Technical Disclosure Commons

Defensive Publications Series

June 25, 2019

Determining the Relevancy of Permissions Requested by an Application Software

Victor Carbune

Sandro Feuz

Recommended Citation

Carbune, Victor and Feuz, Sandro, "Determining the Relevancy of Permissions Requested by an Application Software", Technical Disclosure Commons, (June 25, 2019)
https://www.tdcommons.org/dpubs_series/2300

Determining the Relevancy of Permissions Requested by an Application Software

Abstract:

This publication describes methods that an application software market (application market) uses to determine a relevancy-permission score on an application software's (application) permission requests to access resources, features, user data (*e.g.*, calendar, photos, biometric data), and hardware (*e.g.*, microphones, cameras, global navigation satellite system (GNSS), accelerometers). Depending on the request, an operating system (OS) or the application market may prompt a user to approve the requests. The user, however, may be unsure whether they need to accept or deny a particular permission request made by an application. The user may be unclear whether the access is needed by the application most of the time, some of the time, not needed at all, or whether the request is abusive. To aid the user make informed decisions, the application market uses fuzzing techniques, scripted journeys, data analytics, and machine-learned models to evaluate the permissions requested by the application. After such evaluation, the application market generates a relevancy-permission score for each permission requested by an application, which can be easily-understood by the user of the application before they grant or deny such permission requests.

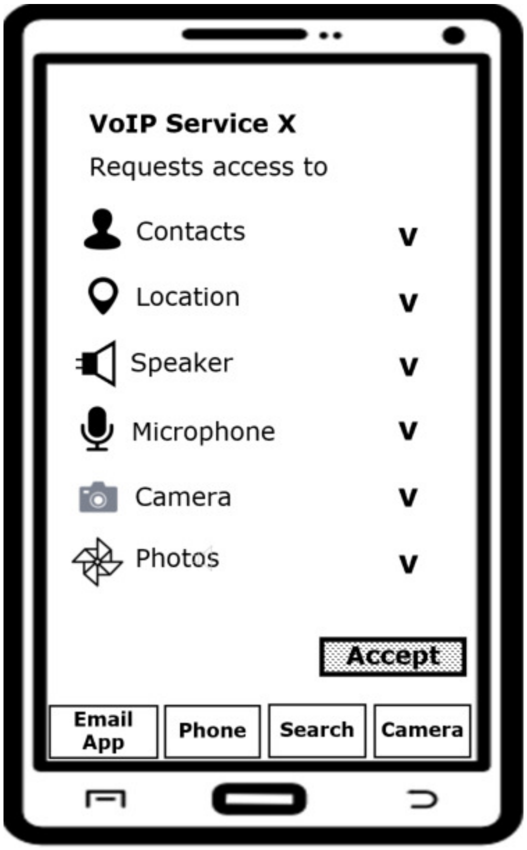
Keywords:

Application software, program, relevancy-permission score, trust score, user permission, application permission, application market, application store, installation, machine learning, ML, model training, fuzzing, fuzzy, artificial intelligence, AI.

Background:

Operating system (OS) developers, application software (application) developers, application software markets (application markets), and user equipment (UE) manufacturers increasingly offer users more product features. A widely-used UE, such as a smartphone, enables the user to call, participate in a video-conferencing session, text, email, bank, invest, shop, search for information, consume several types of media, participate in social networking, play games, navigate to a location, and use a plethora of other applications. In addition, UE manufacturers often integrate accelerometers, gyroscopes, magnetometers, barometers, global navigation satellite system (GNSS) technology (*e.g.*, global positioning satellite (GPS)), proximity sensors, touchscreen sensors, biometric sensors, heart-rate sensors, thermometers, humidity sensors, radar technology, cameras, microphones, and various other sensors in or on the UE, which enhance the user experience and may play a role on the functionality of various applications. Furthermore, the OS and the application market help manage user data, such as contacts data, short message service (SMS) data, notes, calendar data, user biometric data (*e.g.*, fingerprint data, voice-recognition data), credit card numbers, and other user specific data, which may also play a role on the functionality of various applications.

When the user installs an application, the application may request permissions to access resources, features, user data (*e.g.*, calendar), and certain hardware (*e.g.*, microphones, cameras, GNSS, accelerometers). Depending on the request, the OS may prompt the user to approve the requests. Figure 1 illustrates how the OS may prompt the user to accept or deny permission requests made by an application when the user installs an application downloaded from the application market.



VoIP Service X

Requests access to

Contacts V

Location V

Speaker V

Microphone V

Camera V

Photos V

Figure 1

In the example illustrated in Figure 1, Jane downloads a voice over internet protocol (VoIP) application called *VoIP Service X*. Jane's friends are increasingly using *VoIP Service X*, and Jane wants to start using it to stay in touch with her friends. Also, assume Jane is aware that *VoIP Service X* allows users to make calls, send messages, share photos, and communicate using a video-conferencing session. As Jane downloads *VoIP Service X*, the OS and the application market disclose that this application requests permission to access her contacts data, her location, the UE's speaker, the UE's microphone, the UE's camera, and her photos, as illustrated in Figure 1. Jane may understand why *VoIP Service X* requests permissions to access her contacts data, the speaker, the microphone, and the camera, but she may be unclear why *VoIP Service X* requests permission to access her location and her photos. Jane may be thinking, "Does *VoIP Service X* work without me granting access to my GNSS-tracked location?" or "Does *VoIP Service X* work without me granting access to my photos?"

