УДК 621.391

А.О. НІЧЕПОРУК
Хмельницький національний університет

МОДЕЛЬ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЯВЛЕННЯ МЕТАМОРФНИХ ВІРУСІВ НА ОСНОВІ ПОРІВНЯННЯ ЇХ ПОВЕДІНКИ

В роботі запропоновано модель інформаційної технології виявлення метаморфних вірусів, що заснована на порівнянні їх поведінки з використання системи нечіткого логічного висновку. На базі моделей поведінки метаморфних вірусів формується висновок про інфікування виконуваного файлу метаморфним вірусом.

Ключові слова:метаморфні віруси, вектор ознак, поведінка.

A.O. NICHEPORUK
Khmelnytskyi National University, Khmelnytskyi, Ukraine

THE MODEL OF INFORMATION TECHOLOGY FOR DETECTION METAMORPHIC VIRUSES BASED ON THEIR BEHAVIOR

The paper presents a model of information technology detect metamorphic viruses based on their behavior compared with the use of fuzzy logic conclusion. On the basis of behavior of metamorphic virus infection conclusion executable metamorfnym virus.

Keywords: metamorphic virus, vector of signs, behavior.

Вступ

Стрімкий розвиток комп'ютерної техніки супроводжується пропорційним розвитком комп'ютерних вірусів. Факти несанкціонованого доступу чи пошкодження функціонування комп'ютерних систем (КС) свідчать, що сучасні антивірусні технології орієнтовані на виявлення існуючих шкідливих програм, зокрема метаморфних вірусів, проте не здатні в повному обсязі виявляти нові шкідливі сімейства метаморфних вірусів чи видозмінені копії вже існуючих [6].

Постановка задачі

Класичні методи виявлення, що засновані на сигнатурному аналізі, не здатні в повному обсязі протидіяти вірусній загрозі [1,2,4]. Дослідження Університету Сан Хосе, показали, що багато антивірусних засобів покладаються на сигнатурний пошук, та не здатні в повному обсязі виявляти нові метаморфні віруси [5].

Для виявлення метаморфних вірусів було запропоновано ряд рішень. У роботі [8] було запропоновано евристичний статичний метод виявлення метаморфних вірусів, на основі побудови графа потоку управління довільної довжини. Даний метод показав високі результати виявлення тестової вибірки. Проте, згаданий метод був застосований тільки до двох сімейств метаморфних вірусів (NGVCK та VCL32), причому тестова вибірка включала 60 зразків шкідливих програм, що є не достатнім для отримання достовірних даних.

Інший підхід базується на отриманні послідовності API викликів [9] шкідливого ПЗ. Запропонований метод використовує алгоритм вирівнювання оптимальної послідовності API викликів та порівняння їх схожості за допомогою косинус схожості, коефіцієнта Жакара та кореляції Пірсона. Основним недоліком даного методу є значний відсоток хибних спрацювань, оскільки схожість API викликів різних сімейств метаморфних вірусів є досить високою.

У роботі [3] представлено метод виявлення та класифікації шкідливого програмного забезпечення, що заснований на частоті появи інструкцій. Головним недоліком даного методу є низький відсоток виявлення метаморфних вірусів, що використовують техніку переміщення блоків.

Тому, є необхідність в розробці моделі інформаційної технології виявлення метаморфних вірусів, яка б стала основою розробки нових методів і засобів, які дозволили б отримати більш високий ступінь достовірності.

Об'єктом дослідження є структура,механізми функціонування, життєвий цикл поліморфних та метаморфних вірусних програм, що поширюються в PE EXE файли комп'ютерної системи та методи їх виявлення.

Предметом дослідження є інформаційна технологія діагностування комп'ютерних систем на наявність поліморфного та метаморфного програмного коду.

Основна частина

Виявлення поліморфних та метаморфних вірусів є одним з найбільш складних завдань, у зв'язку з використанням ними обфускації програмного коду, використання технології невизначеної точки входу (Entry-Point Obscuring) та антивідлагоджувальними техніками. Поліморфні та метаморфні віруси відносяться до множини шифрованих вірусів.

