УДК 004.67
Савенко О.С., к.т.н, Лисенко С.М., к.т.н., Нічепорук А.О., асп.
Хмельницький національний університет

МЕТОД ВИЯВЛЕННЯ МЕТАМОРФНИХ ВІРУСІВ У КОРПОРАТИВНІЙ МЕРЕЖІ НА ОСНОВІ МОДИФІКОВАНИХ ЕМУЛЯТОРІВ

Савенко О.С., Лисенко С.М., Нічепорук А.О. Метод виявлення метаморфних вірусів у корпоративній мережі на основі модифікованих емуляторів. У статті представлено метод виявлення метаморфних вірусів з використанням мережних модифікованих емуляторів. Ідея методу полягає у формуванні оцінки схожості копій метаморфного вірусу, що досягається шляхом створення на кожному хості свого модифікованого мережного емулятора. Отримані копії метаморфних вірусів порівнюються з використанням метрики Дамерау-Левенштейна. Для формування висновку про інфікування системи метаморфним вірусом здійснюється класифікація отриманих копій з використанням моделей подібності метаморфних вірусів.
Ключові слова: метаморфний вірус, модифікований емулятор, відстань Дамерау-Левенштейна, опкоди.

Савенко О.С., Лысенко С.Н., Ничипорук А.А. Метод выявления метаморфных вирусов в корпоративной сети на основе модифицированных эмуляторов. В статье представлен метод обнаружения метаморфных вирусов с использованием модифицированных эмуляторов. Идея метода заключается в формировании оценки сходства копий метаморфный вируса, что достигается путем создания на каждом хосте своего модифицированного эмулятора. Полученные копии метаморфных вирусов сравниваются с использованием метрики Дамерау-Левенштейна. Для формирования вывода о инфицировании системы метаморфным вирусом осуществляется классификация полученных копий с использованием моделей сходства метаморфных вирусов.
Ключевые слова: метаморфный вирус, модифицированный эмулятор, расстояние Дамерау-Левенштейна, опкоды.

Savenko O.S., Lysenko S.M., Nicheporuk A.O. The method of identifying metamorphic virus in the corporate network based on modified emulators. The article presents a method of detection the metamorphic virus with using a modified network emulators. The idea of a method is to create the estimate of the similarity metamorphic copies of the virus, which is achieved by creating for each host his own modified emulator. Received copies of metamorphic viruses compare by using metrics Damerau-Levenshtein. To form the conclusion that the metamorphic virus infected systems, done the classification of obtained copies using models similarities of metamorfic viruses.
Keywords: metamorphic virus, modified emulator, distance Damerau-Levenshtein, opcodes.

Постановка проблеми. На сьогоднішній день виявлення комп'ютерних вірусів є одним з головних завдань інформаційної безпеки. Віруси завдають збитків як інформації так і самій робочій станції, що негативно позначається на робот всієї комп'ютерної системи вцілому. Серед всієї множини вірусних програм одне з провідних місць займають метаморфні віруси.

Виявлення метаморфних вірусів є складним завданням, у зв'язку з використанням ними техніки обфускації програмного коду. Використання обфускації дозволяє створювати різні копії одного і того ж вірусу. Сума збитків, що спричиняють віруси, зокрема метаморфні, сягають мільйонів доларів. За даними Symantec у 2011 метаморфний вірус Sality інфікував близько 3 мільйонів комп'ютерів у світі [1], а вже станом на кінець 2015 року, за даними ESET, увійшов в п'ятірку найбільш розповсюджених вірусів (1,43% від загальної кількості всіх виявлених загроз) [2], зберігаючи при цьому позитивну динаміку поширення.

Тому, актуальною постає задача розробки методу, що дозволить здійснювати виявлення нових метаморфних вірусів та копій вже існуючих, які здійснюють інфікування .PE .EXE файлів, шляхом приєднання власного коду до корисних програм (benign program).

Аналіз досліджень. Оскільки копії метаморфних вірусів змінюються від покоління до покоління, результати досліджень [3,6] показали неможливість виявлення метаморфних вірусів за допомогою сигнатурного аналізу.

Серед всіх відомих методів виявлення метаморфних вірусів можна виділити два основних підходи: статичні та динамічні методи виявлення. Статичні методи здійснюють аналіз підозрілого файлу без його безпосереднього виконання. Перевагою такої групи методів є трасування всіх можливих шляхів виконання програми, в той час як програма, що аналізується динамічними методами, може виконуватись тільки по одному шляху. Проте, у зв'язку з тим, що всі статичні методи передбачають попереднє дизасемблювання виконуваного файлу, вони не здатні здійснити аналіз виконуваного файлу, до якого було застосовано техніку шифрування або обфускації виконуваного коду [4].

