Rethinking App Permissions on iOS

Michael Lutaaya
Carleton University
Ottawa, Canada
michael.lutaaya@carleton.ca

Abstract
To protect against misuse, mobile operating systems require apps to seek user permission before accessing their personal data. While this measure gives users control, they are often asked for their data without knowing who the data will ultimately be shared with, why, and how often. This paper presents results from the evaluation of a proof-of-concept permission manager for iOS that allows users to adjust privileges granted to apps, substitute their personal data with mock data, and review data that has been transmitted to a server.

Author Keywords
Usable security; privacy; app permissions; design.

ACM Classification Keywords
H.5.2 [Information interfaces and presentation (e.g., HCI)]: User interfaces---prototyping

Introduction
While apps have many potential benefits for mobile users, their access to user data may still result in the disclosure of personal information to unauthorized third parties. Permission managers allow users to adjust the access privileges given to apps installed on their mobile device, but despite these controls, there

Nielsen's *10 Usability Heuristics for User Interface Design* [8]

1. Visibility of System Status
2. Match Between System and Real World
3. User Control and Freedom
4. Consistency and Standards
5. Error Prevention
6. Recognition Over Recall
7. Flexibility and Efficiency of Use
8. Aesthetic and Minimalist Design
9. Help Users Recognize, Diagnose, and Recover From Errors
10. Help and Documentation

Nielsen's *Severity Ratings for Usability Problems* [9]
0: Not a usability problem
1: Cosmetic problem
2: Minor usability problem
3: Major usability problem
4: Usability catastrophe

are fundamental flaws that limit the extent to which users can protect their personal information. Specifically, without adequate knowledge and understanding of how their personal data is used by apps and without control over the data that apps can access, users remain vulnerable.

The main contribution of this paper is the development and heuristic evaluation of a proof-of-concept iOS permission manager that can: grant and revoke access to personal data; maintain a record of apps that have accessed personal data; and provide installed apps with mock data instead of personal data.

Related Work

Several researchers investigating app permissions identify a need for greater clarity regarding protection of privacy so that users can better understand the implications of their decisions and actions [2–4, 11]. Users also struggle to link the scope of a permission with the risks involved in giving permission. These researchers suggest that user-facing text focus on risks instead of the resources involved [5].

In their research, Liu et al. [7] developed an app that makes suggestions regarding permission decisions based on users' privacy preferences. Similarly, Balebako et al. developed an app named Privacy Leaks [2] which notifies users about the transmission of personal data. In both studies, users requested more granularity and control over their data sharing.

Much of the literature on permission management on mobile devices analyzes the Android operating system. While Android and iOS differ fundamentally, they converge in how they approach permission management. For example, both systems prompt the user the first time that an app requests access to a protected resource at runtime. Also, both systems have a built-in permission manager app enabling users to retroactively grant or revoke an app's access to specific data. Furthermore, for certain categories of personal information, iOS uses blatant signals when data is being accessed. For instance, when an app uses the microphone in the background, a red bar appears until the recording ends. iOS also has additional controls relating to location [1]. Despite these enhancements, iOS and Android still do not make information available about frequency of use, reason for use, and whether data has been transmitted over the network, and do not allow users to customize the individual pieces of information made available to apps (e.g., preventing an app from seeing particular photos or calendar events).

Design and Implementation

Based on our literature review, we determined that our new permission manager would require: (i) the ability to enable and disable an app's access to personal data; (ii) a log showing instances where user data was accessed; and (iii) the ability to replace an app's access to real user data with fabricated data or data taken from a public repository. We followed an iterative design process consisting of six iterations. We progressed from low- to high-fidelity prototypes, receiving feedback at each iteration.

This work was influenced by limitations related to developing system-level tools for a closed-source operating system such as iOS. On iOS, apps are sandboxed, meaning that they can not access the data of other apps, and are limited to on-device data made accessible through developer APIs.

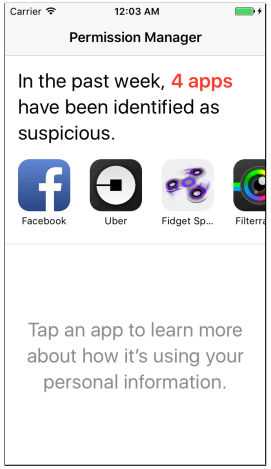

Figure 1: The permission manager's summary screen.

