УДК 004.75:004.49
Савенко О. С., Нічепорук А. О., Паюк В. П.
Хмельницький національний університет

ОЦІНКИ ЕФЕКТИВНОСТІ ТА ДОСТОВІРНОСТІ РОЗПОДІЛЕНИХ СИСТЕМ ВИЯВЛЕННЯ ЗЛОВМИСНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В КОМП'ЮТЕРНИХ СИСТЕМАХ ЛОКАЛЬНИХ МЕРЕЖ

Савенко О. С., Нічепорук А. О., Паюк В. П. Оцінка ефективності та достовірності розподілених систем виявлення зловмисного програмного забезпечення в комп'ютерних системах локальних мереж. В статті запропоновано оцінку ефективності та достовірності розподіленої багаторівневої системи виявлення зловмисного програмного забезпечення в комп'ютерних системах локальних мереж. Проведено огляд та характеристику відомих методів виявлення зловмисного програмного забезпечення на прикладі вірусних програм, що здійснюють мутацію власного коду, а також здійснено порівняння їх достовірності виявлення із розподіленою багаторівневою системою.
Ключові слова: достовірність, розподілена багаторівнева система, зловмисне програмне забезпечення.

Савенко О. С., Ничепорук А. А., Паюк В. П. Оценка эффективности и достоверности распределенных систем обнаружения вредоносных программ в компьютерных системах локальных сетей. В статье предложено оценку эффективности и достоверности распределенной многоуровневой системы обнаружения вредоносных программ в компьютерных системах локальных сетей. Проведен обзор и характеристика известных методов обнаружения вредоносных программ на примере вирусных программ, осуществляющих мутацию собственного кода, а также проведено сравнение их достоверности обнаружения с распределенной многоуровневой системой.
Ключевые слова: достоверность, распределенная многоуровневая система, вредоносное программное обеспечение.

Savenko O. S., Nicheporuk A. O, Payuk V. P. Estimation of efficiency and reliability of distributed malware detection systems in computer systems of the local network. Article presents an estimation of efficiency and reliability of the distributed multilevel system of malicious software detection in computer systems of the local network. An overview and description of known malware detection methods is done on the example of virus programs that perform a mutation of their own code, as well as a comparison of their reliability with a distributed multilevel system.
Keywords: accuracy, distributed multilevel system, malicious software.

Постановка проблеми. На сьогоднішній день актуальність проблеми виявлення зловмисного програмного забезпечення (ЗПЗ) не викликає жодних сумнівів. Оскільки нові типи шкідливих програм створюються і публікуються швидше, ніж додаються можливості виявлення, завжди є прогалина у виявленні, що призводить до інфікування все нових і нових користувачів.

Розвиток технологій та засобів написання шкідливого коду дозволив створювати зловмисне програмне забезпечення, яке представляє собою складні багатофункційні програмні системи та комплекси, які побудовані з використанням ефективних методів створення програмних засобів та методів поширення зловмисного коду. При цьому воно переважно розроблено для використання в комп'ютерних мережах. Для організації ефективної протидії таким засобам важливим є розробка таких систем виявлення ЗПЗ, архітектура яких враховувала б ці особливості. А використання ЗПЗ в мережах вимагає охоплення такими антивірусними системами вузлів мереж, що ставить питання до їх архітектури як до архітектури складного комплексу ЗПЗ. Крім того, досягнення підвищення достовірності виявлення ЗПЗ в межах тільки однієї комп'ютерної системи (КС), яка має вихід в мережу Internet, може бути недостатнім при протидії засобам ЗПЗ, які представлені великим програмним комплексом, що розміщений в багатьох комп'ютерних системах в глобальній мережі, і частини якого комунікують між собою.

У зв'язку з високою важливістю проблеми з поширенням ЗПЗ, протягом останніх років було представлено значну кількість підходів щодо виявлення ЗПЗ. Серед них в [1, 2] представлено архітектура розподіленої багаторівневої системи (РБС) для виявлення шкідливих програм у локальних мережах. Вона ґрунтується на принципах децентралізації та самоорганізації. Запропонована система дозволяє змінювати різні підходи до виявлення. Розподілена система постійно контролює запущені процеси виконуваних програм в комп'ютерних системах мережі. Компонентами розподіленої системи є автономні програмні одиниці з однаковою архітектурою, але кожен з них може самостійно приймати рішення на основі різних зібраних даних з різних комп'ютерних систем мережі. Розроблена розподілена система дозволяє ідентифікувати нові зразки ЗПЗ за допомогою зв'язку між програмними одиницями і відповідним функціональним змістом. Враховуючи складність всієї розробленої системи, необхідним є здійснення оцінки її ефективності, а також достовірності виявлення ЗПЗ.

