Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2023 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи”

Виконав здобувач вищої освіти
IV курсу, групи КБ-19
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Лисенко О.О.
«____» _____ 2023 р.

Керівник проекту
кандидат фізико-математичних наук, доцент
_____ Якименко Н.М.
«____» _____ 2023 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*“*Інформаційні технології*”*_____

Спеціальність *125 “*Кібербезпека*”*

Освітньо-професійна (освітньо-наукова) програма _*“*Кібербезпека*”*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2023 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Лисенку Олександру Олександровичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи*

2. Керівник роботи *Якименко Наталія Миколаївна, канд. фіз.-мат. наук, доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 12-02 від 5.01.2023 року

3. Строк подання студентом роботи до захисту *23.05.2023 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи кібербезпеки для антивірусного захисту операційної системи*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.

2.Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи кібербезпеки в промислову експлуатацію.

6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Структурна схема системи кібербезпеки	*1 аркуш*
Функціональна схема системи кібербезпеки	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Дата видачі завдання « 17 » січня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2023 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2023 р.	
3.	Розробка моделі компонента	20.03.2023 р.	
4.	Розробка структур даних	25.03.2023 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2023 р.	
6.	Програмування алгоритмів	10.04.2023 р.	
7.	Оформлення ПЗ	17.04.2023 р.	
8.	Попередній захист роботи	23.05.2023 р.	

Дата видачі завдання
« 17 » січня 2023 р.

Підпис керівника
_____ Якименко Н.М.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2023 р.

Підпис здобувача
_____ Лисенко О.О.
(прізвище та ініціали)

АНОТАЦІЯ

Лисенко О.О. Програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи. 125 Кібербезпека. Центральноукраїнський національний технічний університет. Кропивницький. 2023.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи кібербезпеки для антивірусного захисту операційної системи.

Метою розробки є програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи.

Результат роботи – програмна реалізація системи кібербезпеки для антивірусного захисту операційної системи.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10/11.

Програму розроблено в середовищі RAD Studio Delphi.

Ключові слова: кібербезпека, антивірусний захист, операційна система

ABSTRACT

Lysenko O.O. Cyber security system software for antivirus protection of the operating system. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2023.

In this graduation thesis for the first (bachelor) level of higher education, software is developed, which is intended for the cyber security system for antivirus protection of the operating system.

The purpose of the development is the software of the cyber security system for antivirus protection of the operating system.

The result of the work is the software implementation of the cyber security system for antivirus protection of the operating system.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on PCs of IBM PC architecture with Windows 10/11 OS.

The program was developed in the RAD Studio Delphi environment.

Keywords: cyber security, antivirus protection, operating system

ЗМІСТ

					ВКРБ-125.23.0012.00.00.ПЗ		
Вим	Арк.	№ докум.	Підп.	Дата			
Розроб.		Лисенко О.О.			Програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи	Літ. Б	Аркуш 1 / Аркушів 79
Перев.		Якименко Н.М.				ЦНТУ КБ-19	
Н.контр.		Гермак В.С.					
Затв.		Смірнов О.А.					

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

КМ	–	комп’ютерна мережа
КСАЗ	–	комплексна система антивірусного захисту
МЕ	–	міжмережний екран
ПЗ	–	програмне забезпечення
ПК	–	персональний комп’ютер
ACL	–	Access Control List
FTP	–	File Transfer Protocol
http	–	HyperText Transfer Protocol
POP3	–	Post Office Protocol Version 3
SMTP	–	Simple Mail Transfer Protocol
VLAN	–	Virtual Local Area Network

ВСТУП

Актуальність теми. Як правило домашній комп'ютер часто використовується не тільки для роботи у позаробочий час, але й для комп'ютерних ігор, особистої переписки, пошуку й перегляду інформації в Інтернет, для відтворення фільмів і музики. При цьому адміністрування домашнього комп'ютера в переважній більшості випадків виробляється винятково власними силами хазяїна. Тому всі програми, призначені для домашнього використання, мають прозорий інтерфейс, нескладні в установці й керуванні, обов'язково супроводжуються зрозумілою навіть для неспеціаліста документацією. Програми для забезпечення антивірусної безпеки також повинні відповідати всім перерахованим вимогам.

Серед необхідних для повноцінного й ефективного захисту домашніх комп'ютерів від шкідливого впливу програм можна виділити:

– Антивірусне програмне забезпечення.

– Програми для захисту від несанкціонованого доступу й мережних хакерських атак.

– Фільтри небажаної кореспонденції.

Перераховані програми можуть як входити в один комплекс по захисту домашнього комп'ютера, так і бути встановлені окремо. Головна перевага першого способу – це наявність єдиного інтерфейсу керування й продумане творцями програм взаємодоповнення кожного з модулів. Установка окремих програм, особливо різних виробників, тільки в деяких випадках може виявитися корисним, наприклад, коли необхідні специфічні функції, але жоден комплексний продукт не може їх забезпечити. У випадку домашнього користувача це зустрічається вкрай рідко і якщо потрібно встановити всі три модулі, то бажано це зробити за допомогою комплексного рішення.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем для антивірусного захисту операційної системи.

– Дослідження системи кібербезпеки для антивірусного захисту операційної системи.

– Програмна реалізація системи кібербезпеки для антивірусного захисту операційної системи.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі для антивірусного захисту операційної системи.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Основний і по сумісництву обов'язковий елемент в антивірусному захисті – це, безумовно, антивірусна програма. Без її не можна говорити про ефективну антивірусну безпеку, якщо мова йде про комп'ютер, здатний обмінюватися інформацією с іншими зовнішніми джерелами. Навіть дотримання користувачем всіх правил не гарантує відсутність шкідливих програм, якщо при цьому не використовується антивірус.

Антивірусне програмне забезпечення – це досить складний програмний комплекс, для його створення потрібні зусилля команди висококваліфікованих вірусних аналітиків, експертів і програмістів з багаторічним досвідом і досить специфічними знаннями й уміннями. Основна технологія антивірусної перевірки – сигнатурний аналіз має на увазі безперервну роботу з моніторингу вірусних інцидентів і регулярний випуск відновлень антивірусних баз. Через ці й інші причини, антивірусні програми не вбудовуються в операційні системи. Вбудованим може бути тільки найпростіший фільтр, що не забезпечує повноцінної антивірусної перевірки.

1.2 Область застосування

Сучасне антивірусне програмне забезпечення має вирішальне значення для захисту і збереження вашої особистої інформації.

Вкрай важливо, щоб викладачі, співробітники та студенти переконалися, що антивірусне програмне забезпечення встановлено та працює належним чином на їхніх персональних комп’ютерах. Існує кілька безкоштовних антивірусних програм, які можна придбати.

Багато з перелічених нижче постачальників пропонують як безкоштовну, так і платну версію свого програмного забезпечення. Ми пропонуємо вам завантажити або придбати їх безпосередньо на сайті постачальника або через авторитетний магазин, як-от Amazon, Best Buy тощо. Переконайтеся, що під час завантаження програмного забезпечення веб-сайт є надійним, наприклад download.cnet.com, support.apple. com або основного постачальника.

Ніколи не купуйте та не завантажуйте антивірусне програмне забезпечення на основі спливаючого вікна чи посилання, надісланого вам в електронному листі (якщо ви щойно не замовили його, і це повідомлення про встановлення). Якщо вам потрібен продукт цього постачальника, завжди переходьте на сайт оригінального постачальника або сайт розповсюдження, який завідомо придатний. чому Зловмисники часто використовують спливаюче вікно чи посилання електронною поштою, щоб спробувати обманом змусити вас завантажити програмне забезпечення, у якому вбудовано зловмисне програмне забезпечення, та/або спробувати викрасти вашу кредитну картку через підроблений процес покупки.

Існують антивірусні програми, які можна завантажити на телефони та планшети. Завжди купуйте ці продукти через схвалені та перевірені постачальниками сайти програмного забезпечення для телефонів/планшетів.

Щоб знайти антивірусне програмне забезпечення для своїх пристроїв, рекомендуємо переглянути параметри, перелічені в цих місцях:

– Пристрої iOS/Apple: iTunes, Apple App Store або www.apple.com.

– Пристрої Android/Google: магазин Google Play play.google.com/store.

– Пристрої Windows: www.microsoftstore.com.

З численним вибором антивірусних продуктів може здатися непосильним прийняти рішення, і не всі продукти зберігають свою ефективність. Вибираючи між різними антивірусними продуктами, враховуйте такі речі, як операційна система, бюджет, додаткові функції та простота використання. Окрім сайтів

постачальників, ось деякі інші веб-сайти, які слід розглянути для отримання інформації.

– AV тест:.

o https://en.wikipedia.org/wiki/AV-TEST;.

o https://www.av-test.org/en/about-the-institute/.

– Найкраща антивірусна програма для Android (https://www.av-test.org/en/antivirus/mobile-devices/android/).

– Найкраще антивірусне програмне забезпечення для Windows Home User (https://www.av-test.org/en/antivirus/home-windows/).

– 12 пакетів безпеки для Mac OS X (https://www.av-test.org/en/news/news-single-view/12-security-suites-for-mac-os-x-put-to-the- тест/).

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

У рамках виконання бакалаврського проектування проведемо огляд існуючих антивірусів.

Таблиця 2.1 – Основні вендори антивірусної індустрії

№	Назва	Країна	Російський сайт	Перевірка Online	Відправити вірус	Примітка
1	2	3	4	5	6	7
2	AhnLab (V3)	Південна Корея	–	Scan	–	3
3	Aladdin (eSafe)	Ізраїль	–	–	–	–
4	AVAST Software (Avast! Antivirus)	Чехія	+	File	Mail	1
5	ArcaVir (ArcaBit)	Польща	–	–	–	–
6	AVG Technologies (AVG)	Чехія	+	–	Mail	1,3
7	Avira (AntiVir)	Германія	+	–	Form	1
8	BullGuard	Данія	–	–	–	5
9	ClamAV (ClamAV)	Польща	+	–	Form	1,2
10	CA Inc. (Vet)	США	+	Scan	Form	–
11	Comodo AntiVirus	США?	–	–	–	1
12	CyberDefender (CyberDefender)	США	–	–	–	–
14	Eset Software (ESET NOD32)	Словаччина	+	Scan	Mail	3

Продовження таблиці 2.1

1	2	3	4	5	6	7
15	Fortinet (Fortinet)	США	–	–	–	–
16	FRISK Software (F–Prot)	Ісландія	–	–	Form	–
17	F-Secure (F–Secure)	Фінляндія	–	Scan	Form	3,5
18	G DATA (G DATA Software)	Германія	–	–	File	5
19	Ikarus Software (Ikarus)	Австрія	+	–	Mail	3
21	McAfee (VirusScan)	США	+	Scan	Mail	3
22	Microsoft (Security Essentials)	США	+	Scan	Form	1
23	MKS Antivirus (mks_vir)	Польща	–	Scan	–	–
24	Norman (Norman Antivirus)	Норвегія	–	–	Form	3
25	Panda Security (Panda Platinum)	Іспанія	+	Scan	Mail	–
26	PC Tools AntiVirus (PC Tools)	Ірландія	+	–	Form	1
27	Prevx (Prevx)	Великобританія	–	–	–	2
28	Protector Plus (Proland)	Індію	–	–	–	–
29	RISING Antivirus (Rising)	Китай	+	–	–	–
32	Softwin (BitDefender)	Румунія	+	Scan	Mail	3
33	Sophos (SAV)	Великобританія	–	–	–	–
34	Spybot–S&D (Safer Networking)	Ірландія	–	–	–	1
35	Steganos (Steganos)	Германія	–	–	–	–
36	Sunbelt Software (Antivirus)	США	–	–	Form	–
37	Symantec (Norton Antivirus)	США	+	Scan	Form	–
38	Quick Heal (CAT labs)	Індію	+	–	Mail	–
39	Trend Micro	Японія	+	–	Form	–

Продовження таблиці 2.1

41	Vexira (Central Command)	США	–	–	Mail	–
42	VirusBuster	Угорщина	–	–	–	–
43	ViRobot (HAURI)	Корея	–	–	Form	–
44	Webroot (Webroot Software)	США	–	–	–	–
45	Zillya! Антивирус	Україна	+	–	Form	1

Примітки:

1. Наявність безкоштовної версії.

2. Наявність безкоштовного антивірусного сканера.

3. Безкоштовні утиліти для видалення вірусів.

4. Позиціонується як HIPS рішення+сторонні сканери (VBA32, BitDefender, DrWEB).

5. Використовує кілька антивірусних движків.

Розглянемо докладно існуючі антивірусні програми.

Dr.web

Переваги:

– Потужні засоби виявлення як відомих, так і невідомих типів вірусів.

– Мала кількість споживаних системних ресурсів і можливість роботи навіть на застарілих моделях комп'ютерів.

– Інтуїтивно зрозумілий інтерфейс, з можливістю настроювань кожного компонента, антивірусу Dr. Web для Windows.

– Використання унікальних методик перевірки дозволяє детектувати вірусні атаки на стадії проникнення.

– Невеликий розмір дистрибутива антивірусної програми.

– Компактність відновлень антивірусної програми й наявність автоматичного відновлення по мережі інтернет дозволяє детектувати самі нові віруси й не допустити їхнього проникнення на комп'ютер.

Состав антивірусу Dr.Web:

– Сканери із графічним і консольним інтерфейсом, дозволяє робити глибокі сканування файлів і операційних систем по запитах користувача.

– Резидентний сторож Spider Guard – здійснює перевірку одержуваних файлів у реальному режимі часу.

– Поштовий сканер Spider Mail – призначений для перевірки поштового трафіка як вихідного, так і вхідного.

– Автоматична програма відновлень Dr.Web Update, утиліта автоматичного відновлення.

– Планувальник завдань Dr. Web – багатофункціональний засіб для запуску завдань компонентам Dr. Web.

Недоліки:

– Не вистачає потоків при скануванні.

– Більша завантаженість ОЗП.

– Пропускає віруси.

Panda Antivirus

Переваги:

– Контроль інтернет-трафіка, переданого через будь-який браузер: Panda antivirus pro 2010 сканує всю вхідну й вихідну інформацію, відправлену користувачеві через браузер. Даний антивірус сполучимий з усіма існуючими на сьогоднішній день версіями оглядачами інтернет-сторінок.

– Сканування електронної пошти, переданої й одержуваної як за допомогою веб-інтерфейсу, так і за допомогою поштових клієнтів. У цей час антивірус Panda підтримує такі відомі програми для відправлення листів як The Bat і Microsoft Outlook.

– Сучасні проактивні технології виявлення й знешкодження вірусних погроз: антивірус Panda здатний вчасно виявити вірус, що перебуває на комп'ютері, або шкідливу програму й видалити їх без заподіяння шкоди функціональності операційної системи.

– Ефективна перевірка трафіка інтернет-пейджерів: антивірус панда повністю контролює всі що відправляються й одержувані користувачем короткі повідомлення. Програма може сканувати трафік таких популярних програм як ICQ, Windows Live Messenger, AOL і MSN.

– Низькі вимоги до ресурсів комп'ютера: даний антивірус може функціонувати навіть на малопотужних комп'ютерних пристроях. Крім того, нова версія продукту відрізняється досить великим зниженням споживання пам'яті ПК.

Недоліки:

– Не занадто зрозуміла система навігації.

– Сильно пропускає.

– Часто видаляє без попиту.

– Повний ефект непомітний.

Norton Antivirus

Переваги:

– Захист від вірусів.

– Захист від програм-шпигунів.

– Огляд електронної пошти.

– Захист від фішингу.

– Захист ідентифікаційних даних.

– Брандмауер.

– Резервне копіювання й відновлення.

– Автоматичне відновлення.

– Підвищення продуктивності.

– Виявлення руткітів.

– Убудована підтримка.

– Автоматичне відновлення Symantec.

Недоліки:

– Вірус може бути вже всередині.

– При онлайн підтримці, майже завжди пропонують скористатися платними послугами.

McAfee

Переваги:

– Захист від різних шпигунських програм.

– Захист від великої кількості вірусів.

– Резервний «бекап» даних.

– Захист від докучливих спамвих повідомлень.

– Захист від фішингу.

– Батьківський контроль.

– Двосторонній брандмауер.

– Захист ваших особистих відомостей.

– Рейтинги безпеки інтернет-сайтів.

Недоліки:

– Дорогі ключі.

– Повільно працює.

– Ловить слабко.

– Здатний чистити архіви винятково з однорівневою глибиною вкладення.

– При скануванні сформованих особливим образом архівів у форматі lha у движку антивірусу виникає помилка переповнення буфера, через що нападаючий одержує можливість виконати на машині шкідливий код.

– Відсутня підтримка російської мови.

NOD32

Переваги:

– Удосконалена система захисту від спроб зовнішнього впливу, здатних негативно вплинути на безпеку комп'ютера.

– Технологія threatsense – аналіз файлів для виявлення вірусів, програм-шпигунів (spyware), непрошеної реклами (adware), phishing-атак і інших погроз.

– Перевірка й видалення вірусів із заблокованих для записів файлів (приміром, захищені системою безпеки windows бібліотеки dll).

У системі присутня модульна структура, що дозволяє більш гнучко контролювати роботу антивірусу:

– AMON – антивірусний монітор. AMON контролює все потенційно небезпечні дії на комп'ютері.

– IMON і EMON – модулі, що забезпечують роботу з електронною поштою, що перевіряють вхідні повідомлення. Дозволяють забезпечити перевірку трафіка, що проходить не тільки за допомогою протоколу POP3, але й перевірити листа, що проходять через інтерфейс MAPI. IMON працює як монітор для перевірки інтернет-трафіка. Він дозволяє перевірити на наявність шкідливих програм всі веб-сайти, які ви відвідуєте.

– DMON – модуль, що служить для перевірки ваших документів, створених за допомогою MS Office. Крім того, DMON дозволяє забезпечувати перевірку файлів, завантажуються автоматично через Internet Explorer (наприклад, елементи Microsoft Active). Модуль є додатковим рівнем захисту до AMON.

Недоліки:

Коли він сам обновляє сигнатуру система висне.

– Видаляє деякі віруси без згоди, виходить відразу віконце що видалив.

– При навантаженні системи він вимикається й виходить віконечко що Система піддається небезпеки.

– Погано шукає в реєстрі.

– Не виявляє віруси які проникли в консоль.

Avira AntiVir

Переваги:

– Вірусна база даних антивірусу AntiVir Personal нараховує близько 150 тисяч вірусів.

– Резидентний сервіс Virus Guard автоматично відслідковує роботу з файлами (наприклад при завантаженні з інтернету).

– Avira AntiVir Personal знаходить макровіруси й уміє лікувати файли уражені ними.

– Захист від програм з автоматичним дозвоном (Costly Dialers).

– Захист від троянів, хробаків і інших шкідливих програм.

– Легке керування.

– Помічник відновлень AntiVir Personal Edition.

– Можливість роботи з розкладу.

Недоліки:

– Самозахисту майже немає.

– Якщо ставити на заражений комп'ютер то вона неспроможна (якщо сильно).

– Помилкові спрацьовування.

– Не дуже гарний firewall.

Avast!

Переваги:

– Високопродуктивний модуль антивірусного сканування.

– Захист від руткітів.

– Захист від шпигунських програм.

– Російський інтерфейс.

– Екрани в реальному часі.

– Антиспамовий фільтр avast!

– Антивірусне сканування.

Недоліки:

– Перед звичайними вірусами працює нормально, але при більше складних, не працює.

– Знаходить не всі віруси.

– Не шукає віруси у флешках.

BitDefender 8 Free Edition

Переваги:

– Захищає комп'ютер у режимі реального часу від відомих вірусів, шпигунського ПЗ (Spyware) і інших шкідливих програм.

– Блокує невідомі віруси із застосуванням сучасних проактиввих технологій.

– Виявляє й видаляє руткіти.

– Щоденне відновлення вірусної бази.

Недоліки:

– Віруси не лікує (тільки видаляє, разом з файлами).

– Мова англійська.

– Відсутність резидентного монітора.

– Досить високі системні вимоги.

Microsoft Security Essentials

Переваги:

– Знищення найнебезпечніших шкідливих програм.

– Видалення відомих вірусів.

– Антивірусний захист у режимі реального часу.

– Видалення відомих шпигунських програм.

– Антишпигунський захист у режимі реального часу.

Недоліки:

– Відсутність проактивною захисту.

– Неможливість відключити функцію відправлення даних в Microsoft SpyNet.

– Працює тільки на ліцензійній ОС.

– Кудись, щось відправляє при підключення до інтернету (вантажить систему).

– Бувають труднощі при видаленні самого антивірусу.

Comodo Antivirus

Переваги:

– Розширений евристичний аналіз і превентивний захист дозволяють перехоплює невідомі віруси.

– Захист у теперішньому часі, моніторинг дій і сканер на вимогу.

– Блокування інтернет-хробаків до того, як вони почнуть свою дію.

– Сканування електронної пошти.

– Автоматичне щоденне відновлення вірусної бази.

Недоліки:

– Саме віруси пропускає.

Outpost Antivirus Pro

Антивірус Outpost Antivirus Pro захищає вас від всіх типів відом і нового шкідливого ПЗ, запобігаючи зараженню або крадіжку ваших даних:

– Від файлових, поліморфних і макро-вірусів.

– Від поштових і мережних хробаків.

– Від шпигунського ПЗ, включаючи трояни й клавіатурні шпигуни (keyloggers).

– Від "дозвонщиков" (dialers) і рекламних програм (adware).

– Від ПЗ для захвата й віддаленого контролю над ПК (botware).

– Від хакерських утиліт і програм-жартів.

– Від схованих погроз (руткітів).

– Від шахрайських і заражених інтернет-сайтів.

– Від витоку паролів, адрес і номерів банківських карт.

Недоліки:

Користувач може скачувати тільки відновлення сигнатур, але не самі оновлені версії компонентів антивірусу.

Trend Micro Antivirus plus Antispyware

Переваги:

– Антивірусний захист.

– Захист від шпигунських програм.

– Сканування безпеки електронної листів.

– Захист від руткітів.

– Проактивний захист від вторгнень.

Недоліки:

– Самозахист не дуже.

– Погано лікує.

VBA32 Antivirus

Антивірус для локальних комп'ютерів, що надійно й швидко працює в реальному масштабі часу (монітор) і по запиту (сканер)

До складу Vba32 Personal входять:

– Файловий сканер.

– Файловий монітор.

– Система автоматичних відновлень.

– Розширення меню Провідника Windows.

– Плагіни для поштових клієнтів "The Bat!", Microsoft Outlook, Microsoft Exchange Client.

– POP3-фільтр для перевірки прийнятих поштових повідомлень незалежно від використовуваного поштового клієнта.

– Script-filter, що захищає від виконання шкідливих скриптів в Microsoft Internet Explorer і Microsoft Outlook Express, а також будь-яких інших додатках, що використовують технологію Microsoft Windows Scripting Host.

Недоліки:

От його тест:

"+" – був знайдений і знешкоджений

"–" – не був знайдений

1.Email-Worm. Win32.Bagle.....–

2.Trojan-PSW. Win32.LdPinch.....+

3.Worm.Win32. Feebs.....+

4.Trojan-Clicker. Win32.Costrat.....–

5.Trojan-Spy. Win32.Goldun.....–

6.Adware.Win32. Look2me.....–

7.Adware. Win32.New DotNet.....–

8.Backdoor. Win32.Haxdoor.....–

9.Trojan-Proxy. Win32.Xorpix.....–

10.Email-Worm. Win32.Scano.....+

Sophos Anti-Virus

Sophos Anti-Virus – Антивірусна програма. Як і покладено сучасному антивірусу, перевіряє все, що можна (і потрібно) – файли, що завантажуються з інтернету, а також файли на локальних і мережних дисках. Звичайно, є й функція автообновлення через інтернет. З особливостей варто відзначити швидку роботу й відносно невелике завантаження системних ресурсів за рахунок використання двоступінчастого контролю: основний час працює тільки крихітний модуль-монітор, що контролює кожний що відкривається файл. У випадку виявлення вірусу або троянської програми можна буде запустити більше важкий модуль, що зробить якісне "зачищення".

AVG Anti-Virus

Переваги:

– Автоматичні відновлення на увесь час користування антивірусом.

– AVG Resident Shield проводить сканування файлів під час їхнього відкриття й програм при їхньому запуску.

– AVG E-mail Scanner перевіряє всю вашу електронну пошту.

– AVG On-Demand Scanner дозволяє користувачеві сканувати комп'ютер на наявність вірусів, як за розкладом так і вручну.

– Турботливий обіг з інфікованими файлами (може лікувати заражені вірусами файли).

Недоліки:

– Великий розмір відновлень.

– Немає автоматичного лікування файлів.

– Помилкові спрацьовування.

– Великий розмір програми.

– Навантаження на систему в слабких комп'ютерів.

F-Secure Anti-Virus

Переваги:

– Захист комп'ютера від шкідливих програм. Перевірка на віруси й шпигунські програми захищає комп'ютер від програм, які можуть викрадати особисту інформацію, завдавати шкоди комп'ютеру або використовувати його в незаконних цілях. При виявленні шкідливих програм будь-якого типу вони за замовчуванням знешкоджуються, перш ніж встигнуть почати діяти.

– Захист від потенційно небезпечних змін у системі. Технологія DeepGuard аналізує вміст файлів і поводження програм, блокує нові й непізнані віруси, хробаки й інші шкідливі програми, які можуть внести потенційно небезпечні зміни в систему.

До потенційно небезпечних змін системи відносяться:

– зміни системних параметрів (реєстру Windows);

– спроби відключити важливі системні програми, наприклад програми забезпечення безпеки;

– спроби змінити важливі системні файли.

DeepGuard в F-Secure Anti-Virus постійно відслідковує подібні зміни й перевіряє кожну програму, що намагається їх здійснити.

Недоліки:

– Великий розмір антивірусу.

– Для нормальної роботи потрібний комп'ютер більш потужний.

2.2 Обґрунтування вибору засобів для побудови системи кібербезпеки та мови програмування

Embarcadero Delphi, раніше Borland Delphi і Codegear Delphi, – інтегроване середовище розробки ПЗ для Microsoft Windows, Mac OS, iOS і Android мовою Delphi (що раніше носила назву Object Pascal), створена спочатку фірмою Borland і на даний момент приналежна й розроблювальна Embarcadero Technologies. Embarcadero Delphi є частиною пакета Embarcadero RAD Studio і поставляється в чотирьох редакціях: Community (поширюється безкоштовно й має обмежену ліцензію на використання в комерційних цілях), Professional, Enterprise і Architect.

