Technical Disclosure Commons

Defensive Publications Series

October 01, 2018

Intelligent and Dynamic Permission Model for User Permissions

Neil Dhillon

Tanmay Wadhwa

Young Lin

Intelligent and dynamic permission model for user permissions

ABSTRACT

A permission model is described that provides a set of user modifiable default application permissions for user devices. A user interface enables the user to easily observe services and permissions being utilized by various apps and to update permission settings. Upon request, the user can grant permissions not granted by default at app initialization. With user consent, user actions to grant or deny access are logged to provide telemetry data regarding the user's app usage patterns. App telemetry data from multiple user devices of consenting users are aggregated and combined with data obtained from app sources. The collected data are provided to a classification model that returns a parameterized vector representation of the confidence of available permissions. App permissions are granted based on the confidence value meeting a predetermined threshold.

KEYWORDS

- App telemetry
- Permission model
- Permission granularity
- User experience
- Operating system
- Permission

BACKGROUND

A conundrum in the world of cybersecurity is that while modern operating systems use layer-upon-layer of security features to restrict access to a root or administrator account of an operating system, much of the data that is of interest to malicious actors can be retrieved by

using an unprivileged user account. In the context of mobile device operating systems, this is even more apparent. An attacker can bypass security layers on a user device to steal data by simply having the user provide permissions in an app. In particular, permission models for mobile devices rely on users to be aware, responsible, and knowledgeable to make informed decisions on how to best protect their own data and privacy. This may not always be the case.

An app that has been granted overly broad permissions can collect user data such as location information, recent messages, pictures/videos, audio data, files stored on the device, etc. without circumventing security features and without user knowledge of such collection of data. While many platforms provide granular controls to toggle permissions for different apps, these advanced settings are seldom used. In general, app permissions tend to be overly broad in scope, long-lived, and are not clearly evident when in-use.

Furthermore, the scope of permissions applied is also overly broad. Permissions are granted on a per-app basis, where all components of an app (e.g., activities and services) have access to the permitted user data, regardless of the specific function of the component. Permissions granted to an app are usually permanent unless explicitly revoked by the user. Once permission is granted to access a particular service, future versions of that app are also automatically granted the same permission. Permission models that strike a balance between usability and security can enhance user experience.

DESCRIPTION

This disclosure describes an intelligent operating system/platform to manage application software (app) permissions to access services, e.g., camera, LED, microphone, etc. A user interface enables the user to easily observe services and permissions being utilized by various apps, as well as enabling the user to configure permissions. Per techniques of this disclosure,

default permissions for apps installed on a user device are provided by a server to the user device operating system upon check-in (initialization).

