ENGINEERING MATHEMATICS

MSC 68M99 **DOI: 10.14529/jcem160301**

PROTECTION SYSTEM OF APPLICATIONS ON "WINDOWS" PLATFORM ON THE BASIS OF ACTIVITY PROFILE

N. O. Artes[1]***,*** nikas.artes@gmail.com***,***
S. M. Elsakov[1]***,*** esergeym@mail.ru***.***
[1] South Ural State University, Chelyabinsk, Russian Federation.

The paper deals with a description of the developed prototype of application protection module based on an activity profile. It is based on proactive protection technology. A description of behaviors injected in controlled processes is given, and an architecture of the security module and interactions of end-users with the system are considered. The module is designed in C#.

Keywords: anti-virus software, malicious software, application activity profile, proactive protection, DLL injection, interaction of malicious software with the system.

Introduction

A malicious software is any software designed to get an unauthorized access to the computing resources of the computer, or to the information stored on the computer, where the purpose is unauthorized use of the computer resources or to harm to the owner of the information, or the owner of the computer, by copying, distortion, deletion or substitution of information [1].

The reasons for the growth of thefts committed with the help of malicious software are an increase of the number of malicious programs, an invention of new successful threats by virus writers, the fact that viruses use the vulnerabilities that have not yet closed by software manufacturers, incorrect security settings (including anti-virus ones) [2].

To organize an effective anti-virus system of network protection, it is important to know actual ways of penetration of malicious programs into the network. Today a software vulnerability, the Internet and websites, removable media, email are the most common ways [3].

There are following "Kaspersky Lab" data during 2015. So, 121 262 075 unique malicious objects (scripts, exploits, executables, etc.) were detected, ransomware programs were detected on 753 684 computers of unique users, 179 209 computers were attacked by cryptographers programs, 34% of Internet users computers at least once are subjected to virus attack [4].

These statistics show that the risk of infecting of computer or workstation is enough big. Therefore, an information security is an important issue to ensure the protection and security of data.

1. Classification by Popular Software Protection Technology

Data security involves ensuring of data reliability and protection of data and programs from unauthorized access, copy, modify. Protection of data and programs from unauthorized access, copy, modification is implemented with software and hardware techniques and technological methods. One of ways to protect data is to use anti-virus software.

Antivirus programs use the following software protection technology [5]:

1. Signature detection method.
2. Methods of proactive protection of software and applications:
   2.1. heuristic analysis,
   2.2. behavioral analysis.

Signature method of detection is a scanner that scans files by comparing the signature files with the dictionary.

A file is called infected, if any part of it is found in the signature database. This form of scanning allows to determine the type of attack with high probability and without false triggering. But there are drawbacks, such as inability to detect new viruses that do not exist in the database. It requires constant updating of signature databases. Upon detection of a new virus, its signatures are created as a result of the manual analysis of several copies of a file belonging to a single virus. A signature should contain only unique lines of this file, which are characteristic such that to ensure minimal possibility of false triggering. It is quite time-consuming process, and a period of detection of the unique signatures of malicious software can be a long time [5].

Heuristic analysis is a technological method of application protection, which is based on the identification of the most probable behavior of malicious software, such that removal or change of files.

Heuristic analysis technique allows to detect previously unknown infections, but the technique is often prone to errors of false triggering and requires a complex manual tuning, so it is rarely used.

Behavioral analysis technology is based on the interception of all important system functions and installation of filters. It allows to monitor all activity on custom system. Its advantages are the following. First is low consumption of resources, which are spent only on an implementation of the functionality in the monitored system calls. Second, the behavioral analysis allows to catch an early unknown virus. Also, in contrast to the heuristic analysis, the behavioral analysis is quite simple to implement. The false triggering is shortcoming of behavioral analysis technology.

The behavioral analysis is most effective to detect already studied, as well as previously unknown, virus software. It was found during the considering of popular techniques of software protection. Also, the behavioral analysis is the easiest to implement.

The most common technology of behavioral analysis is DLL injection [6]. DLL injection is a technology of an interception of function calls in external processes in Windows. Suppose an application runs. Then the operating system creates its process and then exe-file is copied into memory. After that it is determined what kind of library (dll-files) process needs to work (this information is recorded at the beginning of each exe-file), these libraries are found (in the program folder and system folders), and then they are loaded into the process memory. After that it is determined what kind of library functions

are used by the program and where they are (in what kind of library, and where it is in the library). A table of functions import [7] is constructed. A part of it is shown in Figure 1. The essence of DLL injection is a replacement of function address in the table of functions import to address of the injection function (Fig. 2).

