УДК 004.491.42

Савенко О.С.
Хмельницький національний університет

АРХІТЕКТУРА РОЗПОДІЛЕНОЇ БАГАТОРІВНЕВОЇ СИСТЕМИ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В ЛОКАЛЬНИХ КОМП'ЮТЕРНИХ МЕРЕЖАХ

У статті запропоновано архітектуру розподіленої багаторівневої системи виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах, особливістю якої є синтез вимог розподіленості, децентралізованості, багаторівневості, та самоорганізованості, що, на відміну від відомих систем, дозволяє її використовувати автономно для виконання закладених функцій виявлення шкідливого програмного забезпечення. Крім того, програмні модулі системи представлено на основі структури Кріпке, особливістю якої є така самоорганізація, що дає змогу здійснювати обмін знаннями всередині системи, що, на відміну від відомих систем, дозволяє використовувати знання, отримані окремими частинами системи в інших частинах.

Ключові слова: *архітектура, розподілена система, шкідливе програмне забезпечення, метаморфні віруси, структури Кріпке.*

Постановка проблеми. Дослідження відомих програмних засобів виявлення нового шкідливого програмного забезпечення (далі – ШПЗ) вказують на недостатньо високий рівень достовірності виявлення [1]. Реалізація нових принципів, моделей та методів виявлення конкретних типів шкідливого програмного забезпечення шляхом створення відповідних інформаційних систем потребує подальшого розвитку, але досягнення підвищення достовірності виявлення нового та відомо ШПЗ, у якому застосовуються комбіновані технології зловмисного проникнення та поширення, можливе також за умови комбінованого використання нових та відомих методів виявлення.

Сучасне шкідливе програмне забезпечення – це складні багатофункційні програмні системи та комплекси, побудовані з використанням ефективних методів створення програмних засобів та методів поширення зловмисного коду. Її розроблено для використання в комп'ютерних мережах (наприклад, ботнет). Для організації ефективної протидії таким засобам важливим є розроблення системи виявлення ШПЗ, архітектура якої була б подібною до архітектури складного комплексу ШПЗ, але мала б більшу функціональність. Крім того, досягнення підвищення достовірності виявлення ШПЗ у межах тільки однієї комп'ютерної системи (далі – КС), яка має вихід у мережу Internet, може бути недостатнім для протидії засобам ШПЗ, які представлені великим програмним комплексом, що розміщений у багатьох комп'ютерних системах у глобальній мережі. Тому підвищення достовірності виявлення ШПЗ можливе за рахунок залучення груп КС в порівнянні з однією КС. Такими групами можуть виступати КС локальних комп'ютерних мереж (ЛКМ). Тому важливим завданням є здійснення побудови таких ефективних систем для їх використання в локальних комп'ютерних мережах, де уможливлюється встановлення таких систем на усіх комп'ютерних системах мережі, а не локально. Це дозволить підвищити достовірність виявлення під час проведення аналізу зібраних типових даних із різних КС.

Для того, щоб ефективно застосовувати методи та засоби виявлення шкідливого програмного забезпечення, необхідно розробити систему, яка б містила достатню кількість реалізованих ефективних методів у вигляді відповідних підсистем, мала можливість до нарощування та враховувала б майбутні тенденції розвитку як антивірусних засобів, так і шкідливого програмного забезпечення.

Аналіз останніх досліджень і публікацій. Сьогодні значна увага приділяється проблемі виявлення ШПЗ, яка виявляється в порушенні роботи КС, викраденні персональних даних, втручанні у роботу користувачів тощо. Для виявлення

та протидії ШПЗ розроблено низку систем та методів.

У роботі [2] запропоновано систему виявлення кібератак на основі залучення нейромережевих імунних детекторів. Розроблена система складається з двох частин. Перша частина реалізована апаратно й працює постійно в режимі реального часу. Друга частина представлена програмним забезпеченням на виділеному комп'ютері, який використовується для аналізу поточних атак та створення відповідних засобів захисту. Прийняття рішення про можливий уплив ШПЗ здійснюється із залученням системи нейромережевих детекторів, в основу якої закладено алгоритм Мамдані.

У роботі [3] представлено інтелектуальну адаптивну систему виявлення ШПЗ на основі інтеграції штучних імунних систем та штучних нейронних мереж. Така система працює за основними принципами штучної імунної системи, де імунні детектори представляють нейронну мережу і виявляють шкідливий шаблон за допомогою аналізу структури виконуваного коду. Запропонована система володіє функціями самоадаптивності та самонавчання, дозволяє здійснювати виявлення як відомих, так і нових видів ШПЗ.

Авторами роботи [4] запропоновано систему ідентифікації та класифікації для мережних кібератак. Для реалізації системи запропоновано використання комбінації різних методів штучного машинного навчання, зокрема нейронних мереж, імунної системи, нейрофізичних класифікаторів та метод опорних векторів. Особливістю запропонованої системи є багаторівневий аналіз мережевого трафіка, що дає можливість виявляти атаки методом підпису та комбінувати набір адаптивних детекторів на основі методів машинного навчання.

У роботі [5] запропоновано гібридну систему виявлення рідковживаних кібератак на основі використання штучних імунних систем та нечіткої кластеризації (FC-ANN). Спочатку FC-ANN розподіляє навчальні дані на декілька підгруп із використанням методів нечіткої кластеризації. Для отримання остаточних результатів визначається оцінка для кожного елемента зі сформованих підгруп та виконується їх поєднання з використанням штучної імунної системи.

