УДК 004.491.42

О.С. САВЕНКО
Хмельницький національний університет

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

В роботі розроблено модель та архітектуру розподіленої багаторівневої систем виявлення шкідливого програмного забезпечення в локальних комп'ютерних мережах, побудовану на основі принципів децентралізації та самоорганізації. Розподілена багаторівнева система відноситься до реагуючих систем і постійно здійснюватиме моніторинг запущених процесів та виконуваних програм в комп'ютерних системах мережі. Розроблена системи здійснюватиме перевірку наявного програмного забезпечення та запущених процесів в комп'ютерних системах локальної мережі на можливість віднесення до шкідливого програмного забезпечення.

Ключові слова: модель, архітектура, структура Кріпке, шкідливе програмне забезпечення, поведінка, розподілена система, принцип самоорганізації, локальна комп'ютерна мережа.

O.S. SAVENKO
Khmelnytskyi National University

MODEL AND ARCHITECTURE OF THE DISTRIBUTED MULTILEVEL SYSTEM OF DETECTION MALWARE IN THE LOCAL COMPUTER NETWORKS

In the work the model and architecture of distributed multilevel detection systems of malicious software in local computer networks, based on the principles of decentralization and self-organization, has been developed. The basis of the constructed model of distributed multi-level systems is its structural parts, which are represented by program modules, which can be in different states. The transition between the classes of program modules is based on a defined set of transitions. Interaction and communication between software modules is based on their presence in certain states during operation. Distributed multi-level systems is a responsive system that will monitor selected events. Each program module contains a resident mechanism, the motive mechanisms for the transition between states, the transitions between which are given subsets of transitions, the data for which will be formed using artificial intelligence technologies. The developed system will verify the existing software and running processes in the computer systems of the local network to the ability to refer to malicious software. The principles and models of systems development are important for the theory and practice of creating effective systems for detecting malicious software in local computer networks built on the basis of decentralization and self-organization principles. The designed architecture of the distributed multilevel system allows it to be filled with various functions of detecting malicious software in local computer networks. The developed model of the architecture of the software modules of distributed multilevel systems is based on the principles of autonomy and multilevel. It allows you to increase the number of levels of the system without changing its architecture. The basis of the architecture of distributed multi-level systems are software 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.

Keywords: model, architecture, structure Kripke, malware, behaviour, distributed system, self-organization principle, local computer network.

Вступ. Постановка задачі. Відомі антивірусні засоби не завжди підтверджують задекларований рівень достовірності виявлення нового шкідливого програмного забезпечення. На противагу їм на сьогоднішній день існує значна кількість некомерційних засобів, заснованих на нових методах та підходах до виявлення ШПЗ, що відрізняються ознаками, які використовуються для ідентифікації ШПЗ, способу отримання даних, методами аналізу. Реалізація нових принципів, моделей та методів виявлення конкретних типів шкідливого програмного забезпечення (ШПЗ) шляхом створення відповідних інформаційних систем потребує подальшого розвитку, але досягнення підвищення достовірності виявлення нового та відомо ШПЗ, в якому застосовуються комбіновані технології зловмисного проникнення та поширення, можливе також за умови комбінованого використання нових та відомих методів виявлення. 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.

Сучасне шкідливе програмне забезпечення представляє собою складні багатофункційні програмні системи та комплекси, які побудовані з використанням ефективних методів створення програмних засобів та методів поширення зловмисного коду. Для організації ефективної протидії таким засобам, необхідним є розробка теорії та практики створення систем виявлення ШПЗ, основа архітектури яких була б подібною до архітектури складного комплексу ШПЗ, але мала б більшу функційність. Важливим є здійснення побудови таких ефективних систем для їх використання в локальних комп'ютерних мережах, де уможливлюється встановлення таких систем на усіх комп'ютерних системах (КС) мережі, а не локально. Це при проведенні аналізу зібраних типових даних з різних КС підвищить достовірність виявлення.

Проблемою виступає те, що не розроблені інформаційні системи [1–5], які б увібрали в себе розроблені для певних типів шкідливого програмного забезпечення різні методи виявлення. І ці інформаційні антивірусні системи, будучи багатофункціональними, могли б дозволити наповнюватися новим функціоналом і стали б основою для розвитку конкурентного середовища на ринку антивірусних інформаційних систем. Ефективність таких систем виросла б при використанні їх в корпоративних

комп'ютерних мережах, де додатковою перевагою при виявленні шкідливого програмного забезпечення виступають відомості про програмно-апаратне забезпечення комп'ютерних систем мережі та обліку інформації про поведінку програмного забезпечення з різних комп'ютерів мережі.

Тому, з метою підвищення ефективності та достовірності виявлення нового ШПЗ, доцільним завданням є розробка багатофункційних систем виявлення, які базуватимуться на основі сучасних технологій створення складних програмних систем, а також подальший розвиток теорії і практики створення таких систем.

