МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки

Кафедра Інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка додатку для виявлення шкідливого програмного коду

Виконав студент 2 курсу групи МІС-19
спеціальності 122 Комп'ютерні науки
Пінтяк Олександр Сергійович

Керівник к.т.н, доцент
Гнатовська Ганна Арнольдівна

Рецензент к.геогр.н., доцент
Кузніченко Світлана Дмитрівна

Одеса 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки
Кафедра Інформаційних технологій
Рівень вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
_____
" 26 " жовтня 2020 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Пінтяку Олександру Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка додатку для виявлення шкідливого програмного коду

керівник роботи Гнатовська Ганна Арнольдівна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від " 16" жовтня 2020р. №194 «С»

2. Строк подання студентом роботи 7 грудня 2020р

3. Вихідні дані до роботи Ресурси мережі Інтернет, методи виявлення шкідливих програм, алгоритм зворотного поширення помилки, мова програмування С++, середовище Microsoft Visual Studio

4. Зміст розрахунково -пояснювальної записки (перелік питань, які потрібно розробити) Аналіз та дослідження класів вірусів та антивірусних програм . Огляд методів виявлення шкідливих програм. Застосування штучної нейронної мережі в завданнях пошуку вірусів. Розробка додатка для виявлення шкідливого програмного коду. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 26 жовтня 2020р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
	Аналіз та дослідження класів вірусів та антивірусних програм	26.10.20 – 30.10.20	96%	відмін.
	Визначення основних ознак появи вірусів. Постановка задачі	31.10.20 – 03.11.20	96%	відмін.
	Огляд методів виявлення шкідливих програм	04.11.20 – 07.11.20	95%	відмін.
	Застосування штучної нейронної мережі в завданнях пошуку вірусів	08.11.20 – 11.11.20	94%	відмін.
	Розробка додатка для виявлення шкідливого програмного коду	12.11.20 – 01.12.2020	94%	відмін.
	Рубіжна атестація	19.11.20	95%	відмін.
	Тестування антивірусу на прикладі сигнатури bat-вірусу	01.12.20 – 03.12.20	95%	відмін.
	Оформлення пояснювальної записки та презентації	03.12.20 – 06.12.20	95%	відмін.
	Подання роботи на кафедру	07.12.20		
	Перевірка на плагіат	08.12.20		
	Рецензування	16.12.20		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)		95%	відмін.

Студент _____ Пінтяк О.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гнатовська Г.А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Тема магістерської роботи «Розробка додатку для виявлення шкідливого програмного коду».

Актуальність магістерської роботи полягає в застосуванні сучасних методів сигнатурного аналізу та евристичного аналізу на основі багатошарової нейронної мережі для розв'язання завдань пошуку та знешкодження шкідливого програмного забезпечення, що забезпечить ефективний механізм захисту програмного забезпечення.

Об'єкт дослідження – процеси проектування та розробки програмного додатку для пошуку і знешкодження шкідливого програмного коду, заснованих на методах сигнатурного аналізу та евристичного аналізу.

Предмет дослідження – методи сигнатурного аналізу та евристичного аналізу для розв'язання завдань пошуку та знешкодження шкідливого коду.

Мета роботи – розробка програмного додатку для виявлення шкідливого програмного коду, що забезпечує, використовуючи методи сигнатурного та евристичного аналізу, ефективний пошук та знешкодження вірусів.

В роботі було проведено дослідження класів існуючих вірусів та огляд сучасних антивірусних програм, здійснено дослідження і вибір методів виявлення шкідливих програм, виконано моделювання евристичного аналізатора шкідливих програм на основі багатошарової нейронної мережі, розроблено додаток для виявлення шкідливого програмного коду. Практична цінність роботи полягає в розробці додатка, який забезпечує виявлення та знешкодження шкідливого програмного коду, що може стати першою ланкою в створенні антивірусного програмного забезпечення в подальшому.

Ключові слова: ШКІДЛИВИЙ ПРОГРАМНИЙ КОД, СИГНАТУРНИЙ АНАЛІЗ, ЕВРИСТИЧНИЙ АНАЛІЗ, НЕЙРОННА МЕРЕЖА.

Магістерська робота містить 94 сторінки, 1 таблицю, 42 рисунки, 20 посилань.

ANNOTATION

The theme of master's work is "Development of an application to detect malicious programming code".

The relevance of the master's thesis is the application of modern methods of signature analysis and heuristic analysis based on a multilayer neural network to solve problems of finding and neutralizing malware, which will provide an effective mechanism for software protection.

The object of research is the processes of search and neutralization of malicious program code based on the methods of signature analysis and heuristic analysis based on a multilayer neural network.

Subject of research is methods of signature analysis and heuristic analysis to solve problems of search and disposal of malicious code.

The purpose of the work is to develop a software application for detecting malicious software that provides, using the methods of signature and heuristic analysis, effective search and disposal of viruses.

The study of existing viruses and review of modern antivirus programs, research and selection of malware detection methods, simulation of a heuristic malware analyzer based on a multilayer neural network, developed an application to detect malicious code. The Windows operating system is chosen as the target platform.

The practical value of the work is to develop an application that detects and neutralizes malicious software, which can be the first link in the creation of powerful anti-virus software in the future.

Key words: HARMFUL SOFTWARE CODE, SIGNATURE ANALYSIS, HEURIST ANALYSIS, NEURAL NETWORK.

The master's thesis contains 94 pages, 1 table, 42 figures, 20 references.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ОС – операційна система

ПЗ – програмне забезпечення

ШП – шкідливі програми

Boot viruses – завантажувальний вірус

CRC32, Cyclic redundancy code – циклічний надлишковий код, використовується в роботі програм-архіваторів

HEX – шістнадцятирічна система числення – це позиційна система числення по цілочисельній основі 16

HIPS (Host Intrusion Prevention System) – поведінковий блокатор

MD5, Message Digest 5 – 128-бітний алгоритм хешування

RTF, Rich Text Format – вільний міжплатформений формат зберігання розмічених текстових документів

SHA1, Secure Hash Algorithm 1 – алгоритм криптографічного хешування

Stealth virus – вірус-невидимка

ВСТУП

У зв'язку з масовим застосуванням комп'ютерів у різноманітних областях з'явилася величезна кількість комп'ютерних вірусів – програми, які здатні зіпсувати або знищити інформацію, що зберігається в комп'ютері, знищити файлову структуру дисків або просто попсувати нерви користувачеві.

Комп'ютерний вірус – комп'ютерна програма, яка має здатність до прихованого самопоширення. Одночасно зі створенням власних копій віруси можуть завдавати шкоди: знищувати, пошкоджувати, викрадати дані, знижувати або й зовсім унеможливлювати подальшу працездатність операційної системи комп'ютера. Розрізняють файлові, завантажувальні та макро-віруси. Можливі також комбінації цих типів. Нині відомі десятки тисяч комп'ютерних вірусів, які поширюються через мережу Інтернет по всьому світу. Розробники вірусного програмного забезпечення використовують засоби соціальної інженерії і інформацію про вразливості цільового ПЗ щоб заражувати системи і розповсюджувати вірус. Необізнані користувачі ПК помилково відносять до комп'ютерних вірусів також інші види зловмисного ПЗ – програм-шпигунів. Кожного року комп'ютерні віруси причиняють шкоди розміром в декілька мільярдів доларів, викликаючи системні критичні помилки, зупиняючи великі сайти та веб-додатки, знищуючи або модифікуючи файли, підвищуючи час відклику [1] [1)].

Відомо більше 5000 комп'ютерних вірусів, і їх число постійно зростає. Відомі випадки, коли віруси блокували роботу цілих організацій. Зараз існують апаратні і програмні засоби захисту від вірусів, але, вони не дають повного захисту. Дуже часто знання користувачів (іноді і програмістів) щодо вірусів дуже поверхневі, а необхідні навички повністю відсутні. Основною властивістю вірусу є його здатність до самовідтворення. Але крім вірусів

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

таку особливість мають багато цілком мирних програм (наприклад, операційна система). Крім того, вірус повинен яким-небудь чином забезпечити передачу керування собі для того, щоб виконати свою основну задачу. 21 сторіччя принесло, мабуть, самі великі зміни в індустрію антивірусів: замість терміна «вірус» на зміну прийшло поняття «шкідлива програма».

Для одних віруси є бізнесом. Причому не тільки для їхніх авторів, але і для тих, хто з цими вірусами бореться. Бо процвітання компаній, які випускають антивірусні програми не є несподіванкою ні для кого. Для інших – це хобі. Хобі – збирання вірусних колекцій і хобі – написання вірусів. Ще інші – створюють віруси для прояву власної зухвалості і незалежності, у деяких колах подібна діяльність просто необхідна для підняття свого престижу. Є й такі, для кого віруси це витвір мистецтва; зустрічаються лікарі за покликанням, отже, може бути і комп'ютерний лікар за покликанням. Для деяких віруси служать приводом пофілософствувати на теми створення і розвитку комп'ютерного життя. Для інших віруси – це також стаття кримінального кодексу. Але для більшості користувачів комп'ютерів віруси – це щоденний головний біль, причина збоїв у роботі комп'ютера і ворог номер один. Крім цих програм з'явилися ще і інші, які при деяких умовах можуть завдати шкоди (наприклад, відкрити доступ до вашого комп'ютера і даних зловмисникові), передати персональні дані користувача зловмисникові або показувати непрошену рекламу. Шкідлива програма може підписати користувача на поштові розсилання, зробити покупку в Інтернет-магазині і зробити будь-які інші аналогічні дії. Тому кожний комп'ютер повинен бути захищений за допомогою антивірусного програмного забезпечення, здатного виявити ці віруси і вилучити їх з вашого комп'ютера [1][1)].

Антивірусна програма (антивірус) – спеціалізована програма для знаходження комп'ютерних вірусів, а також небажаних (шкідливих) програм

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

загалом, та відновлення заражених (модифікованих) такими програмами файлів, а також для профілактики – запобігання зараженню (модифікації) файлів чи операційної системи шкідливим кодом [2][2)].

Кожного року комп'ютерні віруси причиняють шкоди розміром в декілька мільярдів доларів, викликаючи системні критичні помилки, зупиняючи великі сайти та веб-додатки, знищуючи або модифікуючи файли, підвищуючи час відклику. Тому розробка програмного забезпечення, для виявлення шкідливого програмного коду, яке засновано на використанні сучасних методів виявлення вірусів, є актуальним завданням.

Об'єкт дослідження – процеси пошуку і знешкодження шкідливого програмного коду, заснованих на методах сигнатурного аналізу та евристичного аналізу на основі багатошарової нейронної мережі.

Мета роботи – розробка програмного додатку для виявлення шкідливого програмного коду, що забезпечує, використовуючи методи сигнатурного та евристичного аналізу, ефективний пошук та знешкодження вірусів.

Практична цінність роботи полягає в розробці додатка, який забезпечує виявлення та знешкодження шкідливого програмного коду, що може стати першою ланкою в створенні потужного антивірусного програмного забезпечення в подальшому.

[2)] [2] Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. ДМК Пресс, 2012, 592 с.

1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ КЛАСІВ ВІРУСІВ ТА АНТИВІРУСНИХ ПРОГРАМ

Комп'ютерний вірус – це різновид комп'ютерних програм, які мають здатність до самореплікація (розмноження). Віруси можуть пошкодити або повністю знищити усі дані, підконтрольні користувачеві, від імені якого була запущена заражена програма, а також ушкодити або знищити операційну систему з усіма файлами. Якщо заражений комп'ютер підключений до мережі, вірус може передати дані та файли небажаним особам, крім того, інфікована машина може використовуватися для атак на інші системи. Вірус, що поширює свої копії по мережі – це мережевий хробак. У цей час не існує єдиної системи класифікації і визначення вірусів і шкідливого ПЗ [3][1)].

Віруси можна розділити на класи за наступними ознаками:

- середовище проживання;
- операційна система;
- особливості алгоритму роботи;
- деструктивні можливості.

Прийнято розділяти віруси за об'єктами, що розглядаються (файлові віруси, завантажувальні віруси, скриптові віруси, макро-віруси, мережні хробаки), за операційними системами, що розглядаються, і платформами (DOS, Microsoft Windows, Unix, Linux, Macos), за технологіями, використовуваними вірусом (поліморфні віруси, стелс-віруси), за мовою, якою написаний вірус (асемблер, високорівнева мова програмування, скриптова мова і ін.). За способом зараження віруси можна розділити на резидентні і нерезидентні. Резидентний вірус після проникнення в комп'ютер записує в оперативну пам'ять спеціальну програму (резидентна частина). Це програма-перехоплювач для звернення операційної системи до деяких певних об'єктів (завантажувальних секторів дисків, документів, і т.п.). Після

[1)] [3] Ф.Файтс, П.Джонстон, М.Кратц. Компьютерный вирус: проблемы и прогноз. М., Мир, 1993, 176 с.

одержання такого звернення програма впроваджується в даний об'єкт. Вилучити резидентні віруси з оперативної пам'яті можна тільки за допомогою перезавантаження комп'ютера. До резидентних також слід віднести, наприклад, макро-віруси, оскільки вони активні, поки працює редактор, і видаляються з пам'яті після виходу з нього. Якщо вірус залишає в пам'яті резидентну програму, але ця програма його не поширює, його вважають нерезидентним. Звичайно нерезидентним називають вірус, який не заражає оперативну пам'ять і активний тільки протягом деякого обмеженого часу після появи [4][1)].

При використанні стелс-алгоритмів вірус може повністю або частково сховати себе в системі. Такі віруси дуже важко виявити, тому що вони перехоплюють звернення ОС до заражених об'єктів і підставляють замість себе незаражені ділянки диска. Один з перших файлових стелс-вірусів – вірус «Frodo», перший завантажувальний стелс-вірус «Brain». Дуже багато вірусів використовують самошифрування і поліморфічність. Це робиться для ускладнення процедури детектування вірусу. Ці віруси досить важко виявити, тому що вони містять алгоритми шифрування-розшифрування, внаслідок чого копії того самого вірусу не мають жодного повторюваного ланцюжка байтів. За деструктивними можливостями віруси можна розділити:

- нешкідливі, тобто такі, що зовсім не впливають на роботу комп'ютера (крім зменшення вільної пам'яті на диску);
- безпечні, вплив яких обмежується зменшенням вільної пам'яті на диску і різними ефектами (графічними, звуковими і т.д.);
- небезпечні, які здатні привести до збоїв у роботі комп'ютера;
- дуже небезпечні, в алгоритм роботи яких свідомо закладені процедури, що приводять до втрати програм, знищенню даних і т.д.

[1)] [4] Класифікація комп'ютерних вірусів - Поняття та класифікація комп’ютерних вірусів. Програмні засоби захисту від комп’ютерних вірусів. Класифікація антивірусів. URL: https://sites.google.com/site /siteallaboutviruses /klasifikacia-komp-uternih-virusiv (дата звернення 24.05.2020)

Навіть якщо вірус не має небезпечних процедур, він є програмною помилкою. Наприклад, дотепер зустрічаються віруси, що визначають "COM" або "EXE" не за внутрішнім форматом файлу, а за його розширенням.

1.1 Основні класи вірусів

На сьогоднішній день відомі десятки тисяч різних комп'ютерних вірусів. Незважаючи на такий достаток, число типів вірусів, що відрізняються один від одного механізмом розповсюдження і принципом дії, досить обмежена. Існують і комбіновані віруси, які можна віднести одночасно до декількох типів. Основною і найбільш поширеною класифікацією комп'ютерних вірусів є класифікація за середовищі проживання, або за типами об'єктів комп'ютерної системи, в які впроваджуються віруси [5][1)].

1.1.1 Завантажувальні віруси

Завантажувальні, або бутові – віруси активуються у момент завантаження системи. Для цього їм потрібно розташувати частину свого коду в службових структурах носія, з яко-го відбувається завантаження – жорсткого диска або дискети.

Зараження дискети здійснюється записуванням коду вірусу замість оригінального коду boot-сектора дискети. Зараження жорсткого диска відбувається в один із трьох способів: вірус записує себе замість коду MBR (Master Boot Record – головний завантажуваль-ний запис, таблиця у першому секторі завантажувального диска) чи замість коду boot-сектора завантажувального диска (у Windows – це, як правило, диск C) або модифікує адресу активного boot-сектора в таблиці розділів диска (Disk Partition Table),

---

[1)] [5] Романец Ю. В., Тимофеев П. А., Шаньгин В. Ф. Защита информации в компьютерных системах и сетях. М., Радио и связь, 2001, 376 с.

що знаходиться в MBR. Завантаження з дискети вже відійшло у минуле, позаяк сучасні операційні системи вимагають більші об'єми носіїв для розміщення свого коду. Саме через те, бутові віруси в наш час менш поширені. На жорсткому диску бутові віруси можуть вельми спокійно існувати і завдавати величезної шкоди інформаційним ресурсам. Вони також можуть дуже ефективно протидіяти антивірусним засобам, оскільки саме віруси стартують першими, ще до запуску операційної системи, і тому вони здатні залишити за собою керування критичними для їх існування ресурсами комп'ютера, зокрема, файловою системою. З іншого боку, для цього потрібно фактично утворити для ОС віртуальну машину, що для сучасних систем хоча і є можливим, але потребує великого обсягу програмного коду, що не дуже прийнятно для вірусу [1][1)].

Завантажувальні віруси дуже рідко «уживаються» разом на одному диску через те, що використовують ті самі дискові сектори для розміщення свого коду/даних. У результаті код/дані першого вірусу виявляються зіпсованими при зараженні другим вірусом, і система або зависає, або зациклюється при завантаженні. Вірус у завантажувальному записі представлено на рис.1.1.

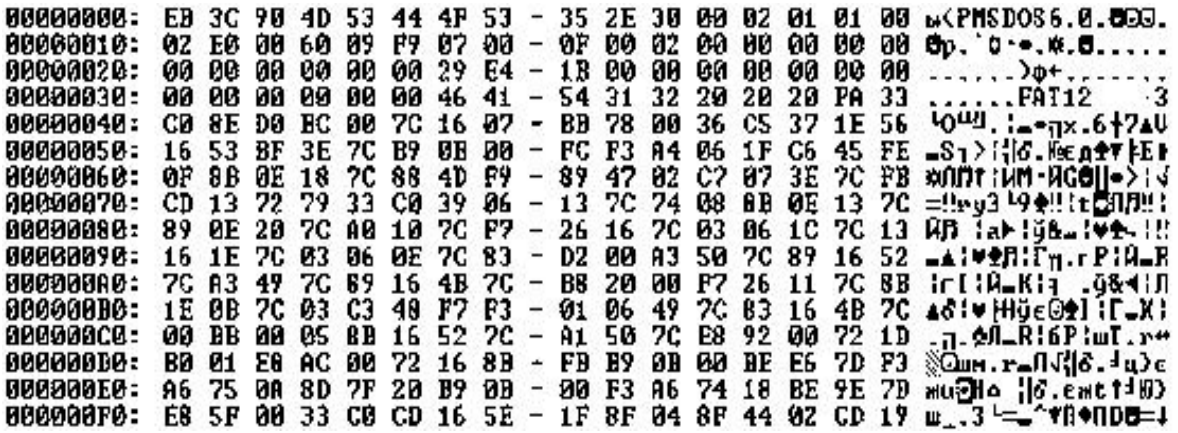

Рисунок 1.1 – Вірус у завантажувальному записі

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

1.1.2 Файлові віруси

Файлові віруси – тип комп'ютерних вірусів, що розмножуються використовуючи файлову систему шляхом запису свого коду в код виконуваного файлу конкретної операційної системи. До потенційно інфікованих типів файлів належать бінарні файли .exe та .com; файли динамічних бібліотек .dll; драйвери .sys; командні файли .bat, .cmd тощо. За способом зараження файлові віруси розділяють на перезаписуючі, паразитичні, віруси-ланки, віруси-хробаки, віруси-компань-йони. На відміну від завантажувальних вірусів, які практично завжди є резидентними, файлові віруси зовсім не обов'язково є резидентними. Розглянемо схему функціонування нерезидентного файлового вірусу. Нехай у нас є інфікований файл, що виконується. Під час запуску такого файлу вірус одержує керування, робить деякі дії і передає керування «хазяїнові» [1][1)].

Він шукає новий об'єкт для зараження – підходящий за типом файл, який ще не заражений. Заражаючи файл, вірус впроваджується в його код, щоб одержати керування при запуску цього файлу. Якщо файловий вірус резидентний, то він установиться на запам'ятовування і одержить можливість заражати файли і проявляти інші здатності не тільки під час роботи зараженого файлу. Заражаючи файл, що виконується, вірус завжди змінює його код – отже, зараження файлу, що виконується, завжди можна виявити. Але, змінюючи код файлу, вірус не обов'язково вносить інші зміни:

- він не зобов'язаний змінювати довжину файлу;
- невикористовувані ділянки коду;
- не зобов'язаний змінювати початок файлу.

Нарешті, до файлових вірусів часто відносять віруси, які «мають деяке відношення до файлів», але не зобов'язані впроваджуватися в їхній код.

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

Таким чином, при запуску будь-якого файлу вірус одержує керування (операційна система запускає його сама), резидентно встановлюється на запам'ятовування і передає керування викликаному файлу. На рис.1.2 показана дія вірусу у файлі mouse.com.

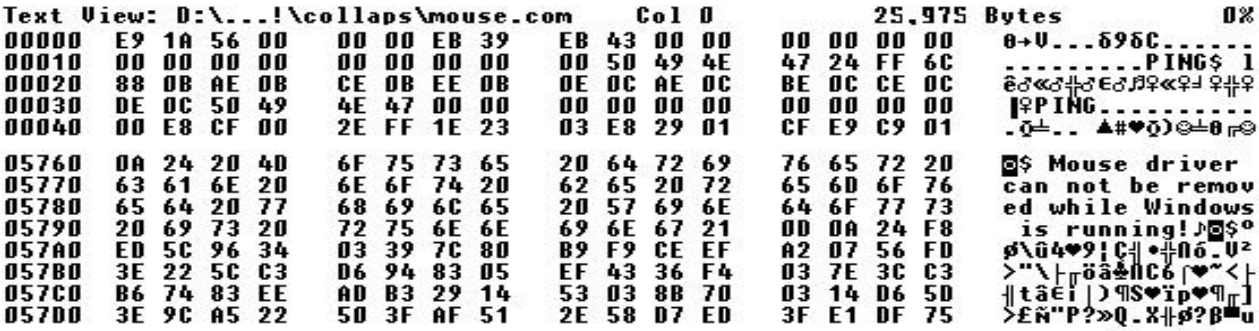

Рисунок 1.2 – Вірус у файлі mouse.com

1.1.3 Поліморфні віруси

Більшість питань пов'язана з терміном «поліморфний вірус». Цей вид комп'ютерних вірусів представляється на сьогоднішній день найнебезпечнішим.

