-to-get request, in their opinion. In this case, end-users were asked by the app to take a photo, save it to the photo gallery, and then share it with the app. Another developer participant had permission requests in the code but did not ask for them in earlier app revisions, so no permissions were requested when the app was released. In a later revision, they showed the permission requests to end-users when the permissions were needed. This was done by switching a Boolean flag in the code to true when a specific permission request was needed. Another developer participant considered using the list of installed apps for fingerprinting; accessing the list of apps did not require a permission request and could be used to identify a device uniquely. The use of different methods to circumvent permissions is not new. Some large companies use multiple data points (e.g., network data) that do not require explicit end-user permission to infer end-users' location [30].

End-User Participant Perspectives. Most end-user participants **had not changed or rarely changed** their smartphone permission settings (never changed: 34%; rarely changed: 35%, Appendix F, Figure 6). Below, we discuss end-user participants' reasons for granting and (not) removing permissions (Table 2):

Reasons for Granting Permissions. The question about the reasons for granting permissions was from Bonné et al. (2017) [11] in which they did a longitudinal 6-week study with 157 participants to understand end-users' decision-making processes with regard to permissions. In the following paragraphs, we include percentages from our end-users and [11] for comparison; the first number in the parenthesis is ours, and the second number is from [11].

Like our developer participants, end-user participants also mentioned that the primary reason for granting permissions was **functionality** and features (79.6% vs. 68.2%), echoing prior findings [16, 97]. Else, they thought apps would not work if certain permissions were not granted (43.7% vs. 23.8%). **Trust** was the other reason for granting permissions (28.2% vs. 32.1%)—similarly, our developer participants wanted to build trust with end-user participants.

Related to the end-user experience theme from our developer interview study (§4.5), some end-user participants granted permission requests because they **wanted the screen to go away** (18.4% vs. 10.2%). A takeaway from our developer study is that developer participants tried not to flood the end-user with permission requests. However, some end-user participants were **overwhelmed** by permissions and wanted them to disappear (Figure 3 also echoes this finding). Besides, some end-user participants said they granted permissions because they had **nothing to hide** (15.5% vs. 18%), and few thought that **nothing bad** would happen (7.1% vs. 14%).

The app's popularity was not a significant decision-making factor (6.8% vs. 9.3%); few end-user participants thought that developers already had data about them (4.9% vs. 13%)—suggesting that they viewed permissions as a guard against their data, and few noted that they could grant the permission later if they wanted to (2.6% vs. 1.4%). Figure 8 in Appendix F shows the summary of these reasons for granting permissions.

Reasons for Not Removing Permissions. 57.9% of end-user participants had not ever removed permissions (Table 2):

- **No need** (27%): some end-user participants never felt the need to remove permissions or thought it was unnecessary;
- Comfortable in & **confident** about granted permissions (10%): some end-user participants were happy with what they had already granted to the apps and did not see a reason to remove permissions;
- **Do not know how** (8%): some end-user participants had no idea how to remove permissions;
- **Trusting the app** (6%): some end-user participants trusted the app and did not see a reason for removing an already given permission;
- **Functionality** (3%): the app provided functionality with the permissions. Therefore, few end-user participants saw a need to remove permissions or else the app might break;
- **Forgotten permissions** (2%): few end-user participants had not thought about removing or had forgotten that they had granted permissions to an app that might need removing;
- **Delete** app instead of removing permission (2%): few end-user participants decided to delete an app that had unnecessary permissions entirely instead of removing the permission, which seemed to be a practice for few participants when they saw too many permissions or felt uncomfortable with permissions; and
- **No harm** (2%): in line with the associated harms, few end-user participants saw no harm in permissions and had no reason to remove an already given permission.

Reasons for Removing Permissions. 42.1% of end-user participants said that they had removed an already given permission from an app (Table 2):

- **No longer needed** (21%): the primary reason to remove permissions was that the feature was no longer needed, such as giving permission to use photos for a certain time and later removing that permission;
- Feeling worried or **uncomfortable** (16%): several end-user participants stated discomfort with having permissions left granted (e.g., contacts, location, and microphone);
- **Loss of trust** (4%): following up on the trust theme described above in the developer and end-user studies, a few end-user participants removed permissions because they saw the news about an app's privacy-unfriendly practices that made them lose their trust and, hence, removed permissions;

Table 2: End-user participants' reasons (not mutually exclusive) for granting, removing, and not removing permissions.

Reasons for Granting Permissions		Reasons for Not Removing Permissions		Reasons for Removing Permissions	
Functionality & features	79.6%	No need	27%	No longer needed	21%
App won't work if perms weren't granted	43.7%	Comfortable about granted perms	10%	Feeling worried or uncomfortable	16%
Trusting the developer	28.2%	Do not know how	8%	Loss of trust	4%
Wanted the screen to go away	18.4%	Trusting the app	6%	Accidental perms	4%
I've nothing to hide	15.5%	Functionality	3%	Deleted the app	2%
Nothing bad would happen	7.1%	Forgotten perms	2%	Battery drainage	2%
App is popular	6.8%	Deleted app instead of removing perm	2%		
Developer already had the information	4.9%	No harm	2%		
Won't be able to grant later	2.6%				

- **Accidental** permissions (4%): few end-user participants granted permissions accidentally during installation or without carefully reading the permission descriptions. After a while, they realized that was a mistake and removed the permissions;
- **Deleted** app (2%): few end-user participants also completely deleted the app when they found that permissions were unnecessary instead of removing them; and
- **Battery** drainage (2%): few end-user participants also explicitly mentioned that they removed permissions because of battery consumption, especially when the permissions were related to some background activity.

4.5 Considering End-Users & Their Experience

Developer Participant Perspectives: Permission Descriptions. Many developer participants made an effort to write understandable descriptions. They acknowledged the value of **writing clear, transparent, and convincing descriptions** to help end-users understand why permission was needed and why access to a specific resource could help the app function correctly. Some developer participants were also motivated by app stores' requirements to provide clear and informative descriptions rather than short descriptions. Some developer participants wrote the description text, some received **help** from their product or design team, and others **tweaked predefined text** from online resources, operating systems, or app stores. Few developer participants had to localize their permission descriptions to **suit their end-user group**. They had end-users rejecting or blindly accepting permission requests because end-users could not understand the text.

Current permission dialogs only contain text, and some developer participants had to show **an extra page** ahead of the built-in permission dialog presented by the operating system to explain in more detail why the permission was requested: "The official dialogue has minimal space to explain the user. So the additional dialogue helps the user understand more about the permission thing, and also, we can customize the dialog." (P3) Although Apple recommends presenting end-users with a "pre-alert screen" before asking for sensitive permissions like location [5], the additional screen or dialog may put an extra burden on the end-user because of extra clicks and interfaces (which may result in negatively impacting the app's rating [62]) as well as the developer (e.g., extra work to create an additional dialog).

In 2014, in the early days of permission descriptions, some developers needed help understanding the value of these descriptions, saw them as unnecessary, and did not fully adopt them [26]. None of our developer participants questioned the existence of these descriptions; many even had favorable perspectives. The push from operating systems and app stores might be the driving force for such a shift in less than ten years, which echos other findings from privacy-related posts on developer forums that operating systems drive the app privacy ecosystem [47, 87].

End-User Participant Perspectives. End-user participants had **mixed opinions** about reading and checking permission descriptions (a little or less: 39.8%, a moderate amount: 30.4%, and a lot or more: 29.8%, see Appendix F, Figure 6).

Developer Participant Perspectives: Assumptions About End-Users. We observed the following assumptions that developer participants made about end-users: (1) end-users are **only sensitive** about specific permissions and grant consent to others, (2) end-users **are not aware** of or do not know much about permissions, (3) end-users **do not care** about permissions as long as the app provides the expected functionality, and (4) **end-users are the responsible party** for any consequences of accepting permissions. For example, a developer participant viewed the location permission as easy to get, whereas the email permission as difficult to get, which could even dissuade end-users from using an app:

> If you give permission for the location, the user can easily allow them, but you can use the permission for messages and emails permission then some users have not provided that and they can choose the alternative app for that because the SMS and mail permissions are very sensitive for the users. (P13)

While the location permission can be viewed as sensitive by Google [33], getting this permission from our developer participants' perspective might be easy. Developer participants might understand the market and end-users' expectations by working with end-users. Permissions like contacts and messages (which may cause losing contacts data or sending messages that can cost money) can be perceived as more sensitive than the location permission by developers [11, 25], which is a more classic privacy-sensitive permission from a research viewpoint [25].

End-User Participant Perspectives. Most end-user participants **disagreed** with the developer participants' assumption that they **did not care** about permission requests (somewhat disagree or less: 81.2%) or that there was no need for permission requests (somewhat disagree or less: 79%). They strongly believed that permissions

were necessary and were here to stay. Most end-user participants were sensitive about certain permissions, as some developer participants assumed about end-users. End-user participants also felt more comfortable granting a permission request if the app came from a known source, such as a large or well-reputed company (we discussed reputation and trust in §4.4). Most end-user participants **agreed** with the developer participants' assumption that it was their **responsibility** to grant or not grant a permission request (somewhat agree or more: 87.4%). Nevertheless, many thought the current permission descriptions were broad and needed more precise and transparent descriptions (Figure 3).

Developer Participant Perspectives: Request Permissions When Needed. Many developer participants tried to request permissions **when they needed it** or when there was a specific use case or functionality related to it. They also acknowledged that end-users should be notified about accessing or using resources on their devices by the app. These included thinking about the end-user experience and **trying not to flood the end-user** with many permission requests, as it could diminish the end-user experience and result in losing end-users.

Some developer participants also mentioned the importance of **trust and reputation** developers (or apps) built over time with end-users. Asking for **too many permissions** might result in **eroding trust** and negatively affecting the **end-user-developer relationship**. Maintaining this relationship could benefit developers in the long run, as end-user retention is one of the key determinants of a successful app [2, 12, 78]. While we did not explicitly ask about breaching end-user trust, one developer participant admitted that they had **misused the contacts data** they collected in a personally-developed app by contacting someone they were not supposed to contact (they asked the contacted person to stop harassing one of the developer participant's family members). Such access would not have been given without the end-user trusting the app or its developer not to misuse their contacts.

