# A3: Automatic Analysis of Android Malware

Luoshi Zhang[1], Yan Niu[2], Xiao Wu[2], Zhaoguo Wang[3], Yibo Xue[4, *]

[1] Computer Science & Technology College, Harbin Univ. of Sci. & Tech, Harbin, China

[2] CNCERT/CC China, Beijing, 100029, China

[3] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

[4]Tsinghua National Lab for Information Sci. & Tech., Beijing, China

luoshi.zh@gmail.com, niuyan@cert.org.cn, wuxiao@cert.org.cn, 13B303001@hit.edu.cn, yiboxue@tsinghua.edu.cn

***Abstract*—Recently, Android malware is spreading rapidly. Although static or dynamic analysis techniques for detecting Android malware can provide a comprehensive view, it is still subjected to time consuming and high cost in deployment and manual efforts. In this paper, we propose A3, an Automatic Analysis of Android malware, to detect Android malware automatically. Unlike traditional reverse-engineering methods, A3 looks for Command & Control (C&C) Server), monitors the sensitive API invoking, and declares I*ntent-filter* automatically. Then it constructs the relationship graph of the function calls and key contexts or paths for detecting Android malware. The experimental results show that A3 can do an effective automatic analysis for Android malware and has a good performance.**

***Keywords-Android, malware, automatic anlaysis, reverse-enginnering***

## I. INTRODUCTION

With the development of scientific and technological progress in recent years, the number of smartphones is explosively growing. Meanwhile, Google's Android overtook others to become the most popular mobile operating system because of its openness. Unfortunately, this openness of Android comes with the growing prevalence of malware. Unlike the Apple's App Store which has strict security checks manually by software security experts, Android takes passive mechanism that allows anyone to publish or download applications on the Android market (e.g. Google play or other application markets). It has been released by Kaspersky Lab[1] that "among all mobile malware, the share of Android-based malware is higher than 46% and still growing rapidly" and " the share of particularly Android malicious apps trying to steal personal data went up to 34%". Given the rampant growth of Android malware, there is a pressing need to effectively detect and analyze them.

Currently, the industry is more concerned with the detail of malicious activities and focusing on analyzing the harmful process manually when facing a new malware. However, the academia tends to focus on how to quickly discover the Android malware from all applications by automatic mechanism, but ignoring the analysis of malicious behavior.

In practice, both are needed, but it requires great efforts. So how to automatically analyze Android malware is urgently required. It not only reduces the cost but also improves the efficiency.

In this paper, we propose A3, an Automatic Analysis of Android malware, to detect Android malware automatically.

A3 firstly considers about the build-in IPs or URLs, which are used for transferring data. Malware are usually for collecting user's privacy information or monetary benefits. So malware has to embed some fixed address as Command and Control Server (C&C Server) to receive data; then tracks the function calls or variable reference relationship that invoking the IPs or URLs automatically; Finally, it checks the *Manifests* file to judge the permission and the *inter-filter* of class that contained above build-in IP/URLs and sensitive API calls. After that, A3 can detect Android malware automatically.

The contributions of this paper are as follows.

(1) Analyze the typical malicious behaviors of Android malware and propose the basic idea for automatic detecting;

(2) Propose A3, an Automatic Analysis of Android malware, to detect Android malwares automatically. We introduce the framework and work procedure of A3 in details.

(3) Demonstrate some experiments and analysis of several Android malwares to verify the effectiveness of A3.

The remainder of the paper is organized as follows. In section II, we present an overview of Android malware, focus on the Android operating system and state-of-art of Android malware detection. In section III, we propose the concept of A3 and illustrate its advantages and limitations. Then in section IV, we describe the static analyze architecture. The section V presents the evaluation of proposed architecture and approach. We conclude this paper in section VI.