Delphi 10.4 Sydney

Випущено 26 травня 2020 року. RAD Studio Delphi 10.4 забезпечує значно поліпшену високопродуктивну нативну підтримку Windows, кращу продуктивність розробки, миттєві підказки code completion, прискорення виконання коду із синтаксисом керованих записів, поліпшення виконання паралельних завдань на сучасних багатоядерних CPU, а також містить більш 1000 виправлень багів, поліпшення продуктивності середовища й бібліотек і багато чого крім того.

Основні можливості Delphi 10.4.1:

− Істотні розширення для Windows: поліпшення для застосунків на моніторах 4K High DPI, інтеграція з новим WebView2 на базі Chromium, використання розширених title bars, таких же, як в Office, Explorer, Google Chrome.

− Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

− Істотне поліпшення Delphi Code Insight (без можливого блокування IDE – в окремому процесі), що допоможе при роботі з великими проектами.

− Тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання.

− Розширена підтримка бібліотек C++: ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode.

− Відладник Win 64 (на LLDB) і збирач для C++.

− Поліпшення для C++: Включена велика кількість поліпшень STL з Dinkumware.

− Підтримка Metal Driver GPU для macOS і iOS.

− Вбудований Fmxlinux.

− Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API. Реалізація компонента Media Player для macOS тепер використовує Avfoundation. Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

− Численні поліпшення швидкості й стабільності роботи нашої бібліотеки The Parallel Programming Library (PPL).

− Додані оновлені драйвери для FireBird, PostgreSQL і SQLite.

− Клієнтські бібліотеки HTTP і REST Client розширені застосунковими можливостями роботи з HTTPS. Також були розширені можливості підтримки Amazon AWS services

− У технологію Visual LiveBindings внесена безліч поліпшень, у тому числі швидкодії, що стосуться, застосунків на VCL і FireMonkey

RAD Studio 10.4 Короткий огляд:

– Істотні розширення для Windows. Створення застосунків, що чудово виглядають, із чіткими елементами інтерфейсу на 4k моніторах High DPI за допомогою нової гнучкої підтримки стилів елементів керування на екрані. Інтеграція із сучасними, безпечними web-технологіями від Microsoft – новим WebView2 на базі Chromium. Використання сучасних розширених title bars, таких же, як в Office, Explorer, Google Chrome, у своїх проектах. Істотні поліпшення надійності налагодження в новому відладнику для C++ Windows 64-bit.

– Зросла продуктивність розробки. Ріст продуктивності за рахунок миттєвої реакції підказок code completion у середовищі IDE. Краща сумісність із уже наявною кодовою базою, і спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю. Швидке зв'язування даних і візуальних елементів за допомогою розширеної технології Visual LiveBindings з підвищеною швидкодією. Просте використання розповсюджених бібліотек C++, наприклад, ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode. Оновлена підтримка Amazon AWS cloud.

– Поліпшення швидкодії і якості. Більш 1000 поліпшень швидкодії і якості. Краща ефективність коду за допомогою нового синтаксису custom managed records. Більш швидке виконання паралельних завдань на сучасних багатоядерних CPU. Переконаєтеся в прискоренні відображення на екрані з підтримкою Metal API на macOS і iOS. Краща сумісність із уже наявною кодовою базою й спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю.

Істотне поліпшення Delphi Code Insight

Як найбільше й головне поліпшення інструментів програмування Delphi за багато років, в 10.4 Delphi Code Insight реалізований через Language Server Protocol (LSP). LSP – це технологія генерації результатів для code completion, навігації й інших сервісів в окремому процесі. Це значить, що code completion і Code Insight одержать більш точні результати без блокування IDE. 10.4 забезпечує набагато більш високу продуктивність розроблювачів, які працюють із більшими проектами, що містять мільйони рядків коду.

Delphi Custom Managed Records

Ключове розширення мови Delphi: тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання. Управляйте тем, як ці структури створюються, копіюються й звільняються з допомогу вашого коду, який буде виконуватися у відповідний момент.

Це розширює потужність конструкцій records в Delphi, які використовуться щоб одержати більшу ефективність у порівнянні із класами.

Єдине керування пам'яттю

Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

У порівнянні з Automatic Reference Counting (ARC), це дає кращу сумісність із існучим кодом і спрощує написання компонентів, бібліотек і застосунків.

ARC модель керування пам'яттю model залишилася для керування рядками й посиланнями на тип інтерфейсу на всіх платформах. Для C++ це означає, що при створенні й звільненні Delphi-style класів в C++ використовується звичайне керування пам'яттю, як у будь-якого heap-allocated класу C++, що значно знижує складність коду.

Розширена підтримка бібліотек C++

В 10.4 ми портували багато популярних бібліотек C++ у C++Builder.

Забезпечивши оптимізовану підтримку бібліотек ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode, поряд із уже підтримуваними Boost і Eigen, які можуть бути додані за допомогою менеджера пакетів Getit.

Win 64-відладник і збирач для C++

В 10.4 з'явився новий відладник C++ для Windows 64-bit. Відладник заснований на LLDB і показує значне збільшення стабільності при налагодженні 64-bit застосунків поряд з новими відладочними можливостями, такими як перегляд і інспекція типів начебто рядків C++ і Delphi, а також колекцій STL, включаючи std::vector, std::map і інших. Крім того, згенерована для застосунку відладочна інформація має інший внутрішній формат, сприяючи більш стабільному й багатому на можливості процесу налагодження, більш докладним перегляду й інспекції в debug-time.

Підвищення якості й швидкодії інструментів

– Велика кількість поліпшень STL від Dinkumware.

– Поліпшені деякі найважливіші методи й області RTL, на базі поліпшень сумісності з популярними бібліотеками C++.

– Поліпшена підтримка Cmake.

– Велика кількість виправлень для підвищення стабільності і якості.

– Відновлення Windows API – Обновлено й додали безліч декларацій API щоб добитися ще більшої інтеграції із платформою Windows.

– Загальні вдосконалення в бібліотеці доступу до БД FireDAC, включаючи оновлені драйвера для FireBird, PostgreSQL і SQLite. Вибір статичного або динамічного підключення SQLite до застосунку.

Змінені стилі VCL для High DPI

В 10.4, архітектура стилізації VCL була суттєво розширена для підтримки High DPI і 4K моніторів. Тепер усі елементи UI на формі VCL автоматично масштабуться під відповідне до монітора дозвіл для показу форми. Був оновлений API стилізації для підтримки стилів high DPI.

Кожний графічний елемент UI може бути обраний з наборів різних масштабів і масштабований до потрібного DPI, що дає чітке зображення елементів UI на всіх моніторах.

Нові High DPI стилі й стилізація окремих VCL компонент

Обновлено велике число вбудованих і преміальних VCL стилів для підтримки нового режиму стилізації High-dpi. Це дозволяє вам створювати застосунку з відмінним дизайном для всіх моніторів.

Розроблювачі VCL застосунків тепер можуть використовувати трохи VCL стилів на різних формах в одному застосунку або в різних компонентів на одній формі. Це також включає стилізацію компонентів загальною темою для платформи. Крім застосункової гнучкості використання стилів, це дозволяє використовувати нестилізуємі компоненти із зовнішніх бібліотек в VCL застосунках, що використовуть стиль.

Поліпшена кроссплатформеність

– Додана підтримка Metal Driver GPU для macOS і iOS.

– Крім підтримки останнього iOS SDK, в RAD Studio 10.4 розроблювачі можуть задовольнити нові вимоги Apple до набору стартових екранів.

– Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

– Користувачам редакцій Enterprise або Architect доступна повна інтеграція Fmxlinux з IDE для створення клієнтських застосунків Linux з GUI.

– Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API.

– Реалізація компонента Media Player для macOS тепер використовує Avfoundation.

Оновлений менеджер пакетів Getit

Менеджер пакетів Getit в IDE був значно вдосконалений.

Дати випуску релізів пакетів тепер видні, і можливе сортування списку по цих датах; відбір тільки встановлених пакетів, контенту, доступного тільки при наявності підписки, багато чого іншого.

Універсальний інсталятор для установки Online і Offline

В 10.4 включений новий універсальний інсталятор, який використовує технологію на базі Getit. Цей інсталятор підтримує як online, так і offline (з ISO) варіанти установки.

Тепер обоє варіанта установки дозволяють вам указати початковий набір можливостей RAD Studio для установки, наприклад, свою комбінацію мов програмування й цільових платформ, мов інтерфейсу, і додавати до нього або видаляти непотрібне в будь-який момент.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне

забезпечення, яке призначено для системи кібербезпеки для антивірусного захисту операційної системи.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи кібербезпеки контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи кібербезпеки в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Антивірусне програмне забезпечення – це утиліта захисту даних, яка встановлюється в комп'ютерну систему з метою захисту від вірусів, шпигунського програмного забезпечення, зловмисного програмного забезпечення, руткітів, троянських програм, фішингових атак, спам-атак та інших онлайн-загроз.

Давайте дізнаємося трохи про вірус.

Вірус – це будь-яка небажана програма, яка потрапляє в систему користувача без його відома. Він може саморозмножуватися та поширюватися. Він виконує небажані та шкідливі дії, які в кінцевому підсумку впливають на продуктивність системи та дані/файли користувача. Комп'ютерний вірус можна розглядати як хворобу комп'ютера, так само як людські віруси, які викликають захворювання у людей.

А як же антивірус ?

Як видно з назви, антивірусне програмне забезпечення – це програма, яка працює проти вірусів. Він виявляє або розпізнає вірус, а потім, виявивши наявність вірусу, працює над видаленням його з комп'ютерної системи. Антивірусне програмне забезпечення працює як профілактичний засіб, тобто воно не лише усуває вірус, але й запобігає будь-якому потенційному вірусу від зараження комп'ютера в майбутньому.

Для чого потрібна антивірусна програма?

Система без антивіруса – це як будинок з відкритими дверима. Відкриті та незахищені двері привернуть у ваш дім усіх зловмисників та грабіжників. Подібним чином незахищений комп'ютер запрошує всі віруси в систему. Антивірус діятиме як зачинені двері з охоронцем для вашого

комп’ютера, відбиваючись від усіх зловмисних вірусів. Отже, ви залишите свої двері відкритими для зловмисників?

Якої шкоди вірус може завдати вашому комп’ютеру?

Якщо ваш комп’ютер атакує вірус, це може вплинути на ваш комп’ютер наступним чином:

– Уповільнює роботу комп'ютера.

– Пошкодити або видалити файли.

– Переформатуйте жорсткий диск.

– Часті збої комп'ютера.

– Втрата даних.

– Нездатність виконувати будь-які завдання на комп’ютері чи в Інтернеті.

Антивірусне програмне забезпечення – це як промінь яскравого світла у світі, повному темних вірусів. Кількість переваг, які вони пропонують, незліченна. Деякі з найбільш помітних переваг:

Захист від вірусів та їх передачі

Антивірусне програмне забезпечення в основному виконує профілактичну функцію. Він виявляє будь-який потенційний вірус і працює над його видаленням. Майте на увазі, що все це здебільшого робиться до того, як вірус завдасть шкоди системі. Отже, це означає, що більшість вірусів протистоять задовго до того, як вони завдадуть шкоди вашим системам. Антивірус може боротися з багатьма вірусами за один день без вашого відома. Avast і Norton є одними з найпопулярніших антивірусних програм, доступних на ринку сьогодні.

Якщо вірус атакував вашу систему, ви потенційно можете передати це друзям, родині та мережам. Отже, якщо ви хочете захистити свою комп’ютерну систему, а також комп’ютери своїх знайомих, подумайте про те, щоб отримати антивірус.

Блокуйте спам і рекламу

Якщо ви проведете коротке опитування щодо того, як віруси потрапляють у комп’ютерні системи своїх жертв, ви будете вражені кількістю вірусів, які

використовують спливаючі рекламні вікна та веб-сайти, щоб проникнути на ваші комп'ютери. Спливаюча реклама та веб-сайти зі спамом є одними з найбільш використовуваних вірусами шлюзів для зараження вашого комп'ютера та пошкодження файлів.

Програмне забезпечення, таке як Bullguard Internet Security, працює проти цих шкідливих вірусів і веб-сайтів, блокуючи їхній прямий доступ до вашої комп'ютерної мережі.

Захист від хакерів і крадіжок даних

Хакери зазвичай використовують зловмисне програмне забезпечення або вірусну програму для доступу до комп'ютера жертви. Вони встановлюють шкідливі програми на комп'ютер без відома жертви. Хакери роблять це, надсилаючи жертвам шкідливі електронні листи. Тоді хакер може легко зламати потрібні файли та програми.

Після цього вони можуть використовувати дані жертви за власним бажанням; вони можуть видалити або пошкодити його та вкрасти, щоб пізніше вимагати викуп. Антишкідливе програмне забезпечення, таке як Malwarebytes , або встановлює блокування від злому, або виконує регулярне сканування, щоб виявити в комп'ютерній мережі будь-якого хакера або хакерських програм. Отже, антивірусне програмне забезпечення забезпечує повний захист від хакерів.

Забезпечує захист від знімних пристроїв

Подумайте про те, скільки разів ви передавали дані на комп'ютер і з нього за допомогою знімних пристроїв, таких як USB. Незліченні, правда?

Можливо, ви постраждали від уповільнення комп'ютера або відключення комп'ютера після підключення USB-пристрою друга. Ви коли-небудь замислювалися, чому так сталося? Це тому, що USB або знімний пристрій служив пристроєм передачі вірусу. Отже, чи варто припинити використання знімних пристроїв, оскільки ніколи не знаєш, який USB може містити вірус?

Немає! Просто придбайте антивірусне програмне забезпечення, яке скануватиме всі знімні пристрої на наявність потенційних вірусів, щоб переконатися, що вірус не передається.

Захищає ваші дані та файли

Антивірусне програмне забезпечення стежить за всіма файлами, які потрапляють у вашу систему. Усі ці файли перевіряються на наявність будь-яких особливостей або зловмисності. Віруси можуть легко передаватися у вашу мережу через заражені файли, а вони, у свою чергу, можуть потенційно пошкодити ваші дані та файли. Ви навіть можете повністю втратити цінні дані через такі віруси.

Рішення від програмного забезпечення Avira забезпечує належний захист ваших даних і файлів.

Зарядіть свій ПК

Подумайте про два комп'ютери, які стоять поруч.

Один страждає від наслідків вірусної атаки, таких як низька швидкість обробки та часті збої. Інший захищений антивірусом. Хто з обох матиме кращу швидкість?

Той, що з антивірусом точно. Це тому, що комп'ютер не має проблем, оскільки антивірус зупинив вірус до того, як він міг завдати реальної шкоди. Деякі антивіруси можуть навіть видаляти та видаляти непотрібні файли з невідомих джерел, щоб звільнити місце на диску, збільшуючи швидкість ПК.

Захист брандмауером від шпигунського програмного забезпечення та фішингових атак

Брандмауер, як правило, відстежує вхідний і вихідний трафік з вашої комп'ютерної мережі. У поєднанні з антивірусом захист брандмауера подвійно перевіряє кожен файл або частину даних, які ви надсилаєте чи передаєте з комп'ютера через Інтернет до іншої мережі.

Те саме стосується файлів і даних, які ви отримуєте із зовнішньої мережі. Ви можете ненавмисно відкрити відверто шкідливий веб-сайт або

електронний лист, а потім стати жертвою фішингової атаки. Фішингова атака відбувається, коли зловмисники спеціально намагаються отримати ваші облікові дані для входу, дані кредитної картки або будь-яку іншу особисту інформацію/дані. Така атака може призвести до значних фінансових втрат або особистих витоків. Двосторонній захист брандмауера від антивірусного програмного забезпечення, такого як Avast , блокує та видаляє будь-які електронні листи чи файли, які можуть завдати вам шкоди таким чином.

Обмежте доступ до веб-сайтів, щоб посилити веб-захист

Доступ до несанкціонованих веб-сайтів може наражати вашу комп'ютерну систему на декілька кіберзагроз, зокрема шпигунське програмне забезпечення, хакери, програми-вимагачі тощо. Ці загрози потенційно можуть загрожувати вашим даним і файлам. Антивірусне програмне забезпечення обмежує ваш доступ до Інтернету, щоб обмежити вашу діяльність у неавторизованих мережах. Це робиться для того, щоб переконатися, що ви отримуєте доступ лише до веб-сайтів, які є безпечними та нешкідливими для вашої комп'ютерної системи.

Стежить за дітьми

Найбільший головний біль для батьків у наш передовий час полягає в тому, що їхні діти можуть відкрити доступ до будь-чого за допомогою Інтернету, будь то добре чи погане.

Батьки не завжди можуть стежити за тим, що діти роблять за комп'ютером. І вони не можуть весь час вчити своїх дітей хорошій і поганій мережі, тому що діти легко дратуються. Антивірусне програмне забезпечення може бути рішенням для таких тривожних батьків. Він може надати інструмент моніторингу, за допомогою якого ви можете стежити за діяльністю своїх дітей безпечним, але ефективним способом. Антивірусне програмне забезпечення надає точні журнали діяльності вашої дитини. ESET – один із найвідоміших антивірусів, який пропонує батьківський контроль.

Захищає ваш пароль

Ви захищаєте свої цінні дані та облікові записи паролем, а потім думаєте, що ваші дані та облікові записи захищені.

Але що, якщо хтось вкраде ваші паролі за допомогою вірусу?

Пізніше зловмисник може шантажувати вас, щоб отримати викуп, або використовувати ваш пароль для доступу до конфіденційної інформації. Окрім використання антивірусу, ви також можете подумати про використання менеджера паролів для кращої безпеки.

Економічно ефективним

Більшість антивірусного програмного забезпечення досить економічно вигідне. Щомісячні або річні пакети, які пропонують компанії-виробники антивірусів, недорогі. Якщо ви врахуєте різноманітність послуг, які надаються з преміум-пакетом антивірусу, ви напевно подумаєте, що вартість, яку вони пропонують, є значно меншою.

Крім того, якщо у вас обмежений бюджет і ви не хочете витрачати гроші на купівлю преміум-версії антивірусів, ви можете отримати безкоштовний антивірус.

Чи краще вашому комп'ютеру без антивірусного програмного забезпечення?

Вірусна атака може завдати такої ж шкоди, як і змусити вас купити новий комп'ютер через те, що ваш старий комп'ютер пошкоджений і не підлягає ремонту. Відсутність захисного механізму для вашої комп'ютерної системи – це все одно, що запросити віруси на свій комп'ютер, забезпечивши їм чіткий і доступний вхід.

Ви коли-небудь захочете пошкодити свій комп'ютер своїми руками за власним бажанням? Якщо ні, то придбайте антивірусне програмне забезпечення якнайшвидше, щоб ви могли користуватися комп'ютером без постійного страху стати жертвою вірусної атаки.

3.2 Розробка структурної схеми

Структурна схема розробленої, у результаті виконання бакалаврського проекту, системи зображена на рисунку 3.1.

З нього ми бачимо, що система складається з наступних структурних блоків:

– Джерела погроз.

– Антивірусне програмне забезпечення на персональному комп’ютері.

– Загрози безпеці.

Розглянемо ці структурні блоки більш детально.

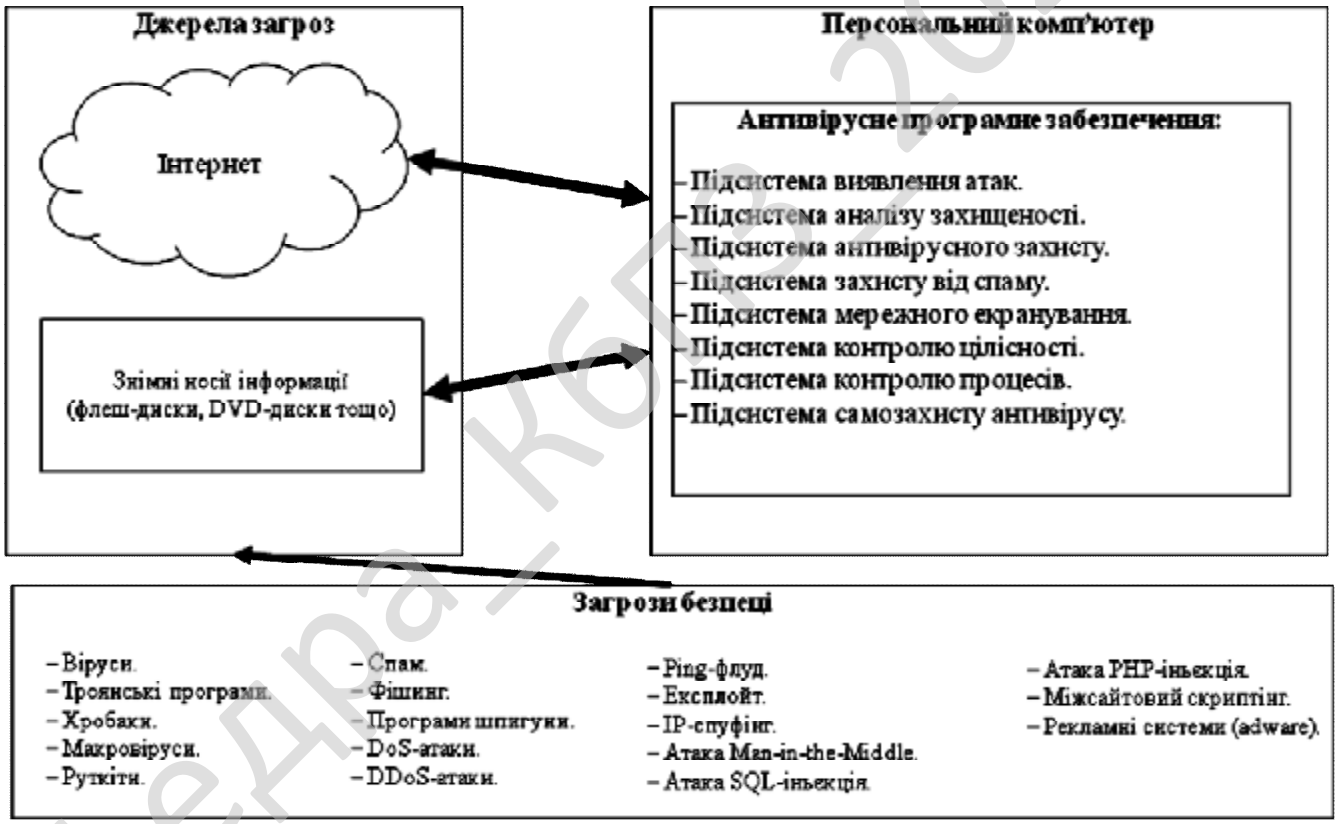

Рисунок 3.1 – Структурна схема системи

До джерел загроз відносяться наступні:

– Загрози з інтернету.

– Загрози при роботі зі змінними носіями інформації.

До антивірусного програмного забезпечення на персональному комп’ютері, відносяться наступні підсистеми:

– Підсистема виявлення атак.

– Підсистема аналізу захищеності.

– Підсистема антивірусного захисту.

– Підсистема захисту від спаму.

– Підсистема мережного екранування.

– Підсистема контролю цілісності.

– Підсистема контролю процесів.

– Підсистема самозахисту антивірусу.

Загальні ознаки комп'ютерних вірусів

Швидше за все, ви чули, як важливо не допустити вірусів, але що таке комп’ютерний вірус? Комп’ютерний вірус, швидше за все, матиме негативний вплив на пристрій, на якому він знаходиться, і його можна виявити за загальними ознаками втрати продуктивності, зокрема:

Швидкість системи

Комп’ютерна система працює повільніше, ніж зазвичай, є однією з найпоширеніших ознак того, що на пристрої є вірус. Це включає в себе повільну роботу самої системи, а також страждання програм і швидкості Інтернету. Якщо на комп’ютері не встановлено потужні додатки чи програми, і він працює повільно, це може бути ознакою зараження вірусом.

Спливаючі вікна

Небажані спливаючі вікна, що з’являються на комп’ютері або у веб-браузері, є ознакою комп’ютерного вірусу. Небажані спливаючі вікна є ознакою шкідливого програмного забезпечення, вірусів або шпигунського програмного забезпечення , що впливає на пристрій.

Програми, що виконуються самостійно

Якщо комп’ютерні програми несподівано закриваються самі по собі, велика ймовірність того, що програмне забезпечення заражене якимось вірусом

або шкідливим програмним забезпеченням. Іншим показником вірусу є те, що програми не завантажуються, коли їх вибрано в меню «Пуск» або на піктограмі робочого столу. Щоразу, коли це трапляється, вашим наступним кроком має бути сканування на віруси та видалення будь-яких файлів у програмах, використання яких може бути небезпечним.

Облікові записи виходять із системи

Деякі віруси створені для впливу на певні програми, що або призведе до їх збою, або змусить користувача автоматично вийти зі служби.

Збій пристрою

Системні збої та несподіване завершення роботи комп'ютера є типовими ознаками вірусу. Комп'ютерні віруси змушують комп'ютери діяти різноманітними дивними способами, які можуть включати самостійне відкриття файлів, відображення незвичайних повідомлень про помилки або натискання клавіш навмання.

Масова розсилка електронних листів з вашого облікового запису електронної пошти

Комп'ютерні віруси зазвичай поширюються електронною поштою. Хакери можуть використовувати облікові записи електронної пошти інших людей для поширення зловмисного програмного забезпечення та здійснення ширших кібератак. Таким чином, якщо обліковий запис електронної пошти надсилає електронні листи в папку «Вихідні», які користувач не надсилав, це може бути ознакою комп'ютерного вірусу.

Зміни на вашій домашній сторінці

Будь-які неочікувані зміни на комп'ютері, як-от зміна домашньої сторінки вашої системи чи оновлення будь-яких налаштувань браузера, є ознаками того, що на пристрої може бути комп'ютерний вірус.

Як комп'ютерні віруси атакують і поширюються?

На початку розвитку комп'ютерів віруси поширювалися між пристроями за допомогою дискет. У наш час віруси все ще можуть поширюватися через

жорсткі диски та пристрої універсальної послідовної шини (USB), але вони, швидше за все, передаються між пристроями через Інтернет.