End-User Participant Perspectives. Some end-user participants **trusted** apps with **fewer permissions** (somewhat agree or more: 42.1%, somewhat disagree or less: 21.4%, Figure 3). They also had mixed opinions over the association of permission requests with the app being perceived as safe or end-user participants feeling comfortable using it (somewhat agree or more: 23%, somewhat disagree or less: 40.5%). These mixed opinions may suggest that some end-user participants were aware of other ways of exploiting apps without requesting permissions (e.g., using an IP address instead of location data, as suggested by a few developer participants, §4.4).

4.6 Developers' Challenges When Working With Permissions

Most developer participants expressed satisfaction with the current permission integration mechanism. They mainly relied on the documentation provided by Apple and Google to learn more about permissions and clarify any confusions they had. They also consulted other resources such as Stack Overflow, GitHub, online tutorials and forums, YouTube videos, and web searches. Below, we discuss our developer participants' pain points in the process of permission integration:

Manual Changes & Checks. Many had to manually update permissions because of *updates* to operating systems and libraries. Participants preferred that permissions were updated automatically without requiring manual changes. They also wanted libraries' permissions to get integrated automatically into the apps without manual involvement: "That third party library is using it, but you have to ask for the permission by yourself." (P14)

Late Feedback. Developer participants often got notified about **additional or unexpected permissions** *after* they developed their app, either from app stores or their testing or design teams. The app rejection from app stores was the primary pain point, causing developer participants to wait for post-app submission feedback and then go back to the project to fix any newly flagged issues. In Android, few developer participants had to remove extra library permissions that they were unaware of because of receiving **warnings from Google Play**, suggesting that nudging developers toward reducing app permissions [60, 61] seemed to be noticed by our developer participants and was viewed as a working mechanism:

> We offloaded the application on Google Play Store, and it showed some warnings about the permissions, and it was like, 'I have never put these permissions inside the app. Where did this come from?' and then yeah, it was from a library. What I did was that I added some commands to just remove the unnecessary permissions, and the library worked. (P5)

Confusing Scopes. When developer participants had to choose which permission to use for a feature, they were often **unsure** what permission provided access to what resource. The mapping between the functionality required by an app and the needed permissions was a scoping issue that some developer participants had trouble understanding. It was even more challenging when **multiple** permissions were used for a **similar purpose**. For example, the location permission comes in various forms on both Android and iOS; developers can limit location data based on whether the app is running in the background or not, as well as the level of detail (e.g., coarse vs. fine) [8, 35]. Because of this confusion, some developer participants **included all** similar permissions to avoid missing a use case that required access to some needed data or resources and to prevent app crashes. P6, for example, wished for better documentation of permissions and their privacy ramifications: "Documentation around what exactly each of these offer in terms of like sort of a reference functionality and perhaps including the privacy concerns around each of those." (P6)

Poor Documentation. Developer participants viewed the documentation provided by Android and iOS overall as **satisfactory**. However, some had issues understanding the documentation, **blindly trusted the sample code** given there, and wanted to see more code samples and guidelines explaining how to fix specific errors. A few developer participants stated that they fully trusted the documentation—setting a high standard for privacy from operating system providers—as these developers said they would copy-paste code samples from the documentation directly into their apps.

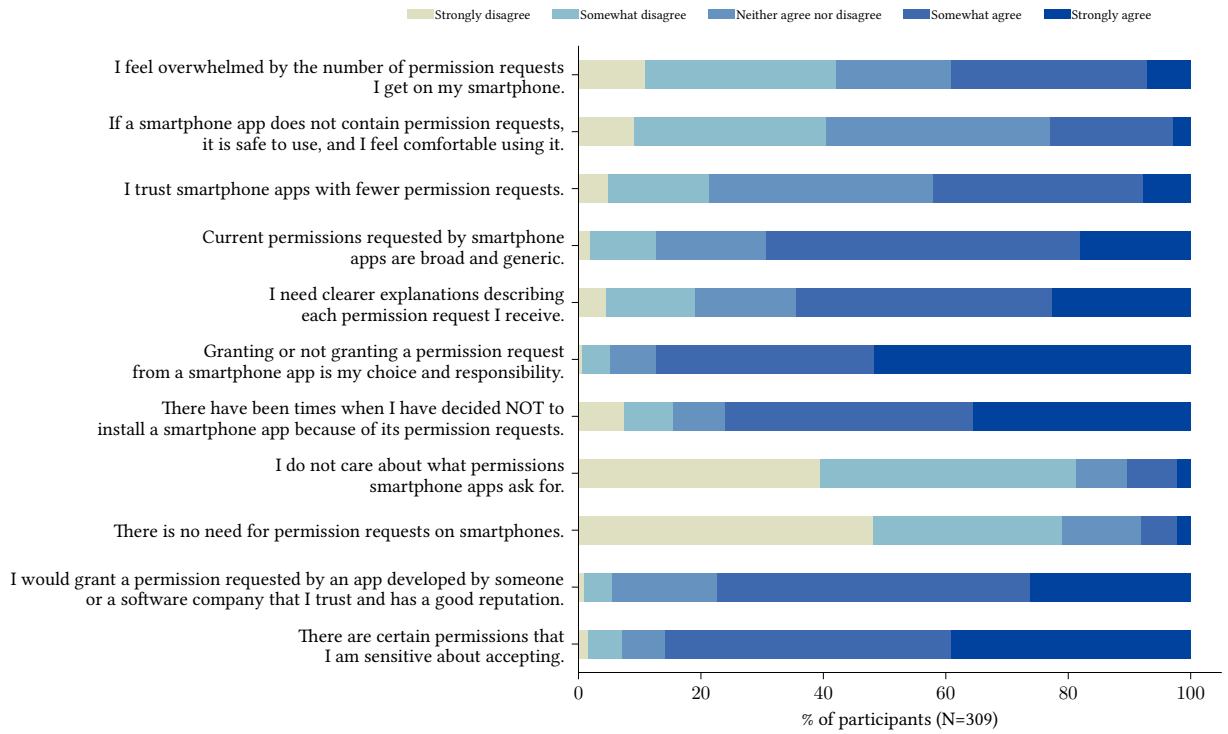

Figure 3: End-User participants' views compared to developer participants' views on and assumptions about end-users.

Not Informing Developers of Permission Changes by the End-User. Few developer participants mentioned that they had trouble knowing **whether or not an end-user granted** permissions. In some cases, Apple's HealthKit, for example, does not inform developers about end-users' choices when (not) granting access to sensitive resources like health data to protect end-users [7].

Limited Support for Hybrid Development. Few developer participants with hybrid app development experience (i.e., writing code once using a framework and exporting the app to several platforms) noted that they had to **keep track** of all the changes in Android and iOS, as well as understand the **different permission models** of the two operating systems.

4.7 Developers' Reasons for (Not) Using Software Development Libraries

Functionality. The primary reason for (not) using a library was its functionality, including features, ease of coding (**usability**), and adding control over code. Cross-platform libraries such as React Native and Flutter were the most mentioned libraries. In particular, some developer participants included libraries to **facilitate permission management**. Conversely, a library could have **too many features**, resulting in requesting **too many permissions**, which might result in the library getting removed from the project. In such scenarios, developer participants might decide to use an **alternative** library or write code from scratch.

Common Practice. Some developer participants added a library because it was part of their **typical programming behavior**. Some basic libraries facilitated connection establishment, dependency management, permission management, image optimization, interface design, database management, and analytics.

Verified by Others. When thinking about what libraries to include or exclude, some developer participants relied on what their **company or client approved**, **open-source** libraries hosted on GitHub, or libraries provided by operating system creators, **large technology companies** (e.g., Apple and Google), or a combination of these stakeholders.

Some developer participants viewed open-source libraries as reliable sources—even though Android repositories on GitHub could have permission-related issues, such as unused permissions [71]. A developer participant mentioned licensing issues with open-source projects that could complicate the use of such libraries.

Privacy Ramifications. In some cases, developer participants were concerned about libraries' **complicated and murky practices** that they could not understand. Therefore, they decided to **reconsider** using such libraries. Third parties (e.g., ad networks and analytics services) could collect data from end-users without developers' knowledge, or the privacy settings of these libraries could be complicated and hard to find, putting an additional burden on developer participants [3, 67, 81]:

> There is no way for us to know that a random SDK that we've pulled in they must be doing date changing or color monitoring—or it might be monitoring your app behind the scenes—actually we don't know. The

> whole thing is insanely complicated and gave me a really big headache. (P19)

Rejection From App Stores. A pain point for a few developer participants was getting an app rejected from app stores, as noted in §4.6. From a different perspective, this could make developer participants **review** their libraries and ensure that they did not collect **unnecessary** data or ask for unnecessary permissions:

> We submitted the app for Apple to review the app and then they said that the ad framework should not be there but if we remove the ad framework which we told them that if we remove the ad framework, Google Analytics will not work itself so ultimately, we had to remove Google analytics because there was no use of keeping the analytics without ad framework on for iOS. (P1)

Considering End-Users & Their Experience. In some cases, developer participants decided to exclude a library because including it could have resulted in **poor end-user experience** (see §4.5 for details about the end-user experience).

5 DISCUSSION AND FUTURE WORK

5.1 Two Sides of the Same Coin: Developer & End-User Perspectives on Permissions

Both our developer and end-user participants acknowledged that privacy was at the heart of the app ecosystem, as well as that permissions were a key mechanism to protect end-users' privacy. Comparing the two stakeholders with the limitation that the data collection methods employed in this work are not the same gives us the following insights:

5.1.1 Are Permission Requests Fully Grounded in App Functionality & Features? Both stakeholders agreed that app functionality was the primary reason for including or granting permissions (§4.4). Similar to what prior studies have found, many of our end-user participants (79.6%) tended to grant permissions necessary for the app to function, showing the importance of why permissions need to be contextualized and relate to functionality [54, 97]. However, context is not static and, in practice, nor are permissions. End-users should be able to maintain permissions depending on context (they may deny a permission request in one context but let it run indefinitely in another until the need to revoke). This begs the question of whether permissions could help (empower) end-users to manage access to data and resources by a non-random app or whether they always need to grant specific permissions; otherwise, the app would not function.

On a similar note, we found that developer participants also made an effort to make permissions contextualized. However, developer participants might face challenges contextualizing all permissions because they could end up including more permissions than needed due to being confused about the scope of some permissions or fear of an app crash. Further, some developer participants mentioned that they needed to include certain third-party libraries in their apps, which requested more permissions than what apps needed, making contextualization difficult. One option would be for developers to write their own libraries, which undermines the concept of object-oriented software development and reusing software libraries.