## II. RELATED WORK

Android malware is usually written in the Java language and deployed as Android Packages archive (APKs) in Android platform. The goal of malware is to gain access to a device for stealing data, damaging, or annoying the user, etc[2]. The criminals tempt the user to install the malicious software and execute some malicious activity without user's awareness. Malware results in threats to the affected user with respect to user information or application safety. These threats include Trojans, worms, botnets, viruses, etc.

*Corresponding author. Tel: +86 010 62772393. E-mail: yiboxue@tsinghua.edu.cn

 

There are two methods about automatic detection of Android malware. One is Misuse Detection, which is a signature-based approach by matching the rules and policies. Papers[3][4] applied this mechanism to detect Android malware. The advantage of this approach is that it has high accuracy. But it is invalid to the new Android malware [5]. The other one is Anomaly Detection, which is different from misuse detection. It usually applies machine learning algorithms for obtaining the known malware behavior and predicting the unknown or new Android malware [6][7]. But it sometimes causes high false positive.

Meanwhile, the analyzing methods of Android malware include dynamic analysis and static analysis. The dynamic analysis method continuously monitors the various running situation of the malware (such as reading and writing operations, API calls, power consumption, incoming and outgoing network information, and so on) and then constructs some models for detecting malware[8]. However, it is subjected to high cost in deployment and manual efforts [5]. The static analysis method only considers the contents of every application after decompiling the code [4]. This method can reduce the cost and improve the performance, but it will face the great difficulty when meeting the code obfuscation technique[5]. So how to use the both advantages of the dynamic analysis and static analysis to design an automatic detecting tool is urgent and important for identifying Android malwares.

## III. THE BASIC IDEA OF A3

### A. Typical malicious behavior

Unlike Trojans or viruses on PC which is mainly controllable and destructive, a report[2] shows that two most common malicious activities on the Android platform are collecting user information(61%) and sending premium-rate SMS messages(52%).

In order to reach this purpose, the malwares usually have the following typical malicious behaviors:

- Build-in IP/URLs. Android malwares have to secretly collect some information and send them to a designated server. Generally speaking, this designated server (e.g. IP or URL) is called Command & Control Server (C&C Server) and usually be embeded into the code.
- Sensitive API invoking. Some sensitive API functions of Android SDK are invoked for help to steal user's privacy information. These functions are mainly related to accessing the contacts (stored both on the phone and the SIM card), call logs, SMS message, Geo-location and Phone data (e.g. phone number, OS version, phone model, SDK version) and so on.
- "I*ntent-filter*" declarations. Due to the actions of accessing and sending the privacy data is prohibited without user's permission, the malicious activity is usually executed in private. So there are some trigger mechanisms to help launching the malicious behavior and these triggers must be declared in the "*intent-filter*" of *manifests* file.

Every independent behavior of the above is normal. But if all of them are used together in a series, there may appear a malicious behavior.

In summary, the behaviors of Android malware can be concluded as following: (1) Automatically connecting to the network via WIFI or 3G; (2) Automatically collecting user's privacy data via sensitive APIs invoking in private; (3) Automatically sending the collected data to C&C server without the user' permission.

### B The Basci Idea of A3

It is difficult and time-consuming to fully reverse-engineering a malware, but catching the above three steps is easy. Thus it is possible to automatically analyze an Android malware by focusing on the above three key points.

The key actions include followings:

- Extracting IP/URLs from the decompiled code automatically by using string or regular expression matching. These IP/URLs are suspected to be C&C server.
- Monitoring the found IP/ URL, setting as a starter, then tracing the relationship of all the function calls until the function equals to system functions(e.g. *onStart* or *onCreat*), thus constructing the relationship graph of the function calls.
- Looking for the sensitive API calls from the graph, such as *TelephonyManager->getDeviceID（access to communications authority）*, verifying the "*inter-filter*" of the involved class from *manifests* file.
- Checking the C&C server and testifying what is the malicious behavior.