Поліморфні віруси – віруси, що модифікують свій код у заражених програмах таким чином, що два екземпляри того самого вірусу можуть не збігатися в жодному біті. Такі віруси не тільки шифрують свій код, використовуючи різні шляхи шифрування, але і містять код генерації шифрувальника і розшифровувача, що відрізняє їх від звичайних шифрувальних вірусів, які також можуть шифрувати ділянки свого коду, але мають при цьому постійний код шифрувальника і розшифровувача. Цей код зашифрований і являє собою безглуздий набір команд. На рис.1.3 показаний список найпоширеніших поліморфних вірусів [1][1)].

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

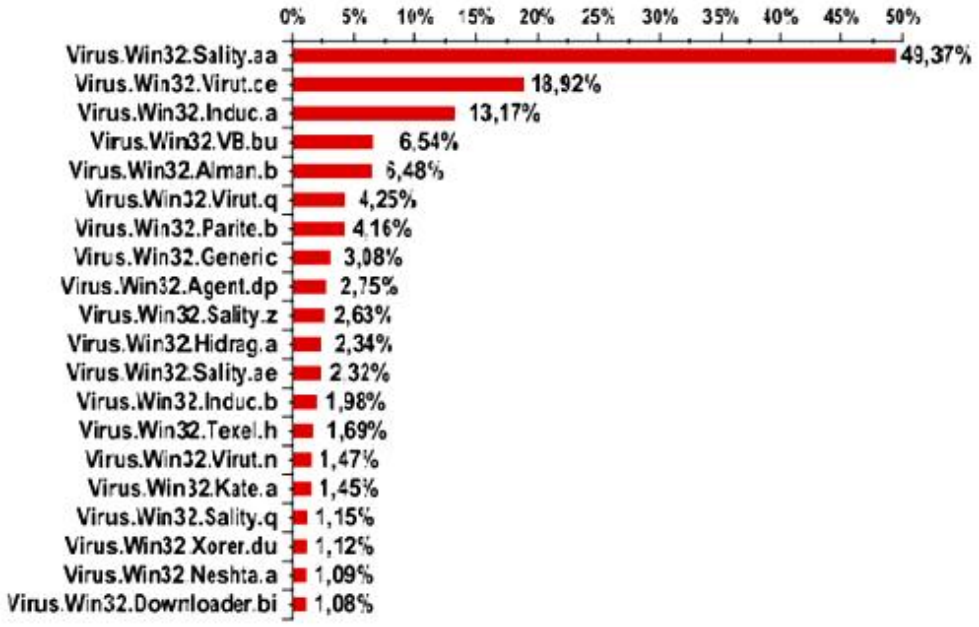

Рисунок 1.3 – Найпоширеніші поліморфні віруси

1.1.4 Стелс-віруси

Stealth virus – вірус, що повністю або частково приховує свою присутність у системі шляхом перехоплення звертань до операційної системи, що здійснюють читання, запис, читання додаткової інформації про заражені об'єкти (завантажувальних секторах, елементах файлової системи, пам'яті і т.ін.). Для пошуку stealth-вірусів рекомендується здійснити завантаження системи із гнучкого диска і провести видалення вірусних програм (перевстановлення системи – low форматування).

Антивіруси-поліфаги ефективні в боротьбі із уже відомими вірусами, тобто тими, чиї методи поведінки вже знайомі розроблювачам і є в базі програми. Якщо вірус невідомий, то він залишиться непоміченим. Головне в боротьбі з вірусами – як найчастіше обновляти версії програми і вірусні бази [1][1)].

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

1.1.5 Макро-віруси

Макро-віруси – програми на мовах (макро-мовах), вбудованих у багатьох системах обробки даних (текстові редактори, електронні таблиці і т.д.). Для свого розмноження такі віруси використовують можливості макро-мов і при їхній допомозі переносять себе з одного зараженого файлу (документа або таблиці) в інші. Найбільше поширення одержали макро-віруси для Micro-soft Word, Excel і Microsoft Outlook.

Розповсюдження макровірусів відбувається в три етапи.

На 1 етапі агент-користувач отримує від іншого користувача або переносить з комп'ютера на комп'ютер файл з макровірусом.

На 2 етапі він виконує із ним якісь дії, що призводять до автоматичного виконання відповідного цій дії службового макросу. Якщо службовий макрос у цьому файлі було підмінено макровірусом, то управління, несподівано та таємно для користувача, отримає саме код макровірусу. Найчастіше макровірус перевизначають макроси, що мають виконуватися при самому відкритті файлу для редагування або перегляду, тож код макровірусу у таких випадках виконується на самому початку роботи користувача із файлом.

На 3 етапі макровірус виконує функції з свого подальшого розповсюдження (пошук на комп'ютері користувача та зараження інших файлів, здатних виконувати макровірус) та/або деструктивні чи інші побічні функції.

Далі розглянемо причини зараження макро-вірусами.

Для існування вірусів у конкретній системі (редакторі) необхідна наявність вбудованої в систему макро-мови з такими можливостями:

- програми на макро-мові прив'язані до документів (AmiPro) чи зберігаються в них (додатки MS Office);
- у макро-мові присутні команди копіювання макро-програм з одного файла в іншій (AmiPro) або переміщати макро-програми у службові файли системи і файли, що редагуються (MS Office);

– є можливість одержання керування макро-програмою без втручання корис-ту-вача (автоматичні чи стандартні макроси), тобто при роботі з файлом за певних умов (відкриття, закриття і т.д.) викликаються макро-програми (якщо такі є), що визначені спеціальним чином (AmiPro) чи мають стандартні імена (MS Office).

Дані можливості макро-мов призначені для автоматичної обробки даних у великих організаціях чи у глобальних мережах і дозволяють організувати так званий "автоматизований документообіг". З іншого боку, можливості макро-мов таких систем дозволяють вірусу переносити свій код в інші файли, і в такий спосіб заражати їх. Більшість макро-вірусів можна вважати резидентними, оскільки вони присутні в області системних макросів протягом усього часу роботи редактора. Вони так само, як резидентні завантажувальні і файлові віруси, перехоплюють системні події і використовують їх для свого розмноження. До подібних подій відносяться різні системні виклики, що виникають при ро-боті з документами Word і таблицями Excel (відкриття, закриття, ство-рен-ня, печатка і т.д.), виклик пункту меню, натискання на яку-небудь кла-ві-шу чи досягнення певного моменту часу. Для перехоплення подій макро-віруси перевизначають один чи декілька системних макросів або функцій. При зараженні деякі макро-віруси перевіряють наявність своєї копії в об'єкті, що заражається, і повторно себе не копіюють. Інші макро-вируси не роблять цього і переписують свій код при кожному зараженні. Якщо при цьому у файлі, що заражається, чи області системних макросів уже визначений макрос, ім'я якого збігається з макросом вірусу, то такий макрос виявляється знищеним [1][1)].

1.1.6 Віруси-компаньйони

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

До категорії "компаньйон" відносяться віруси, які не змінюють фай-лів, що заражаються. Алгоритм роботи цих вірусів полягає в тому, що для файла, який заражається, створюється файл-двійник, причому при запуску зараженого файлу керування одержує саме цей двійник, тобто вірус. Найбільш поширені компаньйон-віруси, що використовують особ-ли-вість DOS першим виконувати.

COM-файл, якщо в одному каталозі при-сутні два файли з тим самим ім'ям, але різними расширеннями імені – .COM і .EXE. Такі віруси створюють для EXE-файлів файли-су-пут-ни-ки, що мають те ж саме ім'я, але з розширенням .COM, наприклад, для файлу XCOPY.EXE створюється файл XCOPY.COM. Вірус записується в COM-файл і ніяк не змінює EXE-файл. При запуску такого файлу DOS першим знайде і виконає COM-файл, тобто вірус, який потім запустить і EXE-файл. Деякі віруси використовують не тільки варіант COM-EXE, але також і BAT-COM-EXE.

Другу групу складають віруси, що при зараженні перейменовують файл у яке-небудь інше ім'я, запам'ятовують його (для наступного запуску файла-хазяїна) і записують свій код на диск під іменем файла, що за-ра-жа-ється. Наприклад, XCOPY.EXE перейменовується в XCOPY.EXD, а вірус записується під ім'ям XCOPY.EXE. При запуску керування одер-жує код вірусу, що потім запускає оригінальний XCOPY, що збері-га-єть-ся під ім'ям XCOPY.EXD. Цікавий той факт, що даний метод працює, напевно, у всіх операційних системах: в DOS, в Windows і OS/2. У третю групу входять так називані "Path-companion" віруси, що "гра-ють" на особливостях PATH.Вони або записують свій код під іменемзараженого файлу, але "вище" на один рівень PATH (ОС, таким чином, першим знайде і запустить файл-вірус), або переносять файл-жертву на один підкаталог вище і т.д [1][1)].

[1)] [1] Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.

1.1.7 Троянські коні, програмні закладки і хробаки

Троянський кінь (троян) – це програма, яка надає стороннім доступ до комп'ютера для здійснення будь-яких дій на місці призначення без попередження самого власника комп'ютера або висилає по певній адресі зібрану інформацію. Звичайно такі програми маскуються під які-небудь корисні утиліти. Віруси можуть нести в собі троянських коней або "троянизувати" інші програми – вносити в них руйнуючі функції. «Троянські коні» є програмами, що реалізують крім функцій, описаних у документації, і деякі інші функції, пов'язані з порушенням безпеки і деструктивними діями. Відзначені випадки створення таких програм з метою полегшення поширення вірусів. Списки таких програм широко публікуються в закордонній пресі. Звичайно вони маскуються під ігрові або розважальні програми і завдають шкоди під красиві картинки або музику. Троянські програми не можуть розповсюджуватися самостійно, тому використовують будь-яку з форм соціальної інженерії. Наприклад, коли користувач отримує електронного листа вкладення електронної пошти може бути замасковане, (наприклад, звичайна форма для заповнення) [3][2)].

Троянська програма може нести вірусне тіло – тоді запустив троянця комп'ютер перетворюється в осередок «зарази».

Для того, щоб спровокувати користувача запустити троянця, файл програми (його назва, іконку програми) називають службовим ім'ям, маскують під іншу програму (наприклад, установки іншої програми), файл іншого типу або просто дають привабливе для запуску назву, іконку і т.п. Зловмисник може перекомпілювати існуючу програму, додавши до її вихідного коду шкідливий, а потім видавати за оригінал або підміняти його.

Троянська програма, будучи запущеною на комп'ютері, може:

[2)] [3] Ф.Файтс, П.Джонстон, М.Кратц. Компьютерный вирус: проблемы и прогноз. М., Мир, 1993, 176 с.

- заважати роботі користувача (жартома, помилково або для досягнення якихось інших цілей);
- шпигувати за користувачем;
- використовувати ресурси комп'ютера для якої-небудь незаконної (а іноді і завдає прямої шкоди) діяльності і т.д.

Загалом, троянські програми виявляються та видаляються антивірусним і антишпигунським ПЗ так само, як і інші шкідливі програми. Проте їх складніше виявити контекстними методами антивірусів (заснованих на пошуку відомих програм), бо їх розповсюдження краще контролюється й екземпляри програм потрапляють до спеціалів антивірусної індустрії з більшою затримкою, ніж саморозповсюджувані шкідницькі програми. Однак еврістичні (пошук алгоритмів) і проактивні (стеження) методи на стільки ж дієві.

Щоб успішно виконувати ці функції, троян може в тій чи іншій мірі імітувати (або навіть повноцінно замінювати) задачу або файл даних, під які вона маскується (програма установки, прикладна програма, гра, прикладний документ, картинка). Схожі шкідливі і маскувальні функції також використовуються комп'ютерними вірусами, але на відміну від них, троянські програми не вміють поширюватися самостійно.

Якщо віруси і «троянські коні» завдають шкоди за допомогою лавиноподібного саморозмножування або явного руйнування, то основна функція вірусів типу «хробак», що діють у комп'ютерних мережах, – злом системи, що атакується, тобто подолання захисту з метою порушення безпеки і цілісності. У більш ніж 80% комп'ютерних злочинів, розслідуваних ФБР, "зломщики" проникають у глобальну мережу системи, що атакується, через Internet. Коли така спроба вдається, майбутнє компанії, на створення якої пішли роки, може бути поставлене під загрозу за деякі секунди. Цей процес може бути автоматизований за допомогою вірусу, що називається мережний хробак. Хробаками називають віруси, які поширюються по глобальних мережах, вражаючи цілі системи, а не окремі програми. Це самий

небезпечний вид вірусів, тому що об'єктами нападу в цьому випадку стають інформаційні системи державного масштабу.

Файлові хробаки створюють власні копії із привабливими для користувача назвами (наприклад, Game.exe, install.exe і ін.) у надії на те, що користувач їх запустить. Віруси-ланки, як і віруси-компаньйони, не змінюють код програми, а змушують операційну систему виконати власний код, змінюючи адресу місця розташування на диску зараженої програми на власну адресу. Після виконання коду вірусу звичайне керування передається програмі, яка викликається користувачем. Руткіт – вірус, що працює на рівні ядра ОС, як правило, вантажується разом з ОС, і на етапі роботи ОС маскується і стає невидимим для основних засобів антивірусного захисту. Один з найнебезпечніших типів вірусів. Паразитичні віруси – це файлові віруси, що змінюють зміст файлу, додаючи в нього свій код. При цьому заражена програма зберігає повну або часткову працездатність. Код може впроваджуватися в початок, середину або кінець програми. Код вірусу виконується перед, після або разом із програмою, залежно від місця впровадження вірусу в програму. Мережні хробаки – це шкідливе ПЗ, що попадає на комп'ютер за рахунок помилок у мережному програмному забезпеченні [3][1)].

1.2 Визначення основних ознак появи вірусів

Сьогодні відомі десятки тисяч комп‘ютерних вірусів, які поширюються через Інтернет по всьому світу. Малообізнані користувачі ПК помилково відносять до комп‘ютерних вірусів також інші види шкідливих програм — програми-шпигуни чи навіть спам. При зараженні комп'ютера вірусом

[1)] [3] Ф.Файтс, П.Джонстон, М.Кратц. Компьютерный вирус: проблемы и прогноз. М., Мир, 1993, 176 с.

важливо його знайти. Для цього треба знати про основні ознаки прояву вірусів Необхідно знати основні ознаки прояви вірусів [6][1)]:

- зменшення вільної пам'яті;
- уповільнення роботи комп'ютера;
- затримки при виконанні програм;
- незрозумілі зміни в файлах;
- зміна дати модифікації файлів без причини;
- незрозумілі помилки Write-protection.

Помилки при інсталяції і запуску ОС:

- відключення 32-розрядного допуску до диску;
- неспроможність зберігати документи Word в інші каталоги, крім Template;
- погана робота дисків;
- файли невідомого походження;
- збій системи.

Ранні ознаки зараження дуже тяжко виявити, але коли вірус переходить в активну фазу, тоді легко помітити такі зміни:

- зникнення файлів;
- форматування HDD;
- неспроможність завантажити комп'ютер;
- неспроможність завантажити файли;
- незрозумілі системні повідомлення, звукові ефекти і т. д.

Ці явища не обов'язково викликаються вірусом, а можуть бути наслідком інших причин. Тому правильна діагностика стану комп'ютера завжди утруднена.Здебільшого, все це в минулому. Зараз основні ознаки – самовільне відкривання браузером деяких сайтів (рекламного характеру), підозріло підвищений інтернет-трафік та повідомлення від друзів, що ваші

[1)] [6] Ознаки зараження комп’ютерними вірусами та як цьому запобігти | Безпечне місто. URL: http://safe-city.com.ua/oznaky-zarazhennya-komp-yuternymy-virusamy-ta-yak-tsomu-zapobigty/ (дата звернення 02.06.2020)

листи електронної пошти до них містили вірус. Розглянувши всю різноманітність існуючих вірусів, перейдемо до методів боротьби з ними [7][1]. Основні способи протидії:

- профілактика заражень;
- знешкодження відомого вірусу;
- знешкодження невідомого вірусу.

1.3 Огляд та дослідження антівірусних программ

На сьогоднішній день вибір надійного антивіруса – справа першочергової важливості для будь-якого користувача, адже що може захистити комп'ютер краще, ніж повноцінне комплексне рішення, спрямоване на збереження даних і стабільну роботу? Ця частина присвячена антивірусним програмам, які дадуть користувачеві всі необхідні інструменти для захисту від троянів і вірусів, допоможуть запобігти зараженню комп'ютера, заблокують шкідливий сайт і будуть не сильно завантажувати систему. Класифікувати антивірусні продукти можна відразу за кількома ознаками, таким, як: використовувані технології антивірусного захисту, функціонал продуктів, цільові платформи. По використовуваних технологіях антивірусного захисту [8][2]:

- класичні антивірусні продукти (продукти, які застосовують тільки сигнатурний метод детектування);
- продукти проактивного антивірусного захисту (продукти, які застосовують тільки проактивні технології антивірусного захисту);
- комбіновані продукти (продукти, які застосовують як класичні, сигнатурні методи захисту, так і проактивні).

---

[1] [7] Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах. М., ДМК Пресс, 2002, 596 с.

[2] [8] Рейтинг антивирусов 2020. Какой антивирус лучше? URL: https://softcatalog.info/ru/obzor/reyting-antivirusov (дата звернення 03.06.2020)

За функціоналом продуктів:

- антивірусні продукти (продукти, що забезпечують тільки антивірусний захист);
- комбіновані продукти (продукти, що забезпечують не тільки захист від шкідливих програм, але і фільтрацію спаму, шифрування та резервне копіювання даних та інші функції);
- за цільовими платформами: Windows, * NIX (до даного сімейства відносяться ОС BSD, Linux, etc), MacOS, для мобільних платформ (Windows Mobile, Symbian, iOS, BlackBerry, Android, Windows Phone та ін.).

Антивірусні продукти для корпоративних користувачів можна також класифікувати по об'єктах захисту:

- для захисту робочих станцій;
- для захисту файлових і термінальних серверів;
- для захисту поштових та Інтернет-шлюзів;
- для захисту серверів віртуалізації.

Нижче ми розглянемо шість найпопулярніших антивірусів: Avast Free Antivirus, Panda Antivirus Pro, AVG Anti-Virus Free, ESET NOD32 Smart Security, Dr.Web Antivirus.

Avast Free Antivirus – відмінний безкоштовний антивірус, який заслужив визнання мільйонів користувачів по всьому світу завдяки надійному захисту від троянів і вірусів в реальному часі. Остання версія Avast може похвалитися оновленим інтуїтивно зрозумілим інтерфейсом, декількома унікальними функціями (AutoSandbox, Intelligent Scanner і т.д.), поліпшеним швидкодією і, головне, однією з найширших баз вірусів в світі (вона щодня поповнюється). На рис 1.4 можливо побачити головне меню антивірусу.

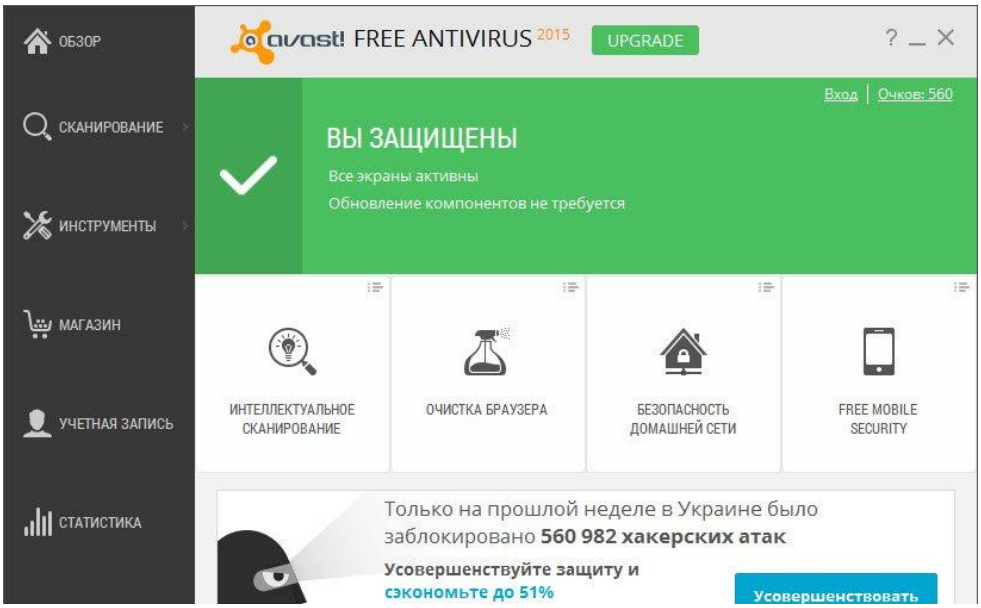

Рисунок 1.4 – Головне меню Avast Free Antivirus

Avast Free Antivirus являє користувачу нову функцію AutoSandbox, яка дозволяє автоматизувати процес приміщення підозрілих файлів в "пісочницю", де можна буде провести повний аналіз файлу і, при необхідності, вилікувати його. Ця функція дозволяє врятувати від миттєвого стирання досить великий відсоток файлів, уникнути системних помилок, пов'язаних з видаленням важливих системних файлів і тп. Додаток звертається з об'єктами акуратніше аналогів. В цілому, Avast Free Antivirus є відмінним вибором для середньостатистичного користувача, надаючи йому все необхідне для утримання системи в чистоті [8][1)].

AVG Anti-Virus Free – популярний антивірус основною характерною рисою якого є глибока інтеграція в систему. Він автоматично сканує файли і програми при їх запуску, що дозволяє уникнути зараження вірусами, троянами і шпигунськими програмами. Також АВГ надає користувачеві сканер, що настроюється за розкладом. Завдяки цій функції ви самі зможете контролювати як процес перевірки комп'ютера на заражені файли, так і

[1)] [8] Рейтинг антивирусов 2020. Какой антивирус лучше? URL: https://softcatalog.info/ru/obzor/reyting-antivirusov (дата звернення 03.06.2020)

процес їх лікування. У новій версії AVG повністю оновлений інтерфейс, який тепер може похвалитися приємним зовнішнім виглядом і зручними меню.

AVG Anti-Virus Free може похвалитися відмінними показниками захисту системи і досить малим споживанням системних ресурсів. Швидкість роботи програми вражає, а сканер електронної пошти позбавить вас від необхідності встановлювати спеціалізовані додатки, так як справляється він на всі 100%. На рис 1.5 можливо побачити головне меню антивірусу.

