DEMO: Starving Permission-Hungry Android Apps Using SecuRank

Vincent F. Taylor
Department of Computer Science
University of Oxford
Oxford, United Kingdom
vincent.taylor@cs.ox.ac.uk

Ivan Martinovic
Department of Computer Science
University of Oxford
Oxford, United Kingdom
ivan.martinovic@cs.ox.ac.uk

ABSTRACT

We demonstrate SecuRank, a tool that can be employed by Android smartphone users to replace their currently installed apps with functionally-similar ones that require less sensitive access to their device. SecuRank works by using text mining on the app store description of apps to perform groupings by functionality. Once groups of functionally-similar apps are found, SecuRank uses contextual permission usage within groups to identify those apps that are less permission-hungry. Our demonstration will showcase both the Android app version of SecuRank and the web-based version. Participants will see the effectiveness of SecuRank as a tool for finding and replacing apps with less permission-hungry alternatives.

Keywords

Android; smartphone app; least privilege; permission

1. INTRODUCTION

Competition among app developers has led to app stores being inundated with groups of functionally-similar general-purpose apps, such as *flashlights* and *alarm clocks*. Within groups of functionally-similar apps, however, the use of dangerous permissions often varies widely. It is known from search engine optimisation that users tend to disproportionately favour the highest ranking search results when performing searches [7]. Thus, users may inadvertently download and install a permission-hungry app when there was a competing app that provided the required functionality, but without requiring as much sensitive access to their device.

We illustrate the problem of widely varying dangerous permission usage by showing the Top 8 search results for the search query "alarm clock" in Fig. 1. The most permission-hungry app uses six dangerous permissions while the least permission-hungry app uses only two dangerous permissions. This is in spite of the fact that all these apps provide the same basic functionality. Moreover, these apps all have millions of downloads and are highly rated, further complicating the task of the average user in choosing a suitable app.

SecuRank is designed to suggest replacements for *general-purpose* apps only. By general-purpose apps, we mean those that provide generic functionality, with more than one apps competing to provide that functionality. Non-general-purpose apps such as *Facebook* and *Bank of America* are not supported by SecuRank because usually there is only one app that matches what the user is looking for. SecuRank only focuses on the so-called *dangerous permissions* present on Android [2], since these are the ones that guard sensitive user data.

1.1 Failure of Run-time Permissions

Android 6.0 (API level 23, code-name *Marshmallow*) aims to return power to the user by introducing run-time permissions [1]. Now, users no longer grant an app's permissions in their entirety at install-time. Instead, they must now selectively accept or reject permissions at run-time. This is a welcome improvement, but as Eling et al. [4] discover, 40.4% of users still accept fine-grained, intrusive and unnecessary run-time permission requests. This supports prior work which shows that users suffer from conditioning and/or a lack of understanding when presented with security warnings [5]. Additionally, app developers may nullify the switch-over to run-time permissions altogether by targeting their apps to API level 22 or lower.

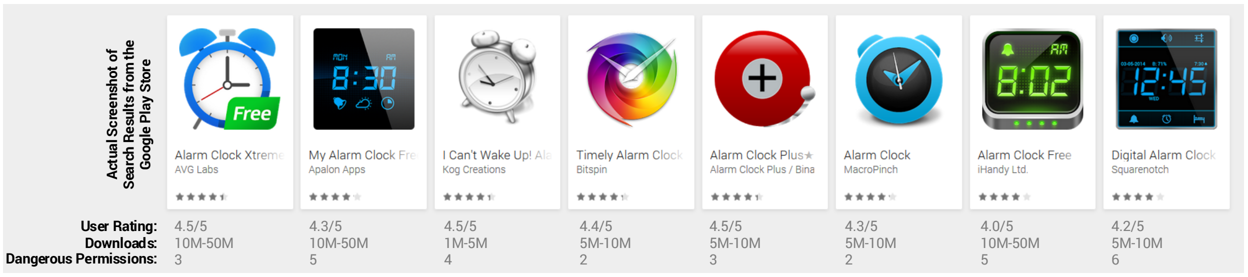
For this reason, permission-hungry apps continue to be a problem in the face of the introduction of run-time permissions. We are motivated to provide a tool that can identify and recommend apps that use only as many dangerous permissions as they need, i.e., apps that follow the principle of least privilege [8]. On Android, using apps that follow the principle of least privilege limits opportunities for adversaries to perform privilege escalation attacks, app collusion attacks, and confused deputy attacks [3].

