УДК 621.391

С. М. ЛИСЕНКО, О. С. САВЕНКО, А. О. НІЧЕПОРУК

Хмельницький національний університет, Хмельницький, Україна

МЕТОД ВИЯВЛЕННЯ ПОЛІМОРФНОГО КОДУ БОТІВ БОТНЕТ-МЕРЕЖ

Запропоновано метод виявлення нових ботнет-мереж, боти яких використовують поліморфний код. Виявлення виконується на базі залучення мульти агентної системи засобами антивірусних агентів, що містять множину сенсорів. В роботі розглянуто та досліджено рівні поліморфізму, а також побудовано їх моделі. Запропоновано включення нового сенсора, на якого покладено функцію виявлення поліморфного коду. Виходячи з властивостей поліморфних вірусів даний сенсор виконує провокативні дії по відношенню до ймовірно інфікованого файлу, повторні запуски підозрілого файлу для ймовірної модифікації власного коду, виявлення ботів шляхом аналізу їх поведінки та можливої зміни свого тіла, що базується на принципах відомих рівнів поліморфізму.

Ключові слова: *бот, ботнет-мережа, поліморфний код, рівні поліморфізму, мульти-агентна система, агент, сенсор*

Вступ

Сьогодні гостро стоїть питання кібер безпеки, оскільки задача захисту даних надзвичайно актуальна. Найбільш небезпечним явищем в сфері розробки шкідливого програмного забезпечення (ШПЗ) є ботнет-мережі. Для приховування присутності ботнет-мережі активно використовують технології поліморфізму, яка передбачає мутацію коду в процесі їх функціонування. Сучасні антивірусні методи використовують різні методи для виявлення шкідливих файлів [1-3], проте вони демонструють високий відсоток помилкових спрацювань.

1. Попередні дослідження

Для виявлення нових ботнет-мереж в [4] запропоновано метод визначення рівня присутності ботнет-мережі шляхом аналізу проявів дій ботів. Виявлення здійснювалося на базі мульти-агентної системи (МАС), агенти якої виконували антивірусне діагностування в кожній комп'ютерній системі (КС). Розроблена система демонструвала 88-96% ефективності виявлення ботнет-мереж.

Зважаючи на тенденції розвитку ботнет-мереж, а саме активного використання технологій приховування шкідливого коду, була протестована вказана вище МАС на предмет її ефективності виявлення ШПЗ, що використовують вказані технології. Виявилося, що система не повністю адаптована до виявлення поліморфного коду: ефективність знизилася на 7-12%.

Таким чином, постає задача побудови нового методу виявлення ботнет-мереж, боти яких містять поліморфний, та його інтеграції в розроблену МАС.

2 Метод виявлення поліморфного коду ботів ботнет-мереж

Для побудови методу виявлення поліморфного коду необхідно дослідити його природу та властивості.

2.1 Рівні поліморфізму

Сьогодні відомо 6 рівнів поліморфізму [5]. Подамо моделі усіх рівнів поліморфізму.

Віруси ***першого*** рівня поліморфізму використовують постійні значення для різних розшифровувачів. Їх можна виявити за деякими постійними ділянкам коду розшифровувача.

Приймемо модель поліморфного вірусу першого рівня кортежем:

$$M_1 = (A, X, G, V, U, \xi, Q, P, R),$$

де A - множина команд певної програми, яка може бути інфікована вірусом, $A = \{a_1, ..., a_n\}$;

V – множина команд вірусу для вибору одного з присутніх у вірусі розшифровувачів, $V = \{v_1, ..., v_m\}$;

X - множина розшифровувачів присутніх у вірусі, $X = \{x_1, ..., x_y\}$;

G – множина команд вірусу x_i розшифровувача, $G = \{g_1, ..., g_\theta\}$;

U – множина шкідливих команд (тіло вірусу),

$U = \{u_1,...,u_W\}$;

ξ - функція вибору x_i розшифровувача, $\xi : V \to X$;

Q - функція створення шкідливих команд (тіла вірусу) шляхом виконання команд $g_{x_i \in G}$ розшифровувача x_i, $Q : G_{x_i} \to U$;

P - функція створення поведінки поліморфного вірусу R шляхом вкорінення тіла вірусу U в команди програми A, $P : A \times U \to R$; функція утворення поведінки поліморфного вірусу R без вкорінення шкідливих команд U в команди програми A шляхом розшифрування одним з розшифровувачів x_i тіла U матиме вигляд: $Q : U \to R$. Таким чином, поліморфний вірус має поведінку, що формується з певної послідовності команд. Базуючись на цьому можна побудувати поведінку вірусу у вигляді послідовності.

