УДК 004.491.42
Савенко О.С., к.т.н., Грибинчук В.І., Кульчицький М.О.
Хмельницький національний університет

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

Савенко О.С., Грибинчук В.І., Кульчицький М.О. Архітектура багаторівневої програмної системи виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах. У статті запропоновано архітектуру розподіленої багаторівневої програмної системи виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах, особливістю якої є синтез в ній вимог розподіленості, децентралізованост та багаторівневості, що дозволяє її використання автономно. Крім того, особливістю автономних програмних модулів системи є така самоорганізація, що дає змогу здійснювати обмін знаннями в середині системи, яка на відміну від відомих систем дозволяє використовувати знання отримані окремими частинами системи в інших частинах.
Ключові слова: архітектура, розподілена система, шкідливе програмне забезпечення, локальні комп'ютерні мережі.

Савенко О.С., Грыбынчук В.И., Кульчицкий М.А. Архитектура многоуровневой программной системы обнаружения вредоносного программного обеспечения в локальных компьютерных сетях. В статье предложена архитектура распределенной многоуровневой программной системы обнаружения вредоносного программного обеспечения в локальных компьютерных сетях, особенностью которой является синтез в ней требований распределенности, децентрализованости и многоуровневости, что позволяет ее использование автономно. Кроме того, особенностью автономных программных модулей системы есть такая самоорганизация, что позволяет осуществлять обмен знаниями внутри системы, которая в отличие от известных систем позволяет использовать знания, полученные отдельными частями системы в других частях.
Ключевые слова: архитектура, распределенная система, вредоносное программное обеспечение, локальные компьютерные сети.

Savenko O.S., PhD, Gribinchuk V.I., Kulchytsky M.O. The architecture of the multi-level software system for detecting malware in local computer networks. The article proposes the architecture of distributed multilevel software system for detection of malicious software in local computer networks, the feature of which is the synthesis of its requirements for distribution, decentralization and multilevel, allowing its use autonomously. In addition, the feature of autonomous program modules of the system 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 gained by separate parts of the system in other parts.
Keywords: architecture, distributed system, malicious software, local computer networks.

Постановка проблеми. На сьогодні шкідливе програмне забезпечення (ШПЗ) 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 (наприклад, ботнет). Для організації ефективної протидії таким засобам важливим є розробка таких систем виявлення ШПЗ, архітектура яких враховувала б ці особливості. Крім того, досягнення підвищення достовірності виявлення ШПЗ в межах тільки однієї комп'ютерної системи (КС), яка має вихід в мережу Internet, може бути недостатнім при протидії засобам ШПЗ, які представлені великим програмним комплексом, що розміщений в багатьох комп'ютерних системах в глобальній мережі, і частини якого комунікують між собою.

Аналіз та тестування відомих програмних засобів виявлення нового шкідливого програмного забезпечення, відомості про яке не містяться в антивірусних засобах, вказують на недостатньо високий рівень достовірності виявлення [1]. Реалізація нових методів виявлення конкретних типів шкідливого програмного забезпечення шляхом створення відповідних інформаційних систем потребує подальшого розвитку з метою підвищення достовірності виявлення ШПЗ.

Аналіз досліджень і публікацій. Проблемі виявлення ШПЗ сьогодні приділяється значна увага. Для виявлення та протидії ШПЗ розроблено ряд систем та методів.

В роботі [2] запропоновано систему виявлення кібератак на основі залучення нейромережних імунних детекторів. Розроблена система складається з двох частин. Перша 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 вплив

ШПЗ здійснюється із залученням системи нейромережних детекторів, в основу якої закладено алгоритм Мамдані.

В 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 та 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 комбінація різних методів штучного машинного навчання, а саме нейронних мереж, імунної системи, нейрофізичних класифікаторів та метод опорних векторів. Відмінною особливістю запропонованої системи є багаторівневий аналіз мережевого трафіку, 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 з використанням методів нечіткої кластеризації. Для отримання остаточних результатів, визначається оцінка для кожного елементу з сформованих підгруп та виконується їх поєднання з використанням штучної імунної системи.

