DOI https://doi.org/10.30525/978-9934-26-254-8-17

DEVELOPMENT OF SOFTWARE TO PROTECT EXECUTABLE FILES

Sharov S. V.

INTRODUCTION

This Information protection is a very important issue under the conditions of information society and the spread of the Internet. According to researchers, the condition of society is unstable in terms of information security. It is explained by such factors: there is a great number of vulnerabilities in operating systems and software which hackers use for their purposes[1]; it can take a lot of time to develop methods of search and protection from a specific malicious code[2]; we can observe constant emergence of new malware, information about which is missing in the databases of anti-virus programs for a while. In addition, the more software is developed, the more opportunities there are for hacking. Moreover, the number of attacks on security systems is constantly growing[3]. As a result, ensuring information security is one of the most important tasks which are set before the information society. And this process must take place on a regular basis taking into account current achievements in the field of information protection.

A variety of computer programs that perform system and application tasks are often attacked by hackers. This is especially true of the Windows operating system which is one of the most common operating platforms among personal computer (PC) and laptop users. Usually, the attacks target executable files that can be modified after the process of disassembly and decompilation. The situation is complicated by the frequent use of «pirated» software. It usually contains a malicious software code that implements either advertising functions, or the

[1] Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

[2] Belaoued M., Derhab A., Mazouzi S., Khan F.A., MACoMal: A Multi-Agent Based Collaborative Mechanism for Anti-Malware Assistance. *IEEE Access*. 2020. Vol. 8. P. 14329–14343. doi: 10.1109/ACCESS.2020.2966321

[3] Namanya A.P., Awan I.U., Disso J.P., Younas M. Similarity hash based scoring of portable executable files for efficient malware detection in IoT. *Future Generation Computer Systems*. 2020. Vol. 110. P. 824–832. doi: 10.1016/j.future.2019.04.044

functions of collection and transmission of personal information, etc. Moreover, in the pirated software the functions of checking the license key are removed, and there are no mechanisms to control the originality of the file, or to protect it from copying, etc.

If the software does not have adequate protection against interference with its structure and functionality, the end user risks losing personal information, money, a working operating system and hardware. A significant number of research works is devoted to the issues of ensuring the software integrity, as well as its protection against unauthorized copying and modification. Methods of protection used for Windows OS are researched in work of A. Ільєнко[4]. The review of methods used for the protection of the program code and executable files is presented in work of I. Stepanenko[5], J. Tevis[6], L. Van Duong[7]. To detect malware, one can use the analysis of format of file[8], hashes results[9], header and section table[10]. The research[11] studies the types of attacks and proper software; it also describes the examples of software protection against unauthorized interference with their structure and work.

---

[4] Ільєнко А.В., Ільєнко С.С., Куліш Т.М. Перспективні методи захисту операційної системи Windows. *Кібербезпека: освіта, наука, техніка*. 2020. № 4(8). С. 124–134. doi: 10.28925/2663-4023.2020.8.124134

[5] Степаненко І.В., Кінзерявий В.М., Наджі А.А.А., Лозінський І.І. Сучасні обфускаційні методи захисту програмного коду. *Безпека інформації*. 2016. № 22(1). С. 32–37. doi: 10.18372/2225-5036.22.10451

[6] Tevis J. E. J., Hamilton J. A. Static analysis of anomalies and security vulnerabilities in executable files. *Proceedings of the Annual Southeast Conference*. 2006. P. 560–565. doi: 10.1145/1185448.1185570

[7] Duong L. Van, Xuan C.Do. Detecting Malware based on Analyzing Abnormal behaviors of PE File. *International Journal of Advanced Computer Science and Applications*. 2021. Vol. 12(3). P. 464–471. doi: 10.14569/IJACSA.2021.0120355

[8] Zhang T., Lee W.H., Gao M., Zhou J. File Guard: automatic format-based media file sanitization. *International Journal of Information Security*. 2019. Vol. 18(6). P. 701–713. doi: 10.1007/s10207-019-00440-3

[9] Namanya A.P., Awan I.U., Disso J.P., Younas M. Similarity hash based scoring of portable executable files for efficient malware detection in IoT. *Future Generation Computer Systems*. 2020. Vol. 110. P. 824–832. doi: 10.1016/j.future.2019.04.044

[10] Maleki N., Bateni M., H. Rastegari. An Improved Method for Packed Malware Detection using PE Header and Section Table Information. *International Journal of Computer Network and Information Security*. 2019. Vol. 11(9). P. 9–17. doi: 10.5815/ijcnis.2019.09.02

[11] Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

Despite the significant number of existing methods and software tools used to ensure information security, hackers continue to violate the integrity of operating systems, steal personal information, etc. Therefore, an urgent need arises to create new ways for software protection, including executable files. The purpose of this research is to analyze the issue of protecting software against the unauthorized access and modification, to report on the development of software to protect executable files in Windows, and to describe the protection algorithm, as well as the functionalities of the computer program.

1. Approaches to secure software

Under the protection of integrity or protection against unauthorized access we will understand the process of data protection, the protection of structure or logic of the software. The choice of methods and ways of protection depends on the purpose, existing and potential threats to information security, as well as on the specific computer system, available hardware and software.

Among the most popular threats to information security there are local, cyber threats[12], including those with elements of artificial intelligence[13], «Man-At-The-End»-attacks[14]. All of them are characterized by a variety of mechanisms of interference with the software. The means of these attacks are usually malicious software that can gain control over the operating system and computer resources; it can perform malicious functions at various levels[15].