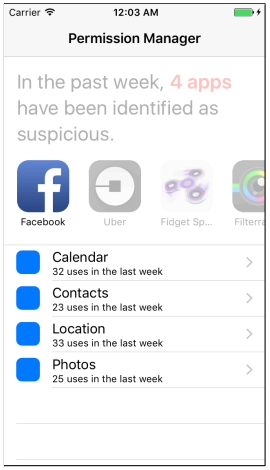

Figure 2: The permission list.

To overcome these limitations, our high-fidelity prototype consisted of an iOS app and a server-side app. The server-side app acts as a stand-in for the operations and on-device data that only system apps are authorized to use (e.g., controlling access to protected resources and collecting data on frequency of use by installed apps).

This approach relied on Ruby on Rails' seeding feature and real apps from the App Store to generate mock data on the server-side that simulates what a real user might experience. The iOS client app delegates responsibilities like data storage and querying to the server. The client app uses the server's JSON API to fetch and parse data before displaying it to the user. The app itself was developed using Xcode 8.3.3, and was written with Swift 3.2.

Final Prototype

In the final prototype, the summary screen (see Figure 1) is displayed when the app first launches. It displays icons of apps that have behaved suspiciously in the last week as determined by the permission manager. Tapping an icon displays the categories of permissions that can be adjusted for the app (see Figure 2). Selecting one of those categories takes the user to the permission detail screen.

On the permission detail screen (see Figure 3), users can (i) grant or revoke the app's access to data from the selected category of permissions, (ii) choose to provide the app with their real data or bogus data, (iii) preview examples of the information that is visible to the app, and (iv) view dates when the app transmitted data from the selected category of permissions to a remote server.

Tapping on a date shows the event list screen (see Figure 4) where each table row represents the time of day where the app transmitted data to a server. Users can inspect these communications in greater detail on the event detail screen by tapping on a row (see Figure 5).

Heuristic Evaluation

We conducted a heuristic evaluation of our high-fidelity prototype to identify usability issues [10]. Three individuals with experience in HCI and usable security/privacy conducted separate heuristic evaluations. Each evaluation session took approximately one hour. Each session began with the evaluators completing a first pass of the app to familiarize themselves with how it works. Then, they were given a list of tasks that a typical app user might perform, and used Nielsen's heuristics (see Page 2) [10] to discover usability issues within the app. Finally, they recorded their findings in an electronic document and assigned each issue a value from Nielsen's severity ratings scale (see Page 2) to signal its importance.

Results

After the evaluations, we consolidated the issues that were identified to eliminate duplicates. Further, for each of the unique usability issues, an average severity rating was calculated using the rating(s) of the evaluator(s) that had discovered the issue. In total, the evaluators identified 17 unique usability issues. Based on the average severity ratings, six of the issues were rated as cosmetic, six as minor, four as major, and only one as a usability catastrophe.

Figure 6 shows the number of usability issues attributed to each heuristic. Heuristic #2 (Match Be-

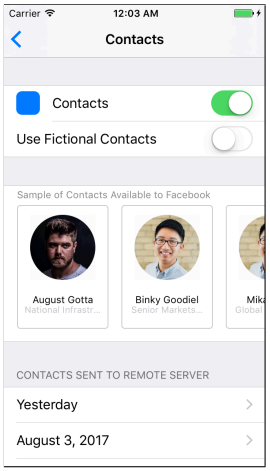

Figure 3: The permission detail screen.

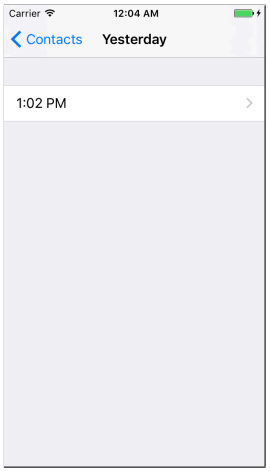

Figure 4: The event list screen.

tween System and Real World) and Heuristic #8 (Aesthetic and Minimalist Design) were the most frequently violated heuristics (seven and six violations, respectively) with approximately 76% of the usability issues being linked to these heuristics. This statistic indicates a need for the information provided by the app to be better tailored to its users as well as the need to revise various user interface elements to prioritize essential information.