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

Для того, щоб ефективно застосовувати методи та засоби виявлення шкідливого програмного забезпечення необхідно розробити систему, яка б включала в себе 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.

Виклад основного матеріалу дослідження. Виявлення шкідливого програмного забезпечення, що здійснюється за допомогою різноманітних засобів, суттєво залежать, зокрема, і від архітектури таких засобів, а також їх позиціонування та місця розміщення в комп'ютерних системах локальних мереж. Враховуючи, що процес виявлення ШПЗ проводитиметься в локальних мережах, то вибір моделі функціонування системи повинен передбачати залучення інформації зі всіх комп'ютерних систем локальної мережі, тобто розміщення на всіх КС цієї системи. Це необхідно для підвищення ефективності і 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 міграція центру прийняття рішень системи в локальній мережі, оскільки атака на нього призведе до виведення всієї системи з робочого стану. Система повинна бути побудована так, щоб розміщені в КС локальної мережі її частини ефективно взаємодіяли між собою для обміну інформацією про стан КС з метою надання додаткової інформації для прийняття рішення. Крім того, система виявлення ШПЗ повинна відповідним чином структуруватись, щоб мати можливість нарощення, причому її збільшення не повинно сповільнювати процес виявлення. Основними функціями такої розподіленої багаторівневої програмної системи (РБПС) є перевірка наявного програмного забезпечення та запущених процесів в КС локальної мережі на можливість віднесення до шкідливого програмного забезпечення. Досягнення відповідності системи заданим характеристикам та покладеним на неї функціям з виявлення ШПЗ формують вимоги до неї, основними з яких будуть такі: розподіленість, децентралізованість, самостійність у прийнятті рішень, багаторівневість, само організованість, адаптивність. Врахування таких складових в моделі системи є основою її архітектури і підвищить надійність функціонування та живучість системи в локальній мережі. Беручи до уваги умову, що виявлення шкідливого програмного забезпечення відбуватиметься у локальній мережі, що складається з комп'ютерних систем, представимо схему структурної розподіленості архітектури системи, яка відображена на рис.1.

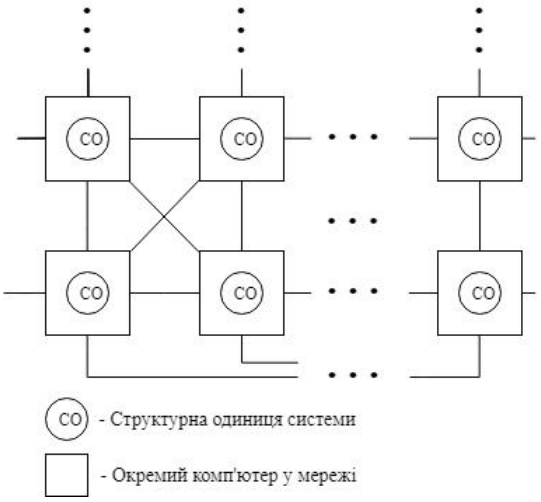

Рис.1 Схема структурної розподіленості архітектури

Кожна одиниця системи на окремих комп'ютерах в мережі є автономною та повноцінною, з'єднаною з кожною іншою. Таким чином жодна з них не виступає у ролі центрального керівного пункту прийняття рішень. Система здатна розширюватися підсистемами, що можуть бути приєднані до основної архітектури. Оскільки структура системи є просторово розподіленою, а кожна одиниця системи здатна приймати самостійні рішення, тому для позначення кожної частини системи будемо використовувати поняття автономного програмного модуля (АПМ) системи.Враховуючи децентралізованість та автономність системи, механізм взаємодії окремих АПМ базується на наступних принципах: АПМ виконуються заплановані завдання без впливу ззовні, кожен АПМ може обмінюватися з кожним іншим АПМ, система здійснює самоконтроль та планує власні цілі.

