Згідно з проведеним дослідженням, у кожній із систем було проведено запис дев'яти тестових сценаріїв (*V*=9). У семи сценаріях було навмисно зроблено помилки, тобто в семи випадках із дев'яти мало б виникнути «падіння» тесту. Система TestComplete виявила лише чотири помилки, Ranorex Studio – теж чотири, Test Studio – п'ять, Silenium IDE – усі сім помилок, але звіт по одній з них не відповідав дійсності «падіння» тесту. Що ж до розробленої системи, то вона виявила шість помилок із семи. У ході дослідження також обчислено показник покриття тестових сценаріїв, покриття коду та обраховано критерій завершеності тестування. Згідно з проведеними обчисленнями, автоматизоване тестування для даного сайту є доцільним: покриття майже усіх тестових сценаріїв (DV=9-1), Загальна кількість рядків коду – 472, з них протестовано 295. Показник покриття коду – 63% та показник завершеності тестування – близько 89%.

Висновки

У ході виконання роботи було досліджено чотири системи автоматизованого тестування GUI і створено нову систему, яка уможливлює автоматичне виконання тестування сайту продажу квитків через Інтернет, тобто система автоматично виконує функції взаємодії через GUI користувача. Показано, що запропонована система за критеріями якості переважає існуючі системи тестування.

Література

1. Автоматическая генерация тестов для графического пользовательского интерфейса по UML диаграммам действий / А.Я. Калинов, А.С. Косачёв, М.А. Посыпкин, А.А. Соколов // Труды Института системного программирования РАН. – 2004. – Т. 8. – Ч. 1.

2. Robinson Ray, AUTOMATION TEST TOOLS, Date Created: 1st March 2001, Last Updated: 11th Sept 2001.

3. www.wikipedia.org

4. Котляров В.П. Основы тестирования програмного обеспечения / В.П. Котляров, Т.В. Колякова. – М. : Бином, 2006. – 285 с.

5. Weilkiens T. Systems Engineering with SysMLUML Modeling, Analysis, Design. – Denise E. M. Penrose, 2007. – 320.

Рецензент: дт.н. Троцишин І.В.
Надійшла 13.2.2012 р.

УДК 004.492.3

С.М. ЛИСЕНКО, А.В. КРАСІЙ, В.В. МЕЛЬНИК
Хмельницький національний університет

ПОБУДОВА ПРОЦЕСУ ВИЯВЛЕННЯ ТРОЯНСЬКИХ ПРОГРАМ

В роботі досліджено достовірність сучасних засобів антивірусного діагностування та виявлено недоліки їх функціонування. Розроблено модель життєвого циклу троянських. Розглянуто особливості побудови процесу діагностування комп'ютерних систем на наявність троянських програм у режимах монітора та сканера.

In the article the main principles of antiviral diagnosis are proposed. Life circle Trojans model was developed. Main ideas of Trojan detection in monitor and scanner modes are presented.

Ключові слова: троянські програми, антивірусне діагностування комп'ютерних систем.

Вступ

Об'єднання комп'ютерних систем (КС) в локальні мережі та підключення їх до глобальної мережі Internet породжує багато проблем, пов'язаних з їх функціонуванням та використанням. Досить значні проблеми для роботи КС в локальних мережах створюють комп'ютерні мережні віруси, зокрема, їх наявність приводить до неправильного функціонування програмного та апаратного забезпечення.

Проблемі антивірусного діагностування комп'ютерних систем, що функціонують у режимі віддаленого доступу, коли ймовірність проникнення в систему шкідливих програмних об'єктів з боку віддалених комп'ютерних систем особливо висока, приділяється значна увага. Аналіз останніх результатів антивірусного діагностування КС показує динамічний ріст кількості комп'ютерних вірусних програм – програм чи деякої сукупності виконуваного коду, які можуть створювати свої копії, впроваджувати їх у файли, системні області комп'ютерної системи, зберігати здатність до розповсюдження, виконувати деструктивні дії. Серед них особливе місце займає окрема множина вірусних програм – троянські програми, які на відміну від класичних вірусних програм проникають у комп'ютерні системи з метою викрадення конфіденційної інформації, що становлять особливу небезпеку [1] і при цьому не створюють своїх копій.

Проведений аналіз АПЗ показав, що усі вони мають засоби діагностування КС на наявність троянських програм з використанням ІТ на основі сигнатурного аналізу, контрольних сум та евристичних аналізаторів. Проте деталі евристичних аналізаторів діагностування нових ТП недоступні та закриті для дослідження та вивчення. З огляду на це оцінка достовірності та ефективності антивірусного діагностування може бути здійснена шляхом дослідження результатів їх тестування.