Комп’ютерні віруси можуть поширюватися через електронну пошту, а деякі навіть здатні захоплювати програмне забезпечення електронної пошти для поширення. Інші можуть приєднуватися до законного програмного забезпечення, в пакетах програмного забезпечення або заражати код, а інші віруси можна завантажувати з скомпрометованих магазинів програм і сховищ зараженого коду. Ключовою особливістю будь-якого комп’ютерного вірусу є те, що він вимагає від жертви виконання його коду або корисного навантаження, що означає, що хост-програма має бути запущена.

Види комп'ютерних вірусів

Існує кілька типів комп’ютерних вірусів, які можуть заразити пристрої. У цьому розділі розповідається про захист комп’ютера від вірусів і про те, як позбутися комп’ютерних вірусів.

Резидентний вірус

Віруси поширюються, заражаючи програми на головному комп’ютері. Резидентний вірус досягає цього, заражаючи програми, які відкриваються користувачем. Нерезидентний вірус здатний заражати виконувані файли, коли програми не запущені.

Багатосторонній вірус

Багатокомпонентний вірус використовує кілька методів для зараження та поширення між комп’ютерами. Зазвичай він залишається в пам’яті комп’ютера, щоб інфікувати жорсткий диск, а потім поширюватися й інфікувати інші диски, змінюючи вміст програм. Це призводить до затримки продуктивності та вичерпання пам’яті програми.

Багатосторонніх вірусів можна уникнути, не відкриваючи вкладення з ненадійних джерел і встановлюючи надійне антивірусне програмне забезпечення. Цьому також можна запобігти, очистивши завантажувальний сектор і весь диск комп’ютера.

Пряма дія

Вірус прямої дії отримує доступ до основної пам’яті комп’ютера та заражає всі програми, файли та папки, розташовані на шляху autoexec.bat, перш ніж видаляється. Цей вірус зазвичай змінює продуктивність системи, але здатний знищити всі дані на жорсткому диску комп’ютера та будь-якому USB-пристрої, підключеному до нього. Вірусів прямої дії можна уникнути за допомогою антивірусних сканерів. Їх легко виявити, як і відновити заражені файли.

Викрадач браузера

Викрадач браузера вручну змінює налаштування веб-браузерів, наприклад замінює домашню сторінку, редагує сторінку нової вкладки та змінює пошукову систему за умовчанням. Технічно це не вірус, оскільки він не може заразити файли, але може завдати величезної шкоди користувачам комп’ютерів, які часто не можуть відновити свою домашню сторінку чи пошукову систему. Він також може містити рекламне ПЗ, яке викликає небажані спливаючі вікна та рекламу.

Зловмисники веб-переглядача зазвичай приєднуються до безкоштовного програмного забезпечення та шкідливих програм із неперевірених веб-сайтів або магазинів додатків, тому використовуйте лише перевірене програмне забезпечення та надійне антивірусне програмне забезпечення.

Перезаписати вірус

Віруси перезапису надзвичайно небезпечні. Вони можуть видаляти дані та замінювати їх власним вмістом файлу чи кодом. Після зараження файли їх неможливо замінити, і вірус може вразити системи Windows, DOS, Linux і Apple. Єдиний спосіб видалити цей вірус – видалити всі заражені ним файли, що може бути руйнівним. Найкращий спосіб захисту від вірусу перезапису – використовувати надійне антивірусне рішення та постійно його оновлювати.

Вірус веб-сценаріїв

Вірус веб-сценаріїв атакує безпеку веб-браузера, дозволяючи хакеру впроваджувати на веб-сторінки шкідливий код або сценарії на стороні клієнта. Це дозволяє кіберзлочинцям атакувати основні веб-сайти, такі як сайти соціальних

мереж, постачальники послуг електронної пошти та будь-які сайти, які дозволяють користувачам вводити дані або переглядати дані. Зловмисники можуть використовувати вірус для надсилання спаму, здійснення шахрайських дій і пошкодження файлів сервера.

Захист від веб-сценаріїв залежить від розгортання програмного забезпечення для захисту веб-переглядача в режимі реального часу, використання безпеки файлів cookie, вимкнення сценаріїв і використання інструментів видалення шкідливого програмного забезпечення.

Інфікатор файлів

Інфектор файлів є одним з найпоширеніших комп'ютерних вірусів. Він перезаписує файли під час їх відкриття та може швидко поширюватися системами та мережами. Це значною мірою впливає на файли з розширеннями .exe або .com. Найкращий спосіб уникнути вірусів-інфекцій файлів – завантажувати лише офіційне програмне забезпечення та розгортати антивірусне рішення.

Мережевий вірус

Мережеві віруси надзвичайно небезпечні, оскільки можуть повністю паралізувати цілі комп'ютерні мережі. Часто їх важко виявити, оскільки вірус може бути прихований на будь-якому комп’ютері в зараженій мережі. Ці віруси можуть легко розмножуватися та поширюватися через Інтернет для передачі на пристрої, підключені до мережі. Надійні, надійні антивірусні рішення та розширені брандмауери мають вирішальне значення для захисту від мережевих вірусів.

Вірус завантажувального сектора

Вірус завантажувального сектора націлений на головний завантажувальний запис (MBR) комп’ютера. Вірус впроваджує свій код у таблицю розділів жорсткого диска, а потім переміщується в основну пам’ять, коли комп’ютер перезавантажується. На наявність вірусу вказують проблеми із завантаженням, низька продуктивність системи та відсутність можливості

визначення місцезнаходження жорсткого диска. Більшість сучасних комп'ютерів мають засоби захисту завантажувального сектора, які обмежують потенціал цього типу вірусу.

Кроки для захисту від вірусу завантажувального сектора включають забезпечення захисту дисків від запису та відсутність запуску комп'ютера з під'єднаними ненадійними зовнішніми дисками.

Дізнайтеся більше про комп'ютерні віруси на прикладах

Існують загальні приклади того, що користувачі комп'ютерів та Інтернету вважають вірусами, але це технічно неправильно.

Чи є троян вірусом?

Троянський кінь – це тип програми, яка видає себе за те, чим вона є, щоб проникнути на пристрій і заразити його шкідливим програмним забезпеченням. Таким чином, троянський кінь – це вірус, замаскований під те, чим він не є. Наприклад, віруси можуть бути приховані в неофіційних іграх, програмах, на сайтах обміну файлами та контрабандних фільмах.

Чи є хробак вірусом?

Комп'ютерний хробак – це не вірус. Хробаки не потребують хост-системи і можуть поширюватися між системами та мережами без дій користувача, тоді як вірус вимагає від користувачів виконання його коду.

Чи є програма-вимагач вірусом?

Програми-вимагачі – це випадки, коли зловмисники блокують жертву в їхній системі чи файлах і вимагають викуп за розблокування доступу. Віруси можна використовувати для здійснення атак програм-вимагачів.

Чи є руткіт вірусом?

Руткіт – це не вірус. Руткіти – це програмні пакети, які надають зловмисникам доступ до систем. Вони не можуть самовідтворюватися або поширюватися між системами.

Чи є програмна помилка вірусом?

«Помилка» – це загальне слово, яке використовується для опису проблем з комп'ютерами, але помилка програмного забезпечення – це не вірус. Помилка – це недолік або помилка в коді програмного забезпечення, якою хакери можуть скористатися для кібератаки або поширення зловмисного програмного забезпечення .

Як захистити комп'ютер від вірусів

Існує кілька способів захисту комп'ютера від вірусів, зокрема:

Використовуйте надійний антивірусний продукт

Надійні комп'ютерні антивірусні продукти мають вирішальне значення для припинення атак зловмисного програмного забезпечення та запобігання зараженню комп'ютерів вірусами. Ці антивірусні концепції захистять пристрої від зараження за допомогою регулярних сканувань, виявлення та блокування шкідливих програм.

Уникайте натискання спливаючих рекламних вікон

Небажана спливаюча реклама, швидше за все, пов'язана з комп'ютерними вірусами та шкідливим програмним забезпеченням. Ніколи не натискайте на спливаючу рекламу, оскільки це може призвести до випадкового завантаження вірусів на комп'ютер.

Скануйте свої вкладення електронної пошти

Популярний спосіб захистити свій пристрій від комп'ютерних вірусів – уникати підозрілих вкладень електронної пошти, які зазвичай використовуються для поширення зловмисного програмного забезпечення. Комп'ютерні антивірусні рішення можна використовувати для сканування вкладень електронної пошти на потенційні віруси.

Скануйте файли, які ви завантажуєте за допомогою файлообмінних програм

Програми для обміну файлами, особливо неофіційні сайти, також є популярними ресурсами для зловмисників для поширення комп'ютерних

вірусів. Уникайте завантаження програм, ігор або програмного забезпечення з неофіційних сайтів і завжди скануйте файли, завантажені з будь-якої файлообмінної програми.

3.3 Розробка функціональної схеми

Антивірус, що розроблений у результаті виконання бакалаврського проектування використовує новітні технології захисту, завдяки яким забезпечується безпека й стабільна робота комп'ютера.

Рисунок 3.2 – Функціональна схема системи

Функціональна схема системи складається з наступних основних функціональних блоків:

– Захист у режимі реального часу.

– Сканування системи.

– Контроль запущених процесів.

– Ядро антивірусної програми.

– Бази даних вірусів.

Розглянемо більш детальніше склад функціональних блоків.

Функціональний блок захисту у режимі реального часу складається з наступних функціональних підсистем:

– Перевірка запущених файлів.

– Перевірка відкритих web-сторінок.

– Блокування посилань на заражені й фішингові веб-сайти.

– Перевірка отриманих поштових повідомлень.

– Захист від спаму й фішингу в поштових програмах.

– Самозахист антивірусу від спроб вимикання з боку шкідливого ПЗ.

Функціональний блок сканування системи складається з наступних функціональних підсистем:

– Фільтр сканування.

– Швидке сканування дисків.

– Повне сканування дисків.

– Вибіркове сканування файлів.

– Лікування, знищення та переміщення у карантин інфікованих файлів.

– Звіти про результатів сканування.

Функціональний блок контролю запущених процесів складається з наступних функціональних підсистем:

– Контроль роботи запущених програм та процесів і обмеження їхнього доступу до важливих областей ОС.

– Список активних процесів.

– Блокування шкідливих та інфікованих процесів.

– Перевірка процесів зі списку автозавантаження.

– Видалення зі списку автозавантаження шкідливих процесів.

Функціональний блок ядра антивірусної програми складається з наступних функціональних підсистем:

– Модуль евристичного аналізу.

– Модуль криптоаналізу.

– Модуль сигнатурного пошуку.

– Модуль пошуку вірусів за контрольними сумами.

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання бакалаврського проектування, наведена на рисунку 3.3.

Першим процесом у розробленій системі являється процес виведення головного вікна програми.

Він взаємодіє з наступними процесами:

– виведення системних ресурсів;

– виведення вікна параметрів антивірусу;

– контроль процесів;

– вікно сканування.

Вікно контролю процесів взаємодіє з контролем процесів, який пов’язаний з наступними процесами:

– виведення статистики;

– зупинки/запуску контролю процесів.

Процес вікна сканування взаємодіє з наступними процесами:

– вибір ресурсів для сканування.

– запуск/призупинення сканування;

– виведення звіту, якій взаємодіє з процесом збереження звіту.

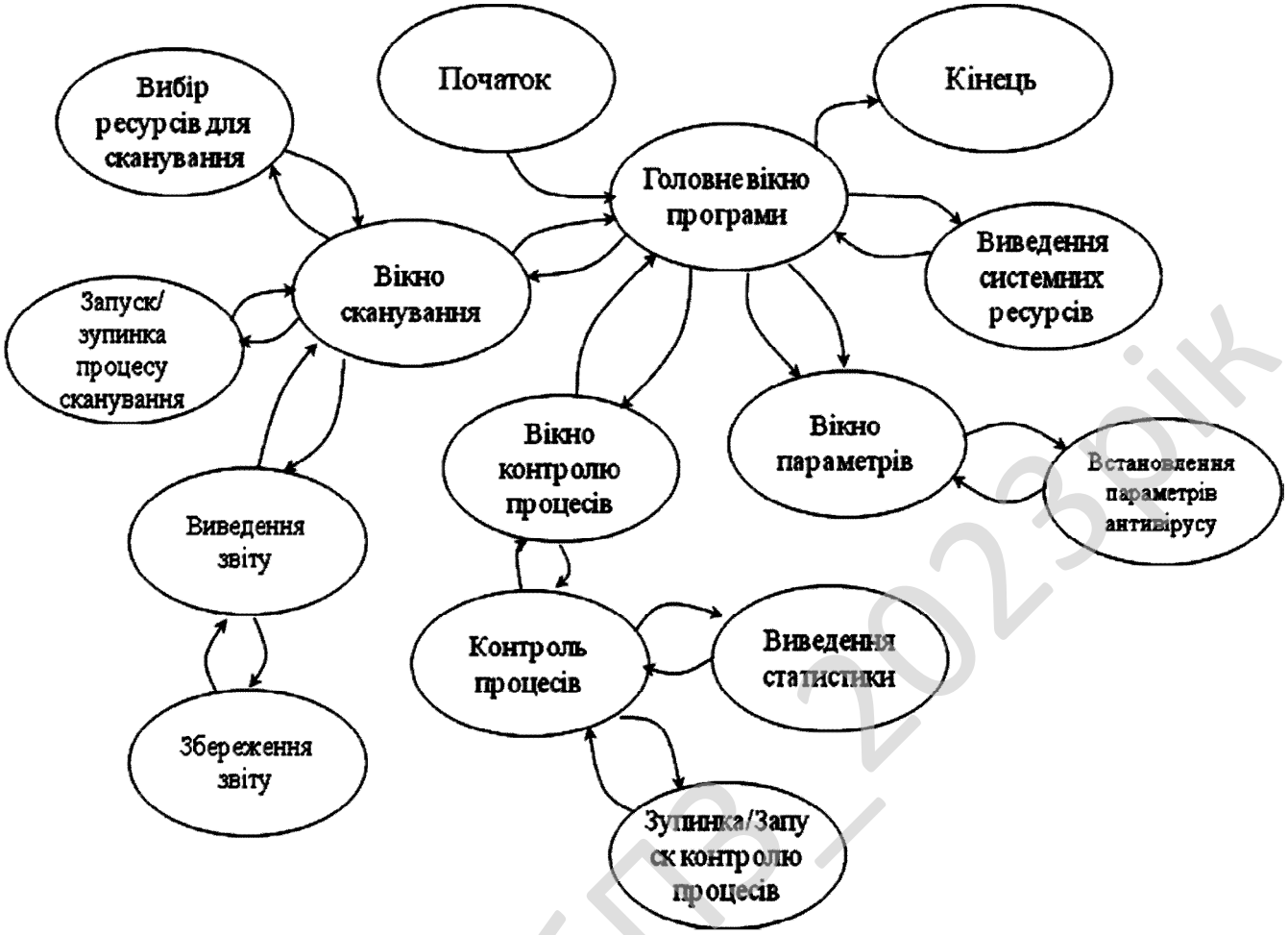

Рисунок 3.3 – Діаграма взаємодії процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

На рисунку 4.1 наведено блок-схему основної програми. Її робота складається з виконання наступних кроків.

Спершу відбувається виведення основного вікна програми.

Після цього відбувається виведення логічних дисків, які є у системі.

Далі користувач обирає сканувати йому систему, або ні.

Якщо він вирішує не сканувати, то відбувається перехід до контролю системи.

У іншому випадку, відбувається вибір дисків для сканування.

Після виконання цієї дії, проводиться пошук вірусів на вказаних дисках.

Якщо вірус знаходиться то він знищується, у іншому випадку, відбувається перехід до формування та виведення на екран звіту, про наявність, або ні вірусів у системі, й знищення, або ні цих вірусів.

Крім пошуку вірусів, розроблений у ході виконання бакалаврського проектування програмний продукт дозволяє проводити контроль процесів, які відбуваються у мережі.

Для цього спершу користувач визначається, чи буде він проводити контроль процесів.

Якщо він вирішує його проводити, то відбувається вивід вікна контролю процесів.

У іншому випадку користувач переходить на вікно закінчення роботи з програмним продуктом.

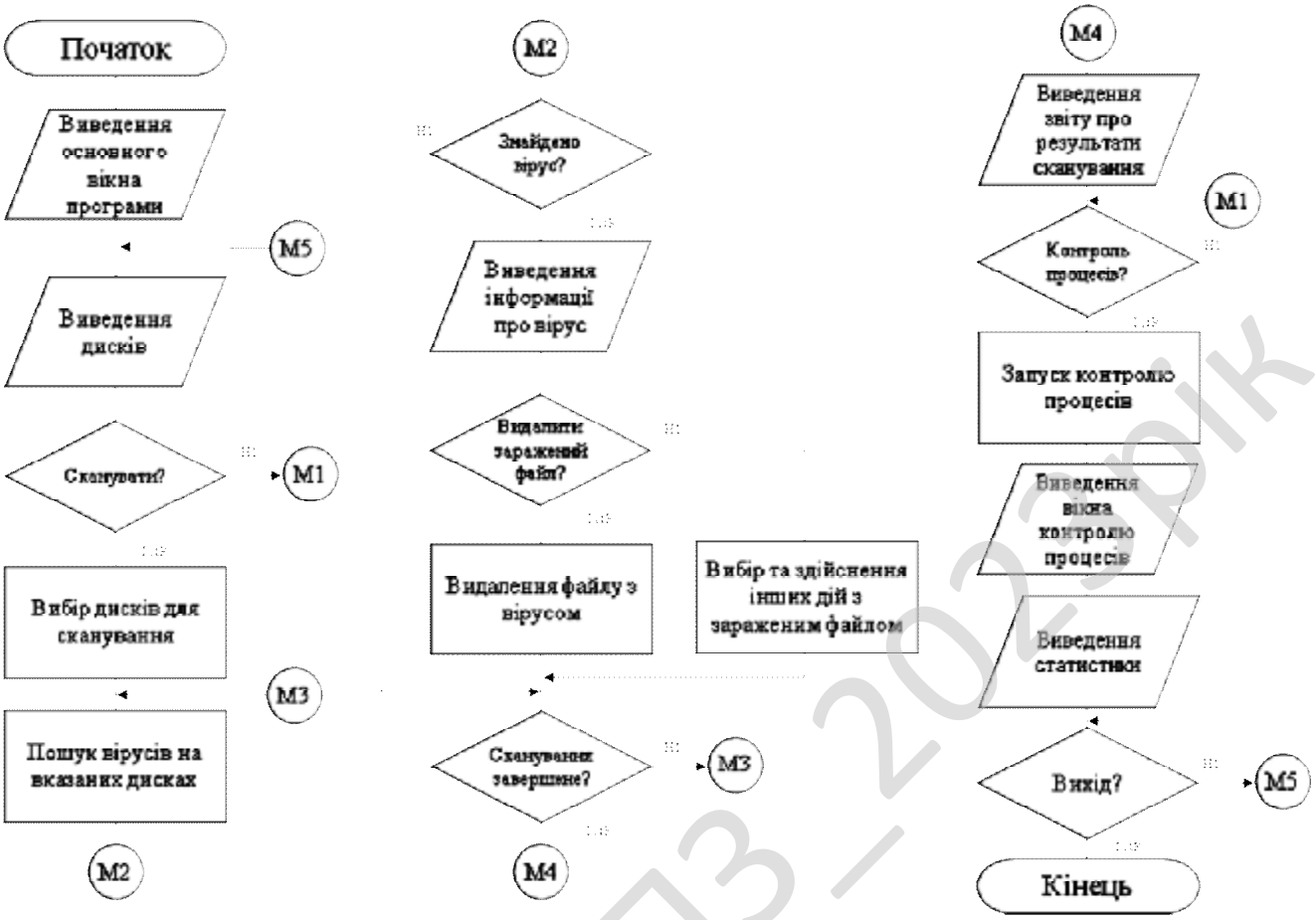

Рисунок 4.1 – Блок-схема роботи основної програми

Після виведення вікна контролю процесів, відбувається виведення статистики дій процесів, які функціонують у системі.

Після цього користувач обирає працювати йому далі з антивірусом, або завершити роботу з програмою.

На рисунку 4.2 зображена блок-схема роботи підпрограми сканування.

Вона працює наступним чином.

Спершу відбувається вибір об’єкту для перевірки.

Якщо це архів, то відбувається розпаковування архіву у тимчасову директорію.

Після цього відбувається пошук вірусів за наступними методами пошуки шкідливого вірусного коду:

– за контрольними сумами;

– сигнатурний пошук;

– криптоаналіз:

– евристичний аналіз.

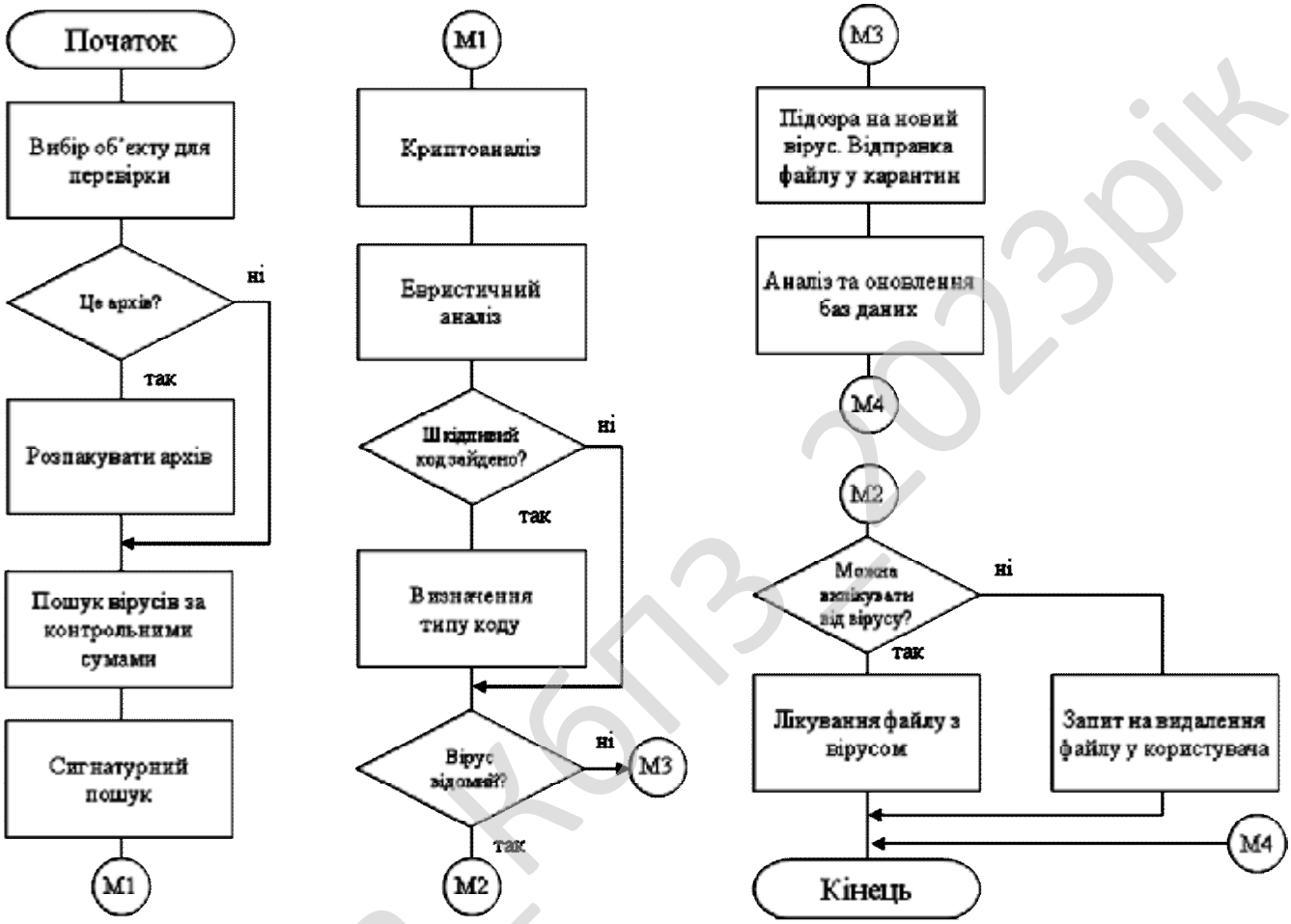

Рисунок 4.2 – Блок-схема роботи підпрограми сканування

Якщо шкідливий код знайдено, то виходячи з заданих правил, визначається це старий відомий вірус, або це новий вірус.

Якщо є підозра на новий вірус то проводиться його аналіз, та він додається до бази даних вірусів.

У іншому випадку, якщо вірус вже відомий, то користувач визначає яку дію з ним проводити: лікування або знищення.

Якщо він обирає лікування, то відбувається спроба вилікувати файл,

видаливши з нього шкідливий код. Якщо це неможливо, то файл знищується.

Розглянемо більш детально роботу антивірусного движка, на якому працює розроблене програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи.

Антивірусний "движок" (Anti-Virus Engine) – це програмний модуль, що призначений для детектування шкідливого програмного забезпечення. "Движок" є основним компонентом будь-якої антивірусної програми, незалежно від її призначення. Движок використовується як у персональних продуктах – персональний сканер або монітор, так і в серверних рішеннях – сканер для поштового або файлового сервера, мережного екрану або проксі-серверу. Як правило, для детектування шкідливих програм, у більшості "движків" реалізовані наступні технології:

– Пошук за "сигнатурами" (унікальній послідовності байт).

– Пошук за контрольними сумами або CRC (контрольної суми з унікальної послідовності байт).

– Використання скороченої маски.

– Криптоаналіз.

– Статистичний аналіз.

– Евристичний аналіз.

– Емуляція.

Розглянемо кожний із цих методів докладніше.

Пошук за "сигнатурами"

Сигнатура – це унікальний "рядок" байт, що однозначно характеризує ту або іншу шкідливу програму. Сигнатурний пошук, у тій або іншій модифікації, використовується для виявлення вірусів та інших шкідливих програм, починаючи з найперших антивірусних програм і дотепер. Незаперечне достоїнство сигнатурного пошуку – швидкість роботи (при використанні спеціально розроблених алгоритмів) і можливості детектування декількох вірусів однією сигнатурою. Недолік – розмір сигнатури для впевненого детектування повинен

бути досить великий, як мінімум 8-12 байт (звичайно для точного детектування використовуються набагато більш довгі сигнатури, до 64 байт), отже, розмір антивірусної бази буде досить великим. Крім цього, останнім часом більшу поширеність одержали шкідливі програми, написані на мовах високого рівня (C++, Delphi, Visual Basic), а в таких програм є окремі частини коду, які практично не змінюються (так звана Run Time Library). Неправильно обрана сигнатура неминуче приведе до помилкового спрацьовування – детектування "чистого", не зараженого файлу як зараженого вірусом. Як рішення цієї проблеми пропонується використовувати або дуже великі сигнатури або використовувати детектування по деяких областях даних, наприклад, таблиці переміщень (relocation table) або текстові рядки, що не завжди добре.

