Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 83 (2016) 1219 – 1225

Procedia Computer Science

Second International Workshop on Mobile Cloud Computing systems, Management, and Security (MCSMS-2016)

SCREDENT: **Sc**alable **Re**al-time Anomalies **De**tection and **N**otification of **T**argeted Malware in Mobile Devices

Paul McNeil*, Sachin Shetty*, Divya Guntu, Gauree Barve

Tennessee State University, 3500 John A. Merritt Blvd. Nashville, TN 37209

Abstract

The ubiquitous availability of Android devices has led to increasing malicious mobile attacks targeting the Android mobile operating system. In recent times, adversaries leverage situational awareness, user and device context to create targeted malware for mobile devices. Several mobile security tools such as Mobile Sandbox, TargetDroid, and ANANAS focus on tailoring the detection schemes for individual users and suffer from scalability by analyzing individual user's activities. To the best of our knowledge, these tools do not incorporate user group profiling in their automated user-behavior driven dynamic analysis. In addition, adaptive and location-based alerts are not provided to mobile users. We propose SCREDENT: Scalable Real-time Anomalies Detection and Notification of Targeted Malware in Mobile Devices, to provide a scalable system to classify, detect, and predict targeted malware in real-time. SCREDENT incorporates behavior-triggering probabilistic models and user grouping to minimize the number of parallel dynamic analysis instances needed. SCREDENT leverages container technology to perform dynamic analysis and allow for modularity as emulation technology improves. SCREDENT uses adaptive, location-based notification principles to create a geographical fence which warn users of malicious attacks. Finally, SCREDENT provides proactive, adaptive alerts to individual users if at least one of the group members has triggered malicious activities in an application currently used by the individual.

Peer-review under responsibility of the Conference Program Chairs

Keywords: Cloud Computing; Data Analytics; Big Data; Malware; Mobile Security; Container Technology; Machine Learning; Modeling; Dynamic Analysis; Android; Mobile Malware Detection; Location-Based Notification

*Corresponding authors. Tel.: 615-963-2160; fax: 615-963-2165
E-mail address: pmcneil@my.tnstate.edu, sshetty@tnstate.edu.

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2016.04.254

1. Introduction

About 84% of all smartphones, worldwide, are Android devices[28] and the majority of these devices are unprotected[29]. Given Android's prominence and general smartphone vulnerability, it is not surprising that the majority malicious mobile attacks are designed for the Android mobile operating system. Lately, mobile malware that use contextual device and user behavioral data to avoid detection and observation have emerged. This targeted malware only executes when certain conditions are met[22, 15, 5]. Several analytic tools such as Mobile Sandbox, TargetDroid, and ANANAS have been developed to take a hybrid approach to identifying targeted malware[18, 9, 2, 4]. There are also tools such as DREBIN, Marvin, and Draco which have mobile applications to provide static on device analysis[17, 12, 11, 7, 18, 21, 24]. Draco, though, combines its on-device static analysis with remote dynamic analysis remotely[11]. These tools analyze individual mobile/user activities to detect malware. In the presence of a large number of mobile devices and users, even distributed analyses will not be sufficient to provide efficient and timely detection. To the best of our knowledge, these tools do not integrate group user profiling with their automated user-behavior driven dynamic analysis to perform targeted malware detection. The profiling of groups of mobile users will provide efficient analyses, significantly reduce redundancy and increase probability of adaptive alerts. Further, the tools suffer from usable alert system which informs the user of potential attacks based on user context and location.

The rest of the paper is organized as follows. Section 2 provides an overview of the SCREDENT architecture. Section 3 describes the machine learning strategies SCREDENT employs. In Section 4, we conclude and discuss future work.

2. SCREDENT Overview

SCREDENT collects user behaviors and contextual data from real users. Next, it creates probabilistic models to represent the data to be executed on a cloud-based targeted malware testbed. The models are used to emulate user group behaviors during the dynamic analysis of Android malware. Risk factor is then determined and an adaptive, location based alert is sent to the end user. If a user is entering an area known for malicious attacks, SCREDENT sends a proactive alert. SCREDENT is a system of systems comprised of three individual subsystems we developed using a top-down systems engineering approach (see Figure 1). The key subsystems of SCREDENT are: **U**ser Behavior **M**odeling and **P**rofiling for **S**martphones (UMAPS), DockerDroid, and **Ta**rgeted **M**alware **A**lert and **No**tification **S**ystem (TAMANOS).

