УДК 621.391

О. С. Савенко[1]
С. М. Лисенко[1]
А. О. Нічепорук[1]

МОДЕЛІ РІВНІВ ПОЛІМОРФНИХ КОМП'ЮТЕРНИХ ВІРУСІВ

[1]Хмельницький національний університет

Розглянуто та доповнено класифікацію поліморфних вірусів. На основі цієї класифікації описані моделі функціонування поліморфних вірусів.

Ключові слова: поліморфізм, *розшифровувач*, інфікована програма, шкідлива програма.

Вступ та постановка задачі

З інтенсивним розвитком та поширенням комп'ютерної техніки гостро постає проблема протидії комп'ютерним вірусом.

Однією з технологій, що утруднює виявлення антивірусними діагностичними системами є поліморфізм комп'ютерних вірусів.

Поліморфні віруси — це віруси в яких розшифровувач самомодифікується. Мета такого шифрування, полягає у неможливості аналізу коду за допомогою дизасемблювання. Цей код зашифрований і являє собою випадковий набір команд. Розшифровка відбувається самим вірусом вже безпосередньо під час виконання. При цьому можливі варіанти: він може розшифрувати себе всього відразу, а може виконати таку розшифровку у процесі свого виконання. Все це робиться з метою утруднення аналізу коду вірусу [1]. Дані перетворення досягаються за допомогою процесу обфускації. *Обфускація* (від лат. Obfuscare — «затінювати, затемнювати») — техніка, спрямована на заплутування коду програми, тобто приведення вихідного тексту або виконуваного коду до працюючого вигляду, але утруднює аналіз такого коду.

Існуючі методи виявлення поліморфного коду, засновані на методі емуляції, зокрема метод побудови абстрактного керуючого графу інфікованого файлу за допомогою зовнішніх предикатів або метод створення відбитків (pattern), мають недоліки пов'язані з тим, що аналізований код не обов'язково може бути кодом, який насправді зараз виконується [2—5]. Тому актуальною є задача подальшого дослідження поліморфних вірусів і побудова моделей їх функціонування, результати яких могли б бути використанні у побудові метода виявлення поліморфних вірусів.

Тому *метою дослідження* є проведення дослідження функціонування ПКВ, з розробленням їх уточненої класифікації.

1. Рівні поліморфізму і їх моделі

Згідно з класифікацією Е. Касперского розрізняють 6 рівнів поліморфізму [6]. Така класифікація передбачає розподіл вірусних програм в залежності від складності коду, з якого складається поліморфний розшифровувач.

***Визначення* 1**. *Інфікована програма* — деяка програма загального чи спеціального призначення, що виконує свої функції, які були закладені в неї при її реалізації, і у яку було вкорінено шкідливий програмний код. В результаті вкорінення шкідливого програмного коду в тіло інфікованої програми вона буде виконувати свої функції, а також функції, що закладені в основу шкідливого програмного коду, зокрема виконання деструктивних дій та пошук файлів для подальшого інфікування. Для вкорінення шкідливого коду можуть використовуватися стратегії розширення секції в структурі формату PE COFF, створення нової секції на початку, в середині та в кінці файлу.

***Визначення* 2.** *Шкідлива програма* — резидентний або не резидентний програмний код, що виконує деструктивні дії. Шкідлива програма може залишатися непоміченою довгий час і виконує деструктивні дії за виконання певної умови, наприклад активація деякого процесу чи служби.

Тому кожен рівень поліморфізму можна розділити на два підрівні — з впровадженням у про-

граму та без впровадження у деяку програму.

1.1. Перший рівень поліморфізму

До цього рівня належать поліморфні віруси, що проводять розшифровку основного тіла вірусу, вибираючи один з декількох постійних розшифровувачів, в яких міститься постійний код.