Систему для виявлення веб-аномалій, що заснована на аналізі збережених на КС POST і GET запитів, представлено в роботі [6]. Авторами запропоновано файл реєстрації (log-файл) поведінки для визначення легітимних користувацьких сеансів. На основі файлу реєстрації здійснюється відбір ознак, що підлягають машинному навчанню. Для встановлення висновку про наявність веб-аномлії залучаються штучні імунні системи та нейронні мережі.

Проведений аналіз показав, що для виявлення ШПЗ відомі системи здійснюють аналіз мережного трафіка, файлів аудиту, пакетів, що передаються по мережі, перевіряють конфігурацію відкритих мережевих сервісів. Для встановлення факту порушення роботи КС відомі системи використовують різні методи машинного навчання, зокрема нейронні мережі, штучні імунні системи, метод опорних векторів, Байєсові мережі, нечітку кластеризація. Проте основним недоліком представлених систем є їх хост-орієнтований підхід до виявлення ШПЗ.

Постановка завдання. Ефективність та достовірність виявлення шкідливого програмного забезпечення, що здійснюється за допомогою різноманітних засобів, суттєво залежать від архітектури таких засобів, а також їх позиціонування та місця розміщення в комп'ютерних системах локальних мереж. Ураховуючи те, що процес виявлення ШПЗ проводитиметься в локальних мережах, то вибір моделі функціонування системи повинен передбачати залучення інформації з усіх комп'ютерних систем локальної мережі, тобто розміщення на всіх КС цієї системи. Це необхідно для підвищення ефективності й достовірності виявлення за рахунок врахування інформації про стан з інших КС для прийняття рішення на конкретній КС. Ці основні вимоги, що система повинна бути розміщена в мережі на кожній КС, впливають на вибір моделі її архітектури. Також важливим для таких систем є те, щоб центр прийняття рішень системи не представлявся та ідентифікувався однозначно, бо його виявлення призведе до атаки на нього для виведення всієї системи з робочого стану. Система повинна бути побудована так, щоб розміщені в КС локальної мережі її частини ефективно спілкувалися між собою для обміну інформацією про стан КС із метою надання додаткової інформації для прийняття рішення. Крім того, система виявлення ШПЗ повинна відповідним чином структуруватись, щоб мати можливість нарощуватись і щоб збільшення не сповільнювало процес виявлення.

Виклад основного матеріалу дослідження. Ураховуючи те, що система функціонуватиме в локальній комп'ютерній мережі та вирішуватиме багато різних завдань із виявлення ШПЗ, її визначальними характеристиками буде розподіленість та багаторівневість. Назвемо її розподіленою багаторівневою системою (далі – РБС).

Основними функціями РБС є перевірка наявного програмного забезпечення та запущених процесів у КС локальної мережі на можливість належності до шкідливого програмного забезпечення. Досягнення відповідності системи заданим характеристикам та покладеним на неї функціям із виявлення ШПЗ формують вимоги до неї, основними з яких будуть такі: розподіленість; децентралізованість; самостійність у прийнятті рішень; багаторівневість; самоорганізованість; адаптивність. Згідно з аналізом поставленої задачі та сформованими вимогами до системи виявлення ШПЗ на основі функцій системи та її характеристик, архітектуру розроблюваної системи можна синтезувати у вигляді сукупності таких компонентів: моделей колективного інтелекту, багатоагентних систем, розподілених систем, децентралізованих систем, самоорганізованих та адаптивних систем. Урахування цих складових частин у моделі системи є основою її архітектури; вона підвищить надійність функціонування та живучість системи в локальній мережі. Інтегруємо основні вимоги в модель розроблюваної системи, архітектура якої зображена узагальненою схемою основних складових частин на рис. 1.

Поетапно ця розподілена архітектура буде наповнюватись підсистемами, у яких реалізовуватимуться інші моделі, що входять до неї. РБС розподілена в просторі і (згідно з характеристичними вимогами) повинна бути децентралізованою, тобто система не повинна мати єдиного центру керування всіма її частинами та можливості прийняття рішень залежно від зміни стану будь-якої КС мережі. У цьому контексті децентралізованість і самостійність прийняття рішень модулем системи, розміщеним у певній КС, не ототожнюються. У поняття децентралізованості системи закладемо функцію вищого рівня абстракції, завданням якої повинна бути здатність системи конкретної КС приймати рішення про початок її виконання (тобто активацію закладених функціоналів), здійснення переходу між визначеними рівнями на основі інформації, отриманої з інших рівнів системи, прийняття рішення про комунікацію з іншими частинами всієї системи, отримання інформації від інших частин системи з різних КС та передання цієї інформації на відповідні рівні завершення роботи. Для частини системи, розміщеної в КС, уведемо поняття програмного модуля (далі – ПМ) системи. Самостійність у прийнятті рішення програмним модулем системи містить такі можливості: прийняття рішення про стан загрози ШПЗ для КС на основі інтегрованої інформації з інших рівнів програмного модуля і передання цього значення на рівень децентралізації; передання на рівень децентралізації значення рівня безпеки; визначення кількості та залучення засобів відповідних рівнів програмного модуля для дослідження ШПЗ. Таким чином, рівень децентралізації відрізнятиметься від рівня прийняття рішень тим, що на його рівні приймаються рішення про загальну організацію функціонування програмного модуля в структурі всієї системи, а на рівні прийняття рішень – про безпосереднє виконання саме основних завдань дослідження ШПЗ.