Аналіз досліджень і публікацій. З метою оцінки ефективності розробленої РБС та порівняння результатів її роботи з відомими системами виявлення ЗПЗ, розглянемо детальніше підходи до виявлення ЗПЗ, що здійснюють мутацію власного вірусного коду (метаморфні та поліморфні віруси).

В роботі [3] представлено метод виявлення метаморфних вірусів, що заснований на представлені програми у вигляді графів, в яких частини коду і його взаємозв'язки моделюються вузлами та ребрами. Автори запропонували методику скорочення графів, що засноване на меншому впливі на семантику певних частин коду. Таке представлення дозволило виокремити семантично сталу частину коду, у якій відсутні команди сміття. Для здійснення процесу виявлення здійснюється пошук максимального ізоморфізму під графа. Запропонована методика була протестована на різних поліморфних вірусах, що буди зібрані із глобальної мережі.

Альтернативним підходом до виявлення метаморфних вірусів є виокремлення характеристичних ознак на основі інформації про потік виконання програми, що представлений послідовністю API викликів. У роботі [4] запропоновано підхід, що передбачає побудову графу потоку керування метаморфного вірусу, з подальшою конвертацією його у векторний простір. Для проведення класифікації було використано наступні класифікатори: одно рівневі дерева прийняття рішень, наївний Байєсів класифікатор, випадковий ліс, метод k-ближніх сусідів та метод послідовної мінімальної оптимізації (SMO).

Ще одним підходом до виявлення обфускованого шкідливого програмного забезпечення є використання техніки, заснованої на нормалізації коду. Ідея підходу полягає в отриманні унікального коду, який представляє всі метаморфні перетворення. Одним із способів цього є переклад коду на проміжну мову. В роботі [5] представлено підхід, який використовує MAIL (Malware Analysis Intermediate Language) з метою оптимізації та автоматизації виявлення шкідливих програм. Кожному оператору MAIL присвоюється шаблон, який можна використовувати для анотації графа потоку керування для відповідності шаблону. Основною метою MAIL є представлення структурної та поведінкової інформації про збірку програми для аналізу та виявлення шкідливих програм таким чином, щоб зробити програму більш читабельною та зрозумілішою. Для здійснення виявлення підозрілу програму спочатку дизасемблюють і переводять у програму MAIL. Граф потоку керування програми (CFG) згодом будується з анотованої програми, яка стає частиною підпису програми і узгоджується з базою даних відомих зразків шкідливих програм для класифікації. Далі CFG поділяється на менші підграфи, що відповідають окремим функціям в шкідливому коді. Для проведення класифікації здійснюється визначення відсотку збігу CFG підозрілої програми із відомими шаблонами, що присутні в базі даних.

В роботі [6] запропоновано визначати схеми повторних інструкцій, що містяться у шкідливих програмах. Авторами побудовано структуру даних, що називається матриця виникнення інструкцій (Instruction Occurrence Matrix), що пов'язує кожен код операції з числом унікальних інструкцій, які містять код операції, і зустрічаються принаймні два рази в програмному коді. Потім IOM відправляється в класифікатор машинного навчання, який встановлює, чи є підозрілий код вірусом чи ні. Для проведення класифікації залучено п'ять алгоритмів, що засновані на деревах прийняття рішень, в яких проміжні вузли містять умови, що застосовуються до одного з атрибутів, вибраних для класифікації, а кінцеві вузли представляють передбачений клас.

Автори роботи [7] використали апарат прихованих марківських моделей (ПММ) для виявлення сімейств метаморфних вірусів. Перевагою використання ПММ є їх здатність знаходити прихований стан або базову структуру в заданих послідовних даних. Автори здійснили навчання багатьох ПММ для різних метаморфних генераторів шкідливих програм, використовуючи представлення опкодів інструкцій у вигляді n-грам. З метою усунення ознак, що зменшують точність навчання моделей, використано міру TF-IDF (TF – term frequency, IDF – inverse document frequency). Результати проведених експериментів показали ефективність виявлення близько 90% для трьох різних сімейств вірусів: zbot, winwebsecand та zeroaccess.

Виклад основного матеріалу дослідження. Здійснення оцінки ефективності розробленої РБС проведемо в порівнянні її використання в хостовому представленні одним програмним модулем та системою, в якій більше одного програмного модуля. Таке порівняння дозволить встановити додаткові витрати на операції обміну між програмними модулями та витратами на залучення додаткових ресурсів КС. Визначення загальної оцінки ефективності розробленої РБС виявлення ЗПЗ в КС локальної мережі здійснимо на основі її суттєвих характеристик та показників, які впливають на її функціонування, завантаженість ресурсів КС та трафіку в мережі. До критеріїв ефективності для

розробленої РБС віднесемо такі: витрати часу на здійснення процесу виявлення хостом; витрати часу на здійснення процесу виявлення всією РБС; оперативність в прийнятті рішень; ресурсоспоживання; достовірність виявлення.