Основна частина. Виявлення шкідливого програмного забезпечення здійснюється за допомогою різноманітних засобів. Ефективність та достовірність виявлення 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 РБС 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 проектована 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. Проектована система повинна бути побудована так, щоб розміщені на КС в мережі її частини ефективно спілкувалися між собою для обміну інформацією про стан КС з метою надання додаткової інформації для прийняття рішення. Розроблювана система виявлення ШПЗ повинна відповідним чином структуруватись, щоб мати можливість нарощуватись і її збільшення не повинно сповільнювати процес виявлення. Таким чином, сучасна проектована система виявлення ШПЗ в локальних мережах повинна володіти такими характеристиками: розподіленість в просторі в локальній мережі на всіх КС; формування єдиної системи всіма частинами, розміщеним в КС; децентралізованість структури для уникнення виведення системи з ладу за умови виявлення єдиного центру прийняття рішень чи управління системою; можливість прийняття рішення модулем, розміщеним в певній КС, за підтримки інших структурних частин системи; багаторівневість для чіткого відокремлення різних за призначенням функціональних частин і можливості нарощування; здатність до самоорганізованості при вирішенні завдань та за умови порушення цілісності системи, під час атак на інформаційно-комунікаційні ресурси мережі або ураженні системи ШПЗ; динамічне формування архітектури з наявних частин робочих КС мережі; адаптивність до середовищ КС мережі та специфіки задач, вирішуваних конкретними КС; можливість автоматизованої підтримки адміністратором мережі для налаштування роботи та отримання звітності; автономність системи у прийнятті рішень; відповідність політиці безпеки підприємства (організації). Відповідність системи заданим характеристикам впливає на її модель, яка стане основою архітектури системи і згідно характеристичних вимог до системи включатиме комбінацію моделей колективного інтелекту, багатоагентних систем, розподілених систем, децентралізованих систем, самоорганізованих та адаптивних систем. Врахування такої моделі підвищить надійність функціонування та живучість системи в локальній мережі.

Рис. 1. Узагальнена схема основних складових архітектури системи

Рис. 2. Схематичне представлення поширення ШПЗ та місце його дослідження в фрагменті глобальної мережі

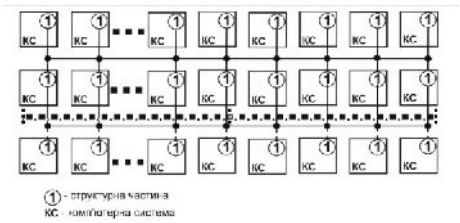

Рис. 3. Схема системи з врахуванням моделі розподіленої архітектури

Виділимо особливості кожної моделі та інтегруємо їх в модель розроблюваної системи. Архітектура розроблюваної системи представлена узагальненою схемою основних складових на рис. 1.

Враховуючи, що процес виявлення ШПЗ відбуватиметься в локальній мережі, тоді спочатку представимо (рис. 2) місце поширення і виявлення ШПЗ, яке в глобальній мережі поширюється постійно і неконтрольовано. А також, зобразимо модель, яка відображає таку характеристику архітектуру системи як розподіленість (рис. 3). Поетапно ця розподілена архітектура буде наповнюватись підсистемами, в яких реалізовуватимуться інші моделі, що входять до проектованої системи.

Введемо позначення для проектованої розподіленої багаторівневої

системи (РБС) і позначимо її символом А, тоді структурні частини системи – A_i, i=1,…,n, де n – кількість програмних модулів. Представлення системи через її структурні частини здійснимо так:

$$A=\{A_1,\ldots,A_n\} \quad (1)$$

Такий формалізований опис надалі бути використаний для оптимізації зв'язків та залежностей між елементами системи та підсистем. Елементи A_i однакові за структурою, але в процесі функціонування системи матимуть різне наповнення та можуть перебувати в різних станах.

РБС розподілена в просторі і згідно характеристичних вимог повинна бути децентралізованою, тобто система не повинна мати єдиного центру керування всіма її частинами та можливості прийняття рішень в залежності від зміни стану будь-якої КС мережі. В цьому контексті децентралізованість і самостійність прийняття рішень модулем системи розміщеним в певній КС не ототожнюються. В поняття децентралізованості проектованої 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 передача цієї інформації на відповідні рівні, завершення роботи. Для частини системи розміщеної в КС введемо поняття програмного модуля (ПМ) системи. Самостійність в прийнятті рішення програмним модулем системи включає такі можливості: прийняття рішення про стан загрози ШПЗ для КС на основі інтегрованої інформації з інших рівнів програмного модуля і передача цього значення на рівень децентралізації; передача 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 задач дослідження ШПЗ. На рис. 4 зображена взаємодія цих двох рівнів та їх основні функції-підсистеми.

Рис. 4. Основні функції рівнів децентралізації та прийняття рішень і схема їх взаємодії

Розглянемо детальніше блоки децентралізації та прийняття рішень і їх взаємодію. Початок роботи конкретного модуля розпочинається із запуску конкретної КС, на яку він встановлений, та безпосередньо запуску цього модуля. Спочатку запускається функція стартового підблоку, в якому вирішується питання про початок і послідовність виконання функцій інших блоків програмного модуля. Першим підблоком, на який передається вказівка про запуск, є підблок, який визначає стан загрози КС від ШПЗ при старті. Для цього він передає вказівку на запуск внутрішніх підблоків блоку прийняття рішень. Після отримання результатів від них, тобто встановлення рівня безпеки, він передає вказівку про запуск і значення рівня безпеки на підблок, який встановлює програмний модуль на відповідний рівень безпеки та запускає визначені для цього рівня функції, тобто визначає необхідний функціонал для роботи програмного модуля. Після цього він направляє вказівку для запуску модуля, який приймає рішення про зв'язок з іншими програмними модулями всієї системи. Якщо рівень безпеки програмного модуля показує, що КС перебуває в критичному стані безпеки, то рішення про комунікацію з іншими складовими системи буде негативним і при самому надкритичному рівні передається на підблок завершення роботи програмного модуля. Якщо ж рішення позитивне, тоді на наступному кроці формується пакет для розсилки всім іншим частинам системи і відправляється. Також, цим підблоком збирається та опрацьовується інформація отримана від інших частин системи. Узагальнена схема зв'язків підблоків блоків децентралізації та прийняття рішень представлені на рис. 5.