Для оцінки достовірності антивірусного діагностування сучасних АПЗ дослідження результатів

тестування роботи з використанням ІТ на основі сигнатурного аналізу, контрольних сум, закладених в АПЗ, є недоцільним. Це зумовлено тим, що дані ІТ діагностують лише відомі ТП і на їх достовірність роботи впливає лише затримка оновлення антивірусних баз. Проте важливим є аналіз результатів роботи евристичних аналізаторів, які виявляють нові ТП.

Для аналізу достовірності антивірусного діагностування розглянутих АПЗ дослідимо дані з відомого ресурсу Virus Bulletin, рейтинг VB100 якого відображає результати останніх порівняльних тестів АПЗ [2]. Результати тестування евристичних аналізаторів представимо таблицею 1.

Таблиця 1

Кількість виявлених нових троянських програм при тестуванні евристичних аналізаторів АПЗ

Засіб антивірусного діагностування	Виявлено троянських програм, %
Avira	70
Microsoft	69
AVG	43
BitDefender 2009	41
Sophos	62
Kaspersky	64
McAfee	55
Nod32 (Eset)	66
Norton 2009 (Symantec)	41
Norman	36
Avast!	59

Результати тестування евристичних аналізаторів на предмет можливості виявлення нових невідомих троянських програм антивірусними програмними засобами представимо гістограмою на рис 1.

Також розглянемо результати тестування АПЗ на предмет достовірності роботи евристичних аналізаторів іншим відомим ресурсом AV-Comparatives [3]. Для тестування АПЗ було згенеровано нові невідомі ТП різних класів (таблиця 2). Оскільки сьогодні найчисельніший клас ТП – Backdoor, тому було окремо згенеровано 4966 ТП даного класу.

Таблиця 2

Результати тестування достовірності евристичних аналізаторів АПЗ

Засіб антивірусного діагностування	Виявлено ТП класу backdoor зі згенерованих 4966	Виявлено ТП інших класів зі згенерованих 13555
Avira	3737 (75%)	9523 (70%)
Avast!	2677 (54%)	5288 (39%)
AVG	2656 (53%)	5823 (43%)
BitDefender 2009	3087 (62%)	6607 (49%)
Norman AV	1637 (33%)	3028 (22%)
Microsoft AVs	3172 (64%)	7850 (56%)
Kaspersky	2826 (57%)	6353 (47%)
McAfee	1686 (34%)	3242 (24%)
Eset Nod32)	2894 (58%)	7416 (55%)
Norton Symantec	1761 (35%)	4690 (36%)

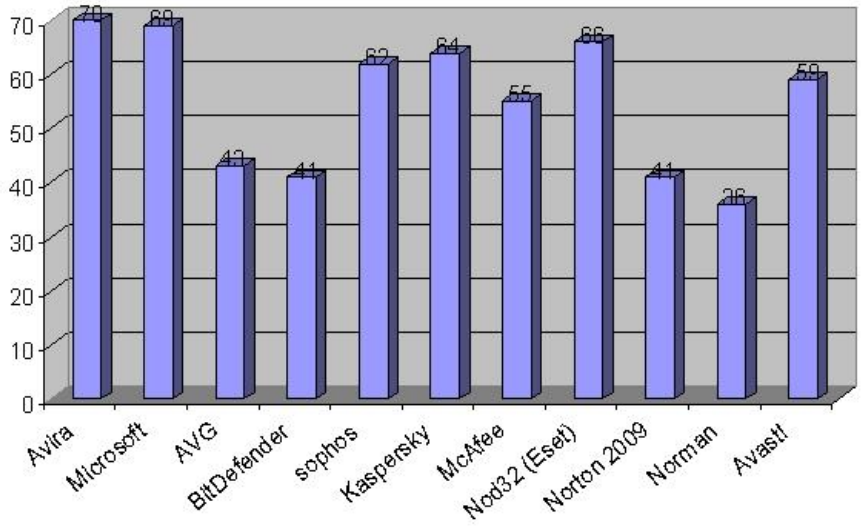

Рис. 1. Результати тестування АПЗ

Результати тестування евристичних аналізаторів на предмет можливості виявлення нових ТП антивірусними програмними засобами представимо гістограмою на рис. 2.