Therefore, a comprehensive security application should be implemented at three interrelated levels: data security (ensuring the confidentiality of information during its transmission and storage); network security (protection of network equipment, resources, services); software security (protection against unauthorized access, copying, modifications, etc.)[16]. For example, one can control the Internet traffic

[12] Srinivas J., Das A. K., Kumar N. Government regulations in cyber security: Framework, standards and recommendations. *Future Generation Computer Systems*. 2019. Vol. 92. P. 178–188. doi: 10.1016/j.future.2018.09.063

[13] Kaloudi N., Jingyue L.I. The AI-based cyber threat landscape: A survey. *ACM Computing Surveys*. 2020. Vol. 53(1). P. 1–34. doi: 10.1145/3372823

[14] Ahmadvand M., Pretschner A., Kelbert F. A taxonomy of software integrity protection techniques. *Advances in Computers*. 2019. Vol. 112. P. 413–486. doi: 10.1016/bs.adcom.2017.12.007

[15] Shiva Darshan S. L., Jaidhar C. D. Performance Evaluation of Filter-based Feature Selection Techniques in Classifying Portable Executable Files. *Procedia Computer Science*. 2018. Vol. 125. P. 346–356. doi: 10.1016/j.procs.2017.12.046

[16] Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

with sniffers. Their purpose is to intercept and store packets for further analysis by the system administrator. Sniffers allow you to detect suspicious activity either of individual users or software and make appropriate adjustments[17]. In order to protect server from targeted DDoS attacks various algorithms of a certain attack detection, including attacks with the use of neural networks, are applied[18].

As Windows OS is of great popularity, most attacks are targeted at this system. New versions of Windows also have software flaws, as evidenced by the emergence of new patches. Moreover, Windows OS will continue to be vulnerable in terms of information security. When it comes to a personal computer running Windows OS, there are several ways to ensure information security:

– Encrypting file system[19].

– The use of built-in tools, such as Microsoft Defender, antimalware scan interface (detection of malicious scripts in RAM), active directory, virtualization based security[20].

– The use of licensed software, its update in automatic or semi-automatic mode.

– The mandatory use of anti-virus programs.

Regarding the protection of confidential data, it is recommended to use different passwords for accounts, network repositories, etc. To facilitate the work with a large array of passwords, you can use a variety of software products (e.g., Password Boss, Password Commander, Kaspersky Password Manager, 1Password). You can also save passwords in a text file. In this case, it is necessary to adhere to strict requirements for its storage and concealment from others. If we mean the protection of commercial information, then various technologies should be used in this case[21].

---

[17] Lubko D., Sharov S., Strokan O. Software development for the security of TCP-connections. *Modern Development Paths of Agricultural Production: Trends and Innovations*. 2019. P. 99–109. doi: 10.1007/978-3-030-14918-5_11

[18] Sanmorino A.A. study for DDOS attack classification method. *Journal of Physics: Conference Series*. 2019. Vol. 1175(1). doi: 10.1088/1742-6596/1175/1/012025

[19] Penchalaiah P., Kumar M. Vijay, Ramesh K.R. A research threshold efficient hybrid encryption schema for secure file system. *International Journal of Recent Technology and Engineering*. 2019. Vol. 8(2). P. 888–891. doi: 10.35940/ijrte.B1167.0782S319

[20] Ільєнко А.В., Ільєнко С.С., Куліш Т.М. Перспективні методи захисту операційної системи Windows. *Кібербезпека: освіта, наука, техніка*. 2020. № 4(8). С. 124–134. doi: 10.28925/2663-4023.2020.8.124134

[21] Zhang X., Lu J., Li D. Confidential information protection method of commercial information physical system based on edge computing. *Neural Computing and Applications*. 2021. Vol. 33(3). P. 897–907.

Executable files or portable executable files (PE) are a major component of any software in Windows OS. In general, the scheme of the «pirated» modification of PE is quite simple. After gaining access to an executable file, hackers use certain tools to analyze the source code and logic of the program; they find vulnerabilities and modify the program[22]. Usually, hackers receive either a modified executable file with additional functions, or crackz which interfere with the original executable file. Then, the modified file is distributed (usually via the Internet) which further exacerbates the issue of using pirated software[23].

As a result, protection of PE against modification and copying is an important task for many developers. Hence, patches, various scanners, firewalls and complex systems to protect against copying are constantly created for them[24]. In this way, threats from known malicious software[25] (such as viruses, spyware, etc.) can be partially avoided. In order to analyze an executable file as for the malicious program code there are different methods: metadata verification, hash value of the file, row deleting, use of anti-virus programs, application of machine learning algorithms[26], analysis of header and section table of an executable file[27], dynamic analysis of malware using intelligent agents[28] etc.

Methods of protecting software against unauthorized execution and copying include: the use of registration and hardware keys, binding to media, use of smart cards, etc. It should be noted that the local storage of

[22] Степаненко І.В., Кінзерявий В.М., Наджі А.А.А., Лозінський І.І. Сучасні обфускаційні методи захисту програмного коду. *Безпека інформації.* 2016. № 22(1). С. 32–37. doi: 10.18372/2225-5036.22.10451

[23] Chang H., Atallah M.J. Protecting software code by guards. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* 2002. Vol. 2320. P. 160–175. doi: 10.1007/3-540-47870-1_10

[24] Bahaa-Eldin A.M., Sobh M.A.A. A comprehensive Software Copy Protection and Digital Rights Management platform. *Ain Shams Engineering Journal*. 2014. Vol. 5(3). P. 703–720. doi: 10.1016/j.asej.2014.03.001

[25] Tevis J. E. J., Hamilton J. A. Static analysis of anomalies and security vulnerabilities in executable files. *Proceedings of the Annual Southeast Conference*. 2006. P. 560–565. doi: 10.1145/1185448.1185570

[26] Duong L. Van, Xuan C. Do. Detecting Malware based on Analyzing Abnormal behaviors of PE File. *International Journal of Advanced Computer Science and Applications*. 2021. Vol. 12(3). P. 464–471. doi: 10.14569/IJACSA.2021.0120355

[27] Maleki N., Bateni M., H. Rastegari. An Improved Method for Packed Malware Detection using PE Header and Section Table Information. *International Journal of Computer Network and Information Security*. 2019. Vol. 11(9). P. 9–17. doi: 10.5815/ijcnis.2019.09.02

[28] Romero-Herrera R., García J.A.J., García V.M.S. Malware analysis based on smart agents and image classification. *Journal of Theoretical and Applied Information Technology*. 2020. Vol. 98(18). P. 3116–3127

software on external media has limitations on the amount and speed of data processing that can be stored there. Therefore, it is appropriate to use the software access keys and store the data on a computer.