We take *Instagram*[9] as an example. Instagram is an online photo-sharing, video-sharing and social networking service that enables users to share pictures and videos on a variety of social networking services, such as Facebook, Twitter, Tumblr and Flickr. In April 2012, it was announced that over 30 million accounts were set up on Instagram.[10]

The malware "Instagram" has the same name of Instagram to cheat users. Till June 2013, the number of cheated users has reach up 60 thousands.

We analyzed the actual process of the malicious Instagram manually, its actual procedure is shown in Fig.1. Fig.1 displays the relationship of the function calls and the sensitive API invoking.

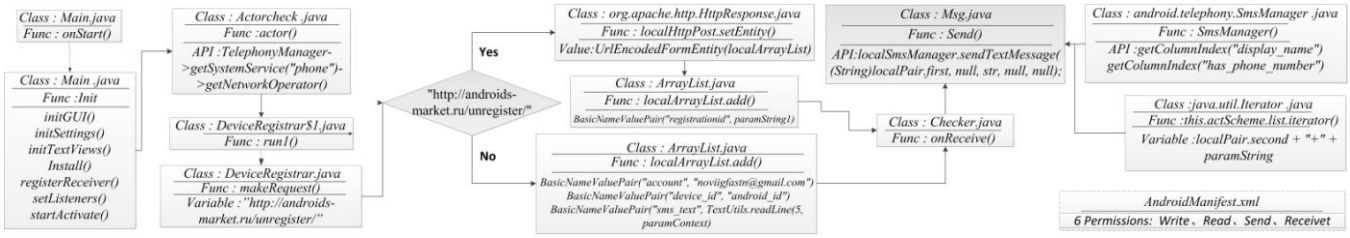


Figure 1. The Actual Procedure of Instagram.

We represent functions as nodes, function calls as edges respectively. And abstract their relationship as graph as shown in Fig.2, which is obtained by Gephi[14] automatically.

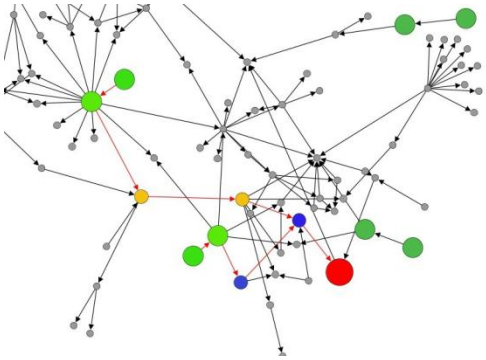

Figure 2. The Relationship Graph of Instagram.

We use GEXF format to represent the function call relationship. In this format, we can get the detailed information about each node: Type of the method (activity, service, receiver) 、 Class name 、 Method name 、 Descriptor 、 Permissions 、 Permissions level (normal, signature, dangerous)、 Android API entropy、 Java API entropy、Dynamic code

Moreover, we colorize the specific nodes with different color:

- Activity/Service/Receiver nodes use green/cyan/ purple respectively.
- Risk (IP, URL, privacy, phone, SMS, money) nodes use color from blue to red.
- Dynamic code nodes use black.

The correlation analysis of manual and automatic analysis is revealed in Fig.3. In Fig.3 the Red node represents the function that contains IP/URL, the Blue node represents the function that contains sensitive API invoking, and the Green node represents the key system function( such as "*onStart*") and the Yellow node means excessive function. The Red line represents the relationship of the function calls.

