International Journal of Advance Research, IJOAR .org
Volume 1, Issue 7, July 2013, Online: ISSN 2320-9194

SECURING MODELS FOR ANDROID MARKET PLACE

Mukul Kesavan, Lakshmi Raj Sharma

Abstract

The Android platform is that the quickest growing market in smart phone operating systems thus far. As such, it has become the most viable target of security threats. The dependence of the Android Market Security Model on its reactive anti-malware system gifts a chance for malware to be present within the Official Android Market and does not include applications outside the official market. This permits applications to masquerade as harmless applications which result in the loss of credentials if precautions are not taken. Most anti-malware applications within the Market use static analysis for detection as a result of it, they are quick and comparatively easy.

Keywords—Android, Security, Behaviour Analysis

INTRODUCTION

Smart phones are currently a target for malicious software which plan to harm personal assets of users. The Android platform, being the quickest growing market these days, faces identical risk. The malware takes advantage of the platform for its being open, complete and free for development that means that there's lack up to the mark for application development. Permitting anyone to develop and to publish applications into the Android Market presents a chance for attackers to simply deliver malicious applications onto unsuspecting users. The presence of the alternative Android market makes this problem worse owing to the shortage in review strategies thus creating them unreliable sources of applications.

The Android platform utilizes an authorization-based security models to possess access to completely different functionalities of devices. This model provides data regarding the access and the privilege capability of an application which to a lot of technical users, may be used as a sign for malicious intent however to traditional users, this data is often neglected thus creating this model unreliable on its own. Static analysis may be a method adapted to discover malware however this method proves to be inadequate as malware will be unobserved by obfuscation. The time it takes to manually check the code provides a chance for malware to infect devices before they are detected.

To address these issues, an automated behavioural analysis system called AMDA is that the resolution. The AMDA system determines malicious behaviour from benign behaviour through the use of machine learning techniques. A behaviour model for trojans, spyware, viruses and exploits are generated and used for classification of applications. The results are verified by forwarding them to skilled system, VirusTotal.

AMDA

AMDA is an automated malware detection system for the Android platform. This paper includes the discussion of the core modules of the system particularly the Feature Extraction Module and also the Behaviour Analysis Module. These modules are liable for behavioural analysis to discover malicious activity of applications based on the extracted features. This development of the system involves categorization of applications based on the AMDA's classification and cross- validation of the results of the skilled system, which typically concludes the functionalities of the system.

Application Acquisition Module

The Application Acquisition module is liable for downloading applications from Android Markets and storing them into the applying repository within the server. Check applications are downloaded both from the Official Android market and a few different Android markets. Benign applications are downloaded from the Official Android Market and non-malware applications are downloaded from the online Android malware providers like VirusTotal and Contagio. For the downloading of check applications, a web-crawler tool is employed applications from different Android market domains which provide free transfer .Ape files. The applications are forwarded to the VirusTotal Malware Verification System (VMS) to be ready to acquire the classification of the check applications for use in later modules. As for downloading the coaching applications, these are acquired manually in order to make sure the validity of the applications.

Applications from the Official Android Market and also the Android malware providers are acquired manually. The downloaded applications are tested for their validity as benign or as known-malware through the VirusTotal VMS. Once the status of each application is confirmed, the applying is passed on to the repository to be employed by following module, the Feature Extraction module.

Web-crawler Sub module

The web-crawler iterates hand-picked sites from a predefined list of domains to search for complimentary Android applications on .apk format. Once the Android application has been found, the web-crawler downloads the file and stores it within the application repository. Once the web-crawler has finished all the URLs within the list, it forwards the applications to the VirusTotal VMS for classification. Once all the applications are classified, the method of locomotion is perennial starting from the first defined domain. The method could take a while to end counting on the user’s web affiliation and is best done a minimum of daily to check and gather freshly uploaded applications from the choice Android markets.

Feature Extraction Module

The Feature Extraction Module is that the one that generates activity log from running applications retrieved from the applying repository of the system. The activity log contains the system calls from application activity which are the features that the module retrieves. For these features to be extracted, the logs are processed by the Virtualization Submodule which handles monitoring and work of application activity. Features acquired from the Virtualization Submodule are filtered through parsing before being forwarded to the Behaviour-based Analysis Module.