Future work may want to employ the theory of Contextual Integrity [59] to explain, in a usable way, to end-users (or developers) the data flows that occur as a consequence of granting (or including) a permission request, giving more information to help make an informed decision. For example, end-users (or developers) may find denying (or excluding) a permission request by an app (or a third-party library) in certain contexts to be appropriate, even if they end up not being able to use the app (or library).

5.1.2 Trust & Reputation as Decision-Making Factors. One recurring theme in both studies revolved around trust and reputation. Several developer participants recognized the value of building a relationship with end-users based on trust by requesting fewer permissions. Further, almost one-third of end-user participants trusted apps that asked for fewer or necessary permissions, echoing prior work findings that "brand reputation" is one of the factors influencing end-users' decision to grant or deny a permission request [73].

5.1.3 Who Is Responsible? Based on our findings and prior work, we observed that developers, in practice, often shift the privacy ramifications of their development choices to end-users or platforms. Our results show that although several developer participants recognized the importance of not overwhelming end-users with unnecessary permission requests, they still thought end-users were ultimately responsible for deciding to (not) grant permissions requests. Further, most of our end-user participants believed this was their responsibility (§4.5). Prior work has also shown that developers often shift the ramifications of using software development platforms to platforms (e.g., ad networks [53, 82]). Despite this belief, many software development platforms view developers as fully responsible for their code and apps (e.g., ad networks [53, 82], Apple App Store [9], and Google Play Store [32]). Therefore, we believe this puts developers in a central place, as the *mediators* between platforms and end-users, with their decisions directly impacting themselves and end-users.

There is a need to raise awareness in the developer community, perhaps through developer forums or academic educational platforms, to create a sense of responsibility and empathy toward end-users. Developers should not be the only responsible entity in the app ecosystem. Instead, responsibility should be shared across different entities (i.e., platforms, developers, and end-users) to reduce the load and burden on developers; exploring avenues for sharing this responsibility could be a future research direction.

5.1.4 Permission Descriptions and Names Need a Fresh Look. Both developer participants and end-user participants agreed that transparent and clear permission descriptions and names (especially when asked at the right time [21]) would help end-users make informed decisions with regard to granting or denying a permission request (§4.5). About two-thirds of end-user participants, however, thought that current permission descriptions were broad and generic, expressing the need for more transparent and informative descriptions (Figure 3), confirming findings from Shen et al. [73]. However, existing permission dialogs need more space to include informative explanations, forcing developers to create custom pages if they want to provide additional details. The extra pages can cause

inconsistencies across apps and introduce an extra burden on developers. The usable security literature suggests that writing useful warning messages is a challenging task for developers [36]. Expecting developers to anticipate all possible contexts and use cases may be unrealistic. Therefore, future research may need to provide clear guidelines for developers on how to write consistent, contextual, transparent, and easy-to-digest permission descriptions (or experiment with visual metaphors) or improve the current permission dialogs of operating systems.

Similarly, some permissions have similar names but facilitate access to different types of data or resources on end-users' smartphones (e.g., coarse location, fine location), leading developers to get confused about the scope of such permissions. For example, several developer participants decided to include all permissions with similar scopes to ensure apps worked and did not crash (§4.6). This could lead to dire privacy ramifications for end-users (who may also get confused about what a permission request does). We believe that although some permissions have similar scopes, they are still different, and clear and descriptive names that match the exact access type of a permission request are needed.

5.1.5 Close the Gap Between End-Users' Beliefs, Comprehension & Knowledge. Most of our end-user participants believed that (1) they were aware of permission requests, (2) why permissions were requested, and (3) permission requests aligned with their expectations. However, we did not ask follow-up questions to test end-users' comprehension of different permission requests (see §4.2 and Figure 2). Besides our findings, a previous survey study has explored end-users' understanding of what different permission requests meant, finding that end-user participants struggled to correctly identify what a permission request did (only 6.1% of participants answered all comprehension questions correctly) [73]. The difference between our findings and Shen et al.'s findings may signal a gap between the "beliefs" and "comprehension and knowledge" components of the human-in-the-loop framework [17]. Future research may empirically investigate the consequences of this gap and suggest avenues to close this gap. As discussed in §5.1.4, one possible improvement would be to provide clear permission names and descriptions.

5.2 The Usual Suspects: Third-Party Data Practices Worry Developers & End-Users

A primary reason for including permissions by developer participants was to use third-party library features. Developers may feel a lack of power and control over third-party libraries, which could normalize the privacy-unfriendly practice of adding libraries requesting unneeded permissions. Future research is needed to understand the impact of normalizing privacy-unfriendly practices of developers on end-users. The literature on third-party privacy-unfriendly data practices is rich, and many studies have shown that third parties collect unwanted data from end-users, even sometimes without developers' knowledge [22, 66, 77]. We believe that libraries need to break down their features into smaller pieces and enable developers to choose desired features. Only permissions needed by the chosen features can be included in apps instead of adding a large library with many unnecessary features.

We also observed in some instances that developer participants associated the privacy ramifications of libraries with permissions requested by libraries—thinking that if a library did not require permissions, it would be safe to use, and there would be no privacy concerns. This mindset is only partially accurate, as libraries can still collect data from end-users even when permissions are not requested (e.g., common libraries used for testing, analytics, and databases [22]). To address this complication, we suggest that third-party libraries transparently communicate their data collection practices to developers through easy-to-understand interfaces, especially since some of our end-user participants (§4.3) were concerned about their data getting harvested and sold to third parties. We also feel that apps should communicate to end-users the practices of these libraries. We speculate that apps that do so would have a competitive advantage over other apps that are not transparent, as privacy-conscious end-users may prefer to use apps that are advertised as privacy-friendly. However, this assumption requires further future research for validation.

5.3 Forgotten Permissions: Possibly Rooted in Scope Confusions & Lack of Awareness of Revoking Procedures

We found that developer participants had several reasons for adding permissions, but they rarely had a reason to remove permissions (§4.4). Some said that they worried that removing permissions could cause an app crash or an error. Therefore, they would not want to try removing permissions from a working app—this is a common issue in apps [95]. Some had trouble understanding the scope of some permissions leading to adding multiple permissions (e.g., location in use, location always) to avoid errors. Lessons learned from creating usable cryptography libraries for developers suggest that providing several implementation routes or using unfamiliar names for variables and functions may eventually lead to insecure actions by developers [36]. Similarly, as discussed in §5.1, we believe permission names and descriptions can benefit from clear naming and scopes to help developers decide on permission requests.

On the other hand, some end-user participants stated they had no idea how to remove permissions (57.9%). While there has been research on how to present permission requests to end-users (e.g., [72, 98]), we are unaware of specific studies that have explored ways to help end-users find and then remove unused or unnecessary permissions. Although some end-user participants were aware of permission settings and tried to change these settings (e.g., 28.68% of Android participants, N=380 [4]), our findings suggest that more work is needed (§4.4). Future research should consider informing end-users and developers about unused or unnecessary permissions periodically—one starting point could be using recommendation tools to periodically nudge and remind end-users and developers of permissions requested by apps, as inspired by Liu et al. [49].

A mechanism that notifies developers of unused and unnecessary permissions *during* development (instead of post submission to the app stores) or when releasing a revision could be a starting point to help developers. It could nudge developers toward removing unused permissions and remind them that end-users prefer apps with fewer permissions. For example, programming plugins for privacy checks (e.g., [46]) could implement such nudges for permissions. Relatedly, Android's recent addition to automatically check unused

permissions for end-users and developers shows promise and highlights the importance of the problem [31, 34]. However, neither our developer nor end-user participants mentioned those checks, which could signal a lack of awareness. iOS, on the other hand, does not provide automatic revoking of unused permissions. Therefore, future research may examine the usability and effectiveness of these recent Android additions and procedures.

6 CONCLUSION

We interviewed 19 developers and surveyed 309 Android and iOS end-users to empirically study how these two stakeholders understand and interact with permissions. The mixed-methods design of our study allowed us to draw contrasting and diverse perspectives on permissions. Both developer and end-user participants associated permissions with app functionality and features. However, sometimes developer participants included multiple permissions due to confusion about the scope of certain permissions or third-party library requirements, which could worry both developers and end-users. We also discuss the implications of our findings and how to improve permissions for both developers and end-users.

ACKNOWLEDGMENTS

We thank CHI23's anonymous reviewers, Pauline Anthonysamy, Tianshi Li, Ola Michalec, Marvin Ramokapane, Jose Such, William Seymour, and Xiao Zhan for their constructive feedback that helped improve the paper. The U.K.'s National Research Centre on Privacy, Harm Reduction, and Adversarial Influence Online (REPHRAIN) partly sponsored this research (EPSRC: EP/V011189/1). Mohammad Tahaei's research and contribution were made when he was at the University of Bristol.

REFERENCES

[1] Desiree Abrokwa, Shruti Das, Omer Akgul, and Michelle L. Mazurek. 2021. Comparing Security and Privacy Attitudes Among U.S. Users of Different Smartphone and Smart-Speaker Platforms. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, 139–158. https://www.usenix.org/conference/soups2021/presentation/abrokwa

[2] Ruba Abu-Salma, M. Angela Sasse, Joseph Bonneau, Anastasia Danilova, Alena Naiakshina, and Matthew Smith. 2017. Obstacles to the Adoption of Secure Communication Tools. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 137–153. https://doi.org/10.1109/SP.2017.65

[3] Noura Alomar and Serge Egelman. 2022. Developers Say the Darnedest Things: Privacy Compliance Processes Followed by Developers of Child-Directed Apps. *Proceedings on Privacy Enhancing Technologies* 2022, 4 (2022), 24 pages. https://doi.org/10.2478/popets-2022-0108

[4] Ashwaq Alsoubai, Reza Ghaiumy Anaraky, Yao Li, Xinru Page, Bart Knijnenburg, and Pamela J. Wisniewski. 2022. Permission vs. App Limiters: Profiling Smartphone Users to Understand Differing Strategies for Mobile Privacy Management. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. ACM, 18 pages. https://doi.org/10.1145/3491102.3517652

[5] Apple. 2022. *Accessing private data.* Apple. Retrieved August 2022 from https://developer.apple.com/design/human-interface-guidelines/patterns/accessing-private-data/

[6] Apple. 2022. *Control access to information in apps on iPhone.* Apple. Retrieved August 2022 from https://support.apple.com/en-gb/guide/iphone/iph251e92810/ios

[7] Apple. 2022. *Protecting User Privacy.* Apple. Retrieved August 2022 from https://developer.apple.com/documentation/healthkit/protecting_user_privacy