Although approximately 71% of the usability problems identified during the heuristic evaluation were either cosmetic or minor, the evaluation also revealed usability issues that have major implications for future development. These issues are as follows:

Poorly Defined Terminology: The evaluators remarked that when the list of "suspicious" apps is presented to the user, it is unclear what "suspicious" means and why the manager chose to bring those particular apps to the user's attention. Similarly, the term "personal information" was found to be too broad. While the app uses the term to refer to the categories of permissions that it monitors, an evaluator noted that the lack of precision could lead users to misinterpret the term as encompassing additional categories beyond what the permission manager supports.

Additional Context Necessary: Some of the information provided to users is not valuable enough on its own. Evaluators found that on the event details screen, including a server's IP address or domain name is not helpful without also giving details such as whether or not that server is controlled by the app developer. In a similar manner, on the event details screen, the app attempts to classify the transmission into one of several categories (e.g., Advertising, Social Networking (see Figure 7)). In this case, users are neither advised on how accurate the classification is nor are they told what the category entails.

Too Idealistic: It was noted by one evaluator that, in reality, apps are often unusable when a permission request is denied. It was argued that allowing users to toggle permissions on and off without also explaining the trade-offs could be problematic. However, it is important to note that this is no different than the existing behaviour of permission managers. In most cases where the app is unusable, developers communicate to the user that the permission must be enabled before proceeding.

The following two themes were rated as major usability problems by evaluators, but reflect relatively simple usability problems that occurred because evaluators were examining a prototype system.

Not Optimized to Fit Content on Screen: The evaluators identified several places in the app, including the event detail screen and the summary screen, where information was truncated because it was too long. Further, when viewing samples of data, the size of the enclosing view meant that content such as maps and photos were constrained could only be so large.

Missing Icons Create Confusion: One evaluator remarked that, as seen in Figure 8, the app's icon on the operating system's home screen caused them to think that the app was in an unfinished state (e.g., still downloading). Evaluators also commented on the icons used in the permission list with one evaluator saying that the icons initially led them to the incorrect conclusion that they represented whether or not the permission had been enabled (see Figure 9).

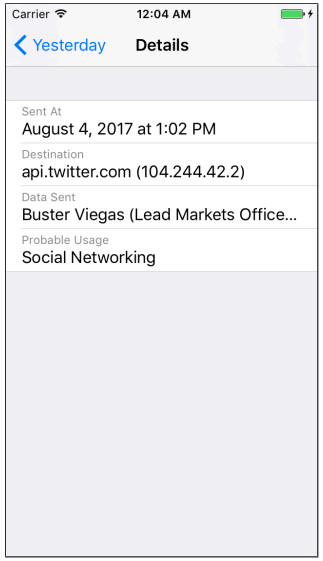

Figure 5: The event detail screen.

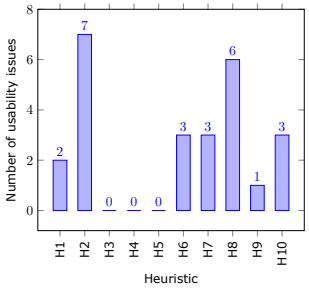

Figure 6: Number of usability issues per heuristic

In both instances, default or temporary icons had been used as placeholders due to time constraints.

As well, since heuristic evaluations are designed to uncover usability issues, the format of this evaluation did not encourage the evaluators to identify the components of the app that were successful. Nonetheless, the evaluators made offhand remarks about the app's ease of use and their willingness to have a similar type of app installed on their personal devices.

Discussion

The majority of the usability issues identified by the evaluators were non-critical, suggesting that the permission manager app is a plausible solution for permission management on mobile devices.

Communicating Complex Information

Heuristic #2 (Match Between System and Real World) being the most frequently violated heuristic is indicative of the challenges associated with conveying the complexity of permission models in a simple and accurate manner. While using metaphors that are more closely aligned with users' mental models may have increased evaluators' understanding, applying them successfully is difficult. Specifically, the metaphors must communicate how apps, third parties, and elements of personal information relate to each other as well as the implications of these relationships; all on a device with limited screen real estate where users expect brief interactions or may be distracted.

Flexibility-Usability Tradeoff

Several comments from the evaluators indicate a desire for information beyond what is presented in the app. Simultaneously, the evaluators remarked that the app was easy to use. These statements present an interesting challenge in handling the need for additional information while maintaining simplicity; a concept best expressed by the Flexibility-Usability Tradeoff design principle [6]. When user needs are well-defined and predictable, simplicity is preferable, but when user needs are variable or still evolving, then flexibility is important. In our case, the overall goals are simple ("should this app have this permission?" and "what has this app done with my data?"), but they entail complex user preferences that may shift over time and the complex relationships and implications discussed above. One reasonable approach may be to provide a simple interface that allows users to drill-down as needed for extra details.

