Android App Collusion

Ervin Oro
ervin.oro@aalto.fi

Tutor: Jorden Whitefield

Abstract

Android supports various communication methods between apps, and colluding apps is an emerging threat to Android based devices. An app collusion is where two or more apps collude in some manor to perform a malicious action that an app cannot perform independently. State-of-the-art malware detection systems analyze apps in isolation, and therefore fail to detect joint malicious actions between colluding two or more apps.

KEYWORDS: keywords, separated by commas

add actual abstract and keywords

1 Introduction

The goal of this seminar topic is to:

1. Survey current Android malware detection literature, techniques, and tools.

2. Survey current Android app collusion literature.

2 Description and definition of app collusion

The Oxford English Dictionary defines collusion as "Secret agreement or understanding for purposes of trickery or fraud; underhand scheming or working with another; deceit, fraud, trickery." [1]. In this context, [2] defines that app collusion is when, in performing a threat, several apps are working together. From this, three aspects of collusion can be derived:

1. Collusion is secret. Conversely, apps working together in collaboration is common and encouraged practice when such collaboration is well documented [!] .

Citation needed 1:

2. Collusion is when all colluding parties are in agreement. A distinctly different but related concept is the “confused deputy” attack, where one app mistakenly exposes itself to other installed apps [!] .

Citation needed 2: Hardy, N.: The confused deputy:(or why capabilities might have been invented. ACM-SIGOPS Operating Systems Review 22(4), 36–38 (1988)

3. Collusion is with malicious intent. In the Android context, it would be the intention to violate one of its security goals, which are to protect application data, user data, and system resources (including the network) [3]. The Android Open Source Project also lists providing "application isolation from the system, other applications, and from the user" as a separate goal, but in this context, collusion would be when apps work together to break isolation with some third component, as for two willing parties there exist several legitimate communication channels [4].

many things affect, including non technical

difficult to distinguish

3 Methods for colluding

3.1 Overt channels

3.2 Covert channels

4 Examples of Android app collusion

A very widely cited example of collusion is an imaginary situation as follows. One app has access sensitive information, but no access to internet. Another app, on the other hand, has access to internet, but no access to any sensitive information. Many authors [!] argue that in this case, one app could pass information to the other one, which could in turn then exfiltrate the information. Some authors [!] have extended this concept to also cover cases where data is passed to multiple apps before being finally exfiltrated. All current research focuses on detecting such situations.

There is one known case of Android app collusion in the wild [5]. Interestingly enough, even though this example is also widely referred to [!], it does not follow the pattern described above.

Citation needed 3:

Citation needed 4:

Citation needed 5: list some references

short description of MoPlus SDK

5 Existing methods for detecting collusions

Bibliography

[1] "collusion, n." in *OED Online*. Oxford University Press, Dec. 2018.

[2] I. M. Asăvoae *et al.*, "Detecting malicious collusion between mobile software applications: the Android™ case," in *Data Analytics and Decision Support for Cybersecurity*. Springer International Publishing, 2017, pp. 55–97.

[3] Android Open Source Project, "Security," 2019.

[4] ——, "Application security," 2019.

[5] J. Blasco *et al.*, "Wild Android collusions," in *VB2016*, Oct. 2016.

[6] F. I. Abro *et al.*, "Android application collusion demystified," in *Future Network Systems and Security*, R. Doss, S. Piramuthu, and W. Zhou, Eds. Cham: Springer International Publishing, 2017, pp. 176–187.

[7] I. M. Asăvoae, H. N. Nguyen, and M. Roggenbach, "Software model checking for mobile security – collusion detection in $\mathbb{K}$," in *Model Checking Software*, M. d. M. Gallardo and P. Merino, Eds. Cham: Springer International Publishing, 2018, pp. 3–25.

[8] H. Chen *et al.*, "Malware collusion attack against SVM: issues and countermeasures," *Applied Sciences*, vol. 8, no. 10, 2018.

[9] ——, "Malware collusion attack against machine learning based methods: issues and countermeasures," in *Cloud Computing and Security*, X. Sun, Z. Pan, and E. Bertino, Eds. Cham: Springer International Publishing, 2018, pp. 465–477.

[10] D. Davidson *et al.*, "Enhancing Android security through app splitting," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Springer International Publishing, 2018, pp. 24–44.

[11] K. Elish *et al.*, "Identifying mobile inter-app communication risks," *IEEE Transactions on Mobile Computing*, 2018.

[12] Google, Inc., "The Google Android Security Team's classifications for potentially harmful applications," Feb. 2017.

[13] McAfee, "Safeguarding against colluding mobile apps," May 2016.

[14] I. Muttik, "Partners in crime: investigating mobile app collusion," in *McAfee Labs threats report*. McAfee, Jun. 2016, pp. 8–15.

[15] Android Open Source Project *et al.*, "Android security 2017 year in review," Mar. 2018.

[16] L. Qiu, Y. Wang, and J. Rubin, "Analyzing the analyzers: FlowDroid/IccTA, AmanDroid, and DroidSafe," in *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis - ISSTA 2018*. ACM Press, 2018.

[17] F. Wei *et al.*, "Amandroid: A precise and general inter-component data flow analysis framework for security vetting of Android apps," *ACM Transactions on Privacy and Security*, vol. 21, no. 3, pp. 1–32, apr 2018.

[18] J. Zhan *et al.*, "Splitting third-party libraries' privileges from Android apps," in *Information Security and Privacy*. Springer International Publishing, 2017, pp. 80–94.

remove nocite

Todo list

add actual abstract and keywords 1
Citation needed 1: 2
Citation needed 2: Hardy, N.: The confused deputy:(or why capabilities might have been invented. ACMSIGOPS Operating Systems Review 22(4), 36–38 (1988) 2
many things affect, including non technical 2
difficult to distinguish 2
Citation needed 3: 3
Citation needed 4: 3
Citation needed 5: list some references 3
short description of MoPlus SDK 3
remove nocite 5
remove list of todos 6
remove list of todos