Дотримання цих принципів при реалізації механізму взаємодії окремих модулів значно впливатиме на ефективність досягнення поставлених перед системою цілей.

Згідно зображеної вище схеми структурної розподіленості системи у мережі, вона представлена однаковими одиницями, що знаходяться на кожному окремому комп'ютері у мережі. Запущені АПМ в мережі формують розподілену систему. За такої схеми система здатна виконувати свої задачі, навіть якщо активними під час її роботи є лише декілька комп'ютерів із встановленими на них АПМ. Тобто система складається з тих АПМ, які є активними.

Децентралізованість архітектури представлено на узагальненій структурній схемі складових системи на рис.2.

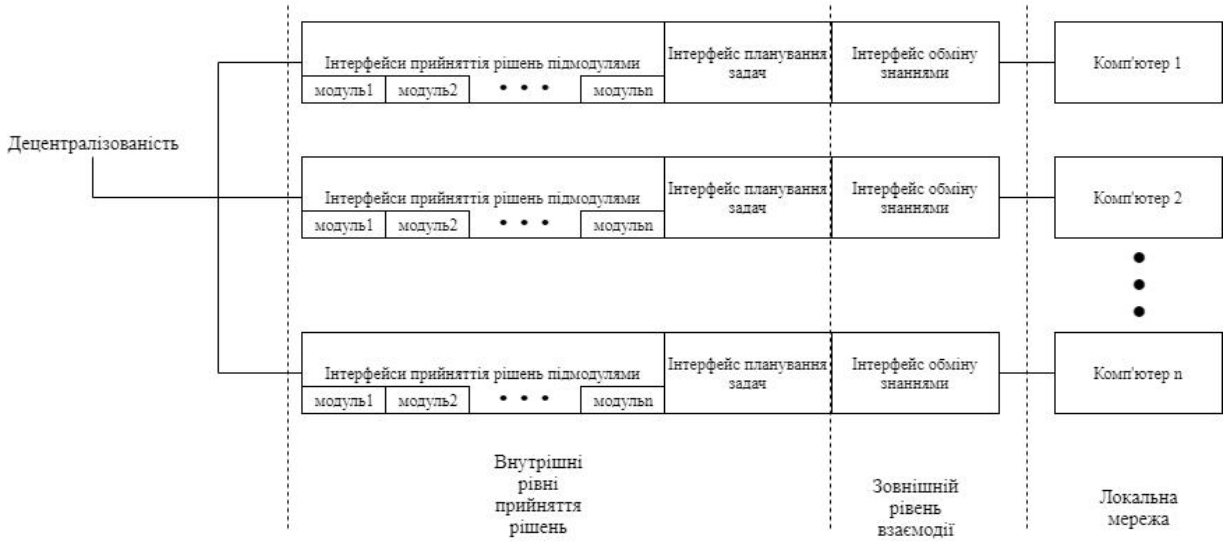

Рис. 2. Узагальнена схема складових системи

Узагальнена схема складових системи відображає реалізовану ідеологію децентралізованості та розподіленості у просторі. Прийняття рішень не здійснюється єдиним центром, проте виконується на внутрішніх рівнях через інтерфейси прийняття рішень. Кожна окрема одиниця є автономною у прийнятті рішень.

Автономність кожної одиниці на внутрішньому рівні включає здатність прийняття рішення про виявлення спроби атаки шкідливим програмним забезпеченням та передачу повідомлення на зовнішній рівень.

На зовнішньому рівні здійснюється обмін знаннями з іншими одиницями, згідно чого система здатна реагувати на спроби атаки, накопичувати «досвід» та адаптуватися до середовища, в якому вона працює, що забезпечує гнучкість системи до зміни обставин.