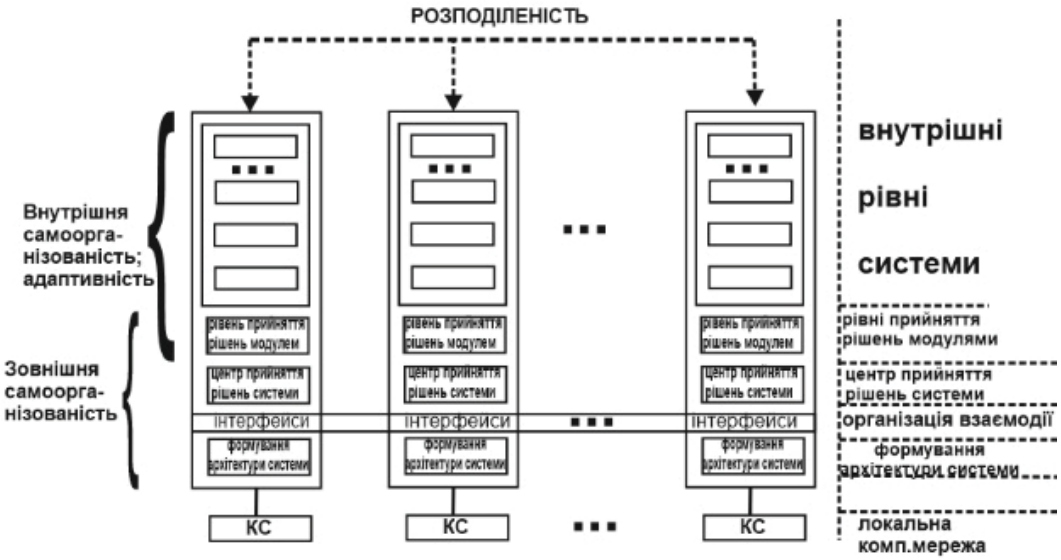

Рис. 1. Узагальнена схема основних складових архітектури системи

Багаторівневість (рис. 2) системи дозволить виділити процеси, які стосуватимуться функціонування системи загалом, процеси функціонування програмного модуля і розділити з рівнями завдання з виявлення різних типів ШПЗ та мережевих атак і необхідних утиліт для вирішення цих завдань. Крім того, багаторівневість дозволить проводити нарощування системи новим функціоналом, а також чітка відокремленість різних за завданнями частин дозволить сумісно використовувати утиліти певних рівнів.

Функціонал, який відповідатиме за самоорганізованість програмного модуля РБС, розміщено окремим рівнем програмного модуля, який взаємодіятиме з модулем прийняття рішень. Крім того, певні рівні програмних модулів матимуть можливість здійснювати самонавчання та на його основі здійснюватиметься зміна стану безпеки. Самоорганізованість на рівні всієї РБС здійснюватиметься в підблоці блоку децентралізації і міститиме функціонування всієї системи та її переходу в різні стани (залежно від зовнішніх змін у КС та мережі). Властивість самоорганізованості РБС тісно пов'язана з адаптивністю і проявлятиметься в зміні структури її системи та рівня організації в процесі свого життєвого циклу в результаті накопичення досвіду, збереженої інформації в пам'яті. Накопичений досвід виражатиметься в зміні параметрів, важливих для мети системи, що змінюватиме спосіб функціонування системи і виражатиме властивість самонавчання.

Формування архітектури РБС у мережі здійснюватиметься протягом життєвого циклу її використання та передбачатиме зберігання попередніх форматів для здійснення аналізу.

Частина рівнів програмних модулів РБС матиме властивості адаптивності для підвищення ефективності виконання поставлених завдань виявлення ШПЗ. Адаптивність РБС проявлятиметься в автоматичній зміні алгоритмів свого функціонування і своєї структури з метою збереження або досягнення оптимального стану у разі зміни зовнішніх умов.

Інтерфейси системи поділено на такі: адміністративний, щоденний звіт, звіт за критичної ситуації, автономний, міжмодульний.

Розподілена багаторівнева система в локальній комп'ютерній мережі представлена однаковими програмними модулями, які розміщені в кожній комп'ютерній системі. Тобто незалежно від того, скільки комп'ютерних систем у локальній мережі, кожна з них містить програмний модуль РБС. Якщо в процесі користування комп'ютерними системами виявиться, що не всі вони в мережі ввімкнені, тоді РБС складається з тих, які активні. Програмні модулі активних КС у локальній мережі формують безпосередньо РБС. Це дозволить навіть за наявності роботи двох комп'ютерних систем підтримувати виконання завдань системи. Кожен програмний модуль РБС містить функціонал для виявлення певного типу шкідливого програмного забезпечення або атаки.

РБС містить у кожному програмному модулі рівень, який відповідає за спілкування між програмними модулями всієї системи, тобто забезпечує роботу каналів зв'язку. За допомогою цього рівня встановлюється зв'язок між програмними модулями системи в цілому. Важливою функцією роботи системи є ідентифікація одним конкретним програмним модулем решти програмних модулів для активації та роботи цілісної системи. Для цього активується функціонал, що здійснює зчитування системної інформації про апаратно-програмне забезпечення КС, зберігає цю інформацію у своєму внутрішньому сховищі, формує на її основі ідентифікаційну ознаку цього конкретного модуля, зберігає її і використовує в пакетах, які розсилатимуться на інші програмні модулі РБС. Під час встановлення всіх програмних модулів РБС у мережі здійснюють їх активацію з метою збору всіх ідентифікаційних ознак. Кожен окремий програмний модуль у системі міститиме характеристики та ідентифікатори всіх програмних модулів системи.