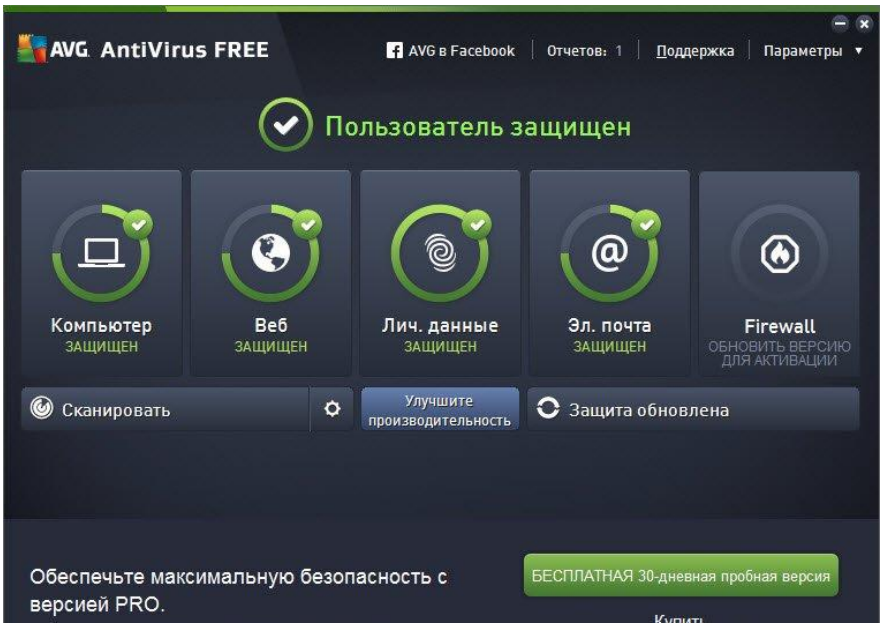

Рисунок 1.5 – Головне меню AVG Anti-Virus Free

Panda Antivirus Pro служить одній єдиній меті – вона захищає комп'ютер користувача від найбільш відомих видів віртуальних загроз. І можна з упевненістю сказати, що справляється вона з цим чудово. Встановивши Панду користувач отримує в своє розпорядження вкрай простий, але досить ефективний щит від будь-якої віртуальної загрози. Досить велика база вірусів Панди постійно поповнюється як розробниками, так і користувачам, яким "везе" знаходити нові різновиди вірусів. Ну а в елементарному інтерфейсі цього безкоштовного антивіруса Panda Dome

розбереться навіть дитина! На рис 1.6 можливо побачити головне меню антивірусу [8][1)].

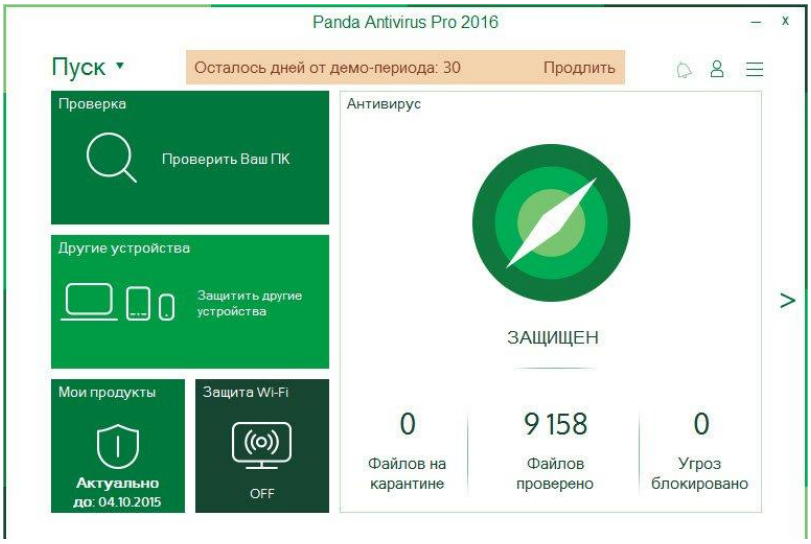

Рисунок 1.6 – Головне меню Panda Antivirus Pro

ESET NOD32 Smart Security – чудове комплексне рішення для захисту вашого комп'ютера від різного роду віртуальних загроз. Віруси, трояни, руткіти, рекламне ПО, спам – все це легко забувається після установки цього чудового антивіруса. "Найкращий захист – це напад", – ймовірно вважають програмісти ESET, так як за замовчуванням на нод32 виставлені досить агресивні налаштування по скануванню і знищення загрози.

ESET NOD32 Smart Security має в своєму арсеналі все необхідне для захисту вашого ПК: кілька ступенів захисту від будь-якого типу небажаного ПО або вірусу, персональний кастомізіруемий фаєрвол для шифровки з'єднання, батьківський контроль, контроль і скан пристроїв, що підключаються, безкоштовна цілодобова технічна і тд. Якщо необхідний антивірус для установки на ноутбук, НОД32 буде практично ідеальним рішенням, так як він має спеціальні профілі для роботи на портативних ПК,

---

[1)] [8] Рейтинг антивирусов 2020. Какой антивирус лучше? URL: https://softcatalog.info/ru/obzor/reyting-antivirusov (дата звернення 03.06.2020)

які дозволяють знизити витрату енергії. На рис 1.7 можливо побачити головне меню антивірусу.

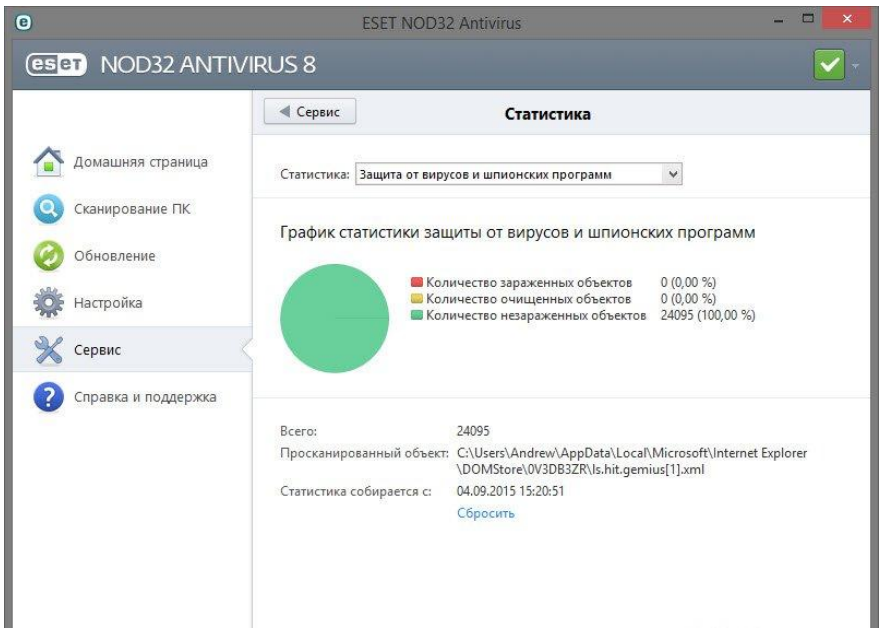

Рисунок 1.7 – Головне меню ESET NOD32 Smart Security

Dr.Web Antivirus – безкоштовне антивірусне рішення для виявлення і знищення вірусів та іншого шкідливого ПЗ. Завдяки ефективному евристичному аналізатору, Доктор Веб легко виявляє нові невідомі види віртуальних загроз навіть в соціальних мережах, а проактивний багатоступенева захист захищає систему від будь-якої небезпеки під час інтернет серфінгу або використання неперевірених носіїв інформації. Складанням база займаються незалежні лабораторії що підвищує якість бази вірусів. На рис 1.8 можливо побачити головне меню антивірусу.

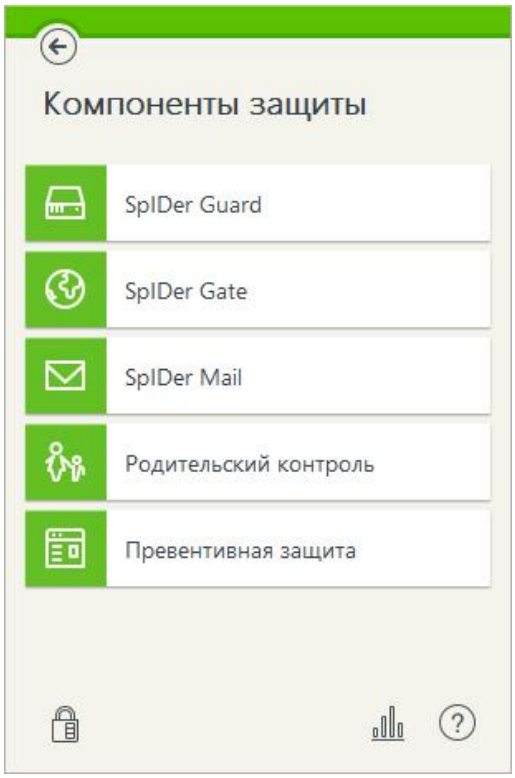

Рисунок 1.8 – Головне меню Dr.Web Antivirus

Kaspersky Free Antivirus зазвичай знаходиться на задвірках рейтингів антивірусів, оскільки багато платні і безкоштовні альтернативи превосходять набір функцій, який може безкоштовно запропонувати Лабораторія Касперського. Програма містить ядро захисту від шкідливих об'єктів з мережі інтернет, здатна забезпечити захист від вірусів, шпигунського ПЗ, яке потрапило на комп'ютер через мережу або зовнішні пристроїв. Також розробник заявляє, що антивірус здатний прискорити роботу персонального комп'ютера, але ми в цьому сильно сумніваємося. Порівняємо дані продукти за деякими критеріями, які вказані у таблиці нижче (табл.2.1).

При складанні оцінок по пошукові вірусів були використані результати тестування антивірусів з Інтернет-ресурсу [8][1)].

[1)] [8] Рейтинг антивирусов 2020. Какой антивирус лучше? URL: https://softcatalog.info/ru/obzor/reyting-antivirusov (дата звернення 03.06.2020)

Таблиця 1.1 – Порівняльна таблиця антивірусів

	Avast	Panda	AVG	ESET NOD32	Dr.Web
Час завантаження системи з антивірусом	<1 хв	<1 хв	>1 хв	>1 хв	>3 хв
Час сканування системних папок	>10 хв	>20 хв	>10 хв	>10 хв	>115 хв
Використання процесора	2,5%	3%	16,5%	10%	20%
Використання пам’яті	40 мб	40 мб	130 мб	110 мб	120 мб
Рейтинг	8	9	7	8	5

Згідно з порівняльним аналізом можна сказати, що Panda Antivirus Pro є одним з найкращих антивірусів, що неможливо сказати про Dr.Web.

1.4 Постановка задачі

Метою магістерської роботи є дослідження, аналіз та використання методів сигнатурного та евристичного аналізу для виявлення шкідливого програмного коду та розробка програмного додатку, що буде виконувати пошук та знешкодження вірусів з відомих сигнатур, що є наявними в базі даних системи. Для досягнення цієї мети були визначені задачі, які необхідно вирішити:

- провести дослідження та аналіз існуючих класів вірусів;
- здійснити огляд сучасного антивірусного програмного забезпечення;
- провести дослідження та здійснити вибір методів виявлення шкідливих програм;
- розробити програмний додаток, який повинен містити базу вірусів і антивірусний сканер, який повинен здійснювати пошук тих вірусів, сигнатури яких зазначені в базі даних;
- додаток повинен забезпечити внесення до карантину заражених вірусами файлів;

– антивірус повинен використовувати сигнатурний метод MD-5 пошуку вірусів та виконувати евристичний аналіз на основі нейронної мережі для пошуку шкідливих програм;
– інтерфейс розробленого програмного додатку для виявлення шкідливого програмного коду повинен бути виконаний в інтуїтивно-зрозумілому для користувача стилі;
– програмний додаток повинен бути протестований на вірусах та шкідливих програмах.

Розроблений в рамках виконання магістерської роботи додаток повинен забезпечувати виявлення та знешкодження шкідливого програмного коду, що може стати в подальшому першою ланкою в створенні потужного антивірусного програмного забезпечення.

2 ОГЛЯД МЕТОДІВ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОГРАМ

Кількість антивірусів, як і кількість вірусів, постійно зростає, також постійно розширюється загальне визначення і класифікація антивірусного ПЗ.

Антивірусна програма – програма для виявлення і лікування шкідливих об'єктів або інфікованих файлів, а також для профілактики – запобігання зараження файлу або операційної системи шкідливим кодом. Антивірусне програмне забезпечення складається з підпрограм, які намагаються виявити, запобігти розмноженню і вилучити комп'ютерні віруси і інші шкідливі програми. Багато сучасних антивірусів дозволяють виявляти і видаляти також троянські програми та інші шкідливі програми [9][1)].

На даний момент антивірусне програмне забезпечення розробляється в основному для ОС сімейства Windows від компанії Microsoft, що викликано великою кількістю шкідливих програм саме під цю платформу (а це, у свою чергу, викликане великою популярністю цієї ОС, також як і великою кількістю засобів розробки, в тому числі безкоштовних і навіть «інструкцій з написання вірусів»). На даний момент на ринок виходять продукти і під інші платформи настільних комп'ютерів, такі як Linux і Mac OS X. Це викликано початком розповсюдження шкідливих програм і під ці платформи, хоча UNIX-подібні системи завжди славилися своєю надійністю. Наприклад, відоме відео «Mac or PC» жартівливо показує перевагу Mac OS над Windows і великим антивірусним імунітетом Mac OS в порівнянні з Windows.

Крім ОС для настільних комп'ютерів і ноутбуків, також існують платформи і для мобільних пристроїв, такі як Windows Mobile, Symbian, iOS, BlackBerry, Android, Windows Phone та ін. Користувачі пристроїв на даних ОС також схильні до ризику зараження шкідливим програмним

[1)] [9] Щербаков А.Ю. Компьютерная безопасность. М., Изд. Молгачева С.В., 2001, 352 с.

забезпеченням, тому деякі розробники антивірусних програм випускають продукти і для таких пристроїв.

Розрізняють такі типи антивірусних програм [10][1)]:

- програми-детектори: призначені для знаходження заражених файлів одним із відомих вірусів. Деякі програми-детектори можуть також лікувати файли від вірусів або знищувати заражені файли. Існують спеціалізовані (тобто призначені для боротьби з одним вірусом) детектори та поліфаги (можуть боротися з багатьма вірусами);
- програми-лікарі: призначені для лікування заражених дисків і програм. Лікування програми полягає у вилученні із зараженої програми тіла вірусу. Також можуть бути як поліфагами, так і спеціалізованими;
- програми-ревізори: призначені для виявлення зараження вірусом файлів, а також знаходження ушкоджених файлів. Ці програми запам'ятовують дані про стан програми та системних областей дисків у нормальному стані (до зараження) і порівнюють ці дані у процесі роботи комп'ютера. В разі невідповідності даних виводиться повідомлення про можливість зараження;
- лікарі-ревізори: призначені для виявлення змін у файлах і системних областях дисків й у разі змін повертають їх у початковий стан.
- програми-фільтри: призначені для перехоплення звернень до операційної системи, що використовуються вірусами для розмноження і повідомляють про це користувача. Останній має можливість дозволити або заборонити виконання відповідної операції. Такі програми є резидентними, тобто вони знаходяться в оперативній пам'яті комп'ютера.

[1)] [10] Типи антивірусних програм – Direside. URL: https://sites.google.com/site/diresideinaction/tipi- antivirusnih-program (дата звернення 06.06.2020)

– програми-вакцини: використовуються для обробки файлів і boot-секторів із метою попередження зараження відомими вірусами (в останній час цей метод використовується все частіше).

Розглянемо основні методи протидії вірусам, застосовувані антивірусами.

2.1 Сигнатурне сканування

Найбільш популярним методом детектування шкідливих програм є перевірка сигнатур. Під сигнатурою мається на увазі фрагмент (або набір фрагментів), який завжди зустрічається в конкретному вірусі і ніколи – в інших програмах (у тому числі і в інших вірусах). Сигнатури використовуються для детектування шкідливих програм, що зберігають свій код постійним від копії до копії.

Антивірусна програма, скануючи файл на наявність вірусу, звертається до антивірусних баз, які складені виробником програми-антивірусу. Сигнатурні методи – точні методи виявлення вірусів, засновані на порівнянні файлу з відомими зразками вірусів. Сигнатурою вірусу вважається сукупність рис, що дозволяють однозначно ідентифікувати наявність вірусу у файлі (включаючи випадки, коли файл цілком є вірусом). Усі разом сигнатури відомих вірусів становлять антивірусну базу. Задачу виділення сигнатур, як правило, розв’язують люди – експерти в області комп'ютерної вірусології, здатні виділити код вірусу з коду програми і сформулювати його характерні риси у формі, найбільш зручній для пошуку. Для одержання сигнатури необхідно мати зразок вірусу. Отже, сигнатурний метод непридатний для захисту від нових вірусів, тому що доти, поки вірус не потрапив на аналіз до експертів, створити його сигнатуру неможливо. Саме тому все частіше великі епідемії викликаються новими вірусами. З моменту появи вірусу в мережі Інтернет до випуску перших сигнатур звичайно проходить кілька годин, і весь цей час вірус здатний заражати комп'ютери

майже безперешкодно. У випадку відповідності будь-якої ділянки коду програми, що аналізується, відомому коду (сигнатурі) вірусу в базах, програма-антивірус може по запиту виконати одну з наступних дій [11][1)]:

- вилучити інфікований файл;
- заблокувати доступ до інфікованого файлу;
- відправити файл у карантин (тобто зробити його недоступним для виконання з метою недопущення подальшого поширення вірусу);
- спробувати «вилікувати» файл, вилучивши вірус із тіла файлу;
- у випадку неможливості лікування/видалення, виконати цю процедуру при наступному перезавантаженні операційної системи.

На сьогодні, сигнатурне сканування є основним методом роботи антивірусного ПЗ. Для того, щоб така антивірусна програма успішно працювала протягом довгого часу, у базу сигнатур вірусів потрібно періодично завантажувати (звичайно, через Інтернет) дані про нові віруси. У сучасному світі в день з'являється кілька десятків нових вірусів, тому якщо базу сигнатур довго не обновляти, то сигнатурне сканування стає практично пошуком.

Переваги цього методу:

- відпрацьована надійність. Метод застосовується давно і з успіхом, можна сказати що це основний метод виявлення вірусу;
- висока швидкодія.

Недоліки:

- проблема лавиноподібного збільшення сигнатур. В результаті бази сигнатур виростають до непристойних розмірів, так що втрачається друга гідність методу. Ситуацію дозволяють шляхом особливих оптимізацій, коли одна сигнатура описує відразу безліч вірусів,

---

[1)] [11] Сучасні антивірусні програми та принцип їх роботи – Безпечний Інтернет. URL: https://sites.google.com/site/bezpecnijinternet1999/sucasni-antivirusni-programi-ta-princip-ieh-roboti (дата звернення 08.06.2020)

однак при цьому возника проблема помилкових спрацьовувань, що зменшує перша перевага.

– проблема виявлення нових вірусів. Вважається, що самі користувачі вносять занадто маленький внесок у збільшення бази даних вірусів. Тобто виявлення нових вірусів – це нібито проблема розробників антивіруса, що з одного боку здається справедливим, з іншого є порушенням принципу «безпека – справа кожного». У багатьох антивирусах вбудована функція «відправити на перевірку», якій слід безборонно користуватися. Основні методи вирішення проблеми – це взаємний обмін інформацією з іншими антивірусними конторами, евристичний, (тобто інтелектуальний, що використовують особливий алгоритм) пошук вірусів в Інтернеті, швидка реакція під час епідемій і свідомість системних інженерів, які аналізують підозрілу активність в мережі.

2.1.1 Метод пошуку за контрольною сумою

Контрольна сума – деяке значення, розраховане по набору даних шляхом застосування певного алгоритму і використовуване для перевірки цілісності даних при їхній передачі або зберіганні. Також контрольні суми можуть використовуватися для швидкого порівняння двох наборів даних на нееквівалентність: з великою ймовірністю різні набори даних будуть мати нерівні контрольні суми. Це може бути використане, наприклад, для детектування комп'ютерних вірусів [12][1)].

З погляду математики контрольна сума є хеш-функцією, використовуваною для обчислення контрольного коду – невеликої кількості біт усередині великого блоку даних, наприклад, мережного пакета або блоку

[1)] [12] Контрольна сума. Вікіпедія URL: https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0_%D1%81%D1%83%D0%BC%D0%B0 (дата звернення 09.06.2020)

комп'ютерного файлу, застосовуваного для виявлення помилок при передачі або зберіганні інформації. Значення контрольної суми додається в кінець блоку даних безпосередньо перед початком передачі або запису даних на який-небудь носій інформації. Згодом воно перевіряється для підтвердження цілісності даних. Популярність використання контрольних сум для перевірки цілісності даних обумовлена тим, що подібна перевірка просто реалізована у двійковому цифровому пристрої, легко аналізується і добре підходить для виявлення загальних помилок, викликаних наявністю шуму в каналах передачі даних. Найпоширенішими алгоритмами є: CRC32, MD5 і SHA-1 [13][1)].

CRC32 – циклічний надлишковий код використовується в роботі програм-архіваторів. MD5 – використовується не тільки для перевірки цілісності даних, але й дозволяє одержати досить надійний ідентифікатор файлу. Останній часто використовується при пошуку однакових файлів на комп'ютері, щоб не порівнювати весь зміст, а порівняти тільки хеш. SHA-1 – використовується для перевірки цілісності даних, що завантажуються, програмою Bittorrent.

Контрольна сума – результат застосування до довільного набору даних якоїсь хеш-функції, що розраховує короткий «дайджест» постійної довжини (наприклад, усього 4 байта). У базі даних антивірусу можна зберігати не самі сигнатури, а їх короткі контрольні суми. Такі ж суми розраховуються «на льоту» за вмістом тестуємих файлів. Розбіжність збереженого зразка з результатом розрахунків означає відсутність відповідного вірусу. А збіг – лише дуже високу (але не одиничну!) імовірність зараження. Причина криється в суті контрольних сум: вони, як і будь-які інші хеш-функції, являють собою відображення великої множини «об'єктів»-прообразів на малу множину «дайджестів»-образів. Відповідно, завжди можлива колізія: кілька різних «об'єктів» можуть мати однакові «дайджести», зокрема «погані»

[1)] [13] Шаньгин В.Ф. Информационная безопасность и защита информации.М., ДМК Пресс, 2014, 702 с.

контрольні суми можуть виявитися не тільки в заражених, але й у цілком «здорових» файлах.

Невисока ймовірність колізій і труднощі їх цілеспрямованого генерування – суть критеріїв якості того або іншого методу розрахунків контрольних сум. На практиці в антивірусах знайшли застосування трохи менш стійкі до колізій, зате набагато швидкодійні «технічні» хеш-функції.