To complicate the static analysis of the logic of the program, encryption or compression of an executable file is used. To complicate the dynamic analysis of the program, hiding library function calls, use of anti debugging methods, counteracting memory dump records are used. Promising are obfuscations methods, which complicate the analysis of the source code of the program and the algorithm of the program for further modification. This task is achieved by complicating the PE program code before the compilation process. In work[29] all modern obfuscation methods are divided into the following groups: code structure transformation; variable transformation; punctuation transformation. In their study they describe the use of these methods on the example of the C++ programming language.

In order to study the logic of software and interference with its structure there are many computer programs designed for disassembly and decompilation[30]. Disassembler allows you to get the source code in the form of an assembler program[31] using direct and recursive approaches. Disassembly is considered one of the most effective and common ways of PE static analysis as for the presence of a malicious program code[32]. Decompiler allows you to get the program code in a high-level language (usually in C++ or Java). It should be added that the disassembly and analysis of executable files is carried out not only to do harm and perform «pirate» actions. Everything depends on the goal set by the programmer (improvement of the software on a legal basis or modification of an executable file for illegal purposes). For example, disassembly software is a useful tool for reverse engineering of a computer program.

An effective way to protect software against interference with its structure and performance is to use special software utilities (protectors). The essence of their work is to create a special software code in the PE structure which is responsible for downloading software to memory. After

---

[29] Степаненко І.В., Кінзерявий В.М., Наджі А.А.А., Лозінський І.І. Сучасні обфускаційні методи захисту програмного коду. *Безпека інформації*. 2016. № 22(1). С. 32–37. doi: 10.18372/2225-5036.22.10451

[30] Tevis J. E. J., Hamilton J. A. Static analysis of anomalies and security vulnerabilities in executable files. *Proceedings of the Annual Southeast Conference*. 2006. P. 560–565. doi: 10.1145/1185448.1185570

[31] Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

[32] Duong L. Van, Xuan C. Do. Detecting Malware based on Analyzing Abnormal behaviors of PE File. *International Journal of Advanced Computer Science and Applications*. 2021. Vol. 12(3). P. 464–471. doi: 10.14569/IJACSA.2021.0120355

the program being downloaded, the control is first passed to the software module of protector which unpacks the PE data, checks their correctness and transfers control to the Original Entry Point. Then, the work of the software begins.

2. Developing and description of software to protect executable files

Requirements and Stages of Development

Considering the existing situations related to the possibility of corrupting executable files and making dangerous modifications to them, we set a task of developing a system utility to protect an executable file. Let us dwell on the basic requirements for the developed computer program.

General requirements for the use of software. The software is offered to use by the principle «as it is». This software is used at the user's own risk. The developer is not responsible for the distortion of data or the original executable file in the process of using or misusing the developed software. It is recommended that you should make a copy of the original executable file when using this computer program. When setting protection options, you should use your common sense.

Interface design requirements. The graphical interface of the software must be flexible and easy to use. It must be intuitively understandable and it must provide possibilities for scaling. The program interface must be the same on computers with different localizations.

The main program window must contain links to all functionalities and it must have the minimum required number of control components (buttons, menu items, input fields). The control components that cannot be used must be explicitly disabled. Each new entry in the event log must start on a new line at the time of the event.

The window of security configuration settings must have simple and clear graphical interface; it must contain a link to the hardware ID management window.

The ID list management window must contain a list of saved hardware IDs; it must be able to add and remove IDs, to view and edit the detailed information about the equipment and its owner.

The development of the software involved the following stages:

– Analysis of the visual area that will be implemented in the software.

– Development of an algorithm for protecting an executable file.

– Choosing tool environment for creating a system computer program.

– Creating graphical interface, and developing software modules.

– Testing and verification of the software.

Algorithm of an Executable File Protection

Each executable file has a clearly defined data structure which is divided into sections. The elements of PE Windows which are often used for protection include the file header[33] that contains a special field EntryPoint (a place in the program where control will be transferred after the program loading into memory), the section of the code, resources, static and dynamic data. The research by J. Tevis and Jr J. Hamilton revealed that analyzing the information stored in MS-DOSStub, FileHeader could detect certain anomalies in an executable file (such as buffer overflow). Also, security vulnerabilities can be detected through the content analysis of section table, symbol table, and import table. However, the content of string table, various sections and export table do not contain the information that could detect the vulnerability of an executable file[34]. Keeping the structure of PE in mind, we can attribute the following methods to static protection: structure modification of an executable file, import tables, static code redirection, and encryption of PE individual parts[35].

The software development involved the development of a module that implements security mechanisms at the start of an executable file. The typical PE architecture influences the ability to integrate fragments of the security program code into the source code. There are following basic ways to embed the security program code in an executable file:

– Integration of the additional code in the space between the header section and the beginning of the first section of the file. Thus, the additional program code has to be rigidly optimized.

– Substitution of the content of the first section which contains the executable code and returning it back after the additional code has been executed. This method also has strict limits on the size of the additional program code.