Пошук за контрольними сумами (CRC)

Пошук за контрольними сумами (CRC – cyclic redundancy check), по суті, є модифікацією пошуку за сигнатурами. Метод був розроблений для запобігання основних недоліків сигнатурного пошуку – розміру бази й зменшення ймовірності помилкових спрацьовувань. Суть методу полягає в тому, що для пошуку шкідливого коду береться не тільки "опорний" рядок – сигнатура, а, точніше сказати, контрольна сума цього рядка, але й місце розташування сигнатури в тілі шкідливої програми. Місце розташування використовується для того, щоб не підраховувати контрольні суми для всього файлу. Таким чином, замість 10-12 байт сигнатури (мінімально) використовується 4 байти для зберігання контрольної суми й ще 4 байти – для місця розташування. Однак метод пошуку за контрольними сумами трохи повільніший, ніж пошук за сигнатурами.

Використання масок для виявлення шкідливого коду досить часто буває ускладнений наявністю шифрованого коду (так звані поліморфні віруси), оскільки при цьому або неможливо вибрати маску, або маска максимального розміру не задовольняє умові однозначної ідентифікації вірусу без помилкових спрацьовувань.

Неможливість вибору маски достатнього розміру у випадку поліморфного

вірусу легко пояснюється. Шляхом шифрування свого тіла вірус домагається того, що більша частина його коду в ураженому об'єкті є змінною, і, відповідно, не може бути обрана як маска.

Для детектування таких вірусів застосовуються наступні методи: використання скороченої маски, криптоаналіз і статистичний аналіз. Розглянемо ці методи докладніше.

Використання скороченої маски

При поразці об'єктів вірус, що використовує шифрування, перетворить свій код у шифровану послідовність даних:

$$S = F\,(T),$$

де T – базовий код вірусу;

S – зашифровані коди вірусу;

F – функція шифрування вірусу, що довільно вибирається з деякої множини перетворень $\{F\}$.

Спосіб скороченої маски полягає в тому, що вибирається перетворення R зашифрованих кодів вірусу S, таке, що результат перетворення (тобто деяка послідовність даних S') не буде залежати від ключів перетворення F, тобто:

$$S = F\,(T),$$

$$S' = R\,(S) = R\,(F\,(T)) = R'\,(T).$$

При застосуванні перетворення R до всіляких варіантів шифрованого коду S результат S' буде постійним при постійному T. Таким чином, ідентифікація уражених об'єктів виконується шляхом вибору S' як скорочена маска й застосування до уражених об'єктів перетворення R.

Криптоаналіз

Цей спосіб полягає в наступному: за відомим базовим кодом вірусу й за відомим зашифрованим кодом (або за "підозріломим" кодом, схожим на зашифроване тіло вірусу) відновлюються ключі й алгоритм програми-розшифровувача. Потім цей алгоритм застосовується до зашифрованої ділянки, результатом чого є розшифроване тіло вірусу.

Як правило, цей спосіб працює значно швидше й займає набагато менше пам'яті, ніж емуляція інструкцій вірусу. Однак рішення подібних систем часто є завданням високої складності.

Причому основна проблема – це математичний аналіз отриманого рівняння або отриманої системи рівнянь. Багато в чому завдання рішення систем рівнянь при відновленні зашифрованого тіла вірусу нагадує класичне криптографічне завдання відновлення зашифрованого тексту при невідомих ключах. Однак тут це завдання звучить трохи інакше: необхідно з'ясувати, чи є даний зашифрований код результатом застосування деякої відомої з точністю до ключів функції. Причому заздалегідь відомі багато даних для рішення цього завдання: ділянка зашифрованого коду, ділянка незашифрованого коду, можливі варіанти функції перетворення. Більш того, сам алгоритм цього перетворення й ключі також присутні в аналізованих кодах. Однак існує значне обмеження, що полягає в тому, що дане завдання повинне вирішуватися в конкретних границях оперативної пам'яті й процедура рішення не повинна займати багато часу.

Статистичний аналіз

Також використовується для детектування поліморфних вірусів. Під час своєї роботи сканер аналізує частоту використання команд процесора, будує таблицю команд, що зустрічаються, процесора, і на основі цієї інформації робить висновок про зараження файлу вірусом. Даний метод ефективний для пошуку деяких поліморфних вірусів, тому що ці віруси використовують обмежений набір команд у декрипторі, тоді як "чисті" файли використовують зовсім інші команди з іншою частотою. Наприклад, всі програми для MS-DOS часто використовують переривання 21h, однак у декрипторі поліморфних DOS-вірусів ця команда практично не зустрічається.

Основний недолік цього методу в тому, що є ряд складних поліморфних вірусів, які використовують майже всі команди процесора й від копії до копії набір використовуваних команд сильно змінюється, тобто за побудованою таблицею частот не представляється можливим виявити вірус.

Евристичний аналіз

Коли кількість вірусів перевищила кілька сотень, антивірусні експерти задумалися над ідеєю детектування шкідливих програм, про існування яких антивірусна програма ще не знає (немає відповідних сигнатур). У результаті були створені так звані евристичні аналізатори. Евристичним аналізатором називається набір підпрограм, які аналізують код файлів, що виконуються, макросів, скриптів, пам'яті або завантажувальних секторів для виявлення в ньому різних типів шкідливих комп'ютерних програм. Існують два принципи роботи аналізатора.

Статичний метод. Пошук загальних коротких сигнатур, які присутні в більшості вірусів (так звані "підозрілі" команди). Наприклад, велика кількість вірусів робить пошук вірусів по масці *.EXE, відкриває знайдений файл, робить запис у відкритий файл. Завдання евристик у цьому випадку – знайти сигнатури, що відбивають ці дії. Потім відбувається аналіз знайдених сигнатур, і, якщо знайдено деяку кількість необхідних і достатніх "підозрілих команд", то приймається рішення про те, що файл інфікований. Великий плюс цього методу – простота реалізації й хороша швидкість роботи, але при цьому рівень виявлення нових шкідливих програм досить низький.

Динамічний метод. Цей метод з'явився одночасно із впровадженням в антивірусні програми емуляції команд процесора (докладніше емулятор описаний нижче). Суть методу полягає в емуляції виконання програми й протоколюванні всіх "підозрілих" дій програми. На основі цього протоколу приймається рішення про можливе зараження програми вірусом. На відміну від статичного методу, динамічний метод більш вимогливий до ресурсів комп'ютера, однак і рівень виявлення в динамічному методі значно вище.

Емуляція

Технологія емуляції коду програм (або Sandboxing) стала відповіддю на появу великої кількості поліморфних вірусів. Ідея цього методу полягає в тому, щоб емулювати виконання програми (як зараженої вірусом, так і "чистої") у спеціальному "оточенні", що називається також буфером емуляції або

"пісочницею". Якщо в емулятор попадає заражений поліморфним вірусом файл, то після емуляції в буфері виявляється розшифроване тіло вірусу, готове до детектування стандартними методами (сигнатурний або CRC пошук).

Сучасні емулятори емулюють не тільки команди процесора, але й виклики операційної системи. Задача написання повноцінного емулятора є досить трудомісткою, не говорячи вже про те, що при використанні емулятора доводиться постійно контролювати дії кожної команди. Це необхідно для того, щоб випадково не виконати деструктивні компоненти алгоритму вірусу.

Слід особливо зазначити, що доводиться саме емулювати роботу інструкцій вірусу, а не трасувати їх, оскільки при трасуванні вірусу занадто велика ймовірність виклику деструктивних інструкцій або кодів, відповідальних за поширення вірусу.

База даних антивірусного "движка"

База даних є невід'ємною частиною антивірусного "движка". Більш того, якщо вважати що добре спроектований "движок" змінюється не так часто, то антивірусна база змінюється постійно, тому що саме в антивірусній базі перебувають сигнатури, контрольні суми й спеціальні програмні модулі для детектування шкідливих програм. Як відомо, нові віруси, мережні хробаки й інші шкідливі програми з'являються із завидною частотою, і тому дуже важливо, щоб відновлення антивірусної бази відбувалися якнайчастіше. Якщо п'ять років тому було досить щотижневих відновлень, то сьогодні просто необхідно одержувати хоча б щоденні відновлення антивірусної бази.

Також дуже важливо, що саме перебуває в антивірусній базі: чи тільки записи про віруси або ще й додаткові програмні процедури. У другому випадку набагато легше обновляти функціонал антивірусного "движка" шляхом звичайного відновлення баз.

Підтримка "складних", вкладених об'єктів

За останні кілька років антивірусні "движки" сильно змінилися. Якщо першим антивірусам для того, щоб вважатися першокласною програмою, було

досить перевіряти системну пам'ять, файли, що виконуються, й завантажувальні сектори, то вже через кілька років у зв'язку з ростом популярності спеціальних утиліт упакування виконавчих модулів перед розроблювачами виникло завдання розпакувати упакований файл перед тим, як його сканувати.

Потім нова проблема – віруси навчилися заражати архівні файли (та й самі користувачі найчастіше пересилали заражені файли в архівах). Антивіруси були змушені навчитися обробляти й архівні файли. В 1995 році з'явився перший макровірус, що заражає документи Microsoft Word. Варто помітити, що формат документів, використовуваний Microsoft Word, закритий, і дуже складний. Ряд антивірусних компаній дотепер не вміють повноцінно обробляти такі файли.

Сьогодні, у зв'язку з величезною популярністю електронної пошти, антивірусні "движки" також обробляють і бази поштових повідомлень і самі повідомлення.

Методи детектування

У типовому антивірусному "движку", що реалізований у кожній антивірусній програмі, використовуються всі необхідні технології для виявлення шкідливих програм: ефективний евристичний аналізатор, високопродуктивний емулятор і, що саме головне, грамотна й гнучка архітектура підсистеми детектування шкідливих програм, що дозволяє використовувати всі перераховані вище методи детектування.

Майже в кожному антивірусному "движку" базовим є метод детектування за контрольними сумами. Цей метод був обраний виходячи з вимоги мінімізації розміру антивірусних баз. Однак архітектура "движка" часто настільки гнучка, що дозволяє використовувати кожний з перерахованих вище методів детектирування, що й робиться для деяких особливо складних вірусів. Це дозволяє домогтися високого рівня детектування вірусів. Докладніше архітектура антивірусного "движка" представлена на схемі далі в тексті.

Практичне застосування способів виявлення поліморфних вірусів (криптоаналіз і статистичний аналіз, застосування скороченої маски й емуляція),

зводиться до вибору найбільш оптимального по швидкодії й обсягу необхідної пам'яті методу. Код більшості вірусів, що самошифруються, досить легко відновлюється процедурою емуляції. Якщо використання емулятора не є оптимальним рішенням, то код вірусу відновлюється за допомогою підпрограми, що реалізує зворотне перетворення – криптоаналіз. Для детектування емуляції вірусів, що не піддаються емуляції і вірусів, для яких не представляється можливим побудувати зворотне перетворення, використовується спосіб побудови скорочених масок.

У деяких, найбільш складних, випадках застосовується комбінація наведених вище способів. Частина коду розшифровувача емулюється, при цьому з розшифровувача виділяються команди, реально відповідальні за алгоритм розшифровки. Потім на основі отриманої інформації будується й вирішується система рівнянь для відновлення коду вірусу і його детектування.

Комбінація способів застосовується також при кратному застосуванні шифрування, коли вірус шифрує своє тіло кілька разів, використовуючи при цьому різні алгоритми шифрування. Комбінований спосіб відновлення інформації або "чиста" емуляція коду розшифровувача часто застосовуються й з тієї причини, що кожний новий вірус повинен бути проаналізований і підключений в антивірусну базу за мінімальні строки, у які далеко не завжди вкладається необхідний математичний аналіз. І в результаті доводиться користуватися більш громіздкими методами детектування вірусу, незважаючи на те, що цілком доцільні методи математичного аналізу алгоритму розшифровувача.

Робота з "складними" об'єктами

Антивірусні "движки" підтримують роботу з величезним числом форматів упакування й архівування. Розроблювачі досить рідко публікують повний (або хоча б досить докладний) перелік підтримуваних форматів. Далі представлена офіційно опублікована інформація про підтримку "складних" форматів в Антивірусі Касперского. В інших антивірусних продуктах список підтримуваних форматів повинен бути приблизно таким же. "Движок" Антивірусу Касперского

підтримує роботу з більш ніж 400 різними утилітами впакування файлів, що виконуються, інсталяторів і архиваторов (усього більше 900 модифікацій, за станом на травень 2011). Серед них пакувальники файлів, що виконуються, і системи шифрування. Самі популярні з них: Diet, AVPACK, COMPACK, Epack, ExeLock, ExePack, Expert, HackStop, Jam, LzExe, LzCom, PaquetBuilder, PGMPAK, PkLite, PackWin, Pksmart, Protect, ProtEXE, RelPack, Rerp, Rjcrush, Rucc, Scramb, SCRNCH, Shrink, Six-2-Four, Syspack, Trap, UCEXE, Univac, UPD, UPX (кілька версій), WWPACK, ASPack (кілька версій), ASProtect (кілька версій), Astrum, BitArts, BJFnt, Cexe, Cheaters, Dialect, DXPack, Gleam, CodeSafe, ELFCrypt, JDPack, JDProtect, INFTool, Krypton, Neolite, ExeLock, NFO, NoodleCrypt, OptLink, PCPEC, PEBundle, PECompact (кілька версій), PCShrink, PE-Crypt, PE-Diminisher, PELock, PEncrypt, PE-Pack (кілька версій), PE-Protect, PE-Shield, Petite, Pex, PKLite32, SuperCede, TeLock, VBox, WWPack32, XLok, Yoda.

Підтримка стількох пакувальників і архіваторов дозволяє скоротити час аналізу нових вірусів, що приводить до збільшення швидкості реакції на появу нового вірусу, і домогтися високого рівня виявлення вже відомих вірусів.

Архіватори й інсталятори (усього більше 60). Самі популярні з них: CAB, ARJ, ZIP, GZIP, Tar, AIN, HA, LHA, RAR, ACE, BZIP2, WiseSFX (кілька версій), CreateInstall, Inno Installer, StarDust Installer, MS Expand, GKWare Setup, SetupFactory, SetupSpecialist, NSIS, Astrum, PCInstall, Effect Office.

Підтримка великої кількості різновидів архіваторів особливо важлива для перевірки поштових систем, тому що гнітюча частина вірусів пересилається поштою в архівіруваному виді. Розпакування об'єктів відбувається незалежно від рівня вкладеності архівів. Наприклад, якщо заражений файл упакований утилітою UPX, а потім файл упакований в архів ZIP, що упакований в архів CAB і т.д., то антивірусний "движок" однаково повинен змогти дістати вихідний файл і виявити вірус.

Слід зазначити, що подібні міркування носять аж ніяк не теоретичний характер. Так, широко відома троянська програма Backdoor.Rbot, що

поширювалася упакованою безліччю різних програм (Ezip, Exe32Pack, ExeStealth, PecBundle, PECompact, FSG, UPX, Morphine, ASPack, Petite, PE-Pack, PE-Diminisher, PELock, PESpin, TeLock, Molebox, Yoda, Ezip, Krypton і ін.).

Алгоритм розпакування архівів звичайно має достатній інтелект, щоб не розпаковувати всілякі "архівні бомби" – архіви невеликого розміру, у які впаковані величезні файли (з дуже високим ступенем стиску) або кілька однакових файлів. Звичайно для перевірки такого архіву потрібно багато часу, але сучасні антивірусні "движки" часто розпізнають подібні "бомби".

Механізм відновлення антивірусних баз і їхній розмір

Відновлення антивірусних баз звичайно виходять по кілька разів у день. Деякі в стані випускати відновлення раз у годину, деякі – раз в дві години. У кожному разі, при сучасному високому рівні небезпеки в Інтернет таке часте відновлення антивірусних баз цілком виправдано.

Розмір відновлень указує на продуманість архітектури антивірусного "движка". Так, розмір регулярних відновлень лідируючих у галузі компаній, як правило, не перевищує 30 Кб. При цьому в антивірусні бази звичайно закладене близько 70% функціональності всього антивірусного "движка". У будь-якому відновленні антивірусної бази може бути додана підтримка нового пакувальника або архіватора. Таким чином, щодня обновляючи антивірусні бази, користувач одержує не тільки нові процедури детектування нових шкідливих програм, але й відновлення всього антивірусу. Це дозволяє дуже гнучко реагувати на ситуацію й гарантувати користувачеві максимальний захист.

Евристичний аналізатор

В евристичному аналізаторі, що входить до складу майже кожного антивірусу, використовуються обидва описані вище методи аналізу – криптоаналіз і статистичний аналіз. Сучасний евристичний аналізатор споконвічно розробляється так, щоб бути розширюваним (на відміну від більшості евристичних аналізаторів першого покоління, які розроблялися для виявлення шкідливих програм тільки у виконуючих модулях).

На сьогоднішній день евристичний аналізатор дозволяє виявляти шкідливі коди у файлах, що виконуються, секторах і пам'яті, а також нові скрипт-віруси й шкідливі програми для Microsoft Office (і інших програм, що використовують VBA), і, нарешті, шкідливий код, написаний на мовах високого рівня, таких як Microsoft Visual Basic.

Гнучка архітектура й комбінація різних методів дозволяє домогтися досить високого рівня детектування нових шкідливих програм. При цьому розроблювачі докладають всі зусилля для того, щоб кількість фіктивних тривог звести до мінімуму. Продукти, що представляються лідерами антивірусної індустрії, надзвичайно рідко помиляються в детектуванні шкідливих кодів.

Схема роботи антивірусного "движка"

У наведеній нижче схемі описаний зразковий алгоритм роботи антивірусного "движка". Варто помітити, що емуляція, пошук відомих і невідомих шкідливих програм відбувається одночасно.

Як було сказано вище, під час відновлення антивірусної бази відбувається також відновлення й додавання модулів розпакування упакованих файлів і архівів, евристичного аналізатора й інших модулів антивірусного "движка".

Оригінальні технології в антивірусних "движках"

Майже кожний розроблювач антивірусних продуктів реалізує якісь свої технології, що дозволяють зробити роботу програми ефективнішою й більш продуктивною. Деякі із цих технологій мають пряме відношення до пристрою "движка", тому що саме від його роботи часто залежить продуктивність усього рішення. Далі буде розглянутий ряд технологій, що дозволяють значно прискорити перевірку об'єктів і при цьому гарантувати збереження високої якості детектування, а також поліпшити детектування й лікування шкідливого програмного забезпечення в архівних файлах.

Почати треба з технології iChecker. Ця технологія і її аналоги реалізовані майже в кожному сучасному антивірусі. Слід зазначити, що iChecker – назва, запропонована фахівцями "Лабораторії Касперского". Експерти, наприклад, Panda

Software називають її UltraFast. Дана технологія дозволяє домогтися розумного балансу між надійністю захисту робочих станцій (і особливо серверів), і використанням системних ресурсів комп'ютера, що захищається. Завдяки цій технології значно скорочується час завантаження (до 30-40%) операційної системи (у порівнянні із традиційними антивірусними захистами) і час запуску додатків при активному антивірусному захисті. При цьому гарантується, що всі файли на дисках комп'ютера були перевірені й не інфіковані. Основна ідея даної технології – не треба перевіряти те, що не змінювалося, і вже було перевірено. Антивірусний "движок" веде спеціальну базу даних, у якій зберігаються контрольні суми всіх перевірених (і не інфікованих) файлів. Тепер, перш ніж віддати файл на перевірку, "движок" підраховує й порівнює контрольну суму файлу з даними, що зберігаються в базі даних. Якщо дані збігаються, то це значить, що файл був перевірений і повторна перевірка не потрібно. Варто помітити що час, затрачуваний на підрахунок контрольних сум файлу – значно менше, ніж час антивірусної перевірки.

Особливе місце в роботі антивірусу займає лікування заархівірованих інфікованих об'єктів. iCure – технологія лікування інфікованих файлів в архівах. Завдяки цій технології інфіковані об'єкти усередині архівних файлів будуть успішно вилікувані (або вилучені, залежно від настроювань антивірусу) без використання зовнішніх утиліт архівації. На сьогоднішній день більшість антивірусів підтримують наступні типи архівів: ARJ, CAB, RAR, ZIP. Завдяки модульній архітектурі й технологіям відновлення антивірусного "движка" користувач, як правило, може легко обновляти й розширювати список підтримуваних типів архіваторів без перезавантаження антивірусу.

iArc – ще одна технологія роботи з архівними файлами. Ця технологія необхідна для роботи з багатотомними архівами. iArc дозволяє перевіряти багатотомні архіви й виявляти віруси навіть, якщо вони будуть упаковані в багатотомний архів, що, у свою чергу, також буде впакований у багатотомний архів.

Багатопоточність. Антивірусний "движок" є багатопоточним модулем, і може одночасно обробляти (перевіряти на наявність шкідливих кодів) кілька об'єктів (файли, сектори, скрипти та ін.).

Більшість із перерахованих вище технологій у тому чи іншому виді реалізовано в кожному сучасному антивірусному продукті.

Поліморфні віруси

Протягом всієї статті часто використовувалися терміни "поліморфні" віруси і віруси здатні до самошифрування. Саме цей тип шкідливих кодів вплинув на розвиток антивірусних технологій.

Самошифрування й поліморфічність використовуються практично всіма типами вірусів для того, щоб максимально ускладнити процедуру детектування вірусу. Поліморфні віруси (polymorphic) – це досить важко виявляємі віруси, які не мають сигнатур, тобто, не містять жодної постійної ділянки коду. У більшості випадків два зразки того самого поліморфного вірусу не будуть мати жодного збігу. Це досягається шифруванням основного тіла вірусу й модифікаціями програми-розшифровувача (декриптора). До поліморфних вірусів відносяться ті, детектування яких неможливо (або вкрай важко) здійснити за допомогою так званих вірусних масок – ділянок постійного коду, специфічних для конкретного вірусу. Досягається це двома основними способами – шифруванням основного коду вірусу з непостійним ключем і випадковим набором команд розшифровувача або зміною самого виконуваного коду вірусу. Існують також інші, досить екзотичні приклади поліморфізму: DOS-вірус "Bomber", наприклад, не зашифрований, однак послідовність команд, що передає керування коду вірусу, є повністю поліморфною.

Поліморфізм різного ступеня складності зустрічається у вірусах всіх типів – від завантажувальних і файлових DOS-Вірусів до Windows-вірусів і навіть макро-вірусів.

Поліморфні розшифровувачі

Найпростішим прикладом частково поліморфного розшифровувача є

наступний набір команд, у результаті застосування якого жоден байт коду самого вірусу і його розшифровувача не є постійним при зараженні різних файлів:

```
MOV reg_1, count ; reg_1, reg_2, reg_3 вибираються з
MOV reg_2, key ; AX,BX,CX,DX,SI,DI,BP
MOV reg_3, _offset ; count, key, _offset також можуть мінятися
_loop:
xxx byte ptr [reg_3], reg_2 ; xor, add або sub
DEC reg_1
Jxx _loop ; ja або jnc  ; далі слідують зашифровані код і дані вірусу
```

Складні поліморфні віруси використовують значно більш складні алгоритми для генерації коду своїх розшифровувачів: наведені вище інструкції (або їхні еквіваленти) переставляються місцями від зараження до зараження, розбавляються нічого не змінюючими командами, типу NOP, STI, CLI, STC, CLC і т.д.

Повноцінні ж поліморфні віруси використовують ще більш складні алгоритми, у результаті роботи яких у розшифровувачі вірусу можуть зустрітися операції SUB, ADD, XOR, ROR, ROL і інші в довільній кількості й порядку. Завантаження й зміна ключів і інших параметрів шифровки виробляється також довільним набором операцій, у якому можуть зустрітися практично всі інструкції процесора Intel (ADD, SUB, TEST, XOR, OR, SHR, SHL, ROR, MOV, XCHG, JNZ, PUSH, POP ...) з усіма можливими режимами адресації. З'являються також поліморфні віруси, розшифровувач яких використовує інструкції аж до Intel386, а влітку 1997 року виявлений 32-бітний поліморфний вірус, що заражає EXE-файли Windows 95.

У результаті, на початку файлу, зараженого подібним вірусом, іде набір безглуздих, на перший погляд, інструкцій. Цікаво, що деякі комбінації, які цілком працездатні, не беруться фірмовими дизассемблерами (наприклад, сполучення CS:CS: або CS:NOP). І серед цієї "каші" з команд і даних зрідка прослизають MOV, XOR, LOOP, JMP – інструкції, які дійсно є "робітниками".

Рівні поліморфізму

Існує розподіл поліморфних вірусів на рівні залежно від складності коду, що зустрічається в розшифровувачах цих вірусів. Такий розподіл уперше

запропонував доктор Алан Соломон, через якийсь час Весселин Бончев розширив його:

Рівень 1: Віруси, що мають деякий набір розшифровувачів з постійним кодом; при зараженні вибирають один з них. Такі віруси є "напів-поліморфними" і носять також назву "олігоморфних" (oligomorphic). Приклади: "Cheeba", "Slovakia", "Whale".

Рівень 2: Розшифровувач вірусу містить одну або кілька постійних інструкцій, основна ж його частина непостійна.

Рівень 3: Розшифровувач містить невикористовувані інструкції – "сміття" типу NOP, CLI, STI і т.д.

Рівень 4: У розшифровувачі використовуються взаємозамінні інструкції й зміна порядку проходження (перемішування) інструкцій. Алгоритм розшифровки при цьому не змінюється.

Рівень 5: Використовуються всі перераховані вище прийоми, алгоритм розшифровки непостійний, можливо повторне шифрування коду вірусу й навіть часткове шифрування самого коду розшифровувача.

Рівень 6: Permutating-Віруси. Зміні підлягає основний код вірусу – він ділиться на блоки, які при зараженні переставляються в довільному порядку. Вірус при цьому залишається працездатним. Подібні віруси можуть бути незашифрованні.

У наведеній вище класифікації є свої недоліки, оскільки вона виконується за єдиним критерієм – можливість детектувати вірус по коду розшифровувача за допомогою стандартного прийому вірусних масок:

Рівень 1: для детектування вірусу досить мати кілька масок.