Для реалізації таких поліморфних вірусів створюється деяка кількість шифрувальників/розшифровувачів, вибирається випадковий шифрувальник і зашифровує тіло вірусу разом з рештою шифрувальників/розшифровувачів. В кожному наступному поколінні розшифровувач розшифровує вірус, передає управління основному тілу, а перед впровадженням вибирає випадковий шифрувальник, яким і зашифровує його. Код вірусу повністю змінюється і «візуально» розпізнати заражений файл вже не представляється можливим. Розглянемо модель першого рівня поліморфізму.

Приймемо модель поліморфного вірусу першого рівня кортежем

$$M_1 = \left(A,\ X,\ G,\ V,\ U,\ \xi,\ Q,\ P,\ R\right), \tag{1}$$

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, $V = \{\nu_1,...,\nu_m\}$; X — множина розшифровувачів присутніх у вірусі, $X = \{x_1,...,x_m\}$; 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, $\xi : \nu_1,..\nu_l \to x_i$; 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: $Q : U \to R$. Модель ПКВ першого рівня з впровадженням у програму зображена на рис. 1. 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:

$R_1^A = g_{\varphi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} a_1 ... a_n u_1 ... u_w$; $R_1 = g_{\varphi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}} u_1 ... u_w$., де значення φ, η визначають, що можливі вірусні команди розшифровувача $g_{\varphi_{x_\varepsilon}} ... g_{\eta_{x_\varepsilon}}$ можуть бути різними для різних розшифровувачів x_ε, ε — номер обраного розшифровувача.

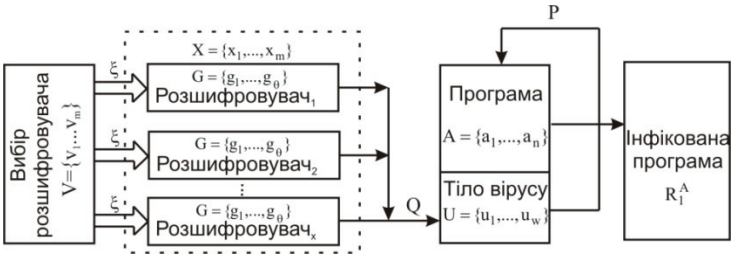

Рис. 1. Модель поліморфного вірусу першого рівня з впровадженням у програму

1.2. Другий рівень поліморфізму

Розшифровувач у вірусах другого рівня поліморфізму має постійною одну чи декілька команд, основна ж його частина непостійна. До такого типу поліморфних вірусів належать віруси, розшифровувачі яких використовують альтернативні команди, різні регістри. Розшифровувач вірусу має постійний алгоритм, але складається з довільно обраної послідовності команд.

Розглянемо модель другого рівня поліморфізму.

Приймемо модель поліморфного вірусу другого рівня кортежем:

$$M_2 = (A, E, U, P, Z, R), \tag{2}$$

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: $Z : E \times U \to R$. Модель ПКВ другого рівня з впровадженням у програму зображена на рис.2.