– Extension of the last section of the file. In this case, the incorrect operation of the executable file is possible after the integration of protective functions.

---

[33] Shiva Darshan S. L., Jaidhar C. D. Performance Evaluation of Filter-based Feature Selection Techniques in Classifying Portable Executable Files. *Procedia Computer Science*. 2018. Vol. 125. P. 346–356. doi: 10.1016/j.procs.2017.12.046

[34] Tevis J. E. J., Hamilton J. A. Static analysis of anomalies and security vulnerabilities in executable files. *Proceedings of the Annual Southeast Conference*. 2006. P. 560–565. doi: 10.1145/1185448.1185570

[35] Bahaa-Eldin A.M., Sobh M.A.A. A comprehensive Software Copy Protection and Digital Rights Management platform. *Ain Shams Engineering Journal*. 2014. Vol. 5(3). P. 703–720. doi: 10.1016/j.asej.2014.03.001

– Adding an additional section to the end of the executable file. When the program is downloaded, the control is passed to the security module which then transfers the control to the original entry point.

We decided that the fourth option should be used for integration of the protection module as the most universal one and as having the least number of shortcomings.

In the work of the developed software, we used the following algorithm to protect an executable file in Windows. Protection of an executable file begins with its headers (sections) loading and with checking their correctness. Then, the sections are loaded into RAM. The import table is disassembled for conversion to an internal, more compact format. This will save about 50% of the size of the import directory (there will be a security module in a compressed file in place of the import directory). The resource directory is being disassembled to move it to the end of the file.

Usually, an executable file has a large amount of redundant data that are not necessary for the operation of the executable file (directory of moving elements, gaps between sections, etc.)[36]. In addition, this situation applies to debugging information (names of procedures, functions, variables), which are sometimes compiled with the binary file and used to find and correct errors in the source code. On the other hand, in some cases, its presence can be used by a hacker to recompile a binary file. It was decided to remove redundant data when applying protection to the executable file.

The next step is to compress the sections using the popular apLib library and then to encrypt the data. The arLib library is designed to compress Windows executables; it is free for commercial and non-commercial use, it requires minimal computer resources, it easily integrates with C ++, and it works quickly with x86 / x64 projects.

Resource directory compression does not occur. It is explained by the fact that many resources of an executable file are used by the operating system. These resources include the program icon, information about its version, the manifest. Their compression can lead to negative effects (for example, loss of program icons, version information, and loss of program performance in some cases). Therefore, it was decided to move the resource directory to a new section of the file, as in some cases the resource directory can share one section with the program code.

[36] Schwarz S., Debray G. Andrews Disassembly of executable code revisited. *Ninth Working Conference on Reverse Engineering*. 2002. 45–54.

After packing, the sections are merged into a continuous array, the checksum of the packed data is calculated, and the security module service headers are initialized and encrypted.

The last stage is to integrate the security module into the executable file. This stage consists of several steps:

– Search for a free address in the virtual address space.

– Moving the module code to this address using the directory of moving items.

– Making changes to the import directory. Since all offsets in the import table are specified relative to the design point of the file (ImageBase), the import table will point to null after moving.

– Adding the prepared section of the protection module and service data necessary for its work to the executable file, as well as modification of the original headers of the executable file.

A simplified structure of the executable file before and after protection is shown in Fig. 1.

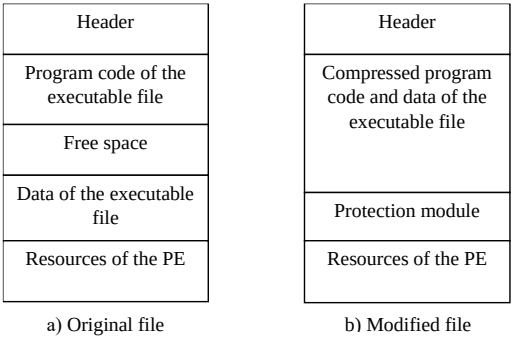

Fig. 1 A simplified structure of the executable file

After downloading the protected executable file, the control is passed to the entry point of the security module, which first checks the integrity of its own configuration, headers and the body of the executable file. Then the presence of the debugger is checked and authorization is performed (depending on the configuration of the security system). The original program code is unpacked. The table of addresses of imported functions is filled in and execution is transferred to the original entry point of the executable file.

It should be noted that the integration of security functions in one section is not considered the only way to protect the executable file. Thus, H. Chang suggest that the security sections should be integrated into the program code

of the Win32 executable file. They are smaller in size compared to one security module, and each of them performs separate functions. However, their simultaneous operation in the program will allow creating a comprehensive protection of the executable file[37]. We believe that different ways have a right to exist since they solve one common goal that is to protect the executable file from external interference.

Choice of the Tool Environment and Characteristics of the Project

Therefore, the Microsoft Visual Studio tool environment and the C++ programming language were used for its development. C++ is one of the most popular programming languages which contains tools for creating effective programs for practically any purpose, from low-level utility software and drivers to complex software. Visual C++ includes one of the best editors of the program code with the IntelliSense technology support. The compiler Microsoft Visual C++ supports both the traditional operation with the use of the machine code and operation with the virtual machines platforms, such CLR environment.

The Qt platform was used to develop the graphical interface[38]. Qt provides a programmer not only with a handy set of class libraries but also with a certain model for applications development, certain frame of their structure. An important advantage of Qt is a well-developed logical set of classes that gives a high level of abstraction to a programmer. As a result, the programmers who use Qt have to write less number of codes than they have to do while using MFC class libraries. It should be noted that the development of graphic interface is one of the most necessary components for training specialists in software development[39].

The software configuration is saved in a structured form in Json format[40]. This format allows you to freely read and modify the configuration in any text editor. The free JsonCpp library was used for software processing of the configuration of the protected executable file.

