MALWARE BEHAVIOR ANALYSIS USING BINARY CODE TRACKING

*Abdumuminov Abdurafiq Abdurashidovich
**Ibragimov Jalaliddin Obidjon o'g'li
*** Shoraimov Khusanboy Uktamboyevich

*Republican center for management of telecommunications networks of Uzbekistan. SUE.
** Teacher of the Department, “Systematic and Practical Programming”, Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi, UZBEKISTAN
*** Teacher of the Department, “Systematic and Practical Programming”, Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi, UZBEKISTAN

Abstract—*The rapidly increasing malware goes beyond personal security threats and has a negative effect on criminal society. To prevent these security threats, many anti-virus vendors and analysts are starving to more efficiently distinguish malicious behavior. In order to contribute to this, in this study, we try to detect malicious behavior by tracking the execution flow of binary code. Our method of tracking the execution flow of the binary code utilizing the BFS(Breath-First Search)algorithm advances static analysis based on binary code, but it can be a method combining the advantage of static analysis and the advantage of dynamic analysis. In addition to visualizing malicious behavior as a graph image based on APIs, it is possible to analyze more obviously malicious behavior.*

Keywords—Malware analysis; Binary Code, Visulization;

I. INTRODUCTION

The rapid development of hacking techniques and computer networks has increased the threat from various malware and its variants, which is also exponentially increasing. Not only malware attack forms are very diversified, and variant malware produced by modulating traditional malware trends to show an explosive increase rate, it poses various security threats such as loss of data, leakage of important information such as personal information, financial, breakage of the system, destruction of the IT infrastructure.

In order to prevent such security threats, malware analyzer and anti-virus vendors are starving to efficiently detect malware. In general, malware analysts are analyzing malware code using static analysis and dynamic analysis. Static analysis can be deeply analyzed as a method to analyze the malware’s structure and binary pattern to grasp malicious behavior, but it not be suitable if a techniques such as code obfuscation is applied. Dynamic analysis, on the other hands, is not suitable for malware analysis that acts on specific acts such as time triggers, as a way to actually execute malware to grasp malicious behavior.

In this paper, we will analyze malicious behavior by tracking the execute path flow on a binary code. We try to complement the drawbacks of dynamic analysis while avoiding the drawbacks of static analysis by tracking the flow of execution on binary code using the BFS (Breath-First Search) algorithm.

The composition of the paper is as follows. Section 2 introduces existing related research for detection of malicious behavior. Section 3 presents the method proposed in this paper, Section 4 shows the results of experiment, and finally in Section 5, conclusion and the direction of future studies are presented.

II. RELATED WORK

Static analysis can be analyzed deeply in order to reverse engineer and analyze at the code level without actually execution malware. Therefore, computer system can be protected from malware, but if techniques such as obfuscation are used for malware, it is an incompatible method because it analyze on the binary code.

Previous malware static analysis based binary code studies ware done taking advantage of the characteristics of attributes in the binary code. These studies generally use statistical methods to detect malicious behavior based the value of the result of analyzing the difference between malware code and normal code. Typically, it is a method to classify differences by outputting statistics of opcode or string in binary code. String or opcode extracted by the above method may be used as a basis for detecting malicious behavior by slicing a character string to a certain standard by a method like n- gram. It standardizes malware files to file DNA, and shows the property values of in the graph.

Recently, in order to visualize malware to analyze malicious behavior more efficiently, API sequence or statistics of all string in the binary code are processed and shown in the spectrum image, also shown in the map image of the file. These methods have the advantages of the following computational complexity and high speed with less data to be kept with a simple statistical method and less information but difficult to respond in real time and are based on simple statistics Therefore, it is not easy to classify various malignant codes. For this reason, recently studies are under way to combine various algorithms with statistical attributes.

III. Proposed Method

A. Static analysis The method proposed in the present study is based on static analysis and uses IDA as a reverse engineering tool. In addition, the method analyzes Windows PE files as malware. IDA reverse engineers PE files, that is, malware to create assembly files as the results of static analysis. In assembly files, algorithms such as conditional statements included in actual malware and branches according to conditional statements are expressed as assembly instructions. In addition, in the present study, the APIs used by malware are used as basic analysis data. Those APIs that are used in PE files are stored in IATs (Import Address Table). IATs for PE files can be easily obtained using reverse engineering tools such as IDA.

B. PE file's Binary Code PE file's programs start at entry points. IDA facilitates finding entry points. In the present study, analysis starts at the entry point.

V. Conclusion

In this paper, we proposed a method to track execution flow on the binary code and detect malicious behavior. We also make API-based graph image, it can analyze malicious behavior more efficiently. Such a graph image can be extended to various researches which are the same as malware similarity and malware classification in the future.

The method proposed by this paper can grasp the behavior of malware such as time trigger which is not suitable for dynamic analysis, tracks the flow of execution, even if techniques such as obfuscation is applied, it can be avoided.

In future research, we will try to visualize more systematically based on the method proposed in this study. In addition we try to develop an automated system that classifies malware for each behavior and calculates similarity.

References

1. Moser, Andreas, Christopher Kruegel, and Engin Kirda. "Limits of static analysis for malware detection." *Computer security applications conference, 2017. ACSAC 2017. Twenty-third annual.* IEEE, 2017.
2. Ding, Yuxin, et al. "Control flow-based opcode behavior analysis for Malware detection." *Computers & Security* 44 (2018): 65-74.
3. Santos, Igor, et al. "Idea: Opcode-sequence-based malware *detection." International Symposium on Engineering Secure Software and Systems. Springer Berlin Heidelberg, 2019.*
4. Santos, Igor, et al. "N-grams-based File Signatures for Malware Detection." *ICEIS (2)* 9 (2019): 317-320.

5.Choi, Young Han, et al. "Toward extracting malware features for classification using static and dynamic analysis." *Computing and Networking Technology (ICCNT), 2019 8th International Conference on.* IEEE, 2019.
6.Askarov B., Yuldashev A., Sultanova D. SYNERGY METHOD FOR SOLVING SOME PROBLEMS OF EDUCATION //ASJ. – 2021. – Т. 2. – №. 56. – С. 15-19.
7.Sultanova D. T. PROSPECTS FOR THE DEVELOPMENT OF TOURISM IN UZBEKISTAN //Экономика и социум. – 2021. – №. 3-1. – С. 289-292.
8.Zufarjonovna J. G. USING WEB-QUEST TECHNOLOGY IN ENGLISH LESSONS AS FOREIGN LANGUAGE //INTERNATIONAL JOURNAL OF SOCIAL SCIENCE & INTERDISCIPLINARY RESEARCH ISSN: 2277-3630 Impact factor: 7.429. – 2022. – Т. 11. – С. 161-164.
9.Zufarjonovna J. G. BENEFITS OF USING WEB-QUEST TECHNOLOGY IN ENGLISH LESSONS AS FOREIGN LANGUAGE //INTERNATIONAL JOURNAL OF SOCIAL SCIENCE & INTERDISCIPLINARY RESEARCH ISSN: 2277-3630 Impact factor: 7.429. – 2022. – Т. 11. – С. 158-160.