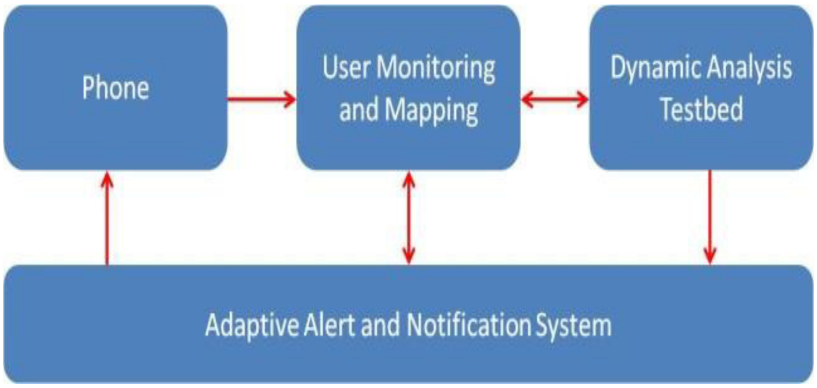

Fig. 1. SCREDENT Architecture.

2.1. UMAPS: *User Behavior Modeling and Profiling for Smartphones*

SCREDENT's user monitoring system, UMAPS, contains two components: logging and mapping (see Fig. 2). The logging component consists of a native Android application which logs replicable contextual and user behavioral data temporarily on the device until a Wi-Fi connection is established (see Table I). The user may change the default settings to upload immediately or whenever mobile data is available. The information logged is then uploaded from the device to the cloud for modeling and dynamic analysis. The native application further performs lightweight static analysis on recently installed application manifests using a remotely trained support vector machine (SVM)[16]. The SCREDENT logging application also uploads the results of the static analysis for applications that are flagged as false positives by users. This allows for amore insightful modifications of the SVM as well as SCREDENT's alert system.

The mapping component then analyzes the uploaded log files, mapping behaviors and probabilities, to create or update stored markovian models for each user (see Figure 3). Then, SCREDENT forms user groups to allow intelligent storage of the models created. Using K-means clustering, intelligent storage decreases the overall number of markov models stored by allowing SCREDENT to flush older model data upon update without losing valuable knowledge of past models[8]. SCREDENT also monitors conceptual drift between each temporarily stored individual model and its corresponding group model. When an individual user model varies too greatly from its corresponding group model, SCREDENT performs K-means clustering again to create new groups (see Figure 3).

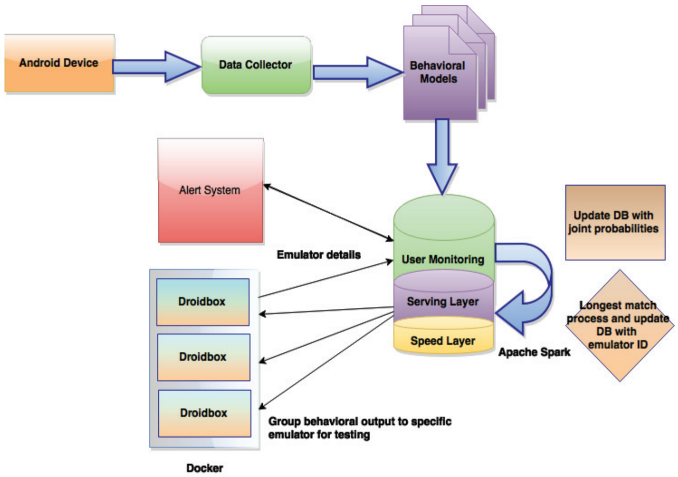

Fig. 2 . UMAPS Architecture.

Table 1. Replicable events in TargetDroid.

