УДК 004.492.3

О.С. САВЕНКО, С.М. ЛИСЕНКО

Хмельницький національний університет, Україна

АЛГОРИТМИ ПОШУКУ ТРОЯНСЬКИХ ПРОГРАМ В ПЕРСОНАЛЬНИХ КОМП'ЮТЕРАХ

Досліджено ситуацію щодо стану розробки шкідливого програмного забезпечення. Подано структуру системи пошуку троянських програм в персональних комп'ютерах, що ґрунтується на інтелектуальному методі їх пошуку. Розроблено алгоритми пошуку троянських програм, які в подальшому дадуть змогу реалізувати систему пошуку троянських програм в персональних комп'ютерах. В результаті проведеного дослідження сучасних методів пошуку троянських програм виявлено їх недоліки: низька ймовірність виявлення нових троянських програм, високі вимоги окремих методів пошуку до апаратного забезпечення.

Ключові слова: *троянська програма, антивірусне програмне забезпечення, алгоритм пошуку троянських програм, нечіткий логічний висновок, алгоритм негативного відбору*

Вступ

Динамічне зростання кількості троянських програм (ТП), які виконують на персональному комп'ютері (ПК) деструктивні або шкідливі дії, змушує розробників антивірусного програмного забезпечення (ПЗ) впроваджувати нові методи та алгоритми їх пошуку та знешкодження. Наявні факти пошкодження та викрадення інформації в ПК, в якому встановлене антивірусне ПЗ, свідчать про недоліки відомих методів виявлення ТП в ПК. Сучасні методи та алгоритми, на яких ґрунтуються ці методи пошуку ТП в ПК, орієнтовані на виявлення відомих ТП, та не повністю адаптовані до розпізнавання нових підозрілих об'єктів.

Сучасні методи пошуку та виявлення троянських програм, які ґрунтуються на алгоритмах сигнатурного аналізу, контрольних сум та евристичного аналізу, мають певні недоліки [1]. Дослідження методів показало, що алгоритми, на яких вони базуються, не вирішують задачі ефективного пошуку нових троянських програм, оскільки не адаптовані до розпізнавання за їх життєвим циклом. Алгоритми, на яких базуються сучасні методи демонструють низьку ймовірність розпізнавання та виявлення нових ТП.

1. Постановка задачі

Сьогодні важливою є задача розробки нових алгоритмів пошуку троянських програм, які б могли стати основою нового ефективного методу пошуку та виявлення невідомих троянських програм в персональних комп'ютерах. Необхідною вимогою до алгоритмів є їх здатність до виявлення як відомих, так нових ТП. Нові алгоритми повинні здійснювати висновок про можливість присутності ТП в ПК та забезпечити високу ймовірність виявлення факту підміни файлів троянськими версіями.

2. Система пошуку троянських програм в персональних комп'ютерах

Для усунення недоліків і вирішення наявних проблем відомих методів розроблено інтелектуальний метод пошуку троянських програм, який базується на моделі [2] та дозволяє здійснити висновок щодо можливої присутності троянської програми в персональному комп'ютері як відомої, так і нової. Метод також передбачає проведення сканування ПК на предмет виявлення факту підміни ПЗ троянськими версіями в системі, що діагностується. Інтелектуальність методу визначають компоненти, що використовуються в процесі здійснення пошуку троянських програм. Під поняттям пошуку будемо розуміти процеси виявлення, ідентифікації та сканування файлів на предмет виявлення їх підміни. Виявлення ТП здійснюємо за її життєвим циклом (ЖЦ), який включає етапи: потрапляння на ПК, активізації та виконання закладених функцій. Система виявлення використовує базу знань – базу поведінкових моделей ТП на її різних етапах ЖЦ [3]. Ідентифікацію ТП здійснюємо за допомогою використання нечіткого логічного висновку (НЛВ) в межах підсистеми аналізу та висновку [4]. Сканування ПК на предмет підміни файлів троянськими версіями здійснюємо з використанням алгоритму негативного відбору, який застосовується в штучних імунних системах (ШІС).

Система пошуку ТП в ПК (рис. 1) включає підсистему виявлення та ідентифікації (монітор) та підсистему аналізу аномалій (сканер).

Монітор включає в себе підсистему моніторингу та підсистему аналізу та висновку.

