

Malicious Code Search in Android Applications by Statistical Analysis Method

Anton N. Mironenko
mironim84@mail.ru

Dostoevsky Omsk State University, OmSU ,Omsk, Russia

Аннотация

The paper proposes a classification of malicious effects on a mobile device and discusses methods of analyzing software that presumably contains malicious code. In addition, a method of protecting mobile devices running the Android operating system from the unwanted impact of software containing malicious code is offered. The method is based on static analysis of the source code of suspicious software. The main idea of the method is that there is a sequential computation of the complexity metrics of the suspicious application code fragments and the comparison of the results with the values from the malware metrics database.

Introduction

The theme of protection of mobile devices from viruses is actual. In the comments to article 273 of the Criminal Code of the Russian Federation «the creation, use and distribution of malicious software for computers» is defined by the definition of computer virus. This term we will use in this work. Computer Virus—is a program that can reproduce itself in several instances, modify (change) the program to which it has joined, and thereby disrupt its normal functioning [1]. The assumption about the relevance of the topic of protection of mobile devices against viruses is based on the fact that currently mobile phones are almost every person. As a rule, the mobile device stores a huge amount of confidential information, such as phone numbers, notes containing passwords, etc. The most popular platform, according to the information of the site NetMarketShare (fig. 1), is Android operating system, almost 67%. All devices operate on the basis of this operating system. According to the article [2], the databases of modern anti-viruses contain more than 10 million images of malicious Android applications, however, most of them—are copies of original viruses. After analyzing the source code of the original viruses, you can see that they are all a combination of a small number of unique functional blocks. This is because viruses for mobile devices typically perform the following similar tasks:

- sending SMS messages to paid numbers;
- acquisition of confidential information of the user (contacts, texts of messages, data from memory cards, etc.);
- collecting information about device;

Copyright © by the paper's authors. Copying permitted for private and academic purposes.

In: Sergey V. Belim, Nadezda F. Bogachenko (eds.): Proceedings of the Workshop on Data, Modeling and Security 2018 (DMS-2018), Omsk, Russia, October 2018, published at <http://ceur-ws.org>

- obtaining administrator privileges to install software without the user's knowledge;
- tracking and interception of logins, passwords and bank card data.

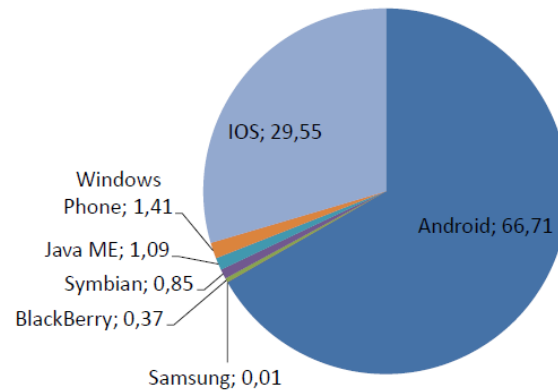


Рис. 1: Mobile operating system statistics from NetMarketShare in February 2017

In work [3] the task of increasing the effectiveness of malware detection in the operating system for mobile devices (for example, Android) is considered. To achieve this goal, an analysis was made of the security of the Android operating system and the formalization of samples of malicious programs in order to identify the characteristics inherent in their behavior. On the basis of the information received, an experimental sample has been developed that includes vectors describing the behavioral character of two types of programs: safe ok and malicious virus. As a result of the research experiments, a classification method is chosen that performs the classification of the proposed sample with the highest accuracy. The task of increasing the efficiency of detection of malicious programs is solved using the developed technique based on the support vector machine and the fuzzy logic apparatus. This technique is implemented as a research prototype of the malware detection system.

All this, confirms the relevance of protecting applications of the Android operating system from viruses.

1 Task statement

In the work of [4] the classification of methods of analysis of the software presumably containing malicious code is given. Two types of analysis are allocated: dynamic and static. Dynamic methods examine the behavior of the program, namely, its interaction with the system, the data collected, the network connections being established, etc. As a rule, the program uses the following information about it and its behavior for analysis:

- the hash sum of the APK file (MD5, SHA—1 (256));
- information about data sent and received over the network;
- information about the reading and writing of files;
- information about the services that were launched;
- information about the collected and sent user data;
- information about the permissions received by the application;
- information about SMS messages that have been sent and carried out calls.