1.2 Demonstration

To this end, we will demonstrate SecuRank[1], a tool that can be used to audit the list of apps installed on a smartphone to determine whether any of them can be replaced with a functionally-similar alternative that requires less sensitive access to the device. Participants will be introduced to the SecuRank website and allowed the opportunity to check if apps of interest to them have less permission-hungry alternatives. Participants will also be able to use the SecuRank

[1]SecuRank is freely available to use at https://securank.me/ and as a free Android app.

CCS'16 October 24-28, 2016, Vienna, Austria
© 2016 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4139-4/16/10.
DOI: http://dx.doi.org/10.1145/2976749.2989032



User Rating:	4.5/5	4.3/5	4.5/5	4.4/5	4.5/5	4.3/5	4.0/5	4.2/5
Downloads:	10M-50M	10M-50M	1M-5M	5M-10M	5M-10M	5M-10M	10M-50M	5M-10M
Dangerous Permissions:	3	5	4	2	3	2	5	6

Figure 1: Snapshot of the Top 8 Google Play Store results for the search query "alarm clock". All apps have very high ratings and provide similar functionality, but the most permission-hungry app uses three times as many dangerous permissions as the least permission-hungry app.

app on demonstration Android devices to examine its user interface and the results provided. Motivated participants may install and run SecuRank on their own Android devices during the demonstration session. This is expected to stimulate further discussions among participants and demonstrators by showcasing real-world results.

2. SECURANK IMPLEMENTATION

SecuRank uses the concept of contextual permission usage to identify permission-hungry apps from groups of functionally-similar apps.

2.1 Identifying Functionally-Similar Apps

App developers are interested in detailing the features of their apps to attract downloads as well as provide keywords for the app store search ranking algorithm. Thus, a majority of apps have very detailed descriptions outlining their features. The idea is that functionally-similar apps will have similar app store descriptions and this can be calculated using a similarity metric. SecuRank identifies functionally-similar apps by using the *cosine similarity* measure to compare app descriptions across the Google Play Store and clustering apps whose description fall above the similarity threshold.

2.2 Contextual Permission Usage

Once groups of functionally-similar apps are identified, dangerous permission usage by apps within each group is analysed to identify those apps that are permission-hungry, i.e., not following the principle of least privilege. To identify misuse of dangerous permissions, we penalise apps that use dangerous permissions that are not common among apps of that particular functionality. The idea is that if all the apps provide similar functionality, then dangerous permission usage outside of the norm might be a useful indicator of an app being over-privileged. This approach is similar to the one used by CHABADA [6]. However, instead of merely identifying apps that are outliers, we provide a tool that leverages the information to recommend functionally-similar apps that are preferable. Additionally, we study the entire Google Play Store to understand the prevalence of outliers, and the extent to which SecuRank can help users to identify and avoid them.

We define the Individual Permission Prevalence (IPP) of a dangerous permission as the fraction of apps in a group of functionally-similar apps using that dangerous permission. App Overall Permission Prevalence (AOPP) is defined as the mean of the IPPs of those dangerous permissions used by an app. Algorithm 1 shows how IPP and AOPP are calculated.

Algorithm 1: Calculate IPP and AOPP for a list of apps

```
Input: List of apps $\beta = \beta_1, \ldots, \beta_n$
Output: IPP, AOPP per app
$permList \leftarrow [\,]$
foreach $app$ in $\beta$ do
    $permList \leftarrow permList + getPermissions(app)$
$IPP \leftarrow \emptyset$
foreach $perm$ in $GetUniqueItems(permList)$ do
    $IPP[perm] \leftarrow permList.count(perm) \div len(\beta)$
$AOPP \leftarrow \emptyset$
foreach $app$ in $\beta$ do
    $temp \leftarrow [\,]$
    foreach $perm$ in $getPermissions(app)$ do
        $temp \leftarrow temp + IPP[perm]$
    $AOPP[app] \leftarrow mean(temp)$
return $AOPP, IPP$
```