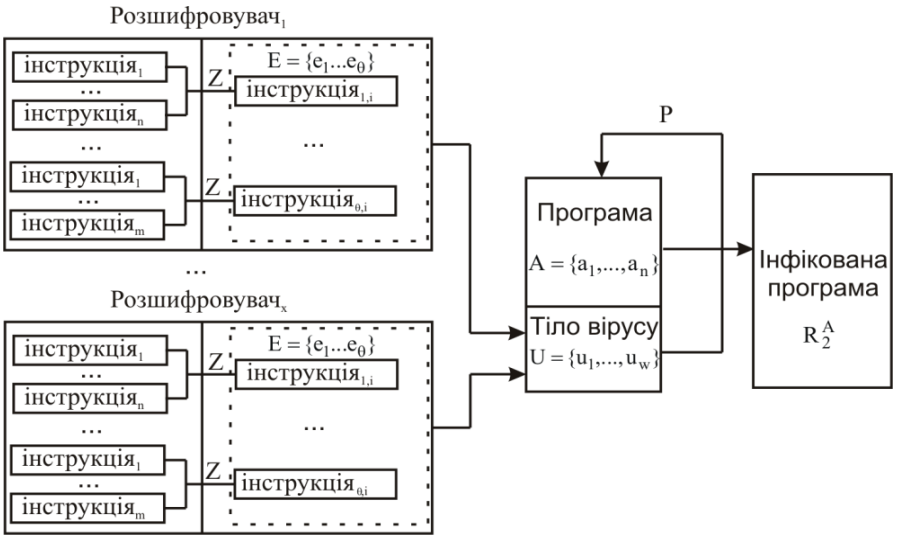

Рис. 2. Модель поліморфного вірусу другого рівня з впровадженням у програму

Поведінки вірусу другого рівня поліморфізма 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 за кожного запуску вірусу.

1.3. Третій та четвертий рівні поліморфізму

Поліморфні віруси третього та четвертого рівнів містять у своєму коді невикористовуванні команди (команди-сміття). Команди-сміття, що включаються в розшифровувач, не виконують ніяких «реальних» дій і не роблять ніякого впливу на виконання розшифровувача. Їх єдиним завданням є приховати призначення розшифровувача. Без використання команд-сміття розшифровувач, а відповідно і вірус можна знайти за допомогою евристичних алгоритмів. Окрім того, команди-сміття дозволять «справжнім» інструкціям перебувати у випадкових місцях (але в постійному порядку). Наприклад, в одному поколінні операції inc і dec знаходяться безпосередньо поруч один з одним, в той час як в іншому поколінні між ними може виявитись одна або декілька нічого незначущих інструкцій. Приймемо модель поліморфного вірусу третього та четвертого рівнів кортежем:

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

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: $Y : E \times B \to R$. Моделі ПКВ третього та четвертого рівнів з впровадженням у програму зображені на рис. 3. та рис. 4.

