Metamorphic Viruses Detection Technique Based on the Modified Emulators

Oksana Pomorova, Oleg Savenko, Sergii Lysenko, Andrii Nicheporuk

Khmelnitsky National University, 11, Institutskaya St.,
29016, Khmelnitsky, Ukraine

o.pomorova@gmail.com, savenko_oleg_st@ukr.net,
sirogyk@ukr.net, andrey.nicheporuk@gmail.com

Abstract. An article presents a new technique for metamorphic viruses detection using modified emulators, placed in the hosts of the network. Proposed technique provides the classification of the metamorphic virus in classes with the usage of the fuzzy logic. Technique makes it possible to detect the metamorphic viruses, which use obfuscation techniques. The results of experimental studies showed the effectiveness of the proposed method of detection metamorphic virus copies at 85%.

Keywords. Malware detection, metamorphic viruses, polymorphic viruses, obfuscation, modified network emulators

Key Terms. Machine Intelligence, KE, and KM for ICT, Software Component, Software System

1 Introduction

The detection of computer viruses is one of the main challenges of information security for today. Viruses execute harmful activity on infected hosts, such as stealing system information, accessing private information, corrupting data, spamming, logging their keystrokes. Among computer viruses one of the leaders are metamorphic viruses. Thus, metamorphic viruses cause damage to cost millions of dollars. According to Symantec in 2011 metamorphic virus Sality infected about 3 million computers in the world [1], and at the end of 2015 it is in the top five of the most spread viruses (1.43% of the total number of detected threats) [2].

Detection of the metamorphic viruses is a very complicated task due to it sus age of the obfuscation techniques for program code. It allows virus to create multiple copies of the same virus and its detection become a very difficult task. In addition, in order to complicate the process of reverse engineering and the data protection obfuscation technique is often used in the trusted applications by software developers [3].

That is why the actual problem is to develop a new technique for metamorphic viruses detection with the involvement of fuzzy logic that will allow the suspicious

ICTERI 2016, Kyiv, Ukraine, June 21-24, 2016

programs classification to one of the metamorphic viruses classes using the modified emulators placed on each host of the network.

2 Related Works

Known techniques for virus detection, based on signature analysis are not able to detect the altered copies of metamorphic virus [4-7]. In order to detect this type of viruses most antivirus scanners use the heuristic's method, which deploys the sequence of API-functions calls, the control flow graph of the program, the structural features of the PE .EXE files, opcode instructions and their combinations.

In [7]a method for metamorphic malware detection is presented. A malware signature is described by the set of control flow graphs the malware contains. Technique uses a distance metric based on the distance between feature vectors. The drawback of the approach is that it is computationally inefficient.

In [4]a statistical technique based on comparison of the similarity between two files infected by two morphed versions of a given metamorphic virus is used. The proposed solution based on static analysis and it uses the histogram of machine instructions frequency in various offspring of obfuscated viruses. The disadvantage of the approach is that it is inefficient for detection of viruses, which use code transposition techniques.

A malware detection system based on API call graph is proposed in [5]. Each malware sample is represented as data dependent API call graph. Graph matching algorithm is used to calculate similarity between the input sample and malware API call graph samples stored in a database. The main drawback that most malware samples are generated from previous existing samples, therefore sequences API calls are similar.

In [8] the technique of botnet detection which bots use polymorphic code is proposed. Performed detection is based on the multi-agent system by means of antiviral agents that contain sensors. Developed technique makes it possible to perform provocative actions against probably infected file. The disadvantage of this technique is large computational complexity of the behavior analysis.

Tha state-of-art demonstrates a need of development new and improve existing techniques for metamorphic viruses' detection.

3 Metamorphic Virus Detection Method-based on the Modified Emulators

A new metamorphic viruses detection technique based on the modified emulators is proposed. It uses the emulation on each host in the network. The main function of the hosts is to implement a single-time emulation and execution of the unknown potentially malicious program and sending the results to the server.