[8] Apple. 2022. *Requesting Authorization for Location Services.* Apple. Retrieved August 2022 from https://developer.apple.com/documentation/corelocation/requesting_authorization_for_location_services

[9] Apple. 2022. *User Privacy and Data Use.* Apple. Retrieved November 2022 from https://developer.apple.com/app-store/user-privacy-and-data-use/

[10] Brett A. Becker, Paul Denny, Raymond Pettit, Durell Bouchard, Dennis J. Bouvier, Brian Harrington, Amir Kamil, Amey Karkare, Chris McDonald, Peter-Michael Osera, Janice L. Pearce, and James Prather. 2019. Compiler Error Messages Considered Unhelpful: The Landscape of Text-Based Programming Error Message Research. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education (ITiCSE-WGR '19)*. ACM, 177–210. https://doi.org/10.1145/3344429.3372508

[11] Bram Bonné, Sai Teja Peddinti, Igor Bilogrevic, and Nina Taft. 2017. Exploring Decision Making with Android's Runtime Permission Dialogs Using in-Context Surveys. In *Proceedings of the Thirteenth USENIX Conference on Usable Privacy and Security (SOUPS '17)*. USENIX Association, 195–210. https://www.usenix.org/system/files/conference/soups2017/soups2017-bonne.pdf

[12] Emily Bonnie. 2022. *The Mobile Marketer's Guide to Mastering User Retention.* Clever Tap. Retrieved August 2022 from https://clevertap.com/blog/guide-to-user-retention/

[13] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101. https://doi.org/10.1191/1478088706qp063oa

[14] Karoline Busse, Mohammad Tahaei, Katharina Krombholz, Emanuel von Zezschwitz, Matthew Smith, Jing Tian, and Wenyuan Xu. 2020. Cash, Cards or Cryptocurrencies? A Study of Payment Culture in Four Countries. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 200–209. https://doi.org/10.1109/EuroSPW51379.2020.00035

[15] Paolo Calciati, Konstantin Kuznetsov, Alessandra Gorla, and Andreas Zeller. 2020. *Automatically Granted Permissions in Android Apps: An Empirical Study on Their Prevalence and on the Potential Threats for Privacy.* ACM, 114–124. https://doi.org/10.1145/3379597.3387469

[16] Weicheng Cao, Chunqiu Xia, Sai Teja Peddinti, David Lie, Nina Taft, and Lisa M. Austin. 2021. A Large Scale Study of User Behavior, Expectations and Engagement with Android Permissions. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 803–820. https://www.usenix.org/conference/usenixsecurity21/presentation/cao-weicheng

[17] Lorrie Faith Cranor. 2008. A Framework for Reasoning about the Human in the Loop. In *Proceedings of the 1st Conference on Usability, Psychology, and Security (UPSEC'08)*. USENIX Association, 15 pages. https://www.usenix.org/legacy/events/upsec08/tech/full_papers/cranor/cranor.pdf

[18] Anastasia Danilova, Alena Naiakshina, Stefan Horstmann, and Matthew Smith. 2021. Do you Really Code? Designing and Evaluating Screening Questions for Online Surveys with Programmers. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE Computer Society, 537–548. https://doi.org/10.1109/ICSE43902.2021.00057

[19] Kenan Degirmenci. 2020. Mobile users' information privacy concerns and the role of app permission requests. *International Journal of Information Management* 50 (2020), 261–272. https://doi.org/10.1016/j.ijinfomgt.2019.05.010

[20] Verena Distler, Matthias Fassl, Hana Habib, Katharina Krombholz, Gabriele Lenzini, Carine Lallemand, Lorrie Faith Cranor, and Vincent Koenig. 2021. A Systematic Literature Review of Empirical Methods and Risk Representation in Usable Privacy and Security Research. *ACM Trans. Comput.-Hum. Interact.* 28, 6 (Dec 2021), 50 pages. https://doi.org/10.1145/3469845

[21] Yusra Elbitar, Michael Schilling, Trung Tin Nguyen, Michael Backes, and Sven Bugiel. 2021. Explanation Beats Context: The Effect of Timing & Rationales on Users' Runtime Permission Decisions. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 785–802. https://www.usenix.org/conference/usenixsecurity21/presentation/elbitar

[22] Álvaro Feal, Julien Gamba, Narseo Vallina-Rodriguez, Primal Wijesekera, Joel Reardon, Serge Egelman, and Juan Estévez Tapiador. 2021. Don't Accept Candy from Strangers: An Analysis of Third-Party Mobile SDKs. In *Data Protection and Privacy, Volume 13: Data Protection and Artificial Intelligence*. Bloomsbury Publishing. https://hdl.handle.net/20.500.12761/779

[23] Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. 2011. Android Permissions Demystified. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS '11)*. ACM, 627–638. https://doi.org/10.1145/2046707.2046779

[24] Adrienne Porter Felt, Serge Egelman, Matthew Finifter, Devdatta Akhawe, and David Wagner. 2012. How to Ask for Permission. In *7th USENIX Workshop on Hot Topics in Security (HotSec 12)*. USENIX Association, 6 pages. https://www.usenix.org/conference/hotsec12/workshop-program/presentation/Felt

[25] Adrienne Porter Felt, Serge Egelman, and David Wagner. 2012. I've Got 99 Problems, but Vibration Ain't One: A Survey of Smartphone Users' Concerns. In *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '12)*. ACM, 33–44. https://doi.org/10.1145/2381934.2381943

[26] Adrienne Porter Felt, Kate Greenwood, and David Wagner. 2011. The Effectiveness of Application Permissions. In *Proceedings of the 2nd USENIX Conference on Web Application Development (WebApps'11)*. USENIX Association, 7. https://www.usenix.org/legacy/events/webapps11/tech/final_files/Felt.pdf

[27] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. 2012. Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*

