Literature Survey on Tools for Identifying and Replacing 3rd-party Libraries in the Context of Android Mobile App Development

Goal of the paper

The goal of the paper is to collect research papers and conduct a literature survey on a given topic, which is "Literature survey on tools for identifying and replacing 3rd-party libraries in the context of Android mobile app development".

The research questions are:

1. What support tools are available to detect third-party libraries used in Android apps?
2. What support tools are available to assist developers if they want to change the third-party libraries in Android apps with 1) an alternative library 2) newer/stable version of the same third-party library?
3. What techniques/approaches (e.g cluster-based approach, machine-learning based, white-list based approach) these tools are using to detect/replace third-party libraries?
4. Which types (e.g ads, billing, network, social networking, game engine etc.) of third-party libraries and related issues/risks (energy consumption, security, performance, etc) can be detected/resolved by these support tools?

Method to conduct the systematic literature survey

The method that was used to conduct literature survey and collect different relevant papers was snowballing. The snowballing method can be done in two ways: forward and backward snowballing. In this paper, the backward snowballing was used.
Before conducting the snowballing method, we need to have a set of initial papers from which we will start looking for related papers that answer to the research questions.
For backward snowballing, the first step is to look at the titles of research papers in the reference list. To know which paper might or might not be relevant, there is a set of criteria and the reference title needs to meet at least one of them:

1. Some kind of tool's name needs to be in the title;
2. Title mentions updating third-party libraries in Android apps;

3. Title mentions detecting third-party libraries in Android apps;
4. Title mentions replacing third-party libraries in Android apps;
5. Title mentions API detection in Android apps.

After finding the possibly relevant papers from the reference list, the second step in backward snowballing was to look at the abstract of the paper. When the abstract doesn't meet the criteria, then the paper needs to be excluded from the set of papers.
The criteria to include/exclude papers from the set is that the abstract has to answer to at least one of the research questions.
The last step of the backward snowballing is to look at the full paper. If the paper answers to the research questions mentioned before, then it can be included in the set of the relevant papers.
To conduct a systematic literature survey, new relevant papers have to be found from the start set of papers using the steps mentioned above and then be either included or excluded from the study. The method can be iterated many times until there are no new papers to be found or until the method has given enough papers to conduct the literature survey.

Snowballing

Start set of papers

To perform the snowballing method, I first received a set of five papers as a starting set. To find more papers to use as a starting set, I worked through the abstract of the received papers and wrote down a group of key phrases that I used to find more papers as a starting point.

The key phrases included:
1. Detecting 3rd party libraries in Android development;
2. 3rd party libraries in Android development;
3. 3rd party library detection;
4. Replacing 3rd party library in Android development;
5. API detection in Android development;
6. Evaluation of 3rd party libraries in Android;
7. Updating 3rd party libraries in Android;

8. API usage in Android.

To include papers in the start set, the papers needed to match the criteria mentioned in the method description.
In the end, the start set consisted of two parts: the initial set of five papers, that was sent to me and seven papers that I found from Google Scholar search, using the key phrases I wrote down. The start set of papers are denoted P1, P2, P3 and so forth.

From the final start set I conducted 2 iterations of backward snowballing and found 22 relevant research papers.

First iteration

From the start set of 12 papers, the forward snowballing was conducted, studying the references of the 12 papers. The papers from the start set were evaluated separately (denoted P1-P12).

The papers are:
P1. Ma Z, Wang H, Guo Y, Chen X. Libradar: Fast and accurate detection of third-party libraries in Android apps. *ICSE '16: Proceedings of the 38th International Conference on Software Engineering Companion,* 2016, 653-656.
P2. Liu B, Jin H, Govindan R. Efficient privilege de-escalation for ad libraries in mobile apps. *MobiSys '15: Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services,* 2015, 89-103.
P3. Backes M, Bugiel S, Derr E. Reliable third-party library detection in Android and its security applications. *CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security,* 2016, 356-367.
P4. Huang J, Borges N, Bugiel S, Backes M. Up-to-crash: Evaluating third-party library updatability on Android. *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019, 15-30.
P5. Salza P, Palomba F, Di Nucci D, De Lucia A, Ferrucci F. Third-party libraries in mobile apps. *Empir Software Eng*, 2019.
P6. Concannon B. Android third party library detection. *Creative Components,* 2018, 153.