Кількість рівнів програмного модуля та їх наповнення залежатимуть від завдань, які вирішуватимуться. Основними завданнями системи є проведення аналізу виконуваних файлів

Функції 1-го рівня	Функції 1-го рівня	■ ■ ■	Функції 1-го рівня
Функції 2-го рівня	Функції 2-го рівня	■ ■ ■	Функції 2-го рівня
■ ■ ■	■ ■ ■	■ ■ ■	■ ■ ■
Функції m-го рівня	Функції m-го рівня	■ ■ ■	Функції m-го рівня
Прийняття рішень модулем 1	Прийняття рішень модулем 2	■ ■ ■	Прийняття рішень модулем m
Центри прийняття рішень системи			
Модуль 1	Модуль 2	■ ■ ■	Модуль n
Організація взаємодії з використанням протоколів			
Модуль 1	Модуль 2	■ ■ ■	Модуль n
Формування архітектури системи			
Модуль 1	Модуль 2	■ ■ ■	Модуль n

Рис. 2. Схема багаторівневості РБС

із метою виявлення прояву шкідливого функціоналу й поведінки виконуваного в КС програмного забезпечення (запущених процесів) на наявність зловмисних дій. Обидва ці завдання об'єднує необхідність проведення аналізу поведінки програмного забезпечення, тобто здійснення виявлення через поведінку. Це дозволить увібрати для розгляду і дослідження достатньо широку множину всього файлового ШПЗ, відділивши в її складовій частині зловмисну поведінку для аналізу. Тобто система міститиме засоби дослідження програмного забезпечення, розміщеного в КС, яке перебуватиме в активному (виконуваному) стані або в зовнішній пам'яті, спільним елементом яких буде відслідковування й аналіз їхньої поведінки.

Ураховуючи те, що в результаті виконання функціоналів підсистем накопичуватимуться великі обсяги різнорідної інформації, то для ефективної роботи РБС потрібні підсистеми програмних модулів для обліку роботи конкретних КС та здійснення оптимізації масивів інформації в системі загалом.

Розподіл завдань у програмних модулях системи міститиме стани, у яких перебуватиме програмний модуль конкретної КС у процесі свого життєвого циклу. Модель РБС, що враховуватиме архітектурні складники і їх стани та зв'язки між ними представимо так:

$$M_A^S = (S, G_A), \qquad (1)$$

де S – множина станів системи, G_A – орієнтований граф узагальнених станів РБС, який зображено на рис. 3.

Відповідно до рис. 3, для кожного стану визначено такі функції:

1 – базовий стан, моніторинг КС, визначення переходу до станів 2, 3, 4, 8;

2 – перевірка виконуваних файлів, визначення переходу до станів 1, 5;

3 – перевірка запущених процесів та мережних активностей КС, визначення переходу до станів 1 та 6;

4 – перевірка виконуваних файлів та запущених процесів, визначення переходу до станів 1 та 7;

5 – перевірка виконуваних файлів із використанням інших програмних модулів, повернення на рівень 2;

6 – перевірка запущених процесів та мережних активностей і їх порівняння з іншими КС мережі, повернення на рівень 4;

7 – оброблення виконуваних файлів та запущених процесів із використанням інших програмних модулів системи;

8 – оброблення та оптимізація інформації з бази пакетів програмного модуля із залученням інформації з інших програмних модулів системи, повернення на рівень 1;

9 – забезпечення формування зв'язку з іншими програмними модулями системи.

Кожен програмний модуль системи має однакову структуру і розділений на чотири рівні залежно від призначення та згрупованих у них завдань:

1) рівень 1 містить моніторинг подій та визначення переходів до наступних рівнів, а також здійснює оброблення інформації, що надходить від інших програмних модулів;

2) рівень 2 містить перевірку виконуваних файлів, перевірку запущених процесів та мережевих активностей без залучення інформації з інших програмних модулів системи;

3) рівень 3 містить виконання завдань рівня 2 із залученням інформації з інших програмних модулів системи;

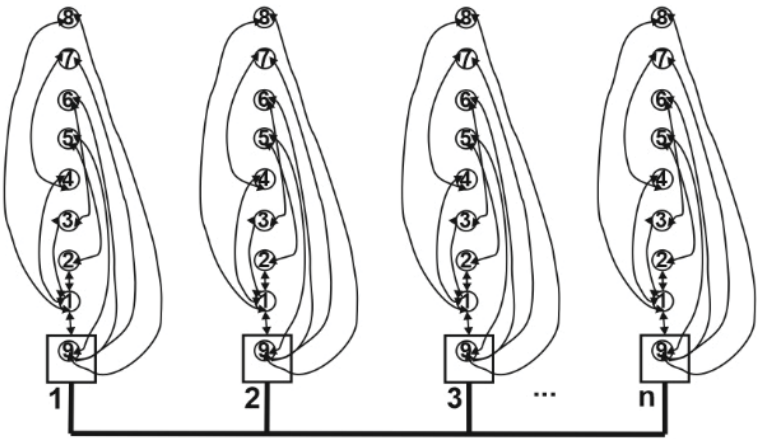

Рис. 3. Орієнтований граф зв'язків узагальнених станів програмного модуля системи

4) рівень 4 здійснює оброблення, оптимізацію та вилучення інформації з бази програмного модуля КС.