Підсистема моніторингу відслідковує події в системі, реагує на програми, дії яких відповідають життєвому циклу ТП.

Підсистема аналізу та висновку за наявними алгоритмами робить результуючий висновок щодо можливості присутності ТП в ПК.

Сканер включає в себе генератор детекторів та підсистему контролю даних. Метою сканування ПК, що діагностується, є виявлення аномалії. Під аномалією будемо розуміти виявлення факту підміни системних фалів троянськими версіями [5].

Рис. 1. Система пошуку троянських програм в персональних комп'ютерах

3. Алгоритми пошуку троянських програм в персональних комп'ютерах

З метою організації програмної реалізації інтелектуального методу пошуку ТП в персональних комп'ютерах розроблено алгоритми, котрі представлені нижче, що дозволяють реалізувати розроблений метод пошуку ТП в ПК.

Для представлення алгоритму моніторингу системних подій в персональному комп'ютері та виявлення підозрілих програмних об'єктів, інтерпретуватимемо будь-яку програму з множини програмних об'єктів персонального комп'ютера як $e \in E$,

де $e = \{m_1, ..m_i, a_{i+}, ..a_n\}$ - скінченний набір можливих програних інструкцій (команд, дій).

Представимо наявність інструкції в програмі, що діагностується, як $m \in e$ $(a \in e)$, вважатимемо потрапляння об'єкта e на ПК через системний порт p як $p \in e$, та вплив програмних інструкцій об'єкта e на структурні одиниці операційної системи як $b \in e$.

Аналогічно приймемо троянську програму $v \in \Psi$ як скінченний набір інструкцій $v = \{m_1, ..m_i, a_{i+}, ..a_n\}$.

Для представлення алгоритму сканування ПК введемо позначення $r_g \leftrightarrow s_i$, що означатиме збіг двох послідовностей, а саме r_g входить послідовність s_i.

3.1. Алгоритм 1. Формування бази поведінкових моделей троянських програм в ПК

1.1. Створити матрицю відношення V_{MP}, де m_i - визначають дії троянської програми v, що дозволяють здійснити її потрапляння на ПК, p_j – системні порти, через які здійснюється потрапляння.

1.2. Створити матрицю відношення L_{AB}, де a_i – дії, що виконує ТП, b_j – структурні одиниці операційної системи ПК, які зазнають впливу від a_i дій троянської програми v.

1.3. Побудувати множини поведінок троянських програм $M_W = \langle T, Q, K \rangle$ шляхом формування множин поведінок троянських програм на етапах їх потрапляння на ПК, їх активізації та виконання закладених дій відповідно:

$$T = \left\langle V_{MP}^t, L_{AB}^t \right\rangle,$$

$$Q = \left\langle V_{MP}^q, L_{AB}^q \right\rangle,$$

$$K = \left\langle V_{MP}^k, L_{AB}^k \right\rangle.$$

1.4. $\forall v \in \Psi$ поки не кінець v виконувати: якщо $m_i^t \in v \cup p_j^t \in v$, то $V_{MP}^t(m_i) = \{p_j\}$, $V_{m_i p_j}^t = 1$, інакше $V_{m_i p_j}^t = 0$, якщо

$m_i^q \in v \cup p_j^q \in v$, то $V_{MP}^q(m_i) = \{p_j\}$, $V_{m_i p_j}^q = 1$, інакше $V_{m_i p_j}^q = 0$, якщо $m_i^k \in v \cup p_j^k \in v$, то $V_{MP}^k(m_i) = \{p_j\}$, $V_{m_i p_j}^k = 1$, інакше $V_{m_i p_j}^k = 0$.

1.5. Якщо $a_i^t \in v \cup b_j^t \in v$, то $L_{AB}^t(a_i) = \{b_j\}$, $L_{a_i b_j}^t = 1$, інакше $L_{a_i b_j}^t = 0$, якщо $a_i^q \in v \cup b_j^q \in v$, то $L_{AB}^q(a_i) = \{b_j\}$, $L_{a_i b_j}^q = 1$, інакше $L_{a_i b_j}^q = 0$, якщо $a_i^k \in v \cup b_j^k \in v$, то $L_{AB}^t(a_i) = \{b_j\}$, $L_{a_i b_j}^k = 1$, інакше $L_{a_i b_j}^k = 0$.