Шифровані віруси – віруси, що самі шифрують власний код для утруднення їх дизасемблювання і виявлення у файлі, пам'яті або секторі. Кожен екземпляр такого вірусу містить тільки короткий загальний фрагмент коду – процедуру розшифрування, яку можна використати в якості сигнатури, для їх виявлення. У разі кожного нового інфікування він автоматично зашифровує себе, і кожна нова його копія відрізняється від попередньої.

Поліморфний вірус (або вірус з самомодифікованими розшифровувачами), як і шифрований вірус, складається з зашифрованого тіла вірусу та процедури його розшифрування [5]. Проте, до складу поліморфних вірусів входить ще одна процедура – двигун мутації (mutation engine), що генерує довільні процедури розшифрування, що змінюються при кожному новому інфікуванні. Наприклад, він може шифрувати своє тіло використовуючи кожен раз різні ключі, та розшифровувати його під час активації. Цикл розшифрування мутує з кожним наступним поколінням вірусу.

На відмінну від поліморфних вірусів, метаморфні віруси можуть бути незашифровані. Функціонування метаморфного вірусу ґрунтується на здатності перекладати, редагувати та переписувати власний код при інфікуванні виконуваного файлу, що робить їх одними з найбільш небезпечних шкідливих програм [5]. Це є головною відмінністю від поліморфних вірусів, що шифрують власне тіло, для уникнення виявлення антивірусними засобами.

Запропонована модель заснована на тому, що, незважаючи на зміну шкідливого коду метаморфного вірусу, його поведінка та виконувані функції залишаються сталими.

Модель процесу складається з наступних кроків:

- дизасемблювання підозрілого коду *Fp*;
- аналіз та побудова вектора ознак для *Fp*;
- емуляція виконання *Fp* та отримання зразка коду *Fs*;
- дизасемблювання *Fs*;
- побудова вектора ознак для *Fs*;
- порівняння векторів ознак для *Fp* та *Fs*;
- збір векторів ознак зразків підозрілих кодів з мережі;
- здійснення нечіткого логічного висновку щодо приналежності підозрілого коду до множини метаморфних вірусів на основі бази моделей поведінок метаморфних вірусів.

Метаморфні віруси відносяться до класу програм, що змінюються при кожному новому інфікуванні. Проте, для уникнення виявлення, деякі зразки метаморфних вірусів здатні розпізнавати виконання у захищеному середовищі, шляхом перевірки апаратних складових системи (підтримка особливих наборів інструкцій, тип пам'яті, розрядність системи та ін) та програмного налаштування системи. При запуску програми їй виділяється визначений адресний простір, стек та інші структури даних, що динамічно змінюються в часі. Відповідно, одній і тій самій програмі, що запущена на одному хості кожного разу буде виділятися нові адреси у пам'яті при кожному новому запуску. Тому, емуляція виконання на кожному хості у корпоративній мережі однієї і тієї ж програми дозволить підвищити імовірність розгортання метаморфного вірусу і, відповідно, отримати змінені копії одного і того ж вірусу. Окрім того виконання емуляції на кожному хості дозволить розпаралелити процес, що знизить апаратні витрати кожної системи.