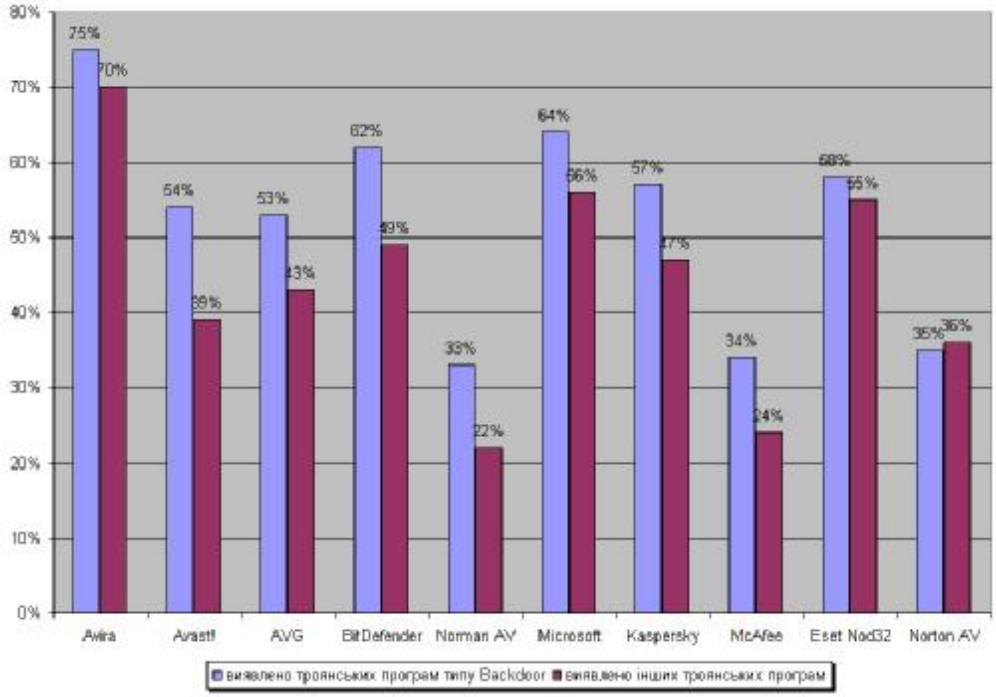

Рис. 2. Результати тестування АПЗ

Отримані дані відомих ресурсів тестування АПЗ, представлені у наведених таблицях, підтверджують факт низької достовірності діагностування КС на наявність троянських програм. АПЗ показали можливість виявлення лише 70% нових троянських програм.

Проведені дослідження програмних засобів діагностування КС на наявність троянських програм дозволяють зробити наступні висновки:

- сучасні програмні засоби діагностування КС не задовольняють вимогам ефективного антивірусного діагностування КС в умовах динамічного росту кількості ТП та росту рівня кваліфікації їх розробників, які швидко реагують на появу нових інформаційних технологій діагностування та добре знають класичні технології;
- тестування АПЗ показує низьку достовірність діагностування нових троянських програм засобами з використанням інформаційних технологій на основі евристичних аналізаторів;
- розробники АЗ приховують можливість виявлення нових троянських програм, наявні ж – демонструють низьку достовірність та ефективність;
- для підвищення ефективності антивірусного діагностування КС необхідним є розробка нової інформаційної технології діагностування КС на наявність ТП, яка б мала здатність виявляти відомі та невідомі ТП, розпізнавання здійснювалося б згідно життєвого циклу ТП, а інформаційна технологія діагностування мала б адаптивну основу.

Результати тестування свідчать про невисоку достовірність роботи АПЗ щодо діагностування нових ТП, що робить актуальною задачу розробки нової адаптивної інформаційної технології діагностування КС, яка б надала більшу достовірність діагностування КС на наявність нових невідомих ТП. Використання даної інформаційної технології дозволило б отримати нові програмні засоби, достовірність роботи яких була б вищою.

Постановка завдання

На основі проведеного аналізу сучасних інформаційних технологій діагностування КС на наявність ТП, недоліків їх роботи; та аналізу їх достовірності розробити основні принципи антивірусного діагностування КС в режимах монітора та сканера.

Дослідити структуру троянських програм та їх життєвий цикл.

Розробити поведінкову модель ТП та поведінкові моделі класів ТП із урахуванням їх функціонального навантаження. Розробити модель процесу діагностування комп’ютерних систем на наявність троянських програм.

Життєвий цикл троянських програм

Життєвий цикл (ЖЦ) троянських програм розділимо на три етапи: потрапляння на віддалену комп’ютерну систему, активізація та виконання закладених деструктивних дій в наборі функціональних можливостей троянської програми. Схему ЖЦ ТП зображено на рис.3.

Рис. 3. Схема життєвого циклу троянської програми

Проте дослідження різновидів ТП [4], їх характеристик таких, як функціональне навантаження та можливість функціонування в певній операційній системі показало, що часто ЖЦ ТП може бути видозмінений шляхом перестановки певних його етапів, чи відсутності одного із них взагалі. Тобто в структурі ЖЦ повинно бути хоча б дві складові із наведених на рис.4, причому етап потрапляння ТП в КС повинен бути обов'язково.