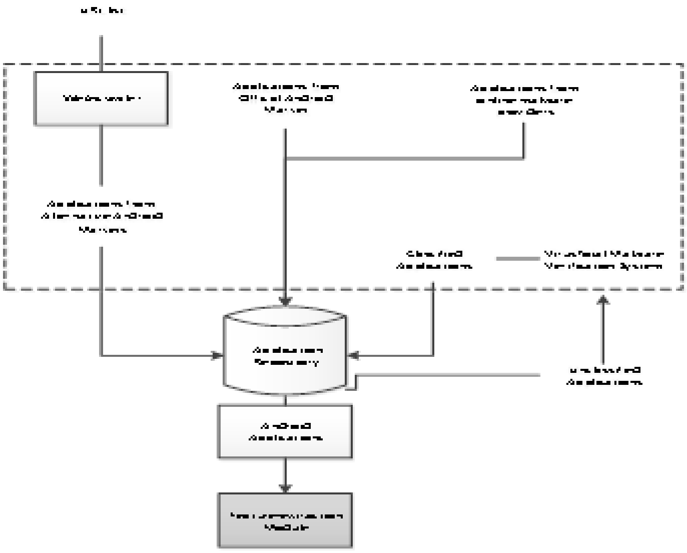


Figure 1. Application Acquisition Module

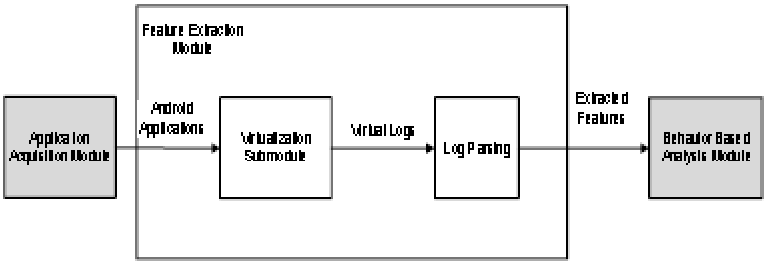

Feature Extraction Module
Application Acquisition Module
Android Applications
Virtualization Submodule
Virtual Logs
Log Parsing
Extracted Features
Behavior Based Analysis Module

Figure 1. Feature Extraction Module

Virtualization Sub module

Android 2.3.3 SDK emulator is employed to run the Android applications as a result of this is the sole medium to automate the generation of application system activity logs without utilization of the actual mobile device. In order to automate the extraction of the system calls generated by the applications, the Android SDK individual is run at the side of Monkey, a tool which simulates user input. According to two, there's no actual distinction to utilize human input to be ready to activate the malicious actions of associate application.

The collection of system calls is completed through the use of Strace. The tool monitors and logs low-level activity in kernel space within the humanoid SDK individual. This method of monitoring low-level system calls ensure that all activity of the applying is recorded. 3 However, the log knowledge contains an activity, which are doubtful for detection of malicious activity. With this problem of noise within the log knowledge, the system utilizes a self-developed parser which might be tailored as to which features are to be collected.

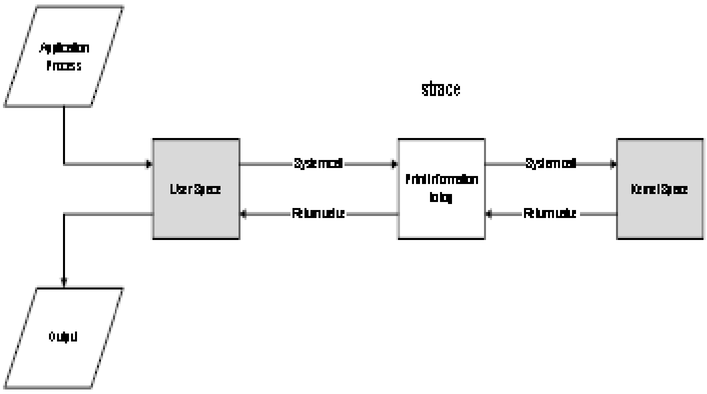


Figure 2. Virtualization Submodule

Features of the system are mapped to activities of the application which may indicate malicious activity. The following system calls are typically executed by Android Malware 34:

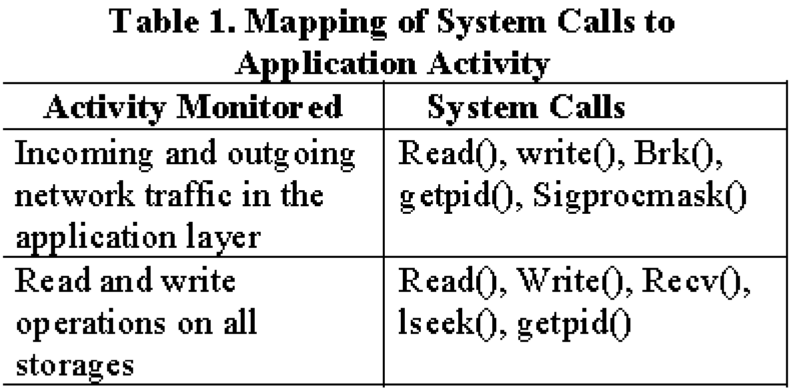
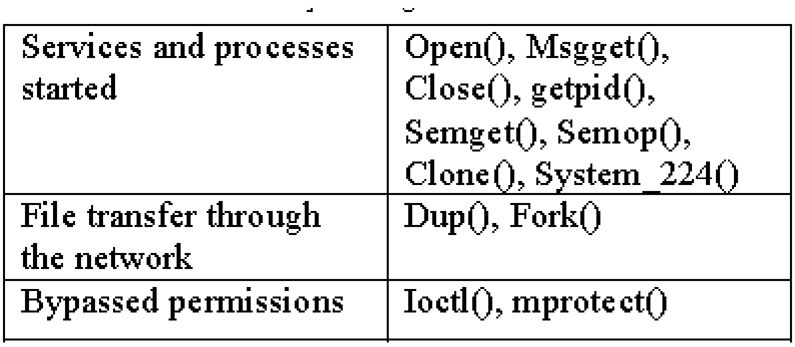


Table 1. Mapping of System Calls to Application Activity

Activity Monitored	System Calls
Incoming and outgoing network traffic in the application layer	Read(), write(), Brk(), getpid(), Sigprocmask()
Read and write operations on all storages	Read(), Write(), Recv(), lseek(), getpid()



Services and processes started	Open(), Msgget(), Close(), getpid(), Semget(), Semop(), Clone(), System_224()
File transfer through the network	Dup(), Fork()
Bypassed permissions	Ioctl(), mprotect()