Для відслідковування поведінки динамічні методи передбачають виконання підозрілого файлу. Одним з головних методів динамічного аналізу є емуляція виконання. Такі динамічні методи включають моніторинг API викликів, файловий моніторинг, монітор процесів, поведінковий аналіз та мережевий монітор [3-5]. Підозріла програма запускається у захищеному середовищі, що дозволяє здійснити аналіз її виконання. Проте, у випадку використання вірусними програмами антивідлашоджувальних та антиемуляційних технологій дані методи не здатні здійснити виявлення вірусу [6].

Обидва класи методів передбачають пошук сталих ознак. Цим ознаками можуть бути граф потоку управління програми, послідовність викликів API функцій, структурна інформація виконуваного файлу, опкоди підозрілої програми.

У роботі [7] запропоновано метод оцінки схожості метаморфних вірусів, шляхом побудови гістограми інструкцій для кожної підпрограми, з подальшим їх порівнянням за допомогою метрики city blocks. Проте, Описаний підхід є не ефективним для вірусів, що використовують техніку перемішування блоків (code transposition).

Іншим підходом виявлення метаморфних вірусів є формування метрики подібності метаморфних вірусів на основі графу потоку виконання програми [8]. Недоліком даного методу є відсутність оцінки послідовності API викликів.

Отже, аналіз предметної області показав необхідність у розробці та вдосконалені існуючих методів виявлення метаморфних вірусів.

Метод виявлення метаморфних вірусів у корпоративній мережі на основі модифікованих емуляторів. Запропонований метод передбачає порівняння зразка коду F_P метаморфного вірусу з його копію F_S ,що отримана з використанням модифікованих емуляторів, шляхом емуляції виконання на кожному хості у мережі підозрілої програми *P*. На рис.1 наведено узагальнену схему методу виявлення метаморфних вірусів у корпоративній мережі.

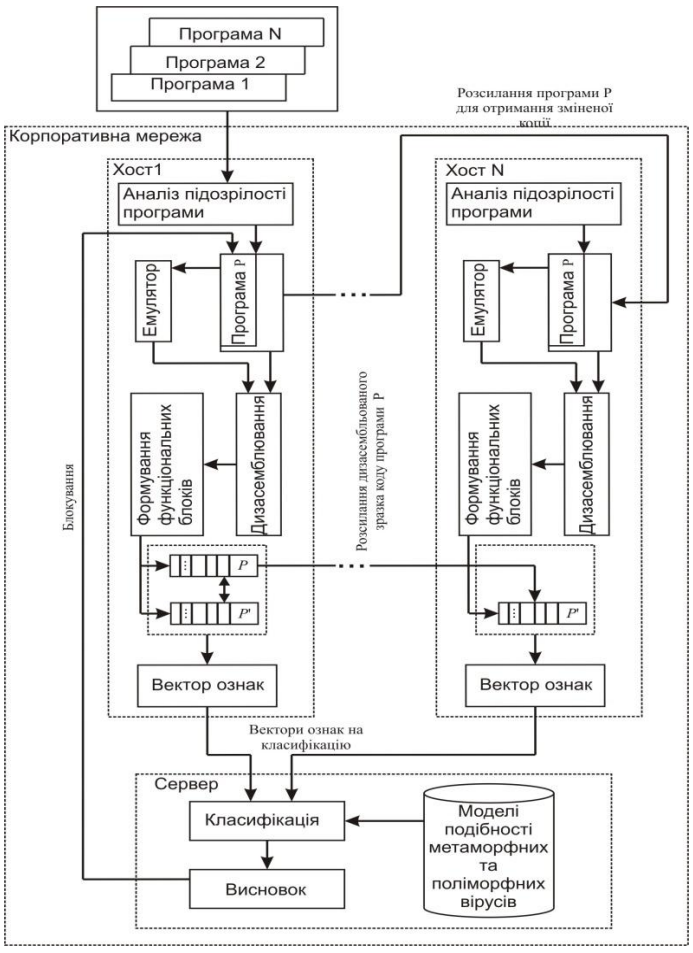


Рис. 1. Узагальнена схема методу виявлення метаморфних вірусів у корпоративній мережі

Для виявлення підозрілих дій на кожному хості корпоративної мережі використовується аналізатор підозрілості програми.