Наприклад, розглянемо загальну схему функціонування ТП класу Trojan-Backdoor клієнт-серверного типу (рис. 4).

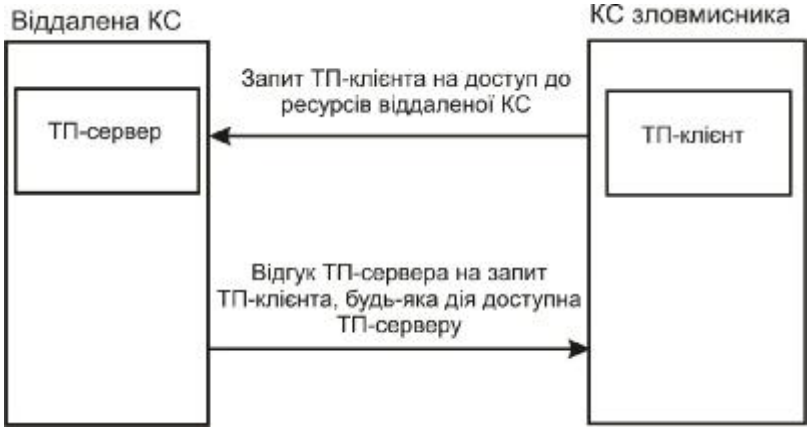

Рис. 4. Схема функціонування ТП типу клієнт-сервер

Дійсно, згідно з даною схемою троянська програма може пройти усі етапи свого ЖЦ, проте у різній послідовності. Наприклад, ТП проходить перший етап ЖЦ проникнення на віддалений КС шляхом маскування, де ТП видає себе за корисне програмне забезпечення, яке присутнє на web-ресурсі; шляхом прикріплення коду ТП до іншого файлу, що завантажується з мережі Internet; ТП завантажується шляхом активації скрипта, прикріпленого до певного посилання на web-ресурсі або шляхом кооперування з worm-вірусом, коли рух ТП мережею здійснюється в складі worm-вірусу. Далі присутня в КС ТП активізує себе шляхом додавання запису в системний реєстр ОС на автоматичний запуск свого тіла на виконання при кожному старті ОС. Третій етап ТП, що розглядається, характеризується діями, закладеними розробниками ТП.

Однак дана схема може підійти і до видозміненого ЖЦ, коли ТП потрапила в КС не через мережу, а шляхом запису з флеш-носія як корисне ПЗ (його серверна частина). Далі після свого запуску не прописуватися на автоматичне виконання, а відкрити будь-який системний порт на прослуховування сигналу від зловмисника (клієнтської частини троянської програми), який при подоланні локального захисту ОС може здійснювати керування віддаленою КС.

Основною складовою троянської програми є модуль її проникнення у віддалену комп'ютерну систему. В структурі троянської програми присутні три модулі (рис. 5). ТП, що потрапила в КС, містить механізм реєстрації себе в системі. Велика кількість троянських програм містять модуль активізації.

Кожна ТП має підпрограму деструктивних дій, тобто модуль виконання деструктивних дій. Наприклад, це може бути реалізація системи віддаленого керування КС у мережі, викрадення інформації, інші шкідливі дії. Особливо небезпечними є ТП, які реалізовані з використанням rootkit-технології, що дозволяє зловмиснику приховати присутність самої шкідливої програми в системі шляхом заміни системних бібліотек, перехоплення й модифікації низькорівневих API-функцій. Антивиявляючий модуль ґрунтується на використанні rootkit-технології.

Рис. 5. Загальна структура троянської програми

Розглянуті життєвий цикл, загальна структура та характерні поведінки ТП в КС дозволяють виконати побудову їх поведінкових моделей. Нехай Θ – множина програмних об'єктів, що потенційно можуть бути троянськими програми n_n, де $n_n \in \Theta$, $n \in N$, тобто $\Theta = \{\nu_1, \nu_2, ..., \nu_n\}$, де n – кількість троянських програм. Життєвий цикл троянської програми позначимо множиною S її етапів $s_i \in S$, $i = \overline{1,3}$. Нехай m – вектор дій та функцій, які реалізують способи та механізми здійснення потрапляння ТП на віддалену КС, p – вектор мережних протоколів, через системні порти яких здійснюється потрапляння ТП в КС. Приймемо a як вектор деструктивний дій троянської програми у віддаленій КС $a \in A$, та b – вектор структурних одиниць операційної системи комп'ютерної системи, які зазнають негативних впливів від деструктивних дій троянської програми $b \in B$. В поведінці ТП на етапі потрапляння ЖЦ такі поняття як дії, що дозволяють виконати потрапляння в КС, та системні порти, через які здійснюється таке потрапляння, знаходяться у бінарному відношенні. Оскільки таке бінарне відношення зручно подавати у вигляді двійкової

(булевої) матриці, у якій якщо і-тий елемент однієї множини відповідає j-му елементові другої множини, то введемо матрицю відношень способів потрапляння ТП в КС та портів $V = \left|V_{mp}\right|$ та матрицю відношень деструктивних дій ТП в КС і структурних одиниць ОС $L = \left|L_{ab}\right|$.