1.6. Кінець алгоритму.

3.2. Алгоритм 2. Моніторинг ПК та виконання нечіткого логічного висновку щодо підозрілості об'єкта, що діагностується.

2.1. $\forall e \in E$ виконувати:

2.2. Прийняти $n_t = 0$, $n_q = 0$, $n_k = 0$, $s_t = 0$, $s_q = 0$, $s_k = 0$, де n та s - лічильники інструкцій, що виконуються об'єктом діагностування в програмному середовищі операційної системи.

2.3. Поки не кінець роботи ПК виконувати:

2.4. $\forall e \in E$, що функціонують в ПК:

2.5. $\forall m = \overline{1,i} \in V_{MP}$ виконувати та $\forall p = \overline{1,j} \in V_{MP}$ виконувати: якщо $V_{m_i p_j} = V_{m_i p_j}^t$, то $n_t = n_t + 1$, якщо $V_{m_i p_j} = V_{m_i p_j}^q$, то $n_q = n_q + 1$, якщо $V_{m_i p_j} = V_{m_i p_j}^k$, то $n_k = n_k + 1$.

2.6. $\forall a = \overline{1,i} \in L_{AB}$ та $\forall b = \overline{1,j} \in L_{AB}$ виконувати: якщо $L_{a_i b_j} = L_{a_i b_j}^t$, то $s_t = s_t + 1$, якщо $L_{a_i b_j} = L_{a_i b_j}^q$, то $s_q = s_q + 1$, якщо $L_{a_i b_j} = L_{a_i b_j}^t$, то $s_k = s_k + 1$.

2.7. З $m \in V_{MP}^t \cup a \in L_{AB}^t$ обчислити кількість операцій, що розраховується з матриці сформованої поведінки етапу потрапляння ТП на ПК, як $n_t' = \sum m_i + \sum a_i$.

2.8. З $m \in V_{MP}^q \cup a \in L_{AB}^q$ обчислити кількість операцій, що розраховується з матриці сформованої поведінки етапу активізації ТП як $n_q' = \sum m_i + \sum a_i$.

2.9. 2.8 З $m \in V_{MP}^k \cup a \in L_{AB}^k$ обчислити кількість операцій, що розраховується з матриці сформованої поведінки етапу виконання закладених функцій, як $n_k' = \sum m_i + \sum a_i$.

2.10. Виконати розрахунок ступенів належності вхідного вектора $N = (n_t, n_q, n_k)$, до нечітких термів $u_{i,jp}$ з бази знань так, що $\mu_{d_j}(n) = \cup w_{jp} \cdot \cap(\mu_{d_j}(n_i))$, де $j = \overline{1,m}$, операції $\cup$ та $\cap$ s- та t-норми відповідно.

2.11. Виконати розбиття універсуму вхідних змінних $X = [0,n]$ так, що для n_t, n_q, n_k згідно заданих інтервалів.

2.12. Виконати розбиття універсуму вихідної змінної $Y = [0,n]$ згідно заданих інтервалів.

2.13. Виконати нечіткий логічний висновок по базі знань як $\bigcup_1^{k_j}\left(\bigcap_1^n n_i = u_{i,jp}\right) \to y = d_j$, де $j = \overline{1,m}$, $u_{i,jp}$ - нечіткий терм вхідної змінної o_i, d_j - нечіткий терм вихідної змінної y.

2.14. Виконати агрегацію нечітких множин шляхом знаходження максимуму функцій належності як $y = \underset{j=\overline{1,m}}{agg}\left(\min\left(\mu_{u_j}(N), \mu_{u_j}(y)\right)/y\right)$, де об'єднання (агрегування agg) нечітких множин виконати як операцію знаходження максимуму.

2.15. Виконати процедуру знаходження чіткого значення вихідної змінної y шляхом виконання дефазифікації як

2.16. $y = \dfrac{\int_{\min}^{\max} y \cdot \mu_{u_j}(N)\,dy}{\int_{\min}^{\max} \mu_{u_j}(N)\,dy}$.