Рівні 1, 3 та 4 обов'язково співпрацюють з іншими програмними модулями системи. Програмний модуль, як правило, перебуває в рівні 1, де здійснює моніторинг подій. Якщо відбуваються зміни в КС, тоді на його рівні приймаються рішення про перехід до вершин 2, 3 або 4, тобто на рівень 2. На рівні 2 досліджуються завдання з рівня 1 методами та засобами без залучення інших частин системи. Якщо результат дослідження виявиться негативним, тобто ШПЗ не буде знайдено, тоді встановлюється потреба у глибшій перевірці і здійснюється перехід до рівня 3, де для визначення застосовуються інші частини системи в мережі. Використання засобів цього рівня залучає розподілені в мережі програмні модулі, чим підвищуватиме ефективність виявлення ШПЗ. На четвертому рівні відбувається оптимізація інформації з бази програмного модуля із залученням інших частин системи, а також прийняття рішення про стан системи в цілому (після отримання інформації зі всіх інших програмних модулів системи).

Таким чином, перший рівень містить засоби, які забезпечують автономність роботи програмного модуля системи. На третьому рівні досягається глибший аналіз досліджуваних об'єктів, порівнюючи з другим рівнем, що підвищує ефективність системи. Четвертий рівень вирішує частину завдань із самоорганізації системи, пов'язаної з оптимізацією накопиченої за час роботи інформації. Кожен рівень та відповідні йому узагальнені підсистеми представляються наборами підрівнів, яким закладено виконання певних функціоналів.

Зображення взаємозв'язку підрівнів представлено на рис. 4 графом із вершинами, що відповідають призначенням підрівнів.

Часова модель поведінки РБС через моделі поведінки її ПМ розглядається не як лінійна послідовність множини обчислень, а як дерево можливих обчислень, тобто розгалужена часова модель із зворотніми зв'язками. Тому представимо РБС моделлю на основі структури Кріпке:

$$M_{РБС} = (S, S_0, R, F_{РБС}), \qquad (2)$$

де S – скінченна множина станів РБС, S_0 – множина початкових станів (причому їх може бути декілька), R – множина переходів між станами, T_A – множина атомарних тверджень пов'язаних із станами і є істинними тільки в цих станах, $F_{РБС}$ – функція, що відображає кожен стан із множини S в підмножину атомів T_A, які є істинними у відображуваних станах. Оскільки система є розподіленою, тоді множину станів системи представимо через підмножини станів, які стосуються програмних модулів A_i, i=1,2,...,n, зокрема $S - U^n_{i=1}S_i$, тобто множини станів ПМ формуватимуть множину станів, у якій перебуватиме система. Множина початкових станів РБС $So - U^n_{i=1}S_{0i}$, причому серед їх елементів не може бути однакових. Із будь-якого стану $s \in S$ існує мінімум один перехід до іншого стану, тобто справедливим є відношення $R \subseteq S \times S$, що означає те, що для будь-якого з множини станів існує відповідний йому стан із цієї ж множини. Якщо множина атомарних тверджень T_A скінчена, тоді множина її підмножин 2^{T_A} буде відображенням $F_{РБС}$ для множини S у ті підмножини атомів, які будуть істинними в $s \in S$.

Представимо структурну частину РБС програмний модуль підмоделлю:

$$M_{A_i} = \langle\ S_i, S_{0i}, R_i, F_{A_i}\ \rangle, \qquad (3)$$

де A_i– це *i*-тий програмний модуль РБС, i=1,2,...,n, R_i – множина переходів між станами $s_{ij} \in s_i$, j – кількість і – тих станів, F_{A_i} – функція відображення множини станів S_i в множину підмножин