P7. Zhang Y, Dai J, Zhang X, Huang S, Yang Z, Yang M, Chen H. Detecting third-party libraries in Android applications with high precision and recall. *IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2018, 141-152.

P8. Han H, Li R, Tang J. Identify and inspect libraries in Android applications. *Wireless Pers Commun 103*, 2018, 491-503.

P9. Zhang J, Beresford A R, Kollmann S A. LibID: Reliable identification of obfuscated third-party Android libraries. *ISSTA 2019: Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis,* 2019, 55-65.

P10. Lim K, Han J, Kim B, Cho S, Park M, Han S. Open-source Android app detection considering the effects of code obfuscation. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 2018, 50-61.

P11. Tang Z, Xue M, Meng G, Ying C, Liu Y, He J, Zhu H, Liu Y. Securing Android application via edge assistant third-party library detection. *Computers & security 80*, 2019, 257-272.

P12. Ogawa H, Takimoto E, Mouri K, Saito S. User-side updating of third-party libraries for Android applications. *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, 2018, 452-458.

In the first iteration I looked at the 12 starting papers, which all together had 495 references listed in the reference list (counting duplicates). Many of the references were the same, so I could exclude those references right away. Then I conducted backwards snowballing and first I looked at the references' titles and those that were irrelevant, I excluded. To decide whether to include or exclude a paper from the set, the titles had to match the criteria mentioned in the beginning. When I had excluded all other papers from the reference list, then I looked at the abstracts of the remaining papers to see if these papers answer any of the questions that was set as a criteria that the papers would need to answer that are mentioned as the research questions.

After this step, I looked at the full papers to be sure if they answer the initial research questions. In the end, from the first iteration, I included in total of 9 research papers to the study.

Second iteration

The second iteration was conducted from the 9 research papers that were found in the first iteration. Again, the papers were evaluated separately (denoted P13-P21).

The papers are:

P13. Narayanan A, Chen L, Chan C K. AdDetect: Automated detection of Android ad libraries using semantic analysis. *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014, 1-6.

P14. Rasthofer S, Arzt S, Bodden E. A machine-learning approach for classifying and categorizing Android sources and sinks. *2014 Network and Distributed System Security Symposium (NDSS)*, 2014.

P15. Li B, Zhang Y, Li J, Feng R, Gu D. AppCommune: Automated third-party libraries de-duplicating and updating for Android apps. *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2019, 344-354.

P16. Derr E, Bugiel S, Fahl S, Acar Y, Backes M. Keep me updated: An empirical study of third-party library updatability on Android. *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, 2187-2200.

P17. Salza P, Palomba F, Di Nucci D, D'Uva C, De Lucia A, Ferrucci F. Do developers update third-party libraries in mobile apps? *ICPC '18: Proceedings of the 26th Conference on Program Comprehension,* 2018, 255-265.

P18. Kula R G, German D M., Ouni A, Ishio T, Inoue K. Do developers update their library dependencies? *Empir Software Eng 23*, 2018, 384-417.

P19. Soh C, Tan H B K, Arnatovich Y L, Narayanan A, Wang L. LibSift: Automated detection of third-party libraries in Android applications. *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*, 2016, 41-48.

P20. Titze D, Lux M, Schütte J. Ordol: Obfuscation-resilient detection of libraries in Android applications. *2017 IEEE Trustcom/BigDataSE/ICESS*, 2017, 618-625.

P21. Wang Y, Wu H, Zhang H, Rountev A. Orlis: Obfuscation-resilient library detection for Android. *2018 IEEE/ACM 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, 2018, 13-23.