(SOUPS '12). ACM, 14 pages. https://doi.org/10.1145/2335356.2335360
[28] Freelancer. 2022. *Hire Freelancers & Find Freelance Jobs Online | Freelancer*. Freelancer. Retrieved August 2022 from https://www.freelancer.co.uk/
[29] Federal Trade Commission (FTC). 1998. *Children's Online Privacy Protection Rule (COPPA)*. Retrieved August 2022 from https://www.ftc.gov/legal-library/browse/rules/childrens-online-privacy-protection-rule-coppa
[30] Federal Trade Commission (FTC). 2016. *Mobile Advertising Network InMobi Settles FTC Charges It Tracked Hundreds of Millions of Consumers' Locations Without Permission*. Retrieved August 2022 from https://www.ftc.gov/news-events/news/press-releases/2016/06/mobile-advertising-network-inmobi-settles-ftc-charges-it-tracked-hundreds\protect\discretionary{\char\hyphenchar\font}{}{}millions-consumers
[31] Google. 2022. *Context | Android Developers*. Google. Retrieved September 2022 from https://developer.android.com/reference/android/content/Context.html#revokeSelfPermissionOnKill(java.lang.String)
[32] Google. 2022. *Developer Policy Center*. Google. Retrieved November 2022 from https://play.google.com/about/developer-content-policy/
[33] Google. 2022. *Permissions on Android*. Google. Retrieved August 2022 from https://developer.android.com/guide/topics/permissions/overview
[34] Google. 2022. *Permissions updates in Android 11*. Google. Retrieved September 2022 from https://developer.android.com/about/versions/11/privacy/permissions
[35] Google. 2022. *Request location permissions*. Google. Retrieved August 2022 from https://developer.android.com/training/location/permissions
[36] Matthew Green and Matthew Smith. 2016. Developers are Not the Enemy! The Need for Usable Security APIs. *IEEE Security & Privacy* 14, 5 (2016), 40–46. https://doi.org/10.1109/MSP.2016.111
[37] Marco Gutfleisch, Jan H. Klemmer, Niklas Busch, Yasemin Acar, M. Angela Sasse, and Sascha Fahl. 2022. How Does Usable Security (Not) End Up in Software Products? Results From a Qualitative Interview Study. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 893–910. https://doi.org/10.1109/SP46214.2022.9833756
[38] David Harborth and Alisa Frik. 2021. Evaluating and Redefining Smartphone Permissions with Contextualized Justifications for Mobile Augmented Reality Apps. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, 513–534. https://www.usenix.org/conference/soups2021/presentation/harborth
[39] Hamza Harkous, Sai Teja Peddinti, Rishabh Khandelwal, Animesh Srivastava, and Nina Taft. 2022. Hark: A Deep Learning System for Navigating Privacy Feedback at Scale. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 1562–1562. https://doi.org/10.1109/SP46214.2022.00133
[40] QSR International. 2022. *Qualitative Data Analysis Software | NVivo*. QSR International. Retrieved May 2022 from https://www.qsrinternational.com/nvivo-qualitative-data-analysis-software/home/
[41] Qatrunnada Ismail, Tousif Ahmed, Kelly Caine, Apu Kapadia, and Michael Reiter. 2017. To Permit or Not to Permit, That is the Usability Question: Crowdsourcing Mobile Apps' Privacy Permission Settings. *Proceedings on Privacy Enhancing Technologies* 2017, 4 (2017), 119–137. https://doi.org/10.1515/popets-2017-0041
[42] Ankit Kariryaa, Gian-Luca Savino, Carolin Stellmacher, and Johannes Schöning. 2021. Understanding Users' Knowledge about the Privacy and Security of Browser Extensions. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, 99–118. https://www.usenix.org/conference/soups2021/presentation/kariryaa
[43] Konrad Kollnig, Anastasia Shuba, Reuben Binns, Max Van Kleek, and Nigel Shadbolt. 2022. Are iPhones Really Better for Privacy? A Comparative Study of iOS and Android Apps. *Proceedings on Privacy Enhancing Technologies* 2022, 2 (2022), 6–24. https://doi.org/10.2478/popets-2022-0033
[44] J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics* 33, 1 (1977), 159–174. https://doi.org/10.2307/2529310
[45] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. 2017. Chapter 8 - Interviews and focus groups. In *Research Methods in Human Computer Interaction* (Second ed.), Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser (Eds.). Morgan Kaufmann, 187–228. https://doi.org/10.1016/B978-0-12-805390-4.00008-X
[46] Tianshi Li, Yuvraj Agarwal, and Jason I. Hong. 2018. Coconut: An IDE Plugin for Developing Privacy-Friendly Apps. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (Dec. 2018), 35 pages. https://doi.org/10.1145/3287056
[47] Tianshi Li, Elizabeth Louie, Laura Dabbish, and Jason I. Hong. 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 (Jan. 2021), 28 pages. https://doi.org/10.1145/3432919
[48] LinkedIn. 2022. *LinkedIn*. LinkedIn. Retrieved August 2022 from https://www.linkedin.com
[49] Bin Liu, Mads Schaarup Andersen, Florian Schaub, Hazim Almuhimedi, Shikun (Aerin) Zhang, Norman Sadeh, Yuvraj Agarwal, and Alessandro Acquisti. 2016. Follow My Recommendations: A Personalized Privacy Assistant for Mobile App Permissions. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. USENIX Association, 27–41. https://www.usenix.org/conference/soups2016/technical-sessions/presentation/liu
[50] Nathan Malkin, David Wagner, and Serge Egelman. 2022. Runtime Permissions for Privacy in Proactive Intelligent Assistants. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*. USENIX Association, 633–651. https://www.usenix.org/conference/soups2022/presentation/malkin
[51] Prashanthi Mallojula, Javaria Ahmad, Fengjun Li, and Bo Luo. 2021. You Are (not) Who Your Peers Are: Identification of Potentially Excessive Permission Requests in Android Apps. In *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 114–121. https://doi.org/10.1109/TrustCom53373.2021.00033
[52] Joseph A. Maxwell. 2010. Using Numbers in Qualitative Research. *Qualitative Inquiry* 16, 6 (2010), 475–482. https://doi.org/10.1177/1077800410364740
[53] Abraham H. Mhaidli, Yixin Zou, and Florian Schaub. 2019. "We Can't Live Without Them!" App Developers' Adoption of Ad Networks and Their Considerations of Consumer Risks. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*. USENIX Association, 20 pages. https://www.usenix.org/conference/soups2019/presentation/mhaidli
[54] Kristopher Micinski, Daniel Votipka, Rock Stevens, Nikolaos Kofinas, Michelle L. Mazurek, and Jeffrey S. Foster. 2017. User Interactions and Permission Use on Android. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, 362–373. https://doi.org/10.1145/3025453.3025706
[55] Matthew Miles and Michael Huberman. 1994. *Qualitative Data Analysis: A Methods Sourcebook*. Sage.
[56] Miro. 2022. *Miro | Online Whiteboard for Visual Collaboration*. Miro. Retrieved August 2022 from https://miro.com/
[57] Preksha Nema, Pauline Anthonysamy, Nina Taft, and Sai Teia Peddinti. 2022. Analyzing User Perspectives on Mobile App Privacy at Scale. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE, 112–124. https://doi.org/10.1145/3510003.3510079
[58] Duc Cuong Nguyen, Erik Derr, Michael Backes, and Sven Bugiel. 2019. Short Text, Large Effect: Measuring the Impact of User Reviews on Android App Security & Privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 555–569. https://doi.org/10.1109/SP.2019.00012
[59] Helen Nissenbaum. 2004. Privacy as Contextual Integrity. *Washington Law Review* 79 (2004), 119. https://digitalcommons.law.uw.edu/wlr/vol79/iss1/10/
[60] Sai Teja Peddinti, Igor Bilogrevic, Nina Taft, Martin Pelikan, Úlfar Erlingsson, Pauline Anthonysamy, and Giles Hogben. 2019. Reducing Permission Requests in Mobile Apps. In *Proceedings of the Internet Measurement Conference (IMC '19)*. ACM, 259–266. https://doi.org/10.1145/3355369.3355584
[61] Sai Teia Peddinti, Nina Taft, Igor Bilogrevic, and Pauline Anthonysamy. 2020. *Helping Developers with Permission Requests*. Google. Retrieved September 2022 from https://security.googleblog.com/2020/02/helping-developers-with-permission.html
[62] Anthony Peruma, Jeffrey Palmerino, and Daniel E. Krutz. 2018. Investigating User Perception and Comprehension of Android Permission Models. In *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft '18)*. ACM, 56–66. https://doi.org/10.1145/3197231.3197246
[63] Prolific. 2022. *Online participant recruitment for surveys and market research*. Prolific. Retrieved August 2022 from https://www.prolific.co
[64] Prolific. 2022. *Representative samples*. Prolific. Retrieved August 2022 from https://researcher-help.prolific.co/hc/en-gb/articles/360019236753-Representative-samples
[65] Abbas Razaghpanah, Rishab Nithyanand, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Mark Allman, Christian Kreibich, and Phillipa Gill. 2018. Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem. In *Proceedings Network and Distributed Systems Security (NDSS) Symposium 2018*. Internet Society, 15 pages. https://doi.org/10.14722/ndss.2018.23009
[66] Joel Reardon, Álvaro Feal, Primal Wijesekera, Amit Elazari Bar On, Narseo Vallina-Rodriguez, and Serge Egelman. 2019. 50 Ways to Leak Your Data: An Exploration of Apps' Circumvention of the Android Permissions System. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 603–620. https://www.usenix.org/conference/usenixsecurity19/presentation/reardon
[67] Irwin Reyes, Primal Wijesekera, Joel Reardon, Amit Elazari Bar On, Abbas Razaghpanah, Narseo Vallina-Rodriguez, and Serge Egelman. 2018. "Won't Somebody Think of the Children?" Examining COPPA Compliance at Scale. *Proceedings on Privacy Enhancing Technologies* 2018, 3 (2018), 63–83. https://doi.org/10.1515/popets-2018-0021
[68] Johnny Saldaña. 2015. *The Coding Manual for Qualitative Researchers*. Sage.
[69] Nithya Sambasivan, Garen Checkley, Amna Batool, Nova Ahmed, David Nemer, Laura Sanely Gaytán-Lugo, Tara Matthews, Sunny Consolvo, and Elizabeth Churchill. 2018. "Privacy is not for me, it's for those rich women": Performative Privacy Practices on Mobile Phones by Women in South Asia. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)*. USENIX Association, 127–142. https://www.usenix.org/conference/soups2018/presentation/sambasivan
[70] Yukiko Sawaya, Mahmood Sharif, Nicolas Christin, Ayumu Kubota, Akihiro Nakarai, and Akira Yamada. 2017. Self-Confidence Trumps Knowledge: A Cross-Cultural Study of Security Behavior. In *Proceedings of the 2017 CHI Conference*

on Human Factors in Computing Systems (CHI '17). ACM, 2202–2214. https://doi.org/10.1145/3025453.3025926
[71] Gian Luca Scoccia, Anthony Peruma, Virginia Pujols, Ivano Malavolta, and Daniel E. Krutz. 2019. Permission Issues in Open-Source Android Apps: An Exploratory Study. In *2019 19th International Working Conference on Source Code Analysis and Manipulation (SCAM)*. IEEE, 238–249. https://doi.org/10.1109/SCAM.2019.00034
[72] Gian Luca Scoccia, Stefano Ruberto, Ivano Malavolta, Marco Autili, and Paola Inverardi. 2018. An Investigation into Android Run-Time Permissions from the End Users' Perspective. In *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft '18)*. ACM, 45–55. https://doi.org/10.1145/3197231.3197236
[73] Bingyu Shen, Lili Wei, Chengcheng Xiang, Yudong Wu, Mingyao Shen, Yuanyuan Zhou, and Xinxin Jin. 2021. Can Systems Explain Permissions Better? Understanding Users' Misperceptions under Smartphone Runtime Permission Model. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 751–768. https://www.usenix.org/conference/usenixsecurity21/presentation/shen-bingyu
[74] Katie Shilton and Daniel Greene. 2019. Linking Platforms, Practices, and Developer Ethics: Levers for Privacy Discourse in Mobile Application Development. *Journal of Business Ethics* 155, 1 (March 2019), 131–146. https://doi.org/10.1007/s10551-017-3504-8
[75] Statista. 2020. *Software developer gender distribution worldwide*. Statista. Retrieved September 2022 from https://www.statista.com/statistics/1126823/worldwide-developer-gender/
[76] Statista. 2022. *Mobile operating systems' market share worldwide from January 2012 to January 2022*. Statista. Retrieved August 2022 from https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/
[77] Iraklis Symeonidis, Gergely Biczók, Fatemeh Shirazi, Cristina Pérez-Solà, Jessica Schroers, and Bart Preneel. 2018. Collateral damage of Facebook third-party applications: a comprehensive study. *Computers & Security* 77 (2018), 179–208. https://doi.org/10.1016/j.cose.2018.03.015
[78] Nayden tafradzhiyski. 2022. *Mobile App Retention*. Business of Apps. Retrieved August 2022 from https://www.businessofapps.com/guide/mobile-app-retention/
[79] Mohammad Tahaei, Alisa Frik, and Kami Vaniea. 2021. Deciding on Personalized Ads: Nudging Developers About User Privacy. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, 573–596. https://www.usenix.org/conference/soups2021/presentation/tahaei
[80] Mohammad Tahaei, Tianshi Li, and Kami Vaniea. 2022. Understanding Privacy-Related Advice on Stack Overflow. In *Proceedings on Privacy Enhancing Technologies (PETS)*. De Gruyter Open, 18 pages. https://doi.org/10.2478/popets-2022-0032
[81] Mohammad Tahaei, Kopo M. Ramokapane, Tianshi Li, Jason I. Hong, and Awais Rashid. 2022. Charting App Developers' Journey Through Privacy Regulation Features in Ad Networks. In *Proceedings on Privacy Enhancing Technologies*. De Gruyter Open, 1–24. https://doi.org/10.2478/popets-2022-0059
[82] Mohammad Tahaei and Kami Vaniea. 2021. "Developers Are Responsible": What Ad Networks Tell Developers About Privacy. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems Extended Abstracts*. ACM, 12 pages. https://doi.org/10.1145/3411763.3451805
[83] Mohammad Tahaei and Kami Vaniea. 2022. Lessons Learned from Recruiting Participants with Programming Skills for Empirical Privacy and Security Studies. 2 pages. https://ropes-workshops.github.io/ropes22/ 1st International Workshop on Recruiting Participants for Empirical Software Engineering, RoPES'22.
[84] Mohammad Tahaei and Kami Vaniea. 2022. Recruiting Participants With Programming Skills: A Comparison of Four Crowdsourcing Platforms and a CS Student Mailing List. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. ACM, Article 590, 15 pages. https://doi.org/10.1145/3491102.3501957
[85] Mohammad Tahaei, Kami Vaniea, Beznosov Konstantin, and Maria K. Wolters. 2021. Security Notifications in Static Analysis Tools: Developers' Attitudes, Comprehension, and Ability to Act on Them. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, 1–17. https://doi.org/10.1145/3411764.3445616
[86] Mohammad Tahaei, Kami Vaniea, and Awais Rashid. 2023. Embedding Privacy Into Design Through Software Developers: Challenges & Solutions. *IEEE Security & Privacy* Special Issue on Usable Security for Security Workers (2023), 11 pages. https://doi.org/10.1109/MSEC.2022.3204364
[87] Mohammad Tahaei, Kami Vaniea, and Naomi Saphra. 2020. Understanding Privacy-Related Questions on Stack Overflow. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM, 1–14. https://doi.org/10.1145/3313831.3376768
[88] Joshua Tan, Khanh Nguyen, Michael Theodorides, Heidi Negrón-Arroyo, Christopher Thompson, Serge Egelman, and David Wagner. 2014. The Effect of Developer-Specified Explanations for Permission Requests on Smartphone User Behavior. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, 91–100. https://doi.org/10.1145/2556288.2557400
[89] Jenny Tang, Eleanor Birrell, and Ada Lerner. 2022. Replication: How Well Do My Results Generalize Now? The External Validity of Online Privacy and Security Surveys. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*. USENIX Association, 367–385. https://www.usenix.org/conference/soups2022/presentation/tang
[90] The European Parliament and the Council of the European Union. 2018. *General Data Protection Regulation (GDPR)*. Retrieved September 2022 from https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679
[91] Twitter. 2022. *Twitter*. Twitter. Retrieved August 2022 from https://twitter.com
[92] Upwork. 2022. *Upwork | The World's Work Marketplace*. Upwork. Retrieved August 2022 from https://www.upwork.com/
[93] Christine Utz, Sabrina Amft, Martin Degeling, Thorsten Holz, Sascha Fahl, and Florian Schaub. 2023. Privacy Rarely Considered: Exploring Considerations in the Adoption of Third-Party Services by Websites. *Proceedings on Privacy Enhancing Technologies* 2023, 1 (2023), 25 pages. https://doi.org/10.48550/ARXIV.2203.11387
[94] Daniel Votipka, Desiree Abrokwa, and Michelle L. Mazurek. 2020. Building and Validating a Scale for Secure Software Development Self-Efficacy. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM, 1–20. https://doi.org/10.1145/3313831.3376754
[95] Sinan Wang, Yibo Wang, Xian Zhan, Ying Wang, Yepang Liu, Xiapu Luo, and Shing-Chi Cheung. 2022. Aper: Evolution-Aware Runtime Permission Misuse Detection for Android Apps. In *Proceedings of the 44th International Conference on Software Engineering (ICSE '22)*. ACM, 125–137. https://doi.org/10.1145/3510003.3510074
[96] Yang Wang, Huichuan Xia, and Yun Huang. 2016. Examining American and Chinese Internet Users' Contextual Privacy Preferences of Behavioral Advertising. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW '16)*. ACM, 539–552. https://doi.org/10.1145/2818048.2819941
[97] Primal Wijesekera, Arjun Baokar, Ashkan Hosseini, Serge Egelman, David Wagner, and Konstantin Beznosov. 2015. Android Permissions Remystified: A Field Study on Contextual Integrity. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, 499–514. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/wijesekera
[98] Primal Wijesekera, Joel Reardon, Irwin Reyes, Lynn Tsai, Jung-Wei Chen, Nathan Good, David Wagner, Konstantin Beznosov, and Serge Egelman. 2018. Contextualizing Privacy Decisions for Better Prediction (and Protection). In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, 1–13. https://doi.org/10.1145/3173574.3173842