Поведінка вірусу R_1^A першого рівня поліморфізму, який утворений вкорінення шкідливих команд U в команди програми A, і поведінка вірусу R_1 утворена без вкорінення шкідливих команд U в команди програми A можна представити послідовностями:

$$R_1^A = g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} a_1...a_n u_1...u_W;$$

$$R_1 = g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} u_1...u_W,$$

де значення ϕ, η визначають, що можливі вірусні команди розшифровувача $g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}}$ можуть бути різними для різних розшифровувачів x_ε, ε – номер обраного розшифровувача

До ***другого*** рівня поліморфізму відносять віруси, розшифровувачі яких мають постійну одну або декілька інструкцій. Наприклад, він може використовувати різні регістри, деякі альтернативні інструкції в розшифровувачі. Такі віруси також можна розпізнати за визначеною сигнатурою в розшифровувачі [5].

Приймемо модель поліморфного вірусу другого рівня кортежем:

$$M_2 = (A, E, U, P, Z, R)$$

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 вірусу), $U = \{u_1,...,u_W\}$; Z - функція утворення шкідливих команд (тіла вірусу) шляхом вибору команд розшифровувача, $Z : E \to U$; P - функція утворення поведінки поліморфного вірусу R шляхом вкорінення тіла вірусу U в програму A, $P : A \times U \to R$; функція утворення поліморфного вірусу R без вкорінення у певну програму матиме вигляд: $Z : E \times U \to R$.

Поведінки вірусу другого рівня поліморфізму R_2^A і R_2 можна представити у вигляді послідовностей: $R_2^A = e_\kappa...e_\lambda a_1...a_n u_1...u_W$; $R_2 = e_\kappa...e_\lambda u_1...u_W$, де значення κ, λ визначають, що можливі вірусні команди розшифровувача $e_\kappa...e_\lambda$ можуть бути різними при кожному запуску вірусу.

Віруси, що використовують в розшифровувачі команди, що не приймають участі в розшифруванні вірусного коду, чи "команд-сміття", відносять до ***третього*** рівня поліморфізму. Дані віруси також можна виявити за допомогою деякої сигнатури, якщо провести відсіювання всіх команд-сміття.

Віруси ***четвертого*** рівня використовують в розшифровувачі взаємозамінювані інструкції без зміни алгоритму розшифрування.

Приймемо модель поліморфного вірусу третього та четвертого рівнів кортежем:

$$M_{3,4} = (A, E, U, B, Y, D, R)$$

де A - множина команд певної програми, яка може бути інфікована вірусом, $A = \{a_1,...,a_n\}$; E - множина вірусних команд розшифровувача, $E = \{e_1...e_\theta\}$; $U = \{u_1,...,u_W\}$ - множина шкідливих команд (тіло вірусу); B - множина «команд-сміття», $B = \{b_1,...,b_t\}$; Y - функція утворення тіла вірусу засобами розшифровувача, який інтегрує «команди-сміття» в множину шкідливих команд, $Y : E \times B \to U$; D - функція утворення поведінки поліморфного вірусу R шляхом вкорінення тіла вірусу U в програму A, $D : A \times U \to R$; функція утворення поведінки поліморфного вірусу R без вкорінення у певну програму матиме вигляд: $Y : E \times B \to R$.