In the second iteration, 9 new papers had all together 337 references listed in the reference list (counting duplicates). The papers shared a lot of the same references from the last iteration, which I excluded from the study. Then I evaluated the references left and based on the criteria mentioned initially, I excluded the irrelevant references. After this step I looked at the remaining papers' abstracts and evaluated if these papers answer the given research questions. As a last step I looked at the whole paper and after the second iteration, I included to the study a total of 4 papers (denoted P22-P25).

The papers are:

P22. Li M, Wang W, Wang P, Wang S, Wu D, Liu J, Xue R, Huo W. LibD: Scalable and precise third-party library detection in Android markets. *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, 2017, 335-346.

P23. Chi Z M. LibDetector: Version identification of libraries in Android applications. *Rochester Institute of Technology*, 2016.

P24. McDonnel T, Ray B, Kim M. An empirical study of API stability and adoption in the Android ecosystem. *2013 IEEE International Conference on Software Maintenance*, 2013, 70-79.

P25. Mojica I J., Adams B, Nagappan M, Dienst S, Berger T, Hassan A E. A large-scale empirical study on software reuse in mobile apps. *IEEE Software, vol. 31, no. 2*, 2014, 78-86.

Analysis

After two iterations of forward snowballing, I have 25 papers including the 12 initial papers. The following is the analysis of the 25 papers to answer the questions mentioned before.

Denotation	Presents a tool?	Type of tool	Approach	Type of library	Link
P1	Yes	Detection	Cluster-based	Any	https://github.com/pkumza/LibRadar
P2	Yes	Detection	Machine-learning based	Ad	-
P3	No	-	-	-	-
P4	No	-	-	-	-
P5	No	-	-	-	-
P6	No	-	-	-	-
P7	Yes	Detection	Class-dependency based	Any	https://github.com/yuanxzhang/LibPecker
P8	No	-	-	-	-
P9	Yes	Detection	Locality-Sensitive Hashing	Any	https://github.com/ucam-cl-dtg/LibID
P10	No	-	-	-	-
P11	Yes	Detection	Machine-learning based	Any	PanGuard
P12	No	-	-	-	-
P13	Yes	Detection	Machine-learning based	Ad	-
P14	Yes	Detection	Machine-learning based	Any	https://github.com/secure-software-engineering/SuSi
P15	Yes	Updating	White-list based	Any	-
P16	No	-	-	-	-
P17	No	-	-	-	-
P18	No	-	-	-	-
P19	Yes	Detection	Package-dependency based	Any	-
P20	Yes	Detection	Iterative heuristic approach	Any	https://github.com/milux/ordol

Denotation	Presents a tool?	Type of tool	Approach	Type of library	Link
P21	Yes	Detection	Using interprocedural code features and similarity digests	Any	https://github.com/presto-osu/orlis-orcis
P22	Yes	Detection	Classification technique	Any	https://github.com/IIE-LibD/libd
P23	Yes	Version Identification	White-list based	Any	https://github.com/zchi88/LibDetector
P24	No	-	-	-	-
P25	No	-	-	-	-

References to conduct snowballing

[1] Kitchenham B. Guidelines for performing systematic literature reviews in software engineering. *EBSE Technical Report*, 2007.

[2] Wohlin C. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. *EASE '14: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering,* 2014, 1-10.

References of papers included in the survey

P1. Ma Z, Wang H, Guo Y, Chen X. Libradar: Fast and accurate detection of third-party libraries in Android apps. *ICSE '16: Proceedings of the 38th International Conference on Software Engineering Companion,* 2016, 653-656.

P2. Liu B, Jin H, Govindan R. Efficient privilege de-escalation for ad libraries in mobile apps. *MobiSys '15: Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services,* 2015, 89-103.

P3. Backes M, Bugiel S, Derr E. Reliable third-party library detection in Android and its security applications. *CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security,* 2016, 356-367.