Рис. 5. Узагальнена схема взаємодії блоків децентралізації та прийняття рішень

Багаторівневість проектованої системи дозволить відділити процеси, які відноситимуться до функціонування системи в цілому, процеси функціонування програмного модуля і розділити з рівнями задачі по виявленню різних типів ШПЗ та мережних атак і необхідних утиліт для вирішення цих задач. Рівні децентралізації та прийняття рішень є базовими і в проектованій багаторівневій системі представлені на рис.6 у взаємодії з таких же рівнів інших програмних модулів РБС. Крім того, багаторівневість дозволить

проводити нарощування системи новим функціоналом, а також чітка відокремленість різних за задачами частин дозволить сумісне використання утиліт 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 зміну стану безпеки. Самоорганізованість на рівні всієї РБС здійснюватиметься в підблоці блоку децентралізації і включатиме функціонування всієї системи та її переходу в різні стани в залежності від зовнішніх змін в 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.

Рівень 1	Децентралізація
Рівень 2	Прийняття рішень
Рівень 3	▪▪▪
▪▪▪	▪▪▪
Рівень n	▪▪▪

Рис. 6. Схема багаторівневості РБС

Формування архітектури РБС в мережі здійснюватиметься на протязі 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 задач виявлення ШПЗ. Адаптивність РБС проявлятиметься в автоматичній зміні алгоритмів свого функціонування і за потреби своєї структури з метою збереження або досягнення оптимального стану при зміні зовнішніх умов.

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

Спроектована таким чином архітектура РБС матиме можливість наповнення різними функціоналами виявлення ШПЗ в локальних мережах і дозволить використовувати таку систему автономно. Дана архітектура системи є концептуальною, тому потребує деталізації рівнів програмних модулів.

Модель та архітектура програмних модулів РБС виявлення шкідливого програмного забезпечення. Проектування повнофункційної РБС потребує розробки її моделі функціонування в локальній комп'ютерній мережі в частині організації взаємодії між її програмними частинами, розміщеними в мережі, та спілкування між ними.

Розподілена багаторівнева система в локальній комп'ютерній мережі згідно її моделі представлена однаковими програмними модулями, які розміщені в кожній комп'ютерній системі. Тобто, не залежно від того скільки комп'ютерних систем в 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 в локальній мережі формують безпосередньо РБС. Це дозволить навіть за наявності роботи двох комп'ютерних систем підтримувати виконання задач системи.

Кожен програмний модуль РБС містить на відповідних рівнях функціонал для виявлення певного типу шкідливого програмного забезпечення або атаки.

РБС містить в 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. Важливим є ідентифікація одним конкретним програмним модулем решти програмних модулів для активації та роботи цілісної системи. Для цього при встановленні програмного на конкретну КС активується функціонал, що здійснює зчитування системної інформації про апаратно-програмне забезпечення КС, зберігає цю інформацію в своєму внутрішньому сховищі, формує на її основі ідентифікаційну ознаку цього конкретного модуля, зберігає її як ідентифікатор програмного модуля і використовує її в пакетах, які розсилатимуться на інші програмні модулі РБС. При встановленні всіх програмних модулів РБС в мережі здійснюють їх активацію з метою збору всіх ідентифікаційних ознак. Кожен окремий програмний модуль в системі після повного встановлення системи міститиме характеристики та ідентифікатори всіх програмних модулів системи. Використання такої ідентифікаційної інформації в якості окремого поля пакету дозволить однозначно ідентифікувати та розрізнювати різні КС в локальній мережі, але при цьому віддаленість КС не важлива. Врахування віддаленості відбувається на рівні підтвердження цілісності системи.

Враховуючи, що стани програмних модулів системи можуть змінюватись протягом часу їх використання, то в програмних модулях будуть накопичуватись статистичні данні роботи як за добу так і більших інтервалів часу. Ця статистична інформація є корисною для уточнення прийняття рішень, а також на певних етапах буде необхідним за певними правилами видаляти частину інформації з неї. Протягом робочого дня такий обмін інформацією між програмними модулями системи може бути багатократним. В зв'язку з цим необхідним буде також розробка підсистеми, яка оптимізуватиме накопичену інформацію.

Кількість рівнів програмного модуля та їх наповнення залежатимуть від завдань, які вирішуватимуться. Основними завданнями проектованої системи є проведення аналізу виконуваних файлів з метою виявлення зловмисного функціоналу і поведінки виконуваного в КС програмного забезпечення (запущених процесів) на наявність зловмисних дій. Обидві ці задачі 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 зловмисну поведінку для аналізу. Тобто, проектована система включатиме засоби дослідження програмного забезпечення розміщеного в КС, яке перебуватиме в активному (виконуваному) стані або в зовнішній пам'яті, спільним елементом яких буде відслідковування і аналіз їх поведінки. В програмному забезпеченні досліджуватиметься наявність таких його можливостей: приховувати ознаки своєї присутності в комп'ютерній системі; володіти здатністю до самокопіювання, в тому числі створення модифікованих своїх копій; мати здатність до асоціювання себе з іншими програмами; мати здатність до переносу своїх фрагментів в інші області оперативної або зовнішньої пам'яті, в тому числі тих, що знаходяться на віддаленому комп'ютері; отримати несанкціонований доступ до компонентів або ресурсів КС; руйнувати або спотворювати код програм в оперативній пам'яті; спостерігати за процесами обробки інформації та принципами функціонування засобів захисту; зберігати фрагменти інформації з оперативної пам'яті в деякій області зовнішньої пам'яті; спотворювати, блокувати або підміняти виведений в зовнішню пам'ять або канал зв'язку інформаційний масив, що утворився в результаті роботи прикладних програм; спотворювати масиви даних, які знаходяться у зовнішній пам'яті; пригнічувати інформаційний обмін в комп'ютерних мережах, фальсифікувати інформацію в каналах зв'язку; нейтралізовувати роботу тестових програм та засобів захисту інформаційних ресурсів КС; постійно або тимчасово змінювати ступінь захищеності секретних даних; приводити в неробочий стан або руйнувати компоненти системи; створювати приховані канали передачі даних; ініціалізувати раніше впроваджене ШПЗ.