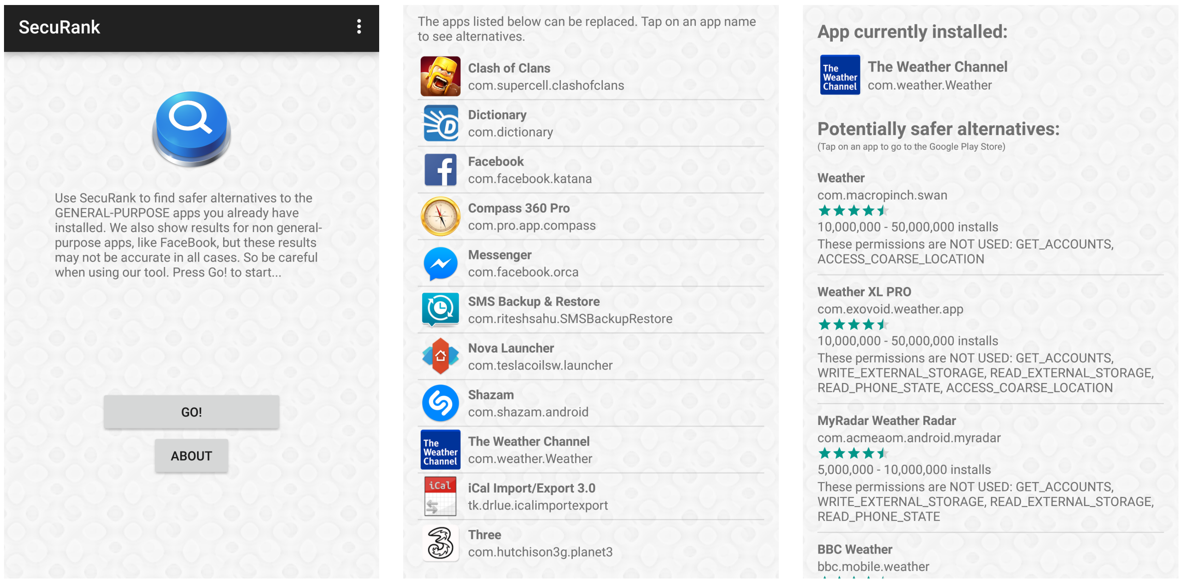
As an example, consider four apps: $app_1 = \{A, B, C\}$, $app_2 = \{A, B\}$, $app_3 = \{A, C\}$, $app_4 = \{A\}$, where A, B and C are dangerous permissions. IPP and AOPP are calculated as follows:

1. Make list of all dangerous permissions, [A, B, C, A, B, A, C, A].
2. Count occurrences of each dangerous permission in list, i.e., $p_A = 4$, $p_B = 2$, $p_C = 2$.
3. IPP is the fraction of occurrences of a dangerous permission to the number of apps, i.e., $IPP_A = \frac{4}{4} = 1.0$, $IPP_B = \frac{2}{4} = 0.5$, $IPP_C = \frac{2}{4} = 0.5$.
4. AOPP is the mean of the IPPs for those dangerous permissions used by an app, i.e., $AOPP_1 = 0.66$, $AOPP_2 = 0.75$, $AOPP_3 = 0.75$, $AOPP_4 = 1.0$.

An app is considered to be less permission-hungry if it has a higher AOPP than the app it is intended to replace.

2.3 Screenshots of SecuRank

Screenshots of the SecuRank Android app's main activities are shown in Fig. 2. The user initiates the scan from the Home Screen. The list of apps on the device is sent to the server for processing. Apps that have less permission-hungry alternatives are presented on the Results screen, and these alternatives can be seen on the Suggestions screen.


SecuRank

Use SecuRank to find safer alternatives to the GENERAL-PURPOSE apps you already have installed. We also show results for non general-purpose apps, like FaceBook, but these results may not be accurate in all cases. So be careful when using our tool. Press Go! to start...

GO!

ABOUT

The apps listed below can be replaced. Tap on an app name to see alternatives.