During the static analysis of the program, its code is examined. The main task of such an analysis is to search for a piece of code (perhaps all) that performs a malicious effect. Methods of static analysis of source code of programs, in contrast to dynamic testing, avoid the problem of curse of the dimension of the test data area and achieve a greater degree of automation of checks for the presence of security defects, based on their design features. The general scheme of carrying out static analysis can be represented as follows in the figure 2.

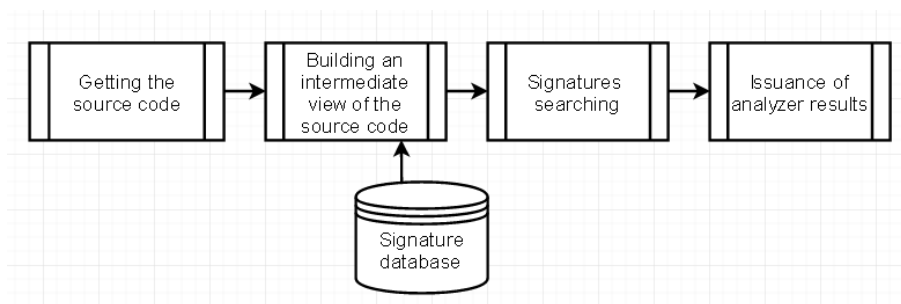


Рис. 2: General scheme of static analysis

The idea of using a mathematical apparatus to detect malicious code is not new. In [5], a new automaton model is presented, which can become the basis for the development of antivirus software. The approach described in the paper is based on the equivalence check technique in algebraic models of sequential programs.

In work [6] the possibility of using signature analysis for detecting security flaws of the code is examined.

In work [7], the method of search of borrowings in the program code with use of complexity metrics is offered. The algorithm of the proposed method consists of two stages: preparation of the program code for further work with it and determining whether the program is plagiarism. Data preparation: The program is represented as a point on an n -dimensional space, each i -coordinate of which is a certain quantitative characteristic. To calculate the coordinates, it was suggested to use the software quality estimation metrics and the number of cycle operators used in the program. Checking the program:

1. the program suspected of plagiarism is represented as a point $A(x_{a1}, x_{a2}, \dots, x_{an})$, where x_{ai} is the value of the quantitative metric;
2. the program that is original is also represented by the point $B(x_{b1}, x_{b2}, \dots, x_{bn})$, where x_{bi} is the value of the quantitative metric;
3. place the points A and B on the n -dimensional space;
4. consider the distance between the points A and B, if it is sufficiently small, then it is considered that the program did not pass the test.

In addition, the results of the experiment confirming the possibility of using this algorithm. In this paper, it is proposed to solve the problem of searching for malicious inserts, taking as a basis the idea proposed and tested in the work [7].

2 Theory

The proposed method can be divided into two stages:

- preparatory stage — « training »;
- working.

At the first stage, the database of malware templates is formed. To do this, you need a large enough database of virus samples, for example [2] or projects «theZoo», «Malware» and the like, which can be found in the «Github» repository. Further from the source codes we calculate their complexity metrics. Thus we get a set of vectors that will be templates for further search of malicious insertion in the code of the program. The coordinates of an individual vector are the numerical values of the code metrics of a particular type of virus. The second stage of the proposed method can be described by the following algorithm:

1. take N lines of the code to be checked and calculate its complexity metrics (exactly the same as when we formed the database of virus pattern templates) — we get the vector;
2. compare the resulting vector with all, alternately, vectors from the base of the patterns of virus insertions;
3. if the vectors coincide — the insertion is detected, if not — return to «step 1»;

4. increase N by 1 and repeat «steps 2 — 3».

The initial value of N can be equal to 1, but it is better to take equal to the number of lines of the code of the shortest insert from the database.

Schematically, the algorithm can be represented as the following scheme is shown in the figure 3.