Кожна окрема дія, що виконується підозрілою програмою, не є небезпечною. Проте, виконання певної послідовності таких дій може свідчити про можливу небезпеку інфікування вірусом. Наприклад, якщо деяка програма записує себе в ключ системного реєстру автозавантаження, зчитує дані введені з клавіатури та з певним інтервалом надсилає повідомлення за певною адресою, то це свідчить про потенційну загрозу.

Кожна програма, що надходить в систему маркується як підозріла чи непідозріла, тобто:

$$P=\{suspicious, non\text{-}suspiciuous\}$$

Подамо вектор ознак, що визначає належність програми до одного з двох класів наступним чином:

$$\overline{U} = (M, Q, J, Y, L, N, H) \quad (1)$$

де, M – спроба програми отримати права адміністратора системи, Q – спроба відкриття або закриття системного порту, J – спроба видалення файлу, Y – створення файлу або процесу, L – перехоплення даних, що вводяться з клавіатури, N – розсилка повідомлень в мережу, H – створення або запис в системний реєстр.

Кожна ознака приймає значення 0 або 1, де 1 свідчить про активізацію відповідної ознаки, 0 – навпаки. Програму вважатимемо за підозрілу, якщо:

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

Програма, для якої $P=suspicious$ надходить в систему виявлення метаморфного коду.

Для отримання зміненого зразку коду F_S, здійснюється емуляція виконання програми P. Процес емуляції виконання представляє собою розбір програмного коду на інструкції та імітацію їх виконання у віртуальному середовищі.

Використання однотипного емулятора на всіх хостах мережі не дозволить з високим ступенем достовірності здійснити виявлення метаморфних вірусів, оскільки і використання однакових емуляторів дозволить отримати лише однакові зразки коду. Для прояву метаморфних властивостей необхідним є забезпечення різних умов виконання шкідливого програмного коду. Тому, на кожному хості створюються модифіковані емулятори.

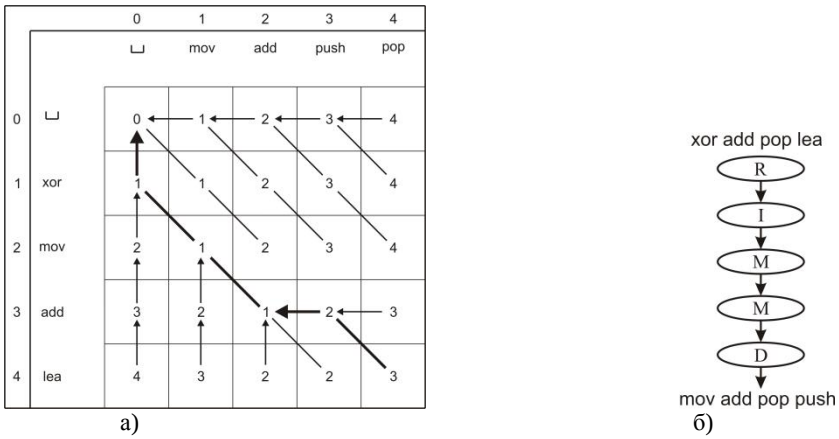
Структура емулятора наступна: віртуальний процесор, визначає набори інструкцій, що доступні для роботи (MMX, SSE, SSE2 та ін.) та включає в себе набори віртуальних регістрів; оперативна пам'ять та віртуальний стек; віртуальний мережний контролер; тип ОС (підтримка API-функцій, системного реєстру та портів) та модуль евристики.

Для протидії антиемуляційним технологіям, що використовуються у метаморфних вірусах, у емуляторі використовується модуль евристики. Для кожної операції, що виконується віртуальним CPU встановлюється фіксований час обробки та здійснюється перевірка повторів виконання певної операції (наприклад, якщо в тілі вірусу є цикл, що здійснює операцію інкременту змінної велику кількість разів).

З метою отримання початкового зразку коду F_P, здійснюється дизасемблювання програми P. Результатом процесу дизасемблювання є множина асемблерних інструкцій x86/x64. Для побудови вектора ознак, з метою спрощення реалізації, використовуються лише опкоди інструкцій, операнди відкидаються.

Отриманий лістинг дизасембльованих інструкцій розбивається на функціональні блоки (ФБ). Кожен ФБ складається з інструкцій, що розташовані між інструкціями переходів (jz, jnz, jmp та ін.).