The server is used for processing the results of the emulation process obtained from the hosts. In order to complicate the reverse engineering process and data protection, the obfuscation techniques are often used for trusted applications by software developers. Therefore, the main task of the server is classification of the feature vec-

tors based on the comparison of the metamorphic viruses' copies, which were obtained from network hosts.

The proposed technique specifies three classes of the metamorphic viruses, which are to be classified. In addition, there is the fourth class of programs, which have similarity to metamorphic viruses by its behavior, but these programs are not malicious. Fig.1 shows a scheme of the metamorphic viruses detection.

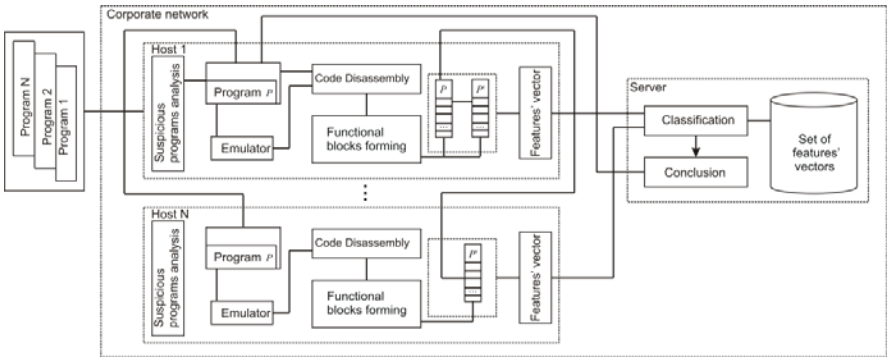

Fig. 1. The scheme of the metamorphic viruses detection

Let us consider the proposed technique. In order to detect suspicious program activity on each host of the corporate network, the analyzer of the suspicion programs is used. Every single operation that is performed by suspicious program is not dangerous one. However, the execution of the sequence of such operations may indicate a possible risk of infection with the virus.

Each application that comes into the system is marked as suspicious or un suspicious. Let us present the feature vector, which defines the program membership to one of two classes as follows:

$$\overline{U} = (M, Q, J, Y, L, N, H),$$

where, *M*–attempt of the program to get the system admin is trator rights, *Q* - attempt to open or close the system port, *J* - attempt to delete the file, *Y* - create a file or process, *L*–key logging attempt, *N* – sending messages to the network, *H* -creation a key or an entry to the registry.

Each feature is able to posses a value 0 or 1, where 1 indicates the activation of the featuretrait,0 -vice versa. Program consider suspicious if:

$$P = suspicious, if\ \forall u \in \overline{U}, (u_i = 1 \wedge u_j = 1).$$

So, if $P = suspicious$ for some program, it comes to the metamorphic viruses detection system. In order to get the modified code sample F_S, the emulation of program *P* is carried out. The emulation process consists of the instruction execution in a virtual environment, and the extraction instructions from the software package.

The usage of the one-type emulators in all the network hosts does not guarantee the metamorphic virus detection with high efficiency, because their usage will produce only the same code samples. In order to detect the metamorphic viruses properties and features, different conditions for malicious code execution are needed.

Therefore, the modified emulators on each host are created. The structure of the emulator includes the virtual processor. It is able to execute the set of instruction such as MMX, SSE, SSE2, etc, and it includes a set of virtual registers. Also, the emulator consists of RAM and virtual stack, virtual network controller; the operating system (it supports API functions, registry and ports).

To avoid the anti-emulation technologies, which are used by metamorphic viruses, the emulator includes a heuristics module. For each operation performed by virtual CPU the fixed, the processing time is determined and the checking of repeating for some operation is executed.

In order to obtain the original sample code F_P, the disassembly of program P is carried out. The result of disassembly is a set assembler instructions x86/x64. In order to construct the feature vector only opcodes are used and operands are discarded.

The resulting listing of the disassembly instructions is partitioned into the functional blocks (FB). One of the techniques that are used to perform the instruction obfuscation is moving of program blocks. It is carried out by using the conditional state transition instructions (jz, jnz, jmpetc).

The main mechanisms creation of the metamorphic viruses copies are the insertion, deletion and transposition of their instructions. For the purpose of finding the similarities between the two FB code samples F_P and F_S the Damerau–Levenshtein distance was used.