2.17. Якщо $y \le 0.6$, то перейти до виконання пункту 2.3, інакше якщо $0.6 < y \le 0.8$, то перейти до виконання алгоритму 3, якщо $0.8 < y \le 1$, то заблокувати виконання функцій об'єкта діагностування e та перейти до виконання алгоритму 3.

2.18. Кінець алгоритму.

3.3. Алгоритм 3. Нечіткий класифікатор підозрілих об'єктів

3.1. Сформувати вектор поведінок троянських програм $O = (o_1, o_2, ...o_n)$.

3.2. Сформувати вектор класів ТП $D=\{d_1,d_2,...,d_m\}$.

3.3. Розрахувати ступені належності об'єкта діагностування класам ТП як $\mu_{d_j}(O)=\cup w_{jp}\cdot\cap\left(\mu_{jp}(o_i)\right)$,

де $j=\overline{1,m}$, операції $\cup$ та $\cap$ s- та t-норми відповідно.

3.4. Задати та виконати розбиття універсаму $X=[0,n]$ на проміжки як в 2.12.

3.5. Виконати класифікацію як в 2.13.

3.6. Виконати нечітку класифікації підозрілого об'єкта як $y=\underset{\{d_1,d_2,...d_m\}}{\arg}\max\left(\mu_{d_1}(P),\mu_{d_2}(P),...,\mu_{d_m}(P)\right)$.

3.7. Кінець роботи алгоритму, виклик алгоритму 4.

3.4. Алгоритм 4. Виявлення факту підміни системних файлів троянськими версіями

4.1. Виконати формування об'єктів антивірусного діагностування e_k, що підлягають скануванню в ПК, де k – кількість об'єктів діагностування.

4.2. Виконати створення множини $s_i\in S$ для кожного об'єкта e_k розрядності $u=\overline{1,64}$: $s_i=\left\langle c_1..c_j,d_1..d_j,z_1..z_j,f_1..f_j,h_1..h_j\right\rangle$.

4.3. $\forall s_i\in S$ виконувати:

4.4. Створити бінарну послідовність $c_j\in s_i$, $j=\overline{17,32}$,

де c_j – 16 останніх результуючих розрядів контрольної суми CRC

4.5. Створити бінарну послідовність $d_j\in s_i$, $j=\overline{1,16}$,

де $j=\overline{1,5}$ – двійкове представлення дня місяця, $j=\overline{6,11}$ – двійкове представлення номера місяця, $j=\overline{12,16}$ – двійкове представлення року.

4.6. Створити бінарну послідовність $z_j\in s_i$, де $j=\overline{1,16}$ – двійкове представлення розміру файлу.

4.7. Створити бінарну послідовність $f_j\in s_i$, $j=\overline{1,2}$, причому якщо файл лише для читання, то $f_1=1$ інакше $f_1=0$, та якщо файл прихований, то $f_2=1$, $f_2=0$.

4.8. Створити бінарну послідовність $h_j\in s_i$, $j=\overline{1,14}$,

де h_i – 14 перших результуючих розрядів контрольної суми MD5.

4.9. Створити множину R детекторів $r_g\in R$,

де r_g – бітові послідовності розмірності $l=u$, $g=\overline{1,k}$,

де k – кількість детекторів.

4.10. $g=1$, поки $g<k$ виконувати випадковим чином генерацію двійкових послідовностей r_g.

4.11. Якщо $\forall s_i\in S\cup r_g\in R\quad r_g\leftrightarrow s_i$, то якщо виконання перевірки факту оновлення об'єкту e_k істина, то виконати $g=g+1$ та перейти до пункту 4.8, інакше вивести повідомлення про факт виявлення аномалії та виконати $g=g+1$.

4.12. Якщо об'єкт e_k зазнав оновлення, то виконати перехід на 4.3.

4.13. Кінець алгоритму.

4. Дослідження параметрів розроблених алгоритмів виявлення троянських програм в ПК

Розроблений алгоритм має два режими: автономний (створення «свого» та генерація детекторів) та оперативний (сканування ПК).

Швидкодія роботи алгоритму [6,7] в автономному режимі залежить від часу для виконання дій:

1. вибору файлів, які підлягають контролю щодо виявлення факту їх підміни троянськими версіями, та для створення їх бінарного представлення;

2. генерації кожного шаблону детекторів m^r;

3. розширення кожного дійсного шаблону до повністю визначеної послідовності;

4. оновлення шаблонів при створенні кожного детектора.