З огляду на механізми створення метаморфними вірусами власних копій, що використовують техніки вставки, видалення та переміщення власних інструкцій, для пошуку схожості між ФБ двох зразків коду F_P та F_S використовується дистанція Дамерау – Левенштейна. Для пошуку дистанції Дамерау – Левенштейна використаємо алгоритм поліноміальної складності Вагнера-Фішера [9], який дає змогу сформувати найкоротший ланцюг перетворення, для приведення множини опкодів програми після емуляції у множину опкодів програми до емуляції.



а) б)

Рис. 2. Матриця Дамерау-Левенштейна для двох ФБ (а) та ланцюг перетворення ФБ1 в ФБ2 (б)

Розглянемо програму *P*, яка складається з множини асемблерних команд p_i, тобто $P=\{p_1, p_2,..., p_k\}$. Розіб'ємо програму *P* на функціональні блоки довільної довжини, що починаються із інструкцій умовного переходу jmp, jz та ін. і закінчуються ними, тобто $P=\{B_1, B_2,..., B_l\}$. Тоді можна записати: $P=\{B_1=\{p_1, p_2,..., p_{i-1}\},..., B_l=\{p_i, p_{i+1},..., p_k\}\}$.

Нехай функціональний блок *B*, що складається з множини опкодів, довжиною $|B|=m$ записується як $p_1, p_2,..., p_m$, де p_i представляє *i*-й опкод *p*. Підмножина опкодів $x_i, x_{i+1},..., x_j$, функціонального блоку *B* буде позначатись $B(i, j)$.

Вага перетворення опкода *a* в опкод *b* позначимо через $w(a,b)$. Таким чином, $w(a,b)$ – вага заміни одного опкоду на другий опкод, коли $a \neq b$, $w(b,a)$ – вага операції транспозиції, $w(a,\varepsilon)$ – вага видалення, а $w(\varepsilon,b)$ – вага вставки *b*.

В базовому алгоритмі вага всіх операцій дорівнює одиниці. В запропонованому алгоритмі вага операцій вставки та видалення опкоду становить 2, операція обміну – 1, а операції заміни одного опкоду на інший визначається наступними правилами:

$$w(a_i,b_j)=\left\{\begin{array}{c} 1, if\ (a_i=mov; b_j=push, b_{j+1}=pop)\ or \\ ((a_i=mov)\ or\ (a_i=xor)\ or\ (a_i=and)\ or\ (a_i=sub) \\ (b_j=mov)\ or\ (b_j=xor)\ or\ (b_j=and)\ or\ (b_j=sub))\ or \\ ((a_i=jnz)\ or\ (a_i=jz)\ or\ (a_i=jmp) \\ (b_j=jnz)\ or\ (b_j=jz)\ or\ (b_j=jmp)), a_i\in B^{F_P}, b_j\in B^{F_S}, a_i\neq b_j \\ 2, otherwise \end{array}\right\} \quad (2)$$

Нехай B_g та B_h – два ФБ, що складаються з послідовності опкодів (довжиною n та m відповідно) над скінченим алфавітом асемблерних інструкцій $A=(a_1, a_2,..., a_k)$, причому B_g ФБ програми F_P, позначимо $B_g^{F_P}$, а B_h – ФБ тієї ж програми після емуляції виконання F_S, позначимо $B_h^{F_S}$. Тоді відстань Дамерау – Левенштейна $dL(B_g^{F_P}, B_h^{F_S})$ обчислимо наступним чином:

$dL(B_g^{F_P}, B_h^{F_S})=OPT(N,M)$, де

$$OPT=\left\{\begin{array}{ll} 0, & i=0,\ j=0 \\ i, & j=0,\ i>0 \\ j & i=0,\ j>0 \\ \min\left\{\begin{array}{l} OPT(i, j-1)+w(a,\varepsilon) \\ OPT(i,-1j)+w(\varepsilon,b) \\ OPT(i,-1, j-1)+w(a,b) \\ OPT(i-2, j-2)+w(b,a) \end{array}\right. & j>0, i>0 \end{array}\right. \quad (3)$$

Після отримання відстані Дамерау-Левенштейна для двох блоків B_g та B_h, формується зважене усереднене значення відповідного параметру вектора ознак для всіх блоків коду. Для отримання зваженої усередненої оцінки параметрів використаємо показник центру розподілу зважене середнє арифметичне (4):

$$dL = \left\lfloor \frac{\sum_{i=1}^{n} dL_i * f_l}{\sum_{i=1}^{n} f_l} \right\rfloor \quad (4)$$

де,

dL_i – відстань Левенштейна для ФБ B_i, f_l – кількість ФБ з значенням dL_i

Оскільки відстань Дамерау-Левенштейна визначає мінімальне значення необхідних операції заміни, вставки, видалення та транспозиції, що відповідно є цілочисельним значенням, отримана ознака округлюється в меншу сторону, для знаходження найменшої різниці між копіями метаморфних вірусів.