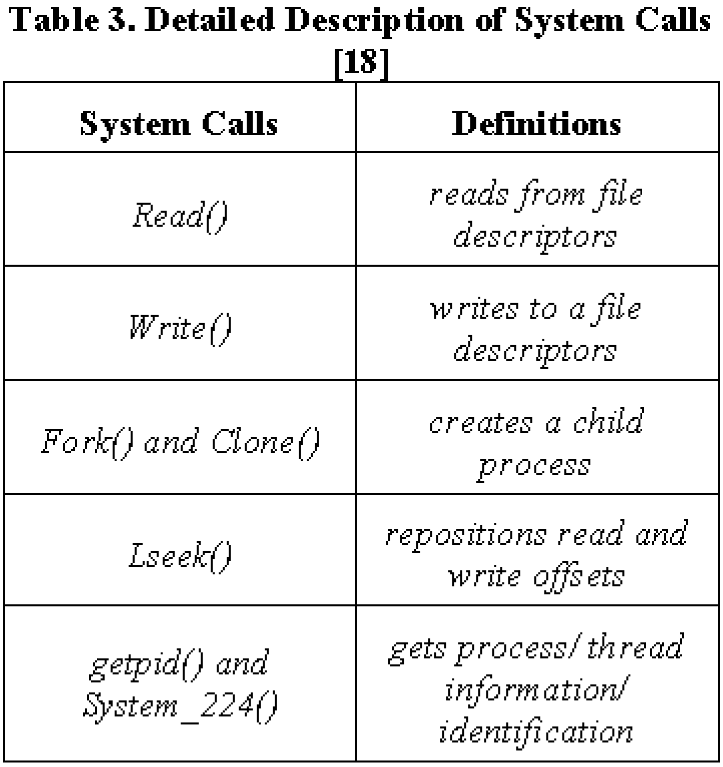
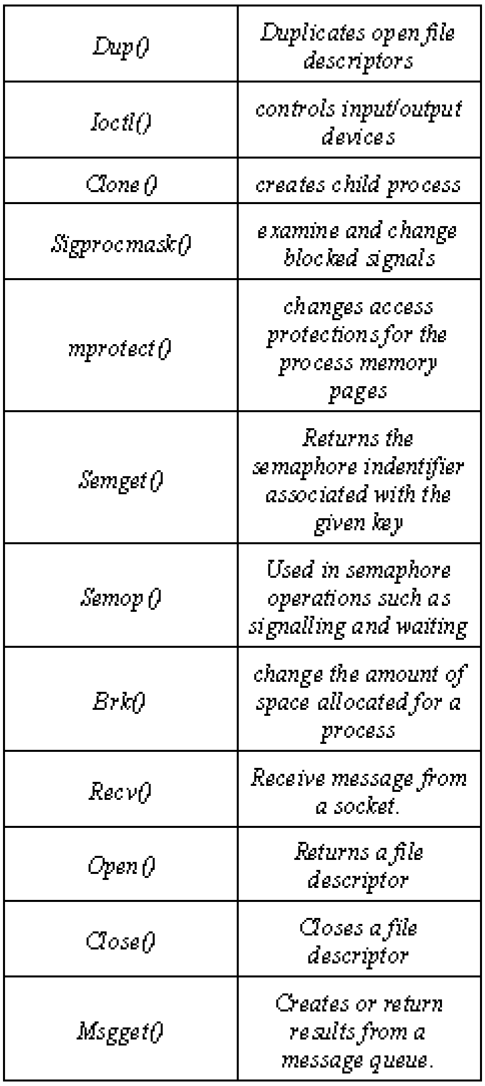


Table 3. Detailed Description of System Calls [18]

System Calls	Definitions
Read()	*reads from file descriptors*
Write()	*writes to a file descriptors*
Fork() and Clone()	*creates a child process*
Lseek()	*repositions read and write offsets*
getpid() and System_224()	*gets process/thread information/ identification*



Dup()	*Duplicates open file descriptors*
Ioctl()	*controls input/output devices*
Clone ()	*creates child process*
Sigprocmask()	*examine and change blocked signals*
mprotect()	*changes access protections for the process memory pages*
Semget()	*Returns the semaphore indentifier associated with the given key*
Semop ()	*Used in semaphore operations such as signalling and waiting*
Brk()	*change the amount of space allocated for a process*
Recv()	*Receive message from a socket.*
Open ()	*Returns a file descriptor*
Close()	*Closes a file descriptor*
Msgget()	*Creates or return results from a message queue.*

Behaviour-based Analysis Module

The behaviour-based Analysis Module is liable for classifying humanoid applications as either benign or malicious. This is done by using machine learning algorithms for the generation of behaviour models of malicious and benign applications. A coaching part, break away the system, is that the one which identifies the behaviour of the applications. This module identifies Android applications into four classifications namely: Virus, Trojan, Spyware, Exploit or Benign.

For the coaching part, behavioural models for each and every sort of Android application are generated by sampling a variety of applications per each classification to run on completely different algorithms. Features of applications extracted from the previous module are translated into an .arff file format for Weka to enable it to collect knowledge. Currently, the module solely generates the accuracy results of the chosen algorithms given feature sets from each sort of malware and of the benign applications.

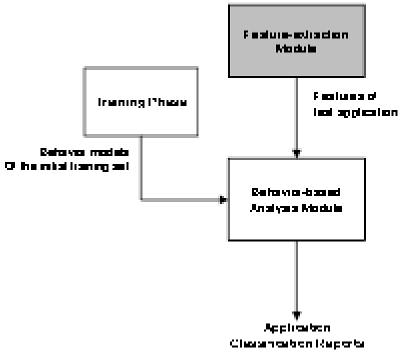

Feature-extraction Module
Training Phase
Features of test application
Behavior models Of the initial training set
Behavior-based Analysis Module
Application Classification Reports