При такому способі представлення елементів поведінки троянських програм у випадку, якщо потрапляння ТП відбулося m-м способом через p-й системний порт, то при побудові матриці відношень V на перетині m-го стовпця та p-го рядка матриця має значення 1, та 0 у іншому випадку. Побудова матриці відношення L здійснюється аналогічно.

Введемо позначення $\longrightarrow$, суть якого полягає у заданні відношення між трьома поняттями, а саме: якщо $s_i \xrightarrow{a} s_{i+1}$, то дія $a \in A$ спричинює перехід із стану s_i в стан s_{i+1}, а результат має булевий тип 0 або 1 в залежності від того чи виконався перехід із стану інший стан чи ні.

Введемо функцію Aff, яка визначає взаємодію між об'єктами та троянськими програмами ν_j, тоді множина $a \in Aff(e_i, \nu_j)$ є набором можливих дій, які троянська програма ν_j завдає об'єкту (об'єктам) e_i. Номінально, $Aff(b_i, \nu_j)$ є множиною впливів, які ТП ν_j завдає об'єкту b_i.

Нехай ε – відношення між ТП та її станами, тоді для $\nu \in \Theta$ та $s \in S$, відношення $\nu \varepsilon s$ означає, що троянська програма ν перебуває в стані s; відношення $\nu \bar{\varepsilon} s$ означає, що троянська програма ν не перебуває в стані s. Приймемо шлях $s_i \xrightarrow{\omega} s_{i+1}$, де $w \in A^*$ ($A^* = \{a_1, a_2, ..., a_k, m_1, m_2, ..., m_n\}$), троянська програма $\nu \varepsilon s_i$ та $\nu \varepsilon s_{i+1}$ але $\nu \bar{\varepsilon} s_0$, де s_0 – стан троянської програми, яка ще не потрапила в КС і не є активізованою, стан s_0 не належить життєвому циклу ТП.

Додатково уведемо поняття Z характеристичних параметрів відношень. Тоді $Z = \{z_k\}$ є вектором деструктивних дій об'єкта з нормованими пріоритетними вагами $P = \{p_k\}$, ($\sum p_k = 1$), що враховують рівень їхньої небезпеки для комп'ютерної системи [5].

Виходячи з вищеописаних параметрів узагальнену поведінкову модель троянських програм представимо наступним чином:

$$M_v = \langle \Theta, S, V, L, Aff, \varepsilon, Z \rangle. \quad (1)$$

З огляду на узагальнену поведінкову модель троянська програма, життєвий цикл якої має усі етапи, проходить можливий шлях:

$$s_0 \xrightarrow{V,L} s_1 \xrightarrow{V,L} s_2 \xrightarrow{V,L} s_3, \quad (2)$$

де $s_i \xrightarrow{V,L} s_{i+1}$ означає можливість видозміненого життєвого циклу, коли, наприклад, етап потрапляння ТП в КС здійснюється не через мережу, або етап активізації виконується шляхом надходження сигналу через мережу, а не локально.

Оскільки об'єктом дослідження є процес діагностування КС на наявність ТП, то важливою задачею є розроблення його моделі, яка повинна включати здатність відображати особливості даного процесу та використовувати розроблені поведінкові моделі ТП.

Весь процес антивірусного діагностування КС на наявність ТП розділимо на два підпроцеси, які можуть працювати паралельно:

1) здійснення антивірусного моніторингу подій в КС;

2) виконання процедури антивірусного сканування КС на предмет виявлення факту підміни системних файлів троянськими версіями.

Здійснення антивірусного моніторингу повинно відбуватися в КС постійно з моменту запуску комп'ютерної системи.

Процедуру виконання антивірусного сканування КС на предмет виявлення факту підміни системних файлів троянськими версіями здійснюватимемо на вимогу користувача або через заданий квант часу.

Позначимо частини процесу антивірусного діагностування КС як Ω та Δ, де Ω – процес моніторингу, а Δ – сканування КС. Тоді формалізовану схему процесу діагностування КС на наявність ТП подамо як на рисунку 6.