Для решти ознак (кількість операцій співпадіння, вставки, видалення, заміни, транспозиції) унормування відбувається аналогічним чином.

Таким чином, вектор ознак схожості копій метаморфних вірусів на основі метрики Дамерау-Левенштейна подамо наступним чином:

$$\overline{S} = \langle dL, T, D, I, R, M \rangle \quad (5)$$

де, T – кількість необхідних операцій обміну опкодів для перетворення блоку програму F_p у F_s ($F_p = F_s$);

D – кількість необхідних операцій видалення опкоду;

I – кількість необхідних операцій вставки опкоду;

R – кількість необхідних операцій заміни відповідних опкодів;

M – кількість співпадінь між опкодами в функціональному блоці програми F_p та F_s.

Для формування нечіткого логічного висновку про інфікування системи метаморфним вірусом, отримані вектори ознак схожості надходять на серверну частину для їх класифікації. Результатом роботи системи є ступінь приналежності кожної копії до одного із сімейств метаморфних вірусів.

Експерименти. Для тестування та генрації копій метаморфних вірусів [12] було використано такі метаморфні генератори: Next Generation Virus Creation Kits (NGVCK), Second Generation Virus Generator (G2) та Virus Creation Lab for Win32 (VCL32). Тестування системи проводилось у корпоративній мережі, що складається з 80 комп'ютерів.

На кожному хості мережі були створені модифіковані мережні емулятори, параметри яких представлено в таблиці 1.

Таблиця 1. Параметри модифікованих мережних емуляторів для хостів мережі

Номер хоста	Початкові параметри мережних модифікованих емуляторів							
	Набір інструкцій	Тип ОС	Розмір віртуальної ОЗП	Системна дата	Віртуальний мережевий контролер	Час виконання емуляції інструкції (xor), од	Адреса початку емуляції	Архітектура ЦП
1	SSE2	W7	8	25.11	Так	1	0x001A4810	x86
2	Hyper-threading	W8.1	8	06.02	Ні	2	0x001A4620	x86
…	…	…	…	…	…	…	…	
N	MMX	W10	4	10.04	Ні	1	0x001A5214	x64

Для здійснення висновку про інфікування метаморфним вірусом використовується система нечіткого логічного висновку.

Система нечіткого логічного висновку побудована за допомогою пакету Fuzzy Logic Toolbox системи Mathlab. Для проведення досліджень в системі були задіяні наступні параметри: алгоритм – Мамдані, метод агрегування, метод акумуляції, метод дефазифікації. Система складається з 6 входів та

одного виходу. В якості функцій приналежності для входів було обрано трапецевидну, для виходу – трикутну. Кожна ознака вектора схожості копій метаморфних вірусів є вхідною лінгвістичною змінною для системи нечіткого логічного висновку. Для кожної лінгвістичної змінної задано терм множину Low, Medium та High.

Результатом роботи системи нечіткого логічного висновку є ступінь приналежності об'єкту до одного з чотирьох класів – трьох шкідливих та одного класу довірених додатків. Якщо ступінь приналежності невідомого об'єкту належить діапазону від 0 до 0,25, то невідомий об'єкт класифікується як довірений додаток; якщо ступінь приналежності лежить на проміжку від 0,26 до 1, то невідомий об'єкт відноситься до одного з класів метаморфних вірусів. Значення від 0,26 до 0,49 визначають 1 перший клас метаморфний вірусів, значення від 0,5 до 0,74 – 2 клас, значення від 0,75 до 1 – 3 клас. У таблиці 2 наведено результати нечіткого логічного висновку для підозрілого файлу. В результаті 15% копій не змінились, 5% віднесено до першого класу метаморфних вірусів, 11,25 – до третього та 68, 75% до другого.

Таблиця 2. Результати тестування системи

№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ	№ хоста	НЛВ
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								

Якщо підозрілий об'єкт віднесено до четвертого класу (значення від 0 до 0,25), то це свідчить, що даний зразок може бути довіреним додатком або метаморфним вірусом і потребує подальшого дослідження.