Поведінка вірусу третього та четвертого рівнів поліморфізму R_3^A та R_4^A, які утворені засобами розшифровувача, який інтегрує «команди-сміття» в шкідливі команди і вставляє шкідливі команди U і поведінки вірусу в команди програми A, і поведінка вірусу R_3 та R_4 утворена без вкорінення шкідливих команд U в команди програми A можна представити послідовностями:

$$R_3^A = e_1...e_\theta a_1...a_n u_1 b_\rho...u_W b_\zeta;$$

$$R_3 = e_1...e_\theta u_1 b_\rho...u_W b_\zeta;$$

$$R_4 = e_1...e_\theta u_\vartheta b_\rho...u_\sigma b_\zeta;$$

$$R_4^A = e_1...e_\theta a_1...a_n u_\vartheta b_\rho...u_\sigma b_\zeta,$$

де значення $\rho,\zeta,\vartheta,\sigma$ визначають, що можливі «команди-сміття» і команди вірусу $u_\vartheta b_\rho ... u_\sigma b_\zeta$ можуть бути різними для кожного нового запуску вірусу.

П'ятий рівень поліморфізму включає в себе властивості всіх перерахованих рівнів, а також розшифровувач може використовувати різні алгоритми розшифровування вірусного коду.

Приймемо модель поліморфного вірусу п'ятого рівня кортежем:

$$M_5 = (A, B, X, G, U, \xi, H, D, R)$$

де A - множина команд певної програми, яка може бути інфікована вірусом, $A = \{a_1,...,a_n\}$; V – множина команд вірусу для вибору одного з присутніх у вірусі розшифровувачів, $V = \{v_1,...,v_m\}$; X - множина розшифровувачів присутніх у вірусі, $X = \{x_1,...,x_y\}$; G – множина команд вірусу x_i розшифровувача, $G = \{g_1,...,g_\theta\}$; U - множина шкідливих команд (тіло вірусу), $U = \{u_1,...,u_w\}$; B - множина «команд-сміття», $B = \{b_1,...,b_h\}$; ξ - функція вибору x_i розшифровувача, $\xi : V \to X$; H – функція утворення шкідливих команд засобами вибору x_i розшифровувача командами $g_{x_i} \in G$ і генерації порядку їх виконання, $H : B \times G_{x_i} \to U$; D - функція утворення поведінки поліморфного вірусу R шляхом вкорінення шкідливих команд U в команди програми A, $D : A \times U \to R$; функція утворення поведінки поліморфного вірусу R без вкорінення у певну програму шляхом розшифрування одним з розшифровувачів x_i командами $g_{x_i} \in G$ і генерації порядку їх виконання матиме вигляд: $D : U \to R$.

Поведінки вірусу п'ятого рівня поліморфізму R_5^A та R_5 можна отримувати послідовності вигляду: $R_5^A = g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} a_1 ... a_n u_\vartheta b_\rho ... u_\sigma b_\zeta$; $R_5^A = g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} u_\vartheta b_\rho ... u_\sigma b_\zeta$, де значення ρ,ζ визначають, що можливі вірусні команди розшифровувача $g_{\phi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}}$ можуть бути різними для різних розшифровувачів x_ε, ϑ - номер обраного розшифровувача, значення $\rho,\zeta,\vartheta,\sigma$ визначають, що можливі «команди-сміття» і команди вірусу $u_\vartheta b_\rho ... u_\sigma b_\zeta$ можуть бути різними для кожного нового запуску вірусу.

До ***шостого*** рівня відносяться нешифровані віруси, що складаються з програмних частин, які "перемішуються" всередині тіла вірусу. Такі віруси називаються пермутуючими (permutating).

Приймемо модель поліморфного вірусу шостого рівня 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 вірусу), $U = \{u_1,...,u_w\}$; 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.

2.2 Сенсор виявлення поліморфного коду

Для виявлення ботнет-мереж, боти яких використовують поліморфний код, пропонується включення в агент MAC нового сенсора S_7. Даний сенсор повинен являти собою віртуальне середовище, що дозволяє здійснювати емуляцію запуску та виконання над потенційно шкідливим ПЗ певних дій. Реакції на вказані дії дозволять зробити висновок про присутність в ньому поліморфного коду.