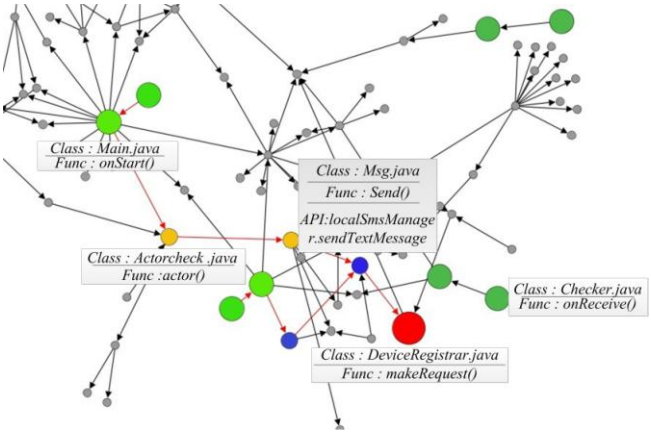


Figure 3. The Correlation Analysis of Manual and Automatic Analysis

If there are some "lines" matching the typical malicious behavior series, it can verify that this software is malware. And looking for such "lines" in graph can be implemented automatically.

## IV. AUTOMATIC ANALYSIS PROCEDURE

According to the above basic ideas, we propose an automatic analysis framework of A3, as shown in Fig.4.

This framework includes decompiling engine, C&C analysis engine, graph analysis engine and API analysis engine. During the process, it mainly focuses on detecting some particular contents, such as the build-in URL or IP string, the relationship of function calls, the sensitive API invoking, the *inter-filter* of class; After that, it constructs the relationship graph about suspicious malicious behavior; Finally identifying and verifying the malicious behaviors of an Android malware.

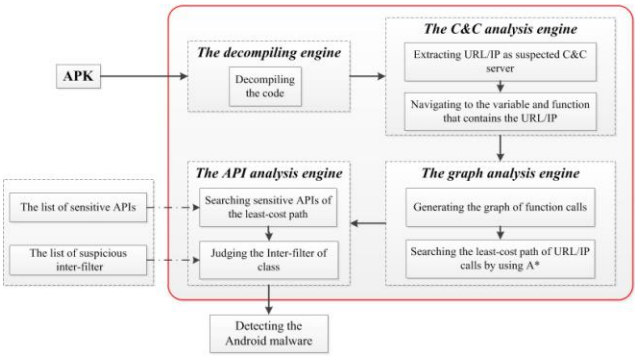


Figure 4. The Framework of A3 System.

The detailed analyzing process is as follows.

1) Decompiling engine is an automated decompiling APK program. We get the manifest file and disassembly codes (*smali* files) by using the *apktool*[12] and *dex2jar*[13].

2) Extracting the IPs and URLs from the decompiled code automatically by string or regular expression matching, and navigating to the variable of URL/IP and function that contains the URL/IP.

3) Based on lexical and syntax analysis, tracing the relationship of the function calls layer-by-layer and automatically constructing the relationship graph about IP or URL. The graph is constituted of different elements: classes, methods, local variables and global variables and so on.

In order to enhance the malware modeling capability and analyze the relevance of between IPs/URLs and sensitive APIs, we adopt A*[14] algorithm in our method .

A* algorithms uses a best-first searching to find a least-cost path from a given initial node to one or more goal node. An example of the route of A3 malware modeling is shown in Fig.5.

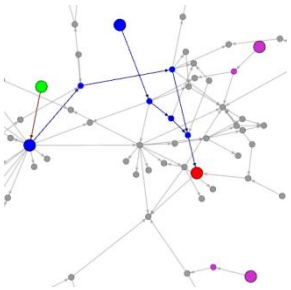

Figure 5. The Route of A3 Malware Modeling.

A3 chooses the key system function, such a "*onCreate*" (Green nodes) or "*onStart*" (Purple nodes), as initial nodes, and chooses risk nodes include IPs, URLs, privacy, phone, sms, and money information.

Then, execute A* algorithm for looking for any path from initial nodes to risk nodes, and extracting a shortest path for estimating the malware behaviors.

Finally, comparing the sensitive API calls with the relationship graph, obtaining the suspected information and the sequence between the sensitive API calls. Meanwhile, further checking the *intent-filter* of class from *Android Manifest* file and judging the launching model of class.

The A3 by using A* algorithm can analyze the malicious behavior automatically and detect the Android malware.

Some of the sensitive API calls are given in Table I.