---

[37] Chang H., Atallah M.J. Protecting software code by guards. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2002. Vol. 2320. P. 160–175. doi: 10.1007/3-540-47870-1_10

[38] Woon H.C., Bau Y.T. Difficulties in learning C++ and GUI programming with qt platform – View of students. *ACM International Conference Proceeding Series*. 2017. Vol. F129684. P. 15–19. doi: 10.1145/3108421.3108429

[39] Chemerys H., Demirbilek M., Bryantseva H., Sharov S., Podplota S. Fundamentals of UX/UI design in professional preparation of the future bachelor of computer science. *AIP Conference Proceedings*. 2022. Vol. 2453. doi: 10.1063/5.0094433

[40] Bourhis P., Reutter J. L., VrgočD. JSON: Data model and query languages. *Information Systems*. 2020. Vol. 89. doi: 10.1016/j.is.2019.101478

The main module of the software consists of six main parts (modules). The modGUI project, on the basis of which the graphical interface of the software was compiled, contains 5 windows. Each window differs in its functionality (Table 1).

Table 1

Assignment of ModGui project forms

Module	Name	Functionality
modGUI.cpp	modgui.ui	The main window of the software
dlgOpts.cpp	dlgOpts.ui	Software configuration window
dlgHWIDLst.cpp	dlgHWIDLst.ui	A window for administering a list of equipment IDs
dlgAddHWID.cpp	dlgAddHWID.ui	A window designed to add a new hardware ID

Description of the Software

Work with the software (executable file modGUI.exe) is carried out using the following controls (buttons):

– «…» allows you to select an executable file for protection using a standard dialog box.

– «Protect» allows you to apply protection to the selected executable file.

– «Settings» allow the user to change the security configuration of the executable file.

– «About the program» allows you to view the information about the software developer.

– «Exit» allows you to quit a computer program.

The window of the software configuration settings allows you to arbitrarily change the security configuration, as shown in Fig. 2.

Fig. 2. Configuration window

The user can set the following options to protect the executable file:

The «Backup» option allows you to save the original executable file just in case before the protection procedure. This useful feature will help the user to return to the unprotected version of the file if necessary, to compare the functionality of both files (protected and unprotected). The name of the backup matches the name of the protected file, but it has *.bak extension.

The «Anti-debug» option allows you to protect the executable file from debugging.

The «No notifications» option blocks all user notifications generated by protection.

The «Protect import table» option allows you to protect the import table from its modifications, interception and recovery during debugging.

The «Code theft» option prevents the code from being stolen at the point of entry of the secure software and it makes impossible to create a snapshot of the process memory.

The «Aggressive mode» option allows performing aggressive actions (locking input, disabling the image on the display, etc.) while finding the debugger. The actions are aimed at complicating the removal of protection from the software product.

The «Password protection» option allows you to protect the software with a static password which must be entered each time you start a protected executable file. If the «Save password» option is activated, you only need to enter the password once. If the trial mode is activated, the password will be asked only after its expiration.

The «Trial Mode» option allows you to limit the life of a protected executable file from the first download of the executable file. The default is 25 days. At the end of the trial period, the program must be either removed from the computer or registered.

The «Bind to PC» option allows you to use the executable file only on the computers whose hardware IDs are stored in the executable file.

In the window which sets the configuration protection of the executable file (see Fig. 1), there is a button «Equipment IDs list» which allows the user to edit the list of allowed hardware IDs (Fig. 3).

This window contains all the information about the saved equipment IDs and their owners. All changes made by the user are local.

In order to prevent incorrect settings, all collected information about the computer is read-only and is added only through the window of adding the hardware ID.

To add a new equipment ID, click the «Add» button. A window will appear for you to insert the equipment ID, name or nickname of the owner. After clicking the «Add» button, the equipment ID will be checked for correctness, decrypted and added to the general list (Fig. 4).

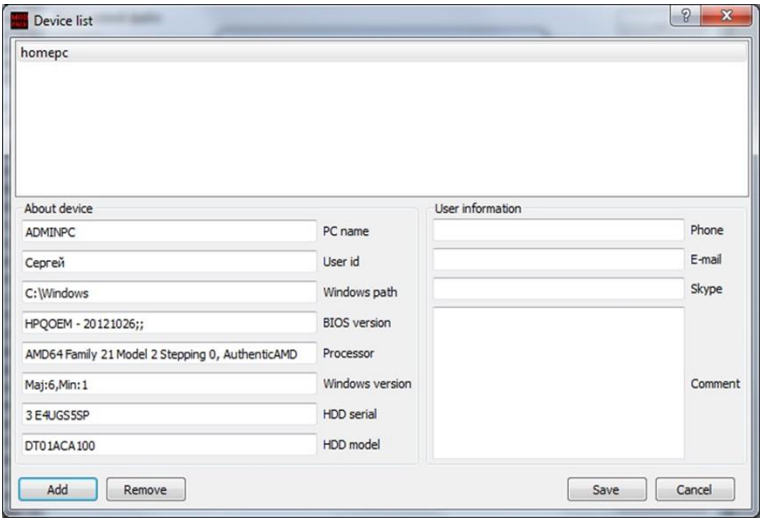

Fig. 3. The window of hardware IDs list control

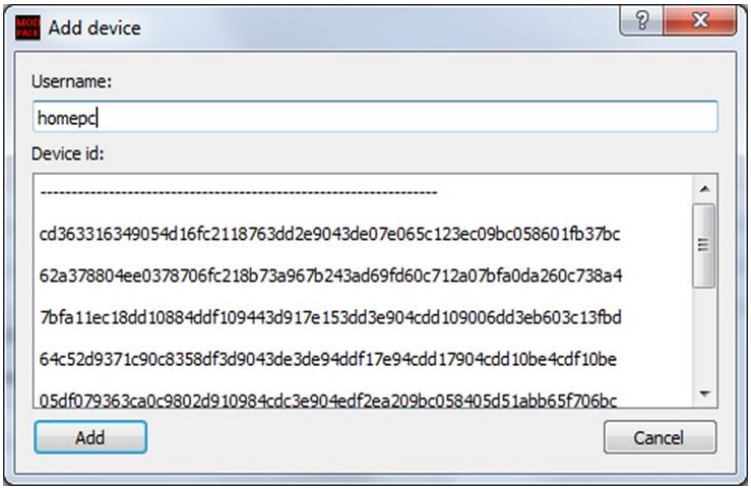

Fig. 4. The window of adding the hardware ID

If necessary, the user can specify the contact information of the identifier owner or add any comment.