Найчастіше використовуються циклічні надлишкові коди CRC. Метод використання заснований на поданні блоку даних у вигляді безперервного полінома з бітовими коефіцієнтами. У якості контрольного коду використовується залишок від ділення цього полінома на більш короткий, що «породжує» поліном, який має довжину *N* бітів. Техніка ділення така:

- у кінець блоку даних додається N-1 нульових бітів;
- замість арифметичного ділення використовується операція «додавання за модулем 2»;
- додавання з «лівими» нулями проміжних залишків не проводиться.

Приклад. Вихідні дані: 1010. Поліном, який «породжується»: 1001. Додаткові біти: 000.

```
1010000   |1001
1001
  1100
  1001
   1010
   1001
     11 ← CRC
```

CRC у якості дільника вибирає не будь-який ланцюжок бітів, а такий, що не розкладається на співмножники. Наприклад, для CRC-16 використовуються поліноми 0x1021 і 0x8005, а в багатьох стандартах зв'язку для CRC-32 зафіксовано 0x04C11DB7 і його «дзеркальне відбиття» 0xedb88320 (рис.2.1). Порівняно недавні дослідження показали високу якість поліномів 0x1EDC6F41, 0x741B8CD7 і 0x814141AB1. Крім того, іноді

застосовуються CRC-48 і CRC-64. Нерідко в стандартах на методи обчислення CRC застосовуються додаткові операції, такі як, наприклад, інвертування бітів результату. Властивостей методу, які розпізнають, це не поліпшує, але лише спрощує апаратну реалізацію алгоритму у зв'язному пристрої.

```
/* Расчет таблицы корректирующих коэффициентов */
make_crctable( void ) {
 int i, j;  DOUBLE r;
 for (i = 0; i <= 255; i++) {
  r = i;
  for (j = 8; j > 0; j--) if (r & 1)  r = (r >> 1) ^ 0xEDB88320;  else  r >>= 1;
  crctable[i] = r;
 }
}

/* Расчет CRC-32 для буфера buf длиной len */
DOUBLE crc32(unsigned char *buf, DOUBLE len) {
  DOUBLE crc = 0xFFFFFFFF;
  while (len--)  crc = (crc >> 8) ^ crctable[(crc ^ *buf++) & 0xFF];
  return crc ^ 0xFFFFFFFF; // Для облегчения аппаратной реализации
}
```

Рисунок 2.1 – Приклад реалізації для «швидкого» обчислення CRC-32

Аналіз контрольних сум – це спосіб відстеження змін в об'єктах комп'ютерної системи. На підставі аналізу характеру змін – одночасність, масовість, ідентичні зміни довжин файлів – можна робити висновок щодо зараження системи. Аналізатори контрольних сум (також використовується назва «ревізори змін») не використовують у роботі додаткові об'єкти і видають вердикт про наявність вірусу в системі винятково методом експертної оцінки. Найбільша популярність аналізу контрольних сум пов'язана зі спогадами про однозадачні операційні системи, коли кількість вірусів була відносно невеликою, файлів було небагато і змінювалися вони рідко. Непоганою альтернативою для CRC є контрольні суми, розраховані по алгоритму Марка Адлера. Вони також є 32-бітовим «дайджестом» блоку даних. Алгоритм Адлера виконується трохи швидше, ніж CRC, але

забезпечує більш високу ймовірність колізій, особливо на коротких наборах даних (рис.2.2) [14][1)].

```
#define BASE 65521
#define NMAX 5552
#define DO1(buf,i)  {s1 += buf[i]; s2 += s1;}
#define DO2(buf,i)  DO1(buf,i); DO1(buf,i+1);
#define DO4(buf,i)  DO2(buf,i); DO2(buf,i+2);
#define DO8(buf,i)  DO4(buf,i); DO4(buf,i+4);
#define DO16(buf)  DO8(buf,0); DO8(buf,8);
#define MOD(a) a %= BASE
/* Расчет к/с Адлера для буфера buf длиной len. Первоначально adler = 1 */
DOUBLE adler32(DOUBLE adler, unsigned char *buf, DOUBLE len) {
 DOUBLE s1 = adler & 0xffff;
 DOUBLE s2 = (adler >> 16) & 0xffff;  int k;
 while (len > 0) {
  k = len < NMAX ? (int)len : NMAX;  len -= k;
  while (k >= 16) { DO16(buf); buf += 16;  k -= 16; }
  if (k != 0) do { s1 += *buf++;  s2 += s1; } while (--k);
  MOD(s1);  MOD(s2);
 }
 return (s2 << 16) | s1;
}
```

Рисунок 2.2 – Алгоритм Адлера

Сьогодні ревізори змін втратили свої позиції і використовуються в антивірусах досить рідко. Частіше подібні технології застосовуються в сканерах при доступі – при першій перевірці з файлу знімається контрольна сума і міститься в кеші, перед наступною перевіркою того ж файлу сума знімається ще раз, порівнюється, і у випадку відсутності змін файл вважається незараженим.

2.1.2 Метод пошуку по HEX рядку в заданій позиції

HEX – це позиційна система числення по цілочисельній основі 16. У якості шістнадцятирічних чисел використовуються цифри від 0 до 9 і

[1)] [14] Технології виявлення вірусів. URL: https://studfile.net/ preview/5368373/page:3/ (дата звернення 10.06.2020)

латинські букви від A до F. Значення чисел від 0 до 9 звичайні, як і в десятковій системі, далі, від 10 до 16 використовуються букви A-F, тобто буква F = 16, далі 11 = 17, 12 = 18 і т.д.

Режим кодування/декодування «як текст» перекладає текст шматками. При кодуванні в HEX кожний символ буде перетворений у двохрозрядну шістнадцятирічну основу ASC-коду символу. При декодуванні система буде зчитувати по два символи і перетворювати їх в ASC-код, а потім у відповідний символ. Наприклад, якщо перетворити число 65535 в HEX у цьому режимі, то вийде: 3635353335. Режим кодування/декодування «як число» перекладає зазначений текст увесь цілком за один раз, як єдине число. Якщо зазначений для кодування текст неможливо перетворити в число, то відбудеться помилка. Наприклад, якщо перетворити число 65535 в HEX у цьому режимі, то вийде: FFFF.

Шаблон підстановки призначений для задання формату виводу шістнадцятирічних даних при кодуванні. Це може бути корисно при впровадженні шістнадцятирічних даних у програмний код.

У шаблоні можна використовувати наступні команди: {index} – порядковий номер конвертованого символу (починаючи з нуля); {hex} – шістнадцятирічний код символу. Інші символи залишаться без змін.

Наприклад, при шаблону arr[{index}] = {hex};, результат перетворення рядка "fox" у шістнадцятирічний вигляд буде таким: arr[0] = 0x66; arr[1] = 0x6F; arr[2] = 0x78;

У різних мовах програмування і технологіях використовуються різні формати уявлення шістнадцятирічних чисел (hex).

Кодер/декодер шістнадцятирічних даних дозволяє працювати з даними у форматі мов програмування: Basic, Qbasic, Visualbasic, C, C++, Visual C++, Pascal, Delphi, Assembler, SQL, а також підтримує роботу із шістнадцятирічними даними у форматі регулярних виразів, і форматі RTF.

2.2 Поведінковий блокатор

Інші назви: проактивний захист, поведінковий блокатор, HIPS. Поведінковий аналіз – технологія, у якій висновок про характер об'єкта, що перевіряється, робиться на основі аналізу виконуваних їм операцій.

Поведінковий аналіз досить вузько застосуємий на практиці, тому що більшість дій, характерних для вірусів, можуть виконуватися і звичайними додатками. Найбільшу популярність одержали поведінкові аналізатори скриптів і макросів, оскільки відповідні віруси практично завжди виконують ряд однотипних дій. Наприклад, для впровадження в систему майже кожний макровірус використовує один і той самий алгоритм: у який-небудь стандартний макрос, що автоматично запускається середовищем Microsoft Office при виконанні стандартних команд (наприклад, "Save", "Save As", "Open", і т.д.), записується код, що заражає основний файл шаблонів normal.dot і кожний документ, що знову відкривається. Засоби захисту, що вшиваються в BIOS, також можна віднести до поведінкових аналізаторів. При спробі внести зміни в MBR комп'ютера, аналізатор блокує дію і виводить відповідне повідомлення користувачеві. Крім цього поведінкові аналізатори можуть відслідковувати спроби прямого доступу до файлів, внесення змін у завантажувальний запис, форматування жорстких дисків і т.д. Поведінкові аналізатори не використовують для роботи додаткових об'єктів, подібних вірусним базам і, як наслідок, нездатні розрізняти відомі і невідомі віруси – усі підозрілі програми апріорі вважаються невідомими вірусами. Аналогічно, особливості роботи засобів, що реалізують технології поведінкового аналізу, не припускають лікування. Як і в попередньому випадку, можливе виділення дій, що однозначно трактуються як неправомірні – форматування жорстких дисків без запиту, видалення всіх даних з логічного диска, зміна завантажувального запису дискети без відповідних повідомлень та ін. Проте, наявність дій неоднозначних – наприклад, макрокоманда створення каталогу на жорсткому диску, змушує

також замислюватися про неправильні спрацьовування і, найчастіше, про тонке ручне настроювання поведінкового блокатора [15][1)].

Антивіруси, що використовують метод виявлення підозрілої поведінки програм, не намагаються ідентифікувати відомі віруси, замість цього вони прослідковують поведінку всіх програм. Якщо програма намагається записати будь-які дані у файл, що виконується (exe-файл), програма-антивірус може позначити цей файл, попередити користувача і запитати про подальші дії. У наш час подібні методи виявлення шкідливого коду, у тому або іншому виді, широко застосовуються в якості модуля антивірусної програми, а не окремого продукту. На відміну від методу пошуку відповідності визначенню вірусу в антивірусних базах, метод виявлення підозрілої поведінки дає захист від нових вірусів, яких ще немає в антивірусних базах. Однак слід ураховувати, що програми або модулі, побудовані на цьому методі, видають також велику кількість попереджень (у деяких режимах роботи), що робить користувача малосприйнятливим до всіх попереджень. Останнім часом ця проблема ще більш погіршилася, тому що стало з'являтися усе більше нешкідливих програм, що модифікують інші exe-файли, незважаючи на існуючу проблему помилкових попереджень. Незважаючи на наявність великої кількості попереджуючих діалогів, у сучасному антивірусному програмному забезпеченні цей метод використовується усе більше й більше.

2.2.1 Евристичний аналіз

Евристичний аналізатор (евристик) – це програма, яка аналізує код об'єкта, що перевіряється і за непрямими ознаками визначає, чи є об'єкт шкідливим. На відміну від сигнатурного методу евристик може детектуючих

[1)] [15] Антивірусні програми – захист локальної мережі. URL: https://www.sites.google.com/site/zahistlokalnoiemerezi/zahist/antivirusni-programi (дата звернення 10.06.2020)

як відомі, так і невідомі віруси (тобто ті, які були створені після евристика). Робота аналізатора, як правило, починається з пошуку в коді підозрілих ознак (команд), характерних для шкідливий-них програм. Цей метод називається статичним аналізом. Наприклад, багато шкідливі коди шукають виконувані програм-ми, відкривають знайдені файли і змінюють їх. Евристик проглядає код програми, але, зустрівши підозрілу команду, збільшує якийсь «лічильник підозрілості» для цього додатка. Якщо після перегляду всього коду значення лічильника перевищує задане граничне значення, то об'єкт зізнається підозрілим. Зрозуміло це може бути застосовано не тільки до програм-мам зберігаються на комп'ютері, але і до інших файлів, які передаються по мережі, що зберігаються на поштових серверах, і т.д. До при-міру, як потенційно небезпечні маркуються вкладення з відсутніх розширеннями файлів. Команди, без дозволу відкривають адресну книгу Outlook Express, що створюють ключі реєстру або активують мережеві порти, теж ідентифікується як потенційні шкідники [16][1)].

Евристичні методи виявляються надзвичайно успішними в разі дуже коротких програмних частин в завантажувальному секторі: якщо, наприклад, програма робить запис в сектор 1, доріжку 0, сторону 0, то це призводить до зміни розділу накопитель. Але крім допоміжної програми FDISK ця команда більше ніде не використовується, і тому в разі її несподіваної появи мова (з великою ймовірністю) йде про завантажувальному вірус.

Тому в сучасних антивирусах статичний аналіз використуються в поєднанні з динамічним. Ідея такого комбінованого підходу полягає в тому, щоб до того як додаток буде запущено на комп'ютері користувача, емулювати його запуск в безпечному віртуальному оточенні, яке називається також буфером емуляції, або «пісочницею». У маркетингових матеріалах

[1)] [16] Цветков В.Я., Булгаков С.В. Эвристический анализ как инструмент информационной безопасности. Журнал Современные наукоемкие технологии. 2010. №1 С. 53-53 URL: https://top-technologies.ru/pdf/ 2010/ 1/22.pdf (дата звернення 24.06.2020)

постачальники використовують ще один термін – «емуляція віртуального комп'ютера в комп'ютері».

Технологія «пісочниць» така: підозрілий файл поміщається в ізольовану від решти системи «пісочницю», де перевіряється його поведінку. При цьому кожен виконуваний файл запускається у віртуальному середовищі, яку стандартний програмний інтерфейс додатків Windows (Application Programming Interface, API) організовує в так званій «в'язниці» (jail). Потенційно шкідливий додаток не може діяти на операційну систему і завдати їй шкоди. Якщо ж воно віддає, наприклад, команду про зміну системних налаштувань, то класифікується як шкідник і може бути або видалено, або переведено в карантин. Поза всякими сумнівами, методи на основі «пісочниці» є надійними і досить точно розпізнають і усувають невідоме досі шкідливе прогпрограмних забезпечення. Проте для практичного виконання вони придатні лише умовно: оскільки нові віруси численні, а їх структура досить складна, емуляція ж на шлюзі триває довго і забирає надто багато ресурсів. Крім того, «пісочниці» приносять мало користі, коли віруси виконують свою програму з тимчасовою затримкою і на момент перевірки не розпізнаються як шкідливі коди.

Переваги методу:

- дуже перспективний напрямок, в майбутньому можливості евристичного модуля зростуть і комп'ютер і інформація будуть краще захищені від несподіваних і новітніх загроз.
- мвристичний модуль може реагувати на загрози, інформації про яких немає в базі сигнатур.
- недоліки методу:
- помилкове спрацьовування на безпечні події. В результаті користувач може в роздратуванні відключити евристичний модуль, зменшивши захист.
- через особливості роботи евристичного модуля є проблема надмірного споживання обчислювальних потужностей. Попросту

кажучи, антивірус зжирає всю пам'ять і процесор, в результаті не те що в гри не пограєш, в Word толком не попрацюєшь. Підсумок той же – відключення модуля і зменшення захисту [17][1)].

За новим шляхом іде технологія блокування поведінки, де комбінуються обидва згадані методу. При її застосуванні завдяки ієрархічній обробці трафіку даних зводиться до мінімуму висока потреба «пісочниці» в ресурсах і зменьшується ймовірність помилкових спрацьовувань евристичних методів: в той час як в «пісочницях» докладно вивчається кожен вхідний файл, блокатори поведінки аналізують поведінку лише тих файлів, які евристичними методами були відзначені як підозрілі через свого походження або присутності відомих складових частин коду. Тим самим значно скорочується кількість файлів, що, в свою чергу, позитивно відбивається на системній навантаженні. Одночасно знижується ризик помилкових спрацьовувань, оскільки підозрілі файли досліджуються з точки зору їх реальної поведінки. Для повноти картини розглянуті методи представлені в таблиці, де проведена їх класифікація за критичним до виявлення атак параметрам.

2.2.2 Логічні методи

Найбільш простий і природній спосіб побудови «розв’язувачів» заснований на застосуванні до множини виявлених симптомів системи «продукцій». Продукцією називається правило вигляду: ЯКЩО «предикат», ТО «вивід».

Предикати (умови) можуть являти собою складні функції над різними симптомами і раніше отриманими виводами, утворені за допомогою логічних дій «І», «АБО» і «НЕ» (рис.2.3).

[1)] [17] Принципи роботи антивіруса. URL: https://kursy.zp.ua/main/principi/uk/virus-principi-roboti-antivirusa.aspx (дата звернення 12.06.2020)

```
ЕСЛИ
 "Открыть_На_Запись" И  "Прочитать_Начало" И  "Записать_В_Конец" И
 "Записать_В_Начало" И  "Закрыть"
ТО
 Вывод := "COM.VIRUS";
КОНЕЦ_ЕСЛИ

       ЕСЛИ
           Вывод="COM.VIRUS" И "Поиск_Файлов"
         ТО
         Вывод := Вывод + "SEARCH";
КОНЕЦ_ЕСЛИ

ЕСЛИ
          Вывод="COM.VIRUS" И "Остаться_в_памяти"
ТО
         Вывод := Вывод + "TSR"
КОНЕЦ_ЕСЛИ
```

Рисунок 2.3 – Приклад логічного методу

Такий підхід має масу недоліків:

- об'єктивність підсумкового виводу сильно залежить від того, наскільки коректно, чітко і несуперечливо вірусолог складе і опише систему правил;
- при необхідності видозмінити цю систему (наприклад, з появою нового типу вірусів) може знадобитися переписати її повністю із самого початку;
- система не дозволяє врахувати порядок появи симптомів у файлі і ін.

3 ЗАСТОСУВАННЯ ШТУЧНОЇ НЕЙРОННОЇ МЕРЕЖІ В ЗАВДАННЯХ ПОШУКУ ВІРУСІВ

ШНМ є спрощеною моделлю мозку людини і являє набір елементів з'єднаних між собою певним чином, так щоб між ними забезпечувалося взаємодію. Ці елементи також називаються нейронами або вузлами. Вони являють собою примітивні процесори обчислювальні можливості, яких зазвичай обмежується деяким правилом комбінування вхідних сигналів і правилом активації, що дозволяє обчислювати вихідний сигнал за сукупністю вхідних сигналів. Вихідний сигнал може надсилатися іншим елементам по зважених зв'язків, з кожною з яких пов'язаний ваговий коефіцієнт або вага. Залежно від вагового коефіцієнта переданий сигнал або посилюється або придушується. Перевага ШНМ полягає в тому що, вони можуть автоматично здобувати знання в процесі навчання і мають здатність узагальнення. Однак, навченість одночасно є і недоліком, так як аналіз навченої нейронної мережі досить складний і має свої нюанси. Але один із самих позитивних якостей використання ШНМ, полягає в тому що, хоча елементи такої мережі мають обмежені обчислювальні можливості, але вся мережа в цілому, що складається як правило, з великого числа таких елементів, здатна виконувати досить складні завдання [18][1)].

3.1 Математична модель нейрона

Нейрон являє собою одиницю обробки інформації в нейронної мережі. Модель нейрона лежить в основі штучних нейронних мереж (рис.3.1). У цій моделі можна виділити три основні елементи.

Перше – це набір сінапсів або зв'язків, кожен з яких характеризується своїм вагою. Зокрема, сигнал x_j на вході синапсу j, пов'язаного з нейроном k,

---

[1)] [18] Нейронні мережі. URL: https://ukrbukva.net/page,3,92840-Neiyronnye-seti.html (дата звернення 12.08.2020)

множиться на вагу w_{kj}. Друге – Суматор складає вхідні сигнали, зважені щодо відповідних синапсів нейрона. Цю операцію можна описати як лінійну комбінацію. Третє – Функція активації обмежує амплітуду вихідного сигналу нейрона. Зазвичай нормалізований діапазон амплітуд виходу нейрона лежить в інтервалі [0,1] або [–1, 1] [18][1)].

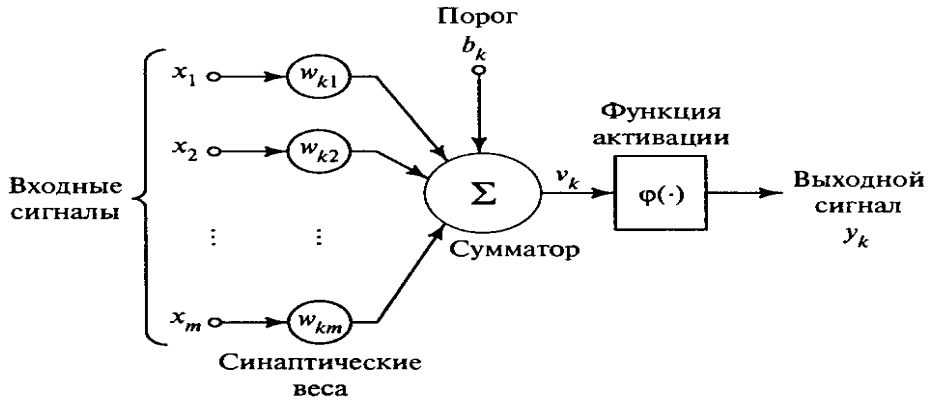

Рисунок 3.1 – Модель нейрона

У моделі нейрона (рис. 3.1) включений пороговий елемент, який позначений символом b_k . Ця величина відображає збільшення або зменшення вхідного сигналу, що подається на функцію активації.

У математичному поданні функціонування нейрона k можна описати наступною парою рівнянь:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j , \tag{3.1}$$

$$y_k = \varphi(u_k + b_k) , \tag{3.2}$$

де $x_1, x_2, \ldots, x_m$ – вхідні сигнали;

[1)] [18] Нейронні мережі. URL: https://ukrbukva.net/page,3,92840-Neiyronnye-seti.html (дата звернення 12.08.2020)

$w_{k1}, w_{k2}, \ldots, w_{km}$ – синаптичні ваги нейрона k;

u_k – лінійна комбінація вхідних впливів;

b_k – поріг;

φ(*) – функція активації;

y_k – вихідний сигнал нейрона.

Використання порога b_k в моделі, постсинаптичний потенціал обчислюється таким чином:

$$v_k = u_k + b_k . \tag{3.3}$$

Поріг b_k є зовнішнім параметром штучного нейрона k. Він присутній у виразі (3.2). Беручи до уваги вираз (3.3), формулу (3.2) можна перетворити до наступного вигляду:

$$y_k = \varphi(v_k) . \tag{3.4}$$

Функція активації, представлена у формулах (3.2, 3.4) як φ(v), визначає вихідний сигнал нейрона. Ця функція може мати різний вид (рис. 3.2).