Підсистеми рівнів програмних модулів РБС, враховуючи перераховані функціональні можливості програмного забезпечення та вирішення проблеми їх виявлення через дослідження і аналіз поведінки, повинні мати такі складові: сигнатурний аналізатор; евристичний та поведінковий аналізатор; емулятор процесора; підсистема виявлення несанкціонованих змін з використанням контрольних кодів цілісності; підсистема моніторингу потенційно небезпечних дій; підсистема створення несправжніх об'єктів атаки в мережі; підсистема внесення невизначеності в роботу об'єктів та засобів захисту КС мережі; контроль ходу виконання програм; підсистема обліку роботи КС.

Узагальнена схема розподілу завдань в програмних модулях системи включатиме стани, в яких перебуватиме програмний модуль конкретної КС в процесі свого життєвого циклу, і представлена на рис.7 у вигляді графу та вершин співвіднесених до різних рівнів.

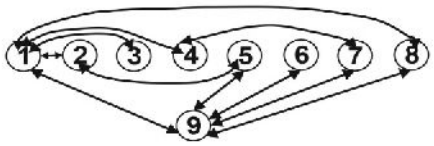

Рис. 7. Граф узагальнених станів програмного модуля системи:
1 – базовий стан, моніторинг КС, визначення переходу до станів 2, 3, 4, 8;
2 – перевірка виконуваних файлів, визначення переходу до станів 1, 5;
3 – перевірка запущених процесів та мережних активностей КС, визначення переходу до станів 1, 6;
4 – перевірка виконуваних файлів та запущених процесів, визначення переходу до станів 1, 7;
5 – перевірка виконуваних файлів з 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 – обробка виконуваних файлів та запущених процесів з використанням інших програмних модулів системи;
8 – обробка та оптимізація інформації з бази пакетів програмного модуля із залученням інформації з інших програмних модулів системи, повернення на рівень 1;
9 – забезпечення зв'язку з іншими програмними модулями системи

Модель РБС, що враховуватиме архітектурні складові і їх стани та зв'язки між ними, представимо:

$$M_A^S = S, G_A, \tag{2}$$

де S – множина станів системи, G_A – орієнтований граф узагальнених станів РБС, який зображено на рис. 8.

Кожен програмний модуль системи має однакову структуру і розділений на чотири рівні в залежності від функційного призначення та згрупованих в них завдань:

1. – рівень 1 включає моніторинг подій та визначення переходів до наступних рівнів, а також здійснює обробку інформації, що надходить від інших програмних модулів;

2. – рівень 2 включає перевірку виконуваних файлів, перевірку запущених процесів та мережних активностей без залучення інформації з інших програмних модулів системи;

3. – рівень 3 включає виконання завдань рівня 2 із залученням інформації з інших

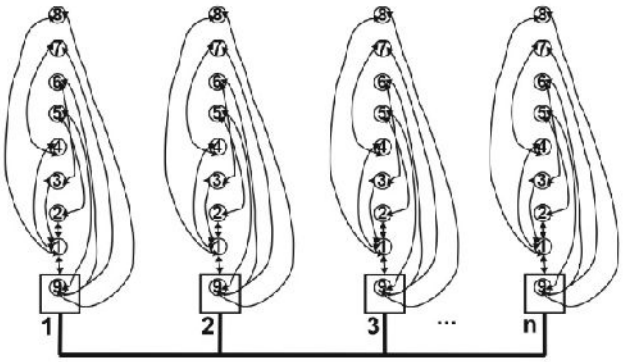

Рис. 8. Орієнтований граф зв'язків узагальнених станів програмного модуля системи

програмних модулів системи;

4. – рівень 4 здійснює обробку, оптимізацію та вилучення інформації з бази програмного модуля КС.

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

Таким чином, перший рівень містить засоби, які забезпечують автономність роботи програмного модуля системи. На третьому рівні досягається глибший аналіз досліджуваних об'єктів в порівнянні з другим рівнем, що підвищує ефективність системи. Четвертий рівень вирішує частину задач із самоорганізації системи, пов'язану з оптимізацією накопиченої за час роботи інформації.

Кожен рівень та відповідні йому узагальнені підсистеми в свою чергу теж представляються наборами підрівнів, яким закладено виконання певних функціоналів. Зображення взаємозв'язку підрівнів представлено на рис. 9 графом з вершинами, що відповідають призначенням підрівнів.

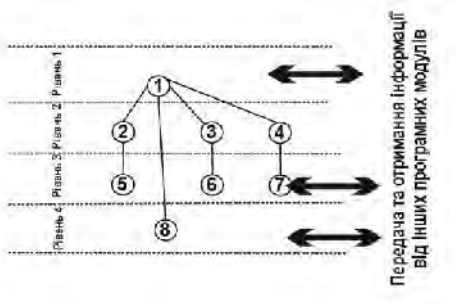

Рис. 9. Граф-схема взаємодії підрівнів програмних модулів системи

Представлення інформації про поточний стан програмних модулів РБС може бути представлений матрицями поточних станів, що дозволяє здійснювати аналіз системи в цілому та готувати рішення про перехід програмних модулів на інші рівні або призупинку роботи комп'ютерних систем в мережі з видачею відповідного звіту для адміністратора. Крім того, представлення інформації про програмні модулі деталізується до станів програмних модулів, що є більш інформативним для прийняття рішень.