Clash of Clans
com.supercell.clashofclans

Dictionary
com.dictionary

Facebook
com.facebook.katana

Compass 360 Pro
com.pro.app.compass

Messenger
com.facebook.orca

SMS Backup & Restore
com.riteshsahu.SMSBackupRestore

Nova Launcher
com.teslacoilsw.launcher

Shazam
com.shazam.android

The Weather Channel
com.weather.Weather

iCal Import/Export 3.0
tk.drlue.icalimportexport

Three
com.hutchison3g.planet3

App currently installed:

The Weather Channel
com.weather.Weather

Potentially safer alternatives:

(Tap on an app to go to the Google Play Store)

Weather
com.macropinch.swan
10,000,000 - 50,000,000 installs
These permissions are NOT USED: GET_ACCOUNTS, ACCESS_COARSE_LOCATION

Weather XL PRO
com.exovoid.weather.app
10,000,000 - 50,000,000 installs
These permissions are NOT USED: GET_ACCOUNTS, WRITE_EXTERNAL_STORAGE, READ_EXTERNAL_STORAGE, READ_PHONE_STATE, ACCESS_COARSE_LOCATION

MyRadar Weather Radar
com.acmeaom.android.myradar
5,000,000 - 10,000,000 installs
These permissions are NOT USED: GET_ACCOUNTS, WRITE_EXTERNAL_STORAGE, READ_EXTERNAL_STORAGE, READ_PHONE_STATE

BBC Weather
bbc.mobile.weather

(a) Home screen **(b)** Results screen **(c)** Suggestions screen

Figure 2: Screenshots of the main activities in the SecuRank Android app.

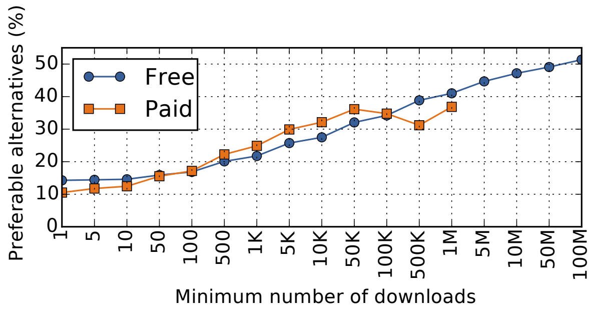


Figure 3: Percentage of apps having preferable (i.e. less permission-hungry) alternatives.

3. RESULTS

We ran the SecuRank tool on the entire Google Play Store to understand the extent to which apps could be replaced with less permission-hungry, i.e., preferable alternatives. Fig. 3 shows the likelihood of apps having preferable alternatives, broken down by the popularity and cost (free or paid) of the app. For both free and paid apps, more popular apps were more likely to have a less permission-hungry alternative.

Across our results, we looked at the quality of the replacement apps that were being suggested by SecuRank. We used the average user rating of an app as a proxy for its quality. SecuRank was able to find preferable alternatives for approximately 210,000 apps in the Google Play Store. Overall, 53.2% of the alternatives had the same (4.9%) or higher (48.3%) rating than the original app. Of the suggestions that had a lower rating, one-third were rated within 0.25 stars (out of five stars) of the original. This suggests that the apps being recommended by SecuRank would be well-received by users, since they too are rated as highly as the apps they replace.

4. REFERENCES

[1] Requesting Permissions at Run Time. http://developer.android.com/training/permissions/requesting.html. Accessed July 2016.

[2] System Permissions. http://developer.android.com/guide/topics/security/permissions.html. Accessed July 2016.

[3] S. Bugiel, L. Davi, A. Dmitrienko, T. Fischer, A.-R. Sadeghi, and B. Shastry. Towards Taming Privilege-Escalation Attacks on Android. In *NDSS*, 2012.

[4] N. Eling, S. Rasthofer, M. Kolhagen, E. Bodden, and P. Buxmann. Investigating Users' Reaction to Fine-Grained Data Requests: A Market Experiment. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 3666–3675, Jan 2016.

[5] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner. Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the 8th Symposium on Usable Privacy and Security (SOUPS 2012)*, SOUPS '12, pages 3:1–3:14, New York, NY, USA, 2012. ACM.

[6] A. Gorla, I. Tavecchia, F. Gross, and A. Zeller. Checking App Behavior Against App Descriptions. In *Proceedings of the 36th International Conference on Software Engineering*, ICSE 2014, pages 1025–1035, New York, NY, USA, 2014. ACM.

[7] J. Lee. No. 1 Position in Google Gets 33% of Search Traffic [Study]. http://searchenginewatch.com/sew/study/2276184/no-1-position-in-google-gets-33-of-search-traffic-study, June 2013. Accessed July 2016.

[8] J. H. Saltzer. Protection and the control of information sharing in multics. *Commun. ACM*, 17(7):388–402, July 1974.