Рівень 2: детектування по масці з використанням "wildcards".

Рівень 3: детектування по масці після видалення інструкцій-"сміття".

Рівень 4: маска містить кілька варіантів можливого коду, тобто стає алгоритмічною.

Рівень 5: неможливість детектування вірусу по масці.

Недостатність такого розподілу продемонстрована у вірусі 3-го рівня поліморфізму, що так і називається – "Level3". Цей вірус, будучи одним з найбільш складних поліморфних вірусів, по наведеному вище розподілу попадає у рівень 3, оскільки має постійний алгоритм розшифровки, перед яким стоїть велика кількість команд-"сміття". Однак у цьому вірусі алгоритм генерування "сміття" доведений до досконалості: у коді розшифровувача можуть зустрітися практично всі інструкції процесора i8086.

Якщо зробити розподіл на рівні з погляду антивірусів, що використовують системи автоматичної розшифровки коду вірусу (емулятори), то розподіл на рівні буде залежати від складності емуляції коду вірусу. Можливо детектування вірусу й інших прийомів, наприклад, розшифровка за допомогою елементарних математичних законів і т.д.

Більш об'єктивною буде класифікація, у якій крім критерію вірусних масок беруть участь і інші параметри, наприклад:

– Ступінь складності поліморфного коду (відсоток від всіх інструкцій процесора, які можуть зустрітися в коді розшифровувача).

– Використання спеціальних прийомів, що утрудняють емуляцію антивірусами.

– Сталість алгоритму розшифровувача.

– Сталість довжини розшифровувача.

Зміна виконуваного коду

Найбільш часто подібний спосіб поліморфізму використовується макро-вірусами, які при створенні своїх нових копій випадковим чином міняють імена своїх змінних, вставляють порожні рядки або міняють свій код яким-небудь іншим способом. Таким чином, алгоритм роботи вірусу залишається без змін, але код вірусу практично повністю міняється від зараження до зараження.

Рідше цей спосіб застосовується складними завантажувальними вірусами. Такі віруси впроваджують у завантажувальні сектори лише досить коротку процедуру, що зчитує з диска основний код вірусу й передає на нього керування.

Код цієї процедури вибирається з декількох різних варіантів (які також можуть бути розведені "порожніми" командами), команди переставляються між собою й т.д.

Ще рідше цей прийом зустрічається у файлових вірусів – адже їм доводиться повністю міняти свій код, а для цього потрібні досить складні алгоритми. На сьогоднішній день відомі всього два таких віруси, один із яких ("Ply") випадковим чином переміщає свої команди по своєму тілу й заміняє їх на команди JMP або CALL. Інший вірус ("TMC") використовує більш складний спосіб – щоразу при зараженні вірус міняє місцями блоки свого коду й даних, вставляє "сміття", у своїх асемблерних інструкціях установлює нові значення оффсетів на дані, міняє константи й т.д. У результаті, хоча вірус і не шифрує свій код, він є поліморфним вірусом – у коді не присутній постійного набору команд. Більш того, при створенні своїх нових копій вірус міняє свою довжину.

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм MMB, в основі якого лежить змішування операцій різних алгебраїчних груп. MMB – ітеративний алгоритм, що складається з лінійних дій (XOR і використання ключа) і паралельного застосування чотирьох великих оборотних нелінійних підстановок. Ці підстановки визначаються за допомогою множення по модулю 2^{32}-1 з постійними множниками. У підсумку з'являється алгоритм, що використовує 128-бітовий ключ і 128-бітовий блок.

Алгоритм MMB оперує 32-бітовими підблоками тексту (x_0, x_1, x_2, x_3) і 32-бітовими підблоками ключу (k_0, k_1, k_2, k_3). Це спрощує реалізацію алгоритму на сучасних 64-бітових процесорах. Чергуючись із операцією XOR, шість разів використовується нелінійна функція *f*. Запишемо операції алгоритму (всі операції з індексами виконуються по модулю 4):

$$x_i = x_i \oplus k_i \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+1} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+2} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_i \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+1} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

$$x_i = x_i \oplus k_{i+2} \text{ для } i = 0..3$$

$$f(x_0, x_1, x_2, x_3)$$

Функція f виконується в три кроки:

1. $x_i = c_i * x_i$ для $i = 0..3$ (Якщо на вході множення одні одиниці, то на виході – теж одні одиниці).

2. Якщо молодший значущий біт $x_0 = 1$, то $x_0 = x_0 \oplus C$. Якщо молодший значущий байт $x_3 = 0$, то $x_3 = x_3 \oplus C$.

3. $x_i = x_{i-1} \oplus x_i \oplus x_{i+1}$ для $i = 0..3$.

Всі операції з індексами виконуються по модулю 4. Операція множення на кроці 1 виконується по модулі 2^{32}-1. Спеціальний випадок для даного алгоритму: якщо другий операнд дорівнює 2^{32}-1, результат теж дорівнює 2^{32}-1. В алгоритмі використовуються наступні константи:

C = 2aaaaaaa, c_0 = 025f1cdb, $c_1 = 2*c_0$, $c_2=2^3 *c_0$, $c_3=2^7 *c_0$.

Константа C – «найпростіша» константа без кругової симетрії, високою трійковою вагою й нульовим молодшим значущим бітом. У константи c_0 є інші особливі характеристики. Константи c_1, c_2 і c_3 – зрушені версії c_0, і служать для запобігання атак, заснованих на симетрії.

Розшифрування виконується у зворотному порядку, Етапи 2 і 3 інверсні їм самим. На етапі 1 замість c_i використовується c_i^{-1}. Значення c_0^{-1} = 0dad4694.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розроблене програмне забезпечення реалізує систему системи кібербезпеки для антивірусного захисту операційної системи від шкідливих програм.

Програмно-апаратні вимоги:

– Загальний обсяг ОЗП: 512 Мбайт.

– Вільний простір на жорсткому диску: 27 Мбайт.

– Операційна система Microsoft Windows.

Дана програма – це простий у використанні і в той же час повноцінний профілактичний антивірусний сканер з високою швидкістю сканування. Сканер має гнучку систему настроювань.

Даний антивірус забезпечує повноцінний захист комп'ютера від шкідливого ПЗ, а система Контроль процесів – постійно контролює всі процеси користувача, що дозволяє запобігти зараженню системи. Докладна система звітності, дозволяє перевірити всю інформацію про сканування, і зробити висновки про захищеність системи.

Дана програма виявляє віруси, троянські програми, руткіти та хробаків. Робить пошук і детектування наступних різновидів шкідливого ПЗ:

1. SpyWare, AdvWare програм.
2. Руткітів та інших шкідливих програм.
3. Мережних і поштових хробаків.
4. Троянських програм.

В програму вбудована потужна модульна система, що забезпечує додавання нових можливостей у сканер. Кожний користувач може створити свій унікальний модуль, що у свою чергу забезпечує максимальну гнучкість сканера.

Головне вікно програми зображене на рисунку 5.1.

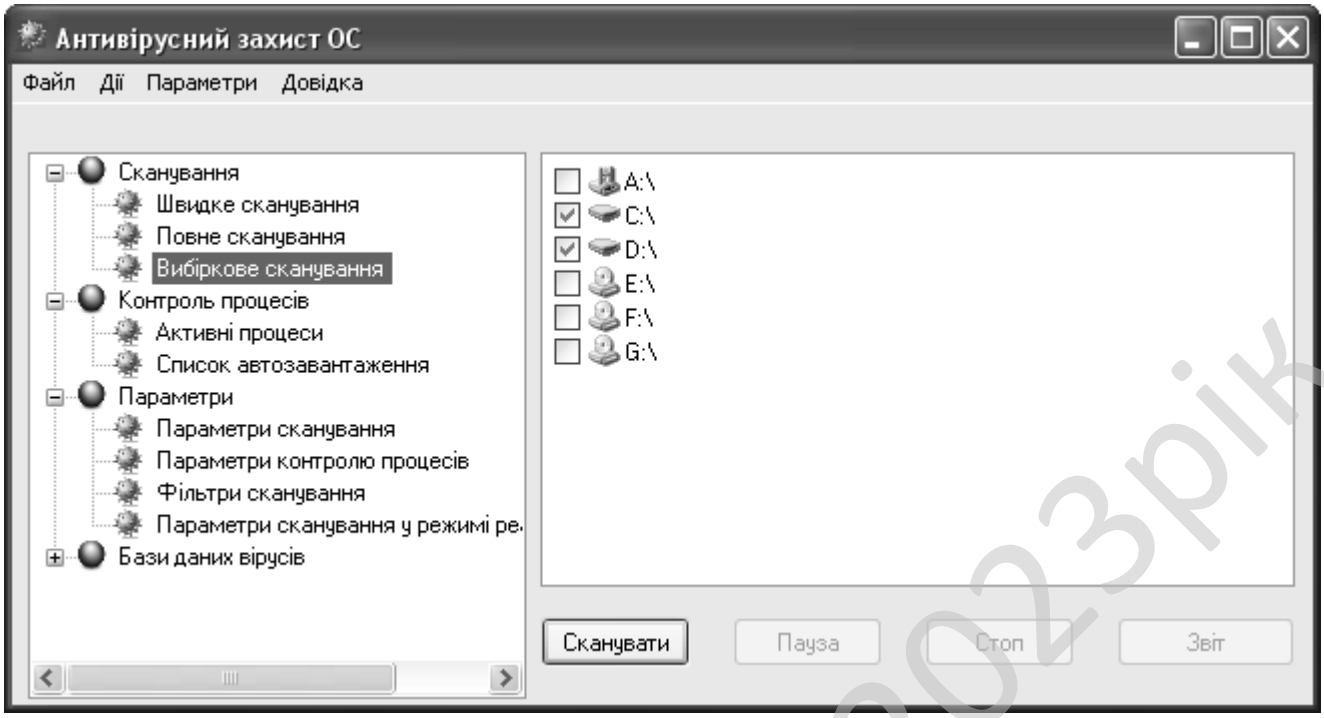

Рисунок 5.1 – Головне вікно програми

На ньому розташовані наступні посилання:

– Сканування.

– Параметри.

– Контроль.

– Про програму…

Та кнопки:

– Сканувати.

– Зберегти звіт.

– Вихід.

А також воно містить список дисків встановлених у системі, з якого користувач вибирає ті, які потрібно сканувати.

Для запуску процесу сканування слід вибрати диски, що необхідно перевірити та натиснути кнопку «Сканувати», процес сканування зображено на рисунку 5.2.

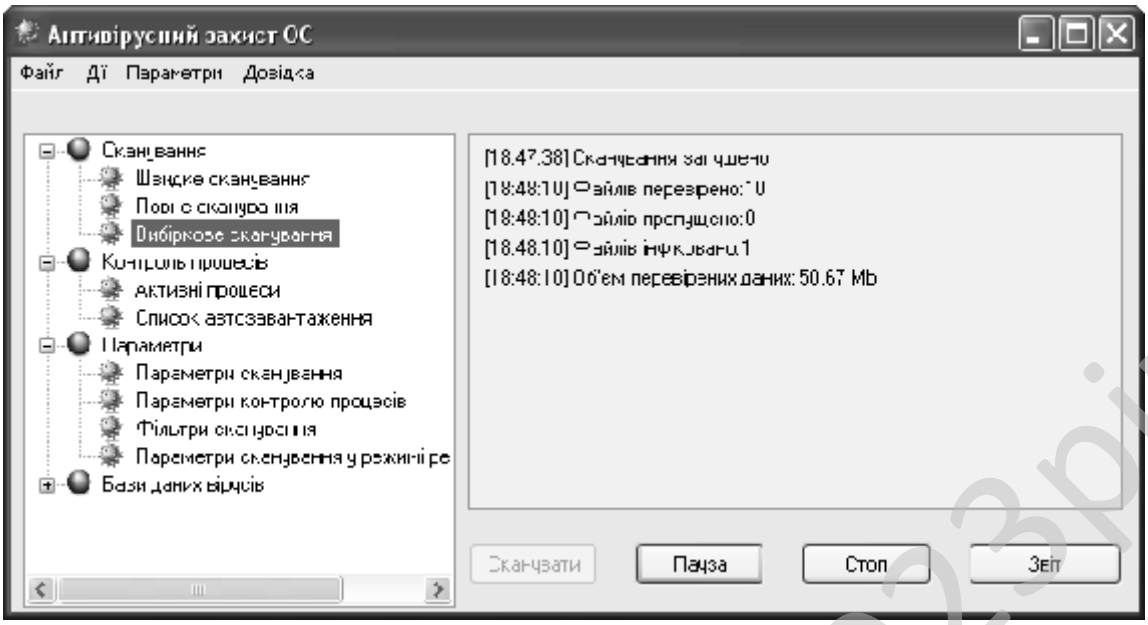

Рисунок 5.2 – Сканування

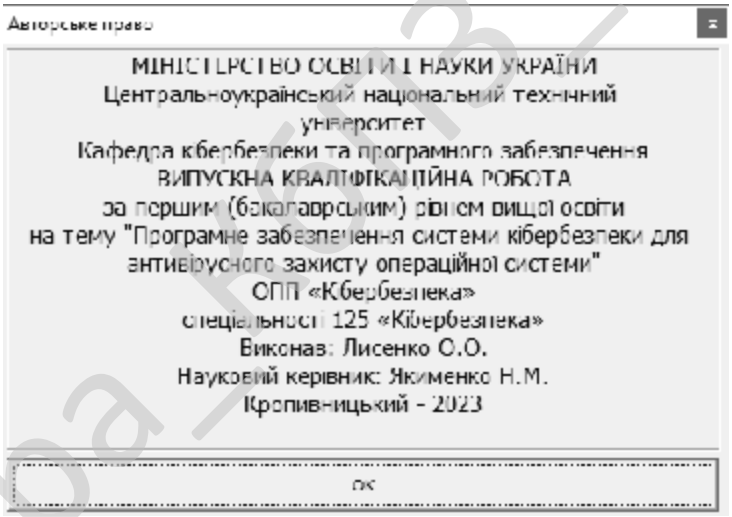

Рисунок 5.3 – Вікно «Про програму…»

В розробленій програмі є система Контроль процесів, призначена для автоматичного виявлення інфікованих об'єктів, шляхом перевірки запущених процесів в системі. Коротку довідку про розроблену програму можна переглянути натиснувши посилання "Про програму…", після чого з'явиться вікно зображене на рисунку 5.3.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки для антивірусного захисту операційної системи.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем для антивірусного захисту операційної системи.

– Досліджена система для антивірусного захисту операційної системи.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки для антивірусного захисту операційної системи.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання для антивірусного захисту операційної системи.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня RAD Studio Delphi. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи кібербезпеки для антивірусного захисту операційної системи. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення

ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи кібербезпеки й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи кібербезпеки Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм ММВ.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Семенов С.Г. Методика математического моделирования защищенной ИТС на основе многослойной GERT-сети / С.Г. Семенов // Вісник Національного технічного університету «Харківський політехнічний інститут». – Х.:НТУ «ХПІ». – 2012. –№62 (968). – С 173-181.

2. Семенов С.Г. Защита данных в компьютеризированных управляющих системах / С.Г. Семенов, В.В. Давыдов, С.Ю. Гавриленко. – LAP Lambert Academic Publishing GmbH & Co. KG (Саарбрюккен, Германия), 2014. – 236 с.

3. Смирнов А.А. Анализ и сравнительное исследование перспективных направлений развития цифровых телекоммуникационных систем и сетей / А.А.Смирнов, В.В.Босько, Е.В.Мелешко // Системи обробки інформації. – Х.: ХУ ПС, 2008. – Вип.7(74). – С.120-123.

4. Смирнов А.А. Усовершенствование метода управления очередями в многопротокольных узлах телекоммуникационной сети / А.А.Смирнов, Е.В.Мелешко // Збірник тез та доповідей другої всеукраїнської науково-практичної конференції «Системний аналіз. Інформатика. Управління». Запоріжжя. Тези доповідей. Запоріжжя: КПУ, 2011.

5. Смирнов С.А. Метод безопасной маршрутизации метаданных в облачные антивирусные системы / А.К. Дидык, С.А. Смирнов // Информационные технологии в управлении, образовании, науке и промышленности: монография / Под редакцией профессора В.С. Пономаренко. – Х.: Видавець Рожко С.Г., 2016. – 566 с.

6. Смирнов С. А. Сравнительные исследования математических моделей технологии распространения компьютерных вирусов в информационно-телекоммуникационных сетях / Мохамад Абу Таам Гани, А. А. Смирнов, А. В. Коваленко, С. А. Смирнов // Системи обробки інформації: зб. наук. праць. – Х.: ХУПС, 2014. – Вип. 9(125). – 105-110.

7. Смирнов С. А. Математическая модель интеллектуального узла коммутации с обслуживанием информационных пакетов различного приоритета / Мохамад Абу Таам Гани, А. А. Смирнов, Н. С. Якименко, С. А. Смирнов // Збірник наукових праць Харківського університету Повітряних Сил. – Харків: ХУПС, 2014. – Вип. 4 (41). – С. 48-52.

8. Смирнов С. А. Исследование показателей качества функционирования интеллектуальных узлов коммутации в телекоммуникационных системах и сетях / Мохамад Абу Таам Гани, А. А. Смирнов, Н. С. Якименко, С. А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України: наук. журн. – Х.: ХУПС, 2014. – № 4(17). – С. 90-95.

9. Смирнов С. А. Усовершенствованный алгоритм управления доступом к «облачным» телекоммуникационным ресурсам / Мохамад Абу Таам Гани, А. А. Смирнов, Н. С. Якименко, С. А. Смирнов // Системи обробки інформації: зб. наук. праць. – Х.: ХУПС, 2015. –Вип. 1(126). – С. 150-153.

10. Smirnov S.A. Method of controlling access to intellectual switching nodes of telecommunication networks and systems / A.A. Smirnov, Mohamad Abou Taam, S.A. Smirnov // International Journal of Computational Engineering Research (IJCER). – Volume 5, Issue 5. – India. Delhi. – 2015. – P. 1-7.

11. Смирнов С. А. Анализ и исследование методов управления сетевыми ресурсами для обеспечения антивирусной защиты даннях / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Системи озброєння і військова техніка: наук. журн. – Х.: ХУПС, 2015. – № 3(43). – С. 100-107.

12. Смирнов С. А. Исследование эффективности метода управления доступом к облачным антивирусным телекоммуникационным ресурсам / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України: наук. журн. –Х.: ХУПС, 2015. – № 3(20). – С. 134-141.

13. Смирнов С. А. Комплекс gert-моделей технологии облачной антивирусной защиты телекоммуникационной системы / А. А. Смирнов,

А. К. Дидык, А. Н. Дреев, С. А. Смирнов // Безпека інформації: наук. - практ. журн. – К.: НАУ, 2015. – Т. 21, № 3. – С. 251-262.

14. Смирнов С. А. Метод безопасной маршрутизации метаданных в облачные антивирусные системы / А. А. Смирнов, А. К. Дидык, С. А. Смирнов // Системи озброєння і військова техніка: наук. журн. – Х.: ХУПС, 2016. – № 2 (46). – С. 146-149.

15. Смирнов С. А. Модели системы нейросетевых экспертов безопасной маршрутизации в облачных антивирусных системах / А. А. Смирнов, А. К. Дидык, А. Н. Дреев, С. А. Смирнов // Системи обробки інформації: зб. наук. праць. – Х.: ХУПС, 2016. – Вип. 3 (140). – С. 36-39.

16. Смирнов С. А. Метод безопасной маршрутизации на базовом множестве путей передачи метаданных в облачные антивирусные системы / В. Л. Бурячок, С. А. Смирнов // Системи управління, навігації та зв'язку. – Полтава, 2016. – Вип. 4(40). – С. 57-62.

17. Смирнов С. А. Способ контроля линий связи телекоммуникационной системы облачного антивируса / А. А. Смирнов, А. К. Дидык, А. Н. Дреев, С. А. Смирнов // Збірник наукових праць Харківського університету Повітряних Сил. – Харків: ХУПС, 2016. – № 2 (47). – С. 148-152.

18. Смирнов С. А. Дослідження та реалізація GERT-моделі технології розповсюдження комп'ютерних вірусів для захисту телекомунікаційних систем / В. Л. Бурячок, Мохамад Абу Таам Гани, С. А. Смирнов // Інформаційні технології та комп'ютерна інженерія: зб. тез доп. наук.-практ. конф., м. Кіровоград, 4 грудня 2014 р. – Кіровоград: КНТУ, 2014. – С. 168.

19. Смирнов С. А. Исследование математических моделей технологии распространения компьютерных виру сов / А. А. Смирнов, Мохамад Абу Таам Гани, С. А. Смирнов // Актуальні питання забезпечення кібернетичної безпеки та захисту інформації: зб. наук. праць міжнар. наук.-практ. конф., м. Київ, 25-28 лютого 2015 р. – К.: Європейський університет, 2015. – С. 90-91.

20. Смирнов С. А. Метод управления доступом к «облачным» ресурсам для защиты телекоммуникационных систем / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Всеукраїнська науково-практична конференція «Інформаційна безпека держави, суспільства та особистості», м. Кіровоград, 16 квітня 2015 р.: зб. тез доп. – Кіровоград: КНТУ, 2015. – С. 50-52.

21. Смирнов С. А. Разработка метода управления доступом в интеллектуальных узлах коммутации / А. А. Смирнов, Мохамад Абу Таам Гани, С. А. Смирнов // Проблеми і перспективи розвитку ІТ-індустрії: зб. тез VII міжнар. наук.-практ. конф., м. Харків, 17-18 квітня 2015 р. – Х.: ХНЕУ, 2015. – С. 14.

22. Смирнов С.А. Реализация метода управления доступом в интеллектуальных узлах коммутации / А.А. Смирнов, Мохамад Абу Таам Гани, С.А. Смирнов // Збірник тез XVII міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування». м. Кіровоград. 17-18 квітня 2015 р. – Кіровоград: КНТУ. – 2015. – С. 91-92.

23. Смирнов С. А. технология передачи сигнатур в облачные антивирусные системы для обеспечения защищенности телекоммуникационных сетей / А. А. Смирнов, С. А. Смирнов // Збірник тез V міжнародної науково-технічної конференції «ITSEC», Київ, 19-22 травня 2015 р. – К.: НАУ 2015. – С. 12-13.

24. Смирнов С. А. Реализация математической модели интеллектуального узла коммутации для обеспечения защищенности телекоммуникационной сети / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Інформаційна та економічна безпека (INFECO-2015): зб. тез II Міжнар. наук.-практ. Інтернет-конф., м. Харків, 21-22 травня 2015 р. – Х.: ХІБС УБС НБУ, 2015. – С. 20-24.

25. Смирнов С. А. Разработка математической модели технологии распространения компьютерных вирусов в информационно-телекоммуникационных сетях / Мохамад Абу Таам Гани, А. А. Смирнов,

С. А. Смирнов // Сборник тезисов XI международной конференции «Стратегия качества в промышленности и образовании», г. Варна, Болгария, 01-06 июня 2015 г. – Варна: ТУВ, 2015. – С. 488-491.

26. Смирнов С. А. Метод управления доступом к облачным телекоммуни-кационным ресурсам для обеспечения защиты даннях / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Комп'ютерні технології та інформаційна безпека: зб. тез доп. міжнар. наук.-практ. конф., м. Кіровоград, 2-3 липня 2015 р. – Кіровоград: КНТУ, 2015. – С. 4-5.

27. Смирнов С. А. Имитационная модель системы управления доступом к облачным антивирусным телекоммуникационным ресурсам / Мохамад Абу Таам Гани, А. А. Смирнов, С. А. Смирнов // Збірник тез першої всеукраїнської науково-практичної конференції «Перспективні напрями захисту інформації» (м. Затока, 7-9 вересня 2015 р.). – Одеса: ОНАЗ, 2015. – С. 90-94.

28. Смирнов С. А. Разработка комплекса gert-моделей технологии облачной антивирусной защиты телекоммуникационной системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Інформаційні технології та взаємодії» (IT & I): зб. тез II міжнар. наук.-практ. конф., м. Київ, 3-5 листопада 2015 р. – К.: КНУ ім. Тараса Шевченка, 2015. – С. 65-67.

29. Смирнов С. А. Разработка моделей телекоммуникационной системы формирования и обработки метаданных в облачных антивирусных системах / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Информационные и телекоммуникационные технологии: образование, наука, практика: сб. тезисов II междунар. научно-практ. конф., г. Алматы, Казахстан, 3-4 декабря 2015 г. – Алматы: КазНИТУ им. К.И. Сатпаева, 2015. – С. 309-313.

30. Смирнов С. А. gert-модели технологии облачной антивирусной защиты / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Безпека українського суспільства в концепції вступу в постіндустріальне суспільство ЄС: зб. тез Круглого столу, м. Київ, 16 грудня 2015 р. – К.: Європейський університет, 2015. – С.41-43.

31. Смирнов С. А. Алгоритмы формирования множества маршрутов передачи метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Актуальні питання забезпечення кібернетичної безпеки та захисту інформації: зб. наук. праць II Міжнар. наук.-практ. конф., м. Київ, 24-27 лютого 2016 р. – К.: Європейський університет, 2016. – С. 140-142.

32. Смирнов С. А. Разработка и реализация метода безопасной маршрутизации метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Securitea informationala 2015-2016: Conferenta internationala (editia a XII-a), Chisinau, Moldova, 3 martie 2016. – Chisinau: ADSEM, 2016. – С. 90-96.

33. Смирнов С. А. Алгоритм формирования базового множества маршрутов передачи метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Інформатика та системні науки (ІСН-2016): зб. тез VII всеукр. наук.-практ. конф., м. Полтава, 10-12 березня 2016 р. – Полтава: ПУЕТ, 2016. – С. 261-263.

34. Смирнов С. А. Система обработки и формирования начального состояния маршрутизации метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Проблеми кібербезпеки інформаційно-телекомунікаційних систем: зб. тез наук.-практ. конф., м. Київ, 10-11 березня 2016 р. – К.: КНУ ім. Тараса Шевченка, 2016. – С. 81-82.

35. Смирнов С. А. Алгоритм безопасной маршрутизации на базовом множестве путей передачи метаданных в программный сервер облачной антивирусной системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Інформаційна безпека та комп'ютерні технології (IS&CT): зб. тез міжнар. наук.-практ. конф., м. Кіровоград, 24-25 березня 2016 р. – Кіровоград: КНТУ, 2016. – С. 73.