Available Events					
Accept call	Change battery status	Uninstall App	Set time zone	Receive SMS	Activate 3G
Change wallpaper	Turn on GPS	Install APKs	Sensor: Accelerometer	Turn of GPS	Change clock
Turn on Airplane mode	Set ringer	Sensor: Gyroscope	Turn off Airplane mode	Cancel Call / Hang up	Turn on screen
Set volume	Sensor: Rotation/Pitch	Turn of Terminal	Connect AC	Send SMS	Get location
Go Home	Lock Terminal	Turn of 3G	Bright auto	Set bright auto	View SMS

2.2. DroidDocker

In the distributed analysis subsystem, DroidDocker, SCREDENT performs both static and dynamic analysis in the cloud (see Figure 4). Here, the on-device SVM model, used to analyze application manifest files, is trained remotely. The SVM classifier uses manifest features identified in recent work[12, 16, 3]. Benign application samples were collected from Google Play and malware samples were collected form Contagion mobile malware minidump. The on-device application is then updated with the resulting SVM. This SVM is regularly updated. These updates leverage information regarding false positives from users to improve the SVM.

The distributed analysis system also executes dynamic analysis. Currently, we employ TargetDroid to perform dynamic analysis[9]. Targetdroid is an Android malware analysis tool that employs behavior-triggering markovian models to detect targeted smartphone malware. It is built around Droidbox, an out-of-the-box Android malware dynamic analysis tool[13]. TargetDroid uses the stochastic models to modify the standard Android emulator settings, like the battery being half full, to aid in triggering malicious behaviors by injecting these events. DroidDocker further facilitates automatically injecting the stored user-triggered events into the emulated environment in a scalable fashion[20]. By modifying TargetDroid, DroidDocker is able to scale its dynamic analysis to fit within the SCREDENT framework. DroidDocker is able to surpass the standard emulator instance limitation and managing event injection with monkeyrunner for each dynamic analysis instance running in parallel[20, 32]. We store the results of SCREDENT's dynamic analysis for subsequent risk analysis and user notification.

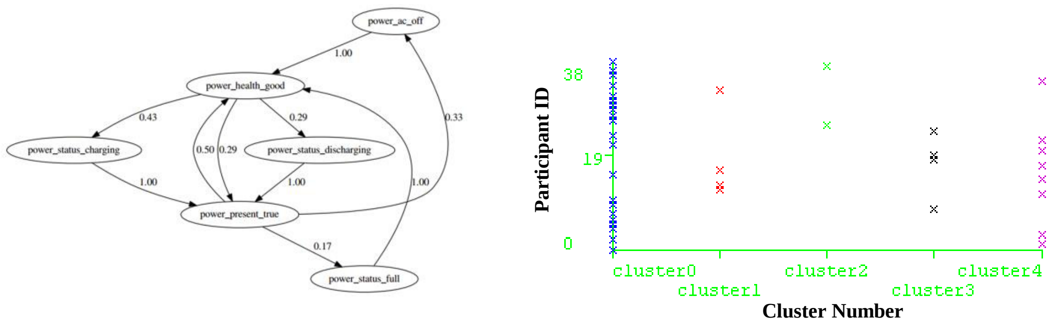

Fig. 3. (a) Markov Model of Battery Status; (b) K-means Group Clustering, k=5.

.

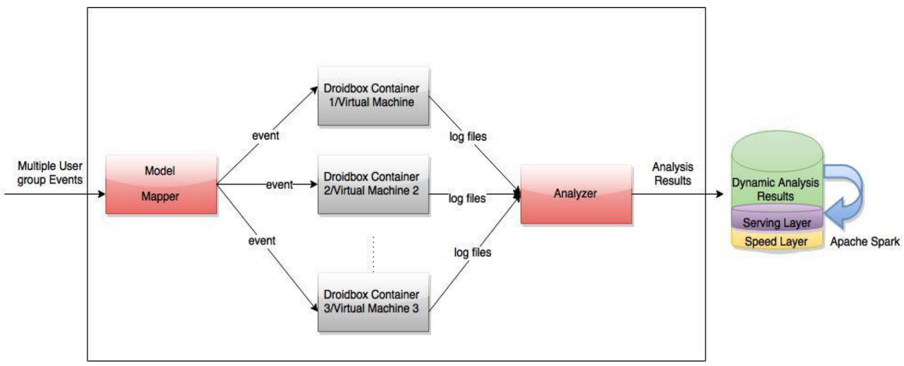

Fig. 4 . DroidDocker Architecture.

2.3. TAMANOS: Targeted Malware Alert and Notification System

SCREDENT uses Targeted Malware Alert and Notification System (TAMANOS) for risk assessment and adaptive alerting (see Figure 5). TAMANOS is an adaptive, location-based targeted malware alert and notification system for Android devices. This subsystem has three main purposes: perform risk assessment of malware analyses, store blacklists, and create adaptive, location-based notifications or proactive alerts.