Розроблена концептуальна модель архітектури програмних модулів РБС базується на принципах децентралізації та багаторівневості. Вона дозволяє здійснювати наповнення всієї системи різними функціоналами виявлення ШПЗ та побудована таким чином, що дозволяє збільшувати кількість рівнів системи без зміни її архітектури. Основою архітектури РБС виступають програмні модулі, які спроектовано однаковими архітектурами, але які дозволяють містити різні данні зібрані з КС. Для ефективної роботи РБС необхідним є розробка принципів взаємодії та узгодження роботи різних програмних модулів між собою та їх відповідних рівнях.

Модель РБС та принципи взаємодії програмних модулів різних КС на основі самоорганізації та багатоагентного підходу. Представлення розподіленої багаторівневої системи у формі ПМ, розміщених в КС локальної мережі, та врахування характеристики децентралізованості системи вимагає розробки такого методу взаємодії програмних модулів різних КС, який би базувався на таких принципах: спілкування програмних модулів між собою; виконання програмними модулями поставлених завдань без будь-якого впливу ззовні системи; самовідтворення РБС; програмні модулі повинні впливати на себе та інші програмні модулі; система та її частини повинна здійснювати самоконтроль; РБС повинна утворювати свої власні цілі.

Врахування при проектуванні РБС цих принципів дозволить створити розподілену децентралізовану систему, яка вирішуватиме поставлені завдання за рахунок наповненого функціоналу та створюватиме власні цілі для роботи. Розробка правильної організації взаємодії модулів системи впливатиме на ефективність виконання поставлених завдань та роботу системи в цілому.

Організація взаємодії програмних модулів системи в процесі спілкування вимагатиме розгляду та опису таких подій на різних етапах їх життєвого циклу:

1) запуск КС та відповідно його програмного модуля спричиняє необхідність виконання зчитування характеристик КС та порівняння їх з тими, що зберігаються в базі; формування пакету інформації з ідентифікатором про початок роботи програмного модуля;

2) відправка пакету інформації з ідентифікатором про початок роботи іншим програмним модулям системи;

3) якщо не всі КС увімкнені, тоді РБС буде формуватись виключно з тих, які вже працюють;

4) модель поведінки користувачів передбачає різний час початку роботи в локальній мережі, тому обов'язково буде КС, яка була увімкнена першою, активувала програмний модуль і з неї було розіслано пакет інформації, але жодна КС не увімкнена, тому відповіді від них отримано не буде; час відправки пакету фіксуватиме програмний модуль;

5) після ввімкнення другої та інших КС мережі, в них формуються пакети інформації про початок роботи програмних модулів системи і надсилаються за реєстром на інші КС мережі, в тому числі і на першу

ввімкнену КС, яка після отримання першого з них відправляє повторно свій пакет інформації про початок роботи; таким чином, вже не буде жодної першої ввімкненої КС, якій не відповів хоча б один програмний модуль іншої КС; обмін пакетів відбудеться усіх з усіма ввімкненими; якщо не всі КС за певний час будуть ввімкнені, а додадуться через тривалий час, тоді всі події по їх формуванні системи будуть ті ж, тобто розсилатимуться пакети інформації про початок роботи;

6) кожен програмний модуль РБС після сформування системи здійснює моніторинг подій своєї конкретної КС і при виникненні події, що переводить його в інший стан, який потребуватиме при виконанні завдання підтримки інших програмних модулів системи, розсилає пакет інформації з відповідним ідентифікатором, що видає завдання іншим програмним модулям, які після його виконання повертають йому результат роботи (час повернення результату, як правило є різним);

7) програмний модуль системи, перебуваючи на певному відповідному рівні, надсилає решті програмних модулів вказівку та інформацію про повне виконання завдання з відповідним ідентифікатором; кожен ПМ виконує перевірку та розсилає усім модулям системи результат; у випадку великої завантаженості КС, рівень якої встановлюється ПМ, тоді надсилається відповідне повідомлення та видане завдання очікує в черзі; сумарний результат оцінюється кожним ПМ та розсилається на інші ПМ; обробка такої статистичної інформації в різних ПМ здійснюється за єдиним алгоритмом та приймається рішення про реагування системи в цілому на отриманий результат, бо він буде на отриманий кожним програмним модулем;

8) завершення роботи КС і її вимкнення відбувається тільки після надсилання решті ПМ системи про вимкнення пакетом з відповідним ідентифікатором.

Важливим елементом аналізу спілкування програмних модулів РБС є побудова її часової моделі у вигляді відповідного графіку від старту та безпосередньо роботи програмних модулів і системи, яка демонструватиме залежність між програмними модулями і їх станами від часу функціонування. Параметри такого представлення зображено на рис. 10.

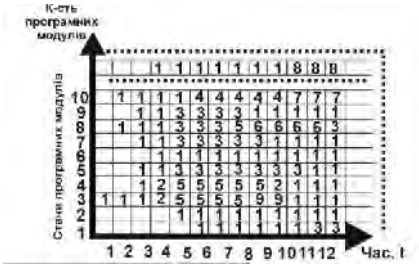

Рис. 1. Часова модель життєвого циклу РБС

Відображення станів ПМ вагами критичності в матриці дозволить проводити динамічну обробку подій, які впливатимуть на наступні стани РБС. Формування такої матриці відбуватиметься за потреби, зокрема у випадку дослідження мережної активності в локальній мережі.

З рис.10 видно, що 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 зі зворотними зв'язками. Тому, представимо РБС моделлю на основі структури Кріпке:

$$M_{РБС} = \quad S, S_0 \ , R, F_{РБС} \quad , \tag{3}$$

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 з множини S в підмножину атомів, які є істинними у відображуваних станах. Оскільки, проектована система є розподіленою, тоді множину станів системи представимо через підмножини станів, які відносяться до програмних модулів A_i, i=1,2,...,n, а саме: $S = \sum_{i=1}^{n} S_i$, тобто множини станів ПМ формуватимуть множину станів, в якій перебуватиме система.