Large Data Sets

One issue that the heuristic evaluation did not surface is whether the app is able to handle large data sets. When there are dozens of entries, scrolling through the event list is quite manageable. However, if an app had continuous and frequent access to a category of personal information (e.g., a jogging app that needs continuous access to the user's location), the list could easily grow to hundreds or thousands of entries. A list of this size is limited in usefulness and thus, future work should attempt to offer summaries of app activity when the access history logs have a large number of entries. As well, to make examining the access history more manageable, alternative visualization methods and mechanisms for allowing users to filter data could be explored.

Limitations and Future Work

Future work should employ user testing to collect feedback from real users and to discover both the app's strengths and its weaknesses. Additionally,

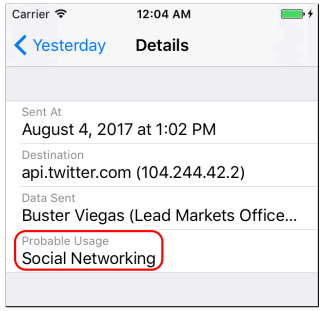

Figure 7: Classification of transmission on event details screen

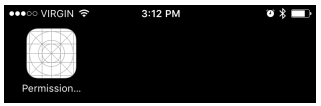

Figure 8: The app's default icon on the home screen.

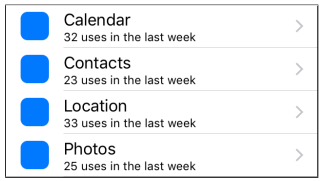

Figure 9: Temporary icons in the permission list.

since the data presented in the app was randomly generated, the evaluators did not have personal attachment to the information. This may have prevented usability issues from surfacing that would have otherwise been found if the data had originated through their ongoing use of the mobile device.

Conclusion

In this paper, we explored and developed a proof-of-concept permission manager for iOS that allows users to adjust the ability for apps to access personal information, to review recent instances where personal information was sent to a server, and to protect user privacy by having apps access fabricated data instead of real data. After evaluating the permission manager using a heuristic evaluation, it was discovered that while evaluators found the permission manager easy to use, the largest number of issues stemmed from unfamiliar concepts and terminology. It is recommended that future research include an investigation into metaphors that communicate the permission models more effectively, the refinement of the user interface to display more details and to better accommodate large volumes of data, and further user testing.

REFERENCES

1. Apple Inc. 2017. *iOS Security - White Paper*. Technical Report. https://www.apple.com/business/docs/iOS_Security_Guide.pdf
2. R. Balebako, J. Jung, W. Lu, L. Cranor, and C. Nguyen. 2013. "Little Brothers Watching You": Raising Awareness of Data Leaks on Smartphones. In *SOUPS*. ACM.
3. J. Boyles, A. Smith, and M. Madden. 2012. Privacy and data management on mobile devices. In *Pew Internet & American Life Project*.
4. A. Felt, S. Egelman, and D. Wagner. 2012a. I've got 99 problems, but vibration ain't one: a survey of smartphone users' concerns. In *SPSM*. ACM.
5. A. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner. 2012b. Android permissions: User attention, comprehension, and behavior. In *SOUPS*. ACM.
6. W. Lidwell, K. Holden, and J. Butler. 2003. *Universal Principles of Design*. Rockport Publishers, Beverly, MA, USA.
7. B. Liu, M.S. Andersen, F. Schaub, H. Almuhimedi, S. Zhang, N. Sadeh, Y. Agarwal, and A. Acquisti. 2016. Follow My Recommendations: A Personalized Privacy Assistant for Mobile App Permissions. In *SOUPS*. USENIX.
8. Jakob Nielsen. 1995a. 10 Heuristics for User Interface Design. (1995). https://www.nngroup.com/articles/ten-usability-heuristics/
9. Jakob Nielsen. 1995b. Severity Ratings for Usability Problems. (1995). https://www.nngroup.com/articles/ten-usability-heuristics/
10. J. Nielsen and R. Molich. 1990. Heuristic evaluation of user interfaces. In *CHI*. ACM.
11. J. Urban, C. Hoofnagle, and S. Li. 2012. Mobile phones and privacy. In *BCLT Research Paper Series*. SSRN.