TABLE I. MOST POPULAR SENSITIVE API CALLS

| ***Type*** | ***Information*** |
|---|---|
| Content | Network information |
| | Device information |
| | SMS message |
| | Contact |
| | GPRS information |
| | Call logs |
| | Photo |
| | Brower history |
| | Note information |
| Function | HTTP socket |
| | Data encode |
| | Activity and Service function |

## V. EXPERIMENTS AND EVALUATIONS

We conduct some experiments to verify and evaluate the effectiveness of our proposed approach.

We randomly collect three Android malware: ***Alsalah, sp_ntm, instagram*** from "*Contagio mobile*"[16] site. In addition, we compare the analysis results with *Androguard*[15] published at Blackhat 2011, which is one well-known Android malware detection tools.

Based on the statistics results of Androgruad about the three Android malware are presented in Table II. The number in Table II shows the frequency of occurrence of sensitive API invoking.

TABLE II. THE ANALYSIS RESULTS OF ANDROGUARD

| ***Application*** | ***Result*** |
|---|---|
| Alsalah | PERM {'PRIVACY': 17, 'NORMAL': 26, 'MONEY': 4, 'INTERNET': 3, 'SMS': 5, 'DANGEROUS': 39, 'SIGNATUREORSYSTEM': 20, 'CALL': 2, 'SIGNATURE': 20, 'GPS': 4} |
| Sp_ntm | PERM {'PRIVACY': 2, 'NORMAL': 3, 'MONEY': 0, 'INTERNET': 1, 'SMS': 0, 'DANGEROUS': 2, 'SIGNATUREORSYSTEM': 0, 'CALL': 0, 'SIGNATURE': 0, 'GPS': 1} |
| instagram | PERM {'PRIVACY': 3, 'NORMAL': 1, 'MONEY': 1, 'INTERNET': 1, 'SMS': 3, 'DANGEROUS': 6, 'SIGNATUREORSYSTEM': 0, 'CALL': 0, 'SIGNATURE': 1, 'GPS': 0} |

The analysis results of A3 about ***Alsalah*** are shown in Fig.6, We can find out that there are obvious relationships between system launching function and the sensitive APIs and the URL/IP. Meanwhile, the length of relationship of malicious behavior is short and isolated, that is because the malicious code is usually embedded in the normal application and has fewer interactions with others.

The correlation analysis of ***Alsalah,*** can be obtained manually, as shown in Fig.6, they can also verify the analysis results of A3.

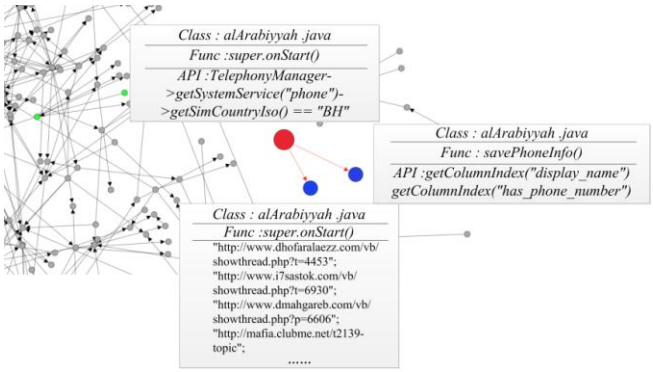


Figure 6. The Correlation Analysis of ***Alsalah***

The correlation analysis of ***sp_ntm*** is shown in Fig.7, it is enough for detecting a malware.