Виходячи з властивостей поліморфних вірусів на сенсор S_7 як емулятора покладено наступні функції, які він повинен виконувати:

провокативні дії по відношенню до ймовірно інфікованого файлу;

повторні запуски підозрілого файлу для ймовірної модифікації власного коду;

виявлення ШПЗ шляхом аналізу його поведінки та можливої зміни свого тіла, що базується на принципах відомих рівнів поліморфізму.

Під провокативними діями мається на увазі виявлення властивості поліморфних вірусів

створення своєї копії зі зміною власного тіла при його видаленні. Дана властивість часто призводить до того, що оригінал може бути виявлено та видалено, а нова копія бота буде невидимою для антивірусу.

Здійснення повторних запусків підозрілого ПЗ може показати ймовірну «зміну» тіла програми в результаті виконання шифрування. Виявлення такої зміни можливе завдяки побудові «відбитків» К еталонного та модифікованого файлів К' та їх подальшого порівняння. «Відбиток» К формується визначеною двійковою послідовністю $K = \alpha, \beta, \chi, \delta, \varepsilon$, де α - назва файлу; β - розмір файлу; χ - дата останньої зміни; δ - системний атрибут; ε - 128 бітний код MD5.

Сенсор S_7 також виконує роль поведінкового аналізатора, який здійснює аналіз дій з урахуванням моделей поліморфних вірусів різних рівнів. На основі знань про поліморфну природу поведінки вірусів та поведінок ботнет-мереж є можливим їх виявлення шляхом динамічного порівняння відомих поведінок з поведінками нових ботнет-мереж. Виявлення поліморфного коду здійснюється з урахуванням відкидання можливих команд-сміття, перестановок команд, команд вибору шифрувальника, самих команд шифрувальника тощо. Відомі поведінки ботів та поведінки досліджуваних об'єктів представляються послідовностями, які в подальшому порівнюються.

Для порівняння шаблонних поведінок з поведінкою, яку демонструє потенційно небезпечна програма, використано алгоритм приблизного порівняння [6], який розв'язує задачу k-приблизного збігу. Використаний алгоритм вимагає O(kn) [7].

Базуючись на знаннях про можливі поведінки ботів було згенеровано 200 поведінок ботів. Враховуючи знання про рівні поліморфізму на базі наявних поведінок було згенеровано 10000 поведінок. Кожна з них представлена послідовністю, алфавітом якої є визначена множина API-функцій $\Omega = \{\omega_1 ... \omega_f\}$, з яких потенційно найчастіше будуються ШПЗ. Для проведення експерименту було досліджено та побудовано поведінку трьох відомих ботів, які є «невідомими» по відношенню до нашої бази поведінок. Вказані боти використовують три рівні поліморфізму. При дослідженні вибору оптимального параметра k пошук тривав для усіх можливих збігів. Результати експерименту нечіткого збігу представлено в таблиці 1.

Результат досліджень показали, що при прийнятому алфавіті точний збіг (k=0) не знаходить рішень; разом з тим при k=2, k=3 розв'язків достатньо мало. Із збільшенням k кількість розв'язків стрімко збільшується, але при цьому час для пошуку збігів також зростає. Таким чином, експериментально встановлено, що для виявлення схожої підозрілої поведінки довжиною R достатньо прийняти параметр k=4. На практиці сенсор S_7 зупиняє пошук збігів при виявленні першої схожої.

Таблиця 1

Результати пошуку приблизного збігу для різних довжин послідовностей та різних значень параметра k

	Алфавіт Ω	Послідовність R	Параметр k	Кількість знайдених послідовностей
P1	300	38	0	0
	300	38	2	0
	300	38	3	0
	300	38	4	1
	300	38	5	2
P2	300	93	0	0
	300	93	2	0
	300	93	3	1
	300	93	4	2
	300	93	5	7
P3	300	71	0	0
	300	71	2	1
	300	71	3	4
	300	71	4	14
	300	71	5	22

Виходячи з концепції функціонування антивірусної МАС, в кожному агенті відбувається очікування спрацювання евристичного S_3 та поведінкового S_4 сенсорів. У випадку їх спрацювання, а також виявлення ними факту розпаковування певного файлу, виконується завантаження підозрілого програмного об'єкта в сенсор-емулятор S_7 (див. рис.1).