In order to evaluate the Damerau–Levenshtein distance the polynomial algorithm complexity of Wagner-Fisher was used. It made it possible to create a short transformation chain in order to transform the set of opcodes of the program after the emulation into the opcode set of the program before the emulation.

Consider a program P, which consists of a set of assembler commands p_i, $P = \{p_1, p_2, ..., p_k\}$. Let us partition the program P into the functional blocks of an arbitrary length. Such blocks start and end by the instructions of the conditional state transitions, such asjmp, jzetc, that is $P = \{B_1, B_2, ..., B_l\}$. Then we can write: $P = \{B_1 = \{p_1, p_2, ..., p_{i-1}\}, ..., B_l = \{p_i, p_{i+1}, ..., p_k\}\}$.

Let us denote program before the emulation F_P, and program after the emulation F_S.

Let us present the functional unit B, which consists of a set of opcodes of length $|B| = m$, as $p_1, p_2, ..., p_m$. So, the subset of opcodes $x_i, x_{i+1}, ..., x_j$ of the functional unit B will be specified as $B(i, j)$.

Let us denote the transformation weight of the opcode a into b as $w(a, b)$. Thus, $w(a, b)$ is the weight of the replacement of one opcode into another one, when $a \neq b$, $w(b, a)$ -weight of the transposition, $w(a, \varepsilon)$ -weight of the deletion, and $w(\varepsilon, b)$ - weight of the insertion for opcode b.

Let us assume that B_g and B_h - two FB, which consist of the opcodes sequence (of n and m length respectively) defined by a finite alphabet of the assembler instruc-

tions $A = (a_1, a_2, ..., a_k)$. Then B_g of the FB program F_P we will denote as $B_g^{F_P}$, and B_h of the same FB program F_S after the emulation we will denote as $B_h^{F_S}$. Then the Damerau-Levenshte in distance $dL(B_g^{F_P}, B_h^{F_S})$ is calculated as $dL(B_g^{F_P}, B_h^{F_S}) = OPT(N, M)$, where

$$OPT = \begin{cases} 0, & i = 0,\ j = 0 \\ i, & j = 0,\ i > 0 \\ j & i = 0,\ j > 0 \\ \min \begin{cases} OPT(i, j-1) + w(a, \varepsilon) \\ OPT(i, -1j) + w(\varepsilon, b) \\ OPT(i, -1, j-1) + w(a, b) \\ OPT(i-2, j-2) + w(b, a) \end{cases} & j > 0, i > 0 \end{cases}. \quad (1)$$

After the Damerau-Levenshte in distance for two bocks B_g and B_h is evaluated, the weighted averages of the corresponding parameter of the feature vector for all code blocks is to be formed. In order to obtain such weighted averages of the parameters, the index of the weighted arithmetic mean is used (2).

$$dL = \left\lfloor \frac{\sum_{i=1}^{n} dL_i * f_l}{\sum_{i=1}^{n} f_l} \right\rfloor, \quad (2)$$

where, dL_i – the Levenshte in distance for FB B_i, f_l - number of FB with the value dL_i.

TheDamerau-Levenshtein distance estimates the minimum value for the required operations of the replacement, insertion, deletion and transposition, and is an integer value. In this case, for the finding of the lowest difference between the metamorphic viruses' copies the obtained values are rounded down.

For the rest of the features the normalization is performed in the same way.

Thus, the feature vector of similarity for metamorphic viruses' copies based on theDamerau-Levenshte in metrics will be presented as follows:

$$\overline{S} = \langle dL, T, D, I, R, M \rangle, \quad (3)$$

where dL – the Damerau-Levenshte in distance for functional unit between programs F_p and F_s; T–number of required operations of the opcodes exchange for the program block's transformation F_p into F_s ($F_p = F_s$); D - number of operations required for the opcode deletion; I - number of operations required for the opcode inser-

tion; R - number of operations required for the opcode replacement; M - number of matches between opcodes of the functional units of programs F_p and F_s.

