Project Proposal

Group Name: Aw, Snap!
Group Members: Weijia Sun, Xinyu Lyu, Mengmei Ye

Project Title
Deep Understanding on Taint Analysis: a Survey on Existing Taint Analysis Techniques

Project Type
Course Report

1 Problem Statement

Taint analysis has been widely applied to enhance application/system security. Researchers/engineers on the security community and the software engineering community have been actively working on taint analysis for many years. Considering the significance of taint analysis in both of the community, we plan to collect the recent research work and do a survey report on the existing taint analysis techniques.

Particularly, we are going to target on the following three research questions in this course project.

Research Questions (RQs):

- RQ1: what are the software-engineering concepts behind taint analysis?
- RQ2: what are the advantages and limitations of taint analysis?
- RQ3: how efficient are the taint analysis techniques for Android malware analysis?

To answer the questions, we will be studying on recent research work published in the security or software-engineering conferences. Also, we will be empirically evaluating the efficiency of the open-source taint analysis tools about how they successfully analyze and detect Android malware.

Specifically, for RQ1, we plan to explain how taint analysis works from the view of software engineering. For example, the software-engineering techniques in taint analysis could contain static analysis, dynamic analysis, symbolic execution analysis, LLVM compiler, as well as the lifetime and propagation of variables [1][2][3]. Based on the techniques, we will be elaborating the workflow of the taint analysis and recent improvement on these techniques.

Furthermore, we will be discussing how the taint analysis could help in the security community for RQ2 (i.e., how taint analysis defends against malware). Also, we will be discussing the challenges (e.g., over-taint problems [4]) of taint analysis and the security problems that taint analysis cannot solve.

In addition, we plan to demonstrate the well-known taint analysis tools and further evaluate their efficiency by executing the malware benchmarks for RQ3. Since there are many applications that support taint analysis, such as for Android systems and web applications, we will be targeting in Android malware analysis only to narrow down our experiment scope. Specifically, we will be reproducing some experimental work similar to [5] and [6] to evaluate the open-source taint analysis tools (e.g., [7] and [8]) for Android systems.

After analyzing all the RQs, we plan to have a summary about when to use which type of taint analysis in order to solve which security threat. Also, we will be classifying the existing taint analysis tools based on malware taxonomy and different types of software-engineering techniques.

2 Project Motivation

There has been much excellent research work published in both of the security community and the software engineering community. By reviewing the research papers published in recent years and evaluating the open-source tools for taint analysis, this comprehensive study could help researchers obtain the summary of recent work related to taint analysis as well as help us gain technical experience on the security and software engineering domains. In detail, the motivation for the project is listed in the following.

1. Contribution to the software-engineering domain. The taint analysis technology originates from the software engineering domain. The survey could be helpful for software-engineering researchers to have a comprehensive view about how the taint analysis technology was developed and improved in recent years.
2. Contribution to the security domain. Based on the RQs in our report, security researchers could have a collection of information on the advantages and disadvantages of taint analysis, which could help them further protect against the security vulnerabilities in the systems/applications.
3. Our personal interest and future career development. All of our group members have both backgrounds on software engineering and security. Working on the taint analysis is a great fit for us, and we have a strong passion for gaining a deep understanding on taint analysis. The survey on taint analysis would be a great opportunity for us to enhance technical skills related to software engineering and security.

3 Project Plan and Timeline

Until now, we have proposed our project overview. In this section, we are going to have a timeline regarding our project progress.

By Mar 26, we plan to review research papers and execute the existing taint analysis tools for our RQ1 and RQ2. Then, we will be narrowing down the scope of the project by working on

Android malware analysis only and having a detailed plan about how we measure the efficiency of the tools.

By Apr 15, we will be following the experiment plan mentioned in the report of Mar 26 and further conducting the experiments for our RQ3.

By May 6, we will be constructing our final report based on the previous reports as well as drawing the conclusion based on the results of RQs. Last but not least, we will be proposing potential research work in the taint analysis domain.

References

[1] Jovanovic, Nenad, Christopher Kruegel, and Engin Kirda. "Pixy: A static analysis tool for detecting web application vulnerabilities." In S&P, pp. 6-pp. IEEE, 2006.

[2] Newsome, James, and Dawn Xiaodong Song. "Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software." In NDSS, vol. 5, pp. 3-4. 2005.

[3] Cadar, Cristian, Daniel Dunbar, and Dawson R. Engler. "KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs." In OSDI, vol. 8, pp. 209-224. 2008.

[4] Schwartz, Edward J., Thanassis Avgerinos, and David Brumley. "All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask)." In S&P, pp. 317-331. IEEE, 2010.

[5] Qiu, Lina, Yingying Wang, and Julia Rubin. "Analyzing the Analyzers: FlowDroid/IccTA, AmanDroid, and DroidSafe." In ISSTA, pp. 176-186. ACM, 2018.

[6] Pauck, Felix, Eric Bodden, and Heike Wehrheim. "Do Android taint analysis tools keep their promises?." In ESEC/FSE, pp. 331-341. ACM, 2018.

[7] Gordon, Michael I., Deokhwan Kim, Jeff H. Perkins, Limei Gilham, Nguyen Nguyen, and Martin C. Rinard. "Information Flow Analysis of Android Applications in DroidSafe." In NDSS, vol. 15, p. 110. 2015.

[8] Arzt, Steven, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, and Patrick McDaniel. "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps." In PLDI, no. 6 (2014): 259-269.