Аналогічно, множина початкових станів РБС $S_0 = \sum_{i=1}^{n} S_{0i}$, причому серед їх елементів не може бути однакових. З будь-якого стану $s \in S$ існує мінімум один перехід до іншого стану, тобто справедливим є відношення $R \subseteq S \times S$, що означає для будь-якого стану з множини станів існує відповідний йому стан з цієї ж множини. Якщо множина атомарних тверджень T_A скінчена, тоді множина її підмножин 2^{T_A} буде відображенням $F_{РБС}$ для множини S в ті підмножини атомів, які будуть істинними в $s \in S$.

Аналогічно, представимо структурну частину РБС програмний модуль:

$$M_{A_i} = \quad S_i, S_{0i} \ , R_i, F_{A_i} \quad , \tag{4}$$

де i=1,2,...,n, R_i – множина переходів між станами $s_{ij} \in S_i$, j – кількість i-х станів, F_{A_i} – функція відображення множини станів S_i в множину підмножин множини T_{A_i}, тобто $2^{T_{A_i}}$. Кожен стан обов'язково має зв'язок з деякими іншими станами. Перехід з одного стану до іншого, якщо між ними є зв'язок, відобразимо послідовністю

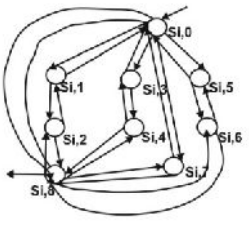

Рис. 11. Структура Кріпке програмного модуля A_i

$s_{ij}s_{ip}$, де i – номер програмного модуля, j та p – 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 на протязі його експлуатації. РБС в процесі експлуатації характеризуватиметься множиною послідовностей переходів з стану в стан програмних модулів. Побудуємо структуру Кріпке для ПМ за його діаграмою переходів (рис. 11).

В табл. 1 представимо стани з множини станів програмного модуля S_i двійковими послідовностями, а також кодування програмних модулів.

Таблиця 1

Стани з множини станів програмного модуля

Номер програмного модуля S_i	Стани	Кодування станів	Кодування програмного модуля S_i
i	$s_{i,0}$	0000	Представлення десяткового числа і двійковим числом з 6-ма розрядами (кількість розрядів може бути змінена за потреби)
	$s_{i,1}$	0001	
	$s_{i,2}$	0010	
	$s_{i,3}$	0011	
	$s_{i,4}$	0100	
	$s_{i,5}$	0101	
	$s_{i,6}$	0110	
	$s_{i,7}$	0111	
	$s_{i,8}$	1000	

Наприклад, для i=5 тобто для п'ятого програмного модуля системи його код визначатиметься послідовністю 000101 і тоді відповідно множина станів:

$S_5 = \{0000000101, 0001000101, 0010000101, 0011000101, 0100000101, 0101000101, 0110000101, 0111000101, 1000000101\}$.

Позначимо V – множину змінних, які впливатимуть на зміну станів модулів, тоді $V = \sum_{i=1}^{n} V_i$, де V_i – множина змінних i-го модуля. Тоді, V' – множина змінних, які впливатимуть на зміну наступних станів модулів, тоді $V' = \sum_{i=1}^{n} V'_i$, де V'_i – множина змінних i-го модуля.

R_i – множина переходів між станами i-го модуля на множинах змінних та визначається так:

$$R_i = \{ \begin{array}{llllllll} 0000i & 0001i, & 0001i & 0000i, & 0001i & 0100i, & 0100i & 0001i, \\ 0100i & 1000i, & 1000i & 0100i, & 0000i & 1000i, & 1000i & 0000i, \\ 0000i & 0010i, & 0010i & 0000i, & 0010i & 0101i, & 0101i & 0010i, \\ 0101i & 1000i, & 1000i & 0101i, & 0000i & 0011i, & 0011i & 0000i, \\ 0011i & 0110i, & 0110i & 0011i, & 1000i & 0110i, & 0110i & 1000i, \\ 0000i & 0111i, & 0111i & 0000i, & 1000i & 0111i, & 0111i & 1000i \end{array} \}$$

Функцію F_{A_i} відображення множини станів S_i в множину підмножин множини T_{A_i} задамо таблично фрагментом(табл. 2).

Таблиця 2

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

Логічні значення властивостей $p_{i,j}$ у вершинах графу з рис. 8 та множини елементів, які породжуватимуть переходи та впливатимуть на процес переходів з вершини у вершину, частково представлені в таблиці 3.