A SCREENING SURVEY FOR THE INTERVIEW STUDY WITH DEVELOPERS

[Answer options to close-ended questions were randomized where appropriate.]

You will have 30 to 60 seconds to answer each question on the next four pages. Each page states the time.

- Which of these websites do you most frequently use as aid when programming?
  - Wikipedia
  - LinkedIn
  - Stack Overflow
  - MemoryAlpha
  - I haven't used any of the websites above for programming
  - I don't program
- Choose the answer that best fits the description of a compiler's function.
  - Refactoring code
  - Connecting to the network
  - Aggregating user data
  - Translating code into executable instructions
  - Collecting user data
  - I don't know
- Choose the answer that best fits the definition of a recursive function.
  - A function that runs for an infinite time

  - A function that does not have a return value
  - A function that can be called from other functions
  - A function that calls itself
  - A function that does not require any inputs
  - I don't know
- Which of these values would be the most fitting for a Boolean?
  - Small
  - Solid
  - Quadratic
  - Red
  - True
  - I don't know

The remaining questions are NOT timed.

- Please answer the next two questions, given the following pseudocode algorithm.

```
main {
    print(func("hello world"))
}

String func(String in ) {
    int x = len(in)
    String out = ""
    for (int i = x - 1; i >= 0; i--) {
        out.append(in [i])
    }
    return out
}
```

- What is the parameter of the function?
  - String out
  - String in
  - int i = x - 1; i >= 0; i- -
  - Outputting a String
  - int x = len(in)
  - I don't know
- Please select the returned value of the pseudocode above.
  - hello world
  - hello world 10
  - dlrow olleh
  - world hello
  - HELLO WORLD
  - hello world hello world hello world hello world
  - I don't know
- Please select the statement that best describes your primary role at your current or most recent job.
  - Jobs NOT related to computer science, informatics, computer engineering, or related fields
  - Designing products (e.g., UI designer, interaction designer)
  - Developing software (e.g., programmer, developer, web developer, software engineer)
  - Testing software (e.g., tester, quality analyst, automation engineer)
  - Managing software development (e.g., project manager, IT manager, scrum master)
  - Privacy and/or security engineering (e.g., security engineer, privacy engineer, penetration tester, ethical hacker, cryptographer)
  - Other (please specify)
- Are you a student?
  - Yes, I'm a student in computer science or related fields
  - Yes, I'm a student but NOT in computer science or related fields
  - No, I'm NOT a student
- How many years of experience do you have in software development? [Numbers only]
- How many years of experience do you have in Android programming? [Numbers only]
- How many years of experience do you have in iOS programming? [Numbers only]
- How many years old are you? [Numbers only]
- In which country do you currently reside? [List of countries]
- If you can't find your country in the above question options, please enter it here. [Open-ended question]
- What is your gender?
  - Male
  - Female
  - Non-binary
  - Prefer not to say
  - Prefer to self describe

B DEMOGRAPHICS FOR THE INTERVIEW STUDY WITH DEVELOPERS

Table 3 shows a summary of developer demographics.

C INTERVIEW GUIDE FOR THE INTERVIEW STUDY WITH DEVELOPERS

- Can you tell me about your job? What do you do?
- Can you tell me about the apps that you have developed?
  - What kind of apps do you develop?
  - What are the age groups of your users?
  - What kind of data do you collect in your apps?
- Have you ever needed to share data between different apps? [If yes]
  - Why, and how?
  - Has it been between the apps you developed or between other apps and your apps?
- What types of permissions do you often include in your apps?
- Is there a set of permissions you often include, or do you pick permissions per app?
- How do you decide on which permissions to include and which permissions not to include?
- How do you think permissions work?
- Do you ever update the permissions of your apps? [If yes]
  - Why?
  - How frequently?
  - Can you think of an example?
- Have you ever removed permissions? If so, why?
- What is the most confusing thing you have experienced when working with permissions?

Table 3: A summary of developer demographics.

PID	Yrs of expr in software dev	Yrs of expr in Android dev	Yrs of expr in iOS dev	Age	Continent of residence	Gender	Recruitment platform
P1	10	3	7	35–44	Asia	Male	Freelancer
P2	5	5	4	25–34	Asia	Male	Freelancer
P3	5	5	4	25–34	Asia	Male	Freelancer
P4	5	4	0	25–34	Europe	Male	LinkedIn
P5	22	13	0	35–44	Asia	Prefer not to say	LinkedIn
P6	6	3	2	25–34	North America	Male	Prolific
P7	22	1	10	35–44	North America	Male	Prolific
P8	5	0	4	18–24	Asia	Male	Freelancer
P9	5	5	1	25–34	Asia	Male	Freelancer
P10	7	0	5	25–34	Asia	Female	Freelancer
P11	8	8	1	25–34	Asia	Male	Upwork
P12	3	3	0	25–34	Africa	Female	Upwork
P13	5	5	5	25–34	Asia	Male	Upwork
P14	5	2	5	25–34	Asia	Male	Upwork
P15	5	3	1	25–34	Africa	Male	Upwork
P16	12	6	6	35–44	South America	Male	Upwork
P17	7	5	1	25–34	Europe	Prefer not to say	Upwork
P18	6	4	1	25–34	Africa	Male	Upwork
P19	8	8	8	25–34	Europe	Female	LinkedIn

  - What sources did you consult to sort out the confusion?
- How do you think your choices of permissions impact your users' choices of apps?
  - What do you think the reasons are behind users accepting or rejecting permissions?
  - How do you think as a developer you can better guide your users to accept permissions?
  - How do you decide on what to include in the descriptions?
  - How does adding descriptions impact your work?
- Do you include third-party libraries or SDKs in your apps? [such as ads, logging tools, and analytics] [If yes]
  - How often?
  - Can you name a few that you often include in your apps? For what functionality do you use the SDKs?
  - How do you decide on permissions of these libraries?
  - How do you know what permissions the library needs?
  - Have you ever decided not to include a library because of its permissions?
- Now, I am going to send you a list of permissions in the chat. [We showed participants the 15 most commonly-used permissions in Android, iOS, or both, taken from Kollnig et al. [43], based on participants' platform expertise.] I would like you to have a look at them and tell me in your opinion how each permission behaves, or what it does, when you include the permission in your app.
  - Please go back to the list and tell me which ones do you think may have a privacy consequence for your users? Why, and how?
- What data do you consider private or sensitive for your users?
- Have you ever heard about any privacy regulations?
  - Have you considered making any of your apps compliant with any privacy regulations? If yes, why and how?
  - Have you made any changes to your permissions because of a privacy regulation?
- Can you think about any potential harms of permissions to your users?
  - Can you think of any privacy consequences of permissions for your users?
- How would you want to see permissions managed in mobile apps to make it easier for you to manage and integrate permissions?
- If you were to redesign the permission models, how would you do it?
  - What information and support should be included in mobile operating systems to help you better understand the language, options, and interfaces of permissions?
- Would you like to share any other experiences related to our conversation today, specifically about permission models in smartphone operating systems?