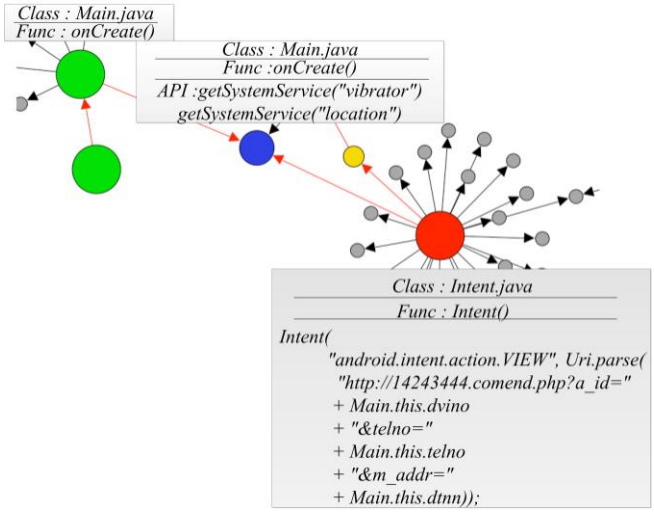


Figure 7. The Correlation Analysis of ***sp_ntm***

## VI. CONCLUSION AND FUTURE WORK

There are emerging a lot of Android malwares on Android platform. They are very harmful to users. So how to detect Android malwares quickly and effectively is urgently required and facing a big challenge.

Traditional manual reverse-engineering is very time-consuming, so we proposed A3: an Automatic Analysis of Android malware to detect Android malwares automatically.

A3 focuses on the sensitive API calls around build-in URL or IP and communication contents and invocations, so it has universal process to detect all kinds of malwares. In addition, A3 has the common framework for analyzing the Android malicious behavior and does not need to extract the feature or train model for malwares, so it is faster than other methods based on signatures identification or machine learning algorithm.

Of course, when developers make greater use of code encryption and code obfuscation technology, A3 also has some limitations when facing with the sophisticated malwares. We will dig further for them in the future.

## ACKNOWLEDGMENT

This work was supported by the National Key Technology R&D Program of China under Grant No.2012BAH46B04

## REFFERENCES

[1] Number of the Week: at Least 34% of Android Malware Is Stealing Your Data. http://www.kaspersky.com/about/news/virus/2011/Number_of_the_Week_at_Least_34_of_Android_Malware_Is_Stealing_Your_Data.

[2] Felt, Adrienne Porter, et al. A survey of mobile malware in the wild. In Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices (SPSM'11), 2011.

[3] H. Kim, J. Smith, and K. G. Shin. Detecting energy-greedy anomalies and mobile malware variants. In Proceedings of the 6th international conference on Mobile systems, applications, and services (MobiSys'08), 2008.

[4] Y. Zhou, Z. Wang, W. Zhou, et al. Hey, you, get off of my market: Detecting malicious apps in official and alternative Android markets. In Proceedings of the 19th Annual Network & Distributed System Security Symposium, 2012.

[5] Wu, Dong Jie, et al. DroidMat: Android Malware Detection through Manifest and API Calls Tracing. Information Security (Asia JCIS), 2012.

[6] A. Bose, X. Hu, K. G. Shin, et al. Behavioral detection of malware on mobile handsets. In Proceedings of the 6th international conference on Mobile systems, applications, and services (MobiSys'08), 2008.

[7] Shabtai, Asaf, et al. "Andromaly": a behavioral malware detection framework for android devices. Journal of Intelligent Information Systems 38.1 (2012): 161-190.

[8] Zhao, Min, et al. AntiMalDroid: An Efficient SVM-Based Malware Detection Framework for Android. Information Computing and Applications. Springer Berlin Heidelberg, 2011. 158-166.

[9] http://en.wikipedia.org/wiki/Instagram Retrieved April 9, 2012.

[10] "Instagram for Android – Available Now". Instagram Blog. Facebook. April 3, 2012. Retrieved April 9, 2012.

[11] http://gephi.org/

[12] Android-apktool. http://code.google.com/p/android- apktool/.

[13] Android-dex2jar. https://code.google.com/p/dex2jar/.

[14] A*. http://en.wikipedia.org/wiki/A*

[15] Androguard. http://code.google.com/p/androguard/.

[16] Contagio mobile. http://contagiominidump.blogspot.com/.