Рис. 6. Узагальнена схема процесу діагностування КС на наявність ТП

Розглянемо кожну частину процесу діагностування КС на наявність ТП.

Процес антивірусного моніторингу складається з етапів $\Omega = \{\Omega_1, \Omega_2, \Omega_3, \Omega_4, \Omega_5, \Omega_6\}$, а виконання процедури антивірусного сканування КС на предмет виявлення факту підміни системних файлів троянськими версіями включатиме етапи $\Delta = \{\Delta_1, \Delta_2, \Delta_3, \Delta_4\}$ (див. рис. 2.20): Ω_1 – процедура

відслідковування (моніторингу) потоків, що здійснюються через системні порти КС; Ω_2 – процедура відслідковування виконання системних функцій в КС; Ω_3 – процедура блокування виконання програмним об'єктом системних функцій або функцій троянської програми, підозрілість яких визначена на інших етапах процесу антивірусного діагностування; Ω_4 – виконання процедури фазифікації в межах системи нечіткого логічного висновку (НЛВ) для введення нечіткості шляхом задання ступенів підозрілості функціонування ПЗ та ступенів небезпеки інфікування КС; Ω_5 – робота машини логічного висновку в межах системи НЛВ; Ω_6 – виконання процедури дефазифікації в межах системи НЛВ для визначення ступеня небезпеки інфікування КС троянською програмою.

Для реалізації кожного етапу необхідні відповідні параметри, що представлені множиною векторів $\Phi_{mon} = \{\bar{j}_1, \bar{j}_2, \mathbf{K}, \bar{j}_6\}$, де кожен вектор складається з множини параметрів відповідного етапу: $\bar{\Phi}_i = \{j_1^i, j_2^i, \mathbf{K}, j_{n_i}^i\}$, де n_i – кількість параметрів *i*-го етапу, $i = \overline{1,6}$ (рис. 7).

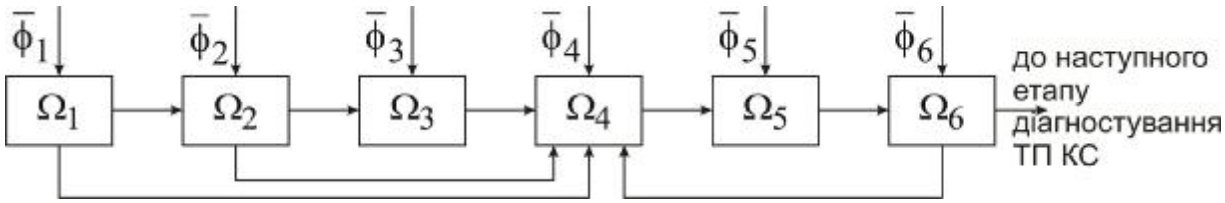

Рис. 7. Формалізована схема процесу моніторингу системних подій в КС

Процес сканування включає наступні етапи (див. рис. 8):

Δ_1 – виконання формування набору файлів, що підлягають процедурі створення набору захищених бінарних послідовностей; Δ_2 – виконання генерації набору шаблонів файлів, відібраних на попередньому етапі та виконання кодування даних у визначеному форматі; Δ_3 – виконання генерації детекторів; Δ_4 – виконання етапу сканування системи співставлення захищених двійковий послідовностей об'єктів антивірусного діагностування зі згенерованими на попередньому етапі детекторами.

Аналогічно процесу моніторингу, для реалізації кожного етапу процесу діагностування КС на наявність ТП для кожного етапу необхідні відповідні параметри, що представлені множиною векторів $\Phi_{scan} = \{\bar{j}_7, \bar{j}_8, \mathbf{K}, \bar{j}_{10}\}$, де кожен вектор складається з множини параметрів відповідного етапу: $\bar{\Phi}_i = \{j_1^i, j_2^i, \mathbf{K}, j_{n_i}^i\}$, де n_i – кількість параметрів *i*-го етапу, $i = \overline{1,4}$.

Таким чином, формалізована схема процесу діагностування КС на наявність ТП включає дві основні частини: моніторинг системних подій в КС та сканування КС на виявлення факту підміни системних файлів троянськими версіями. З урахуванням зворотних зв'язків між етапами процесу діагностування, які полягають у здійсненні передачі даних від етапу Ω_6 до Δ_1, схему процесу діагностування КС на наявність ТП представимо на рис. 9.