Час виконання оперативного режиму, при якому виконується безпосереднє виявлення факту підміни, залежить від кількості файлів, що підлягають перевірці та кількості детекторів, згенерованих під час роботи автономного режиму.

Швидкість роботи алгоритму сканування залежатиме від апаратного забезпечення персональ-

ного комп'ютера. Час роботи алгоритму в автономному режимі складе:

$t = O\left((l - r + 1) \cdot N_s \cdot m^r\right) + O\left((l - r + 1) \cdot N_r \cdot m^r\right)$ де

N_R – число рядків-детекторів після їх перевірки на збіг з рядками «свого»;

l – довжина рядка;

r – число розрядів послідовності збігу,

N_S – число рядків «свого».

Результати дослідження роботи алгоритму в залежності від його параметрів подано на рис. 2. Графік демонструє залежність часу виконання генерації детекторів від показника збігу r.

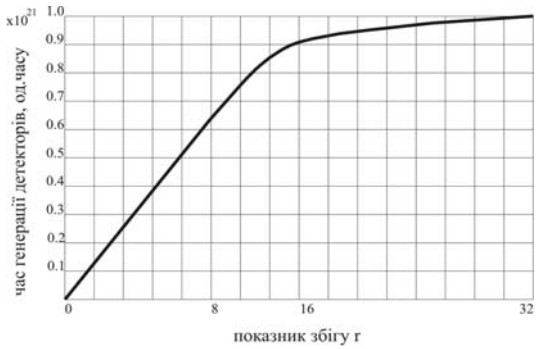

Рис. 2. Залежність часу генерації детекторів від показника збігу r

Розроблений алгоритм має ймовірнісний характер, тому можна виконати налаштування його параметрів, застосувавши:

$f = (1 - P_M)^{N_S}$, $N \geq (1 - P_f) / P_m$, де P_m – ймовірність збігу між двома випадковими рядками; P_f – ймовірність не збігу (не виявлення) випадкового рядка з будь-яким із N_S набору рядків «свого».

Результати дослідження ймовірності виявлення факту підміни файлів при різних параметрах алгоритму в представлено в таблиці 1 та на рис3. Як видно з рис.3 значно вищу ймовірність виявлення факту підміни алгоритм демонструє при r=32, проте при цьому необхідно генерувати більшу кількість детекторів.

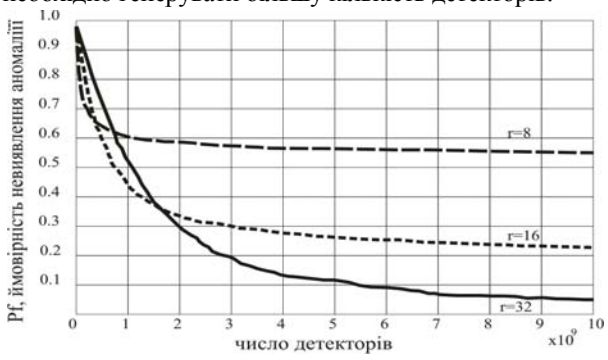

Рис. 3. Залежність ймовірності не виявлення аномалії від показника збігу r та кількості детекторів

Таблиця 1.

Ймовірності виявлення аномалії при різних параметрах його складових

m	r	l	P_m
2	8	64	0,001121
2	8	128	0,0001254
2	16	64	0,000255
2	16	128	0,000548
2	32	64	0,00458
2	32	128	0,000054

Розроблені алгоритми вимагають виконання наступних умов:

1. алгоритм моніторингу ПК та виконання НЛВ щодо підозрілості об'єкта діагностування використовує базу поведінкових моделей ТП. Тому є необхідним поповнення бази новими поведінками. Несвоєчасне додавання поведінок ТП може знизити можливість виявлення ТП;

2. алгоритм виявлення факту підміни системних файлів троянськими версіями демонструє високі показники ефективності своєї роботи. Проте на зниження показника правильної ідентифікації ТП може вплинути факт оновлення системного ПЗ. В алгоритмі передбачене виконання перевірки факту оновлення ПЗ, проте може виявитися неможливим виконання перевірки довіри до джерела оновлення ПЗ. Тому необхідно здійснювати моніторинг оновлення ПЗ та створення сховища довірених джерел.