The «License-tool.exe» utility is used to obtain the hardware ID of the current computer. When you run the utility, it collects information about your computer's operating system and hardware. The collected information is then encrypted and displayed in a window. For user convenience, the hardware ID is stored on the clipboard. It is also offered to save it to a file for later transfer.

The use of the developed software has several features:

– The developed software does not require the source texts of executable files.

– The executable files that are planned to be used in the environments which are run by virtual machines (such as Wine) will fail to work correctly when using the option of binding to the PC hardware IDs.

– Some software might cause false activation of the tamper protection option. Therefore, the option of aggressive protection should not be used without good reason.

– If you use the trial mode option, a system date failure might cause the false activation of protection when adjusting the date or time.

– In some cases, when using the «Code theft» option, it is possible to produce an inoperable executable file. This is most often caused either by the use of self-modifying code or undocumented features of the operating system.

As a result, the authors advise not to select all the protection options when using the developed software. This can cause difficulties when using protected executable files.

Despite significant developments in the protection of software and operating systems, the issue of information security requires further research and practical application. To date, there is no computer program that is completely protected from hackers. There is no system software that is able to completely protect executable files from «pirated» modifications. This is a logical phenomenon, as we can observe constant development of hacker attacks[41]; new computer viruses and malware are being developed[42].

[41] Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

[42] Belaoued M., Derhab A., Mazouzi S., Khan F.A., MACoMal: A Multi-Agent Based Collaborative Mechanism for Anti-Malware Assistance. *IEEE Access*. 2020. Vol. 8. P. 14329–14343. doi: 10.1109/ACCESS.2020.2966321

The issue of information security should be taken very seriously. First of all, it concerns the need for protective actions and their complexity. The degree of protection depends on the specifics of the organization that requires it, as well as on common sense. If it concerns a single person, malware can lead to the loss of irrelevant information[43], or it can damage the software that can be reinstalled. If the software contains important information for the economy of the country, enterprises, other structures, then the protection must be at a high level. It primarily concerns the state regulation of cybersecurity issues[44] and liability for different kinds of offences in the field of information protection.

In order to protect critically important executable files, it is recommended to apply a complex approach to the selection of protective functions[45], to use methods of static and dynamic protection of PE[46], as well as a complex of modules where each module performs its protective function[47]. At the same time, the protection of executable files can lead to some negative consequences:

– Reducing the software productivity.

– User dissatisfaction with the constant entry of passwords and security actions.

– Collisions that may occur in the protection system. Failure of any component of the debugged system may result in its termination.

– Sometimes there are additional financial costs.

If the software is used by end users to perform the information processing functions (such as a text editor or file manager), then the use of a high level of protection is a debatable issue. It should be added that users must monitor their personal important data, store them on the secure

[43] Azmee A. A. et al. Performance analysis of machine learning classifiers for detecting PE Malware. *International Journal of Advanced Computer Science and Applications*. 2020. Vol. 11(1). P. 510–517. doi: 10.14569/ijacsa.2020.0110163

[44] Srinivas J., Das A. K., Kumar N. Government regulations in cyber security: Framework, standards and recommendations. *Future Generation Computer Systems*. 2019. Vol. 92. P. 178–188. doi: 10.1016/j.future.2018.09.063

[45] Ільєнко А.В., Ільєнко С.С., Куліш Т.М. Перспективні методи захисту операційної системи Windows. *Кібербезпека: освіта, наука, техніка*. 2020. № 4(8). С. 124–134. doi: 10.28925/2663-4023.2020.8.124134

[46] Bahaa-Eldin A.M., Sobh M.A.A. A comprehensive Software Copy Protection and Digital Rights Management platform. *Ain Shams Engineering Journal*. 2014. Vol. 5(3). P. 703–720. doi: 10.1016/j.asej.2014.03.001

[47] Chang H., Atallah M.J. Protecting software code by guards. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2002. Vol. 2320. P. 160–175. doi: 10.1007/3-540-47870-1_10

media, and they must not show them to third parties. It has long been known that the users, who do not understand the importance of information protection, the ways of stealing or distorting it, often do not follow the rules of information security. It can result in loss of information, impairment of software efficiency, financial loss of an organization[48] etc. It is especially important for the virtual space safety which is characterized by various negative consequences[49]. In this case it is recommended to enhance the level of information culture, to improve knowledge of cybersecurity[50], especially in adolescents and students. The extension of the employees' knowledge of information security[51] will give an opportunity to raise corporate security to a higher level and to reduce the information loss.

It should be noted that this software can be used in the process of professional training in such specialties as «Computer Science». The above specialists must be able to develop and use the software, to apply appropriate methods and tools[52] to protect information resources of various types, create the user interface that could be intuitively understood. On the example of the developed computer program, you can learn about the logic of this type of program and the ways of software protection. Also, you can compare their effectiveness. Additionally, for the professional training in cybersecurity it is possible to use blended learning technology[53], mass open online courses and distance courses.