Оскільки, РБС при прийнятті рішень використовує інформацію з ПМ окремо, а також залучає групу ПМ для прийняття рішень, то здійснимо визначення ефективності її ПМ окремо.

Визначимо час, який витрачається на здійснення процесу виявлення ЗПЗ в КС локальної мережі, наступним чином:

$$t_{ПМ} = \sum_{i=1}^{10} t_i , \tag{1}$$

де $t_{ПМ}$ – час роботи окремого ПМ РБС; t_1 – час перебування ПМ в стані 1; t_2 – час витрачений на перевірку всіх виконуваних програм в окремій КС програмним модулем РБС без залучення решти ПМ, тобто перебування ПМ в стані 2; t_3 – час перевірки процесів та мережних пакетів одним ПМ в КС, тобто передування в стані 3; t_4 – час витрачений на перевірку в окремій КС програмним модулем РБС без залучення решти ПМ, тобто перебування ПМ в стані 4; t_5 – час витрачений на перевірку вибраних виконуваних програм в окремій КС програмним модулем РБС із залученням декількох ПМ, тобто перебування ПМ в стані 5; t_6 – час перевірки процесів та мережних пакетів одним ПМ в КС із залученням декількох ПМ, тобто перебування в стані 6; t_7 – час витрачений на перевірку в окремій КС програмним модулем РБС із залучення декількох ПМ, тобто перебування ПМ в стані 7; t_8 – час витрачений на здійснення оптимізації інформації в окремій КС програмним модулем РБС без залучення решти ПМ, тобто перебування ПМ в стані 8; t_9 – час витрачений на відправку пакетів із завданнями для декількох ПМ РБС; t_{10} – час витрачений на обробку завдань від інших ПМ РБС. Якщо ПМ перебував в стані 1 постійно, тоді час витрачений на його функціонування співпадає з часом перебування в стані 1. Час перебування ПМ в станах 1-4 та 8 відповідає одноосібній роботі ПМ РБС. Співвідношення, яке визначається формулою 2, встановлює частку часу p_1 роботи ПМ одноосібно в складі РБС:

$$p_1 = \frac{\sum_{i=1}^{4} t_i + t_8}{t_{ПМ}} . \tag{2}$$

Якщо ПМ не здійснював зв'язку з іншими ПМ РБС, тоді $p_1 = 1$. В протилежному випадку буде зафіксована частка часу, яка дозволить встановити час взаємодії ПМ з іншими ПМ РБС. Аналогічно введемо частку часу p_2 роботи ПМ на залучення інших ПМ РБС, яку визначимо за формулою 3:

$$p_2 = \frac{\sum_{i=5}^{7} t_i + t_9}{t_{ПМ}} . \tag{3}$$

А також, введемо частку часу p_3 на обробку запитів від інших ПМ РБС і визначимо її за формулою 4:

$$p_3 = \frac{t_{10}}{t_{ПМ}} . \tag{4}$$

Час витрачений на обробку запитів від інших ПМ РБС включає тривалість передачі одиниці обсягу даних для окремого ПМ з решти ПМ мережі, тривалість дослідження програмного коду ймовірного ЗПЗ та тривалість відправлених решті ПМ в мережі результатів. Тому, введемо для цих показників відповідний час, позначивши $t_{10,1}$ – тривалість передачі, $t_{10,2}$ – тривалість дослідження, $t_{10,3}$ – тривалість відправлення результатів. Тоді, $t_{10} = t_{10,1} + t_{10,2} + t_{10,3}$. В результаті витрати часу на здіснення процесу дослідження програмних об'єктів всією РБС отримуємо за формулою 4:

$$t_{РБС} = \sum_{i=1, i \neq j}^{n} a_i \cdot t_{i,10} + t_k , \tag{5}$$

де $t_{РБС}$ – витрати часу всією РБС на вирішення завдань, які виникають в одному з ПМ; a_i – бінарні

коефіцієнти, які вказують на залучення/не залучення *i*-того ПМ для вирішення завдань *j*-того ПМ; $t_{i,10}$ – витрати часу *i*-того ПМ для вирішення завдань *j*-того ПМ; t_k – час витрачений *j*-тим ПМ для вирішення завдання в k -тому стані; *n* – кількість ПМ в РБС.