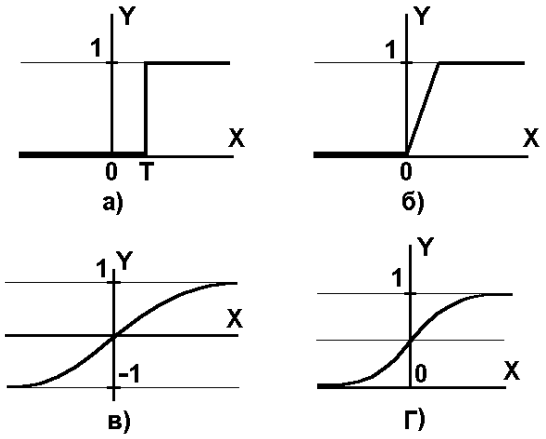

Рисунок 3.2 – Функція активації:

а) функція одиничного стрибка; б) лінійний поріг (гістерезис);

в) сигмоїдальна функція; г) гіперболічний тангенс.

Сигмоїдальна функція:

$$\varphi = \frac{1}{1 + \exp(-a\nu)}, \tag{3.5}$$

де a – параметр нахилу сигмоїдальной функції, змінюючи цей параметр можна управляти крутизною функції.

Основна перевага цієї функції в тому, що вона диференційовна на всій осі абсцис і має дуже просту похідну:

$$\varphi'(\nu) = \varphi(\nu) \cdot (1 - \varphi(\nu)) \tag{3.6}$$

3.2 Класифікація нейронних мереж

Існує кілька способів класифікації ШНМ. Зокрема по топології (рис.3.3), виділяють: повнозв'язні мережі (в яких кожен з нейронів передає свій вихідний сигнал усім іншим нейронам, у тому числі й самому собі); багатошарові мережі. У них нейрони об'єднуються в шари (шаром є сукупність нейронів з єдиними вхідними сигналами).

У багатошаровій мережі входом вхід нульового шару, а виходом – вихід останнього шару. Проміжні шари (між вхідним і вихідним шаром) прийнято називати прихованими – багатошарова нейромережа може містити один або кілька прихованих шарів; слабкозв'язані мережі [19][1)].

[1)] [19] Нейронні мережі. URL: https://www.asimovinstitute.org /neuralnetwork-zoo. (дата звернення 14.07.2020)

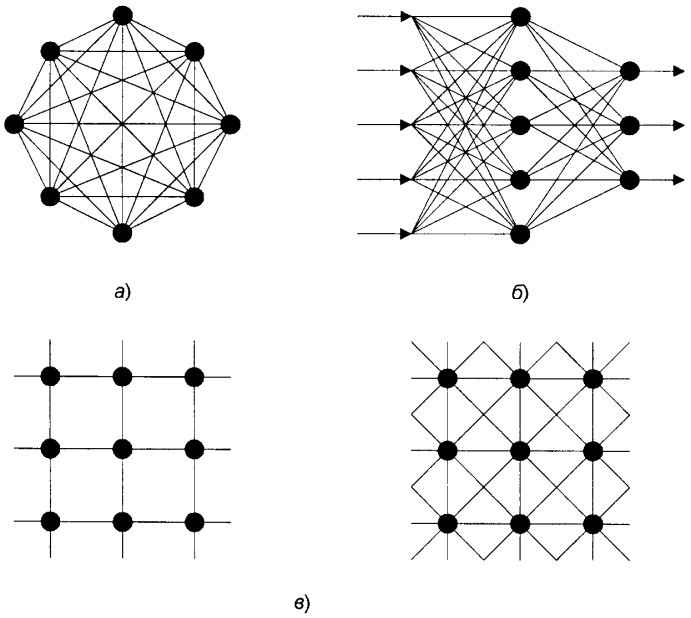

Рисунок 3.3 – а) повнозв'язні мережі; б) багатошарова мережа з послідовними зв'язками; в) слабкозв'язані мережі

За наявністю зворотних сигналу зв'язків багатошарові мережі підрозділяються на:

- мережі без зворотних зв'язків (або прямого поширення). Це нейромережі, в яких сигнали передаються від шару до шару у напрямку від входу до виходу, причому якщо не обумовлено протилежне, то вихідний сигнал шару N подається на входи шару N + 1;
- мережі зі зворотними зв'язками (або рекурентні). Це мережі, в яких сигнал від наступних шарів передається на попередні.

За типом функції активації нейронів ШНМ можна розділити на гомогенні ШНМ (у всіх нейронів однакова функція активації) і гетерогенні, в яких застосовуються різні функції активації.

Всі моделі ШНМ, які не можна віднести ні до однієї з вище перерахованих груп називаються неструктурованими.

3.2.1 Багатошаровий персептрон

Багатошаровий персептрон (MLP) – нейронна мережа прямого поширення сигналу (без зворотних зв'язків), в якій вхідний сигнал перетвориться в вихідний, проходячи послідовно через кілька шарів. Багатошарові персептрони мають три відмітних ознаки. Перше: кожен нейрон мережі має нелінійну функцію активації. Важливо підкреслити, що дана нелінійна функція є гладкою (тобто усюди диференційованою). Найпопулярнішою функцією, що задовольняє цій вимозі, є сигмоїдальна. Друге: мережа містить один або декілька шарів прихованих нейронів, які не є частиною входу або виходу мережі. Ці нейрони дозволяють мережі навчатися вирішення складних задач, послідовно витягуючи найбільш важливі ознаки з вхідного образу (вектора). Третє: мережа має високий ступінь зв'язності, що реалізовується за допомогою синаптичних з'єднань. Зміна рівня зв'язності мережі вимагає зміни безлічі синоптичних з'єднань або їхніх вагових коефіцієнтів [20][1)].

Врахуємо той факт що використовувана в даній роботі MLP відноситься до синхронних – це коли стан змінюється відразу у цілої групи нейронів, як правило, у всього шару.

Алгоритмічно хід часу в такий ШНМ задається ітераційним виконанням однотипних дій над нейронами. Багатошарова мережа представлена тришаровим персептроном з повністю пов'язаними шарами (рис. 3.4).

Тут доречно відзначити важливу роль нелінійності активаційної функції. Так як, якби вона не володіла даними властивістю або не входила в алгоритм роботи кожного нейрона, результат функціонування будь-якої *m*-

---

[1)] [20] Порівняльна характеристика нейронних мереж. URL: https://cyberleninka.ru /article/n/obzor-svyortochnyh-neyronnyh-setey-dlya-zadachi-klassifikatsii -izobrazheniy (дата звернення 16.08.2020)

шарової ШНМ з ваговими матрицями $W^{(i)}$, i=1,2,...m для кожного шару і зводився б до розмноження вхідного вектора сигналів X на матрицю

$$W^{(\Sigma)}=W^{(1)}\cdot W^{(2)}\cdot\ldots\cdot W^{(m)}, \tag{3.7}$$

тобто фактично така m-шарова НМ еквівалентна одношаровій НМ з ваговою матрицею єдиного шару $W^{(\Sigma)}$:

$$Y=XW^{(\Sigma)}. \tag{3.8}$$

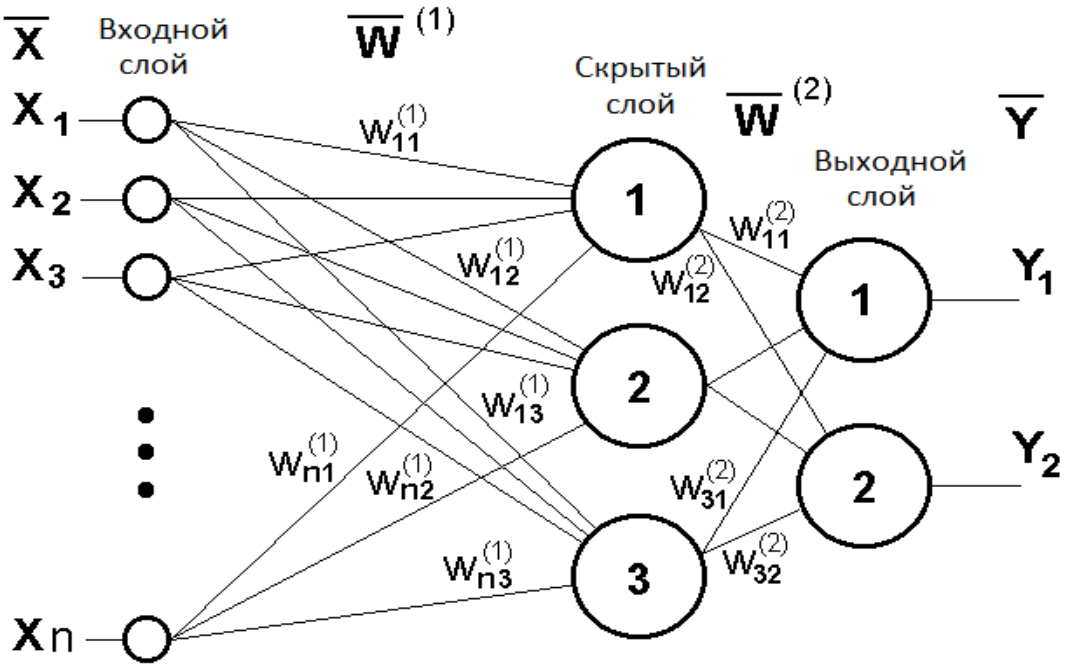

Рисунок 3.4 – Багатошаровий персептрон

Введення нелінійності MLP мереж, взагалі кажучи, збільшує обчислювальну потужність мережі, тобто дозволяє з меншого числа нейронів з «нелінійними» синапсами сконструювати ШНМ, що виконує роботу звичайної ШНМ з великим числом стандартних нейронів і більш складної конфігурації.

3.2.2 Навчання нейронної мережі

Здатність до навчання є фундаментальною властивістю мозку. Наявність цієї властивості в ШНМ є найбільш характерною і привабливою рисою, і відрізняє від інших обчислювальних систем.

У ШНМ навчання розглядається як налаштування параметрів мережі для вирішення поставленого завдання. В якості таких параметрів зазвичай виступають синоптичні коефіцієнти (ваги зв'язків). Крім ваг зв'язків в параметри мережі можуть включатися також пороги (зсуву). Мета навчання ШНМ – досягти бажаної вихідний реакції мережі на деякий безліч вхідних сигналів зване навчальною вибіркою. Вхідний і вихідний безлічі сигналів зручно інтерпретувати як вектора. Процес навчання ШНМ здійснюється шляхом послідовного пред'явлення вхідних векторів з навчальної вибірки з одночасною підстроюванням параметрів мережі відповідно до деякої процедурою, званої алгоритмом навчання. Процедура навчання проводиться до тих пір, поки не буде досягнута бажана вихідна реакція ШНМ для всієї навчальної вибірки. У математичному сенсі навчання ШНМ являє собою ітераційну процедуру, спрямовану на таке підстроювання параметрів мережі, щоб деякий функціонал якості звертався в оптимум для всієї навчальної вибірки. У ролі такого функціоналу зазвичай використовується функція помилки, що характеризує ступінь близькості відображення вхідного вектора в бажаний вихідний. У загальному випадку функціонал якості (функція помилки) може мати довільний вигляд, тому навчання ШНМ перетворюється на задачу багатоекстремальної неопуклої багатовимірної оптимізації. Для формування процесу навчання необхідно, насамперед, мати модель зовнішнього середовища, в якому функціонує ШНМ, тобто визначити доступну для мережі інформацію. Ця модель визначає парадигму навчання. У рамках певної парадигми навчання далі конструюються правила підстроювання параметрів, такі як конкретний алгоритм навчання. Існують три парадигми навчання: «з учителем», «без вчителя» (самонавчання) і

змішана. Коли в мережі тільки один шар, алгоритм її навчання з учителем досить очевидний, оскільки правильні вихідні стани нейронів єдиного шару свідомо відомі, і підстроювання синаптичних зв'язків йде в напрямку, мінімізації помилки на виході мережі. За цим принципом будується, наприклад, алгоритм навчання одношарового персептрона. У багатошарових ж мережах оптимальні вихідні значення нейронів всіх шарів, крім останнього, як правило, не відомі, і двох чи більше шаровий персептрон вже неможливо навчити, керуючись тільки величинами помилок на виходах ШНМ. Найбільш прийнятний варіант вирішення цієї проблеми – поширення сигналів помилки від виходів ШНМ до її входів, в напрямку, зворотному прямому поширенню сигналів у звичайному режимі роботи. Цей алгоритм навчання ШНМ називається «алгоритм зворотного поширення помилки».

3.2.3 Алгоритм зворотного поширення помилки

Відповідно до методу найменших квадратів, мінімізуємою цільовою функцією помилки ШНМ є величина:

$$E(w) = \frac{1}{2}\sum_{j,p}(y_{j,p}^{(N)} - d_{j,p})^2 , \qquad (3.9)$$

де $y_{j,p}^{(N)}$ – реальний вихідний стан нейрона j вихідного шару N нейронної мережі при подачі на її входи p-го способу;

d_{jp} – ідеальний (бажаний) вихідний стан цього нейрона.

Підсумовування ведеться по всіх нейронам вихідного шару і по всіх оброблюваних мережею образах. Мінімізація ведеться методом градієнтного спуску, що означає підстроювання вагових коефіцієнтів наступним чином:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \frac{\partial E}{\partial w_{ij}}, \tag{3.10}$$

де w_{ij} – ваговий коефіцієнт синоптичного зв'язку, що з'єднує i-ий нейрон шару n-1 з j-им нейроном шара n,

η – коефіцієнт швидкості навчання, $0<\eta<1$.

Як показано в,

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j} \cdot \frac{\partial s_j}{\partial w_{ij}}, \tag{3.11}$$

де y_j – вихід нейрона j,

s_j – зважена сума його вхідних сигналів, тобто аргумент активаційної функції.

Так як множник dy_j/ds_j є похідною цієї функції по її аргументу, з цього випливає, що повинна бути гладкою (зазвичай застосовується сигмоїдальна функція). Тоді в такому випадку:

$$\frac{dy_j}{ds_j} = \alpha \cdot s_j \cdot (1 - s_j). \tag{3.12}$$

Третій множник $\partial s_j/\partial w_{ij}$, очевидно, дорівнює виходу нейрона попереднього шару $y_i^{(n-1)}$.

Стосовно першого множника в (3.11), він легко складається наступним чином:

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot \frac{\partial s_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} w_{jk}^{(n+1)}. \tag{3.13}$$

Тут підсумовування по *k*-й компоненті виконується серед нейронів шару *n*+1. Запровадивши нову змінну

$$\delta_j^{(n)} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j}, \tag{3.14}$$

було отримано рекурсивну формулу для розрахунків величин $\delta_j^{(n)}$ шару *n* з величин $\delta_k^{(n+1)}$ більш старшого шару *n*+1:

$$\delta_j^{(n)} = \left[\sum_k \delta_k^{(n+1)} \cdot w_{jk}^{(n+1)} \right] \cdot \frac{dy_j}{ds_j}. \tag{3.15}$$

Для вихідного шару:

$$\delta_l^{(N)} = (y_l^{(N)} - d_l) \cdot \frac{dy_l}{ds_l}. \tag{3.16}$$

Тепер ми можемо записати (3.10) в розкритому вигляді:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}. \tag{3.17}$$

Іноді для додання процесу корекції ваг деякої інерційності, згладжує різкі скачки при переміщенні по поверхні цільової функції, (3.17) доповнюється значенням зміни ваги на попередній ітерації:

$$\Delta w_j^{(n)}(t) = -\eta \cdot (\mu \cdot \Delta w_{ij}^{(n)}(t-1) + (1-\mu) \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}), \tag{3.18}$$

де μ – коефіцієнт інерційності,

t – номер поточної ітерації.

Повний алгоритм «зворотного поширення помилки», навчання ШНМ.

Подати на входи мережі один з можливих образів і в режимі звичайного функціонування ШНМ, коли сигнали поширюються від входів до виходів, розрахувати значення останніх.

$$s_j^{(n)} = \sum_{i=0}^{M} (y_i^{(n-1)} \cdot w_{ij}^{(n)}) - b_j^{(n)}, \tag{3.19}$$

де M – число нейронів в шарі n–1,

$b_j^{(n)}$ – поріг задає зсув;

$y_i^{(n-1)}=x_{ij}^{(n)}$ – i-ий вхід нейрона j шару n.

$$y_j^{(n)} = f(s_j^{(n)}), \tag{3.20}$$

де $f(*)$ – сигмоід

$$y_q^{(0)}=I_q, \tag{3.21}$$

де I_q – q-а компонента вектора вхідного образу.

Розрахувати $\delta^{(N)}$ для вихідного шару за формулами (3.16). Розрахувати за формулою (3.17) або (3.18) зміни ваг $\Delta w^{(N)}$ шару N.

Розрахувати за формулами (3.15) і (3.17) (або (3.15) і (3.18)) відповідно $\delta^{(n)}$ і $\Delta w^{(n)}$ для всіх інших n=N-1,...,1.

Скорегувати всі ваги в НМ

$$w_{ij}^{(n)}(t) = w_{ij}^{(n)}(t-1) + \Delta w_{ij}^{(n)}(t). \tag{3.22}$$

Якщо помилка мережі істотна, перейти на крок 1. В іншому випадку – кінець.

Мережі на кроці 1 поперемінно у випадковому порядку пред'являються всі тренувальні образи, щоб мережа, не забувала одні дані по міру запам'ятовування інших (рис. 3.5).

З виразу (3.17) випливає, що коли вихідне значення $y_i^{(n-1)}$ прагне до нуля, ефективність навчання помітно знижується. При довічних вхідних векторах в середньому половина вагових коефіцієнтів НЕ буде коригуватися, тому область можливих значень виходів нейронів [0,1] бажано зрушити в межі [–0.5, + 0.5], що досягається простими модифікаціями логістичних функцій.

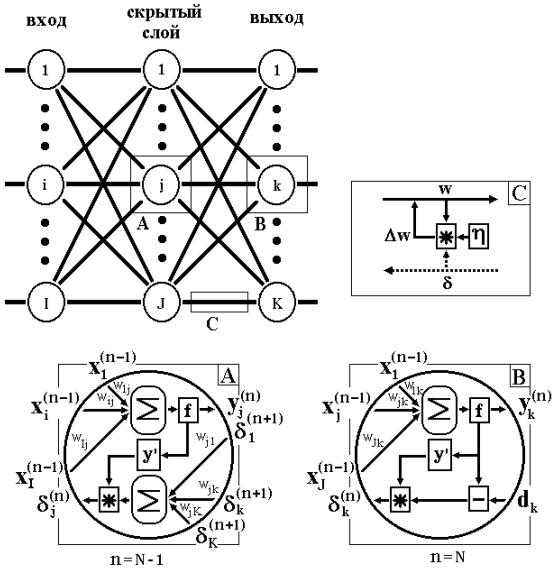

Рисунок 3.5 – Діаграма сигналів в мережі при навчанні за алгоритмом зворотного поширення помилки

Наприклад, сигмоїд з експонентою перетвориться до виду:

$$f(x) = -0.5 + \frac{1}{1 + e^{-\alpha \cdot x}}. \qquad (3.23)$$

Розглянемо питання ємності НМ, тобто числа образів, що пред'являються на її входи, які вона здатна навчитися розпізнавати. Для мереж з кількістю шарів більше двох, він залишається відкритим. Для НМ з двома шарами, тобто вихідним і одним прихованим шаром, детерміністська ємність мережі C_d оцінюється так:

$$N_w/N_y<C_d<N_w/N_y\cdot\log(N_w/N_y), \qquad (3.24)$$

де N_w – число прибудованих ваг,

N_y – число нейронів у вихідному шарі.

Слід зазначити, що даний вираз отримано з врахуванням деяких обмежень. По-перше, число входів N_x і нейронів в прихованому шарі N_h повинно задовольняти нерівності $N_x+N_h>N_y$. По-друге, $N_w/N_y>1000$. Однак вищенаведена оцінка виконувалася для мереж з активаційними функціями нейронів у вигляді порога, а ємність мереж з гладкими активаційними функціями (2.23), зазвичай більше. Крім того, фігурує в назві ємності прикметник "детерміністський" означає, що отримана оцінка ємності підходить абсолютно для всіх можливих вхідних образів, які можуть бути представлені N_x входами. Насправді розподіл вхідних образів, як правило, володіє деякою регулярністю, що дозволяє НМ проводити узагальнення і, таким чином, збільшувати реальну ємність. Оскільки розподіл образів, в загальному випадку, заздалегідь не відомо і можна говорити про таку ємності тільки імовірно, але зазвичай вона разу в два перевищує ємність детерміністську. При аналізі, орієнтованому на визначення ємності ШНМ, логічно порушити питання про необхідну потужність вихідного шару мережі, що виконує остаточну класифікацію образів. Справа в тому, що для розділення безлічі вхідних образів, наприклад, за двома класами достатньо всього одного виходу. При цьому кожен логічний рівень – "1" і "0" – позначатиме окремий клас. На двох виходах можна закодувати вже 4 класу і так далі. Однак результати роботи мережі, організованої таким чином, не

дуже надійні. Для підвищення достовірності класифікації бажано ввести надмірність шляхом виділення кожному класу одного нейрона у вихідному шарі або, що ще краще, кілька нейронів, кожен з яких навчається визначати приналежність образу до класу зі своєю ступенем вірогідності. Наприклад: високою, середньою і низькою.

Такі НМ дозволяють проводити класифікацію вхідних образів, об'єднаних в нечіткі (розмиті або пересічні) безлічі. Ця властивість наближає подібні ШНМ до умов реального життя. Розглянута НМ має кілька «вузьких місць». По-перше, в процесі навчання може виникнути ситуація, коли великі додатні або від’ємні значення вагових коефіцієнтів змістять робочу точку на сигмоїд багатьох нейронів в область насичення. Малі величини похідної від логістичної функції приведуть у відповідність з (3.15) і (3.16) до зупинки навчання, що паралізує НМ. По-друге, застосування методу градієнтного спуску не гарантує, що буде знайдений глобальний, а не локальний мінімум цільової функції. Ця проблема пов'язана ще з однією, а саме – з вибором величини швидкості навчання. Доведення збіжності навчання в процесі зворотного поширення засновано на похідних, тобто збільшення ваг і, отже, швидкість навчання повинні бути нескінченно малими, проте в цьому випадку навчання відбуватиметься неприйнятно повільно. З іншого боку, занадто великі корекції ваг можуть призвести до постійної нестійкості процесу навчання. Тому в якості η зазвичай вибирається число менше 1, але не дуже маленьке, наприклад, 0.1, і воно, взагалі кажучи, може поступово зменшуватися в процесі навчання [18][1)].

[1)] [18] Нейронні мережі. URL: https://ukrbukva.net/page,3,92840-Neiyronnye-seti.html (дата звернення 12.08.2020)

4 РОЗРОБКА ДОДАТКА ДЛЯ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО КОДУ

У магістерській роботі розробляється додаток для виявлення шкідливого програмного коду (антивірусна програма) зі зручним і зрозумілим інтерфейсом для користувачів.

Архітектура розробленого антивірусу має такий вигляд: модуль відновлення, модель сканера, модуль резидентного захисту і модуль карантину (рис.4.1).