TAMANOS uses a SVM to determine the maliciousness of an application to user groups based on the dynamic analysis performed by DroidDocker. If malicious, TAMANOS creates customized location-based notifications for each user in the group using individual preferences. These preferences influence whether the alert is visual, tactile, or auditory. Proactive alerts are also sent to users if they enter an area known for malicious attacks if the user is at risk. For example, if a user is entering a neighborhood known for Bluetooth attacks while the user's Bluetooth is on, the user will receive a warning. This proactive functionality extends strategies used in the Early Alert System (EARS) location-based notification system to limit energy consumption and maintain user privacy[14]. Further, TAMANOS maintains the outcome of risk assessments for each application analyzed by SCREDENT. This limits redundancies of analysis and allows faster notification of malicious applications of known malware.

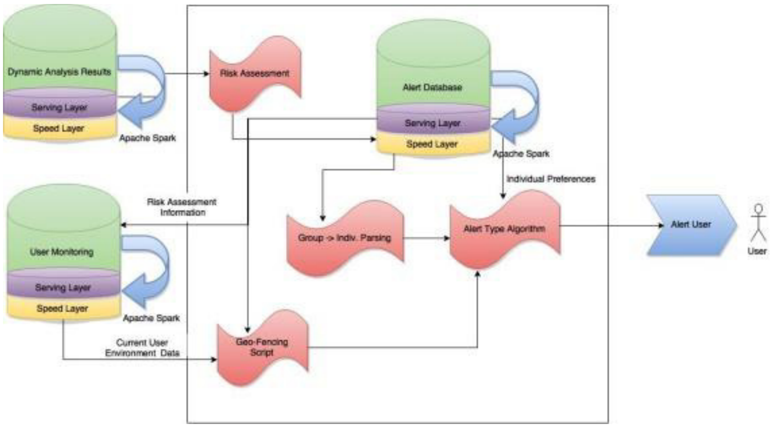

Fig. 5 . TAMANOS Architecture.

3. SCREDENT Scalable Real-time Environment

Two of SCREDENT's main objectives are realized through its infrastructure: scalability and real-time processing. Here, we look at scalability primarily in terms of data collection, analysis, and storage. SCREDENT's native on-device application allows remote data collection to occur simultaneously from a large number of devices. SCREDENT employs MongoDB for persistent and temporary storage in each subsystem[23]. MongoDB allows for simple horizontal scalability as the amount data collected and models stored increases[23]. Cassandra is another option for scalable storage but requires well defined schema. The Android operating system and emulation technologies are constantly evolving, choosing such a rigid structure would be ill-advised. Unlike Cassandra, MongoDB is forgiving when it comes to data schemes which may need to change as the state-of-the-art does[23].

SCREDENT's use of MongoDB also aids its objective to achieve real-time processing. SCREDENT implements lambda architecture using MongoDB and Apache Spark in each subsystem[30, 19, 31]. Each database is separated into a speed, serving, and batch layer. Apache Spark is used to manage constantly running queries to create necessary views[19]. Further, the Spark infrastructure allows for fast message passing from one SCREDENT subsystem to another and model training[31].

SCREDENT implements scalable analysis using Docker's container technology as the testbed platform[10]. SCREDENT manages manage the creation, scheduling, and execution of virtual clones in the cloud using Docker. SCREDENT's modular distributed analysis subsystem connects to standalone databases of contextual or user behavior data to a desired analysis tool's docker image. As stated previous, SCREDENT currently uses TargetDroid for dynamic analysis. Since TargetDroid is out-of-box, it shares the 15 emulator instance restriction that the standard Android emulator possesses[9]. Using Docker allows SCREDENT to easily surpass this limitation and do so with a lower overhead than starting a new virtual machine each time an instance limitation is reached[10]. SCREDENT creates or executes TargetDroid instances according to a longest match algorithm to pair user group models with the emulators that best match the real devices of user group members.