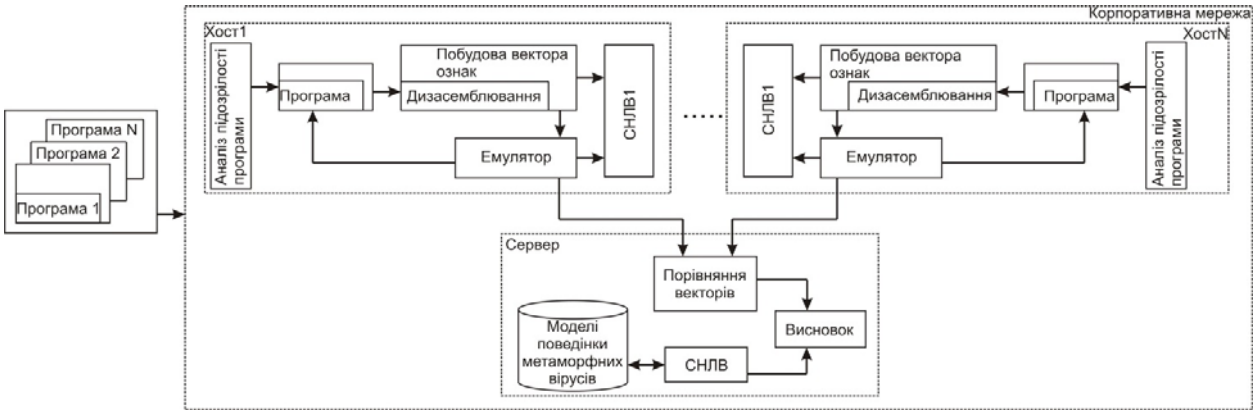

Рис 1 Узагальнена схема моделі виявлення метаморфних вірусів

Розглянемо детальніше кожну складову системи виявлення метаморфних вірусів у корпоративній мережі.

З метою аналізу підозрілої програми, на етапі дизасемблювання машинний код перетворюється на інструкції асемблера.

На основі отриманого коду, будується вектор ознак для підозрілого коду *Fp*, який може бути поданий наступним чином,

$$\overline{\Psi} = (C_T, C_A, C_L, C_S, C_M, C_J, C_{EP}, C_{ID}, C_{VA}, C_{NS}, C_{RVA}, \overline{\Theta}), \quad (1)$$

де $C_T, C_A, C_L, C_S, C_M, C_J$ – ознаки перевищення частки команд типу пересилки даних, арифметичних, логічних, роботи зі стеком, керування та переходів відповідно;

Для класифікації шкідливого коду авторами [4] було здійснено аналіз частоти появи інструкцій у виконуваних файлах PE EXE. В результаті було отримано відсоткове співвідношення команд певного типу до загальної кількості всіх команд. Було з'ясовано що 90,5% від загальної кількості всіх команд становлять 14 команд архітектури x8086 (mov, push, call, pop, cmp, jz, lea, test, jmp, add, jnz, rtn, xor, and).

Для нашого дослідження дані команди були розподілені на 6 груп: арифметичні, логічні, пересилки даних, керування, переходів, роботи зі стеком.

Таблиця 1

Умова активізації ознак перевищення частки команд типу пересилки даних, арифметичних, логічних, роботи зі стеком, керування та переходів

Команда		Вірус, %	Довірений додаток, %	Умова активізації ознаки вектора
Команди пересилки даних				
K_{mov}	mov	16.1	25.3	$\{(16.1 \le K_{mox} \le 25.3) \vee (K_{mov} < 16.1)\}$
K_{lea}	lea	5.5	3.9	$\{(3.9 < K_{lea} \le 5.5) \vee (K_{lea} > 5.5)\}$
Арифметичні команди				
K_{add}	add	3.5	3.0	$\{(3.0 < K_{add} \le 3.5) \vee (K_{add} > 3.5)\}$
Команди роботи зі стеком				
K_{push}	push	19.5	22.7	$\{(19.5 \le K_{push} < 22.7) \vee (K_{push} \ge 22.7)\}$
K_{pop}	pop	6.3	7.0	$\{(2.7 \le K_{pop} < 5.0) \vee (K_{pop} \ge 5.0)\}$
Логічні команди				
K_{and}	and	1.5	1.3	$\{(1.3 < K_{and} < 1.5) \vee (K_{and} \ge 1.5)\}$
K_{xor}	xor	2.1	1.9	$(1.9 < K_{xor} < 2.1)$
Команди керування				
K_{retn}	retn	2.0	2.2	$\{(2.0 < K_{retn} \le 2.2) \vee (K_{retn} < 2.0)\}$
K_{call}	call	9.1	8.7	$\{(8.7 < K_{call} \le 9.1) \vee (K_{call} > 9.1)\}$
Команди передачі керування				
K_{jmp}	jmp	2.7	3.0	$(2.7 \le K_{jmp} < 3.0)$
K_{jnz}	jnz	3.2	2.6	$\{(2.6 < K_{jnx} \le 3.2) \vee (K_{jnz} > 3.2)\}$