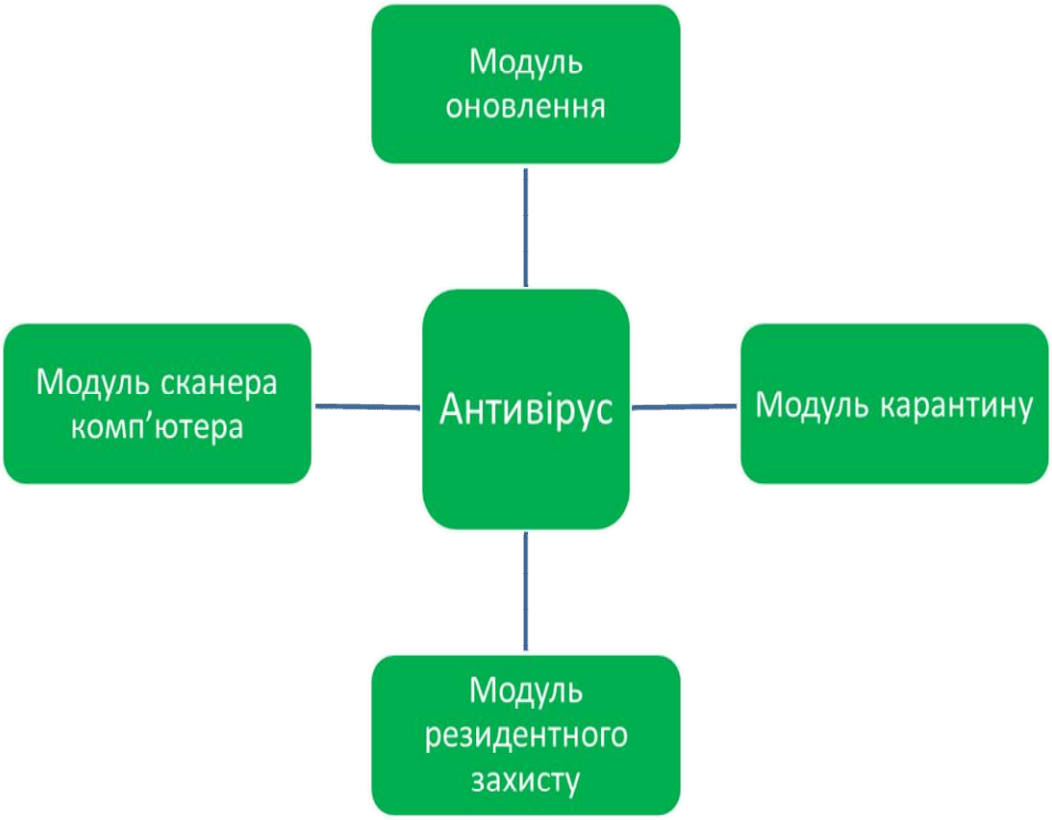

Рисунок 4.1 – Архітектура додатку для виявлення шкідливого програмного коду

Алгоритм роботи антивірусу показано на рис.4.2.

Рисунок 4.2 – Алгоритм роботи антивірусу

Сканер має гнучку систему настроювань, яка прийдеться по смакові як простим, так і просунутим користувачам. Антивірус забезпечує повноцінний захист комп'ютера від шкідливого ПЗ, а система Process Control – постійно контролює всі процеси користувача, що дозволяє запобігти зараженню системи. Величезну користь у пошуках вірусів приносять монітори, які в резидентному режимі постійно відслідковують і блокують усі вірусоподібні дії програм і будь-які операції із зараженими об'єктами (наприклад, копіювання документів, запуск програм і т. ін.). Розрізняють антивірусний монітор і поштовий. Антивірусна програма повинна перевіряти систему, тверді або знімні диски. Результати перевірки заносяться в журнал.

Пошук шкідливих програм в дипломному проекті здійснюється двома методами: сигнатурним і евристичним методами. Евристичний аналіз здійснюється на основі нейронної мережі.

4.1 Моделювання евристичного аналізатора шкідливих програм

ШП – це, з одного боку, файл із певним вмістом, з іншого боку – сукупність дій, що виконуються в ОС, із третьої – сукупність кінцевих ефектів в ОС. Тому й ідентифікація ШП може бути зроблена на різних рівнях: по ланцюжках байт, по діях, по впливу на ОС і т.д. Відповідно до цього виділяються наступні способи збору даних для розпізнавання ШП, кожний з яких реалізує окремий агент-детектор і за допомогою яких можна виявити більшість вірусів:

- монітор списку автозавантаження;
- монітор завантажувального розділу;
- монітор розділу HKLM\SOFTWARE\icrosoft\ Windowsnt\Current Version\ Winlogon реєстру;
- монітор системного файлу C:\windows\ system32\drivers\etc\hosts.

Для розв'язання задачі розпізнавання ШП у складі ЕА (евристичного аналізатора), який виконує імовірнісне розпізнавання на основі зваженої оцінки деякої кількості ознак, пропонується використовувати штучні нейронні мережі. Схематично модель такого ЕА ШП складається з наступних блоків (рис. 4.3).

Рисунок 4.3 – Модель евристичного аналізатора шкідливих програм

Розглянемо кожний з блоків.

Блок моніторингу. Функцією даного блоку є моніторинг поведінки шкідливих і не шкідливих об'єктів з метою одержання протоколу їх роботи (послідовностей виклику API функцій і переданих їм аргументів).

Блок порівняння. Даний блок розглядає протоколи роботи декількох програм від блоку моніторингу і порівнює їх. Результатом роботи буде безліч однакових фрагментів (ознак) у протоколах різних програм одного сімейства.

Бібліотека ознак. Даний блок зберігає в собі всі ознаки, виявлені блоком порівняння, і веде статистику їх зустрічальності. На основі даної статистики кожній ознаці привласнюється рейтинг, що характеризує зустрічальність даного ознаки. Таким чином, фрагмент, який був знайдений у протоколах усіх програм, буде мати найбільший рейтинг, а фрагмент, який знайдений у найменшій кількості програм – найменший.

Блок прийняття рішень (БПР). Функція даного компонента – прийняття рішення про приналежність або неприналежність розглянутої програми до деякого сімейства шкідливих програм. Блок моніторингу є системою, що моделює штучне оточення для роботи програм.

Основною задачею даного блоку є збір наступних даних:

- імена викликуваних у процесі роботи програми функцій WinApi;
- передані досліджуваною програмою аргументи при виклику функцій;
- послідовність виклику функцій.

На вхід блоку надходить об'єктний файл, який запускається на виконання в штучно створеному середовищі. Під час виконання програми оброблювачами викликуваних нею функцій ведеться збір даних і формується протокол. По завершенню виконання програми сформований протокол видається на вихід блоку для наступної обробки. У якості модуля моніторингу використовувався емулятор imul, розроблений Лабораторією Касперського. Блок порівняння реалізує алгоритм, спрямований на виявлення загальних для досліджуваних програм особливостей поведінки. На вхід блоку надходять протоколи роботи файлів, що виконуються, отримані від емулятора. На виході необхідно одержати множину загальних фрагментів

для всіх протоколів, які були подані на вхід. Обробка відбувається у два етапи. Перший. Фільтрація вхідних протоколів і очищення їх від даних, які не несуть інформацію про події, пов'язані з діями над файловою системою, системним реєстром, процесами або роботою в Інтернет. Другий. Із протоколу викидається вся інформація про виклики, які не входять у наступний список:

- API для роботи з файловою системою;
- API для роботи з Internet;
- API для роботи з вікнами;
- API для роботи з реєстром.

Блок прийняття розв'язків є основним компонентом ЕА, який призначений для розпізнавання як ШП визначеного сімейства, так і не ШП і здатний виконувати класифікацію вхідних векторів на дві групи. Він може бути реалізований з використанням різних технологій, зокрема на основі ШНМ.

4.2 Виділення характерних ознак

Для розпізнавання ШП необхідно вибрати перелік ознак, за якими це розпізнавання буде проводитися. Для того, щоб ефективно розпізнати шкідливий код, необхідно скласти бібліотеку подій, якими є програмні дії, пов'язані із системними викликами, що приводять до змін у системі. Складання бібліотеки подій робиться на основі аналізу великої кількості ШП і виділення в них характерних фрагментів, що часто зустрічаються. Основною задачею є збір найбільш повного набору значимих ознак. Працююча програма виконує багато різних дій – змінює значення регістрів, прапорів процесора, областей пам'яті і т. ін. Але не всі з них повинні враховуватися при розпізнаванні вірусів.

Більшість існуючих систем поведінкового аналізу базуються на використанні унікальних подій, характерних тільки для одного окремо

взятого сімейства вірусів. Для одержання більш ефективного і достовірного результату таких подій повинно бути трохи більше їх кількості і чим більша їх унікальність – тим вище буде відсоток правильних спрацьовувань. Найбільш пріоритетними при виборі повинні бути наступні типи подій:

- події, пов'язані із системним реєстром та файловою системою;
- події, пов'язані з роботою мережі;
- події, пов'язані з маніпулюванням вікнами;
- події, пов'язані з маніпулюванням процесами;
- події, пов'язані з установкою перехоплювачів системних подій.

4.3 Реалізація сигнатурного методу

Програмний додаток антивірусу реалізований мовою C++ у середовищі Microsoft Visual Studio. Розглянемо деякі фрагменти коду. Антивірус повинен сканувати диски, каталоги, файли на виявлення вірусу (рис.4.4, 4.5).

```
#include "stdafx.h"
#include unsigned
 long total = 0,
cured = 0;
void walk(char *Dir, char *Mask) {
 _WIN32_FIND_DATA buf;
HANDLE h;
SetCurrentDirectory(Dir);
if ((h=FindFirstFile(Mask, &buf))==INVALID_HANDLE_VALUE)
 return;
while (1) {
if (buf.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY) {
if (buf.cFileName[0] != '.')
walk(buf.cFileName, Mask );
}
else {
if (infected(buf.cFileName)) {
cure(buf.cFileName); cured++;
}
total++; }
if (!FindNextFile(h, &buf))
break;
}
FindClose(h);
SetCurrentDirectory(".."); }
```

Рисунок 4.4 – Процедури рекурсивного сканування каталогів

```
// Поиск сигнатуры в файле
Function ScanFile(AFileName : string) : Boolean;
begin
SetStatusBarText(AFileName);
LoadFileToBuffer(AFileName);
if SearchSign('75 2C 2A 65 07 BA 37 6C', 4096, 0) >= 0
then Result:=true else Result:=false;
FreeBuffer;
end;
```

Рисунок 4.5 – Лістинг коду для детектування і лікування поштового хробака E-Worm.Avron.a

У даному антивірусі використовується три методи виявлення вірусів, з додаванням нових методів підвищиться ефективність антивірусу. Також, дуже важливо постійно обновляти базу даних вірусів, тому що постійно з'являються нові віруси (рис.4.6).

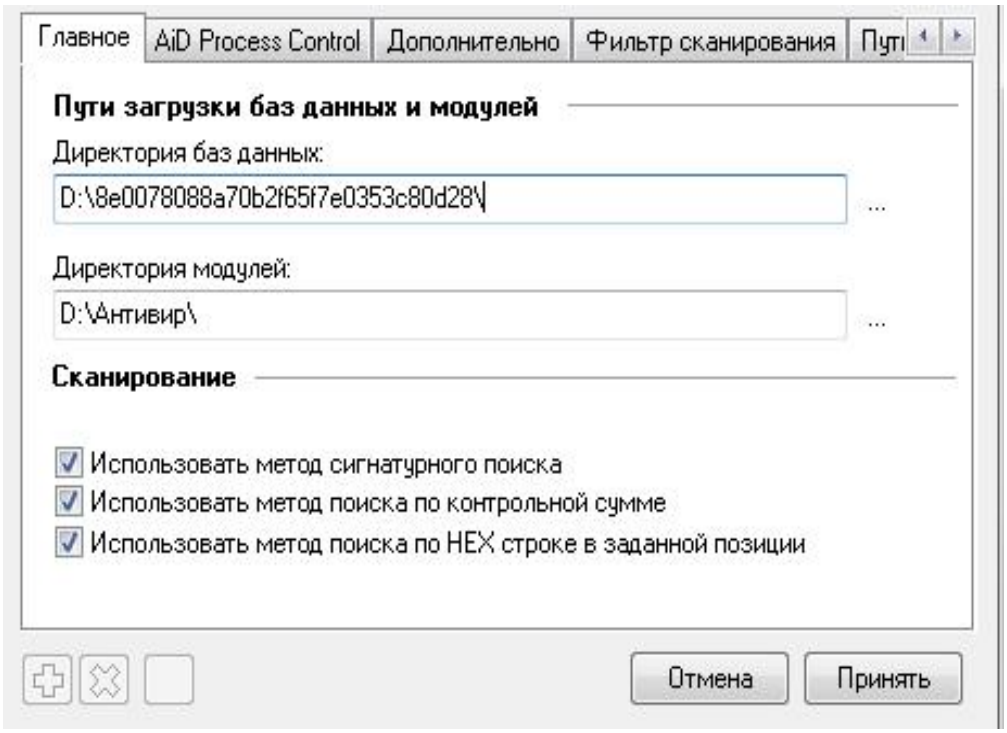

Рисунок 4.6 – Відновлення бази даних вірусів

4.4 Інструкція роботи з програмою

У головному вікні програми знаходяться вкладки до різних функцій і можливостей програми (рис.4.7).

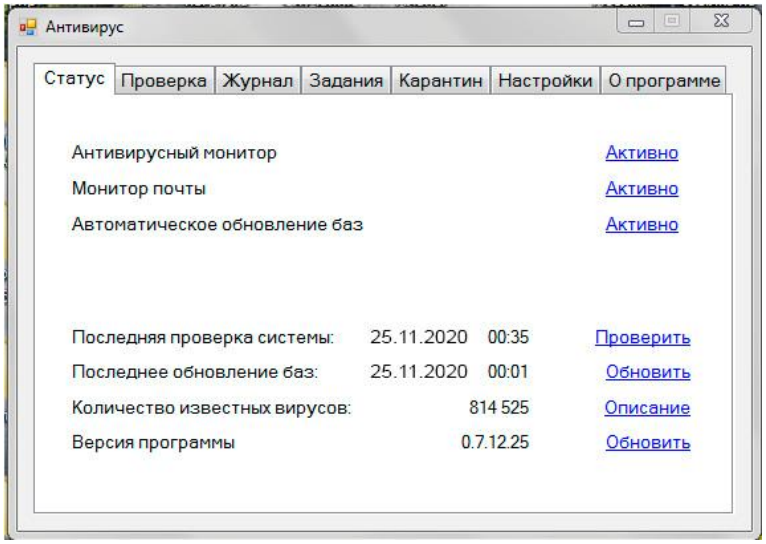

Рисунок 4.7 – Головне вікно

Програма має наступні можливості:

- відключити/включити Антивірусний монітор, Монітор пошти і Автоматичне відновлення антивірусних баз;
- переглянути інформацію про останню перевірку системи (і запустити перевірку), про останнє відновлення баз (і обновити бази), про кількість відомих вірусів (переглянути список і опис), про версію програми (запустити відновлення).

Якщо нажати на вкладку «Перевірка», з'явиться можливість запуску (припинення, скасування) перевірки системи (рис.4.8). Тут же ми можемо вибрати область перевірки, є можливість зберігати результати в журнал і виключити комп'ютер по завершенню перевірки.

Під час перевірки виводиться чисельна інформація про перевірені і заражені файли, час перевірки, а також індикатор тривалості перевірки.

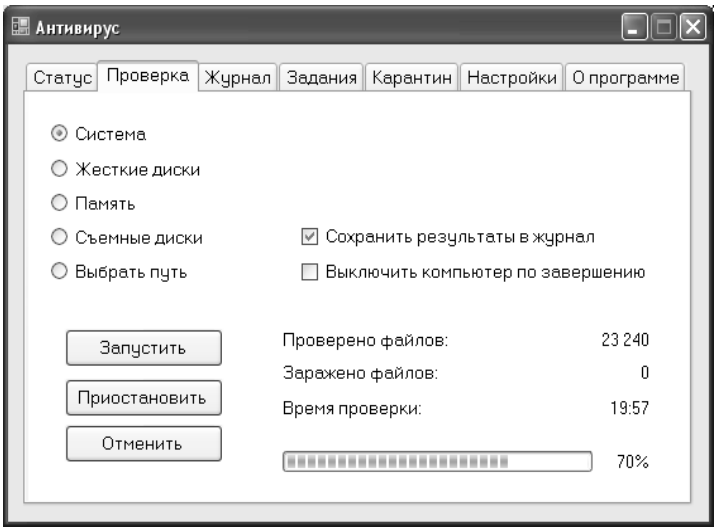

Рисунок 4.8 – Вкладка «Перевірка»

Якщо нажати на вкладку «Журнал», з'явиться інформація з бази даних про всі записані в журналі події (перевірка системи, відновлення програми, відновлення антивірусних баз) (рис.4.9).

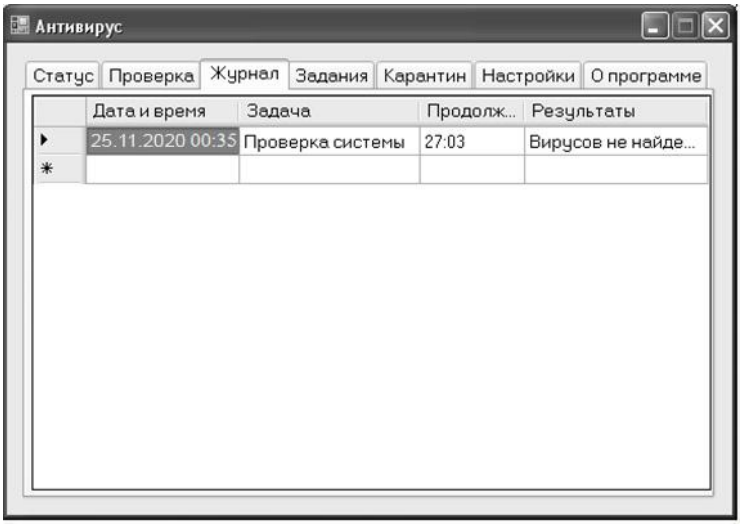

Рисунок 4.9 – Вкладка «Журнал»

Якщо нажати на вкладку «Задача», з'явиться інформація з бази даних про задачу для програми (наприклад, запуск перевірки системи 25.11.2020 в 16:00 перебуває в статусі «Очікування»). Коли задача виконається, статус

зміниться на «Виконане». Можна додати нову задачу клацанням правої кнопки миші і вибрати з контекстного меню «Додати задачу» (рис.4.10).

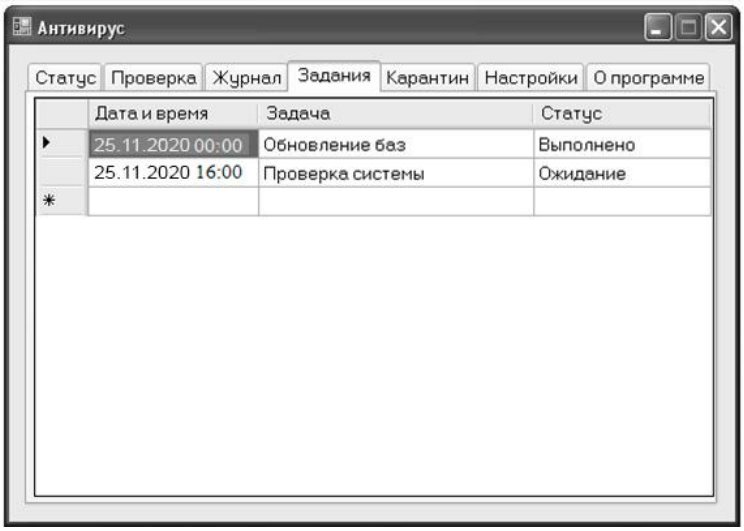

Рисунок 4.10 – Вкладка «Завдання»

Якщо нажати на вкладку «Карантин», з'явиться інформація про файли, які перебувають у карантині. Файли можна вилучити, відновити, спробувати вилікувати клацанням правої кнопки миші, вибравши з контекстного меню відповідні пункти (рис.4.11).

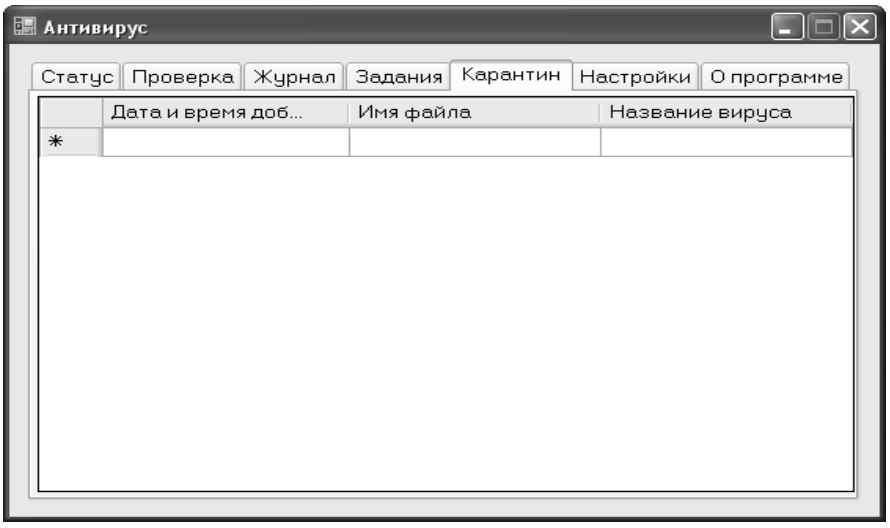

Рисунок 4.11 – Вкладка «Карантин»

Якщо нажати на вкладку «Настроювання», з'явиться можливість гнучко настроїти програму.

Перевіряти архіви, макроси, настроїти рівень евристичного аналізу, настроїти частоту автоматичного відновлення, включити/виключити монітори, інтеграцію в контекстне меню в Windows, автозапуск. Також можна настроїти послідовність дій програми, якщо знайдений вірус (Лікувати, Помістити в карантин, Вилучити, Запитати в користувача). Настроювання автоматично зберігаються, якщо вибрати будь-яку іншу вкладки в головному вікні програми або з контекстного меню при клацанні на праву кнопку миші (рис.4.12).