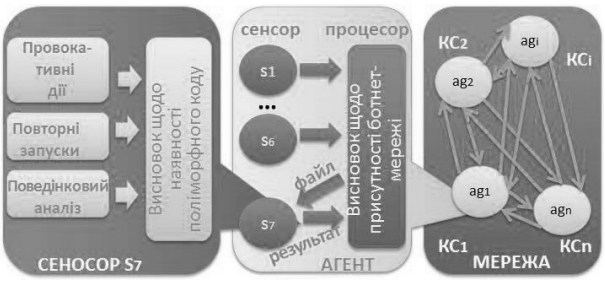

Рис. 1. Робота сенсора S_7 в агенті мульти-агентної системи, що функціонує в локальній мережі

Якщо в результаті провокативних дій виявлено створення нового файла, або якщо у випадку повторних запусків виявлено зміну тіла файла, або в результаті аналізу поведінки програми виявлено поведінку вірусу певного рівня поліморфізму, то сенсор S_7 повідомляє процесор агента про

необхідність блокування даного файлу, а також розсилання інформації про даний файл іншим агентам мульти-агентної системи. Якщо перераховані дії по відношенню до файла не виявляють ознаки присутності віруса даний файл залишає сенсор S_7.

4. Експерименти

З метою перевірки ефективності запропонованого підходу виявлення ботнет-мереж, боти яких містять поліморфний код було проведено ряд експериментів. Вони проводилися на базі розробленої МАС. Основною метою експерименту є визначення ефективності виявлення ботнет-мереж із залученням сенсора S_7 та без нього.

Для експериментальних досліджень було згенеровано 50 програм з властивостями ботнет-мереж без поліморфного коду, а також 50 їх аналогів, що містять поліморфний код (згенеровані програми використовують перші чотири рівні поліморфізму). Кожен експеримент проводився протягом 24 годин. В якості віртуального середовища, в якому працює сенсор S_7, було використано віртуальну машину Oracle VirtualBox; в якості гостьової операційної системи була Widows7.

Результати експериментів подано в таблиці 2.

Таблиця 2

Результати експериментів

Результати МАС	Виявлення		Хибні спрацювання
	%	Кількість	Кількість
без сенсора s_7; програми не містять поліморфний код	**90**	45	5
без сенсора s_7; програми містять поліморфний код	**76**	38	5
з сенсором s_7; програми містять поліморфний код	**92**	46	6

Експериментальні результати свідчать про приріст ефективності виявлення ботнет-мереж, що використовують поліморфний код, засобами МАС, до якої було включено сенсор s7.

Висновки

Запропоновано метод виявлення нових ботнет-мереж, боти яких використовують поліморфний код. Виявлення виконується на базі залучення мульти-агентної системи засобами антивірусних агентів, що містять множину сенсорів.

В роботі розглянуто та досліджено рівні поліморфізму, а також побудовано їх моделі.

Запропоновано включення нового сенсора, на якого покладено функцію виявлення поліморфного коду. Виходячи з властивостей поліморфних вірусів даний сенсор виконує провокативні дії по відношенню до ймовірно інфікованого файлу, повторні запуски підозрілого файлу для ймовірної модифікації власного коду, виявлення ботів шляхом аналізу їх поведінки та можливої зміни свого тіла, що базується на принципах відомих рівнів поліморфізму.

Експериментальні дослідження показали приріст ефективності виявлення ботнет-мереж, що містять поліморфний код на 16% при залученні розробленого сенсора у порівнянні з його відсутністю. При цьому ріст хибних спрацювань не значний.

Недоліком запропонованого методу є достатньо велика обчислювальна складність на етапі аналізу їх поведінки та можливої зміни свого тіла, що базується на принципах відомих рівнів поліморфізму.

Література

1. IMDS: Intelligent Malware Detection System [Text] / Ye Yanfang, Wang Dingding, Li Tao, Ye Dongyi // Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, 12-17 серпня 2007 р. – SanJose, California. – С. 1043-1047.