Таким чином система, в цілому, володіє властивістю адаптуватися для підвищення ефективності виявлення та реагування на атаки шкідливого програмного забезпечення. Ця властивість включає в себе гнучку зміну функціонування підмодулів для досягнення оптимальної роботи в середовищі.

Кожний АПМ у розподіленій системі ідентифікується по системним характеристикам комп'ютерної системи, на яку він встановлений та її IP-адреси у мережі. З цією метою при першому запуску АПМ на комп'ютері виконується сканування інформації про апаратні характеристики системи.

Абстракція децентралізованості включає в себе здатність кожної одиниці системи на окремому комп'ютері запустити певний функціонал виявлення шкідливого програмного забезпечення, виконати перехід між різними рівнями на основі інформації отриманої з іншого рівня, провести обмін повідомленнями з іншими одиницями всієї системи, отримати знання від інших частин та передати їх на інтерфейси прийняття рішень.

Таким чином, на внутрішньому рівні прийняття рішень відбуваються основні задачі виявлення шкідливого програмного забезпечення, в той час як на зовнішньому рівні відбувається обмін результатами та прийняття більш глобальних рішень стосовно загальної організації роботи системи у мережі. Зовнішній рівень є каналом взаємодії та відповідає за спілкування між активними АПМ системи. За його допомогою є можливою комунікація між окремими АПМ розподіленої системи.

Взаємодія внутрішнього рівня прийняття рішень та зовнішнього рівня взаємодії представлено на рис. 3.

Рис.3. Схема взаємодії внутрішнього та зовнішнього рівнів

Робота кожного АПМ системи розпочинається із запуску комп'ютера в мережі, на якому він інстальований, та запуску цього модуля. Після запуску модуля виконується перехід до планування задач модуля. Відповідно до запланованої задачі відбувається активізація окремого підмодуля та запуск його функціоналу. Результати функціонування цього підмодуля надсилаються іншим частинами системи через зовнішній рівень взаємодії. Це дозволяє іншим АПМ системи бути сповіщеними про поточний стан кожної іншої частини системи.

У разі виявлення певним підмодулем атаки шкідливого програмного забезпечення активізується механізм реагування системи на загрозу. Цей механізм включає в себе процедуру відправки критичного повідомлення на усі інші АПМ системи. Якщо АПМ знаходится у

критичному стані приймається рішення про завершення роботи модуля. Описана взаємодія між внутрішнім та зовнішнім рівнями відображена на рис. 4.