In order to make a conclusion about the infection by metamorphic virus, constructed feature vectors of the similarity are sent to server, where they are analyzed by the fuzzy inference system for the purpose of its classification [9].

The input linguistic variables for fuzzy inference systems are the feature vectors of similarity for the copes of the metamorphic viruses (3). The terms of the linguistic variable are Low, Medium and High.

As the membership function for input variables the trapezoidal one was chosen, and for output –the triangular. Forfeature dL we can present equations as follows:

$$\mu_{low}(x)=\begin{cases}0, & 72<x\\ \dfrac{72-x}{64}, & 8<x\le 72\\ 1, & 0\le x\le 8\end{cases} \qquad \mu_{medium}(x)=\begin{cases}0, & x<16\\ \dfrac{x-16}{49}, & 16\le x<65\\ 1, & 65\le x<96\\ \dfrac{145-x}{49}, & 96\le x<145\\ 0, & 145\le x\end{cases}$$

$$\mu_{high}(x)=\begin{cases}0, & 96<x\\ \dfrac{x-96}{38}, & 96\le x<134\\ 1, & 134\le x\le 161\end{cases}$$

Fuzzy inference system uses 38 rules for making the conclusion about belonging of the metamorphic virus to one of the class:

1 *if* (*dL is Low*) *and* (*T is Low*) *and* (*D is Medium*) *and* (*I is Hight*) *and* (*R is Low*) *and* (*M is Medium*) *then class*1

2 *if* (*dL is Low*) *and* (*T is Medium*) *and* (*D is Medium*) *and* (*I is Hight*) *and* (*R is Low*) *and* (*M is Medium*) *then class*1

38 *if* (*dL is Hight*) *and* (*T is Low*) *and* (*D is Hight*) *and* (*I is Hight*) *and* (*R is Medium*) *and* (*M is Low*) *then class*3

A result of the fuzzy inference system is a determination of the member ship degree of each virus copy to one of the class of the metamorphic viruses.

4 Experiments

In order to determine the efficiency of the proposed method several experiments were held. For this purpose the set of metamorphic viruses was generated, and metamorphic generators Next Generation Virus Creation Kits, Second Generation Virus Generator and Virus Creation Lab for Win32were used [10].They are able to infect .EXE and .DLL files and perform the obfuscation operations with files.

For experiments, the university network with 80 users' hosts and one server was used. Each host us edits modified emulator. The settings of the emulators are presented in Table 1.

Table 1. The settings of the modified emulators, presented on each host of the network

Host's number	The initial network settings of modified emulators							
	A set of instructions	OS type	The size of virtual RAM	System date	Virtual network controller	Runtime emulation instruction,	Start address of emulating	Architecture of CPU
1	SSE2	W7	8	25.11	Yes	1	0x001A4810	x86
2	Hyper-threading	W8.1	8	06.02	No	2	0x001A4620	x86
…	…	…	…	…	…	…	…	
80	MMX	W10	4	10.04	No	1	0x001A5214	x64

Fuzzy inference system was built using Fuzzy Logic Toolbox package in Matlab system. Built system has the following parameters: algorithm –Mamdani, aggregation method, accumulation method, the method defuzzification. The system consists of six inputs and one output.

The result of fuzzy inference system is a conclusion about membership of the unknown program to one of three metamorphic viruses' class or program is trusted. If the resulting value of the membership degree for unknown object belongs to the range from 0 to 0.25 - it is classified as a trusted application; if the membership degree of is in the range of 0.26 to 1, then the unknown object belongs to one of the metamorphic viruses' classes. Values from 0.26 to 0.49 determine the first class of the metamorphic viruses, values from 0.5 to 0.74 –the second class, the value of 0.75 to 1 –the third class.

Table 2 and figure 2 demonstratethe results of fuzzy inference system for suspicious file. As a result,15% of the copies have not changed, 5% were classified asthe first class,11.25%-the third one, and 68, 75% as the second class.