Figure 3. Behavior-based Analysis Module

Conclusion

The system, given the potential to classify unknown applications based from its knowledge, will be accustomed to reason completely different Android applications within the market. With the online crawler at hand, the system has the potential to automatically transfer and classify new applications uploaded to the various different markets. Aside from this, the system has the power to classify malware to different types using behaviour-based analysis. With this at hand, the system can act as antivirals that would easily provide classification results to users. However, skilled systems or completely different classification sources modification classifications from time to time. This happens once a lot of antivirus engines are ready to classify applications as from once the applying was 1st classified or as a result of there are a lot of malware families being identified. With this, there's a clear lack of standards within the classification scheme of applications. This lack of standards contributes to the in utility of classifying malware into completely different classifications aside from simply classifying it as malware. Another issue would be that malware families would have a variety of different malware families which makes it even tougher to differentiate between malware varieties.

Future Work

Further work to be done is that the ability to discover advanced malware attacks like Zero-day attack. Implementation of Behaviour-based analysis with permission-based also can be done to work out malicious Android applications. Administrative program such as an AMDA humanoid Application will allow easier analysis and access of the system.

References

[1] Srivastava, H., Choudhury, T., & Vashisht, V. Counter Strike: Prompt Gaming Time for Android and iPhone. International Journal of Scientific & Engineering Research.

[2] Sharma, R. Study of Latest Emerging Trends on Cyber Security and its challenges to Society. International Journal of Scientific & Engineering Research.

[3] Gharehchopogh, F. S., Abbaspour, F., Tanabi, M., & Maleki, I. Review and Evaluation of Performance Measures in the Mobile Operating Systems.

[4] Gharehchopogh, F. S., Rezaei, R., & Maleki, I. Mobile Cloud Computing: Security Challenges for Threats Reduction. International Journal of Scientific & Engineering Research.

[5] Garg, P., & Sharma, V. Secure Data Storage in Mobile Cloud Computing. International Journal of Scientific & Engineering Research.

[6] Chauhan, A., Mishra, G., & Kumar, G. (2012). Survey on Data mining Techniques in Intrusion Detection. Lap Lambert Academic Publ. International Journal of Scientific & Engineering Research

[7] Ibrahim, H. E., Badr, S. M., & Shaheen, M. A. (2012). Phases vs. Levels using Decision Trees for Intrusion Detection Systems. arXiv preprint arXiv:1208.5997.

[8] Singh, M., Singh, G., & Sharma, S. (2012). Human Protein Function Prediction from Sequence Derived Features using See5. International Journal of Scientific & Engineering Research

[9] Ali, K. B., & Gosain, A. (2012). Predicting the quality of object-oriented multidimensional (OOMD) model of data warehouse using decision tree technique. Int J Sci Eng Res, 1-5.

[10] Kulkarni, M. S. V. (2011). Mining knowledge using Decision Tree Algorithm.International Journal of Scientific and Engineering Research, 2(5), 131-136

[11] Pitale, V. V. K. R. R., & Tajane, K. PERFORMANCE IMPROVEMENT USING INTEGRATION. International Journal of Scientific & Engineering Research

[12] Bhoria, M. P., & Garg, K. An Imperial learning of Data Mining Classification Algorithms in Intrusion Detection Dataset. International Journal of Scientific & Engineering Research

[13] Padmavathi, J. (2012). Logistic regression in feature selection in data mining.International Journal of Scientific & Engineering Research, 3(8).

[14] Bhoria, M. P., & Garg, K. An Imperial learning of Data Mining Classification Algorithms in Intrusion Detection Dataset. International Journal of Scientific & Engineering Research.

[15] Firdhous, M., Hassan, S., & Ghazali, O. A Comprehensive Survey on Quality of Service Implementations in Cloud Computing. International Journal of Scientific & Engineering Research.

[16]T. Vennon, “Threat Analysis of the Android Market,” 2010. [Online]. Available:http://www.globalthreatcenter.com/wp-content/uploads/2010/06/Android-Market-Threat-Analysis-6-22-10-v1.pdf [Accessed: October 30, 2012]

[17] K. Elish, D. Yao, and B. Ryder, “User-Centric Dependence Analysis For Identifying Malicious Mobile Apps,” in Proceedings of the IEEE CS Security and Privacy Workshop, 2012. San Francisco, CA.