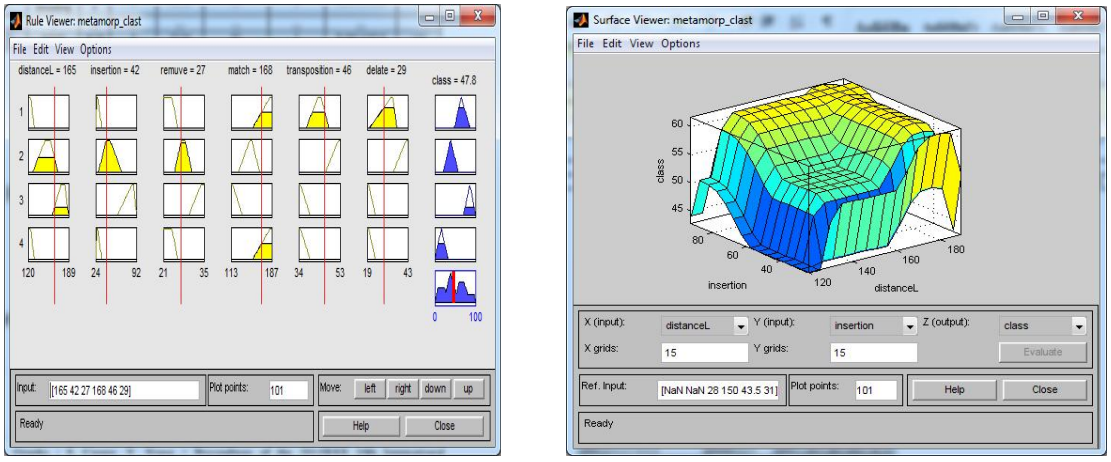


Рис. 3. Результат нечіткого логічного висновку для вектора схожості копій метаморфного вірусу.

Висновки. Аналіз предметної області показав необхідність у вдосконалені існуючих методів виявлення метаморфних вірусів. Авторами запропоновано метод виявлення метаморфних вірусі з використанням модифікованих мережних емуляторів на хостах локальної мережі. Класифікація вірусів здійснюється на основі системи нечіткого логічного висновку. Такий підхід дозволяє підвищити достовірність виявлення метаморфних вірусів. Віднесення досліджуваного програмного забезпечення до одного з класів дозволяє стверджувати, що таке програмне забезпечення є метаморфних вірусом, а не корисним додатком, до якого застосована техніка обфускації програмного коду. Було проведено ряд експериментів, і виявлено 85% копій, що були віднесені до одного з трьох класів метаморфних вірусів.

Список використаних джерел

1. Falliere, N.: Sality: Story of a Peer-to-Peer Viral Network. Technical report, Symantec Labs (2011)
2. Вирусная статистика. Обзор киберугроз ноября 2014 года [Електронний ресурс]. – режим доступу: https://www.esetnod32.ru/company/viruslab/statistics/?id= 896397
3. Alsagoff, S.N Malware self protection mechanism issues in conducting malware behavior analysis in a virtual environment as compared to a real environment. In: Proc. of International Symposium in Information Technology (ITSim), pp 1326-1331 (2010)
4. Bayer, U., Kirda, E., Kruegel C. Improving the Efficiency of Dynamic Malware Analysis. In: Proc. of 25th Symposium on Applied Computing (SAC), New York, pp. 1871-1878 (2010)
5. Park, Y., Reeves, D. S. Identification of Bot Commands by RunTime Execution Monitoring. In: Proc. of Security Applications Conference, ACSAC'09, Honolulu, pp. 321-330 (2009)
6. Peidai, X., Xicheng, L., Yongjun, W., Su, J. Eliminate Evading Analysis Tricks in Malware using Dynamic Slicing. In: International Journal of Security and Its Applications, vol.7, pp. 245-256 (2013)
7. Rad, B. B., Masrom, M.: Metamorphic Virus Variants Classification Using Opcode Frequency Histogram. In: 14th WSEAS international conference of computers, pp. 147-155, WSEAS (2010)
8. Cesare, S., Xiang, Y.: Malware variant detection using similarity search over sets of control flow graphs. In: 10th International Conference on Trust, Security and Privacy in Computing and Communications, Washington, DC, USA, pp. 181-189 (2011)
9. Wagner, R., Fisher, M.: The string to string correction problem. Journal of the ACM, 21, pp. 168-178 (1974)