Висновок

В результаті проведеного дослідження сучасних методів пошуку троянських програм виявлено їх недоліки: низька ймовірність виявлення нових троянських програм, високі вимоги окремих методів пошуку до апаратного забезпечення.

Для усунення недоліків розглянутих методів розроблено новий інтелектуальний метод пошуку троянських програм в персональних комп'ютерах, який базується на використанні алгоритмів нечіткої логіки та штучних імунних систем. Для реалізації інтелектуального методу пошуку троянських програм розроблено нові алгоритми, суть яких полягає у відслідковуванні системних подій разом із виконанням нечіткого логічного висновку щодо підозрілості об'єкта та перевірці файлів на факт їх підміни троянськими версіям за допомогою алгоритму негативного відбору. Результати дослідження розроблених алгоритмів показали можливість виявлення нових троянських програм з ймовірністю в 70-95 %, що дасть змогу їх застосуват в програмній реалізації системи пошуку троянських програм в персональних комп'ютерах.

Література

1. Савенко О. Дослідження методів антивірусного діагностування комп'ютерних мереж / О. Савенко, С. Лисенко // Вісник Хмельницького національного університету. – 2007. – № 2, т 2. – С. 120-126.

2. Савенко О. Модель процесу пошуку троянських програм в персональному комп'ютері / О. Савенко, С. Лисенко // Радіоелектронні і комп'ютерні системи. – 2008. – № 7(35). – С. 87-92.

3. Савенко О.С. Поведінкова модель троянських програм / О.С. Савенко, С.М. Лисенко // CSIT-2007: міжнар. наук.-техн. конф.: тези доповідей. – 2007. – С. 129-132.

4. Савенко О. Розробка процесу виявлення троянських програм на основі використання штучних імунних систем / О. Савенко, С. Лисенко //Вісник Хмельницького національного університету. – 2008. – №5. – С. 183-188.

5. Negative selection: How to generate detectors: How to generate detectors /M. Ayara, J. Timmis, L.N. de Lemos, R. de Castro //1st International Conference on Artificial Immune Systems, University of Kent at Canterbury, 2002. – P. 89-98.

6. D'haeseleer P. An immunological approach to change detection: algorithms, analysis, and implications / P. D'haeseleer, S. Forrest, P. Helman // Proc. IEEE Symposium on Research in Security and Privacy, Oakland, CA, May 1996.

Надійшла до редакції 15.01.2009

Рецензент: д-р техн. наук, проф. В.А. Заславський, Київський національний університет ім. Т.Г. Шевченка, Київ, Україна.

АЛГОРИТМЫ ПОИСКА ТРОЯНСКИХ ПРОГРАММ В ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРАХ

О.С. Савенко, С.Н. Лысенко

Исследовано ситуацию разработки вредоносного программного обеспечения. Представлено структуру системы поиска троянских программ в персональных компьютерах, которая основывается на интеллектуальном методе их поиска. Разработаны алгоритмы поиска троянских программ, которые в дальнейшем дадут возможность реализовать систему поиска троянских программ в персональных компьютерах.

Ключевые слова*:* троянская программа, антивирусное программное обеспечение, алгоритм поиска троянских программ, нечеткий логический вывод, алгоритм негативного отбора

THE SEARCH ALGORITHMS FOR TROJAN PROGRAMS IN THE PERSONAL COMPUTERS

O.S. Savenko, S.M. Lysenko

The situation of development of the malicious software is researched. The structure of search of trojan programs system in personal computers which is based on an intellectual method of their search is presented. Search algorithms for trojan programs which can be used in the search system for trojan programs in personal computers are developed.

Key words: trojan program, антивірусне програмне забезпечення, the search algorithm for the trojan programs, fuzzy inference, negative selection algorithm

Савенко Олег Станіславович – канд. техн. наук, доцент, декан факультету комп'ютерних систем та програмування Хмельницького національного університету, Хмельницький, Україна, e-mail: kism@beta.tup.km.ua.

Лисенко Сергій Миколайович – асистент кафедри системного програмування Хмельницького національного університету, Хмельницький, Україна, e-mail: sirogyk@ukr.net, foros@datasvit.km.ua, kism@beta.tup.km.ua.