[18]T. Isohara, K. Takemori and A. Kubota, “Kernel-Based Behaviour Analysis for Android Malware,” in proceedings of the 2011 Seventh International Conference on Computational Intelligence and Security. Saitama, Japan. 2011.

[19]I. Burguera., U. Zurutuza and S Tehrani, “Crowdroid: Behaviour-Based Malware Detection System for Android,” in Proceedings of the 18th ACM Conference on Computer and Communications Security, 2011. Chicago, IL. 17 October 2011.

[20] T. Blasing and et al, “An Android Application Sandbox System for Suspicious Software Detection,” in Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software, 2010.

[21] Oracle, 2008. “Data Mining Concepts: Regression,” 2008.[Online].Available:http://docs.oracle.com/cd/B28359_01/datamine.111/b28129/regress.htm#DMCON005 [Accessed: October 30, 2012]

[22] B. Sans., “On the Automatic Categorisation of Android Applications,” 2012. [Online]. Available:
http://paginaspersonales.deusto.es/isantos/publications/2012/Sanz_2012_CCNC_Android_Apps_Categorisation.pdf. [Accessed: October 25, 2012]

[23] A. Shabtai and C. Glezer, “ “Andromaly” a behavioural malware detection framework for android devices,” 2010. [Online]. Available: http://posgrado.escom.ipn.mx/biblioteca/%E2%80%9CAndromaly%E2%80%9D%20a%20behavioural%20malware%20detection.pdf

[24] P. Flach and N. Lachiche, "Naïve Bayesian Classification of Structured Data," [Online]. Available: http://www.cs.bris.ac.uk/~flach/papers/mlj04-1BC-final2.pdf [Accessed: November 1, 2012]

[25] H. Zhang, "The Optimality of Naïve Bayes," [Online]. Available: http://courses.ischool.berkeley.edu/i290-dm/s11/SECURE/Optimality_of_Naive_Bayes.pdf [Accessed: November 1, 2012]

[26] S. Kotsiantis, I. D. Zaharakis and P. E. Pintelas, "Supervised Machine Learning: A Review of Classification and Combining Techniques," [Online]. Available: www.cs.bham.ac.uk/~pxt/IDA/class_rev.pdf [Accessed: November 2, 2012]

[27] J. Chan, K. Chan, and A. Yeh, "Detecting the Nature of Change in an Urban Environment: A Comparison of Machine Learning Algorithms," American Society for Photogrammetry and Remote Sensing, , vol. 67, No. 2, pp. 213-225, February 2001.

[28] L. Breiman and A. Cutler, "Random Forests," [Online]. Available: http://stat-www.berkeley.edu/users/breiman/RandomForests/cc_home.htm#intro [Accessed: November 3, 2012]

[29] L. Moutinho and G.D. Hutcheson, "Dictionary of Quantitative Methods in Management," [Online]. Available: http://www.research-training.net/addedfiles/READING/MNLmodelChapter.pdf [Accessed: November 4, 2012]

[30] I. Rish, "An Emprical Study of the naïve Bayes Classifer," [Online]. Available: www.cc.gatech.edu/~isbell/reading/papers/Rish.pdf [Accessed: November 5, 2012]

[31] Weka, 2008, "Weka: Primer," 2012. [Online]. Available: http://weka.wikispaces.com/Primer [Accessed: November 5, 2012]

[32]. J. Tiedemann, "Interpreting Weka Output," [Online]. Available: http://www.let.rug.nl/tiedeman/ml06/InterpretingWekaOutput [Accessed: November 5, 2012]

[33] J. He, "Linux System Call Quick Reference," [Online]. Available: http://www.digilife.be/quickreferences/qrc/linux%20system%20call%20quick%20reference.pdf [Accessed: November 7, 2012]

[34] T.Borovicka, M.Jirina Jr., P. Kordik and M. Jirina, "Selecting Representative Data Sets," [Online]. Available: http://cdn.intechopen.com/pdfs/39037/InTech-Selecting_representative_data_sets.pdf [Accessed: December 2, 2012]

[35] ESET Labs, 2013. "Trends for 2013: Astounding growth of mobile malware.," [Online]. Available: http://go.eset.com/us/resources/white-papers/Trends_for_2013_preview.pdf