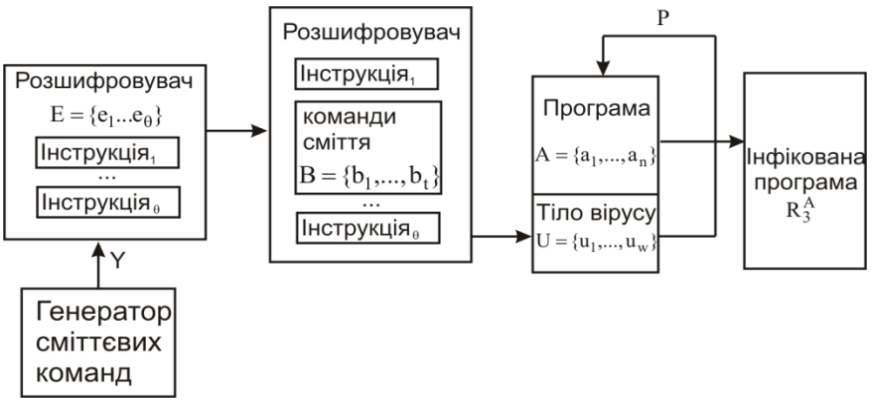

Рис. 3. Модель поліморфного вірусу третього рівня з впровадженням у програму

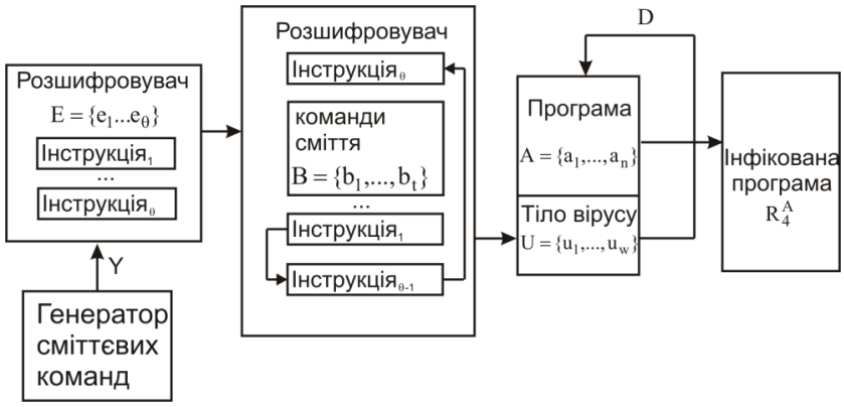

Рис. 4. Модель поліморфного вірусу четвертого рівня з впровадженням у програму

Поведінка вірусу третього та четвертого рівнів поліморфізму R_3^A та R_4^A, утворених 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:

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

$R_3 = e_1 ... e_\theta u_1 b_\rho ... u_w b_\zeta$; $R_4^A = e_1 ... e_\theta a_1 ... a_n u_\vartheta b_\rho ... u_\sigma b_\zeta$; $R_4 = e_1 ... e_\theta u_\vartheta b_\rho ... u_\sigma b_\zeta$, де значення ρ, ζ, ϑ, σ визначають, що можливі «команди-сміття» і команди вірусу $u_\vartheta b_\rho ... u_\sigma b_\zeta$ можуть бути різними для кожного нового запуску вірусу.

1.4. П’ятий рівень поліморфізму

Включає в себе використання всіх перерахованих вище рівнів. Розшифровувач може використовувати різні алгоритми розшифрування вірусного коду. Також можливе використання, для розшифрування основного вірусного коду, розшифровки частини самого ж розшифровувача або декількох розшифровувачів, що почергово розшифровують один одного, або, безпосередньо, вірусний код. Як правило, детектування вірусів цього рівня поліморфізму за допомогою сигнатури неможливо. Приймемо модель поліморфного вірусу п’ятого рівня кортежем

$$M_5 = (A, B, X, G, U, \xi, H, D, R), \tag{4}$$

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, $V = \{\nu_1, ..., \nu_m\}$; X — множина розшифровувачів, присутніх у вірусі, $X = \{x_1, ..., x_m\}$; 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, $\xi : \nu_1,..\nu_l \to x_i$; 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: $D : U \to R$. Модель ПКВ п'ятого рівня з впровадженням у програму зображена на рис. 5.