P4. Huang J, Borges N, Bugiel S, Backes M. Up-to-crash: Evaluating third-party library updatability on Android. *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019, 15-30.

P5. Salza P, Palomba F, Di Nucci D, De Lucia A, Ferrucci F. Third-party libraries in mobile apps. *Empir Software Eng*, 2019.

P6. Concannon B. Android third party library detection. *Creative Components,* 2018, 153.

P7. Zhang Y, Dai J, Zhang X, Huang S, Yang Z, Yang M, Chen H. Detecting third-party libraries in Android applications with high precision and recall. *IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2018, 141-152.

P8. Han H, Li R, Tang J. Identify and inspect libraries in Android applications. *Wireless Pers Commun 103*, 2018, 491-503.

P9. Zhang J, Beresford A R, Kollmann S A. LibID: Reliable identification of obfuscated third-party Android libraries. *ISSTA 2019: Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis,* 2019, 55-65.

P10. Lim K, Han J, Kim B, Cho S, Park M, Han S. Open-source Android app detection considering the effects of code obfuscation. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 2018, 50-61.

P11. Tang Z, Xue M, Meng G, Ying C, Liu Y, He J, Zhu H, Liu Y. Securing Android application via edge assistant third-party library detection. *Computers & security 80*, 2019, 257-272.

P12. Ogawa H, Takimoto E, Mouri K, Saito S. User-side updating of third-party libraries for Android applications. *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, 2018, 452-458.

P13. Narayanan A, Chen L, Chan C K. AdDetect: Automated detection of Android ad libraries using semantic analysis. *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014, 1-6.

P14. Rasthofer S, Arzt S, Bodden E. A machine-learning approach for classifying and categorizing Android sources and sinks. *2014 Network and Distributed System Security Symposium (NDSS)*, 2014.

P15. Li B, Zhang Y, Li J, Feng R, Gu D. AppCommune: Automated third-party libraries de-duplicating and updating for Android apps. *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2019, 344-354.

P16. Derr E, Bugiel S, Fahl S, Acar Y, Backes M. Keep me updated: An empirical study of third-party library updatability on Android. *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, 2187-2200.

P17. Salza P, Palomba F, Di Nucci D, D'Uva C, De Lucia A, Ferrucci F. Do developers update third-party libraries in mobile apps? *ICPC '18: Proceedings of the 26th Conference on Program Comprehension,* 2018, 255-265.

P18. Kula R G, German D M., Ouni A, Ishio T, Inoue K. Do developers update their library dependencies? *Empir Software Eng 23*, 2018, 384-417.

P19. Soh C, Tan H B K, Arnatovich Y L, Narayanan A, Wang L. LibSift: Automated detection of third-party libraries in Android applications. *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*, 2016, 41-48.

P20. Titze D, Lux M, Schütte J. Ordol: Obfuscation-resilient detection of libraries in Android applications. *2017 IEEE Trustcom/BigDataSE/ICESS*, 2017, 618-625.

P21. Wang Y, Wu H, Zhang H, Rountev A. Orlis: Obfuscation-resilient library detection for Android. *2018 IEEE/ACM 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, 2018, 13-23.

P22. Li M, Wang W, Wang P, Wang S, Wu D, Liu J, Xue R, Huo W. LibD: Scalable and precise third-party library detection in Android markets. *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, 2017, 335-346.

P23. Chi Z M. LibDetector: Version identification of libraries in Android applications. *Rochester Institute of Technology*, 2016.

P24. McDonnel T, Ray B, Kim M. An empirical study of API stability and adoption in the Android ecosystem. *2013 IEEE International Conference on Software Maintenance*, 2013, 70-79.

P25. Mojica I J., Adams B, Nagappan M, Dienst S, Berger T, Hassan A E. A large-scale empirical study on software reuse in mobile apps. *IEEE Software, vol. 31, no. 2*, 2014, 78-86.