D CODEBOOK FOR THE INTERVIEW STUDY WITH DEVELOPERS

Table 4 shows the codebook and the number of unique participants mentioning each theme.

E QUESTIONNAIRE FOR THE SURVEY STUDY WITH END-USERS

[Answer options to close-ended questions were randomized where appropriate.]

Table 4: Codebook for the interview study with developers. Occurrences show the unique number of participants in each theme.

Theme	Occurrences	Theme	Occurrences
Reasons for adding, updating, or removing permissions	19	*Reasons for (not) using software development libraries*	19
Features & requirements	19	Reasons for exclusion	14
Libraries	17	Permission management	14
Common practice	16	Extra or unnecessary features	6
Operating systems & app stores	15	Adds or loses control over code	5
App development life cycle	7	Privacy implications	6
Code or app Crashes	5	App rejection from app store	5
Workarounds	4	End-users and UX	5
		Reasons for inclusion	19
Developers' challenges when working with permissions	19	Functionality	19
Manual changes & checks	15	Adds control over code	5
Late feedback	13	Common practice	14
Confusing scopes	12	Usable	13
Poor documentation	9	Verified by others	12
Not informing developers of permission changes by the end-user	5	Company, client-approved	6
Limited support for hybrid development	2	Open-source	6
		Operating systems or large tech companies	4
Considering End-Users & their experiences	19	Permission management	6
Permission descriptions	18		
Localization of permission descriptions	3	*Privacy conceptualization*	19
Who writes permission descriptions	15	Security implies privacy	14
Developer writes text	12	Privacy measures	8
Predefined text for perms	5	Nuanced permissions	7
Someone else provides the text	6	Processing data locally	3
Assumptions about end-users	14		
End-Users are sensitive about certain permissions	12	*Why do permissions exist?*	19
End-Users don't know (aren't aware)	7	Access control	19
End-Users don't care	5	User consent for accessing a resource	18
Shifting responsibility to end-users	12	Terms like "turning on or off a switch"	7
Request permissions when needed	15	As a key to a door	3
End-Users should be notified	11	As a bridge	2
Trust & reputation	9		

- On average, how many hours do you spend on your primary smartphone on a daily basis? Please estimate your daily average usage in hours. [Slider 0–24]
- For how many years have you been using a smartphone? (Numbers only)
- Please name five smartphone apps that you have frequently used in the past year? Use commas to separate the items. For example, app1, app2, app3, app4, app5.
- In this survey, we use the term "permissions" or "permission dialogs" to refer to the following example dialogs in smartphone operating systems like Android and iOS. [Example dialogs from Android and iOS documentation were included]
- In one sentence, why do smartphone apps request permissions? There is no right or wrong answer, please use your own words, as we are not looking for a correct or a perfect answer.
- In one sentence, how do you explain what permissions requested by apps do in smartphones? There is no right or wrong answer, please use your own words, as we are not looking for a correct or a perfect answer.
- Please select the top five permission requests that you have frequently seen in the past year on your smartphone? – Photos or Gallery – Location – Camera – Internet – Contacts – Microphone – Sensors or Motion – Calendars – Bluetooth – Biometrics (e.g., Touch ID or Face ID) – Storage or files – WiFi or Network state
- If a smartphone app uses permissions, it means that the app . . .
  – Can see the content of all the files on the device you are using
  – Is not a risk to infect your device with a computer virus
  – Will automatically prompt you to update your web browser software if it is out of date
  – Can access certain resources on your device
  – I don't know
- Why do you choose to grant permissions requested by smartphone apps? (Select all that apply)
  – I want to use a specific feature that requires permission
  – I think the app will not work otherwise
  – I trust the app developer
  – Because the app is popular
  – I will not be able to grant permissions later
  – I have nothing to hide
  – I want the permission screen to go away
  – Nothing bad will happen
  – The app developer already has the requested information about me
  – I don't know

- To what extent do you agree or disagree with the following statements? [Scale: Strongly disagree, Somewhat disagree, Neither agree nor disagree, Somewhat agree, Strongly agree]
  – There are certain permissions that I am sensitive about accepting.
  – I do not care about what permissions smartphone apps ask for.
  – There have been times when I have decided NOT to install a smartphone app because of its permission requests.
  – Granting or not granting a permission request from a smartphone app is my choice and responsibility.
  – I need clearer explanations describing each permission request I receive.
  – Current permissions requested by smartphone apps are broad and generic.
  – I trust smartphone apps with fewer permission requests.
  – If a smartphone app does not contain permission requests, it is safe to use, and I feel comfortable using it.
  – I feel overwhelmed by the number of permission requests I get on my smartphone.
  – I would grant permission requested by an app developed by someone or a software company that I trust and has a good reputation.
  – There is no need for permission requests on smartphones.
- Thinking about the five smartphone apps that you have frequently used in the past year, to what extent do you agree or disagree with the following statements? [Scale: Strongly disagree, Somewhat disagree, Neither agree nor disagree, Somewhat agree, Strongly agree]
  – I am aware of the permissions used in the smartphone apps that I frequently use. – Permissions requested by the smartphone apps that I frequently use are aligned with my expectations.
  – I understand why certain permissions are requested by smartphone apps that I frequently use.
- Have you ever removed permission that you have granted?
  – Yes, I have removed permission that I granted before. Please explain why and how.
  – No, I have not ever removed permission that I granted before. Please explain why.
- Do you think permissions requested by a smartphone app you use could harm you?
  – Yes, I can see harm in permission requests. Please explain why
  – No, I do not see any harm in permission requests. Please explain why.
- In your opinion, to what extent does granting the following permissions on your device have privacy implications for you? [Scale: None at all, A little, A moderate amount, A lot, A great deal]
  – Photos or Gallery – Location – Camera – Internet – Contacts – Microphone – Sensors or Motion – Calendars – Bluetooth – Biometrics (e.g., Touch ID or Face ID) – Storage or files – WiFi or Network state
- Please rate the following statements. [Scale: None at all, A little, A moderate amount, A lot, A great deal]
  – How often do you check the permissions requested by an app before installing it on your smartphone?
  – How often do you read the description and explanation of permission dialogs?
  – How often do you contact the developer of a smartphone app to ask for more information and clarifications about the app's permissions?
  – How often do you change the settings of your smartphone to manage the permissions of a smartphone app? For example, to remove an already given permission or to grant an already removed permission.
- To what extent do you agree or disagree with the following statements? [Scale: Strongly disagree, Somewhat disagree, Neither agree nor disagree, Somewhat agree, Strongly agree]
  – After installing an app on my smartphone, it would bother me if the app started asking me to grant permissions.
  – If I installed an app on my smartphone, I would think twice before granting any of the app permissions.
  – If I accepted app permissions, my activities on my smartphone would be monitored or tracked at least part of the time.
  – If I accepted app permissions, I would be concerned that the app would know more information about me.
  – If I accepted app permissions, I would be concerned that the app would monitor my activities on my smartphone.
  – If I accepted app permissions, others would know about me more than I would be comfortable with.
  – If I accepted app permissions, information about me that I consider to be private or sensitive would be more easily accessible to others than I would want.
  – If I accepted app permissions, information about me would be out there. Also, if that information were used, my privacy would be invaded.
  – If I accepted app permissions, I would be concerned that the app might use my personal information for other purposes without notifying me or getting my authorization.
  – If I accepted app permissions, I would be concerned that the app might use my information for other purposes.
  – If I accepted app permissions, I would be concerned that the app might share my personal information with other entities without getting my authorization.

F ADDITIONAL PLOTS FOR THE SURVEY STUDY WITH END-USERS

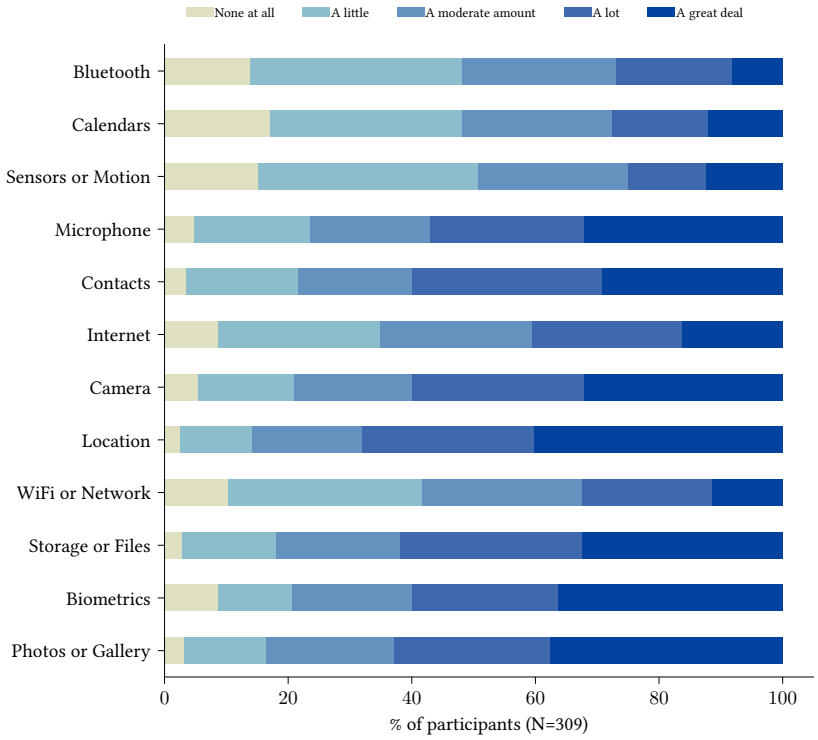

Figure 4: End-User participants' answers to "In your opinion, to what extent does granting the following permissions on your device have privacy implications for you?"