[48] Carlton M., Levy Y., Ramim M. Mitigating cyber attacks through the measurement of non-IT professionals' cybersecurity skills. *Information and Computer Security*. 2019. Vol. 27(1). P. 101–121. doi: 10.1108/ICS-11-2016-0088

[49] Kovacevic A., Putnik N., Toskovic O. Factors Related to Cyber Security Behavior. *IEEE Access*. 2020. Vol. 8. P. 125140–125148. doi: 10.1109/ACCESS.2020.3007867

[50] Mamonov S., Benbunan-Fich R. The impact of information security threat awareness on privacy-protective behaviors. *Computers in Human Behavior*. 2018. Vol. 83. P. 32–44. doi: 10.1016/j.chb.2018.01.028

[51] Stefaniuk T. Training in shaping employee information security awareness. *Entrepreneurship and Sustainability Issues*. 2020. Vol. 7(3) P. 1832–1846. doi: 10.9770/jesi.2020.7.3(26)

[52] Шаров С. В., Лубко Д. В. Розробка та використання сніферу як засобу забезпечення безпеки ТСР з'єднань. *Системи обробки інформації*. 2017. № 5. С. 138–144.

[53] Yang S., Newman R. Rotational Blended Learning in Computer System Engineering Courses. *IEEE Transactions on Education*. 2019. Vol. 62(4). P. 264–269. doi: 10.1109/TE.2019.2899095

CONCLUSIONS

Thus, today, software protection is a very important issue that the developers are facing. Quite often malicious attacks target executable files that run on Windows OS. By accessing the PE source code, hackers can analyze the logic of the program and make certain modifications in it. Various mechanisms and methods, including protectors, are used to protect an executable file from interfering with its structure. They change the source code of the executable file before the compilation process. Such actions complicate the work of disassemblers and decompilors; they allow providing Copyright. At the same time, a one-sided view of the software protection, particularly of an executable file, will not lead to effective results. It is worth noting that various protection mechanisms ought to be used against such threats, from cryptological methods of personal data protection to the software that makes it impossible to disassemble and modify executable files in Windows.

The main purpose of the developed system program is to protect Windows executable files from unauthorized copying and reverse engineering. To do this, the user can select built-in security options, such as «Import table protection», «Password protection», «Trial mode» and others. At the same time, one should consider the specific needs to protect an executable file and not to select all the options together. Microsoft Visual Studio and the C ++ programming language were used as tools for the software development. The Qt platform was used to develop the graphical interface. Also, we used free libraries JsonCpp (software processing of the executable file security configuration) and arLib (Windows executable files compression). The software has a simple graphical interface and can be used by users who do not have specific knowledge in the field of software protection. In further research, we plan to increase the number of options for protecting an executable file.

SUMMARY

The research discusses the development of software to protect executable files in Windows from unauthorized copying and reverse engineering. It has been found that with the development of information technologies attention has increased to the issues related to the unauthorized use, modification and distribution of software. By accessing the source code of an executable file Windows, hackers can analyze the logic of the program and make certain modifications. The use of effective protection technologies can complicate the process of hacking a software product, making it impractical in terms of time and effort. The research

covers some means and methods used to protect the software code from reverse software engineering. It has been found that a common method of protecting software from unauthorized modification is the use of program protectors. They change the source code of an executable file complicating its disassembly and analysis. The research dwells on the general mechanism demonstrating the way the developed software operates; its functionalities are described. It is noted that the user can modify the original executable file using various security options. To develop the software we used Microsoft Visual Studio, the C++ programming language, the Qt platform, free JsonCpp libraries, and arLib. The developed software does not require output texts of executable files; it can be used by users who do not have specific knowledge in the field of software protection. In future, it is expected to increase the number of security options supported by the software

BIBLIOGRAPHY

1. Ahmadvand M., Pretschner A., Kelbert F. A taxonomy of software integrity protection techniques. *Advances in Computers*. 2019. Vol. 112. P. 413–486. doi: 10.1016/bs.adcom.2017.12.007

2. Azmee A. A. et al. Performance analysis of machine learning classifiers for detecting PE Malware. *International Journal of Advanced Computer Science and Applications*. 2020. Vol. 11(1). P. 510–517. doi: 10.14569/ijacsa.2020.0110163

3. Bahaa-Eldin A.M., Sobh M.A.A. A comprehensive Software Copy Protection and Digital Rights Management platform. *Ain Shams Engineering Journal*. 2014. Vol. 5(3). P. 703–720. doi: 10.1016/j.asej.2014.03.001

4. Belaoued M., Derhab A., Mazouzi S., Khan F.A., MACoMal: A Multi-Agent Based Collaborative Mechanism for Anti-Malware Assistance. *IEEE Access*. 2020. Vol. 8. P. 14329–14343. doi: 10.1109/ACCESS.2020.2966321

5. Bourhis P., Reutter J. L., VrgočD. JSON: Data model and query languages. *Information Systems*. 2020. Vol. 89. doi: 10.1016/j.is.2019.101478

6. Carlton M., Levy Y., Ramim M. Mitigating cyber attacks through the measurement of non-IT professionals' cybersecurity skills. *Information and Computer Security*. 2019. Vol. 27(1). P. 101–121. doi: 10.1108/ICS-11-2016-0088

7. Chang H., Atallah M.J. Protecting software code by guards. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial*

Intelligence and Lecture Notes in Bioinformatics). 2002. Vol. 2320. P. 160–175. doi: 10.1007/3-540-47870-1_10

8. Chemerys H., Demirbilek M., Bryantseva H., Sharov S., Podplota S. Fundamentals of UX/UI design in professional preparation of the future bachelor of computer science. *AIP Conference Proceedings*. 2022. Vol. 2453. doi: 10.1063/5.0094433