36. Смирнов С. А. Исследование способа контроля линий связи телекоммуникационной системы для облачных антивирусов / А. А. Смирнов,

С. А. Смирнов, А. К. Дидык // Збірник тез першої міжнародної науково-практичної конференції «Проблеми науково-технічного та правового забезпечення кібербезпеки у сучасному світі» (ПНПЗК-2016), м. Харків, 30 березня - 1 квітня 2016 р. – Х.: НТУ «ХПІ», 2016. – С. 14.

37. Смирнов С. А. Разработка способа контроля линий связи телекоммуникационной системы для облачных антивирусов / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Матеріали XVIII міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування» (м. Кіровоград, 15-16 квітня 2016 р.). –Кіровоград: КНТУ, 2016. – С. 182-186.

38. Смирнов С. А. Разработка и исследование способа контроля линий связи телекоммуникационных сетей для облачных антивирусных систем / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Проблеми і перспективи розвитку ІТ-індустрії: VIII міжнар. наук.-практ. конф., м. Харків, 28-29 квітня 2016 р.: зб. тез. – Х.: ХНЕУ, 2016. – С. 48.

39. Смирнов С. А. Модель системы нейросетевых экспертов безопасной маршрутизации для облачных антивирусных систем / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Інформаційна та економічна безпека (INFECO-2016): зб. тез ІІІ міжнар. наук.-практ. конф., м. Харків, 28-30 кві. 2016 р. – Х.: ХННІ ДВНЗ «УБС», 2016. – С. 178-182.

40. Смирнов С. А. Метод безопасной маршрутизации метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. К. Дидык // Сборник тезисов XII международной конференции «Стратегия качества в промышленности и образовании» (г. Варна, Болгария, 30 мая - 02 июня 2016 г.). – Варна: ТУВ, 2016. – С. 581-585.

41. Смирнов С. А. Оценка эффективности метода безопасной маршрутизации метаданных в облачные антивирусные системы / А. А. Смирнов, С. А. Смирнов, А. С. Коваленко // РадіоЕлектроніка та ІнфоКомунікації: зб. тез першої наук. - техн. конф., м. Київ, 11-16 вересня 2016 р. – К.: НТУУ «КПІ», 2016. – С. 17.

42. Современные телекоммуникации. Технологии и экономика / [В.Л. Банкет, О.В. Бондаренко, П.П. Воробьенко и др.]; под ред. С.А. Довгого. – М.: Эко-Трендз, 2003. – 320 с.

43. Столлингс В. Современные компьютерные сети / Вильям Столлингс. –СПб.: Питер, 2003. – 778 с.

44. Таненбаум Э. Компьютерные сети / Эндрю Таненбаум; пер. с англ. А. Леонтьев. – СПб.: Питер, 2002. – 848 с.

45. Телекоммуникационные системы и сети: учебное пособие. В 3 томах / [В.В. Величко, Е.А. Субботин, В.П. Шувалов, А.Ф. Ярославцев]; под ред. В.П. Шувалова. – М.: Горячая линия-Телеком, 2005, т. 3 – 592 с.

46. Хайкин С. Нейронные сети: полный курс / С. Хайкин. – М.:Вильямс, 2006. – 1103 с.

47. Шелухин О.И. Фрактальные процессы в телекоммуникациях: моногр. / О.И. Шелухин, А.М. Тенякшев, А.В. Осин – М.: Радиотехника, 2003. – 480 с.

48. Elwalid, D. Mitra, I. Saniee, and I. Widjaja. Routing and Protection in GMPLS Networks: From Shortest Paths to Optimized Designs // Journal of lightwave technology. – 2003. – №21(11), P. 2828-28-38.

49. A.B. Bagula, M. Botha, and A.E Krzesinski. Online Traffic Engineering: The Least Interference Optimization Algorithm // IEEE Communications Society – 2004, P. 1232-1236.

50. Anees Shaikh, Jennifer Rexford, and Kang G. Shin. Evaluating the Impact of Stale Link State on Quality-of-Service Routing // IEEE/ACM Transactions on Networking. – 2001. – №9(2), P. 162-176.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки для антивірусного захисту операційної системи.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 12-02 від 5.01.2023 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи кібербезпеки для антивірусного захисту операційної системи.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи кібербезпеки з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи кібербезпеки для антивірусного захисту операційної системи;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп'ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14" 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище RAD Studio Delphi.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Пояснювальна записка – 79 аркушів.

8 Етапи розробки

8.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2023 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 8.06.2023 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

першим (бакалаврським) рівнем вищої освіти

_____ Якименко Н.М.

Програмне забезпечення системи кібербезпеки для антивірусного захисту операційної системи

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 40

Літера: РП

Кропивницький – 2023 року

Файл debug.dpr – головний файл проекту

```
program debug;
// Список підключаємих модулів
uses
  Forms,
  SysUtils,
  avKernel in '..\AntiVir Scanner Modues\avKernel.pas',
  avTypes in '..\AntiVir Scanner Modues\avTypes.pas',
  avMonitor in '..\AntiVir Scanner Modues\avMonitor.pas',
  avScanner in '..\AntiVir Scanner Modues\avScanner.pas',
  avHex in '..\AntiVir Scanner Modues\avHex.pas',
  avDataBase in '..\AntiVir Scanner Modues\avDataBase.pas',
  avHash in '..\AntiVir Scanner Modues\avHash.pas',
  avExt in '..\AntiVir Scanner Modues\avExt.pas',
  avAPI in '..\AntiVir Scanner Modues\avAPI.pas',
  avConfig in '..\AntiVir Scanner Modues\avConfig.pas',
  avShield in '..\AntiVir Scanner Modues\avShield.pas',
  langs in 'langs.pas',
  AntiVir_Main in 'AntiVir_Main.pas' {MainForm},
  AntiVir_SelInfo in 'AntiVir_SelInfo.pas' {InformationForm},
  AntiVir_Options in 'AntiVir_Options.pas' {OptionsForm},
  AntiVir_PluginInfo in 'AntiVir_PluginInfo.pas' {PluginAPIForm},
  AntiVir_AddPath in 'AntiVir_AddPath.pas' {AddUserPathForm},
  AboutFrm in 'AboutFrm.pas' {AboutForm},
  AntiVir_SelDir in 'AntiVir_SelDir.pas' {SelDirFrm},
  AntiVir_Message in 'AntiVir_Message.pas' {MessageFrm},
  AntiVir_HideForm in 'AntiVir_HideForm.pas' {HideForm},
  AntiVir_Monitor in 'AntiVir_Monitor.pas' {MonitorForm},
  AntiVir_InfectedAction in 'AntiVir_InfectedAction.pas' {ActionForm},
  AntiVir_Splash in 'AntiVir_Splash.pas' {SplashForm};

{$R *.res}

begin
  Application.Initialize;
  Application.Title := 'Scanner';
  Application.CreateForm(TMainForm, MainForm);
  Application.CreateForm(TInformationForm, InformationForm);
  Application.CreateForm(TOptionsForm, OptionsForm);
  Application.CreateForm(TPluginAPIForm, PluginAPIForm);
  Application.CreateForm(TAddUserPathForm, AddUserPathForm);
  Application.CreateForm(TAboutForm, AboutForm);
  Application.CreateForm(TSelDirFrm, SelDirFrm);
  Application.CreateForm(TMessageFrm, MessageFrm);
  Application.CreateForm(THideForm, HideForm);
  Application.CreateForm(TMonitorForm, MonitorForm);
  Application.CreateForm(TActionForm, ActionForm);
  Application.CreateForm(TSplashForm, SplashForm);
  {Show Splash form}
  SplashForm.CRLabel.Caption   := 'Kernel '+GetKernelVersion;
  SplashForm.CRLabel00.Caption := 'Build ' +GetKernelBuild;
  SplashForm.Show;
  {}
  Init;
  langs.SwitchAllFormsToLng(01,01,ExtractFilePath(Paramstr(0))+'default.lng');
  {init kernel}
  MainForm.InitScannerKernel;
  {Hide Splash Form}
  SplashForm.Hide;
  Sleep(200);
  {Create Tray Icon}
  MainForm.CreateTray;
  {}
  if OptionsForm.AUTORUN.Checked then begin
    OptionsForm.ChangeReg('Scanner',False);
  end else begin
```

```
    OptionsForm.ChangeReg('Scanner',True);
  end;
  {}
  if ParamStr(1) <> '' then
  MainForm.StartScan(ParamStr(1));
  {}
  if OptionsForm.PCAutoLoad.Checked then begin
    MonitorForm.StartMonitor;
  end;
  {}
  Application.Run;
end.
```

Файл AntiVir_Options.pas – параметри антивірусу

```
unit AntiVir_Options;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Buttons, ComCtrls, registry, avKernel, avTypes;

type
  TOptionsForm = class(TForm)
    Bevel: TBevel;
    TopPanel: TPanel;
    BackImage: TImage;
    InformationLabel: TLabel;
    InfoLabel: TLabel;
    ApplyBTN: TButton;
    CanselBTN: TButton;
    OptionsPages: TPageControl;
    optTabOther: TTabSheet;
    optTabPathes: TTabSheet;
    optTabModules: TTabSheet;
    AutoSaveReport: TCheckBox;
    ReportSavePath: TEdit;
    EditSaveReportBTN: TSpeedButton;
    optTabFilter: TTabSheet;
    ExtList: TListView;
    PathList: TListView;
    APIList: TListView;
    AddBTN: TSpeedButton;
    DelBTN: TSpeedButton;
    EditBTN: TSpeedButton;
    SaveDialog: TSaveDialog;
    DisplayScnFiles: TCheckBox;
    optReportLabel: TLabel;
    optSysLabel: TLabel;
    RegisterSysMenu: TCheckBox;
    OPTModulePanel: TPanel;
    ModulesLOAD: TCheckBox;
    optModInfLabel: TLabel;
    optModListLabel: TLabel;
    optShieldLabel: TLabel;
    USESHIELD: TCheckBox;
    SHIELDSILENT: TCheckBox;
    optTabMain: TTabSheet;
    DBDirLabel: TLabel;
    DBPATH: TEdit;
    Bevel6: TBevel;
    optPathesLabel: TLabel;
    SpeedButton1: TSpeedButton;
    ModDirLabel: TLabel;
    MODULESPATH: TEdit;
    SpeedButton2: TSpeedButton;
    Bevel7: TBevel;
    optScanLabel: TLabel;
    SCNSUBDIR: TCheckBox;
    SCNHEX: TCheckBox;
    SCNCRC: TCheckBox;
    SCNHEXINPOS: TCheckBox;
    SCNBIT: TCheckBox;
    AUTORUN: TCheckBox;
    AUTOHIDE: TCheckBox;
    Image1: TImage;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Bevel5: TBevel;
    Bevel3: TBevel;
    Bevel4: TBevel;
```

```
    optTabPC: TTabSheet;
    optPCLabel: TLabel;
    Bevel8: TBevel;
    PCAutoLoad: TCheckBox;
    PCAutoKill: TCheckBox;
    PCAutoAction: TCheckBox;
    PCDelInfect: TRadioButton;
    PCSkipInfect: TRadioButton;
    optPCInfoLabel: TLabel;
    SHOWBALOONHINT: TCheckBox;
    procedure ApplyBTNClick(Sender: TObject);
    procedure optTabOtherShow(Sender: TObject);
    procedure optTabFilterShow(Sender: TObject);
    procedure optTabPathesShow(Sender: TObject);
    procedure optTabModulesShow(Sender: TObject);
    Procedure SaveOptions;
    procedure CanselBTNClick(Sender: TObject);
    procedure APIListDblClick(Sender: TObject);
    procedure AddBTNClick(Sender: TObject);
    procedure DelBTNClick(Sender: TObject);
    procedure EditBTNClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure EditSaveReportBTNClick(Sender: TObject);
    procedure FileTAddAction(key, name, display, action: String);
    procedure FileTDelAction(key, name: String);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure optTabMainShow(Sender: TObject);
    procedure ChangeReg(StrName: ShortString; delete: boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  OptionsForm: TOptionsForm;

implementation

uses AntiVir_Main, AntiVir_PluginInfo, AntiVir_AddPath, AntiVir_SelDir,
AntiVir_HideForm;

{$R *.dfm}
//***********Запис у реєстр системи***********
procedure TOptionsForm.ChangeReg(StrName: ShortString; delete: boolean);
  var
  reg: TRegistry;
begin
  Reg := nil;
  try
    reg := TRegistry.Create;
    reg.RootKey := HKEY_LOCAL_MACHINE;
    reg.LazyWrite := false;
    reg.OpenKey('Software\Microsoft\Windows\CurrentVersion\Run',false);
  if not delete then reg.WriteString(StrName, ParamStr(0)+' -M')
  else reg.DeleteValue(StrName);
    reg.CloseKey;
    reg.free;
  except
    if Assigned(Reg) then Reg.Free;
  end;
end;

procedure TOptionsForm.FileTDelAction(key, name: String);
  var
  myReg: TRegistry;
begin
  try
```

```
    myReg:=TRegistry.Create;
    myReg.RootKey:=HKEY_CLASSES_ROOT;
    if key[1] = '.' then
      key := copy(key,2,maxint)+'_auto_file';
    if key[Length(key)-1] <> '\' then
      key:=key+'\';
    myReg.OpenKey('\'+key+'shell\', true);
    if myReg.KeyExists(name) then
      myReg.DeleteKey(name);
    myReg.CloseKey;
    myReg.Free;
  except
  end;
end;

procedure TOptionsForm.FileTAddAction(key, name, display, action: String);
  var
  myReg:TRegistry;
begin
  try
    myReg:=Tregistry.Create;
    myReg.RootKey:=HKEY_CLASSES_ROOT;
    if name='' then name:=display;

    if key[1] = '.' then
      key:= copy(key,2,maxint)+'_auto_file';

    if key[Length(key)-1] <> '\' then
      key:=key+'\';
    if name[Length(name)-1] <> '\' then
      name:=name+'\';
    myReg.OpenKey(key+'Shell\'+name, true);
    myReg.WriteString('', display);
    MyReg.CloseKey;
    MyReg.OpenKey(key+'Shell\'+name+'Command\', true);
    MyReg.WriteString('', action);
    myReg.Free;
  except
  end;
end;

Procedure TOptionsForm.SaveOptions;
  var
  i:integer;
begin
  if AUTORUN.Checked then
  begin
    ChangeReg('Scanner',False);
  end else
  begin
    ChangeReg('Scanner',True);
  end;

 //******//

        OPT_MODULES_LOAD        := ModulesLOAD.Checked;
        OPT_DB_DIR              := DBPATH.Text;
        OPT_MODULE_DIR          := MODULESPATH.Text;
        OPT_USE_SHIELD          := USESHIELD.Checked;
        OPT_SILENT_SHIELD_MODE  := SHIELDSILENT.Checked;
        OPT_SCAN_SUBDIR         := SCNSUBDIR.Checked;
        OPT_USE_HEX_MODE        := SCNHEX.Checked;
        OPT_USE_CRC_MODE        := SCNCRC.Checked;
        OPT_USE_HEX_INPOS       := SCNHEXINPOS.Checked;
        OPT_SEND_SCAN_FILE      := DisplayScnFiles.Checked;
        OPT_USE_BYTE_MODE       := SCNBIT.Checked;
//******//
        ClearOtherParamList;
//******//
```

```
        if SHOWBALOONHINT.Checked then AddOtherParamString('SHOWBALOONHINT=ON')
        else AddOtherParamString('SHOWBALOONHINT=OFF');

        if PCAutoLoad.Checked then AddOtherParamString('PROCCONTROLAUTOMODE=ON')
        else AddOtherParamString('PROCCONTROLAUTOMODE=OFF');

        if PCAutoKill.Checked then AddOtherParamString('PROCCONTROLAUTOKILL=ON')
        else AddOtherParamString('PROCCONTROLAUTOKILL=OFF');

        if PCAutoAction.Checked then
AddOtherParamString('PROCCONTROLAUTOACTION=ON')
        else AddOtherParamString('PROCCONTROLAUTOACTION=OFF');

        if PCDelInfect.Checked then
AddOtherParamString('PROCCONTROLDELINFECT=ON')
        else AddOtherParamString('PROCCONTROLDELINFECT=OFF');

        if PCSkipInfect.Checked then
AddOtherParamString('PROCCONTROLSKIPINFECT=ON')
        else AddOtherParamString('PROCCONTROLSKIPINFECT=OFF');

        if AutoSaveReport.Checked then AddOtherParamString('AUTOSAVEREPORT=ON')
        else
        AddOtherParamString('AUTOSAVEREPORT=OFF');
        AddOtherParamString('AUTOSAVEREPORTTO='+ReportSavePath.Text);

        if RegisterSysMenu.Checked then
AddOtherParamString('REGISTERSYSMENU=ON')
        else AddOtherParamString('REGISTERSYSMENU=OFF');

        if AutoRun.Checked then AddOtherParamString('AUTORUN=ON')
        else
        AddOtherParamString('AUTORUN=OFF');

        if AutoHide.Checked then AddOtherParamString('AUTOHIDE=ON')
        else
        AddOtherParamString('AUTOHIDE=OFF');

        if HideForm.ShowHideTip.Checked then AddOtherParamString('HIDETIP=ON')
        else
        AddOtherParamString('HIDETIP=OFF');

        ClearExtList;
        for i := 0 to ExtList.Items.Count-1 do
        AddToExtList(ExtList.Items.Item[i].Caption);

        for i := 0 to PathList.Items.Count-1 do
        AddOtherParamString('PATH='+PathList.Items.Item[i].Caption);
//******//
        SaveConfig_;
//******//
end;

procedure TOptionsForm.ApplyBTNClick(Sender: TObject);
begin
  SaveOptions;
  MainForm.CreateDrivesList(MainForm.PathList);
  if RegisterSysMenu.Checked then
  begin
    FileTAddAction('*','AntiVir.Scan',MainForm.SysMenu,ParamStr(0)+' %1');
    FileTAddAction('Directory','AntiVir.Scan',MainForm.SysMenu,ParamStr(0)+'
%1');
    FileTAddAction('Drive','AntiVir.Scan',MainForm.SysMenu,ParamStr(0)+' %1');
  end else
  begin
    FileTDelAction('Drive','AntiVir.Scan');
    FileTDelAction('Directory','AntiVir.Scan');
    FileTDelAction('*','AntiVir.Scan');
  end;
```

```
  Close;
end;

procedure TOptionsForm.optTabOtherShow(Sender: TObject);
begin
  AddBTN.Enabled := False;
  DelBTN.Enabled := False;
  EditBTN.Enabled := False;
end;

procedure TOptionsForm.optTabFilterShow(Sender: TObject);
begin
  AddBTN.Enabled := true;
  DelBTN.Enabled := true;
  EditBTN.Enabled := true;
end;

procedure TOptionsForm.optTabPathesShow(Sender: TObject);
begin
  AddBTN.Enabled := True;
  DelBTN.Enabled := True;
  EditBTN.Enabled := False;
end;

procedure TOptionsForm.optTabModulesShow(Sender: TObject);
begin
  AddBTN.Enabled := False;
  DelBTN.Enabled := False;
  EditBTN.Enabled := False;
end;

procedure TOptionsForm.CanselBTNClick(Sender: TObject);
begin
  Close;
end;

procedure TOptionsForm.APIListDblClick(Sender: TObject);
begin
  if APIList.ItemIndex <> -1 then
  begin
    PluginAPIForm.NameEdit.Text := APIList.Selected.Caption;
    PluginAPIForm.AutorEdit.Text := APIList.Selected.SubItems[0];
    PluginAPIForm.OtherMemo.Text := APIList.Selected.SubItems[1];
    PluginAPIForm.PathEdit.Text := APIList.Selected.SubItems[2];
    PluginAPIForm.ShowModal;
  end;
end;

procedure TOptionsForm.AddBTNClick(Sender: TObject);
begin
  if optTabFilter.Showing then
  begin
    with ExtList.Items.Add do begin
      Caption := '';
      ImageIndex := 3;
      EditCaption;
    end;
  end;
  if optTabPathes.Showing then AddUserPathForm.Showmodal;
end;

procedure TOptionsForm.DelBTNClick(Sender: TObject);
begin
  try
    if optTabFilter.Showing then ExtList.Items.Delete(ExtList.Selected.Index);
    if optTabPathes.Showing then PathList.Items.Delete(PathList.Selected.Index);
  except
  end;
end;
```

```
procedure TOptionsForm.EditBTNClick(Sender: TObject);
begin
  if optTabFilter.Showing then
    if ExtList.ItemIndex <> -1 then
      ExtList.Selected.EditCaption;
end;

procedure TOptionsForm.FormShow(Sender: TObject);
begin
  optTabMain.Show;
end;

procedure TOptionsForm.EditSaveReportBTNClick(Sender: TObject);
begin
  if SaveDialog.Execute then ReportSavePath.Text := SaveDialog.FileName;
end;

procedure TOptionsForm.SpeedButton1Click(Sender: TObject);
begin
  SelDirFrm.ShowModal;
  if SelDirFrm.ModalResult = mrOk then
  begin
    DBPATH.Text := SelDirFrm.ShellTreeView.Path + '\';
  end;
end;

procedure TOptionsForm.SpeedButton2Click(Sender: TObject);
begin
  SelDirFrm.ShowModal;
  if SelDirFrm.ModalResult = mrOk then
  begin
    MODULESPATH.Text := SelDirFrm.ShellTreeView.Path + '\';
  end;
end;

procedure TOptionsForm.optTabMainShow(Sender: TObject);
begin
  AddBTN.Enabled := False;
  DelBTN.Enabled := False;
  EditBTN.Enabled := False;
end;

end.
```