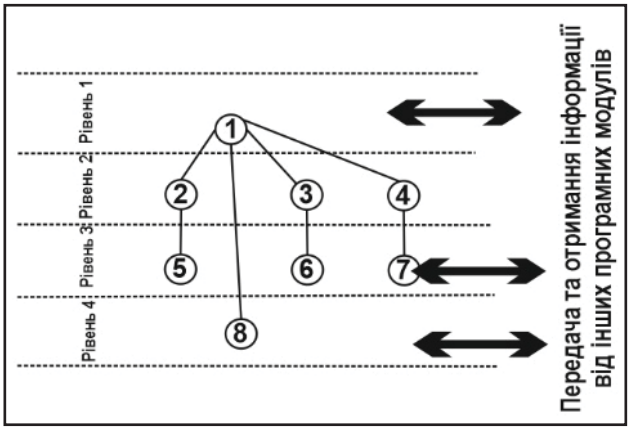

Рис. 4. Граф-схема взаємодії підрівнів програмних модулів системи

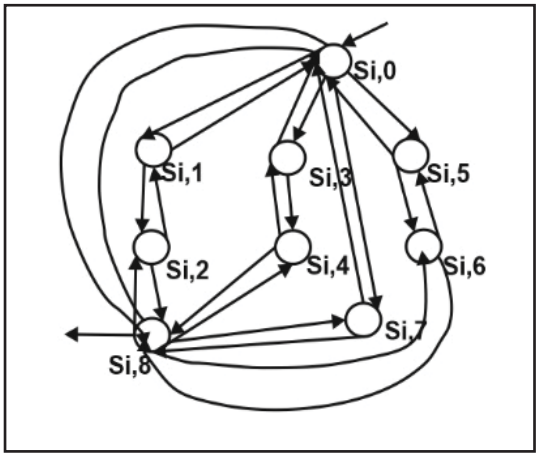

Рис. 5. Структура Кріпке програмного модуля A_i

множини T_{A_i}, тобто $2^{T_{A_i}}$. Кожен стан обов'язково має зв'язок із деякими іншими станами. Перехід з одного стану до іншого, якщо між ними є зв'язок, відобразимо послідовністю $s_{ij}s_{i\rho}$, де i – номер програмного модуля, j та ρ – стани цього ж модуля. Тоді послідовності $s_{ij}s_{ip}s_{ij}s_{ih}s_{iy}s_{iu}s_{ie}s_{ik}\ldots$ означатимуть переходи ПМ з одного стану в інший протягом його експлуатації. РБС у процесі експлуатації характеризуватиметься множиною послідовностей переходів із стану в стан програмних модулів. Структуру Кріпке для ПМ за його діаграмою переходів зображено на рис. 5.

R_i – множина переходів між станами i-того модуля на множинах змінних, що визначається так (див. табл. 1):

Функцію F_{A_i} відображення множини станів S_i у множину підмножин множини T_{A1} задамо таблично. Фрагмент представлено в табл. 2.

Самовідтворення РБС здійснюється на початку роботи кожного з програмних модулів, додаванням кожного нового програмного модуля, видаленням певного програмного модуля. Ці події зберігають систему і дозволяють вирішувати поставлені на неї завдання. Цей рівень розгляду РБС представляє цю властивість як таку, що стосується існування системи. Описані події з організації спілкування між ПМ показують принцип утворення РБС та її самовідтворення в процесі життєвого циклу її частин. Мінімальна кількість працюючих КС, на яких встановлено програмні модулі (не менше двох). Але використання дуже малої кількості ПМ системи не є ефективним, бо тоді для підвищення достовірності виявлення ШПЗ недостатньою є кількість інформації від різних КС локальної мережі. Наявність тільки одного ПМ у складі РБС, тобто однієї ввімкненої КС, не дозволить використати можливості переходу в інші стани з метою детальнішого дослідження поведінки ШПЗ, а використовуватиме лише можливості на рівні однокористувацьких антивірусних програмних засобів. Самовідтворення РБС на рівні її структурних частин відбуватиметься шляхом переведення ПМ, які тривалий час перебувають в одному зі станів, до базового стану. Також РБС здійснюватиме аналіз статистики роботи ПМ КС для оптимізації передавання завдань із метою залучення до їх вирішення найбільш активних КС.

Програмні модулі РБС (у разі перебування на третьому і четвертому рівнях для вирішення поставлених завдань) залучатимуть технології самонавчання, що впливатиме на зміну їх роботи під час вирішення таких же завдань на наступних етапах життєвого циклу; при цьому вони, передаючи завдання для оброблення на інші ПМ, та результати їх виконання впливатимуть на інші програмні модулі; вплив ПМ, як структурних частин системи, на інші ПМ дозволить еволюціонувати РБС.

РБС на рівні її структурних частин ПМ здійснюватиме самоконтроль, що проявлятиметься в періодичній перевірці комплектності системи, здійсненні аналізу наявності ПМ, які тривалий час перебувають в одному й тому ж стані та потребують автоматичного зняття поточних завдань із виконання та переведення до іншого стану, оброблення баз ПМ для оптимізації та розподіл ПМ на декілька груп згідно з аналізом їх станів протягом тривалого проміжку часу.

Важливим елементом самоорганізації РБС є розроблення механізмів утворення своїх власних цілей. До таких цілей належать: динамічне формування системи; розподіл та співвіднесення всіх структурних частин за групами завантаженості,

Таблиця 1

R_i = {	0000i	0001i,	0001i	0000i,	0001i	0100i,	0100i	0001i,	
	0100i	1000i,	1000i	0100i,	0000i	1000i,	1000i	0000i,	
	0000i	0010i,	0010i	0000i,	0010i	0101i,	0101i	0010i,	
	0101i	1000i,	1000i	0101i,	0000i	0011i,	0011i	0000i,	
	0011i	0110i,	0110i	0011i,	1000i	0110i,	0110i	1000i,	
	0000i	0111i,	0111i	0000i,	1000i	0111i,	0111i	1000i	}

Таблиця 2

Кодований перехід зі стану в стан

Властивість	Кодований перехід	Стан
$p_{i,0}$	0000i	$s_{i,0}$
$p_{i,1}$	0000i, 0001i	$s_{i,1}$
…	…	…
$p_{i,47}$	0111i, 1000i	$s_{i,47}$

оброблення критичних подій у системі, колективне виконання завдань, розв'язуване одним ПМ, оброблення й оптимізація накопичених статистичних даних.

Основою побудованої моделі РБС є її структурні частини, які представляються програмними модулями, що можуть перебувати в різних станах. Перехід між станами ПМ здійснюється на основі визначеної множини переходів. Взаємодія та спілкування між ПМ базується на основі їх перебування в певних станах під час експлуатації. РБС є реагуючою системою, яка здійснюватиме моніторинг визначених подій. Кожен програмний модуль містить резидентний механізм, рушійні механізми для переходу між станами, переходи, між якими задаються підмножинами переходів, дані для яких формуватимуться з використанням технологій штучного інтелекту.

Зображення віконних форм програмної реалізації РБС зображено на рис. 6.