The example in Figure 1 demonstrates how the user may be unsure whether they need to accept or deny a permission request by an application even in the case when the OS or the application market clearly disclosed the permissions requested by the application. The user may be unclear whether the permission is needed by the application most of the time, some of the time, not needed at all, or whether the request is abusive.

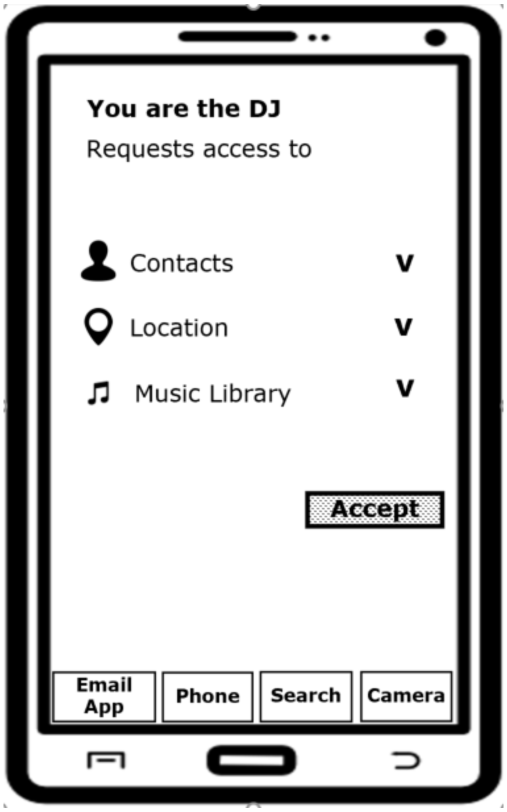
The example in Figure 1 helps show that it is desirable for the OS or the application market to let Jane know whether *VoIP Service X* works without the GNSS-tracked location. If Jane wants her friends to know her location when she uses *VoIP Service X*, she can choose to grant such permission request, but if Jane does not want her friends to know her location, she can choose to deny such permission request and still use *VoIP Service X*. The example in Figure 1 also shows that it is desirable for the OS and the application market to let Jane know whether she can use *VoIP Service X* without granting access to her photos. If Jane wants to use *VoIP Service X* to send photos to her friends, she can choose to grant such permission request, but if Jane does not want to use *VoIP Service X* to send photos to her friends, she can choose to deny such permission request and still be able to use *VoIP Service X* to call, send messages, and participate in video-conferencing sessions with her friends.

Therefore, it is desirable to have a technological solution that is transparent to all parties, such as the user, the application developer, the application market, and the OS, which can determine a relevancy-permission score for each permission requested by an application. In addition, unlike a "terms of use" language that an application may require the user to accept before they use the application, which can be long and cumbersome for the user to fully-comprehend, it is desirable that the relevancy-permission score can be easily-understood by the user of the application before they grant or deny such permission requests.

Description:

This publication describes methods that an application software market (application market) uses to determine a relevancy-permission score on an application software's (application) permission requests to access resources, features, user data (*e.g.*, calendar, photos, biometric data), and hardware (*e.g.*, microphones, cameras, global navigation satellite system (GNSS), accelerometers). When a user downloads the application, the application market may inform the user on the application's permission requests. Although the user may be informed on the permission requests they are granting or denying, they may be unclear on the reason of such requests and whether it is truly necessary or not.

Even in the cases when the application developers request permissions without the intent to abuse such permissions, but to offer the user a better user experience with the application, the user and the application market may be unclear on when and how the application uses the requested permissions. In addition, the application developers may also be unclear on how a user may use the application and how often the user uses certain features of the application because certain features may be hidden within the application. As described herein, a hidden application feature is a feature that the user may have to push buttons, click tabs, or open menus several times to activate the feature and that few users may ever utilize. To demonstrate the hidden application feature, consider the example illustrated in Figure 2.



You are the DJ
Requests access to

Contacts ✓
Location ✓
Music Library ✓

Figure 2

Assume Tom and some of his contacts use a music application called *You are the DJ* that allows the user to broadcast music to their contact list. For a modest fee that may be negotiated between the music industry and the application developer, the user may listen to a playlist being broadcasted by their friend. The example *You are the DJ* application requests permission to access the contacts data, the location, and the music library, as in Figure 2. Tom's playlists are popular among his friends. Tom, however, rarely broadcasts his location because that information is of little interest to his contacts. Nevertheless, one day Tom wants to broadcast his location along with his playlist, and he clicks on three tabs to enable the location-broadcasting feature. Tom is visiting the tomb of one of his favorite artists, Jim Morrison, at the *Père Lachaise Cemetery* in Paris. As Tom is standing in front of Jim Morrison's tomb, he starts broadcasting music by *The Doors* (a band associated with Jim Morrison) along with Tom's current location.