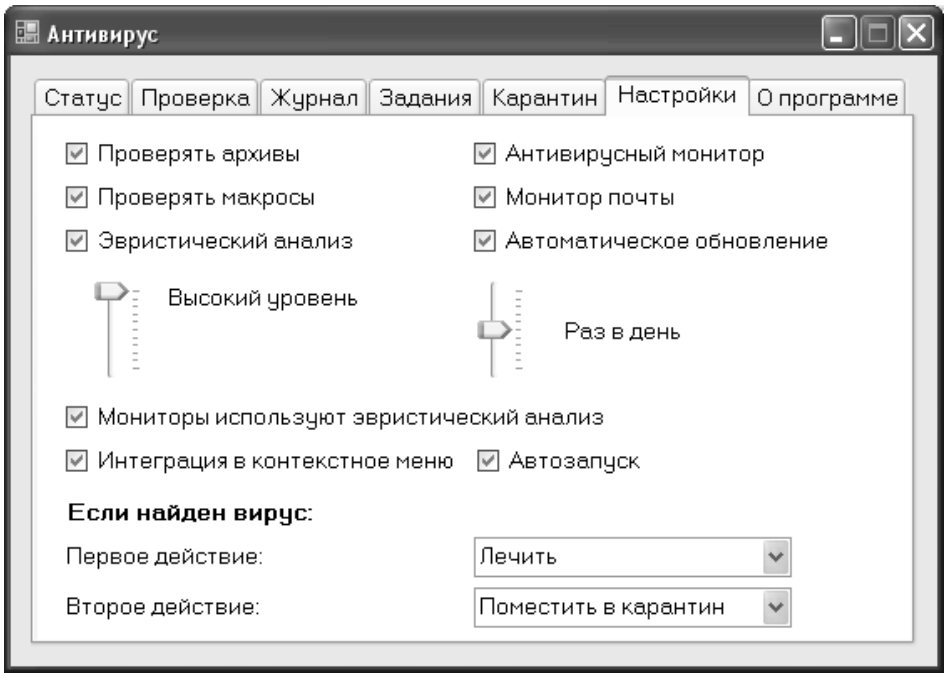

Рисунок 4.12 – Вкладка «Настроювання»

Якщо нажати на вкладку «Про програму», з'явиться інформація про версію програми. База вірусів зображена на рис. 4.13.

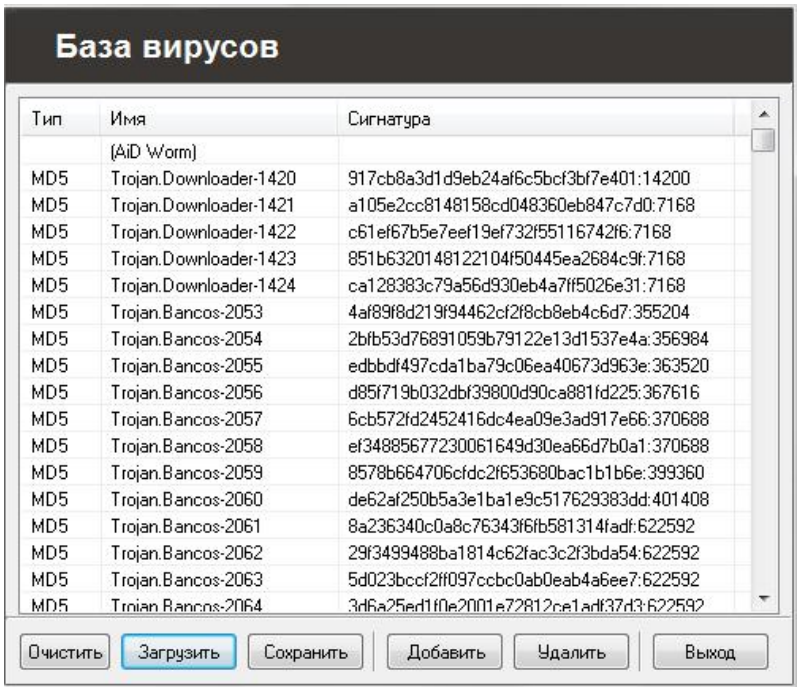

Рисунок 4.13 – База вірусів

4.5 Створення bat-вірусу

Bat-віруси – це шкідливий код, записаний у звичайному блокноті. Створити bat-віруси дуже легко, але в той же час, вони являють собою серйознішу загрозу. Простенький код, записаний у текстовому редакторі може завдати серйозної шкоди комп'ютеру, аж до повної деструкції ПК.

Відкриваємо довільну папку, у прикладі це буде папка C:/Intel, натискаємо праву кнопку миші: Створити → Текстовий документ.

У текстовому документі, що відкрився, пишемо наступний код:

@encho off

FOR /L %%i IN (1,1,3) DO md %%i

@encho off – відключення режиму відображення на екрані інформації про роботу команд.

FOR /L %%i – цикл по параметрах IN (1,1,3), перша цифра позначає назву першої папки, друга позначає хід циклу (приріст одиниці), третя цифра – загальну кількість створюваних папок.

DO md %%i – створення порожніх папок, місце розташування нових папок у тій же папці, де був запущений .bat-вірус.

Після запису коду зберігаємо текстовий документ у форматі .bat

Потім запускаємо Batvirus.bat. Запуск дозволений тільки із правами адміністратора, тому що код написаний в .bat файлі. Даний код створює 3 порожні папки. Якщо в коді замість цифри 3 поставити, наприклад 1000, то в такому випадку створюється 1000 порожніх папок. У такому випадку система може зависнути. Bat-вірус можна сховати. Для цього Batvirus.bat додаємо в архів, у властивостях архіву вказуємо «Створити Sfx-архів» (рис. 4.14). У результаті створюється архів Winrar Warcraft.exe, що саморозпаковується.

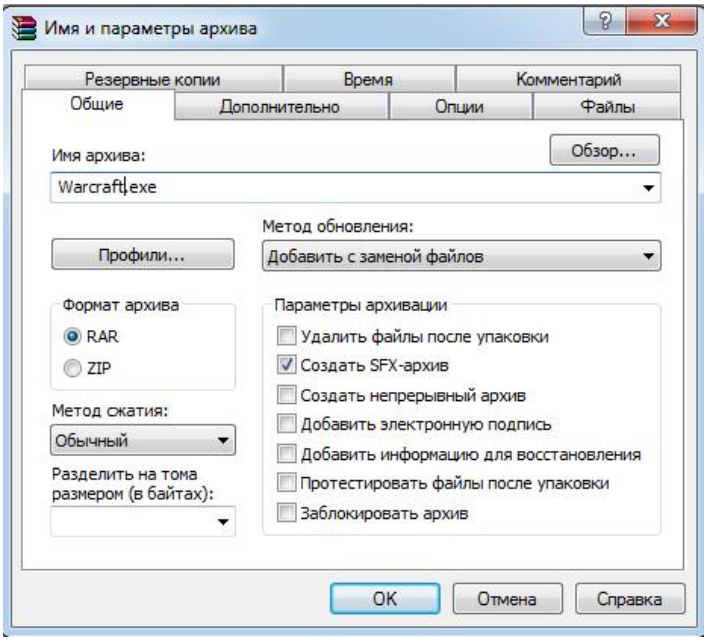

Рисунок 4.14 – Параметри архіву

4.6 Створення шкідливої програми

Однією із шкідливих програм є спеціальна програма для скимінга клавіатури користувача. Скимінг клавіатури – це запис усіх зроблених користувачем натискань клавіш на клавіатурі з метою одержання важливої

інформації (наприклад: паролі від соціальних мереж, пошти, номера кредитних карток і т.д.).

Принцип роботи скимінга клавіатури:

- відправляємо файл.exe користувачеві, чиї конфіденційні дані нас цікавлять;
- користувач відкриває даний .exe і скимінг починає свою роботу.

Запуск проводиться із правами адміністратора. Після запуску додатка не з'являється ніякої форми і у пуску не з'являється значок додатка, тому що додаток зроблений невидимим. Його можна побачити лише в запущених процесах, натискаємо на клавіатурі одночасно «Ctrl + Alt + Del» і з'являється «Диспетчер задач Windows», натискаємо на вкладку «Процеси». На вкладці, що з'явилася, видний процес запущеного додатка 123.exe (рис.4.15).

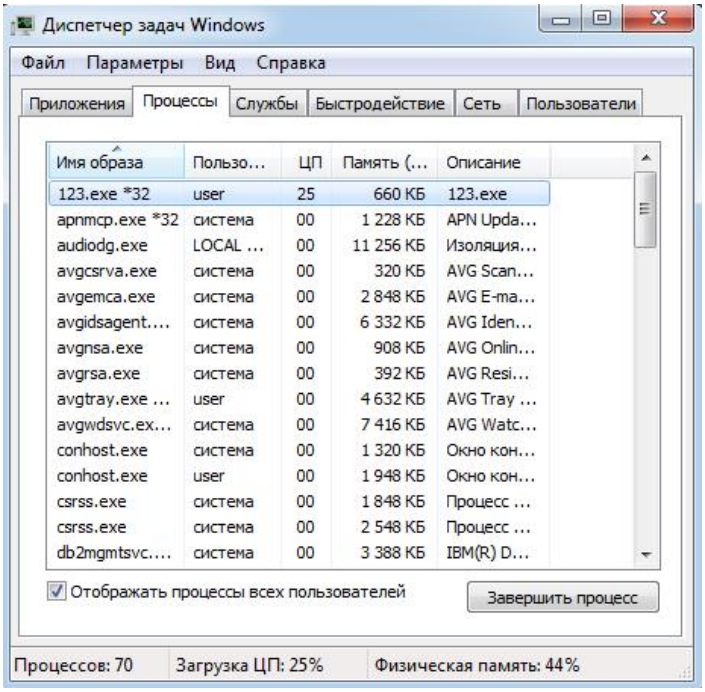

Рисунок 4.15 – Запущений додаток 123.exe в «Диспетчерові задач»

Після запуску додатка він автоматично записує всі набрані набори клавіш і зберігає їх в умовний документ log.txt, що знаходиться на диску C.

Для прикладу в текстовому редакторі набрали довільний текст «WARNING VIRUS» (рис.4.16).

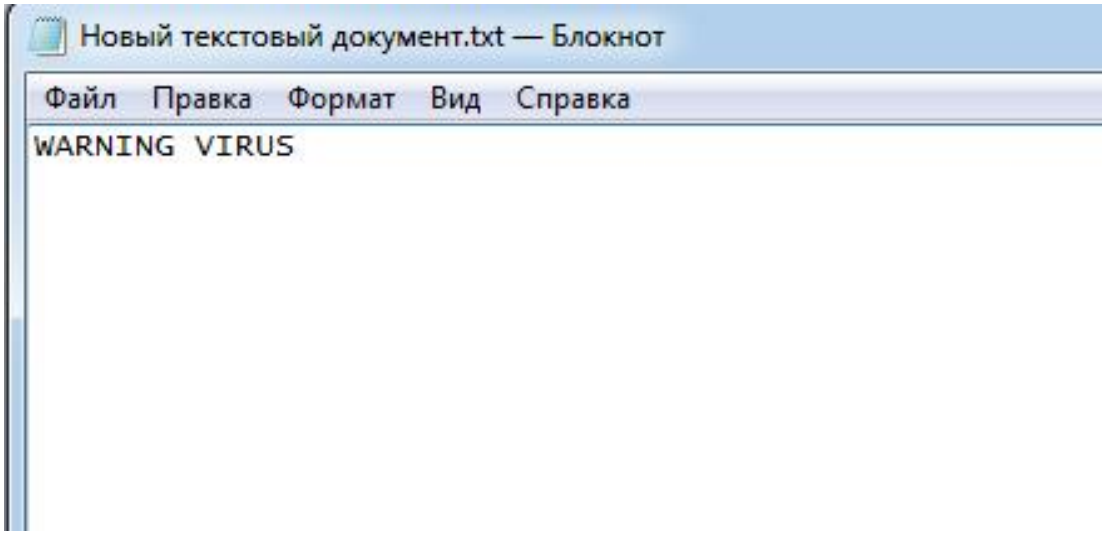

Рисунок 4.16 – Запис «WARNING VIRUS» у текстовий документ

Тепер у Диспетчерові задач відключаємо процес 123.exe. І заходимо на диску С у текстовий документ log.txt. Як видно з рис.4.17, набраний текст записаний в текстовий документ log.txt.

4.7 Тестування антивірусу на прикладі сигнатури bat-вірусу.

Протестуємо роботу антивірусу на прикладі сигнатури створеного bat-вірусу. Для цього занесемо сигнатуру вірусу в базу і перевіримо антивірусом. Щоб знайти сигнатуру заархівованого bat-вірусу Warcraft.exe, скористуємося програмою MD5 Filechecker.exe. Запускаємо програму і у пошуку вказуємо вірус, далі натискаємо «розрахувати» і з'являється контрольна MD5-сума (рис.4.17).

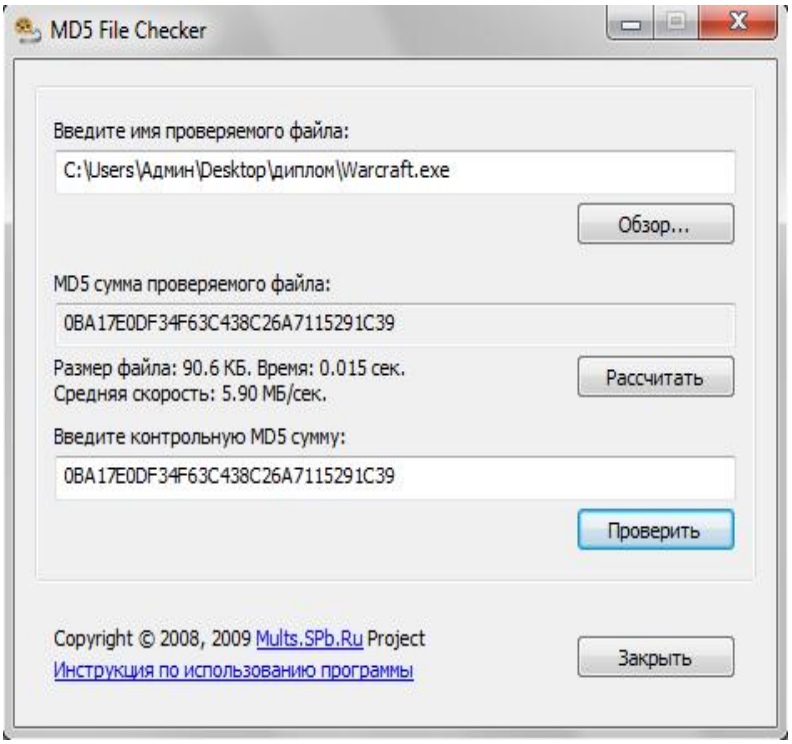

Рисунок 4.17 – Розрахунки контрольної MD5-суми файлу Warcraft.exe

Далі записуємо в базу вірусів нову сигнатуру. Указуємо назву і контрольну суму MD5 файлу 1.exe (рис.4.18).

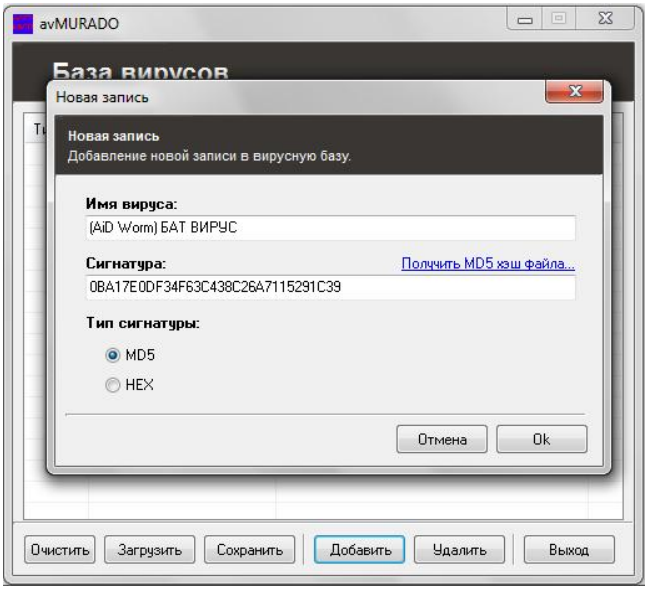

Рисунок 4.18 – Запис нової сигнатури

І в базі вірусів з'являється нова сигнатура. Зберігаємо базу і у сканері вказуємо директорію знаходження нової створеної бази, що включає сигнатуру заархівованого bat-вірусу 1.exe (рис.4.19).

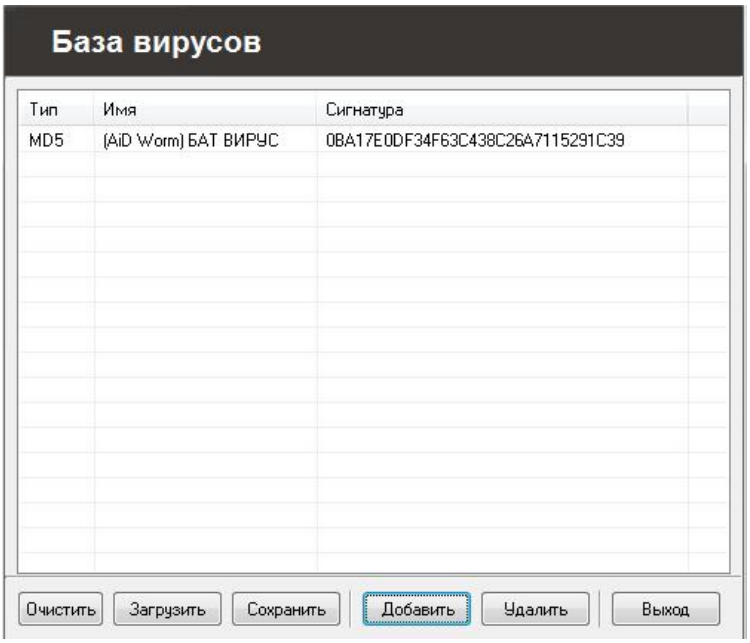

Рисунок 4.19 – Нова сигнатура в новій базі вірусів

У такий же спосіб, знаходимо сигнатуру MD5 створеної шкідливої програми і додаємо знайдену сигнатуру в базу вірусів.

Для моделювання роботи евристичного аналізатора був обраний наступний набір інструментів:

- моделювання блоку моніторингу проводилося з використанням спеціального програмного забезпечення емулятора системи Microsoft Windows – imul, яке емулює роботу центрального процесора, API функцій ОС і її внутрішніх структур і ідеально підходить для задачі моніторингу поведінки програм і збору необхідних даних;
- модуль аналізу даних виконує порівняння ознак ШП, які призначені для перетворення і дослідження інформації від емулятора системи;

– моделювання ШНМ проводилося мовою програмування C#. Реалізована ШНМ одержує в якості вхідних даних файл з навчальною вибіркою.

Даний етап роботи припускає запуск ШП на емуляторі і одержання протоколів їх роботи. Для експерименту були взяті наступні ШП:

Сімейство троянських програм, призначених для викрадення паролів, включаючи наступні модифікації: Trojan-psw.Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw.Win32.Ldpinch.bdl; Trojan-psw.Win32. Ldpinch.bik; Trojan-psw.Win32. Ldpinch.bki; Trojan-psw.Win32. Ldpinch.bkk.

Дані модифікації були запущені на емуляторі, і для кожної з них був отриманий протокол. Протоколи були проаналізовані за допомогою блоку порівняння, який вилучив з них службову інформацію про системні виклики, що не змінюють систему. Відфільтровані протоколи порівнювалися попарно. Приклад роботи програми показаний на рис.4.20. Результатом роботи даної програми з'явилася множина загальних для всіх вхідних протоколів фрагментів (характерних поведінкових ознак). Програма виявила наступні ознаки для набору троянських програм із сімейства Trojan-psw.Win32.Ldpinch.

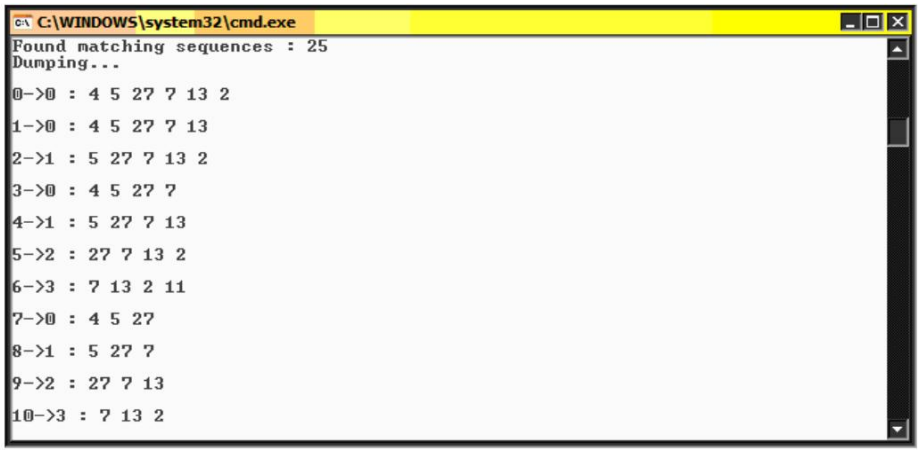

Рисунок 4.20 – Вікно програми Блоку порівняння

Ознака 1 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.azw; Trojan-psw.Win32.Ldpinch. bdl; Trojan-psw.Win32. Ldpinch.bik; Trojan-psw.Win32. Ldpinch.bkk.

Ознака 2 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bdl; Trojan-psw.Win32.Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki; Trojan-psw.Win32.Ldpinch.bkk.

Ознака 3 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bdl; Trojan-psw.Win32.Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki; Trojan-psw.Win32.Ldpinch.bkk.

Ознака 4 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bdl; Trojan-psw.Win32.Ldpinch.bik; Trojan-psw. Win32.Ldpinch.bkk.

Ознака 5 – зустрічається в наступних модифікаціях ШП: Trojan-psw.Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bdl; Trojan-psw.Win32.Ldpinch.bki.

Ознака 6 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bki.

Ознака 7 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bdl; Trojan-psw.Win32.Ldpinch.bik; Trojan-psw. Win32.Ldpinch.bki; Trojan-psw.Win32.Ldpinch.bkk.

Ознака 8 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki; Trojan-psw. Win32.Ldpinch.bkk.

Ознака 9 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bik; Trojan-psw .Win32.Ldpinch.bki; Trojan-psw. Win32.Ldpinch.bkk.

Ознака 10 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki; Trojan-psw. Win32.Ldpinch.bkk.

Ознака 11 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki.

Ознака 12 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup; Trojan-psw.Win32.Ldpinch.azw; Trojan-psw. Win32. Ldpinch.bik; Trojan-psw.Win32.Ldpinch.bki; Trojan-psw. Win32.Ldpinch.bkk.

Ознака 13 – зустрічається в наступних модифікаціях ШП: Trojan-psw. Win32.Ldpinch.aup: Trojan-psw.Win32.Ldpinch.azw: Trojan-psw.Win32. Ldpinch.bki.