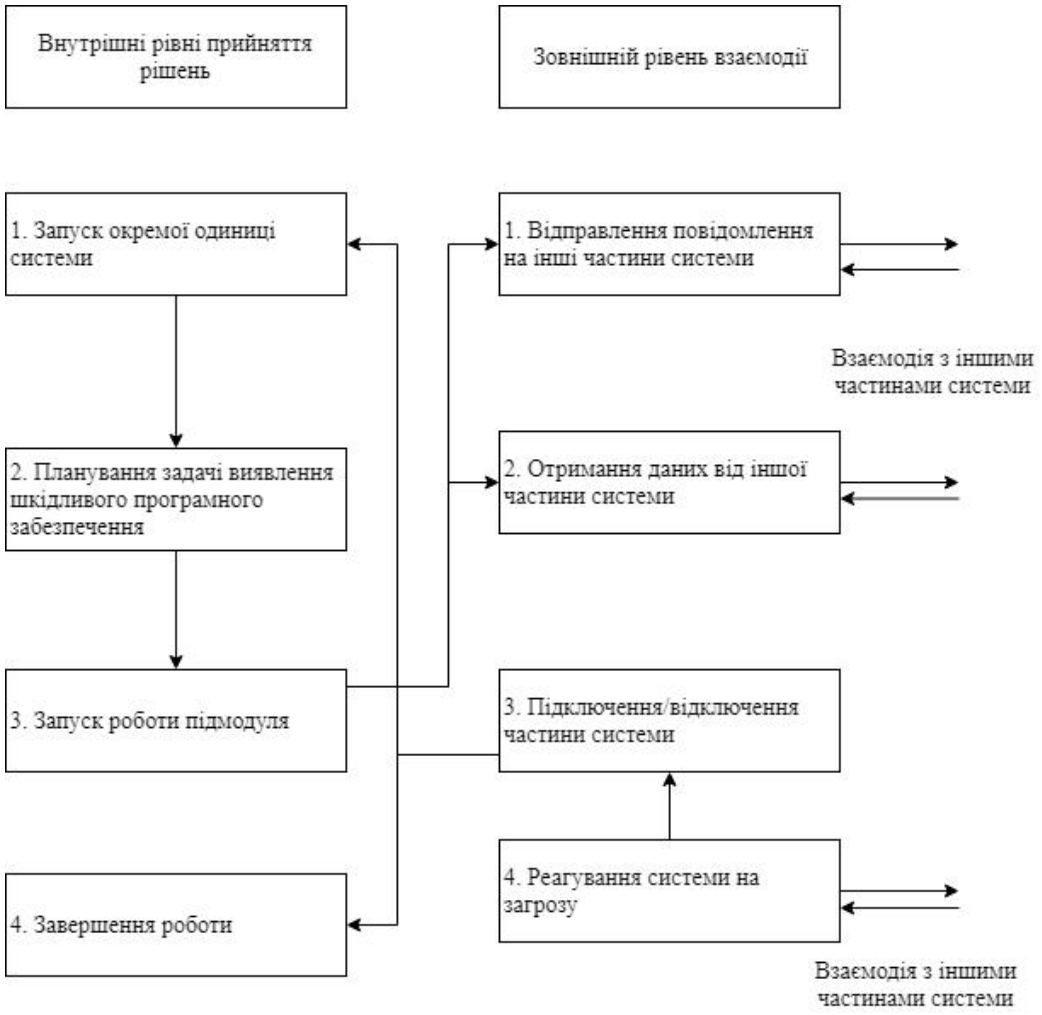

Рис. 4. Взаємодія між внутрішнім та зовнішнім рівнями

Частини системи взаємодіють між собою за допомогою повідомлень, причому кожне з них складається із заголовку та тіла.

В якості заголовку повідомлення виступає АПМ-ініціатор повідомлення – ідентифікатор АПМ, який надіслав повідомлення. Як ідентифікатор АПМ використовується IP-адреса комп'ютерної системи, на якій цей модуль встановлений.

Тіло повідомлення містить два поля:

1) тип повідомлення – ідентифікатор типу повідомлення;

2) дані – безпосередньо дані повідомлення.

Тип повідомлення використовується для отримання інформації про дані, що міститимуться безпосередньо у тілі повідомлення.

Можливі типи повідомлень представлено в таблиці 1. Залежно від типу повідомлення його тіло міститиме різні дані.

Таблиця 1. Типи повідомлень

№	Тип повідомлення	Опис типу
1	Модуль активовано	Повідомлення, що надсилається усім частинами системи у випадку активації модуля на конкретній комп'ютерній системі. Виступає у якості прохання відгукнутися усіх активних модулів у системі.
2	Модуль деактивовано	Повідомлення, що надсилається усім частинами системи у випадку деактивації модуля на конкретній комп'ютерній системі
3	Повідомлення-вітання	Повідомлення, що надсилається тому модулю, який запросив відгукнутися усіх інших активних частинам системи, з метою виявлення усіх активних частин системи.
4	Зміна задачі	Повідомлення, що надсилається усім частинами системи у випадку зміни виконуваної задачі конкретним АПМ.
5	Задача виконана	Повідомлення, що надсилається усім частинами системи у випадку завершення роботи над виконуваною задачею конкретним АПМ.
6	Повідомлення-опитування	Повідомлення, що надсилається усім частинами системи з метою отримання інформації про поточний стан кожної частини.
7	Повідомлення-стан	Повідомлення, що надсилається тому модулю, який запросив відгукнутися усіх інших активних частинам системи, з метою отримання інформації про поточний стан кожної частини.
8	Виявлено загрозу	Повідомлення, що надсилається усім частинами системи з метою інформування про виявлення загрози.