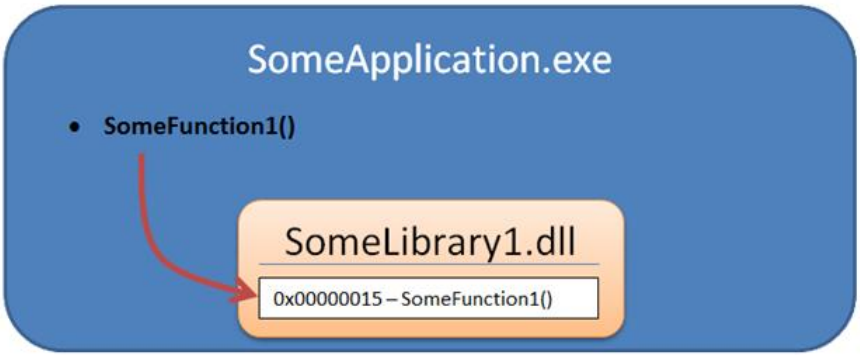

Fig. 1. A Fragment of Table of Functions Import

This is done as follows. An additional library SomeLibrary2.dll, in which function SomeFunction2() will be located, is developed. Next, the library is loaded into the memory of another process, and the table of import functions is changed so that now it contains the entry "function SomeFunction1() — library SomeLibrary2.dll — $\%function\_address\_SomeFunction2()\%$".

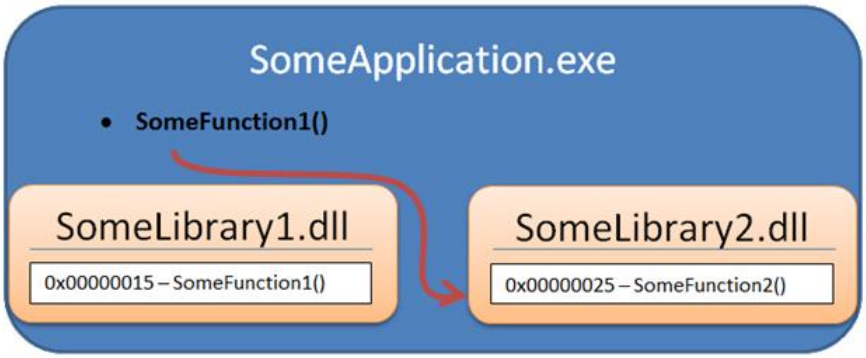

Fig. 2. The Modified Table of Functions Import

2. A prototype of Application Protection Module Based on the Activity Profile

The proposed application security module is based on DLL injection technology.

Architecture of active protection module has a multi-level template and is divided into two parts. They are an injection layer and a layer of behavior of intercepted WIN API functions in the selected process. A layer of behavior of active protection module has two behaviors – data collection mode and protection mode. Module architecture is shown in Fig. 3.

On the interception layer the module injects one of two behaviors, which is described in the layer of API functions behavior, in the selected processes.

In the data acquisition mode, an active protection module implements a behavior that forces certain intercepted WIN API functions to add a directory visited by process, a registry key, etc. to the white list file (which is the activity profile of the intercept API function). In the protection mode, the module also injects behavior, which already checks the activity of the same intercepted WIN API functions with the activity profile constructed earlier.

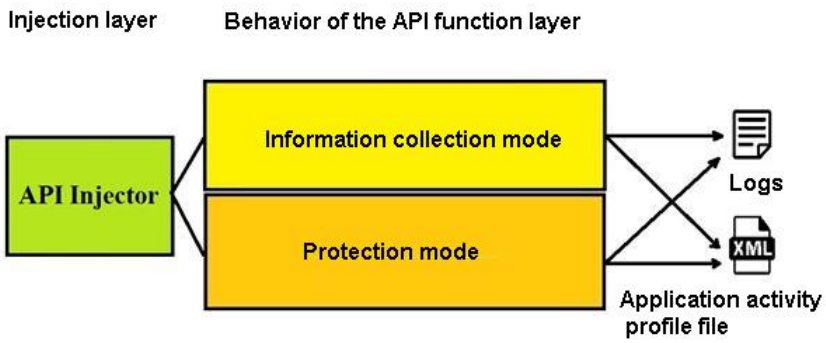

Fig. 3. An Architecture of Active Protection Module

If the activities of the intercept WIN API function is not valid, then the module creates an entry in the log and completes the work of WIN API functions as a fail.

Module architecture provides simple support for application and adding of new API functions to intercept and new behavior for injection. The programmer can easily add a new WIN API function to capture and implement a new behavior for these functions.

In order to limit the area of responsibility and to prevent conflicts, for each WIN API function its own activity profile file having its own whitelist key is created. For each selected process, in which the selected behavior is injected, a subdirectory to store files of profiles activity for each intercept WIN API function is created.