В результаті досягнення ефективності роботи всієї РБС в порівнянні з одним ПМ полягає у визначенні відношення частки між ними, яке представлено формулою 6:

$$\frac{t_{РБС}}{t_k} = 1 + \frac{\sum_{i=1, i \neq j}^{n} a_i \cdot t_{i,10}}{t_k}, \tag{6}$$

де – час витрачений *j* – тим ПМ для вирішення завдання в *k* – тому стані; *n* – кількість ПМ в РБС.

Критерій ефективності полягає в мінімізації величини $k_e = \frac{t_{РБС}}{t_k}$.

Якщо величина в чисельнику дорівнює нулеві, тобто інші ПМ РБС не залучались до виконання завданння одного з ПМ, і при цьому завдання виконано, то результат є найефективнішим. Якщо ж величина в чисельнику відмінна від нуля, тобто залучався для виконання завдань одного з ПМ хоча б один з ПМ РБС додатково, то тоді порівняння таких значень вказує на ефективність роботи РБС при вирішенні завдань як певного типу, що вимагає перебування в певному стані ПМ, так і кількості залучених ПМ. Таким чином, показник ефективності РБС за часом визначається за формулою 5.

Оперативність в прийнятті рішень розробленою РБС полягає в мінімізації часу від моменту виявлення ЗПЗ до його блокування не тільки в одній КС, в якій було виявлено, але і в усіх інших КС, в яких необхідне дослідження на наявність такого зловмисного програмного коду. Крім того, оперативність включатиме мінімізацію часу вимкнення РБС ураженої КС, в якій міститься ПМ РБС. Показники оперативності визначаються за формулами 7:

$$p_{o,1} = \min |\max |a_i \cdot t_{i,10}| + t_k|, \tag{7}$$
$$p_{o,2} = \min |t_{11,j} + \max |t_{i,12}||,$$

де a_i – бінарні коефіцієнти, які вказують на залучення/не залучення *i* – того ПМ для вирішення завдань *j* – того ПМ; $t_{i,10}$ – витрати часу *i* – того ПМ для вирішення завдань *j* – того ПМ; t_k – час витрачений *j* -тим ПМ для вирішення завдання в k -тому стані; $t_{11,j}$ – витрати часу *j*-того ПМ для вимкнення *j*-ої КС; $t_{i,12}$ – витрати часу *i*-тим ПМ для прийняття рішення про вимкнення *j*-того ПМ.

Ресурсоспоживання окремою КС та всіма КС в локальній мережі, в які встановлені ПМ РБС залежить від залучених ресурсів для дослідження програмного коду ймовірно зловмисного ПЗ в КС. Такими характеристиками є середня тривалість емуляції *j*-ї інструкції *i*-тої КС мережі та прогнозована кількість при обробці завдань за час роботи ПМ в КС. Ресурсоспоживання r_e визначимо на основі середніх обсягів витраченої оперативної пам'яті, завантаженості процесора для виконання процесів системи виявлення ЗПЗ та середній обсяг даних, які передані каналами мережі.

Для визначення достовірності процесу виявлення ЗПЗ використаємо формулу, що заснована на залученні матриці помилок (confusion matrix):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{8}$$

де *TP* – кількість вірно виявлених зразків ЗПЗ, *FN* – кількість хибно класифікованих зразків ЗПЗ, *TN* – кількість вірно ідентифікованих корисних програм, *FP* – є кількість корисних програм неправильно класифікованих як ЗПЗ.

Для визначення показника, що використовуватиметься при розрахунку загальної ефективності, врахуємо час, який витрачено на помилки першого, другого та третього роду, який визначимо за формулою 9:

$$t_{d,РБС} = t_{d,1} + t_{d,2} + t_{d,3}, \tag{9}$$

де $t_{d,РБС}$ – час витрачений на помилки при виявленні; $t_{d,1}$ – час витрачений на помилки першого роду; $t_{d,2}$ – час витрачений на помилки другого роду; $t_{d,3}$ – час витрачений на помилки третього роду. Також, потрібно врахувати виникнення цих помилок в різних КС при дослідженні одного і того ж програмного об'єкту. Показник достовірності полягатиме в мінімізації величини p_d, яка визначається за формулою 10:

$$p_d = \min |1 - \frac{t_{d,РБС}}{t_{РБС}}|, \quad (10)$$

Визначимо показник ефективності витрат часу на здійснення процесу виявлення ЗПЗ в локальних мережах наступним чином: унормуємо показники p_d, $p_{o,1}$, $p_{o,2}$, k_e та знайдемо їх добуток.