Приклад використання розподіленої багаторівневої системи [7]**.** Прикладом застосування РБС розглянемо підсистему виявлення метаморфних вірусів у локальній комп'ютерній мережі. Використання мережі продиктоване наявністю, окрім обфускаційних технік, антиемуляційних засобів, що перешкоджають здійсненню процесу емуляції виконання. Використання емулятора є одним із головних методів виявлення метаморфних вірусів, що призводить до низької ефективності виявлення. Тому здійснення виявлення метаморфних вірусів із високою ефективністю, які застосовують антиемуляційні технології, засобами однієї КС є неможливим, тому пропонується залучення локальної комп'ютерної мережі.

Із метою виявлення підозрілих дій на кожному хості застосовується аналізатор підозрілості програми. Основним його завданням є відстеження потоку API викликів, що здійснюються в процесі виконання невідомою програмою. У разі застосування до програми технік обфускації програмного коду, її API виклики залишаються незмінними, видозмінюються лише параметри та значення, що повертається відповідною функцією [7]. Кожна окрема підозріла дія, яка представлена викликом API функції, під час виконання невідомої програми, не є небезпечною. Проте виконання певної послідовності таких дій може свідчити про можливу небезпеку інфікування шкідливим ПЗ, зокрема метаморфним вірусом.

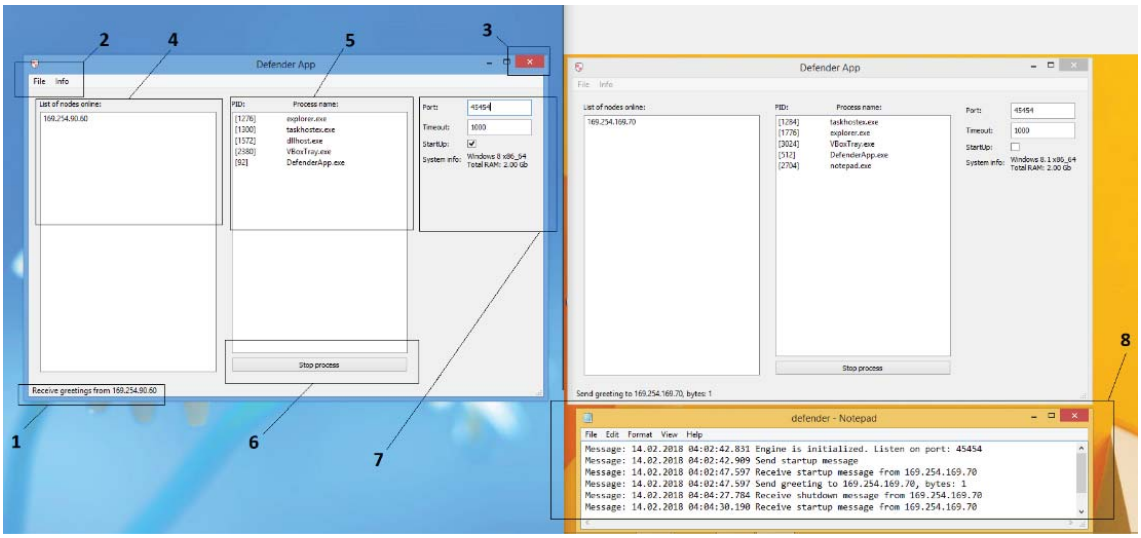

Рис. 6. Інтерфейсні вікна розробленої РБС складаються з таких компонентів: 1) видання інформативних повідомлень; 2) головне меню. File/Quit – завершити роботу, Info/About – показати додаткову інформацію (назва, короткий опис, автор, версія); 3) у разі закритті головного вікна, програма згортається у system tray та продовжує свою роботу. Для повного завершення роботи необхідно скористатися File/Quit, або ПКМ на піктограмі програми у системному вікні та Quit; 4) список IP-адрес машин, на яких також запущено екземпляр програми. Оновлюється автоматично при запуску/завершені програми на інших машинах; 5) список процесів у системі, запущених у КС. Оновлюється згідно із заданим інтервалом таймера; 6) команда «зупинити вибраний зі списку процес»; 7) панель налаштувань. Port – номер порту для комунікації між екземплярами програм по мережі, Timeout – інтервал таймеру для моніторингу системних процесів, StartUp – вкл./викл. автозавантаження програми при запуску ОС, System info – характеристики системи; 8) лог файл. Створюється у директорії з виконуваним модулем. Колекціонує повідомлення/попередження/помилки роботи програми.

Виявлення здійснюється на основі відстеження API викликів, що описують потенційно небезпечну поведінку метаморфного вірусу, та порівнянні дизасембльованого коду функціональних блоків метаморфного вірусу з кодом функціональних блоків його зміненої версії. Для створення зміненої версії метаморфного вірусу на хостах мережі встановлюються модифіковані емулятори, що забезпечують змінне середовище виконання. Із метою підвищення загальної ефективності виявлення метаморфних вірусів система передбачає пошук відповідності між функціональними блоками метаморфного вірусу та його зміненої версії. Для формування висновку про схожість підозрілої програми на метаморфних вірусах використовується система нечіткого логічного висновку. У разі недостатнього прояву шкідливої поведінки та підвищення рівня достовірності для виявлення метаморфного вірусу залучаються інші хости мережі. Детальніше про реалізований метод у [8].

Висновки. Архітектура розподіленої багаторівневої системи базується на принципах децентралізації та самоорганізації і дозволяє здійснювати її наповнення різними функціоналами виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах. РБС належить до реагуючих систем, яка постійно здійснюватиме моніторинг запущених процесів та виконуваних програм у комп'ютерних системах мережі. Об'єктами для дослідження з боку РБС є перевірка наявного програмного забезпечення та запущених процесів у КС локальної мережі на можливість належності до шкідливого програмного забезпечення.