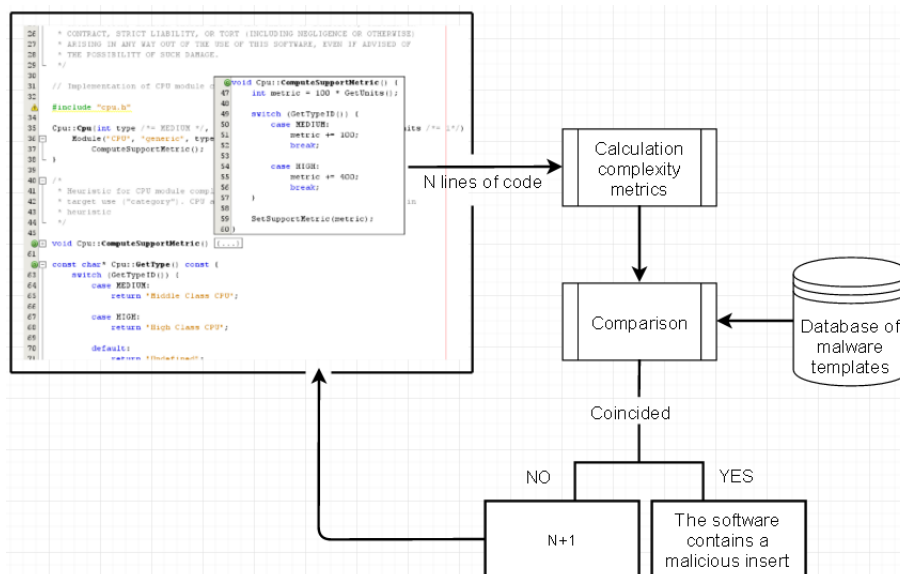


Рис. 3: The algorithm for detecting malicious inserts

Conclusions

The work offered a static method for detecting malicious insertions in the code of Android programs. The proposed method is based on the use of code complexity metrics. The possibility of using complexity metrics to compare two programs was tested earlier in other works, for example, [7], thus we can say that their application is also possible to solve the task. Nevertheless, it is necessary to approbate this algorithm.

Список литературы

- [1] Ст. 272, Уголовный кодекс Российской Федерации от 13.06.1996 N 63-ФЗ (ред. от 19.02.2018). URL: <http://www.consultant.ru/cons/cgi/online.cgi?req=doc&base=LAW&n=291258&dst=976>.
- [2] Обзор фрагментов кода самых популярных типов вирусов. URL: <https://xakep.ru/2014/09/18/android-malware-source/>.
- [3] S.V. Zhernakov, G.N. Gavrillov. Sistema obnaruzheniya vredonosnyh programm v operacionnoj sisteme Android. *Vestnik UGATU*, 2(72), 2016. URL: <https://cyberleninka.ru/article/n/sistema-obnaruzheniya-vredonosnyh-programm-v-operatsionnoy-sisteme-android>.
- [4] O.V. Skulkin, I.Ju. Mihajlov, A.I. Makarov. Principy obnaruzheniya vredonosnyh prilozhenij dlja Android. URL: https://www.anti-malware.ru/analytics/Threats_Analysis/principles_detection_malicious_applications_Android#part2.
- [5] R.I. Podlovchenko, N.N. Kuzyurin, V.S. Shcherbina, V.A. Zakharov. Using algebraic models of programs for detecting metamorphic malwares. *Fundamentalnaya i prikladnaya matematika*, 15(5):181–198, 2009.
- [6] A.S. Markov, A.A. Fadin. Problemy avtomatizacii vyjavlenija programmnyh zakladok i defektov bezopasnosti koda metodom staticheskogo signaturnogo analiza. URL: <https://npo-echelon.ru/doc/0717.pdf>.

- [7] A.N. Mironenko. Metod opredelenija zaimstvovanij v programmnom kode s ispol'zovaniem ego metrik slozhnosti. *Informacionnaja bezopasnost' i zashhita personal'nyh dannyh. Problemy i puti ih reshenija. Materialy VIII Vserossijskoj nauchno-prakticheskoj konferencii*, 2016, pp. 87–89.

Поиск вредоносного кода в приложениях операционной системы Android методом статистического анализа

А.Н. Мироненко

В работе предлагается классификация вредоносных воздействий на мобильное устройство и рассматриваются методы анализа программного обеспечения, предположительно содержащего вредоносный код. Кроме того, предлагается метод защиты мобильных устройств, работающих по управлению операционной системы Android, от нежелательного воздействия программного обеспечения содержащего вредоносный код. Метод основан на статическом анализе исходный кода подозрительного программного обеспечения. Основная идея метода в том, что происходит последовательное вычисление метрик сложности участков кода подозрительного приложения и сравнение полученных результатов со значениями из базы метрик вредоносных программ.