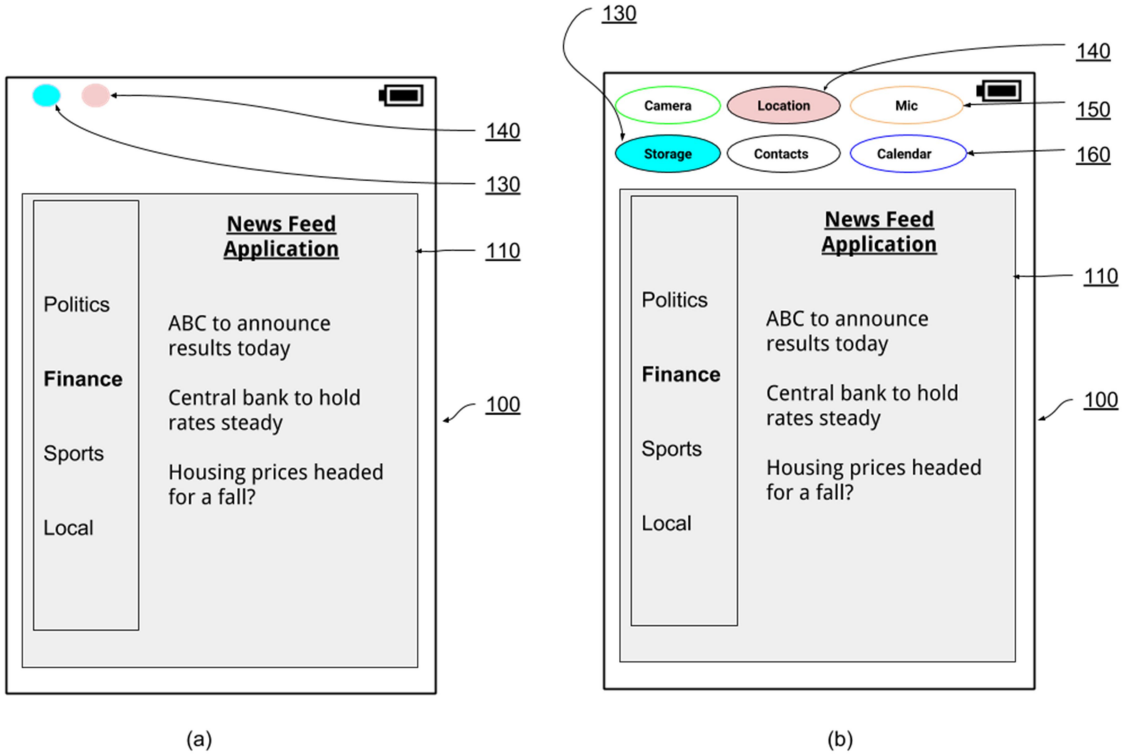

Fig. 1: Visualization and configuration of app permissions on a user device

Fig. 1 illustrates an example user interface that displays specific permissions and services utilized by an app (110) on a user device (100). Fig. 1(a) illustrates colored dots (130, 140) indicative of the utilized permissions that are displayed on the device's status menu. The user interface also enables the user to verify, e.g., by swiping down on the status menu, permissions used by different installed apps. Fig. 1(b) illustrates user verification of permissions by tapping on colored tiles (130-160) that are color coded. The color coded tiles correspond to different services (In the illustrated example, storage (130) and location (140) utilize differently colored tiles) and are indicative of services that are permitted to be accessed by different installed apps on the user device. In the illustrated example, news feed application

(110) is permitted access to the storage and location services, and is not granted access to microphone (150) and calendar (160).

Permission to access services is provided only to apps that are running in the foreground or in the background. The user is enabled to observe permissions being invoked by an app at any time. When a specific activity or service that is associated with an app is granted a permission but is not currently using the permission, the colored dot(s) are not displayed on the status menu.

A permission requested by an activity or service associated with an app that is not granted by default at initialization can be granted by the user, e.g., by swiping down on the status menu and tapping on the tile corresponding to the permission intended to be granted. With specific user permission, user actions to grant and deny access for apps to services are logged by the operating system to provide telemetry data regarding the user's app usage patterns.

App telemetry data from multiple user devices from users that provide permission for collection of such data are aggregated (without user-specific identifiers) to provide information about apps installed on user devices, permissions requested by the apps, permissions granted to the apps, and app usage patterns. The telemetry data are aggregated and combined with data obtained from app sources (such as app developers or online marketplaces that sell apps) that indicate app developer reputation, number/locations of app installs, and app type/purpose.

Additionally, dynamic and static analysis of signals from malware scanners indicative of the safety of the app (for example, whether dynamic class loading is performed by the app), complexity, domain information of backend components, and the types of data sent to/from the backend components are also obtained.

The obtained data and signals are provided to a machine learning classification model for inference. The model returns a parameterized vector representation of the confidence of each available permission for each activity and service within a given app. As an illustrative example, a vector representation output of [0.42 0.87 0.35] can represent the confidence values for the permissions “phone,” “location,” and “camera,” respectively for a “newsfeed” activity, indicative of a lower confidence level for a “newsfeed” activity to obtain permission to a phone and camera service than for a location service on the user device. Permissions for services are disabled by default when the confidence values for the services are below a predetermined threshold (for example, a value of 0.8). Permissions in the output vector that are not explicitly requested by an app receive a corresponding confidence value of 0.

The model inference being performed on the server-side as opposed to client-side (i.e. on the user device) allows for data to be crowdsourced from multiple sources. The classification model can thus be continually updated and trained on the latest datasets such that changes in usage patterns and newly published and updated apps are considered.

A default permission output vector for each component of an app is computed at the server and utilized to initialize user device permissions. Such vector is sent to user devices, e.g., using a push service, and updates a local cache on the user device. Default permissions granted to an application are updated by the device operating system based on data received from the server. Users are provided with options to override these settings, e.g., to explicitly enable/disable a certain permission for a certain app or even disable use of the described techniques.

When an app is launched, the specific activity that is running in the foreground is granted when the permissions meet a predetermined threshold confidence score. Similarly,

background services are only granted permissions by default if the confidence score meets a predetermined threshold. When an app is not in a state of execution (not running), all of its permissions are revoked by default. Techniques described in this disclosure can be utilized to apply the security principle of "deny by default" to user device app permissions.

The behavior of specific components of an app is considered when permission to services are granted. For example, that a flashlight app needs access to an LED of the user device for its function, but not the device camera, microphone, local file storage, or location sensor. In this instance, permission can be granted to only the LED while intelligently blocking all other permissions requested by the app and providing a notification to the user. The user has the option to override or disable the permissions suggested by the model.

As app usage patterns and other signals change, the permissions granted to an app become ephemeral and dynamic. A previous version of an app may have needed the permission to write to a device's SD card but permission may be denied to future versions of the app, even if the app developer requests this (unnecessary) permission granted previously.

Users commonly choose the default option when presented with a complex set of options. By defaulting to a permission model that applies an algorithmic and intelligent solution to the grant of app permissions on a user's behalf, cases of abuse of permissions are mitigated, thereby improving the security of the operating system and user device. While the foregoing description refers to server implementations, the permissions may be generated at the user device instead of at the server.

Further to the descriptions above, 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 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.

CONCLUSION

A permission model is described that provides a set of user modifiable default app permissions for user devices. A user interface enables the user to easily observe services and permissions being utilized by various apps, as well as enable the user to configure permissions. Permissions requested by an activity or service associated with an app that are not granted by default at initialization can be granted by the user. With specific user permission, user actions to grant and deny access are logged to provide telemetry data regarding the user's app usage patterns. App telemetry data from multiple user devices are aggregated and combined with data obtained from app sources. A classification model returns a parameterized vector representation of the confidence of available permissions based on the data. Permissions are automatically granted based on the confidence value meeting a predetermined threshold and are user-configurable.