Файл AntiVir_Main.pas – основна програма

```
unit AntiVir_Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, Menus, ImgList, XPMan, avKernel,
avTypes, ShellAPI, ShlObj,
  AppEvnts, OneHist, langs, jpeg;

  const
  WM_NOTIFYTRAYICON = WM_USER + 1;
  WM_MINERESTORE = WM_USER    + $877;

  type
  TIconType = (itSmall, itLarge);

  type
  NotifyIconData_50 = record
   cbSize: DWORD;
   Wnd: HWND;
   AntiVir_ID: UINT;
   AntiVir_Flags: UINT;
   AntiVir_CallbackMessage: UINT;
   hIcon: HICON;
   szTip: array[0..MAXCHAR] of AnsiChar;
   dwState: DWORD;
   dwStateMask: DWORD;
   szInfo: array[0..MAXBYTE] of AnsiChar;
   AntiVir_Timeout: UINT; // union with AntiVir_Version: UINT;
   szInfoTitle: array[0..63] of AnsiChar;
   dwInfoFlags: DWORD;
  end;

  const
  NIF_INFO      =        $00000010;
  NIIF_NONE     =        $00000000;
  NIIF_INFO     =        $00000001;
  NIIF_WARNING  =        $00000002;
  NIIF_ERROR    =        $00000003;

  type
  TBalloonTimeout = 10..30;
  TBalloonIconType = (bitNone,
                      bitInfo,
                      bitWarning,
                      bitError);

type
  TMainForm = class(TForm)
    MainPages: TPageControl;
    ScanPathesTab: TTabSheet;
    ScanningTab: TTabSheet;
    ReportTab: TTabSheet;
    BottomPanel: TPanel;
    ScanBTN: TButton;
    SaveBTN: TButton;
    PathList: TListView;
    Bevel1: TBevel;
    ScanList: TListView;
    ReportMemo: TMemo;
    ImageList: TImageList;
    DrivesImg: TImageList;
    PathMenu: TPopupMenu;
    AddFolder: TMenuItem;
    DeletePath: TMenuItem;
    N1: TMenuItem;
```

```
Reftesh: TMenuItem;
SaveDialog: TSaveDialog;
XPManifest: TXPManifest;
Bevel4: TBevel;
DelMenu: TPopupMenu;
Del: TMenuItem;
TrayMenu: TPopupMenu;
mnuShowAntiVirScanner: TMenuItem;
mnuHideAntiVirScanner: TMenuItem;
N2: TMenuItem;
mnuOptions: TMenuItem;
N4: TMenuItem;
mnuHelp: TMenuItem;
mnuAbout: TMenuItem;
N7: TMenuItem;
mnuExit: TMenuItem;
Image1: TImage;
TopPn: TPanel;
Bevel3: TBevel;
Image2: TImage;
RightPanel: TPanel;
ExitBTN: TButton;
TopRightPanel: TPanel;
Image3: TImage;
VersionLabel: TLabel;
AboutBTN: TLabel;
DelAll: TMenuItem;
ApplicationEvents: TApplicationEvents;
ProgressBar: TProgressBar;
ScanTopBtn: TLabel;
ScanMenu: TPopupMenu;
mnuSelScanPath: TMenuItem;
mnuShowReport: TMenuItem;
N12: TMenuItem;
OptionTopBtn: TLabel;
PCTopBtn: TLabel;
mnuAntiVirProcessControl: TMenuItem;
N19: TMenuItem;
mnuPCShow: TMenuItem;
N21: TMenuItem;
mnuPCRun: TMenuItem;
mnuPCPause: TMenuItem;
mnuPCStop: TMenuItem;
mnuScanStart: TMenuItem;
mnuStopScan: TMenuItem;
N13: TMenuItem;
mnuSaveReport: TMenuItem;
N26: TMenuItem;
mnuGoToTray: TMenuItem;
SOURCESTRING: TListBox;
LabelPanel: TPanel;
ScanFile: TLabel;
procedure DelAllClick(Sender: TObject);
procedure FormResize(Sender: TObject);
procedure ExitBTNClick(Sender: TObject);
procedure ScanListDblClick(Sender: TObject);
procedure ScanBTNClick(Sender: TObject);
procedure InitScannerKernel;
Procedure StartScan(Parametr: String);
procedure SaveBTNClick(Sender: TObject);
procedure DeletePathClick(Sender: TObject);
procedure RefteshClick(Sender: TObject);
procedure AddFolderClick(Sender: TObject);
function CreateDrivesList(ListView: TListView): boolean;
procedure AboutBTNClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure HelpBTNClick(Sender: TObject);
procedure DelMenuPopup(Sender: TObject);
```

```
    procedure DelClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormDestroy(Sender: TObject);
    procedure FormHide(Sender: TObject);
    procedure mnuHideAntiVirScannerClick(Sender: TObject);
    procedure mnuShowAntiVirScannerClick(Sender: TObject);
    procedure mnuExitClick(Sender: TObject);
    procedure mnuOptionsClick(Sender: TObject);
    procedure mnuHelpClick(Sender: TObject);
    procedure mnuAboutClick(Sender: TObject);
    procedure ApplicationEventsMinimize(Sender: TObject);
    procedure AppMinimize(Sender: TObject);
    procedure FormPaint(Sender: TObject);
    procedure ScanListCustomDrawItem(Sender: TCustomListView;
      Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
    function BalloonTrayIcon(const Window: HWND; const IconID: Byte; const
Timeout: TBalloonTimeout; const BalloonText, BalloonTitle: String; const
BalloonIconType: TBalloonIconType): Boolean;
    procedure ScanTopBtnClick(Sender: TObject);
    procedure mnuShowReportClick(Sender: TObject);
    procedure mnuSelScanPathClick(Sender: TObject);
    procedure PCTopBtnClick(Sender: TObject);
    procedure OptionTopBtnClick(Sender: TObject);
    procedure mnuGoToTrayClick(Sender: TObject);
    procedure mnuPCShowClick(Sender: TObject);
    procedure mnuPCRunClick(Sender: TObject);
    procedure mnuPCPauseClick(Sender: TObject);
    procedure mnuPCStopClick(Sender: TObject);
    procedure TrayMenuPopup(Sender: TObject);
    procedure ScanMenuPopup(Sender: TObject);
    procedure mnuScanStartClick(Sender: TObject);
    procedure mnuStopScanClick(Sender: TObject);
    procedure mnuSaveReportClick(Sender: TObject);
    procedure CopyRightLabelClick(Sender: TObject);
    Procedure CreateTray;
  protected
    procedure MineRestore(var Msg: TMessage); message WM_MINERESTORE;
    procedure SendScanning(var Msg: TMessage); message WM_COPYDATA;
  private
    Procedure WMSysCommand(var message: TWMSysCommand); message WM_SysCommand;
    procedure WMTRAYICONNOTIFY(var Msg: TMessage); message WM_NOTIFYTRAYICON;
    { Private declarations }
  public
    FileCN       : Integer;
    FileInfected  : Integer;
    FileIgnored   : Integer;
    FileDVC       : integer;

    MonFileCN       : Integer;
    MonFileInfected  : Integer;

    Path          : TStringList;
    DeActiveTray  : Boolean;

//***************************************************************************//

        AntiVirMonitor      : String;
        AntiVirInit         : String;
        LoadAPI          : String;
        LoadDB           : String;
        CreateDrvList    : String;
        OptFileNotFnd    : String;
        LoadOptFile      : String;
        InitProcedures   : String;
        initShield       : String;
        ErrorInit        : String;
        LogBevel         : String;
        DBKnowledge      : String;
        SCNOBJ           : String;
```

```
        ScanExecute     : String;
        ScanEnd         : String;
        PrepareToScan   : String;
        FileIgnor       : String;
        FileIfect       : String;
        FileScanned     : String;
        DataScanned     : String;
        IGNORED         : String;
        SKIPBYSIZE      : String;
        INFECTED        : String;
        STOPB           : String;
        RETURNB         : String;
        SCANB           : String;
        SCNFILE         : String;
        FileDel         : String;
        FileNotDel      : String;
        PATHNOSEL       : String;
        SysMenu         : String;
        NfoAntiVirScanner   : String;
        NfoAntiVirKernel    : String;
        NfoAntiVirBuild     : String;
        DelDialog       : String;
        DelAllDialog    : String;
        DelError        : String;
        HelpNOFound     : String;
        avShieldMes     : String;
        avError         : String;
        DelResult       : String;
        AllInfected     : String;
        DeleteInfected  : String;
        SkippedInfected : String;
        AntiVirCloseDlg     : String;
        AllreadyInScan  : String;
        ProcControlSt   : String;
        ErrorKillProc   : String;
        PCActive        : String;
        PCPaused        : String;
        PCStoped        : String;
        PCInit          : String;
        PCPause         : String;
        PCStop          : String;
        PCRestore       : String;
        LASTDBDATA      : String;
        DATABASEdate    : String;
        BASELOADED      : String;
        DBerrorI1       : String;
        DBerrorI2       : String;
        DBerrorI3       : String;

        MLoad           : String;
        MunLoad         : String;

  end;

 //***************************************************************************//
 // Створення головної форми
 resourcestring
   Return          = #13#10;
   AntiVirScannerCapt  = 'Антивірусний захист операційної системи';
   AntiVirScannerVS    = '';
var
   MainForm      : TMainForm;
   inScan        : Boolean = False;
   NeedToReturn  : Boolean = False;
   FirstRun      : Boolean = True;
   P             : TPoint;
   MayClose      : boolean=false;
implementation
```

```
uses AntiVir_SelInfo, AntiVir_Options, AntiVir_AddPath, AboutFrm, Math,
AntiVir_Message, AntiVir_HideForm,
  AntiVir_Monitor, AntiVir_InfectedAction, AntiVir_PluginInfo;
{$R *.dfm}

//***************************************************************************//

Procedure TMainForm.WMSysCommand(var message: TWMSysCommand);
begin
  If message.CmdType = SC_MINIMIZE then mnuHideAntiVirScanner.Click
  Else Inherited;
End;

//***************************************************************************//

procedure TMainForm.SendScanning;
var
  pcd: PCopyDataStruct;
begin
  pcd := PCopyDataStruct(Msg.LParam);
  if not inScan then
  begin
    StartScan(PChar(pcd.lpData));
  end
  else begin
    MessageDlg(AllreadyInScan,mtError,[mbOK],0);
  end;
end;

procedure TMainForm.MineRestore(var Msg: TMessage);
begin
  if (Msg.Msg = WM_MINERESTORE) then
  begin
    mnuShowAntiVirScanner.Click;
  end;
end;

//***************************************************************************//

function TMainForm.BalloonTrayIcon(const Window: HWND; const IconID: Byte; const
Timeout: TBalloonTimeout; const BalloonText, BalloonTitle: String; const
BalloonIconType: TBalloonIconType): Boolean;
  const
  aBalloonIconTypes : array[TBalloonIconType] of
                      Byte = (NIIF_NONE, NIIF_INFO, NIIF_WARNING, NIIF_ERROR);
  var
  NID_50 : NotifyIconData_50;
begin
  if Not OptionsForm.SHOWBALOONHINT.Checked then Exit;
  FillChar(NID_50, SizeOf(NotifyIconData_50), 0);
  with NID_50 do begin
    cbSize := SizeOf(NotifyIconData_50);
    Wnd := Window;
    AntiVir_ID := IconID;
    AntiVir_Flags := NIF_INFO;
    StrPCopy(szInfo, BalloonText);
    AntiVir_Timeout := Timeout * 1000;
    StrPCopy(szInfoTitle, BalloonTitle);
    dwInfoFlags := aBalloonIconTypes[BalloonIconType];
  end;
  Result := Shell_NotifyIcon(NIM_MODIFY, @NID_50);
end;

procedure TMainForm.WMTRAYICONNOTIFY(var Msg: TMessage);
begin
  case Msg.LParam of
    WM_LBUTTONUP:
      begin
      if Not DeActiveTray then
```

```
        begin
          MayClose := False;
          GetCursorPos(p);
          MayClose:= false;
          DeActiveTray := False;
          showwindow(Application.handle, SW_SHOW);
          showwindow(MainForm.handle, SW_SHOW);
          Application.Restore;
        end
        else
        begin
          SetForegroundWindow(HideForm.Handle);
        end;
      end;
    WM_RBUTTONUP:
      begin
        if Not DeActiveTray then
        begin
          GetCursorPos(p);
          TrayMenu.Popup(P.X, P.Y);
        end;
      end;
  end;
end;

Procedure TMainForm.CreateTray;
  var
  tray: TNotifyIconData;
begin
  with tray do
  begin
    cbSize := SizeOf(TNotifyIconData);
    Wnd := MainForm.Handle;
    AntiVir_ID := 1;
    AntiVir_Flags := NIF_ICON or NIF_MESSAGE or NIF_TIP;
    AntiVir_CallBackMessage := WM_NOTIFYTRAYICON;
    hIcon :=  Application.Icon.Handle;
    szTip := 'AntiVir Scanner';
  end;
  Shell_NotifyIcon(NIM_ADD, Addr(tray));
end;

Procedure DestroyTray;
  var
  tray: TNotifyIconData;
begin
  with tray do
  begin
    cbSize := SizeOf(TNotifyIconData);
    Wnd := MainForm.Handle;
    AntiVir_ID := 1;
  end;
  Shell_NotifyIcon(NIM_DELETE, Addr(tray));
end;

//*Функція визначення шляху**********************************//

Function GetShortPathBC(lPath:string): string;
  var
  D,F,P: String;
  i    : integer;
begin
  D := lPath[1]+':\';
  F := ExtractFileName(lPath);
  ShowMessage(D+'..'+F);
end;

Function GETParam(Str: String): String;
  var
```

```
  TMP,Str1,Str2 : String;
  PS: integer;
begin
  Result := '';
  TMP := STR;
  if TMP <> '' then
  if pos('=',TMP) <> 0 then
  begin
    ps := pos('=',TMP);
    Str1 := Copy(TMP,0,ps-1);
    Str2 := Copy(TMP,ps+1,length(Tmp));
    Result := Str2;
  end;
end;

Function GETParamName(Str: String): String;
  var
  TMP,Str1,Str2 : String;
  PS: integer;
begin
  Result := '';
  TMP := STR;
  if TMP <> '' then
  if pos('=',TMP) <> 0 then
  begin
    ps := pos('=',TMP);
    Str1 := Copy(TMP,0,ps-1);
    Str2 := Copy(TMP,ps+1,length(Tmp));
    Result := Str1;
  end;
end;

//***Функція завантаження опцій ***//

Procedure LoadOptions;
  var
  i: integer;
begin
  LoadConfig_;
  OptionsForm.ModulesLOAD.Checked     := OPT_MODULES_LOAD;
  OptionsForm.DBPATH.Text             := OPT_DB_DIR;
  OptionsForm.MODULESPATH.Text        := OPT_MODULE_DIR;
  OptionsForm.USESHIELD.Checked       := OPT_USE_SHIELD;
  OptionsForm.SHIELDSILENT.Checked    := OPT_SILENT_SHIELD_MODE;
  OptionsForm.SCNSUBDIR.Checked       := OPT_SCAN_SUBDIR;
  OptionsForm.SCNHEX.Checked          := OPT_USE_HEX_MODE;
  OptionsForm.SCNCRC.Checked          := OPT_USE_CRC_MODE;
  OptionsForm.SCNBIT.Checked          := OPT_USE_BYTE_MODE;

  OptionsForm.SCNHEXINPOS.Checked     := OPT_USE_HEX_INPOS;
  OptionsForm.DisplayScnFiles.Checked := OPT_SEND_SCAN_FILE;

  OptionsForm.PathList.Clear;
  OptionsForm.ExtList.Clear;
  for i := 0 to AntiVirConfig.Count-1 do begin

    if GETParamName(AntiVirConfig[i]) = 'EXT' then
      with OptionsForm.ExtList.Items.Add do begin
        Caption := GetParam(AntiVirConfig[i]);
        ImageIndex := 3;
      end;
      if GETParamName(AntiVirConfig[i]) = 'SHOWBALOONHINT' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then
OptionsForm.SHOWBALOONHINT.Checked := False else
      OptionsForm.SHOWBALOONHINT.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'PROCCONTROLAUTOMODE' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then OptionsForm.PCAutoLoad.Checked
:= False else
```

```
      OptionsForm.PCAutoLoad.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'PROCCONTROLAUTOKILL' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then OptionsForm.PCAutoKill.Checked
:= False else
      OptionsForm.PCAutoKill.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'PROCCONTROLAUTOACTION' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then
OptionsForm.PCAutoAction.Checked := False else
      OptionsForm.PCAutoAction.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'PROCCONTROLDELINFECT' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then OptionsForm.PCDelInfect.Checked
:= False else
      OptionsForm.PCDelInfect.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'PROCCONTROLSKIPINFECT' then
      if GetParam(AntiVirConfig[i]) = 'OFF' then
OptionsForm.PCSkipInfect.Checked := False else
      OptionsForm.PCSkipInfect.Checked := True;

      if GETParamName(AntiVirConfig[i]) = 'HIDETIP' then begin
      if GetParam(AntiVirConfig[i]) = 'OFF' then HideForm.ShowHideTip.Checked :=
False else
      HideForm.ShowHideTip.Checked := True;
      end;

      if GETParamName(AntiVirConfig[i]) = 'PATH' then begin
      with OptionsForm.PathList.Items.Add do begin
        Caption := GetParam(AntiVirConfig[i]);
        if DirectoryExists(Caption) then ImageIndex := 4 else ImageIndex := 5;
      end;
      end;

      if GETParamName(AntiVirConfig[i]) = 'AUTOSAVEREPORT' then
      if GetParam(AntiVirConfig[i]) = 'ON' then
OptionsForm.AutoSaveReport.Checked := true else
      OptionsForm.AutoSaveReport.Checked := False;

      if GETParamName(AntiVirConfig[i]) = 'REGISTERSYSMENU' then
      if GetParam(AntiVirConfig[i]) = 'ON' then
OptionsForm.RegisterSysMenu.Checked := true else
      OptionsForm.RegisterSysMenu.Checked := False;

      if GETParamName(AntiVirConfig[i]) = 'AUTORUN' then
      if GetParam(AntiVirConfig[i]) = 'ON' then OptionsForm.AUTORUN.Checked :=
true else
      OptionsForm.AUTORUN.Checked := False;

      if GETParamName(AntiVirConfig[i]) = 'AUTOHIDE' then
      if GetParam(AntiVirConfig[i]) = 'ON' then OptionsForm.AUTOHIDE.Checked :=
true else
      OptionsForm.AUTOHIDE.Checked := False;

      if GETParamName(AntiVirConfig[i]) = 'AUTOSAVEREPORTTO' then
OptionsForm.ReportSavePath.Text := GETParam(AntiVirConfig[i]);
  end;
end;

function GetHDDSerial(ADisk : char): dword;
  var
  SerialNum : dword;
  a, b : dword;
  VolumeName : array [0..255] of char;
begin
  Result := 0;
  if GetVolumeInformation(PChar(ADisk + ':\'), VolumeName, SizeOf(VolumeName),
  @SerialNum, a, b, nil, 0) then
```

```
    Result := SerialNum;
end;

function TMainForm.CreateDrivesList(ListView: TListView): boolean;
  var
  Bufer : array[0..1024] of char;
  RealLen, i : integer;
  S : string;
begin
  ListView.Clear;
  RealLen := GetLogicalDriveStrings(SizeOf(Bufer),Bufer);
  i := 0; S := '';
  while i < RealLen do begin
    if Bufer[i] <> #0 then begin
      S := S + Bufer[i];
      inc(i);
    end else begin
      inc(i);
      with ListView.Items.Add do begin
        Caption := S;
        if GetDriveType(PChar(S)) = DRIVE_RAMDISK then ImageIndex := 3;
        if GetDriveType(PChar(S)) = DRIVE_FIXED then ImageIndex := 3;
        if GetDriveType(PChar(S)) = DRIVE_REMOTE then ImageIndex := 0;
        if GetDriveType(PChar(S)) = DRIVE_CDROM then ImageIndex := 1;
        if GetDriveType(PChar(S)) = DRIVE_REMOVABLE then ImageIndex := 2;
      end;
      S := '';
    end;
  end;

  For i := 0 to OptionsForm.PathList.Items.Count-1  do begin
    with ListView.Items.Add do begin
      Caption := OptionsForm.PathList.Items[i].Caption;
      ImageIndex := OptionsForm.PathList.Items.Item[i].ImageIndex;
    end;
  end;
  Result := ListView.items.Count > 0;
end;

procedure OnAddToLogStr(LogString: String; ID: integer);
  var
  TMP : String;
begin
  with MainForm.ScanList.Items.Add do begin
    if ID = -1 then
      Caption := LogString
    else begin
      Caption := FormatDateTime('[hh:mm:ss]',now) + ' ' + LogString;
      MainForm.ReportMemo.Lines.Add(Caption);
      if ID = 2 then begin
        TMP := LogString;
        system.Delete(Tmp,1,pos(']',Tmp)+1);
        SubItems.Add(TMP);
      end;
      ImageIndex := ID;
    end;
    ImageIndex := ID;
  end;
  SendMessage(MainForm.ScanList.Handle, WM_VSCROLL, SB_BOTTOM, 0);
end;

procedure AddToMonLogStr(LogString: String; ID: integer);
  var
  TMP : String;
begin
{ }
end;
```

```
//***Функція вибору параметрів сканування на віруси***//

procedure OnScanComplete;
  var
  ScanEndBalloonText: String;
  i: integer;
begin
  MainForm.ProgressBar.Max := 1;
  MainForm.ProgressBar.Position := MainForm.ProgressBar.Max;
  MainForm.ScanBTN.Caption := MainForm.RETURNB;
  NeedToReturn := True;
  inScan := False;
  MainForm.Path.Clear;

  for i := 0 to MainForm.PathList.Items.Count-1 do
    MainForm.PathList.Items.Item[i].Checked := false;

  MessageBeep(MB_ICONASTERISK);
  MainForm.SaveBTN.Enabled := true;
  MainForm.ScanFile.caption  := MainForm.ScanEnd;
  OnAddToLogStr(‘’,-1);
  OnAddToLogStr(MainForm.ScanEnd,0);
  OnAddToLogStr(‘’,-1);
  OnAddToLogStr(MainForm.FileScanned+inttostr(MainForm.FileCN),0);
  OnAddToLogStr(MainForm.FileIgnor+inttostr(MainForm.FileIgnored),0);
  OnAddToLogStr(MainForm.FileIfect+inttostr(MainForm.FileInfected),0);
  OnAddToLogStr(MainForm.DataScanned+Format(‘%.2f’,[ScannedDataSize / 1024 /
1024])+’ Mb’,0);
  MainForm.ReportMemo.Lines.Add(MainForm.LogBevel);
  if OptionsForm.AutoSaveReport.Checked then begin
    MainForm.ReportMemo.Lines.SaveToFile(OptionsForm.ReportSavePath.Text);
  end;

  ScanEndBalloonText := MainForm.ScanEnd + ‘:’ + Return + Return
                        +’ >> ‘+MainForm.FileScanned+inttostr(MainForm.FileCN) +
Return
                        +’ >> ‘+MainForm.FileIgnor+inttostr(MainForm.FileIgnored)
+ Return
                        +’ >>
‘+MainForm.FileIfect+inttostr(MainForm.FileInfected) + Return
                        +’ >>
‘+MainForm.DataScanned+Format(‘%.2f’,[ScannedDataSize / 1024 / 1024])+’ Mb’;

  MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,ScanEndBalloonText,’AntiVir
Scanner’,bitInfo);
end;

//***Функція початку сканування***//

Procedure OnScanStart;
  var
  i: integer;
begin
  MainForm.FileDVC := 0;
  MainForm.ProgressBar.Position := 0;
  MainForm.ProgressBar.Max := 0;

  ClearExtList;
  for i := 0 to OptionsForm.ExtList.Items.Count-1 do begin
    AddToExtList(ExtractFileExt(OptionsForm.ExtList.Items.Item[i].Caption));
  end;

  MainForm.ScanBTN.Caption := MainForm.STOPB;
  MainForm.SaveBTN.Enabled := False;
  MainForm.ScanList.Clear;
  MainForm.ScanningTab.Show;
  MainForm.FileCN       := 0;
  MainForm.FileInfected := 0;
  MainForm.FileIgnored  := 0;
```

```
  inScan := True;
  NeedToReturn := False;
  OnAddToLogStr(MainForm.ScanExecute,0);
  if AntiVirScanner.AvAction = TScanDir then
  else
  OnAddToLogStr(MainForm.SCNOBJ+AntiVirScanner.FileName,0);
  OnAddToLogStr('',-1);
  MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,MainForm.ScanExecute,'AntiVir
Scanner',bitInfo);
  AntiVirScanner.Resume;
end;

//***Функція підключення ядра антивіруса***//

Procedure AntiVirKernelMessageAPI(MES: Integer; const Pr_0: Integer = 0; Pr_1:
String = ''; Pr_2: String = '');
begin

  if MES = MES_NONE then Exit;

  if mes = MES_LOCKINPUT then
  begin
    MainForm.ProgressBar.Enabled := False;
    MainForm.ScanBTN.Enabled := False;
  end;

  if mes = MES_UNLOCKINPUT then
  begin
    MainForm.ProgressBar.Position := 0;
    MainForm.ProgressBar.Enabled := True;
    MainForm.ScanBTN.Enabled := True;
  end;

  if MES = MES_SCANMAXPROGRESS then begin
    MainForm.FileDVC := mainForm.FileCN;
    MainForm.ProgressBar.Max := Pr_0-MainForm.FileDVC;
  end;

  if MES = MES_PREPARINGTOSCAN then MainForm.ScanFile.Caption :=
MainForm.PrepareToScan;

  if mes = MES_INITKERNEL then OnAddToLogStr(MainForm.AntiVirInit,0);

  if mes = MES_INITAPI then OnAddToLogStr(MainForm.LoadAPI,0);

  if mes = MES_LOADBASES then OnAddToLogStr(MainForm.LoadDB,0);

  if mes = MES_LOADCONFIG then OnAddToLogStr(MainForm.LoadOptFile,0);

  if mes = MES_INITSHIELD then OnAddToLogStr(MainForm.initShield,0);

  if mes = MES_ERRORONINIT then OnAddToLogStr(MainForm.ErrorInit,2);

  if MES = MES_LOADDBDATE then begin
    MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.BASELOADED+ ExtractFileName(Pr_1)+'
('+MainForm.DATABASEdate+_ConvertDate(Pr_2)+')');
  end;

  if MES = MES_ERROR then begin
    MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.avError);
  end;

  if MES = MES_ONSCANEXECUTE then
    OnScanStart;

  if MES = MES_ONSCANCOMPLETE then
    OnScanComplete;
```

```
  if MES = MES_ONPROGRESS then begin
    if MainForm.ProgressBar.Enabled then begin
      MainForm.FileCN := MainForm.FileCN + 1;
      if MainForm.ProgressBar.Max > 0 then
        MainForm.ProgressBar.Position := MainForm.FileCN-MainForm.FileDVC;
      MainForm.ScanFile.caption  := '['+inttostr(MainForm.FileCN)+']
'+ExtractFileName(Pr_1);
    end
    else
      MainForm.ScanFile.caption  := ExtractFileName(Pr_1);
    if OPT_SEND_SCAN_FILE then
MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+MainForm.SCNFILE
+ Pr_1);
  end;

  if MES = MES_ONVIRFOUND then begin
    OnAddToLogStr('['+MainForm.INFECTED+' - '+Pr_2+'] '+Pr_1,2);
    MainForm.FileInfected := MainForm.FileInfected + 1;
    MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,Pr_1 ,'['+MainForm.INFECTED+'
- '+Pr_2+'] ',bitError);
  end;

  if MES = MES_ONREADERROR then begin
    OnAddToLogStr('['+MainForm.IGNORED+'] '+Pr_1,1);
    MainForm.FileIgnored  := MainForm.FileIgnored + 1;
  end;

  if MES = MES_SKIPBYSIZE then begin
    OnAddToLogStr('['+MainForm.SKIPBYSIZE+'] '+Pr_1,1);
    MainForm.FileIgnored  := MainForm.FileIgnored + 1;
  end;

  if MES = MES_ADDTOLOG then begin
    OnAddToLogStr(Pr_1,Pr_0);
  end;

  if MES = MES_SHIELD_INFECT then begin
    MessageFrm.Caption := 'avShield Messsage';
    MessageFrm.InformationLabel.Caption := 'avShield Messsage';
    MessageFrm.InfoLabel.Caption := 'Warning!';
    MessageFrm.Memo1.Text := MainForm.avShieldMes;
  end;

end;

//***Функція ініціалізація ядра антивірусa***//

procedure TMainForm.InitScannerKernel;
  var
  i:integer;
begin

 //******//
        AntiVirMonitor      := SOURCESTRING.Items[0];
        AntiVirInit         := SOURCESTRING.Items[1];
        LoadAPI          := SOURCESTRING.Items[2];
        LoadDB           := SOURCESTRING.Items[3];
        CreateDrvList    := SOURCESTRING.Items[4];
        OptFileNotFnd    := SOURCESTRING.Items[5];
        LoadOptFile      := SOURCESTRING.Items[6];
        InitProcedures   := SOURCESTRING.Items[7];
        initShield       := SOURCESTRING.Items[8];
        ErrorInit        := SOURCESTRING.Items[9];
        LogBevel         := SOURCESTRING.Items[10];
        DBKnowledge      := SOURCESTRING.Items[11];
        SCNOBJ           := SOURCESTRING.Items[12];
        ScanExecute      := SOURCESTRING.Items[13];
        ScanEnd          := SOURCESTRING.Items[14];
```

```
        PrepareToScan   := SOURCESTRING.Items[15];
        FileIgnor       := SOURCESTRING.Items[16];
        FileIfect       := SOURCESTRING.Items[17];
        FileScanned     := SOURCESTRING.Items[18];
        DataScanned     := SOURCESTRING.Items[19];
        IGNORED         := SOURCESTRING.Items[20];
        SKIPBYSIZE      := SOURCESTRING.Items[21];
        INFECTED        := SOURCESTRING.Items[22];
        STOPB           := SOURCESTRING.Items[23];
        RETURNB         := SOURCESTRING.Items[24];
        SCANB           := SOURCESTRING.Items[25];
        SCNFILE         := SOURCESTRING.Items[26];
        FileDel         := SOURCESTRING.Items[27];
        FileNotDel      := SOURCESTRING.Items[28];
        PATHNOSEL       := SOURCESTRING.Items[29];
        SysMenu         := SOURCESTRING.Items[30];
        NfoAntiVirScanner   := SOURCESTRING.Items[31];
        NfoAntiVirKernel    := SOURCESTRING.Items[32];
        NfoAntiVirBuild     := SOURCESTRING.Items[33];
        DelDialog       := SOURCESTRING.Items[34];
        DelAllDialog    := SOURCESTRING.Items[35];
        DelError        := SOURCESTRING.Items[36];
        HelpNOFound     := SOURCESTRING.Items[37];
        avShieldMes     := SOURCESTRING.Items[38];
        avError         := SOURCESTRING.Items[39];
        DelResult               := SOURCESTRING.Items[40];
        AllInfected     := SOURCESTRING.Items[41];
        DeleteInfected  := SOURCESTRING.Items[42];
        SkippedInfected := SOURCESTRING.Items[43];
        AntiVirCloseDlg     := SOURCESTRING.Items[44];
        AllreadyInScan  := SOURCESTRING.Items[45];
        ProcControlSt   := SOURCESTRING.Items[46];
        ErrorKillProc   := SOURCESTRING.Items[47];
        PCActive        := SOURCESTRING.Items[48];
        PCPaused        := SOURCESTRING.Items[49];
        PCStoped        := SOURCESTRING.Items[50];
        PCInit          := SOURCESTRING.Items[51];
        PCPause         := SOURCESTRING.Items[52];
        PCStop          := SOURCESTRING.Items[53];
        PCRestore       := SOURCESTRING.Items[54];
        LASTDBDATA      := SOURCESTRING.Items[55];
        DATABASEdate    := SOURCESTRING.Items[56];
        BASELOADED      := SOURCESTRING.Items[57];
        DBerrorI1       := SOURCESTRING.Items[58];
        DBerrorI2       := SOURCESTRING.Items[59];
        DBerrorI3       := SOURCESTRING.Items[60];

        MLoad           := SOURCESTRING.Items[61];
        MunLoad         := SOURCESTRING.Items[62];

        InitKernel(AntiVirKernelMessageAPI);
        LoadOptions;

  //***Функція створення списку дисків***//

  CreateDrivesList(PathList);

  for i := 0 to GetPluginAPICount do
    with OptionsForm.APIList.Items.Add do
      begin
        Caption := GetPluginAPIName(i) + ‘
(‘+ExtractFileName(GetPluginAPIPath(i))+’)’;
        SubItems.Add(GetPluginAPIAutor(i));
        SubItems.Add(GetPluginAPIInfo(i));
        SubItems.Add(GetPluginAPIPath(i));
      end;

  ReportMemo.Lines.Add(‘’);
```

```
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoAntiVirScanner
+AntiVirScannerVS);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoAntiVirKernel
+GetKernelVersion);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoAntiVirBuild
+GetKernelBuild);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+DBKnowledge+IntToStr(GetDBRecCount));

  if GetDBVersionDate = '01.01.1980' then
    ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+LASTDBDATA+'0')
  else
    ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+LASTDBDATA+GetDBVersionDate);

  ReportMemo.Lines.Add(LogBevel);
  ReportMemo.Lines.Add('');

  if OptionsForm.RegisterSysMenu.Checked then begin
    OptionsForm.FileTAddAction('*','AntiVir.Scan',SysMenu,ParamStr(0)+' %1');
    OptionsForm.FileTAddAction('Directory','AntiVir.Scan',SysMenu,ParamStr(0)+'
%1');
    OptionsForm.FileTAddAction('Drive','AntiVir.Scan',SysMenu,ParamStr(0)+'
%1');
  end else
  begin
    OptionsForm.FileTDelAction('Drive','AntiVir.Scan');
    OptionsForm.FileTDelAction('Directory','AntiVir.Scan');
    OptionsForm.FileTDelAction('*','AntiVir.Scan');
  end;
end;

//***Функція початку сканування***//

Procedure TMainForm.StartScan(Parametr: String);
  var
  T : String;
begin
  if GetDBRecCount = 0 then
  begin
    MessageFrm.Caption := DBerrorI1;
    MessageFrm.InformationLabel.Caption := DBerrorI1;
    MessageFrm.InfoLabel.Caption := DBerrorI2;
    MessageFrm.Memo1.Text := DBerrorI3;
    MessageFrm.ShowModal;
    Exit;
  end;

  if Parametr = 'DRV' then
  begin
    AntiVirScanner := TAvScanner.Create(true);
    AntiVirScanner.NeedForAPI := TRUE;
    AntiVirScanner.AvAction   := TScanDir;
    Path.Add(ExtractFileDrive(Paramstr(0))+'\');
    AntiVirScanner.Dirs       := Path;
    OnScanStart;
    exit;
  end;

  if DirectoryExists(Parametr+'\') then
  begin
    AntiVirScanner := TAvScanner.Create(true);
    AntiVirScanner.NeedForAPI := TRUE;
    AntiVirScanner.AvAction   := TScanDir;
    Path.Add(Parametr+'\');
    AntiVirScanner.Dirs       := Path;
    OnScanStart;
    exit;
  end;
```

```
  if FileExists(Parametr) then
  begin
    AntiVirScanner := TAvScanner.Create(true);
    AntiVirScanner.NeedForAPI := false;
    AntiVirScanner.AvAction := TScanFile;
    AntiVirScanner.FileName := Parametr;
    OnScanStart;
    exit;
  end;

end;

procedure TMainForm.ExitBTNClick(Sender: TObject);
begin
  Close;
end;

procedure TMainForm.ScanListDblClick(Sender: TObject);
begin
  if ScanList.ItemIndex <> -1 then
  begin
    InformationForm.InfoMemo.Text := ScanList.Selected.Caption;
    InformationForm.ShowModal;
  end;
end;

procedure TMainForm.ScanBTNClick(Sender: TObject);
  var
  i: integer;
  err: boolean;
begin
  err:= false;

  for i := 0 to PathList.Items.Count-1 do
  begin
    if PathList.Items.Item[i].Checked then
    begin
      Path.Add(PathList.Items.Item[i].Caption);
      if not DirectoryExists(PathList.Items.Item[i].Caption+'\') then
        begin
          MessageDlg(PATHNOSEL,mtError,[mbOk],0);
          Exit;
        end;
    end;
  end;

 { if GetDBRecCount = 0 then
  begin
    MessageFrm.Caption := DBerrorI1;
    MessageFrm.InformationLabel.Caption := DBerrorI1;
    MessageFrm.InfoLabel.Caption        := DBerrorI2;
    MessageFrm.Memo1.Text               := DBerrorI3;
    MessageFrm.ShowModal;
    Exit;
  end;     }

  if NeedToReturn = false then
  begin
    if inScan = False then
    begin
      if PATH.Count-1 <> -1 then
      begin
        AntiVirScanner := TAvScanner.Create(true);
        AntiVirScanner.FreeOnTerminate := True;
        AntiVirScanner.NeedForAPI      := true;
        AntiVirScanner.AvAction        := TScanDir;
        AntiVirScanner.Dirs            := MainForm.Path;
        OnScanStart;
```

```
      end
      else begin
        MessageDlg(PATHNOSEL,mtError,[mbOk],0);
      end;
    end
    else begin
      CloseScanThread;
    end;
  end else
  begin
    ScanBTN.Caption := ScanB;
    MainForm.SaveBTN.Enabled := False;
    NeedToReturn := False;
    ScanPathesTab.Show;
  end;
end;

procedure TMainForm.SaveBTNClick(Sender: TObject);
  var
  Report: TStringList;
  i: integer;
begin
  if SaveDialog.Execute then
  begin
    Report:= TStringList.Create;
    For i := 0 to ScanList.Items.Count-1 do
      Report.Add(ScanList.Items.Item[i].Caption);
    Report.SaveToFile(SaveDialog.FileName);
    Report.Free;
  end;
end;

procedure TMainForm.DeletePathClick(Sender: TObject);
begin
  try
    if PathList.ItemIndex <> -1 then
      if PathList.Selected.ImageIndex > 3 then
      begin
        OptionsForm.PathList.Items.Delete(PathList.Selected.Index-
((PathList.Items.Count-1) - (OptionsForm.PathList.items.count-1)));
        PathList.Items.Delete(PathList.Selected.Index);
      end;
    OptionsForm.SaveOptions;
  except
  end;
end;

procedure TMainForm.RefteshClick(Sender: TObject);
begin
  CreateDrivesList(PathList);
end;

procedure TMainForm.AddFolderClick(Sender: TObject);
begin
  AddUserPathForm.ShowModal;
end;

procedure TMainForm.AboutBTNClick(Sender: TObject);
begin
 DBKnowledge+IntToStr(GetDBRecCount);
 AboutForm.ShowModal;
end;

procedure TMainForm.FormShow(Sender: TObject);
begin
  VersionLabel.Caption := AntiVirScannerVS;
end;

procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction);
```

```
begin
  if MessageDlg(AntiVirCloseDlg,mtInformation,[mbYes]+[mbNo],0) = 6 then begin
    if OptionsForm.AutoSaveReport.Checked then begin
      MainForm.ReportMemo.Lines.SaveToFile(OptionsForm.ReportSavePath.Text);
    end;
  end else Action := caNone;
end;

procedure TMainForm.HelpBTNClick(Sender: TObject);
begin
  if FileExists(ExtractFilePath(paramstr(0))+'\Help.chm') then
    ShellExecute(0,'',PChar(ExtractFilePath(paramstr(0))+'\Help.chm'),nil,nil,1)
  else
    MessageDlg(HelpNOFound,mtError,[mbOk],0);
end;

procedure TMainForm.DelMenuPopup(Sender: TObject);
begin
  if (ScanList.ItemIndex <> -1) and (ScanList.Selected.ImageIndex = 2) and
(inScan = False) then
  begin
    Del.Visible := true;
  end
  else
    Del.Visible := False;

  if (ScanList.ItemIndex <> -1) and (inScan = False) then
    DelAll.Visible := true
  else
    DelAll.Visible := false;
end;

procedure TMainForm.DelAllClick(Sender: TObject);
  var
      i,d,e,c: integer;
begin
  d:=0;
  e:=0;
  c:=0;
  if MessageDlg(DelAllDialog,mtInformation,[mbCancel]+[mbYes],0) = 6 then
  begin
        for i := 0 to ScanList.Items.Count - 1 do
        if ScanList.Items.Item[i].ImageIndex = 2 then
      begin
            c:=c+1;
                  try
                        if DeleteFileBC(ScanList.Items.Item[i].SubItems[0]) then
          begin
           d:=d+1;
                        ScanList.Items.Item[i].ImageIndex := 4;

      ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileDel+ScanList.Ite
ms.Item[i].SubItems[0]);
          end
                        else begin

      ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileNotDel+ScanList.
Items.Item[i].SubItems[0]);
            e:=e+1;
                        end;
                  except
                  end;
   end;

  MessageDlg(DelResult + Return
                                  + Return
                              + AllInfected + IntToStr(c) + Return
                        + DeleteInfected + IntToStr(d) + Return
                        + SkippedInfected + IntToStr(e),mtInformation,[mbOK],0);
```

```
  end;
end;

procedure TMainForm.DelClick(Sender: TObject);
begin
  if MessageDlg(DelDialog,mtInformation,[mbCancel]+[mbYes],0) = 6 then
  begin
  try
    if DeleteFileBC(ScanList.Selected.SubItems[0]) then
    begin
      ScanList.Selected.ImageIndex := 4;

ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileDel+ScanList.Selected.
SubItems[0]);
    end
    else begin

ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileNotDel+ScanList.Select
ed.SubItems[0]);
      MessageDlg(DelError,mtWarning,[mbOk],0);
    end;
  except
  end;
  end;
end;

procedure TMainForm.FormCreate(Sender: TObject);
begin
  Path := TStringList.Create;
  TopPn.ControlStyle            := ControlStyle + [csOpaque];
  TopRightPanel.ControlStyle    := ControlStyle + [csOpaque];
  Caption                       := AntiVirScannerCapt;
  TopPn.DoubleBuffered                := true;
  TopRightPanel.DoubleBuffered     := true;
  PathList.DoubleBuffered           := true;
  ScanList.DoubleBuffered           := true;
  BottomPanel.DoubleBuffered   := true;
  MonFileCN                     := 0;
  MonFileInfected               := 0;
end;

procedure TMainForm.AppMinimize(Sender: TObject);
begin
  ShowWindow(Application.Handle, SW_HIDE);
end;


procedure TMainForm.FormDestroy(Sender: TObject);
begin
  DestroyTray;
end;

procedure TMainForm.FormHide(Sender: TObject);
begin
  showwindow(Application.handle, SW_HIDE);
  showwindow(MainForm.handle, SW_HIDE);
end;

procedure TMainForm.FormResize(Sender: TObject);
begin
  PathList.Columns.Items[0].Width := PathList.Width - 25;
  ScanList.Columns.Items[0].Width := ScanList.Width - 25;
end;

procedure TMainForm.mnuHideAntiVirScannerClick(Sender: TObject);
begin
  DeActiveTray := True;
  MayClose := True;
  showwindow(Application.handle, SW_HIDE);
```

```
  showwindow(MainForm.handle, SW_HIDE);
  if not HideForm.ShowHideTip.Checked then
  begin
    HideForm.Show;
    SetForegroundWindow(HideForm.Handle);
    Application.BringToFront;
  end else DeActiveTray := False;
end;

procedure TMainForm.mnuShowAntiVirScannerClick(Sender: TObject);
begin
  DeActiveTray := False;
  showwindow(Application.handle, SW_SHOW);
  showwindow(MainForm.handle, SW_SHOW);
  Application.Restore;
  MayClose := False;
end;

procedure TMainForm.mnuExitClick(Sender: TObject);
begin
  Close;
end;

procedure TMainForm.mnuOptionsClick(Sender: TObject);
begin
  if not inScan then begin
    LoadOptions;
    OptionsForm.Show;
  end;
end;

procedure TMainForm.mnuHelpClick(Sender: TObject);
begin
  if FileExists(ExtractFilePath(paramstr(0))+'\Help.chm') then
    ShellExecute(0,'',PChar(ExtractFilePath(paramstr(0))+'\Help.chm'),nil,nil,1)
  else
    MessageDlg(HelpNOFound,mtError,[mbOk],0);
end;

procedure TMainForm.mnuAboutClick(Sender: TObject);
begin
DBKnowledge+IntToStr(GetDBRecCount);
  if GetDBVersionDate = '01.01.1880' then

  try
    AboutForm.ShowModal;
  except
  end;
end;

procedure TMainForm.ApplicationEventsMinimize(Sender: TObject);
begin
  mnuHideAntiVirScanner.Click;
end;

procedure TMainForm.FormPaint(Sender: TObject);
begin
  if FirstRun then
    if OptionsForm.AUTOHIDE.Checked then
    begin
      mnuHideAntiVirScanner.Click;
    end;
  FirstRun := false;
end;

procedure TMainForm.ScanListCustomDrawItem(Sender: TCustomListView;
  Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
begin
  with ScanList.Canvas.Brush do
```

```
  begin
    case Item.ImageIndex of
      0: Color := $00FFF1EC;
      2: Color := $00ECECFF;
      1: Color := $00ECFBFF;
      4: Color := $00EDFFEC;
    end;
  end;
end;

procedure TMainForm.ScanTopBtnClick(Sender: TObject);
begin

ScanMenu.Popup(MainForm.Left+ScanTopBtn.Left+3,MainForm.Top+ScanTopBtn.Top+38);
end;

procedure TMainForm.mnuShowReportClick(Sender: TObject);
begin
  if not inScan then
    ReportTab.Show;
end;

procedure TMainForm.mnuSelScanPathClick(Sender: TObject);
begin
  if not inScan then
    ScanPathesTab.Show;
end;

procedure TMainForm.PCTopBtnClick(Sender: TObject);
begin
  MonitorForm.Show;
end;

procedure TMainForm.OptionTopBtnClick(Sender: TObject);
begin
  if not inScan then begin
    LoadOptions;
    OptionsForm.ShowModal;
  end;
end;

procedure TMainForm.mnuGoToTrayClick(Sender: TObject);
begin
  mnuHideAntiVirScanner.Click;
end;

procedure TMainForm.mnuPCShowClick(Sender: TObject);
begin
  MonitorForm.Show;
end;

procedure TMainForm.mnuPCRunClick(Sender: TObject);
begin
  MonitorForm.StartPC.Click;
end;

procedure TMainForm.mnuPCPauseClick(Sender: TObject);
begin
  MonitorForm.PausePC.Click;
end;

procedure TMainForm.mnuPCStopClick(Sender: TObject);
begin
  MonitorForm.StopPC.Click;
end;

procedure TMainForm.TrayMenuPopup(Sender: TObject);
begin
  mnuPCRun.Enabled := MonitorForm.StartPC.Enabled;
```

```
  mnuPCPause.Enabled := MonitorForm.PausePC.Enabled;
  mnuPCStop.Enabled := MonitorForm.StopPC.Enabled;
  if inScan then mnuOptions.Enabled := False else mnuOptions.Enabled := True;
end;

procedure TMainForm.ScanMenuPopup(Sender: TObject);
begin
  mnuPCRun.Enabled := MonitorForm.StartPC.Enabled;
  mnuPCPause.Enabled := MonitorForm.PausePC.Enabled;
  mnuPCStop.Enabled := MonitorForm.StopPC.Enabled;
  mnuSaveReport.Enabled := SaveBTN.Enabled;
  if inScan then mnuScanStart.Enabled := False else mnuScanStart.Enabled :=
True;
  if inScan then mnuStopScan.Enabled := True else mnuStopScan.Enabled := False;
end;

procedure TMainForm.mnuScanStartClick(Sender: TObject);
begin
  ScanBTN.Click;
end;

procedure TMainForm.mnuStopScanClick(Sender: TObject);
begin
  ScanBTN.Click;
end;

procedure TMainForm.mnuSaveReportClick(Sender: TObject);
begin
  SaveBTN.Click;
end;

procedure TMainForm.CopyRightLabelClick(Sender: TObject);
  Const
  begin
  ShellExecute(0,'',pChar(''+URL),NIL,NIL,SW_SHOWNORMAL);
end;

end.
```