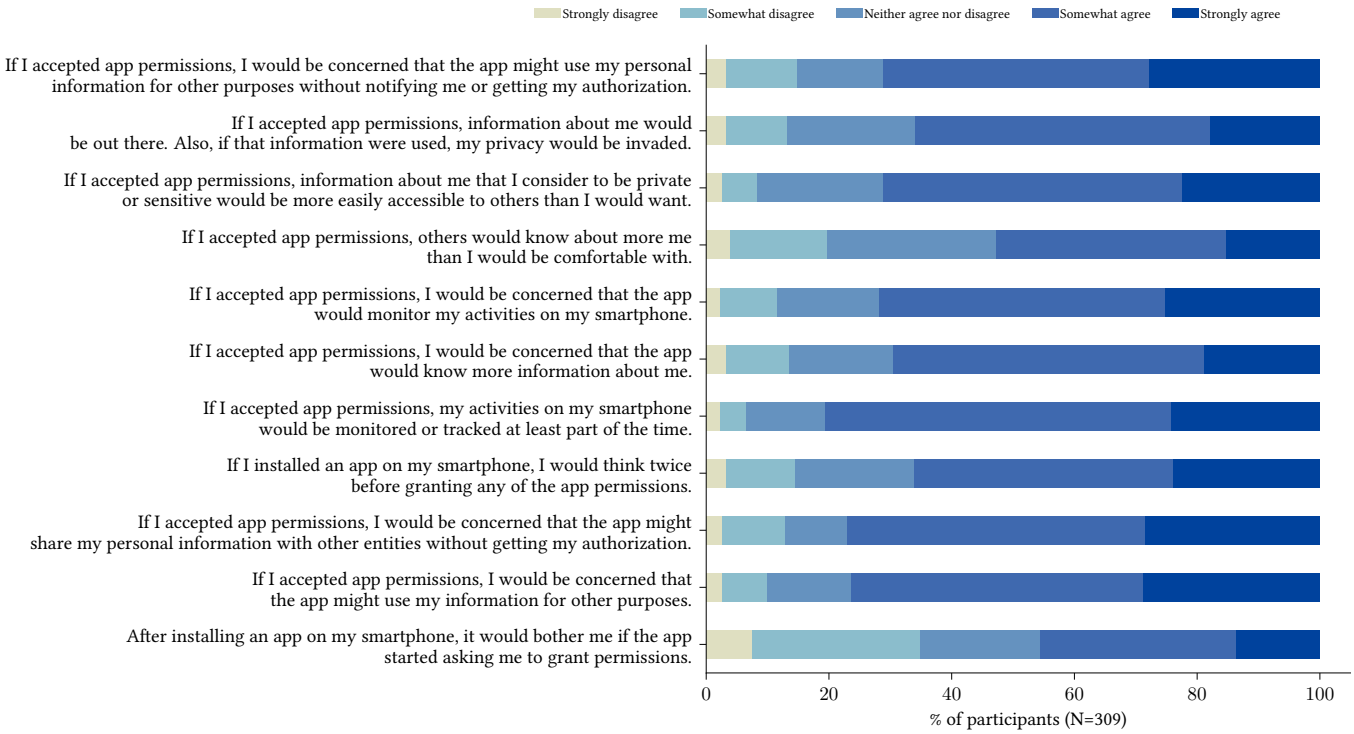

Figure 5: End-User participants' answers to questions about privacy ramifications of permissions.

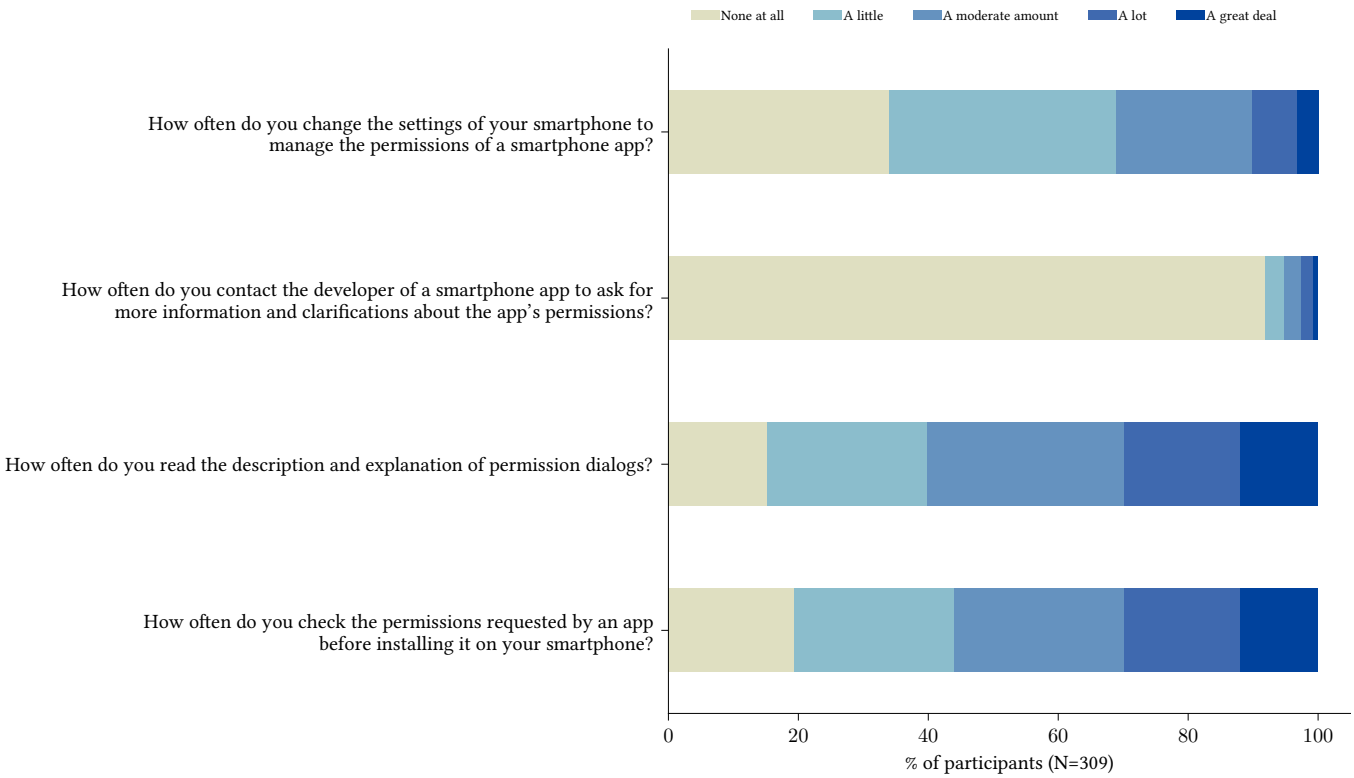

Figure 6: End-User participants' answers to questions rooted in the developer study.

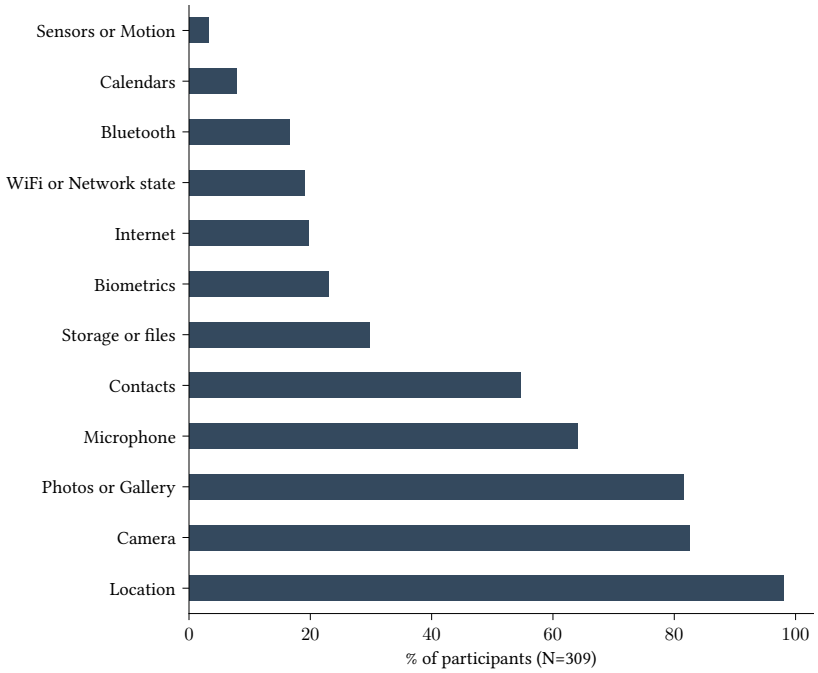

Figure 7: End-User participants' answers to "Please select the top five permission requests that you have frequently seen in the past year on your smartphone?"

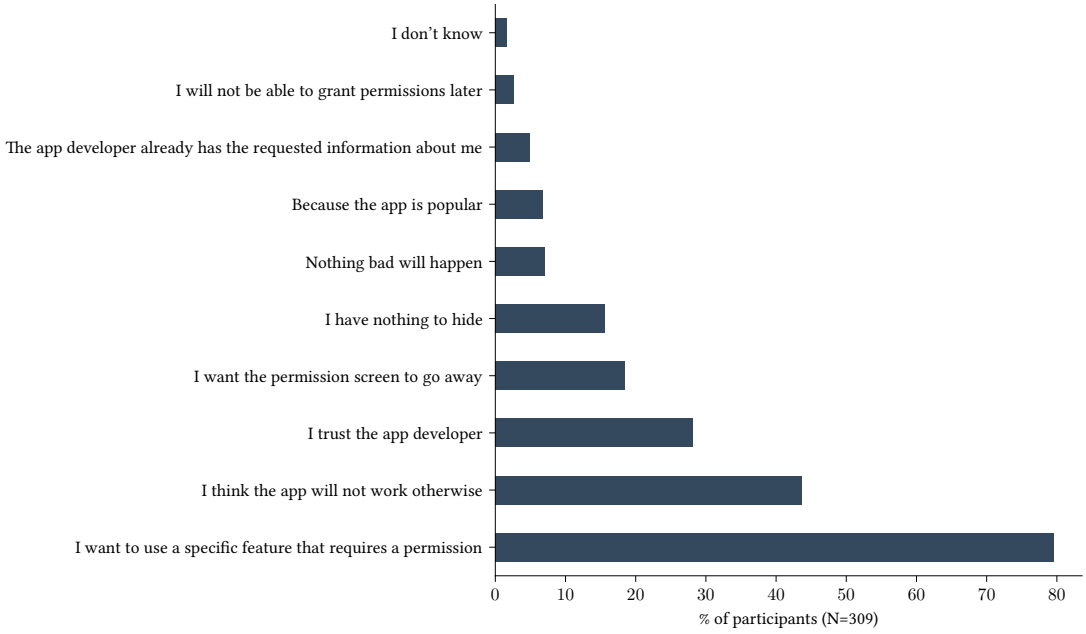

Figure 8: End-User participants' answers to "Why do you choose to grant permissions requested by smartphone apps? (Select all that apply)."