4. Limitations

SCREDENT requires access to Wi-Fi or mobile data at some point conduct its deeper analysis. If the user never enables either, then SCREDENT cannot perform its needed updates. Similarly, SCREDENT requires access to the network GPS in order to perform location-based notifications. SCREDENT shares several attack surfaces, such as man-in-the-middle attacks when uploading log files or attacks to the containers[25, 15, 1]. Additional efforts need to be taken to minimize these surfaces. SCREDENT's dynamic analysis testbed framework allows for a different analysis tool to be substituted as the state of the art progresses. However, SCREDENT currently uses TargetDroid for analysis and inherits TargetDroid limitations[9]. SCREDENT also only replicates a limited amount of interactions as limited by the standard Android SDK. This replication limitation may be remedied by using alternative emulators, such as QEMU-based, or Google releases newer standard emulators[27].

5. Conclusion and Future Work

The increase of next generation mobile malware has inspired the development of next generation malware detection tools. These tools often focus on individual users and fail to provide customized alerts. We presented SCREDENT as a targeted malware classification, detection, and alert tool. SCREDENT uses user group profiling to increase the scalability of data collection and storage. SCREDENT also leverages real time processing to improve analysis performance and user experience.

Although the SCREDENT framework has been implemented, there are optimizations that need to be made. For instance, large scale, long term, end to end testing of SCREDENT needs to be completed. The observations made from real world deployment will provide information regarding improving the models used in each subsystem and how they can be modified for increased accuracy. Large scale testing will also reveal key areas for performance optimizations throughout the system, such as algorithms reducing dynamic analysis runtime. Further, user testing can be done to improve the adaptive user notification experience.

Acknowledgements

This work is supported by Office of the Assistant Secretary of Defense for Research and Engineering agreement FAB750-15-2-0120, NSF CNS-1405681, NSF DUE-1431382, NSF DGE-1303365, and DHS 2014-ST-062-000059.

References

1. Bui, T., 2015. Analysis of Docker Security.
2. Eder, T., Rodler, M., Vymazal, D., Zeilinger, M., 2013. ANANAS - A Framework for Analyzing Android Applications, in: 2013 Eighth International Conference on Availability, Reliability and Security (ARES). Presented at the 2013 Eighth International Conference on Availability, Reliability and Security (ARES), pp. 711–719.
3. Felt, A.P., Chin, E., Hanna, S., Song, D., Wagner, D., 2011a. Android permissions demystified, in: Proceedings of the 18th ACM Conference on Computer and Communications Security. ACM, Chicago, Illinois, USA, pp. 627–638.

4. Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., Weiss, Y., 2011. "Andromaly": a behavioral malware detection framework for android devices. J Intell Inf Syst 38, 161–190.
5. Felt, A.P., Finifter, M., Chin, E., Hanna, S., Wagner, D., 2011b. A survey of mobile malware in the wild, in: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. ACM, Chicago, Illinois, USA, pp. 3–14.
6. Tam, K., Khan, S.J., Fattori, A., Cavallaro, L., 2015. CopperDroid: Automatic Reconstruction of Android Malware Behaviors, in: Proc. of the Symposium on Network and Distributed System Security (NDSS).
7. Burguera, I., Zurutuza, U., Nadjm-Tehrani, S., 2011. Crowdroid: behavior-based malware detection system for android, in: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. ACM, pp. 15–26.
8. Frank, E., Hall, M., Trigg, L., Holmes, G., Witten, I.H., 2004. Data mining in bioinformatics using Weka. Bioinformatics 20, 2479–2481.
9. Suarez-Tangil, G., Conti, M., Tapiador, J., Peris-Lopez, P., 2014. Detecting Targeted Smartphone Malware with Behavior-Triggering Stochastic Models, in: Kutyłowski, M., Vaidya, J. (Eds.), Computer Security - ESORICS 2014, Lecture Notes in Computer Science. Springer International Publishing, pp. 183–201.
10. Merkel, D., 2014. Docker: lightweight Linux containers for consistent development and deployment. Linux J. 2014, 2.
11. Bhandari, S., Gupta, R., Laxmi, V., Gaur, M.S., Zemmari, A., Anikeev, M., 2015. DRACO: DRoid Analyst Combo an Android Malware Analysis Framework, in: Proceedings of the 8th International Conference on Security of Information and Networks, SIN '15. ACM, New York, NY, USA, pp. 283–289.
12. Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., Rieck, K., n.d. Drebin: Efficient and explainable detection of android malware in your pocket. Proc. of 17th Network and Distributed System Security Symposium, NDSS 14.
13. Lantz, P., Desnos, A., Yang, K., 2012. DroidBox: Android application sandbox.
14. Jin, G., Deng, J., Nguyen, T., Gao, P., Wooster, M.T., Qari, S.H., 2015. Efficient Cloud-Based Real-Time Geo-Information Delivery for Mobile Users, in: Mobile Data Management (MDM), 2015 16th IEEE International Conference on. IEEE, pp. 251–254.
15. Suarez-Tangil, G., Tapiador, J.E., Peris-Lopez, P., Ribagorda, A., 2014. Evolution, Detection and Analysis of Malware for Smart Devices. Communications Surveys & Tutorials, IEEE 16, 961–987.
16. Ham, H.-S., Kim, H.-H., Kim, M.-S., Choi, M.-J., Ham, H.-S., Kim, H.-H., Kim, M.-S., Choi, M.-J., 2014. Linear SVM-Based Android Malware Detection for Reliable IoT Services, Linear SVM-Based Android Malware Detection for Reliable IoT Services. Journal of Applied Mathematics, Journal of Applied Mathematics 2014, 2014, e594501.
17. Lindorfer, M., Neugschwandtner, M., Platzer, C., 2015. MARVIN: Efficient and Comprehensive Mobile App Classification through Static and Dynamic Analysis, in: Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual. Presented at the Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual, pp. 422–433. doi:10.1109/COMPSAC.2015.103
18. Spreitzenbarth, M., Freiling, F., Echtler, F., Schreck, T., Hoffmann, J., 2013. Mobile-sandbox: Having a Deeper Look into Android Applications, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13. ACM, New York, NY, USA, pp. 1808–1815.
19. MongoDb and the Lambda Architecture | Expressive Code [WWW Document], n.d. URL http://www.expressivecode.org/2014/11/10/mongodb-and-the-lambda-architecture/ (accessed 12.29.15).
20. monkeyrunner | Android Developers [WWW Document], n.d. URL http://developer.android.com/tools/help/monkeyrunner_concepts.html (accessed 11.16.15).
21. Gianazza, A., Maggi, F., Fattori, A., Cavallaro, L., Zanero, S., 2014. Puppetdroid: A user-centric ui exerciser for automatic dynamic analysis of similar android applications.
22. Hasan, R., Saxena, N., Haleviz, T., Zawoad, S., Rinehart, D., 2013. Sensing-enabled channels for hard-to-detect command and control of mobile devices, in: Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security. ACM, Hangzhou, China, pp. 469–480.
23. Ruflin, N., Burkhart, H., Rizzotti, S., 2011. Social-data storage-systems, in: Databases and Social Networks. ACM, Athens, Greece, pp. 7–12.
24. Yuhui, F., Ning, X., 2015. The Analysis of Android Malware Behaviors. International Journal of Security & Its Applications 9.
25. Wangen, G., 2015. The Role of Malware in Reported Cyber Espionage: A Review of the Impact and Mechanism. Information 6, 183–211.
26. Using the Emulator | Android Developers [WWW Document], n.d. URL http://developer.android.com/tools/devices/emulator.html (accessed 11.16.15).
27. Bellard, F., 2005. QEMU, a Fast and Portable Dynamic Translator, in: Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC '05. USENIX Association, Berkeley, CA, USA, pp. 41–41.
28. IDC: Smartphone OS Market Share [WWW Document], n.d. . www.idc.com. URL http://www.idc.com/prodserv/smartphone-os-market-share.jsp (accessed 12.30.15).
29. Protect your Android device from malware [WWW Document], n.d. . CNET. URL http://www.cnet.com/how-to/protect-your-android-device-from-malware/ (accessed 12.30.15).
30. Marz, N., Warren, J., 2015. Big Data: Principles and Best Practices of Scalable Realtime Data Systems, 1st ed. Manning Publications Co., Greenwich, CT, USA.
31. Reyes-Ortiz, J.L., Oneto, L., Anguita, D., 2015. Big Data Analytics in the Cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf. Procedia Computer Science, INNS Conference on Big Data 2015 Program San Francisco, CA, USA 8-10 August 2015 53, 121–130.
32. Using the Emulator | Android Developers [WWW Document], n.d. URL http://developer.android.com/tools/devices/emulator.html (accessed 11.16.15).