Рис. 8. Формалізована схема процесу сканування КС

Рис. 9. Формалізована схема процесу діагностування КС на наявність ТП

Для формалізації виконання етапів антивірусного діагностування подамо модель процесу діагностування КС на наявність ТП з урахуванням параметрів, які використовують вищевказані етапи у вигляді [7–9]:

$$M_u = \langle \{E, R, M_W, f_m\}, \{E, H, S, D, E_v, f_s\} \rangle, \quad (1)$$

де для етапів $\Omega_1 - \Omega_6$: E – множина об'єктів діагностування в режимі монітора $e_k \in E$, а саме множина файлів КС, причому $\Theta \in E$; R – результуюче число $R \in [0,1]$, яке свідчить про ступінь небезпеки інфікування КС троянською програмою; відношення e між об'єктами та станами, причому для $n \in \Theta$ та $s \in S$; M_W – множина поведінкових моделей троянських програм; $f_m(I_m, I'_m, I''_m)$ – функція адаптивності діагностування КС в режимі монітора, параметри якої змінюються в залежності від вхідних даних, де I_m – набір діагностичної інформації, $I_m = \langle \Theta, V, L, R, \rangle$); I'_m – вектор результатів антивірусного діагностування, $I'_m = \langle R_1, R_2, ..., R_n \rangle$; I''_m - набір даних про виявлене шкідливе програмне забезпечення, які збираються і використовуються в майбутньому як знання, $I''_m = \langle E, R \rangle$; для етапів Δ_1, Δ_2, Δ_3, Δ_4: E – множина об'єктів діагностування в режимі сканера $e_k \in E$, H – множина об'єктів $h \in H$, що підлягають процедурі сканування на предмет можливого факту їх підміни; S – множина захищених двійкових послідовностей $s \in S$; D – множина детекторів, згенерованих для сканування системи $d \in D$, E_v - множина файлів КС, що були підмінені троянськими версіями; $f_s(I_s, I'_s, I''_s)$ – функція адаптивності діагностування КС в режимі сканера, де I_s – набір діагностичної інформації, $I_s = \langle H, S, D \rangle$; I'_s – результати антивірусного сканування представлені набором файлів, що були підмінені троянськими версіями, $I'_s = \langle E_1, E_2, ..., E_n \rangle$; I''_s – вектор інформації про оновлення системних файлів та встановлення нового ПЗ, як компонентів об'єкта діагностування $I''_s = \langle E'_1, E'_2, ..., E'_n \rangle$ (рис. 10).

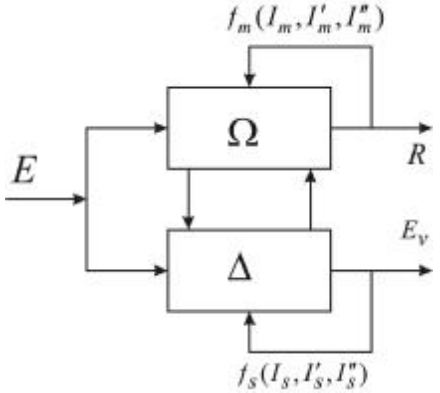

Рис. 10. Формалізована схема процесу діагностування КС на наявність ТП з урахуванням функцій адаптивності

Таким чином, троянська програма $v_n \in \Theta$, як і будь-який інший програмний об'єкт, що функціонує в КС, є скінченною послідовністю певних дії з множини $A^* = \{a_1, a_2, ..., a_i, m_1, m_2, ..., m_n\}$. Здійснення діагностування КС на наявність ТП за поведінкою програмних об'єктів виконуватиметься за допомогою постійного відслідковування та порівняння поведінки програмних об'єктів із поведінками відомих троянських програм. Множини поведінок бази на різних етапах ЖЦ ТП $T = \langle V^t, L^t \rangle$, $Q = \langle V^q, L^q \rangle$, $I = \langle V^i, L^i \rangle$ утворюють генеральну сукупність даних. Перевірка однорідності для різних поведінок не потрібна, оскільки в процесі діагностування комп'ютерних систем на наявність троянських програм порівняння поведінки програмних об'єктів відбувається з усіма поведінками в базі, а тому вибірки немає.

Запропонована модель процесу діагностування комп'ютерних систем на наявність троянських програм включає в себе дві незалежно-функціональні частини, які дають можливість здійснити виявлення шкідливих програмних об'єктів шляхом відслідковування системних подій та сканування комп'ютерних систем. Кожна функціональна частина має усі необхідні параметри для ефективного діагностування комп'ютерних систем. Процес діагностування є адаптивним, оскільки враховує можливість автоматичної зміни параметрів діагностування в залежності від особливостей комп'ютерних систем та накопичення знань для подальшого їх використання.

Модель процесу діагностування комп'ютерних систем на наявність троянських програм є основою для побудови адаптивної інформаційної технології діагностування комп'ютерних систем на наявність нових троянських програм, та дозволяє підвищити достовірність діагностування.