Стани, в яких перебувають АПМ змінюються під час їх роботи. Уся інформація стосовно зміни робочого стану надсилається усім іншим активним у мережі АПМ через повідомлення. Спеціальний підмодуль виконує обмін повідомленнями з іншими частинами системи. Якщо жоден інший АМП у системі не активований, відповіді на розіслане повідомлення отримано не буде. Час відправки та отримання кожного повідомлення зафіксовується. Обмін повідомленнями відбувається усіма АПМ з усіма іншими активними АПМ. Кожний АПМ зберігає у локальній базі даних статистичні дані стосовно власної роботи та інформацію стосовно результатів роботи інших частин системи. Така збережена інформація може бути використана для гнучкої адаптації, тобто автоматичної корекції роботи системи до обставин у середовищі, адже кожен модуль є поінформованим стосовно інших частин системи.

Залежно від кількості активних АПМ системи у мережі та від інтенсивності завантаження кожного конкретного модуля, за певну одиницю часу у базу даних може бути записана велика кількість інформації. Причому значний відсоток записаної інформації становитиме чисто технічні записи. Наприклад, повідомлення про те, що у певний момент часу АПМ активізувався на певній комп'ютерній системі. Звісно, через певний період часу такого роду інформація може бути вилучена з бази. Тому з метою оптимізації знань, накопичених у базі даних, передбачено підмодуль, функціонал якого забезпечує здійснення вилучення неактуальних для системи даних. В результаті накопичення досить великого об'єму знань, це дозволяє покращити ефективність роботи системи в цілому.

Кожен запис у базі даних має спеціальну мітку, залежно від типу даних, які він представляє:

1) звичайне "технічне" повідомлення;
2) попередження;
3) помилка у роботі якогось підмодуля АПМ;
4) критичний стан АПМ;
5) повідомлення про результат виконання завдання.

Усі АПМ у системі є структурно ідентичними та складаються з набору підмодулів, кожен з яких виконує певну задачу. Узагальнена схема робочих станів, в яких може перебувати кожен АПМ в процесі своєї роботи представлена на рис. 5 у вигляді графу.

Отже, можна виділити 9 основних станів, у яких знаходитимется АПМ певної комп'ютерної системи у конкретний момент часу:

1) активізація автономного програмного модуля;

2) планування наступної задачі на виконання, визначення переходу до станів 3, 4, 5, 6, 7, 8;

3) перевірка файлів на жорсткому диску, визначення переходу до станів 2, 5;

4) перевірка запущених процесів в оперативній пам'яті, визначення переходу до станів 2, 6;

5) сканування виконуваних файлів на жорсткому диску, перехід до стану 2;

6) сканування запущених процесів, перехід до стану 2;

7) виконання оптимізації записів у базі даних, перехід до стану 2;

8) завершення роботи автономного програмного модуля;

9) позначення комунікації з іншими модулями у системі, формально АПМ не переходить до цього стану.

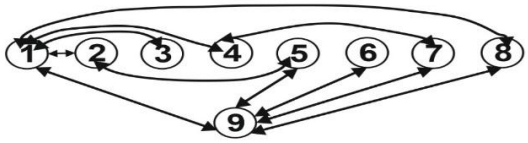

Рис. 5. Узагальнена схема робочих станів АПМ системи у вигляді графу

АПМ потрапляє у стан 1 при його активізації або користувачем, або комп'ютерною системою, на яку він встановлений, що спричиняє процес перевірки характеристик комп'ютерної системи та порівняння їх із збереженими раніше характеристиками. Після чого відбувається формування спеціального повідомлення про активацію роботи іншим АПМ у системі.