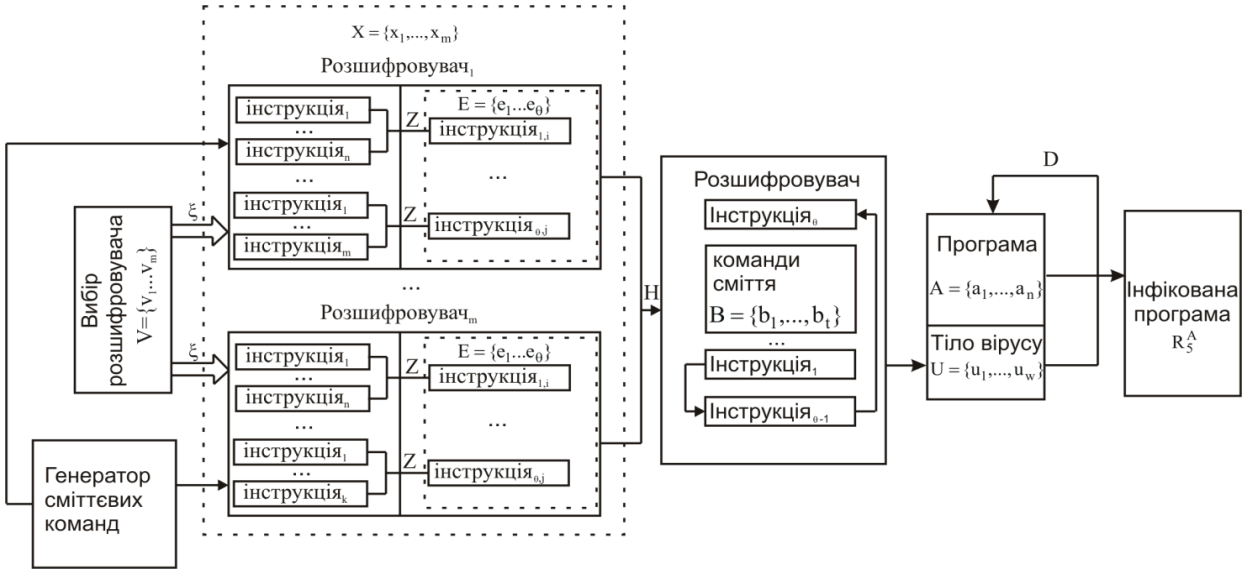

Рис. 5. Модель поліморфного вірусу п'ятого рівня з впровадженням у програму

Поведінки вірусу п'ятого рівня поліморфізма R_5^A та R_5 можна отримувати послідовності вигляду:

$R_5^A = g_{\varphi_{x_\varepsilon}}...g_{\eta_{x_\varepsilon}} a_1...a_n u_\vartheta b_\rho...u_\sigma b_\zeta$; $R_5^A = g_{\varphi_{x_\varepsilon}}...g_{\eta_{x_\varepsilon}} u_\vartheta b_\rho...u_\sigma b_\zeta$, де значення ρ, ζ визначають, що можливі вірусні команди розшифровувача $g_{\varphi_{x_\varepsilon}}...g_{\eta_{x_\varepsilon}}$ 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.

1.5. Поліморфні віруси шостого рівня

Поліморфні віруси шостого рівня складаються з програмних одиниць-частин. Вони постійно змінюються в тілі і переміщують свої підпрограми (інсталяції, зараження, обробника переривання, аналізу файлу і т. д.). Характерною особливістю таких вірусів є наявність плям. При цьому в різні місця файлу записується кілька блоків коду, що зумовлює назву методу. Такі плями в цілому утворюють поліморфний розшифровувача, який працює з кодом в кінці файлу. Для реалізації методу навіть не потрібно використовувати команди-сміття — підібрати сигнатуру буде все одно неможливо. Подібні віруси можуть бути незашифровані. Приймемо модель поліморфного вірусу шостого рівня кортежем

$$M_6 = (A, E, U, C, R), \tag{5}$$

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: $C : E \times U \to R$. Модель ПКВ шостого рівня з впрова-

дженням у програму зображена на рис. 6.

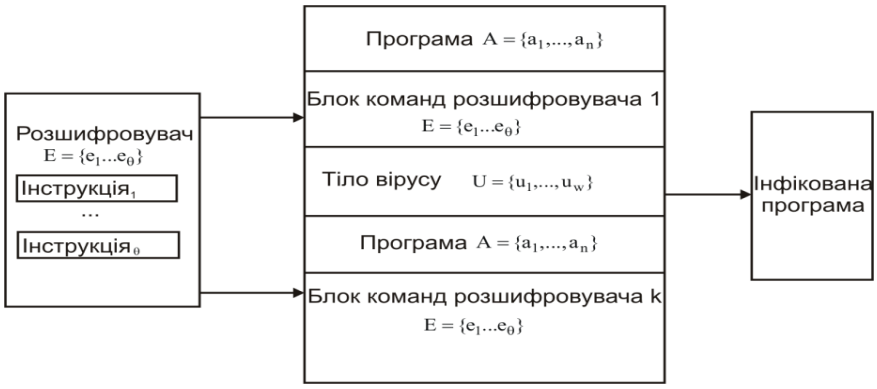

Рис. 6. Модель вірусу п'ятого рівня з впровадженням у програму

Поведінки вірусу шостого рівня поліморфізму R_6^A та R_6 можуть бути представлені послідовностями

$$R_6^A = a_1 e_\varphi ... a_i e_\eta a_{i+1} u_\vartheta ... a_n u_\sigma ; \quad R_6^A = a_1 e_\varphi u_\vartheta a_2 ... a_n e_\eta u_\sigma ; \quad R_6 = e_\varphi ... e_\eta u_\vartheta ... u_\sigma ; \quad R_6 = e_\varphi u_\vartheta ... e_\eta u_\sigma ,$$

де значення $\varphi, \eta, \vartheta, \sigma$ визначають, що можливі команди розшифровувача та шкідливі команди $e_\varphi ... e_\eta u_\vartheta ... u_\sigma$ можуть бути різними для кожного нового запуску вірусу.