The diagram of cases of active protection module use is shown in Fig. 4. In this diagram of use cases in UML notation several "actors" are provided. Here the actors are used to refer to a consistent set of roles that users can play in the process of interaction with the planned system.

Consider the details of each actor.

1. "Administrator" is a user who has the right to choose the inject processes running on a workstation, injecting of data collection behavior and active protection. Also the administrator can view the file of profile of WIN API functions activity, and view the log files in which system messages of program are written, as well as a directory or system registry keys such that they do not belong to the profile activity file, but intercepted WIN API functions are trying to access to them.
2. "User" differs from the administrator in the following way. He can not inject different behaviors in any of the processes on a computer, but he can view the log file and work in process with injected behavior of intercepted WIN API functions.

Consider every action in details:

1. Work in processes with injected behavior is an ability of user and administrator to work in process such that that our module is connected to it.

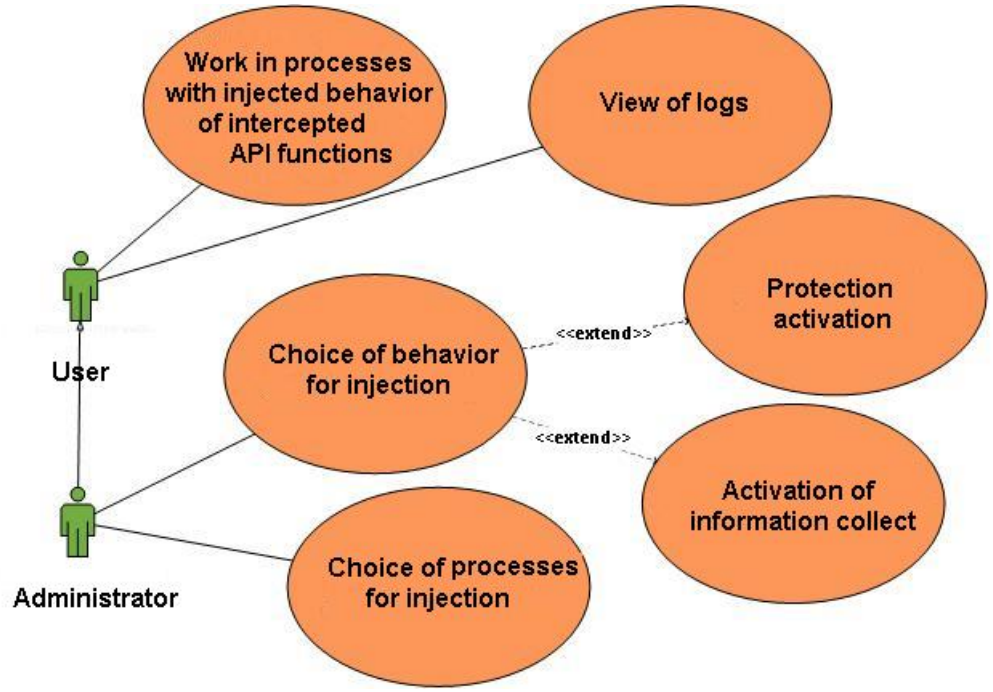

Fig. 4. Diagram of Use Cases

2. View of the logs is an ability to view system messages and recorded directories or registry keys, to which intercepted WIN API functions tried to get by not authorized way.
3. Selection of processes for injection is an action associated with a selection of the desired process from the list of running processes on the computer to inject a certain behavior.
4. Selection of behavior for injection is an action, which consists of the following two sub actions.
   4.1. Activation of data collection is an injection of behavior of certain WIN API functions interception in the selected process and a composition of the activity profile of the so-called keys of white list that can be registry keys, directories, IP-addresses such that intercepted WIN API function try to get access to them. A selection of key of the white list or key of activity profile is determined by the programmer. They are selected according to specific intercepted WIN API function. For each intercepted WIN API function its own file of whitelist – a profile of application activity – is created.
   4.2. Activation of protection is an injection of behavior of interception of the same WIN API functions into the selected process and an installation of the validity of the access to the key of the activity profile such that the intercepted WIN API function try to get access to them. Admission to the key is called not valid, if it is not in the whitelist file. In this case, the key is recorded in the logs file with information about WIN API function which tried to get access.

Conclusion

The developed active protection module is built on a behavioral protection technology, DLL injection and a creation of activity profile files. The module protection from known viruses increases with increasing of the number of WIN API functions, which are intercepted by the module. There are a large number of WIN API functions in Windows. Therefore an intercept of several WIN API functions is realized in the designed module for demonstrative purposes. One can also note the advantage of this module in the Auto setting of protection for activated mode of data collection. This allows to configure the active protection module to the needs of a particular system. The module was analyzed from the point of view of safety. It was tested. The source code is posted in the open access on GitHub: https://github.com/NikArtes/Modul-of-active-protection

References

1. *Official Website of Kaspersky Lab. The Classification of Malware*, available at: http://www.kaspersky.ru/internet-security-center/threats/malware-classifications (accessed on 20 March 2016).