У стан 8 АПМ потрапляє при деактивації або користувачем, або при завершенні роботи комп'ютерної системи, на яку він встановлений. Деактвація АПМ відбувається лише після того, як усі інші активні АПМ будуть поінформовані спеціальним повідомленням про вимкнення.Основним робочим станом є 2 стан, на якому виконуються збір результатів виконання попередньої задачі та планування наступної. Залежно від результатів моніторингу подій у системі або змін на конкретній комп'ютерній системи відбувається прийняття рішення про перехід на один із станів - 3, 4, 5, 6 та 7, - на яких АПМ зайнята конкретною задачею. Стани 3 та 5 позначають процес перевірки та сканування файлів на жорсткому диску комп'ютерної системи, а 4 та 6 - перевірка та сканування запущених процесів. Перебуваючи в 7 стані, АПМ здійснює оптимізацію записів у локальній базі даних.

Стан 9 позначає процес обміну отриманими результатами з іншими активними частинами системи.Переходячи з одного стану в інший АПМ формує спеціальне повідомлення з інформацією про перехід до виконання певного завдання. Після завершення завдання АПМ розсилає повідомлення з отриманими результатами. Обробка отриманих результатів на всіх АПМ виконується за єдиним алгоритмом, після чого відбувається реагування на отриманий результат. Кожний АПМ позначено як "Defender Application" за назвою реалізованої системи. Перелік функціональних можливостей, якими володіє розроблена система з погляду користувача цієї системи відображено на use-case діаграмі на рис. 7.

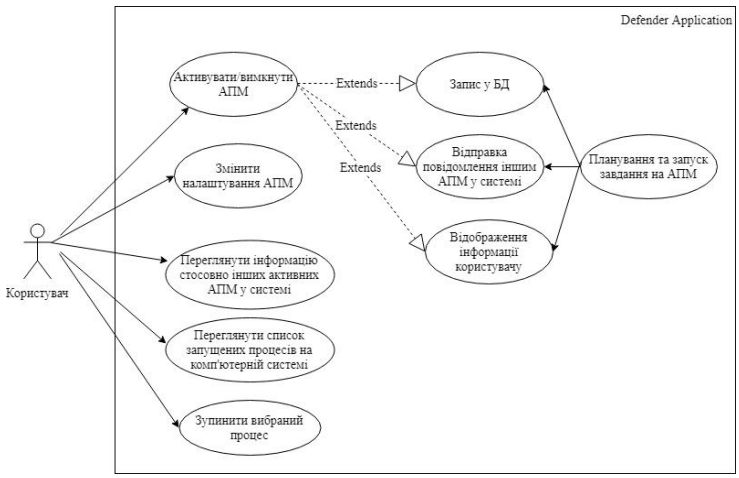

Рис. 7. Use-case діаграма розробленої системи

Взаємодія між користувачем та розробленою системою відображена на рисунку 3.9 у вигляді діаграми послідовностей.