Отже, дослідження характеристик РБС виявлення ЗПЗ надало можливість розробити методику для визначення ефективності мережних систем. Отримана методика дозволить здійснити порівняння ефективності розробленої РБС з відомими системами.

Експериментальні дослідження. Для проведення експерименту було використано 150 зразків вірусних програм, що отримані з ресурсу VX Heavens [8]. Всі вірусні програми належать до вірусних сімейств: Virut, Zmist та Bifrose (по 50 екземплярів кожного вірусного сімейства). Окрім вірусних програм було залучено 50 корисних додатки, що є виконуваними файлами операційної системи Windows (mspaint, bfsvc, тощо).

Метою проведення експерименту було визначення рівнів достовірності виявлення і кількості хибних спрацювань для розробленої РБС та відомих систем виявлення ЗПЗ. Для визначення достовірності виявлення розробленої РБС було використано формулу (8). Для відомих систем було використано задекларовані у першоджерелі показники достовірності виявлення та хибного спрацювання. Результати проведеного експерименту наведено у таблиці 1.

Таблиця 1 – Порівняння достовірності виявлення поліморфних вірусів розробленою РБС та відомими системами

Назва роботи	Достовірність виявлення Accuracy (найкращий показник)	Рівень хибних спрацювань (найкращий показник)	Розмір тестової вибірки (довірені додатки/зловмисне програмне забезпечення)
Розподілена багаторівнева система [1]	96%	5%	50/150
Malware Detection based on Dependency Graph using Hybrid Genetic Algorithm [3]	88%	11%	0/18
A graph mining approach for detecting unknown malwares [4]	97,8%	48%	2140/2305
MAIL: Malware Analysis Intermediate Language [5]	93,92	3,02%	1137/250
Static analysis for the detection of metamorphic computer viruses using repeated-instructions counting heuristics [6]	94%	2%	500/500
Identifying metamorphic virus using n-grams and hidden markov model [7]	90%	4%	–

Отриманні результати свідчать, що достовірність виявлення поліморфних вірусів розробленою РБС склала близько 96% з рівнем хибних спрацювань 5%.

Висновки. Розроблені аналітичні залежності для здійснення оцінки ефективності розподілених систем виявлення ЗПЗ. Вони дозволяють оцінити часові показники ефективності, що є важливим для впровадження та використання таких систем.

Проведено огляд та характеристику відомих методів виявлення зловмисного програмного забезпечення на прикладі вірусних програм, що здійснюють мутацію власного коду, а також здійснено порівняння їх достовірності виявлення із розподіленою багаторівневою системою.

Напрямком подальших досліджень є дослідження впливу апаратних компонентів системи на їх ефективність в цілому.

Reference

1. Markowsky G. Distributed Malware Detection System Based on Decentralized Architecture in Local Area Networks / G. Markowsky, O. Savenko, A. Sachenko // Advances in Intelligent Systems and Computing. – 2019. – Vol. 871. – P.582–598.
2. Савенко О.С. Архітектура багаторівневої програмної системи виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах. / О.С. Савенко, В.І. Грибинчук, М.О. Кульчицький // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2018. – № 30-31. – С. 132-140.
3. Kim K. Malware Detection based on Dependency Graph using Hybrid Genetic Algorithm / K. Kim, B. R. Moon // Proceedings of the 12th annual conference on Genetic and evolutionary computation. – Portland, Oregon (USA), 07–11 July, 2010. – P. 1211–1218.
4. Eskandari M. A graph mining approach for detecting unknown malwares / M. Eskandari, S. Hashemi // Journal of Visual Languages and Computing. – 2012. – Vol. 23. – No 3. – P. 154–162.
5. Alam S. MAIL: Malware Analysis Intermediate Language / S. Alam, R. N. Horspool, I. Traore // Proceedings of the 6th International Conference on Security of Information and Networks. – Aksaray (Turkey), 26-28 November, 2013. – P. 233–240.
6. Canfora G. Static analysis for the detection of metamorphic computer viruses using repeated-instructions counting heuristics / G. Canfora, A. N. Iannaccone, C. A. Visaggio // Journal of Computer Virology and Hacking Techniques. – 2013. – Vol. 10. – No 1. – P. 11–27.
7. Thunga S. P. Identifying metamorphic virus using n-grams and hidden markov model / S. P. Thunga, R. K. Neelisetti // Proceedings of the 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI). – Kochi (India), 28 September, 2015. – P. 2016–2022.
8. VX Heavens Computer virus collection [Electronic resource: [Web-site]. – Electronic data. – Mode of access: http://vx.netlux.org (Viewed on May 7, 2019). – Title from the screen.