Розроблена модель архітектури програмних модулів РБС базується на принципах автономності та багаторівневості. Вона дозволяє здійснювати збільшення кількості рівнів системи без зміни її архітектури. Основою архітектури РБС виступають програмні модулі з однаковими архітектурами, проте кожен із них може самостійно ухвалювати рішення на основі різних даних зібраних із різних КС мереж. Для ефективної роботи РБС необхідним є розроблення методів і моделей взаємодії та узгодження роботи різних програмних модулів між собою та їх відповідних рівнів і деталізація структури її станів.

Список літератури:

1. Virus Bulletin URL: https://www.virusbulletin.com/testing/ (дата звернення: 25.03.2018).

2. High performance adaptive system for cyber attacks detection / Komar M. et al. The 9-th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications: Proceedings (Bucharest, 21-23 September 2017). Bucharest, 2017. P. 853–858.

3. Golovko V., Bezobrazov S. Neural Network Artificial Immune System for Malicious Code Detection. Brest State Technical University. 2015, P. 1–7.

4. Branitskiy A., Kotenko I. Hybridization of computational intelligence methods for attack detection in computer networks. Journal of Computational Science. 2017. № 23 P. 145–156.

5. Wang G., Hao J.,Ma J., Huang L. A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering. Expert Systems with Applications: An International Journal. 2010. Vol. 37. Issue 9. P. 6225–6232.

6. Host-based Web Anomaly Intrusion Detection System, an Artificial Immune System Approach / Khalkhali I., Azmi R., Azimpour-kivi M., Khansari M. International Journal of Computer Science. 2011. Vol. 8. Issue 5. № 2. P. 14–24.

7. Approach for the Unknown Metamorphic Virus Detection / Savenko O., Lysenko S., Nicheporuk A., Savenko B. The 9-th IEEEE. International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications: Proceedings (Bucharest, 21-23 September, 2017). Bucharest, 2017. P. 453–458.

8. Спосіб виявлення метаморфних вірусів на основі статистичних метрик для визначення еквівалентних функціональних програмних блоків. Патент України на корисну модель МПК G06F 21/55 / Савенко О.С. та ін. № 118456; заявл. 23.02.2017; опубл. 10.08.2017, Бюл. № 15/2017.

АРХИТЕКТУРА РАСПРЕДЕЛЕННОЙ МНОГОУРОВНЕВОЙ СИСТЕМЫ ОБНАРУЖЕНИЯ ВРЕДОНОСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В ЛОКАЛЬНОЙ КОМПЬЮТЕРНОЙ СЕТИ

В статье предложена архитектура распределенной многоуровневой системы обнаружения вредоносного программного обеспечения в локальных компьютерных сетях, особенностью которой является синтез требований распределенности, децентрализованости, многоуровневости, и самоорганизованности, что, в отличие от известных систем, позволяет ее использовать автономно для выполнения заложенных функций обнаружения вредоносного программного обеспечения. Кроме того, программные модули системы представлены на основе структуры Крипке, особенностью которой является такая самоорганизация, что позволяет осуществлять обмен знаниями внутри системы, что, в отличие от известных систем, позволяет использовать знания, полученные отдельными частями системы в других частях.

Ключевые слова: *архитектура, распределенная система, вредоносное программное обеспечение, метаморфный вирусы, структуры Крипке.*

ARCHITECTURE OF DISTRIBUTED AND MULTILEVEL SYSTEM FOR DETECTION MALICIOUS SOFTWARE IN LOCAL COMPUTER NETWORKS

The paper proposes the architecture of a distributed multilevel system for detecting malware in local computer networks, the feature of which is the synthesis of distribution, decentralization, multilevel, and self-organization requirements in it, which, unlike the known systems, allows it to be used autonomously for the implementation of its functions detection of malware. In addition, the program modules of the system are represented on the basis of the structure of the Kripke, the feature of which is the same organization, which allows the exchange of knowledge in the middle of the system, which, unlike the known systems, allows you to use the knowledge obtained by separate parts of the system in other parts.

The architecture of a distributed multilevel system is based on the principles of decentralization and self-organization and allows it to be filled with various functions of detection of malicious software in local computer networks. The distributed multi-level system relates to responsive systems, which will continuously monitor the running processes and executable programs in computer systems of the network. Objects for research from a distributed multilevel system are the testing of existing software and running processes in computer systems of the local network to the ability to refer to malicious software. It allows you to increase the number of levels of the system without changing its architecture. The basis of the architecture of the system are program modules with the same architecture, but each of them can independently take decisions based on various data collected from different computer systems of the network.

The distributed multilevel system at the level of its structural parts of the software modules will carry out self-monitoring, which will be manifested in the periodic verification of the completeness of the system, the analysis of the availability of software modules, which for a long time are in the same state and require the automatic removal of current tasks for execution and transfer to another state , processing of software modules for optimization and distribution of software modules into several groups according to the analysis of their states over a long period of time.

An important element of self-organization of the system is the development of mechanisms in it for the formation of its own goals. For such purposes we shall include the following: dynamic formation of the system; distribution and correlation of all structural units by groups of load, processing of critical events in the system, collective execution of tasks solved by one software modules, processing and optimization of accumulated statistical data.

Key words: *architecture, distributed system, malicious software, metamorphic viruses, Kripke structures.*