2. *Official Website of Kaspersky Lab. What Conditions are Need to Spread Malware*, available at: http://www.kaspersky.ru/internet-security-center/threats/hacking-system-vulnerabilities (accessed on 20 March 2016).

3. Artes N.O., Elsakov S.M. Comparative Analysis of an Interaction of Different Types of Malware on the "Windows" System. *Sbornik Trudov II Vserossijskoj Nauchno-Prakticheskoj Konferencii – Proceedings of the II All-Russian Scientific-Practical Conference*. Chelybinsk, Publishing Center of SUSU, 2015, pp. 11–18. (in Russian)

4. *Kaspersky Security Bulletin 2015. Summary Statistics for 2015*, available at: https://securelist.ru/analysis/ksb/27543/kaspersky-security-bulletin-2015-osnovnaya-statistika-za-2015-god/ (accessed on 2 February 2016).

5. Shangin V.Ph. *Information Security of Computer Systems and Networks*. Moscow, ID "FORUM" – INFRA-M Publ., 2011. (in Russian)

6. Koziol J., Litchfield D. *The Art of Hacking and System Protection*. St. Petersburg, Peter Publ., 2011. (in Russian)

7. Kaspersky K. *Computer Viruses Inside and Outside*. St. Petersburg, Peter Publ., 2012. (in Russian)

Nikita O. Artes, Undergraduate, Department of Differential and Stochastic Equations, South Ural State University (Chelyabinsk, Russian Federation), nikas.artes@gmail.com.

Sergey M. Elsakov, Candidate of Physico-Mathematical Sciences, Department of Differential and Stochastic Equations, South Ural State University (Chelyabinsk, Russian Federation), esergeym@mail.ru.

Received June 11, 2016

УДК 4.056.5 **DOI: 10.14529/jcem160301**

СИСТЕМА ЗАЩИТЫ ПРИЛОЖЕНИЙ НА ПЛАТФОРМЕ «WINDOWS» НА ОСНОВЕ ПРОФИЛЯ АКТИВНОСТИ

Н. О. Артес, С. М. Елсаков

В работе рассматривается описание разработанного прототипа модуля защиты приложений на основе профиля активности. За основу была взята технология проактивной защиты. Была разобрана архитектура модуля защиты, дано описание поведений, инжектируемых в подконтрольные процессы, и рассмотрены взаимодействия конечных пользователей с системой. Модуль разработан на языке C#.

Ключевые слова: антивирусная программа, вредоносное ПО, профиль активности приложений, проактивная защита, инжектирование dll, взаимодействие вредоносной программы с системой.

Литература

1. *Официальный сайт лаборатории Касперского. Классификация вредоносных программ*, доступ: http://www.kaspersky.ru/internet-security-center/threats/malware-classifications (запрос 20 Марта 2016).
2. *Официальный сайт лаборатории Касперского. Какие условия нужны для распространения вредоносных программ*, доступ: http://www.kaspersky.ru/internet-security-center/threats/hacking-system-vulnerabilities (запрос 20 Марта 2016).
3. Артес Н.О. Сравнительный анализ взаимодействия разных видов вредоносных программ на систему «Windows» / Н.О. Артес, С.М. Елсаков // Сборник трудов II всероссийской научно-практической конференции. – Челябинск: Издательский центр ЮУрГУ, 2015. – С. 11–18.
4. *Kaspersky Security Bulletin 2015. Основная статистика за 2015 год*, доступ: https://securelist.ru/analysis/ksb/27543/kaspersky-security-bulletin-2015-osnovnaya-statistika-za-2015-god/ (запрос 2 Февраля 2016).
5. Шаньгин, В.Ф. Информационная безопасность компьютерных систем и сетей / В.Ф. Шаньгин. – М.: ИД «ФОРУМ» – ИНФРА-М, 2011.
6. Козиол, Дж. Искусство взлома и защиты систем / Дж. Козиол, Д. Личфилд. – СПб.: Питер, 2011.
7. Касперски, К. Компьютерные вирусы изнутри и снаружи / К. Касперски – СПб.: Питер, 2012.

Артес Никита Олегович, магистрант, кафедра дифференциальных и стохастических уравнений, Южно-Уральский государственный университет (г. Челябинск, Российская Федерация), nikas.artes@gmail.com.

Елсаков Сергей Михайлович, кандидат физико-математических наук, кафедра дифференциальных и стохастических уравнений, Южно-Уральский государственный университет (г. Челябинск, Российская Федерация), esergeym@mail.ru.

Поступила в редакцию 11 июня 2016 г.