Значення ознаки перевищення частки команд пересилки даних C_T вектору $\overline{\Psi}$ можна визначити:

$$C_T = \begin{cases} 2, if\ (K_{mov} \in [16.1, 22.2)\ or\ K_{mov} < 16.1)\ and\ (K_{lea} \in [3.9, 5.5)\ or\ K_{lea} > 5.5) \\ 1, if\ (K_{mov} \in [16.1, 22.2)\ or\ K_{mov} < 16.1)\ and\ (K_{lea} \notin [3.9, 5.5)\ or\ K_{lea} < 5.5) \\ or\ (K_{mov} \notin [16.1, 22.2)\ or\ K_{mov} > 16.1)\ and\ (K_{lea} \in [3.9, 5.5)\ or\ K_{lea} > 5.5) \\ 0, otherwise \end{cases}$$

З урахуванням значень із таблиці 1, ознаки C_A, C_L, C_S, C_M, C_J визначаються аналогічним чином.

C_{EP} – наявність зміни точки входу програми; Одним із найбільш поширених методів інфікування файлів вірусом є зміна точки входу у програму, тобто заміщення оригінальної адреси з якої починається виконання файлу на адресу першої команди підпрограми вірусу, що відповідає за розшифрування основного тіла вірусу, у випадку зашифрованих вірусів, або на адресу першої команди тіла вірусу, якщо вірусне тіло незашифроване.

Точка входу у довірених додатків зазвичай розміщується у заголовку виконуваного файлу, тому поява її у іншій секції свідчить про інфікування вірусом EXE файлу.

Визначимо вектор зміни параметрів секції:

$$\overline{\Theta} = (S_N, Q_S, V_S, A)\ ,$$

де S_N – зміна назви секції виконуваного файлу

Q_S – зміна ознаки фізичного розміру секції виконуваного файлу

V_S – зміна ознаки віртуального розміру секції виконуваного файлу

A – зміна атрибуту секції виконуваного файлу

$$F = \begin{cases} S_i(S_N, Q_S, V_S, A) \in \overline{\Theta}, i = \overline{1, C_{NS}} \\ 1, if\ (Fp_{S_N} \neq Fs_{S_N})\ or\ (Fp_{Q_S} \neq Fs_{Q_S})\ or\ (Fp_{V_S} \neq Fs_{V_S})\ or\ (Fp_A \neq Fs_A) \\ 0, otherwise \end{cases}$$

Тобто зміна однієї з ознак вектору зміни параметрів секції свідчить про зміну структури виконуваного файлу і вектор зміни параметрів секції встановлюється в 1.

Таблиця 2

Порівняння векторів ознак

$Fp\ \forall(\psi_1, \psi_2, ..., \psi_n) \in \Psi, \psi_i = 1$

C_T				C_A		C_L				C_S				C_M				C_J			
Fp		Fs		Fs	Fp	Fp		Fs		Fp		Fs		Fp		Fs		Fp		Fs	
mov		lea		add		and		xor		push		pop		retn		call		jmp		jnz	
16,1	15,8	5,5	5,4	3,5	3,5	1,5	1,4	2,1	2,1	22,7	22,7	7	7,1	9,1	9,1	2	2	2,7	2,8	3,2	3,2

C_{EP}		C_{ID}		C_{VA}		C_{NS}		C_{RVA}	
Fs	Fp	Fs	Fp	Fs	Fp	Fs	Fp	Fs	Fp
0x1000	0x6000	C:\	C:\	0x4680	0x8680	3	3	0x5200	0x5200

C_{ID} – зміна поля DataDirectory структури Image_Optional_Header;

C_{VA} – зміна значення поля віртуальної адреси VirtualAddres;

C_{NS} – зміна значення кількості секцій EXE файлу NumberOfSection;

C_{RVA} – наявність зміни поля RVA значення в кожній структурі Image_Import_Descriptor.