2. Застосування отриманих моделей

Для оцінки адекватності розроблених моделей розглянемо реальні програми з шкідливим поліморфним кодом. З цією метою створимо набір поведінок, що описують типові дії поліморфних вірусів «команд-сміття» (табл. 1), розшифровувача (табл. 2) та тіла вірусу (табл. 3).

Таблиця 1

Множина «команд-сміття»

№	B	«Команди-сміття»
1	b1	Однобайтні інструкції NOP, CLI, STI, CLC, LAHF, SAHF, CLD, CMC
2	b2	Поява команд CMP та TEST без наступного використання команд умовного переходу Jxx
3	b3	Поява після команди TEST команд, що не є Jnz або Jz
4	b4	Використання декількох команд співпроцесора fwait, fninit та ін. у відносно невеликому фрагменті коду
5	b5	Поява в коді програми команд XCHG RG, RG
6	b6	Поява «вироджених переходів», наприклад JMP 0000
7	b7	Використання нічого не значущих переривань (наприклад відображення дати, часу)
8	b8	Використання функцій переривань INT 21h, INT 10h, INT 16h з функціями, що не несуть корисного навантаження, зокрема в регістрі AH значення 0Bh, 0Dh та ін.
9	b9	Поява команди типу MOV RG1, RG1

Таблиця 2

Опис дій розшифровувачів

№	E		Опис дій розшифровувача
1	E1	e1	запис в регістр ECX розміру вірусу
		e2	запис в регістр EDI адреси початку вірусу
		e3	запис в регістр EAX ключа розшифрування
		e4	циклічне виконання додавання по модулю два подвійного слова, що розміщується в EDI з ключем розшифрування, що знаходиться в EAX
		e5	перехід до наступного подвійного слова для розшифрування

Продовження табл. 2

№	E		Опис дій розшифровувача
2	E2	e1	визначення початку тіла вірусу
		e2	запис у регістр BP тіла вірусу
		e3	запис в регістр BX наступного слово тіла вірусу
		e4	розшифрування тіла вірусу шляхом додавання константи до BX
		e5	перехід до наступного слова
		e6	запуск циклу до тих пір поки не розшифрується все тіло вірусу
3	E3	e1	запис першого ключа розшифрування у регістр AX
		e2	запис першого ключа розшифрування у регістр CX
		e3	розміщення початку тіла вірусу в регістр DI
		e4	розшифрування першого байту тіла вірусу за допомогою операції XOR з другим ключем (регістр CX)
		e5	розшифрування першого байту тіла вірусу за допомогою операції XOR з ключем номер 1 (регістр AX)
		e6	циклічний перехід до наступного байту тіла вірусу
4	E4	e1	встановлення значення кінця вірусного тіла в регістр EAX
		e2	розшифрування тіла вірусу за допомогою логічної операції NOT
		e3	вибір наступного подвійного слова для розшифрування
		e4	запуск циклу до тих пір поки не розшифрується все тіло вірусу
5	E5	e1	запис кількість байтів тіла вірусу в регістр DX
		e2	розміщення в регістрі AL ключа розшифрування
		e3	вибірка байту для розшифрування
		e4	розшифрування байту тіла вірусу за допомогою операції віднімання
		e5	зменшення вказівника на кількість байтів
		e6	циклічне розшифрування тіла вірусу, поки регістр DX не стане рівним нулю

