XXXXXXXX

Daniel E. Krutz and xxxxxxxxx
Software Engineering Department
Rochester Institute of Technology
{dxkvse, XXXXX}@rit.edu

ABSTRACT

Mobile devices have not only changed the way we use computing, but also the way we live. Android has grown to be the most popular platform in the world, largely to its flexibly to work on a wide range of devices and the ability for user's to install applications (apps) from a wide range or sources.

The mobile revolution has opened the door for a variety of new types of apps and new developers into the apps race. Unfortunately, mobile apps are not immune to the problems which plague conventional software including bugs, and security vulnerabilities. Examining version control systems (VCS) of open source applications is a good way of understanding when, why and who introduced defects and various types of security vulnerabilities.

In the following work, we examine over **[XXXX]** open source applications and over **[XXXX]** versions of these applications in order to gain a better understanding of why bugs and security vulnerabilities are created in apps, when they typically appear in the lifecycle of the apps, and if the same vulnerabilities typically reappear in apps.

Categories and Subject Descriptors

D.2.7 [**Software Engineering**]: Maintenance;

Keywords

Code Clones, Concolic Analysis, Software Engineering

1. INTRODUCTION

2. RELATED WORKS

3. RESEARCH QUESTIONS

RQ1: How does time affect security and quality of the app?

RQ2: How do committers affect the quality of an app? - Diversity (number) of developers - Experience of developers - Are some developers more - Work across many applications

RQ3: What tendencies do Overprivs have in apps?

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$15.00.

- Exist at beginning and not get fixed. - Exist at beginning, but are fixed. - Not exist at beginning and are later added to app - Are fixed and then come back - If there is one, are there many?

- % of apps with at least 1 over prove in development cycle - See "XOverPrivCount.xls" - 122/339 = 36% - Latest version of app

49/339 had at least 1 over prove ???? - This is off - Avg with at least 1 = 1.84 - Count with at least 1 49 2 18 3 8 4 5 5 4 6 2 7 2 8 2 9 0

– If an App has an overpriv, how likely is it to be underprived?

4. ANDROID APPLICATIONS

5. APP COLLECTION & STATIC ANALYSIS

6. PUBLICLY AVAILABLE DATASET

7. EVALUATION & ANALYSIS

8. LIMITATIONS

[update all of this from ICSE paper] While Stowaway is a powerful statical analysis tool which has been used in a substantial amount of previous research [1, 2, 4], it does suffer some drawbacks. Malicious code may be obfuscated and unnecessary API methods inserted into the application, rationalizing the permission [6]. Static analysis techniques can also be hindered by the Java reflection and may lead to inaccuracies [3, 5]. These types of limitations are inherent to all statical analysis tools.

We only analyzed applications from GooglePlay and not other sources such as AppksAPK or F-Droid, which would have led to more varied application origins. However, we feel the diversity of our applications was already quite robust since we collected 30,020 applications from 41 genres.

We also only examined free applications in our research due to cost constants. Thus, the measurements comparison of apps is not representative of the entire Android app market. Our results only apply as a comparison of free apps, not with paid apps.

9. FUTURE WORK

10. CONCLUSION

References

[1] J. Jeon, K. K. Micinski, J. A. Vaughan, N. Reddy, Y. Zhu, J. S. Foster, and T. Millstein. Dr. android and mr. hide: Fine-grained security policies on unmodified android. 2011.

[2] P. Pearce, A. P. Felt, G. Nunez, and D. Wagner. Addroid: Privilege separation for applications and advertisers in android. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, pages 71–72, New York, NY, USA, 2012. ACM.

[3] M. Sridharan and R. Bodík. Refinement-based context-sensitive points-to analysis for java. *SIGPLAN Not.*, 41(6):387–400, June 2006.

[4] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen. Investigating user privacy in android ad libraries.

[5] O. Tripp, M. Pistoia, S. J. Fink, M. Sridharan, and O. Weisman. Taj: Effective taint analysis of web applications. In *Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '09, pages 87–97, New York, NY, USA, 2009. ACM.

[6] W. Xu, F. Zhang, and S. Zhu. Permlyzer: Analyzing permission usage in android applications. In *Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on*, pages 400–410, 2013.