Ознаки $C_T, C_A, C_L, C_S, C_M, C_J$ визначають наявність частки команд відповідного типу, що представлена у відсотках, від загальної кількості команд підозрілого файлу, відносно довірених додатків [4],

Якщо частка для підозрілого коду перевищує визначене порогове значення, то відповідний елемент вектора позначається “1”, інакше – “0”.

Після отримання вектору ознак з емулятора необхідно порівняти отримане значення із вектором ознак, що сформувався після дизасемблювання.

В реальних умовах з метою імітації виконання підозрілого коду використовується емулятор. Використання емулятора надасть можливість отримати змінені версії метаморфного вірусу *Fs*.

Для аналізу підозрілої програми *Fs* та створення вектору ознак, код після емуляції виконання перетворюється у машинний код.

На наступному кроці будується вектор ознак для коду, отриманого після емуляції виконання.

На наступному етапі відбувається порівняння векторів ознак для *Fp* та *Fs*, і якщо вони відрізняються, то формується висновок про наявність метаморфного вірусу в системі.

Проте, деякі сімейства метаморфних вірусів здатні розпізнавати здійснення емуляції в віртуальному середовищі та приховувати свою шкідливу діяльність. Тому на наступному етапі висновок щодо приналежності підозрілого коду до множини сімейств метаморфних вірусів здійснюється за допомогою системи нечіткого логічного висновку, яка заснована на базі моделей поведінок метаморфних вірусів.

Моделі поведінки представлені базою ознак прояву функцій шкідливого коду. Ознаки проявів функцій шкідливого коду, представлені шістьма рівнями моделей поведінок поліморфних (метаморнфних) вірусів [7].

Приймемо модель метаморфного вірусу кортежем:

$$M_6 = (A, E, U, C, R)$$

де A - множина команд певної програми, яка може бути інфікована вірусом, $A = \{a_1, ..., a_n\}$; E - множина вірусних команд розшифровувача, $E = \{e_1 ... e_\theta\}$; U- множина шкідливих команд (тіло віруса), $U = \{u_1, ..., u_w\}$; C - функція утворення поведінки поліморфного вірусу R шляхом розміщення команд програми a_i, команд розшифровування, команд тіла вірусу блоками в певному порядку, $C: A \times E \times U \to R$; функція утворення поліморфного вірусу R без вкорінення у певну програму матиме вигляд: $C: E \times U \to R$.

Поведінки вірусу шостого рівня поліморфізму R_6^A та R_6 можуть бути представлені послідовностями: $R_6^A = a_1 e_\phi ... a_i e_\eta a_{i+1} u_\vartheta ... a_n u_\sigma$; $R_6^A = a_1 e_\phi u_\vartheta ... a_n e_\eta u_\sigma$; $R_6 = e_\phi ... e_\eta u_\vartheta ... u_\sigma$; $R_6 = e_\phi u_\vartheta ... e_\eta u_\sigma$, де значення $\phi, \eta, \vartheta, \sigma$ визначають, що можливі команди розшифровувача та шкідливі

команди $e_\phi ... e_\eta u_\vartheta ... u_\sigma$ можуть бути різними для кожного нового запуску вірусу.

На наступному етапі проводиться збір векторів ознак зразків підозрілих кодів з мережі.

Висновок

Запропоновано модель інформаційної технології виявлення метаморфних вірусів на основі порівняння векторів ознак підозрілого файлу з його копією після емуляції виконання, та формування нечіткого логічного висновку на основі моделей поведінки метаморфних вірусів. Запропонована модель є основою реалізації методу виявлення нових метаморфних віусів або копій вже існуючих.