Таблиця 3

Опис дій тіла вірусу

№	U	u	Опис основних дій вірусу	Назва вірусу
1	U1	u1	генерація виключення для переходу на кільце Ring0	Chernobyl
		u2	ініціалізація на кільці Ring0	
		u3	перевірка на наявність копії вірусу у пам'яті	
		u4	встановлення власного обробника API файлової системи	
		u5	отримання PE сигнатуру із заголовку файлу	
		u6	процедура зараження файлу і запис вірусу в кінець секції коду	
		u7	процедура виконання деструктивних дій (стирання BIOSEEPROM та секторів НЖМД)	
2	U2	u1	перехоплення API функції	Win32.Virut
		u2	впровадження вірусу в процес explorer.exe, шляхом розширення останньої секції	
		u3	завершення процесів антивірусних програм	
		u4	зміна PE заголовка, зміна полів AddressOfEntryPoin тструктури IMAGE_NT_HEADERS32	
		u5	виконання процедури розсилки спаму та крадіжки персональних даних	
3	U3	u1	створення ключа реєстру [HKCR\CLSID\{C111980D-B372-44b4-8095-1B6060E8C647}], який містить посилання на виконуваний файл вірусу	Win32.Alman
		u2	зміна точки входу програми на точку входу вірусу	
		u3	розбір структури PE файлу, що підлягає зараженню	
		u4	створення нової секції коду та редагування заголовку PE файлу	
		u5	виконання блокування системних служб, завантаження інших шкідливих програм та крадіжка персональних даних	
		u6	пошук нових файлів для зараження	
4	U4	u1	пошук всіх запущених процесів і впровадження у їх адресний простір	Win32.Polipos
		u2	створення нової секції для запису тіла вірусу	
		u3	перехоплення API функцій ExitProcess, CreateProcess, CreateFileAтаін	
		u4	зараження нових файлів	
		u5	видалення файлів бази даних антивірусних програм	
		u6	виконання функцій бот-мереж (учасник мережі Gnutella)	

Для визначення рівня поліморфізму розглянемо фрагмент поліморфного коду.

```
;розшифровувач
Decrypt:
movbx,[bp+0D2B] ;
add    bx,9D64
xchg   [bp+0D2B],bx
mov    bx,8F31
sub    bx,bp
mov    bp,8F33
sub    bp,bx
jnz    Decrypt
;Тіло вірусу
cmpbp,di
LoopOfWriteCodeToSections:
add    edx, SizeOfScetionTable
movebx, (SizeOfRawData-@9)[edx]
sub    ebx, (VirtualSize-@9)[edx]
jbeEndOfWriteCodeToSections
rolsi,1
push   ebx
sub    eax, 08h
mov    [eax], ebx
movebx, (PointerToRawData-@9)[edx]
add    ebx, (VirtualSize-@9)[edx]
push   ebx
movecx,ecx
nop
cmpesi,eax
push   edi
movebx, (VirtualSize-@9)[edx]
add    ebx, (VirtualAddress-@9)[edx]
test edi,ebp
jb 0560
add    ebx, (ImageBase-@9)[esi]
mov    [eax+4], ebx
movebx, [eax]
add    (VirtualSize-@9)[edx], ebx
or  (Characteristics-@9)[edx], 40000040h
StartToWriteCodeToSections:
sub    ebp, ebx
int 01h
jbeSetVirusCodeSectionTableEndMark
add    edi, ebx
EndOfWriteCodeToSections:
 loop   LoopOfWriteCodeToSections
 nop
```

Ґрунтуючись на даних з табл. 1—3 та моделей (1—6), побудуємо поведінку поліморфного вірусу

$$R = e_1 - e_6\, b_2\, u_5\, b_9\, b_1\, b_2\, u_5\, b_3\, u_5\, u_6\, b_7\, u_6\, b_1.$$

Така поведінка відповідає третьому рівню поліморфізму без впровадження вірусу у програму $R_3 = e_1 ... e_0 u_1 b_\rho ... u_w b_\zeta$, отже $R \approx R_3 = e_1 ... e_0 u_1 b_\rho ... u_w b_\zeta$.