One way to determine the relevancy-permission score of the application is to use data analytics. As the application requests certain permissions, the application market or the OS may inform the user on the relevancy of the requested permissions. Referring to the example in Figure 2, the application market may inform Tom that 0.05% of users enable the GNSS-tracked location application permission for the *You are the DJ* application. Given that most users choose not to broadcast their location while using the example *You are the DJ* application, Tom may choose to deny access to such permission request.

In addition, applications may have numerous features. Referring to Jane's example in Figure 1, *VoIP Service X* asks permission to access Jane's contacts data, her location, the UE's speaker, the UE's microphone, the UE's camera, and her photos. The application asks this access because it uses them to support the application features, but the application may support most features used by users by asking for only a portion of the permissions. Jane may use *VoIP Service X* to only make and receive calls. In that case, Jane may only grant permission to her contacts data, the UE's speaker, and the UE's microphone, and she may choose to deny access to her location, the UE's camera, and her photos.

The examples in Figure 1 and in Figure 2 help illustrate the desire for the application market to offer such information to users regarding the application. This data analytics, however, may not be available or visible to the application market and the user. A mechanism that can automatically determine what permissions are needed for the user to use most of the features of the application benefits the user and helps protect the user's privacy. If an application asks for "p" number of permissions, the application market may inform the user that granting "n" out of "p" (where "n" is less than "p") number of permissions enables the user to utilize 99%, 98%, 95%, and so forth of the application's features. Then the user may make an informed decision before granting or denying the requested permissions.

To create data analytics, initially the application market may utilize a fuzzing technique that simulates possible user journeys using the application and determines the context in which the application uses the access granted by the permissions. The application may run in a simulation sandbox. As described herein, a simulation sandbox refers to an isolated computing environment that testers, software developers, and engineers use to mimic the characteristics of a production environment, such as an OS.

Several types of fuzzing techniques may be utilized to determine how the application uses the access granted by the permissions. For example, one type of fuzzing may randomly push buttons, click tabs, or open menus that the application shows on the UE's screen. This "brute-force" fuzzing technique may count how many times the application uses the access granted by the permissions (*e.g.*, 5 out of 200 clicks (2.5%)). This allows the fuzzing technique to start creating data analytics on the use of the access granted.

The application market may also use a scripted approach based on how users utilize the application. The application market anonymizes the users' identities and learns the users' journeys with the application. Based on the users' journeys, engineers of the application market can build scripted journeys and use these scripts in the simulation sandbox.

In addition to compiling data on the number of times the application triggers the use of the granted access, the fuzzing technique and the scripted journey can determine at what journey flow the application triggers the use of the granted access and how hidden are the application features that trigger the use of granted access. Referring again to the example of Tom in Figure 2, the application market can use this data to inform Tom whether he truly needs to grant certain permissions in order for him to use certain hidden features of the example *You are the DJ* application.

The application market can also generate data analytics based on user demographics, such as a user's country, state, city, age group, gender, and so forth. Through various clustering methods, the fuzzy model can learn how different sets of users utilize the application. For example, the elderly may generally not use the camera when using *Application Software A*, the European Union citizens may generally not use location when using *Application Software B*, the New Yorkers may often use location when using *Application Software C*, the teenagers may generally use the camera when using *Application Software D*, and so forth.

The proliferation of applications translates to new applications added to the application market on a daily basis. The application market may lack analytics data, fuzzing techniques, or scripted journeys for newly-developed applications. In such cases, the application market may use a machine-learned model for navigating the application. Such a model takes as input the current application screen and issues the coordinates of the next tap. The model can be trained on anonymized data from popular applications used by the user. The machine-learned model may consist of neural-network layers, including pre-trained layers, such as convolutional neural networks for processing screenshots. The machine-learned model may also consist of recurrent neural network layers for handling arbitrary long sequences of input data. The inputs to the machine-learned model are user journeys that trigger requests for access in existing applications. The output of the machine-learned model is a predicted relevancy-permission score for newly-developed applications.

To iteratively refine the fuzzy techniques, scripted journeys, data analytics, and the machine-learned model, the application market can allow the user to manually review permissions requested. The user may correct a low relevancy-permission score using user annotated pipelines. Information regarding these user manual corrections can be used to retrain the machine-learned models, update the scripted journeys, or improve data analytics.

Additionally, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs, or features described herein may enable collection of user information (e.g., information about a user's social network, social actions or activities, profession, a user's preferences, or a user's current location), and if and when the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user, or a user's geographic location may be generalized where location information is obtained (such as to a city, ZIP code, or state level), so that a particular location of a user cannot be determined. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user. The user may also select to ignore an application market's relevancy-permission score.

In conclusion, the fuzzing techniques, the scripted journeys, data analytics, and the machine-learned model enable an application market to flag applications that request permissions that are not needed for the application's functionality and help protect the user's privacy.

References:

[1] Knox, Matt. Predictive Permissioning for Mobile Devices. US Pub. 2018/0288616, filed March 28, 2017, and published October 4, 2018.