9. Duong L. Van, Xuan C. Do. Detecting Malware based on Analyzing Abnormal behaviors of PE File. *International Journal of Advanced Computer Science and Applications*. 2021. Vol. 12(3). P. 464–471. doi: 10.14569/IJACSA.2021.0120355

10. Kaloudi N., Jingyue L.I. The AI-based cyber threat landscape: A survey. *ACM Computing Surveys*. 2020. Vol. 53(1). P. 1–34. doi: 10.1145/3372823

11. Kovacevic A., Putnik N., Toskovic O. Factors Related to Cyber Security Behavior. *IEEE Access*. 2020. Vol. 8. P. 125140–125148. doi: 10.1109/ACCESS.2020.3007867

12. Lubko D., Sharov S., Strokan O. Software development for the security of TCP-connections. *Modern Development Paths of Agricultural Production: Trends and Innovations*. 2019. P. 99–109. doi: 10.1007/978-3-030-14918-5_11

13. Main A. and Oorschot P.C. Software protection and application security: Understanding the battleground. *International Course on State of the Art and Evolution of Computer Security and Industrial Cryptography*. 2003. Vol. June, P. 8.

14. Maleki N., Bateni M., H. Rastegari. An Improved Method for Packed Malware Detection using PE Header and Section Table Information. *International Journal of Computer Network and Information Security*. 2019. Vol. 11(9). P. 9–17. doi: 10.5815/ijcnis.2019.09.02

15. Mamonov S., Benbunan-Fich R. The impact of information security threat awareness on privacy-protective behaviors. *Computers in Human Behavior*. 2018. Vol. 83. P. 32–44. doi: 10.1016/j.chb.2018.01.028

16. Namanya A.P., Awan I.U., Disso J.P., Younas M. Similarity hash based scoring of portable executable files for efficient malware detection in IoT. *Future Generation Computer Systems*. 2020. Vol. 110. P. 824–832. doi: 10.1016/j.future.2019.04.044

17. Penchalaiah P., Kumar M. Vijay, Ramesh K.R. A research threshold efficient hybrid encryption schema for secure file system. *International Journal of Recent Technology and Engineering*. 2019. Vol. 8(2). P. 888–891. doi: 10.35940/ijrte.B1167.0782S319

18. Romero-Herrera R., García J.A.J., García V.M.S. Malware analysis based on smart agents and image classification. *Journal of Theoretical and Applied Information Technology*. 2020. Vol. 98(18). P. 3116–3127

19. Sanmorino A.A. study for DDOS attack classification method. *Journal of Physics: Conference Series*. 2019. Vol. 1175(1). doi: 10.1088/1742-6596/1175/1/012025

20. Shiva Darshan S. L., Jaidhar C. D. Performance Evaluation of Filter-based Feature Selection Techniques in Classifying Portable Executable Files. *Procedia Computer Science*. 2018. Vol. 125. P. 346–356. doi: 10.1016/j.procs.2017.12.046

21. Schwarz S., Debray G. Andrews Disassembly of executable code revisited. *Ninth Working Conference on Reverse Engineering*. 2002. 45–54.

22. Srinivas J., Das A. K., Kumar N. Government regulations in cyber security: Framework, standards and recommendations. *Future Generation Computer Systems*. 2019. Vol. 92. P. 178–188. doi: 10.1016/j.future.2018.09.063

23. Stefaniuk T. Training in shaping employee information security awareness. *Entrepreneurship and Sustainability Issues*. 2020. Vol. 7(3) P. 1832–1846. doi: 10.9770/jesi.2020.7.3(26)

24. Tevis J. E. J., Hamilton J. A. Static analysis of anomalies and security vulnerabilities in executable files. *Proceedings of the Annual Southeast Conference*. 2006. P. 560–565. doi: 10.1145/1185448.1185570

25. Woon H.C., Bau Y.T. Difficulties in learning C++ and GUI programming with qt platform – View of students. *ACM International Conference Proceeding Series*. 2017. Vol. F129684. P. 15–19. doi: 10.1145/3108421.3108429

26. Yang S., Newman R. Rotational Blended Learning in Computer System Engineering Courses. *IEEE Transactions on Education*. 2019. Vol. 62(4). P. 264–269. doi: 10.1109/TE.2019.2899095

27. Zhang T., Lee W.H., Gao M., Zhou J. File Guard: automatic format-based media file sanitization. *International Journal of Information Security*. 2019. Vol. 18(6). P. 701–713. doi: 10.1007/s10207-019-00440-3

28. Zhang X., Lu J., Li D. Confidential information protection method of commercial information physical system based on edge computing. *Neural Computing and Applications*. 2021. Vol. 33(3). P. 897–907. doi: 10.1007/s00521-020-05272-0

29. Ільєнко А.В., Ільєнко С.С., Куліш Т.М. Перспективні методи захисту операційної системи Windows. *Кібербезпека: освіта, наука,*

техніка. 2020. № 4(8). С. 124–134. doi: 10.28925/2663-4023.2020.8.124134

30. Степаненко І.В., Кінзерявий В.М., Наджі А.А.А., Лозінський І.І. Сучасні обфускаційні методи захисту програмного коду. *Безпека інформації*. 2016. № 22(1). С. 32–37. doi: 10.18372/2225-5036.22.10451

31. Шаров С. В., Лубко Д. В. Розробка та використання сніферу як засобу забезпечення безпеки TCP з'єднань. *Системи обробки інформації*. 2017. №5. С. 138–144.

Information about the author:
Sharov Sergii Volodymyrovych,
Candidate of Pedagogical Science, Associate Professor,
Associate Professor at the Department of Computer Sciences,
Dmytro Motornyi Tavria State Agrotechnological University
66, Zhukovskoho str., Zaporizhzhia, 69600, Ukraine