Таблиця 3

Логічні значення властивостей та множини елементів

Перехід зі стану	A_i	До станів програмного модуля									Множини змінних	
		$s_{i,0}$	$s_{i,1}$	$s_{i,2}$	$s_{i,3}$	$s_{i,4}$	$s_{i,5}$	$s_{i,6}$	$s_{i,7}$	$s_{i,8}$	V_i	V_i'
$s_{i,0}$	$p_{i,0}$	1	0	0	0	0	0	0	0	0	$v_{i,0,0}$	$v'_{i,0,0}$
	$p_{i,1}$	1	1	0	0	0	0	0	0	0	$v_{i,0,1}$	$v'_{i,0,1}$
	$p_{i,2}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,3}$	1	0	0	1	0	0	0	0	0	$v_{i,0,3}$	$v'_{i,0,3}$
	$p_{i,4}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,5}$	1	0	0	0	0	1	0	0	0	$v_{i,0,5}$	$v'_{i,0,5}$
	$p_{i,6}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,7}$	1	0	0	0	0	0	0	1	0	$v_{i,0,7}$	$v'_{i,0,7}$
	$p_{i,8}$	1	0	0	0	0	0	0	0	1	$v_{i,0,8}$	$v'_{i,0,8}$
…	…	…	…	…	…	…	…	…	…	…	…	…
$s_{i,8}$	$p_{i,0}$	1	0	0	0	0	0	0	0	0	$v_{i,0}$	$v'_{i,0}$
	$p_{i,1}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,2}$	1	0	1	0	0	0	0	0	0	$v_{i,8,2}$	$v'_{i,8,2}$
	$p_{i,3}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,4}$	1	0	0	0	1	0	0	0	0	$v_{i,8,4}$	$v'_{i,8,4}$
	$p_{i,5}$	0	0	0	0	0	0	0	0	0	-	-
	$p_{i,6}$	1	0	0	0	0	0	1	0	0	$v_{i,8,6}$	$v'_{i,8,6}$
	$p_{i,7}$	1	0	0	0	0	0	0	1	0	$v_{i,8,7}$	$v'_{i,8,7}$
	$p_{i,8}$	1	0	0	0	0	0	0	1	1	$v_{i,8,8}$	$v'_{i,8,8}$

В кожному програмному модулі здійснюються переходи між станами згідно табличної функції F_{A_i} за схемою зображеною на рис. 12.

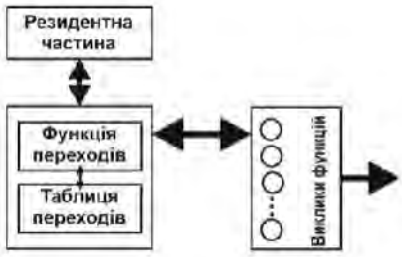

Рис. 12. Механізм переходів між станами ПМ

Функціонування РБС без будь-якого впливу ззовні системи, крім етапів установки системи адміністратором, додавання додатково її програмних модулів та перезапуску у випадку виникнення критичної події, що зупиняє роботу КС та системи. За умови виникнення певних визначених подій РБС призупиняє роботу процесів в КС та видає повідомлення користувачам і адміністратору. При виконанні основних поставлених завдань участь адміністратора та користувачів не передбачена.

Самовідтворення РБС здійснюється при початку роботи кожного з програмних модулів, додаванні кожного нового програмного модуля, видаленні певного програмного модуля. Ці події зберігають систему і дозволяють вирішувати поставлені на неї завдання. Цей рівень розгляду РБС представляє цю властивість як таку, що відноситься до існування системи. Описані події по організації спілкування між ПМ показують принцип утворення РБС та її самовідтворення в процесі життєвого циклу її частин. Мінімальна кількість працюючих КС, на яких встановлені 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 в складі РБС, тобто однієї ввімкненої КС, не дозволить використати можливості по 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 з станів, до базового стану. А також, РБС здійснюватиме аналіз статистики роботи ПМ КС для оптимізації передавання завдань з метою залучення до їх вирішення найбільш активних КС.

Формування РБС включатиме такі етапи:

1) при встановленні програмного модуля системи на кожну КС локальної мережі необхідна процедура первинної активації всієї системи, яка передбачатиме за вказівкою адміністратора мережі початок роботи і обмін інформацією між програмними модулями про всі КС, на які встановлена система, визначення

ідентифікатора кожного програмного модуля та прописування цих даних у внутрішні бази програмних модулів, що крім характеристик КС також включатиме і час їх початку роботи; активація кожного програмного модуля системи відбуватиметься не в одну мить, а в різні часові проміжки, зокрема і збір характеристик конкретних КС, що впливатиме на початок роботи всієї системи, тому поки не буде отримано на кожній КС підтвердження про завершення встановлення всіх програмних модулів системи, то робота система не розпочинається; після введення відповідної вказівки адміністратором програмні модулі і система в цілому розпочинають автономну роботу; така послідовність дій при активізації виділяє наступні події: запуск програмного модуля в КС, введення логіна і пароля адміністратора, зчитування характеристик системи, формування ідентифікатора програмного модуля, прописування характеристик та ідентифікатора у внутрішню базу, відправка адміністратором від КС першого пакету на інші КС, отримання пакетів від інших програмних модулів, перевірка адміністратором отримання пакетів зі всіх визначених КС та вказівка-підтвердження адміністратором кожному модулю про завершення етапу активації системи; події цього етапу вважатимемо стартовим станом; при цьому в пакеті відображається відповідний ідентифікатор, що сигналізує про першу активацію системи адміністратором;

2) доповнення системи новими програмними модулями здійснюватиметься за допомогою адміністратора безпосередньо з тієї КС, яка додається в локальну мережу (нова або була на ремонті), при цьому в пакеті відображається відповідний ідентифікатор, що сигналізує про додавання в систему адміністратором ще однієї КС; ця подія заставляє інші програмні модулі системи після отримання вказівки долучити цей програмний модуль і надіслати пакет про свої данні, після отримання пакетів від всіх програмних модулів адміністратор активує новий програмний модуль, який стає частиною системи, при цьому всі решта модулів створюють нову сторінку в своїй базі і прописують оновленні данні про структуру системи та використовують їх в подальшій роботі; при збереженні даних про структуру системи враховується час активації кожного програмного модуля;

3) вилучення КС з локальної мережі призводить до втрати ПМ системи; виявлення цієї події іншими ПМ може відбутись при коректному вимкненні КС, тоді ПМ цієї передає пакет з відповідним ідентифікатором; якщо ж відбувся аварійний вихід КС, тоді інформація про відсутність частини системи переданою не буде, а виявиться під час спроб інших ПМ системи спілкуватися з цим ПМ.