Висновки

З'ясовано, що жодна із відомих ІТ діагностування КС на наявність ТП, що існують на сьогоднішній день, не діагностують КС з високою достовірністю.

Розроблено поведінкові моделі троянських програм та класів троянських програм комп'ютерних систем шляхом врахування особливостей функціонування троянських програм протягом їх життєвого циклу та деструктивного характеру дій в комп'ютерній системі, що уможливлює підвищити достовірність їх виявлення в комп'ютерних системах.

Розроблено методику моделювання поведінки троянських програм та процес її занесення до антивірусної бази. Для перевірки вірності розроблених поведінкових моделей було доведено врахування в

розроблених поведінкових моделях усіх особливостей функціонування троянських програм в комп'ютерних систем шляхом порівняння реальних шкідливих програмних об'єктів із розробленими моделями.

Розроблено модель процесу діагностування комп'ютерних систем на наявність троянських програм, яка базується на залученні компонентів штучного інтелекту, зокрема нечіткої логіки та алгоритмів штучних імунних систем, і відрізняється від відомих тим, що використовує поведінкові моделі класів троянських програм, дозволяє адаптувати процес діагностування до окремо взятої комп'ютерної системи і не потребує побудови баз сигнатур.

Література

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

2. Williamson M. M. Virus throttling / M. M. Williamson, J. Twycross, J. Griffin, and A. Norman // Virus Bulletin. – 2009.

3. Proactive/Retrospective test. Anti-Virus comparative [Електронний ресурс]. – Режим доступу : http://av-comparatives.org.

4. Савенко О. Модель процесу пошуку троянських програм в персональному комп'ютері / Олег Савенко, Сергій Лисенко // Радіоелектронні і комп'ютерні системи. – 2008. – № 7. – С. 87–92.

5. Савенко О.С. Поведінкова модель троянських програм / О.С.Савенко, С.М. Лисенко // Комп'ютерні науки та інформаційні технології (CSIT-2007): міжнар. наук.-техн. конф., 27–29 вересня 2007 р. : тези доповідей. – 2007. – С. 129–132.

6. Система пошуку троянських програм з використанням нечіткого логічного висновку: зб. наук. праць міжнародної науково-практичної конференції [«Інтелектуальний аналіз інформації IAI-2008»], (Київ, 14-17 травня 2008 р.) / [редкол.: С.В. Сирота та ін]. – К.:Просвіта. – 2008. – С. 413–431.

7. Графов Р.П. Использование нечеткой логики для поиска троянских программных продуктов в вычислительных системах / Р.П. Графов, О.С. Савенко, С.М. Лисенко // Вісник Чернівецького національного університету. – 2009. – № 6. – С. 85–91.

8. Савенко О.С. Алгоритми пошуку троянських програм в персональних комп'ютерах / О.С. Савенко, С.М. Лисенко // Радіоелектронні і комп'ютерні системи. – 2009. – № 5. – С. 120–126.

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

Рецензент: к.т.н. Косенков В.Д.
Надійшла 13.2.2012 р.

УДК 004.891.3: 004.3

Т.О. ГОВОРУЩЕНКО, А.В. БАЧИНСЬКИЙ
Хмельницький національний університет

ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ МЕТРИК СКЛАДНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Авторами статті обрано метрики складності ПЗ етапу проектування та досліджено граничні значення метрик складності, а також проведено оцінювання ефективності кожної метрики складності ПЗ етапу проектування з точки зору оцінювання значущості її інформації за адитивним критерієм ефективності.

The authors chose the design stage software complexity metrics, investigated the limits of complexity metrics, conducted the evaluating of the effectiveness of each design stage software complexity metric in terms of its information relevance assessment for additive criterion of efficiency.

Ключові слова: програмне забезпечення (ПЗ), етап проектування ПЗ, метрики складності ПЗ етапу проектування, граничні значення метрик складності ПЗ, значущість метрик складності ПЗ, кількість інформації метрик складності ПЗ, ефективність метрик складності ПЗ, адитивний критерій ефективності.

Вступ

Сучасна індустрія програмного забезпечення характеризується високою конкуренцією. Для успішної роботи на цьому ринку софтверна компанія повинна розробляти, впроваджувати та супроводжувати ПЗ швидко, вкладаючись в термін, та із задовільною складністю. Ряд софтверних компаній вкладають значні кошти в модернізацію процесів розроблення ПЗ з метою підвищення його якості.

За особливостями і властивостями життєвого циклу програм їх доцільно ділити на ряд класів і категорій, з яких найбільш розрізняються два великих класи – малі й великі, що визначаються кількістю рядків програмного коду [1].

До класу програм малого розміру належать програми, які створюються одним або декількома (3–5)