Таким чином, було виявлено 13 характерних поведінкових ознак для набору троянських програм із сімейства Trojanpsw.Win32.Ldpinch і підраховані рейтинги їх зустрічальності:

Ознака 1: 4/6 = 0.66;
Ознака 2: 6/6 = 1;
Ознака 3: 6/6 = 1;
Ознака 4: 5/6 = 0.83;
Ознака 5: 4/6 = 0.66;
Ознака 6: 3/6 = 0.5;
Ознака 7: 6/6 = 1;
Ознака 8: 5/6 = 0.83;
Ознака 9: 5/6 = 0.83;
Ознака 10: 5/6 = 0.83;
Ознака 11: 5/6 = 0.83;
Ознака 12: 5/6 = 0.83;
Ознака 13: 3/6 = 0.5.

Далі були сформовані навчальні вибірки з рейтингів зустрічальності ознак у шкідливих і не шкідливих об'єктах і запущений процес навчання як ШНМ.

Подальша обробка отриманих даних, а також розпізнавання нових програм проводилося за допомогою НМ.

Розпізнавання шкідливих програм сімейства Trojanpsw.Win32.Ldpinch У якості тестового завдання розпізнавання ШП було взято раніше розглянуте сімейство Trojan-psw.Win32.Ldpinch. Для навчання нейронної мережі сформуємо навчальну вибірку з рейтингів зустрічальності ознак у шкідливих і не шкідливих об'єктах, представлену в таблиці на рис. 4.21.

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	1	1	0.83	0.66	0.5	1	0.83	0.83	0.83	0.83	0.83	0.5
2	0.66	1	1	0.83	0.66	0.5	1	0.83	0.83	0.83	0.83	0.83	0.5
3	0.66	1	1	0.83	0.83	0.66	0	1	0	0	0	0	0
4	0.66	1	1	0.83	0	0	1	0.83	0.83	0.83	0.83	0.83	0
5	0	1	1	0	0.66	0.5	1	0.83	0.83	0.83	0.83	0.83	0.5
6	0.66	1	1	0.83	0	0	1	0.83	0.83	0.83	0.83	0.83	0
7	0	0	0	1	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0.66	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0.5	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0.5
12	0	0	0	0	0.83	0.5	0	0	0	0	0	0	0
13	0	0	0	0	0	0.83	0.66	0	0	0	0	0.83	0

Рисунок 4.21 – Навчальна вибірка для розпізнавання шкідливих програм сімейства Trojan-psw.Win32.Ldpinch

Далі розглянуто систему евристичного аналізатора на основі багатошарової нейронної мережі.

Реалізована програма містить зрозумілий користувальницький інтерфейс (рис. 4.22), де для логічності певні його складові об'єднані в групи: «Режими роботи», «Вхідні параметри», «Режими навчання».

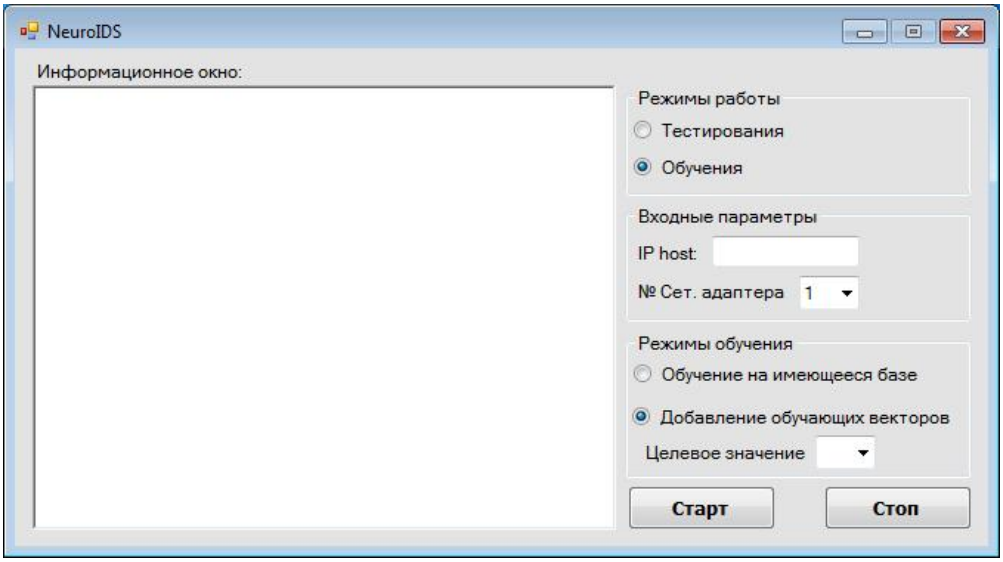

Рисунок 4.22 – Інтерфейс програми

Інтерфейс програми складається з наступних складових:

- інформаційного вікна – в ньому виводиться інформація, отримана в результаті роботи системи;
- група «Режими роботи» включає в себе два перемикача, що дозволяють вибрати режим роботи системи «Тестування» або «Навчання»;
- група «Вхідні параметри» включає в себе поле «IP host» для введення ip адреси хоста і поля «№ Сет. адаптера »для вибору номера використовуваного мережевого адаптера.

Група «Режими навчання» включає в себе два перемикача, що дозволяють вибрати режим навчання системи «Навчання на наявній базі» або «Додавання навчальних векторів». А також поле «Цільове значення» використовується в другому режимі навчання відповідно для вибору цільового значення навчання (1 – атака, 0 – нормальна поведінка).

Кнопка «Старт», натискання якої переводить систему з режиму очікування в один з можливих режимів роботи.

Кнопка «Стоп», натискання якої призводить до завершення виконання роботи програми.

Режим навчання. Для того, щоб запустити програму в режимі навчання, необхідно виконати наступний перелік дій.

- запустити на виконання додаток;
- у групі «Режими роботи» встановити перемикач напроти поля «Навчання»;
- у групі «Режими навчання» встановити перемикач навпроти одного з можливих варіантів;
- якщо вибраний режим «Навчання на існуючій базі», то після натискання кнопки «Старт», система виконає навчання, потім збереже вагові коефіцієнти і порогові значення у файл. І по закінченню видасть інформаційне вікно з повідомленням, що підтверджує успішність навчання (рис. 4.23).

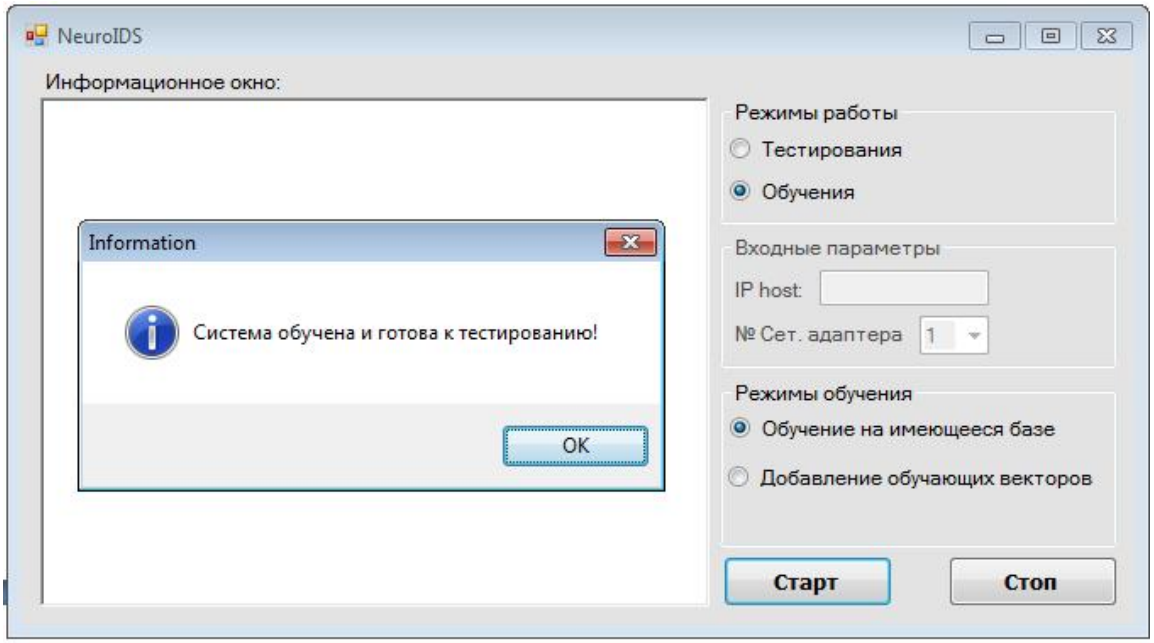

Рисунок 4.23 – Результат виконання навчання на існуючій базі

Якщо вибраний режим «Додавання навчальних векторів», то в групі «Вхідні дані» необхідно заповнити поле «IP host», вести ip адреси хоста, мережевий трафік якого необхідно аналізувати, і в полі «№ Сет. адаптера» вказати номер використовуваного мережевого адаптера.

У групі «Режими навчання» в полі «Цільове значення» вказати мету навчання (1 – шкідлива програма, 0 – нормальна поведінка).

Натиснути кнопку «Старт». Після того як система формує навчальні вектора і додасть їх в базу, в інформаційне вікно виведеться результат аналізу даних, і в підсумку вона видасть інформаційне вікно з повідомленням (рис. 4.24), підтверджуючим успішність виконання даної процедури.

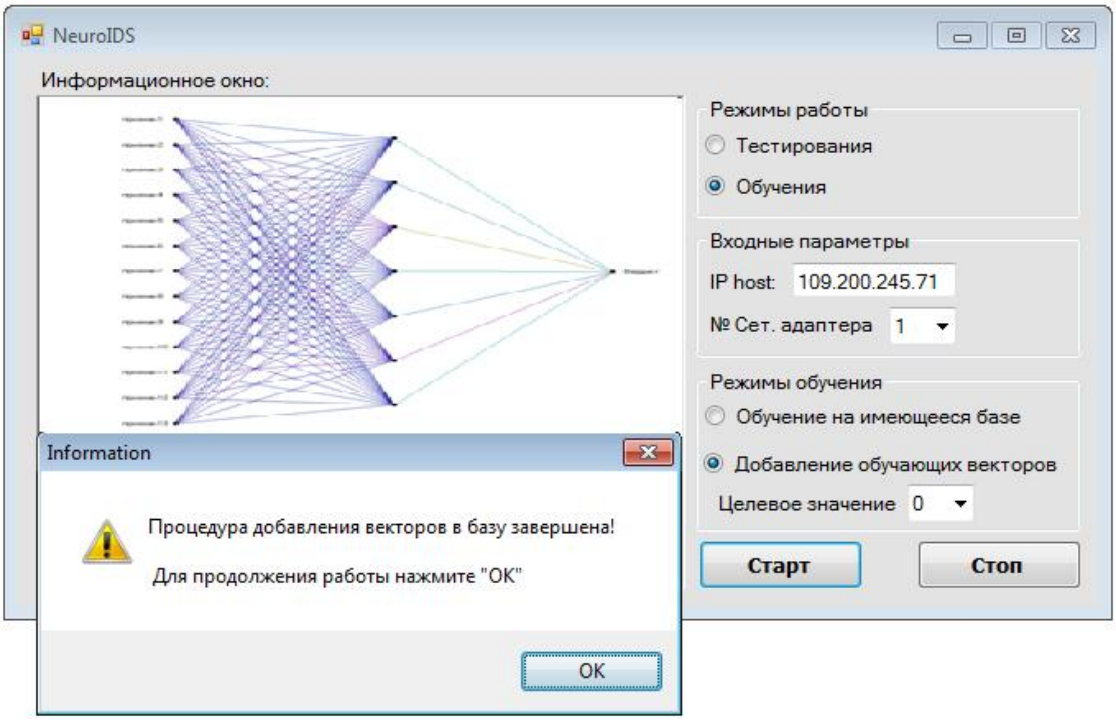

Рисунок 4.24 – Результат виконання програми при додаванні навчальних векторів в базу

Режим тестування. Для того, щоб протестувати програму в режимі тестування, необхідно виконати наступний перелік дій.

- запустити на виконання додаток;
- у групі «Режими роботи» встановити перемикач навпроти поля «Тестування»;
- у групі «Вхідні дані» необхідно заповнити поле «IP host», ввести ip адреси хоста, мережевий трафік якого необхідно аналізувати, і в полі «№ Сет. адаптера» вказати номер використовуваного мережевого адаптера;
- натиснути кнопку «Старт»;
- після того, як система закінчить процес тестування, в інформаційне вікно виведеться результат аналізу даних, і в підсумку вона видасть інформаційне вікно з повідомленням (рис. 4.25), яке підтверджує завершення виконання даної процедури.

Далі виконаємо обробку даних за допомогою створеної нейронної мережі і зробимо класифікацію нових зразків ШП: Trojan-psw.Win32. Ldpinch. cgi; Trojan-psw.Win32.Ldpinch.byc. Перевіримо правильність розпізнавання, подавши на вхід нейромережі дані про ознаки невідомих їй об'єктів.

Результати розпізнавання наведені на рис. 4.25 і 4.26. З результатів розпізнавання видно, що навчена нейронна мережа успішно впоралася з покладеною на неї задачею: обидві модифікації були розпізнані вірно.

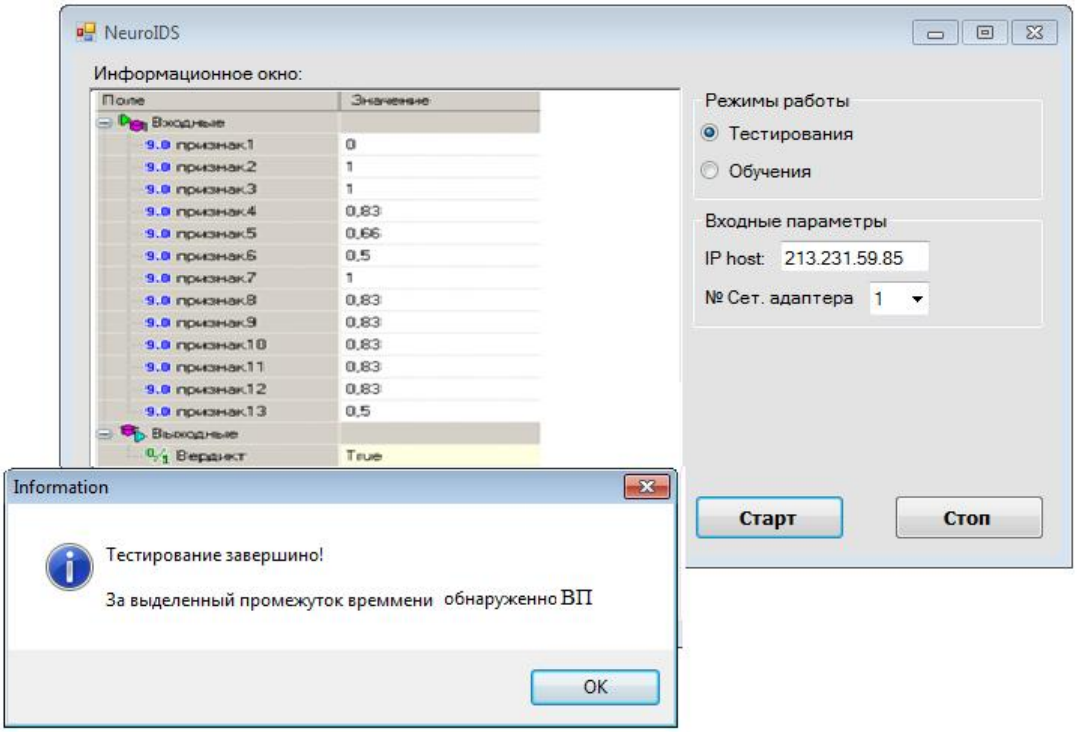

Рисунок 4.25 – Результат виконання програми в режимі тестування. Розпізнавання нової модифікації Trojan-psw.Win32.Ldpinch.cgi

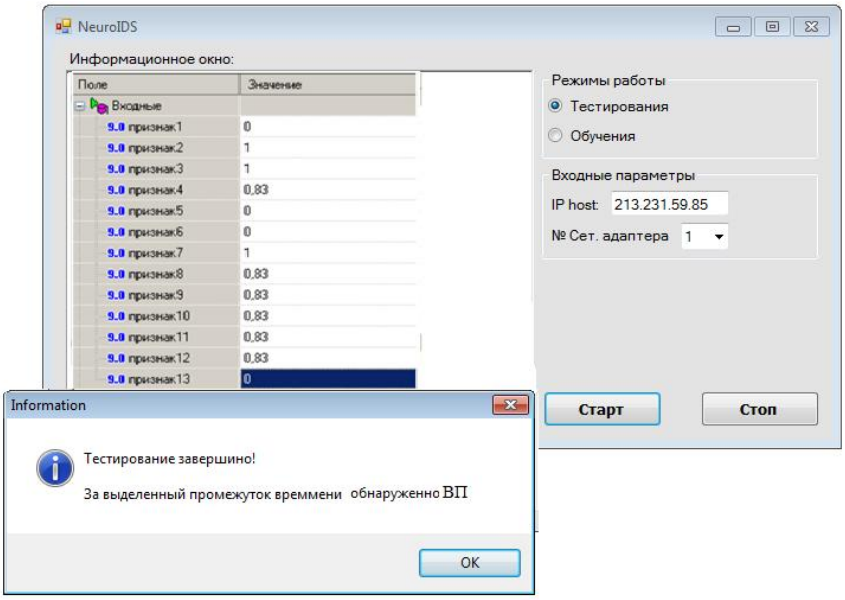

Рисунок 4.26 – Розпізнавання нової модифікації Trojan-psw.Win32.Ldpinch.byc

ВИСНОВКИ

При виконанні магістерської роботи було розроблено програмний додаток для виявлення шкідливого програмного коду, який складається з бази наявних вірусів і антивірусного сканера.

Пошук шкідливого програмного коду засобами розробленого додатку дозволяє користувачу обрати аналізуєму директорію, та використовуючи сигнатури зазначених в базі вірусів, виявляти заражені файли та відносити їх до карантину. Пошуковий сканер розробленого додатку для виявлення шкідливого програмного коду використовує сигнатурний метод пошуку вірусів MD-5 та виконує евристичний аналіз на основі нейронної мережі для пошуку шкідливих програм. Застосована нейронна мережа має один прихований шар, 13 входів і один вихід. Для навчання багатошарової мережі MLP використовувався алгоритм зворотного поширення помилок. В розробленому додатку створена база даних вірусів з можливістю додавання нових сигнатур вірусів. Пошук шкідливого програмного коду проводиться в .exe, .dll, .com, .bat, .scr форматах файлів.

В роботі було проведено дослідження класів існуючих вірусів, огляд сучасних антивірусних програм, дослідження і вибір методів виявлення шкідливих програм, виконано моделювання евристичного аналізатора шкідливих програм на основі багатошарової нейронної мережі, розроблено додаток для виявлення шкідливого програмного коду. В якості цільової платформи обрана операційна система сімейства Windows.

Тестування додатку для виявлення шкідливого програмного коду проводилося на прикладі деяких вірусів: троянська програма, яка вилучає паролі користувача і bat-вірус. Тестовий Bat-вірус був створений в текстовому редакторі Блокнот та заархівований. В результаті тестування розробленого додатку всі наявні і запропоновані віруси та шкідливий програмний код були виявлені.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Климентьев К. Е. Компьютерные вирусы и антивирусы: взгляд программиста. М., ДМК Пресс, 2013, 656 с.
2. Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. М., ДМК Пресс, 2012, 592 с.
3. Ф.Файтс, П.Джонстон, М.Кратц. Компьютерный вирус: проблемы и прогноз. М., Мир, 1993, 176 с.
4. Класифікація комп'ютерних вірусів – Поняття та класифікація комп’ютерних вірусів. Програмні засоби захисту від комп’ютерних вірусів. Класифікація антивірусів. URL: https://sites.google.com/site /siteallaboutviruses /klasifikacia-komp-uternih-virusiv (дата звернення 24.05.2020).
5. Романец Ю. В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. М., Радио и связь, 2001, 376 с.
6. Ознаки зараження комп’ютерними вірусами та як цьому запобігти | Безпечне місто. URL: http://safe-city.com.ua/oznaky-zarazhennya-komp-yuternymy-virusamy-ta-yak-tsomu-zapobigty/ (дата звернення 02.06.2020).
7. Соколов А.В., Шаньгин В.Ф. Защита информации в распределенных корпоративных сетях и системах. М., ДМК Пресс, 2002, 596 с.
8. Рейтинг антивирусов 2020. Какой антивирус лучше? URL: https://softcatalog.info/ru/obzor/reyting-antivirusov (дата звернення 03.06.2020).
9. Щербаков А.Ю. Компьютерная безопасность. М., Изд. Молгачева С.В., 2001, 352 с.
10. Типи антивірусних програм – Direside. URL: https://sites.google. com /site/ diresideinaction/tipi- antivirusnih-program (дата звернення 06.06.2020).

11. Сучасні антивірусні програми та принцип їх роботи – Безпечний Інтернет. URL: https://sites.google.com/site/bezpecnijinternet1999/sucasni-antivirusni-programi-ta-princip-ieh-roboti (дата звернення 08.06.2020).
12. Контрольна сума. Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0_%D1%81%D1%83%D0%BC%D0%B0 (дата звернення 09.06.2020).
13. Шаньгин В.Ф. Информационная безопасность и защита информации. М., ДМК Пресс, 2014, 702 с.
14. Технології виявлення вірусів. URL: https://studfile.net/preview/5368373/page:3/ (дата звернення 10.06.2020).
15. Антивірусні програми – захист локальної мережі. URL: https://www.sites.google.com/site/zahistlokalnoiemerezi/zahist/antivirusni-programi (дата звернення 10.06.2020).
16. Цветков В.Я., Булгаков С.В. Эвристический анализ как инструмент информационной безопасности. Журнал Современные наукоемкие технологии. 2010. №1 С. 53-53 URL: https://top-technologies.ru/pdf/2010/ 1/22.pdf (дата звернення 24.06.2020).
17. Принципи роботи антивіруса. URL: https://kursy.zp.ua/main/principi/uk/virus-principi-roboti-antivirusa.aspx (дата звернення 12.06.2020).
18. Нейронні мережі. URL: https://ukrbukva.net/page,3,92840-Neiyronnye-seti.html (дата звернення 12.08.2020).
19. Нейронні мережі. URL: https://www.asimovinstitute.org /neuralnetwork-zoo. (дата звернення 14.07.2020).
20. Порівняльна характеристика нейронних мереж. URL: https://cyberleninka.ru/article/n/obzor-svyortochnyh-neyronnyh-setey-dlya-zadachi-klassifikatsii -izobrazheniy (дата звернення 16.08.2020).