Jan Jürjens, Kurt Schneider (Hrsg.): Software Engineering 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 113

App Mining

Vitalii Avdiienko[1] Konstantin Kuznetsov[2] Alessandra Gorla[3] Andreas Zeller[4]
Steven Arzt[5] Siegfried Rasthofer[6] Eric Bodden[7]

Abstract: A fundamental question of security analysis is: When is a behavior normal, and when is it not? We present techniques that extract *behavior patterns* from thousands of apps—patters that represent *normal behavior,* such as "A travel app normally does not access stored text messages". Combining data flow analysis with app descriptions and GUI data from both apps and their stores allows for massive machine learning, which then also allows to detect yet unknown malware by classifying it as abnormal.

Extended Abstract

Most existing malware detectors work *retrospectively,* checking an unknown app against features and patterns known to be malicious. Such patterns can either be given explicitly ("Text messages must only be sent after the user has declared her consent"), or induced implicitly from samples of known malware ("This app contains code known to be part of the TDSS trojan."). If a novel app is sufficiently different from known malware, though, this retrospective detection can fail.

In our work, we thus conversely investigate the idea that, given access to a sufficiently large set of "benign" apps, one might be able to detect novel malware not by its similarity with respect to existing malware, but rather through its *dissimilarity with respect to those benign applications.* As a measure for establishing similarity or dissimilarity with respect to the norm, we explore the *usage of sensitive data* in an app.

In our CHABADA [Go14] work, we check *implemented* app behavior against *advertised* app behavior. As a proxy for the advertised behavior of an app, we use its natural language description from the Google Play Store. Asa proxy for its implemented behavior, we use the set of Android application programming interfaces (APIs) that are used from within the app binary. The key idea is to associate descriptions and API usage to detect anomalies: "This 'weather' application accesses the messaging API, which is unusual for this category" (See Fig. 1). Applied on a set of 22,500+ Android applications, our CHABADA prototype identified several anomalies; additionally, it flagged 56% of novel malware as such, without requiring any known malware patterns.

[1] Saarland University, Saarbrücken, Germany, avdiienko@cs.uni-saarland.de
[2] Saarland University, Saarbrücken, Germany, kuzntesov@cs.uni-saarland.de
[3] IMDEA Software Institute, Madrid, Spain, alessandra.gorla@imdea.org
[4] Saarland University, Saarbrücken, Germany, zeller@cs.uni-saarland.de
[5] TU Darmstadt, Darmstadt, Germany, steven.arzt@cased.de
[6] TU Darmstadt, Darmstadt, Germany, siegfried.rasthofer@cased.de
[7] Paderborn University, Padeborn, Germany, eric.bodden@uni-paderborn.de

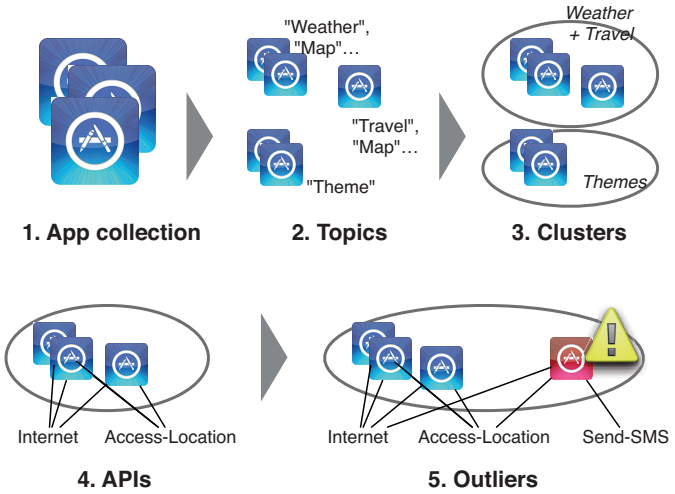

Abb. 1: How CHABADA detects apps with unadvertised behavior. Starting from a collection of "good" apps (1), we identify their description topics (2) to form clusters of related apps (3). For each cluster, we identify APIs used (4), and then identify outliers with unusual APIs for that cluster (5).

In our MUDFLOW work [Av15], we apply static taint analysis on the Android apps from the Google Play Store to determine, for every sensitive data source, the sensitive APIs to which this data flows. By applying MUDFLOW on the 2,866 most popular apps collected from the Google Play Store, we can extract typical usage of sensitive sources across these apps. In our experiment on a set of 10,552 malicious apps leaking sensitive data, MUDFLOW recognized 86.4% of the malware as such, with a false positive rate of 11.7%, which is remarkable given that MUDFLOW is not trained on malware samples. Companies like Google and Microsoft now use techniques similar to CHABADA and MUDFLOW to identify malware.

Our current work focuses on anomalies in *user interfaces*—Does this button do what it claims to do? Is this user interface consistent and complete? Again, we can apply machine learning on thousands of instances to learn what is normal—and what is abnormal. We see a great future in app mining, and we call for experts in program analysis, machine learning, natural language analysis, and human-computer interaction to join their forces to exploit these new data sources and opportunities.

References

[Av15] Avdiienko, Vitalii; Kuznetsov, Konstantin; Gorla, Alessandra; Zeller, Andreas; Arzt, Steven; Rasthofer, Siegfried; Bodden, Eric: Mining Apps for Abnormal Usage of Sensitive Data. In: Proceedings of the 37th International Conference on Software Engineering. ICSE 2015, 2015.

[Go14] Gorla, Alessandra; Tavecchia, Ilaria; Gross, Florian; Zeller, Andreas: Checking App Behavior Against App Descriptions. In: Proceedings of the 36th International Conference on Software Engineering. ICSE 2014, ACM, New York, NY, USA, pp. 1025–1035, June 2014.