Table 2. The result of fuzzy inference system

№ of host						Result of the fuzzy inference system					
I	II	I	II	I	II	I	II	I	II	I	II
1	0,65	15	0,57	29	0,81	42	0,72	55	–	68	–
2	–	16	0,86	30	0,70	43	0,51	56	0,61	69	0,72
3	0,19	17	0,57	31	0,55	44	0,53	57	0,63	70	0,83
4	0,64	18	0,63	32	–	45	0,63	58	0,84	71	–
5	0,45	19	0,56	33	0,62	46	0,78	59	0,67	72	0,53
6	0,59	20	0,68	34	0,23	47	0,62	60	0,69	73	–
7	–	21	0,72	35	0,59	48	–	61	0,67	74	0,59
8	0,52	22	0,41	36	0,54	49	0,70	62	–	75	0,47
9	–	23	0,69	37	0,68	50	0,26	63	0,51	76	0,74
10	0,61	24	0,79	38	–	51	0,56	64	0,82	77	–
11	0,14	25	0,68	39	0,39	52	0,81	65	0,69	78	0,58
12	0,69	26	0,24	40	0,56	53	0,74	66	0,36	79	0,57
13	0,78	27	0,56	41	0,54	54	0,56	67	0,52	80	0,73
14	0,35	28	0,69								

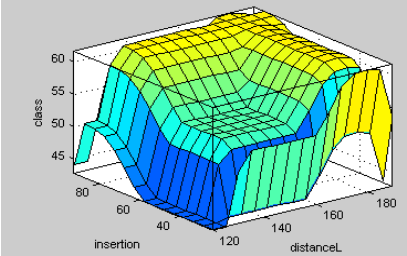

Fig. 2. The result of fuzzy inference about membership of the unknown program to one of metamorphic viruses' class

5 Conclusions

The analysis of the subject area revealed the need to improve existing techniques for metamorphic viruses' detection. In the article, the new technique for metamorphic viruses' detection using modified emulators in network hosts is proposed. The classification of viruses into classes of metamorphic viruses is based on a usage of the fuzzy inference system. Proposed technique makes it possible to detect metamorphic viruses that use obfuscation techniques of the program code. Such approach enables the increase of the efficiency of the metamorphic viruses detection. The results of experimental studies have demonstrated the efficiency technique for metamorphic viruses' detection at about 85%.

References

1. Falliere, N.: Sality: Story of a Peer-to-Peer ViralNetwork. Technical report, Symantec Labs (2011)
2. Virus statistics. Overview cyber November 2014. online: https://www.esetnod32.ru/company/viruslab/statistics/?id= 896397 [in Russian]
3. Attaluri, S., McGhee, S., Stamp, M.: Profile Hidden Markov Models and Metamorphic Virus Detection. Journal on Computer Virology, 5, 151--169 (2009)
4. Rad,B. B., Masrom, M.: Metamorphic Virus Variants Classification Using Opcode Frequency Histogram. In: Proc. 14th WSEAS Int Conf on Computers, pp. 147--155, WSEAS (2010)
5. Elhadi, A. A. E., Maarof, M. A., Barry, B. I. A.: Improving the Detection of Malware Behaviour Using Simplified Data Dependent API Call Graph. Int. J. of Security and Its Applications, 7(5), 29--42 (2013)
6. Kaushal, K., Swadas, P., Prajapati, N.: Metamorphic Malware Detection Using Statistical Analysis. Int. J. of Soft Computing and Engineering, 2, 49--53 (2012)
7. Cesare, S., Xiang, Y.: Malware variant detection using similarity search over sets of control flow graphs. In: Proc. 10th Int. Conf. on Trust, Security and Privacy in Computing and Communications, Washington, DC, USA, pp. 181-189 (2011)
8. Pomorova, O., Savenko, O., Lysenko, S., Kryshchuk, A., Nicheporuk, A.: A technique fordetection of bots which are using polymorphic code. In: Proc. 21st Int. Conf, CN, Springer, Brunów, Poland, pp. 265--276 (2014)

9. Shtovba, S, Pankevich, O., Nagorna, A.: Analyzing the criteria for fuzzy classifier learning. Automatic control and computer sciences, 49(3), 123--132 (2015)
10. VX Heavens Computer virus collection. online: http://vx.netlux.org