Висновок

Проаналізовано та вдосконалено класифікацію поліморфних вірусів. На основі цієї класифікації розроблені моделі функціонування поліморфних вірусів, що описують їх поведінку та їх функційні особливості. Описані моделі функціонування поліморфних вірусів можуть бути основою для побудови методу діагностування комп'ютерних систем на наявність поліморфного шкідливого коду.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Szor P. Hunting for Metamorphic / P. Szor, P. Ferrie// Virus Bulletin. — Sept, 2001. — P. 123—144.

2. Christodorescu M. Static Analysis of Executables to Detect Malicious Patterns / M. Christodorescu, S. Jha // 12th conferenceon USENIX Security Symposium, Aug., 2003, Berkeley, USA, 2003. — P. 169—186.

3. Christodorescu M. Semantics-Aware Malware Detection / M. Christodorescu, S. Jha, S. A. Seshia, D. Song, Bryant, R. E. // Proceedings of the 2005 IEEE Symposium on Security and Privacy, 2005. Washington, DC, USA. — 2005. — P. 32—46.

4. Kruegel C. Detecting Kernel-Level Rootkits Through Binary Analysis / C. Kruegel, W. Robertson, G. Vigna // Proceedings of the 20th Annual Computer Security Applications Conference, 2004, Washington, DC, USA. — 2004. — P. 91—100.

5. Bayer U. TTAnalyze : A Tool for Analyzing Malware / U. Bayer, C. Kruegel, E. Kirda // 15th European Institute for Computer Antivirus Research (EICAR 2006) Annual Conference, Wien. — 2006.

6. Касперский Е. В. Компьютерные вирусы: что это такое и как с ними бороться / Е. В. Касперский. — М. : СК Пресс, 1998. — 288 с.

Рекомендована кафедрою захисту інформації ВНТУ

Стаття надійшла до редакції 15.10.2014

Савенко Олег Станіславович — канд. техн. наук, доцент, доцент кафедри системного програмування;
Лисенко Сергій Миколайович — канд. техн. наук, доцент, доцент кафедри системного програмування;
Нічепорук Андрій Олександрович — аспірант кафедри системного програмування, e-mail: paunni@mail.ru.
Хмельницький національний університет, Хмельницький

O. S. Savenko[1]
S. M. Lysenko[1]
A. O. Nicheporuk[1]

Models of levels of polymorphic viruses

[1]Khmelnytskyi National University

Classification of polymorphic viruses is considered and complemented in the paper. On the basis of this classification the models of functioning of polymorphic viruses are described.

Keywords: polymorphism, decryption routine, infected program, malware.

Savenko Oleg S. — Cand. Sc. (Eng.), Assistant Professor of the Chair of System Programming;
Lysenko Sergii N. — Cand. Sc. (Eng.), Assistant Professor of the Chair of System Programming;
Nicheporuk Andrii O. — Post-Graduate Student of the Chair of System Programming, e-mail: paunni@mail.ru

О. С. Савенко[1]
С. Н. Лысенко[1]
А. А. Ничепорук[1]

Модели уровней полиморфных компьютерных вирусов

[1]Хмельницкий национальный университет

Рассмотрена и дополнена классификация полиморфных вирусов. На основе этой классификации описаны модели функционирования полиморфных вирусов.

Ключевые слова: полиморфизм, расшифровщик, инфицированная программа, вредоносная программа.

Савенко Олег Станиславович — канд. техн. наук, доцент, доцент кафедры системного программирования;
Лысенко Сергей Николаевич — канд. техн. наук, доцент, доцент кафедры системного программирования;
Ничепорук Андрей Александрович — аспирант кафедры системного программирования, e-mail: paunni@mail.ru