Література

1. Ször P. Striking Similarities: Win32/Simile and Metamorphic Virus Code, white paper. – Symantec Security Response, 2003.

2. Rad B.B. Metamorphic Virus Variants Classification Using Opcode Frequency Histogram / Rad B. B., Masrom M. – Proc. ICCOMP 10 the 14th WSEAS international conference of computers, 2011. – pp. 147-155.

3. Zhang Q. MetaAware: Identifying Metamorphic Malware / Q. Zhang, Douglas S. Reeve – Proc. Twenty Third Annual IEEE Conference on Computer Security Applications, 2007. – pp. 411-420

4. Bilar D. Statistical structure: fingerprinting malware for classification and analysis / D. Bilar – Proceding of black hat, 2008.

5. Kaspersky Lab What a metamorphic virus – Definition [Електронний ресурс] режим доступу: http://usa.kaspersky.com/internet-security-center/definitions/metamorphic-virus#.VkO9VdLhDIU

6. E. Al Daoud Computer virus strategies and detection methods / E. Al Daoud, I. H. Jebril, B. Qaibeh Open Problems compt., math., vol. 1 No.2 September 2008

7. Лисенко С. М. Метод виявлення поліморфного коду ботів ботнет-мереж / С. М. Лисенко, О. С. Савенко, А. О. Нічепорук // Радіоелектрон. і комп'ют. системи. - 2014. - № 5. - С. 129-134.

8. E. Al Daoud et al Detecting Metamorphic viruses by using Arbitrary Length of Control Flow Graphs and Nodes Alignment / Daoud E. Al. – UbiCC Journal, Vol. 4, No 3, pp.628–633, 2009.

9. A.H. Sung / Static analyzer of vicious executables (SAVE) A.H. Sung Proc. 20th Annual Computer Security Applications Conference, 2004.

Рецензія/Peer review : 4.11.2015 р. Надрукована/Printed :5.12.2015 р.
Рецензент: д.т.н., проф. Мартинюк В.В.

УДК 004.491.2

К.Ю. БОБРОВНІКОВА
Хмельницький національний університет

МОДЕЛЬ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЯВЛЕННЯ БОТ-МЕРЕЖ НА ОСНОВІ АНАЛІЗУ DNS-ТРАФІКА

В роботі запропоновано модель інформаційної технології виявлення бот-мереж, яка ґрунтується на аналізі ознак, вилучених з корисного навантаження DNS-повідомлень. Модель заснована на властивості групової активності ботів в DNS-трафіку, враховує особливості поведінки груп хостів, властиві бот-мережам, та застосовує кластерний аналіз векторів ознак, які вказують на використання бот-мережами технологій ухилення від виявлення на основі DNS.

Ключові слова: бот-мережа, DNS-трафік, групова активність в DNS-трафіку, технології ухилення бот-мереж

K.Y.BOBROVNIKOVA
Khmelnitsky National University, Khmelnitsky, Ukraine

THE MODEL OF INFORMATION TECHNOLOGY FOR BOTNETS DETECTION BASED ON DNS-TRAFFIC ANALYSIS

Abstract - The model of information technology for botnets detection that based on an analysis of the features obtained from the payload of DNS-messages was proposed. The model is based on the property of bots group activity in the DNS-traffic, takes into account abnormal behaviors of the hosts' group, which are similar to botnets, and uses a cluster analysis of the feature vectors, which indicate the botnet's usage the DNS-based evasion techniques.

Keywords: botnet, DNS-traffic, group activity in DNS-traffic, botnet`s evasion techniques

Вступ

Бот-мережі є одним з найбільш небезпечних видів шкідливого програмного забезпечення. Щороку по всьому світу бот-мережами інфікується близько 500 млн. персональних комп'ютерів, кожну секунду – близько 18 [1]. Бот-мережі використовуються для здійснення DDoS-атак, поширення шкідливого