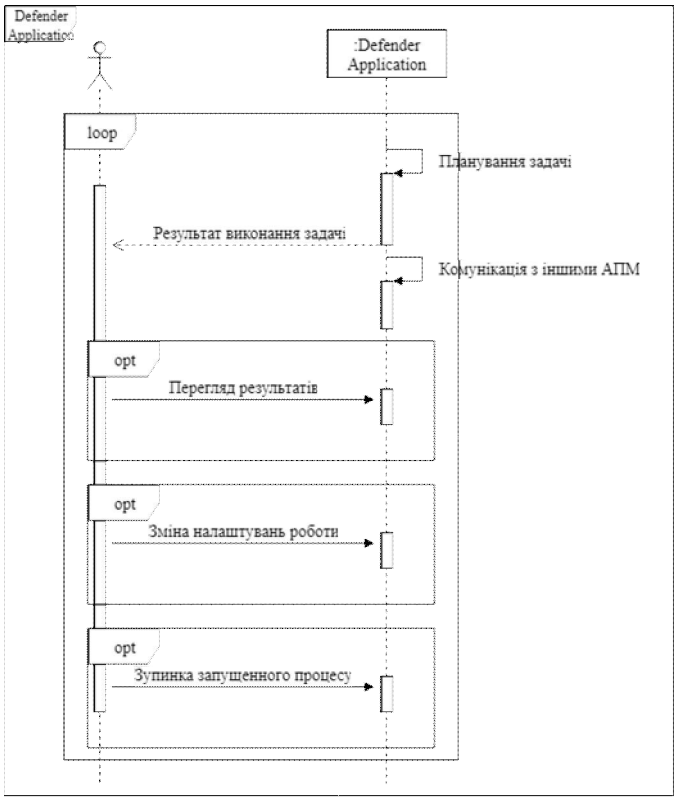

Рис. 8. Діаграма послідовностей розробленої системи

Як видно із наведеної вище діаграми послідовностей, кожен окремий модуль системи встановлений на конкретній комп'ютерній системі є автономним, тобто не потребує впливу ззовні для прийняття рішень стосовно планування завдань та обміну повідомленнями з іншими модулями у системі. Користувач може лише переглядати результати, змінювати деякі налаштування та використовувати додатковий функціонал, наприклад, зупиняти запущені процеси. Головне вікно користувацького інтерфейсу відображено на рис. 9.

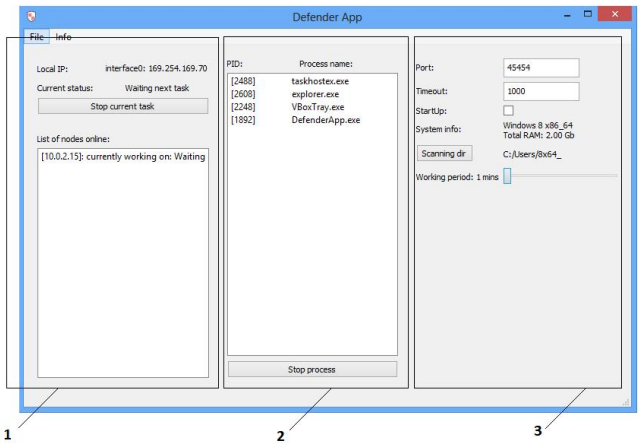

Рис. 9. Головне вікно користувацького інтерфейсу Defender App

Як видно із рисунку 9, користувацький інтерфейс розділено на три основні області:

1) інформація про поточний стан роботи та список інших модулів системи, які є активними на даний момент у мережі;

2) список запущених процесів на даній комп'ютерній системі;

3) налаштування.

Висновки. Розроблена архітектура розподіленої багаторівневої програмної системи базується на принципах децентралізації та самоорганізації і дозволяє здійснювати її наповнення різними функціоналами виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах. РБПС відноситься до реагуючих систем, яка постійно здійснюватиме моніторинг запущених процесів та виконуваних програм в комп'ютерних системах мережі. Об'єктами для дослідження зі сторони РБПС є перевірка наявного програмного забезпечення та запущених процесів в КС локальної мережі на можливість віднесення до шкідливого програмного забезпечення. Основою архітектури РБПС виступають автономні програмні модулі з однаковими архітектурами, але при цьому кожен з них може самостійно приймати рішення на основі різних даних зібраних з різних КС мережі. Для ефективної роботи РБПС необхідним є розробка методів і моделей взаємодії та узгодження роботи різних програмних модулів між собою та їх відповідних рівнях і деталізація структури її станів.

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. No.23 P. 145–156.
5. A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering / Wang G., Hao J.,Ma J., Huang L. *Expert Systems with Applications: An International Journal*. 2010. Vol. 37. Issue 9. P. 6225–6232.