Програмні модулі РБС при перебуванні на третьому і четвертому рівнях для вирішення поставлених завдань залучатимуть технології самонавчання, що впливатиме на зміну їх роботи при вирішенні таких же завдань на наступних етапах життєвого циклу; при цьому вони, передаючи завдання для обробки на інші ПМ, та результати їх виконання впливатимуть на інші програмні модулі; в цілому 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, здійсненню аналізу наявності ПМ, які тривалий час перебувають в одному й тому ж стані та потребують автоматичного зняття поточних завдань з виконання та переведення до іншого стану, обробка баз ПМ для оптимізації та розподіл ПМ на декілька груп згідно аналізу їх станів протягом тривалого проміжку часу.

Важливим елементом самоорганізації РБС є розробка в ній механізмів для утворення своїх власних цілей. До таких цілей віднесемо такі: динамічне формування системи; розподіл та співвіднесення всіх структурних частин за групами завантаженості, обробка критичних подій в системі, колективне виконання завдань, розв’язуване одним ПМ, обробка та оптимізація накопичених статистичних даних.

Основою побудованої моделі РБС є її структурні частині, 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, данні для яких формуватимуться з використанням технологій штучного інтелекту.

Висновки. Отримані принципи та моделі розробки систем є важливим для теорії і практики створення ефективних систем виявлення шкідливого програмного забезпечення в локальних комп’ютерних мережах, побудованих на основі принципів децентралізації та самоорганізації.

Спроектована архітектура розподіленої багаторівневої системи базується на принципах децентралізації [6] та самоорганізації і дозволяє здійснювати її наповнення різними функціоналами виявлення шкідливого програмного забезпечення в локальних комп’ютерних мережах. РБС відноситься до реагуючих систем, яка постійно здійснюватиме моніторинг запущених процесів та виконуваних програм в комп’ютерних системах мережі. Об’єктами для дослідження зі сторони РБС буде перевірка наявного програмного забезпечення та запущених процесів в КС локальної мережі на можливість віднесення 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, але при цьому кожен з них може самостійно примати рішення на основі різних даних зібраних з різних КС мережі. Для ефективної роботи РБС необхідним є розробка методів і моделей взаємодії та узгодження роботи різних

програмних модулів між собою на їх відповідних рівнях і деталізація структури її станів.

Література

1. Komar M. High performance adaptive system for cyber attacks detection / M. Komar, V. Kochan, L. Dubchak, A. Sachenko, V. Golovko, S. Bezobrazov, I. Romanets // Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 21–23 September, 2017: Bucharest, Romania, 2017. – Vol. 2. – PP. 853–858.

2. Komar M. Intelligent Cyber Defense System Using Artificial Neural Network and Immune System Techniques / M. Komar, A. Sachenko, S. Bezobrazov, V. Golovko // Communications in Computer and Information Science. – 2017. – Vol. 783. – PP. 36–55. – ISSN 1865-0929, ISBN 978-3-319-69964-6.

3. Golovko V. Evolution of Immune Detectors in Intelligent Security System for Malware Detection / V. Golovko, S. Bezobrazov, V. Melianchuk, M. Komar // Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2011), Prague (Czech Republic), 2011. – Vol. 2. – P. 722–726.

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

5. Котенко И.В. Система сбора, хранения и обработки информации и событий безопасности на основе средств Elastic Stack / И.В. Котенко, А.А. Кулешов, И.А. Ушаков // Труды СПИИРАН. – 2017. – № 5(54). – С. 5–34.

6. Пат. на корисну модель 108238 Україна, МПК G06F 21/55. Мультиагентний спосіб локалізації бот-мереж у корпоративних комп'ютерних мережах / Поморова О.В., Савенко О.С., Крищук А.Ф., Лисенко С.М., Бобровнікова К.Ю., Нічепорук А.О. ; власник Хмельницький національний університет. – № u201600127 ; заявл. 04.01.2016 ; опубл. 11.07.2016, Бюл. № 13/2016.

References

1. Komar M. High performance adaptive system for cyber attacks detection / M. Komar, V. Kochan, L. Dubchak, A. Sachenko, V. Golovko, S. Bezobrazov, I. Romanets // Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 21–23 September, 2017: Bucharest, Romania, 2017. – Vol. 2. – PP. 853–858.

2. Komar M. Intelligent Cyber Defense System Using Artificial Neural Network and Immune System Techniques / M. Komar, A. Sachenko, S. Bezobrazov, V. Golovko // Communications in Computer and Information Science. – 2017. – Vol. 783. – PP. 36–55. – ISSN 1865-0929, ISBN 978-3-319-69964-6.

3. Golovko V. Evolution of Immune Detectors in Intelligent Security System for Malware Detection / V. Golovko, S. Bezobrazov, V. Melianchuk, M. Komar // Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2011), Prague (Czech Republic), 2011. – Vol. 2. – R. 722–726.

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

5. Kotenko I.V. Sistema sbora, hranenija i obrabotki informacii i sobytij bezopasnosti na osnove sredstv Elastic Stack / I.V. Kotenko, A.A. Kuleshov, I.A. Ushakov // Trudy SPIIRAN. – 2017. – № 5(54). – S. 5–34.

6. Pat. na korisnu model' 108238 Ukraïna, MPK G06F 21/55. Mul'tiagentnij sposib lokalizaciï bot-merezh u korporativnih komp'juternih merezhah / Pomorova O.V., Savenko O.S., Krishhuk A.F., Lisenko S.M., Bobrovnikova K.Ju., Nicheporuk A.O. ; vlasnik Hmel'nic'kij nacional'nij universitet. – № u201600127 ; zajavl. 04.01.2016 ; opubl. 11.07.2016, Bjul. № 13/2016.

Рецензія/Peer review : 09.02.2018 р. Надрукована/Printed :28.03.2018 р.
Стаття рецензована редакційною колегією