2. Automatic generation of string signatures for malware detection [Text] / Kent Grin, Scott Schneider, Xin Hu, Tzi-cker Chiueh // Recent Advances in Intrusion Detection, ser. Lecture Notes in Computer Science, E. Kirda, S. Jha, and D. Balzarotti, Eds. – Berlin, 2009. – С. 101-120.

3. Virus detection using data mining techniques [Text] / J. H. Wang, P. S. Deng, Y. S. Fan, L. J. Jaw, Y. C. Liu // Proceedings of the IEEE 37th Annual 2003 International Carnahan Conference on Security Technology. – 2003.

4. Multi-agent Based Approach for Botnet Detection in a Corporate Area Network Using Fuzzy Logic [Text] / O. Pomorova, O. Savenko, S. Lysenko, A. Kryshchuk // Kwiecien A., Gaj, P., Stera, P. (eds.) CN2013. – Springer, Heidelberg Dordrecht London New York. – 2013. – С. 146-156.

5. Касперский, Е. В. Компьютерные вирусы: что это такое и как с ними бороться [Текст] / Е. В. Касперский. – М. : СК Пресс, 1998. – 288 с.

6. Jokinen, P. A comparison of approximate string matching algorithms [Text] / P. Jokinen, J. Tarhio, E. Ukkonen // Practice and Experience – 1996. – Volume 26, Issue 12. – С. 1439-1458.

7. Smyth, B. Computing Patterns In Strings [Text] / B. Smyth // Williams, Moscow. – 2006. – С. 496.

Поступила в редакцию 12.02.2014, рассмотрена на редколлегии 24.03.2014

Рецензент: д-р техн. наук, проф. А. В. Горбенко, Национальный аэрокосмический университет им. Н. Е. Жуковского «ХАИ», Харьков, Украина.

TECHNIQUE FOR POLYMORPHIC CODE OF THE BOTNET'S BOTS

S. Lysenko, O. Savenko, A. Nicheporuk

Abstract: The new technique of botnet detection which bots use polymorphic code was proposed. Performed detection is based on the multi-agent system by means of antiviral agents that contain sensors. For detection of botnet, which bots use polymorphic code, the levels of polymorphism were researched and its models were built. The new sensor for polymorphic code detection within antivirus agent of multi-agent system was developed. Developed sensor performs provocative actions against probably infected file, restarts of the suspicious file for probably modified code detection, behavior analysis for modified code detection, based on the principles of known levels of polymorphism.

Key words: botnet, polymorphic code, levels of the polymorphism, multi-agent system, agent, sensor.

МЕТОД ВЫЯВЛЕНИЯ ПОЛИМОРФНОГО КОДА БОТОВ БОТНЕТ-СЕТЕЙ

С. Лысенко, О. Савенко, А. Ничепорук

Предложен метод выявления новых ботнет-сетей, боты которых используют полиморфный код. Выявление осуществляется на базе привлечения мульти-агентной системы посредством антивирусных агентов, содержащих сенсоры. В работе исследованы уровни полиморфизма и построены их модели. Разработан сенсор обнаружения полиморфного кода. Исходя из свойств полиморфных вирусов, сенсор выполняет провокационные действия в отношении вероятно инфицированного файла, повторные запуски подозрительного файла для вероятной модификации собственного кода, выявления ботов путем анализа их поведения и возможного изменения своего тела, основанный на принципах известных уровней полиморфизма.

Ключевые слова: бот, ботнет-сеть, полиморфный код, уровни полиморфизма, мульти-агентна система, агент, сенсор

Лисенко Сергій Миколайович – канд. техн. наук, доц., Хмельницький національний університет, Хмельницький, Україна, e-mail: sirogyk@ukr.net.

Савенко Олег Станіславович – канд. техн. наук, доц., Хмельницький національний університет, Хмельницький, Україна, e-mail: savenko_oleg_st@ukr.net.

Нічепорук Андрій Олександрович – аспірант, Хмельницький національний університет, Україна, e-mail: paunni@mail.ru.