Файл AntiVir_Monitor.pas – монітор (контроль процесів)

```
unit AntiVir_Monitor;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, ExtCtrls, avKernel, avTypes, TLHelp32, Psapi;

type
  TMonitorForm = class(TForm)
    TopPanel: TPanel;
    BackImage: TImage;
    InformationLabel: TLabel;
    Image1: TImage;
    InfoLabel: TLabel;
    Bevel: TBevel;
    StartPC: TButton;
    PausePC: TButton;
    ClosePC: TButton;
    LastInfectBox: TGroupBox;
    Edit1: TEdit;
    Edit2: TEdit;
    LastFileBox: TGroupBox;
    Edit3: TEdit;
    InfoPCLabel: TGroupBox;
    PCScaned: TLabel;
    PCInfected: TLabel;
    PCStat: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    PCTime: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Timer1: TTimer;
    StopPC: TButton;
    Timer2: TTimer;
    procedure StartPCClick(Sender: TObject);
    procedure PausePCClick(Sender: TObject);
    procedure ClosePCClick(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure StopPCClick(Sender: TObject);
    procedure Timer2Timer(Sender: TObject);
    procedure CreateParams(var Params: TCreateParams); override;
    Procedure StartMonitor;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  MonitorForm : TMonitorForm;
  H,M,S       : integer;
  MonPaused   : Boolean = False;
  isMonRun    : Boolean = False;
  ProcList    : TStrings;
  FileLast    : String;
  FileLastID  : integer;
implementation

uses AntiVir_Main, AntiVir_InfectedAction, AntiVir_Options;
//***Функція створення параметрів сканування***//

procedure TMonitorForm.CreateParams(var Params: TCreateParams);
begin
  inherited CreateParams(Params);
```

```
  with params do
    ExStyle := ExStyle or WS_EX_APPWINDOW;
end;

//***Функція відображення вікна попередження про віруси***//

Procedure ShowAlarmForm(FileName,VirName: String);
  var
  ActFrm : TActionForm;
begin
  if OptionsForm.PCAutoAction.Checked then
  begin
    if OptionsForm.PCDelInfect.Checked then
      if Not DeleteFileBC(FileName) then ShowMessage(MainForm.DelError);
    Exit;
  end;
  ActFrm := TActionForm.Create(nil);
  with ActFrm do begin
    Edit1.Text := FileName;
    Edit2.Text := VirName;
  end;
  ActFrm.Show;
  SetForegroundWindow(ActFrm.Handle);
  ActFrm.SetFocus;
end;

//***Функція створення журналу перевірки***//

procedure CreateWinProcessList(List: Tstrings);
  var
  hSnapShot: THandle;
  ProcInfo: TProcessEntry32;
begin
  if List = nil then Exit;
  hSnapShot := CreateToolHelp32Snapshot(TH32CS_SNAPPROCESS, 0);
  if (hSnapShot <> THandle(-1)) then
  begin
    ProcInfo.dwSize := SizeOf(ProcInfo);
    if (Process32First(hSnapshot, ProcInfo)) then
    begin
      List.Add(ProcInfo.szExeFile);
      while (Process32Next(hSnapShot, ProcInfo)) do begin
        List.Add(ProcInfo.szExeFile);
        end;
    end;
    CloseHandle(hSnapShot);
  end;
end;

procedure CreateWinNTProcessList(List: TStrings);
  var
  PIDArray: array [0..1023] of DWORD;
  cb: DWORD;
  I: Integer;
  ProcCount: Integer;
  hMod: HMODULE;
  hProcess: THandle;
  ModuleName: array [0..300] of Char;
begin
  if List = nil then Exit;
  EnumProcesses(@PIDArray, SizeOf(PIDArray), cb);
  ProcCount := cb div SizeOf(DWORD);
  for I := 0 to ProcCount - 1 do
  begin
    hProcess := OpenProcess(PROCESS_QUERY_INFORMATION or
      PROCESS_VM_READ,
      False,
      PIDArray[I]);
    if (hProcess <> 0) then
```

```
    begin
      EnumProcessModules(hProcess, @hMod, SizeOf(hMod), cb);
      GetModuleFilenameEx(hProcess, hMod, ModuleName, SizeOf(ModuleName));
      if FileExists(ModuleName) then
        List.Add(ModuleName);
      CloseHandle(hProcess);
    end;
  end;
end;

procedure GetProcessList(List: Tstrings);
  var
  ovi: TOSVersionInfo;
begin
  if List = nil then Exit;
  ovi.dwOSVersionInfoSize := SizeOf(TOSVersionInfo);
  GetVersionEx(ovi);
  case ovi.dwPlatformId of
    VER_PLATFORM_WIN32_WINDOWS: CreateWinProcessList(List);
    VER_PLATFORM_WIN32_NT: CreateWinNTProcessList(List);
  end
end;

//***Функція знищення процесу вірусу***//

function KillProcess(ProcCapt: String): boolean;
  var
  ProgCap       : string;
  hSnapShot     : THandle;
  AntiVir_Process      : PROCESSENTRY32;
  r             : longbool;
  KillProc      : DWORD;
  hProcess      : THandle;
  cbPriv        : DWORD;
  Priv,PrivOld  : TOKEN_PRIVILEGES;
  hToken        : THandle;
  dwError       : DWORD;
begin
  ProgCap:= ProcCapt;
  hSnapShot:=CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS,0);
  AntiVir_Process.dwSize := Sizeof(uProcess);

  try
    if(hSnapShot<>0)then
    begin
      r:=Process32First(hSnapShot, AntiVir_Process);
      while r <> false do
      begin
        if  ProgCap = AntiVir_Process.szExeFile then
          KillProc:= AntiVir_Process.th32ProcessID;
        r:=Process32Next(hSnapShot, AntiVir_Process);
      end;
    CloseHandle(hProcess);
    CloseHandle(hSnapShot);
    end;
  except
  end;

  hProcess:=OpenProcess(PROCESS_TERMINATE,false,KillProc);
  if hProcess = 0 then
  begin
    cbPriv:=SizeOf(PrivOld);
    OpenThreadToken(GetCurrentThread,TOKEN_QUERY or
TOKEN_ADJUST_PRIVILEGES,false,hToken);
    OpenProcessToken(GetCurrentProcess,TOKEN_QUERY or
TOKEN_ADJUST_PRIVILEGES,hToken);
    Priv.PrivilegeCount:=1;
    Priv.Privileges[0].Attributes:=SE_PRIVILEGE_ENABLED;
    LookupPrivilegeValue(nil,'SeDebugPrivilege',Priv.Privileges[0].Luid);
```

```
    AdjustTokenPrivileges(hToken,false,Priv,SizeOf(Priv),PrivOld,cbPriv);
    hProcess:=OpenProcess(PROCESS_TERMINATE,false,KillProc);
    dwError:=GetLastError;
    cbPriv:=0;
    AdjustTokenPrivileges(hToken,false,PrivOld,SizeOf(PrivOld),nil,cbPriv);
    CloseHandle(hToken);
  end;

  if TerminateProcess(hProcess,$FFFFFFFF) then
  begin
    Result := True;
  end
  else
  begin
    Result := False;
  end;
end;

//***Функція перехвату управління процесами***//

Procedure ExecuteProcessControl;
  var
  i, ID: integer;
begin
  ProcList := TStringList.Create;
  GetProcessList(ProcList);
  For i := 0 to ProcList.Count-1 do
  begin
    Application.ProcessMessages;
    MainForm.MonFileCN := MainForm.MonFileCN + 1;
    MonitorForm.Label4.Caption := inttostr(MainForm.MonFileCN);
    MonitorForm.Edit3.Text := ProcList[i];
    ID := _ScanFileEx(ProcList[i]);
    if ID <> -1 then begin
      MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.ProcControlSt+ ' ' + '['+MainForm.INFECTED+' - '+GetVirusName(ID)+']
'+ProcList[i]);
      MainForm.MonFileInfected := MainForm.MonFileInfected + 1;
      MonitorForm.Label5.Caption := inttostr(MainForm.MonFileInfected);
      MonitorForm.Edit2.Text := GetVirusName(id);
      MonitorForm.Edit1.Text := ProcList[i];
      MainForm.BalloonTrayIcon(MainForm.Handle
,1,10,ProcList[i],'['+MainForm.INFECTED+' - '+GetVirusName(id)+' ]',bitError);
      if OptionsForm.PCAutoKill.Checked then
        if Not KillProcess(ExtractFileName(ProcList[i])) then
Showmessage(MainForm.ErrorKillProc);
      ShowAlarmForm(ProcList[i],'['+MainForm.INFECTED+' - '+GetVirusName(id)+'
]');
    end;
  end;
  FileLast   := ProcList[ProcList.count-1];
  FileLastID := ProcList.count-1;
end;

//***Функція управління процесами***//

Procedure StartProcessControl;
begin
  if isMonRun = False then begin
    ExecuteProcessControl;
    MonitorForm.Timer2.Enabled := true;
    isMonRun := true;
    MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.PCInit);
  end else
    if MonPaused then begin
      MonPaused := False;
      MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.PCRestore);
```

```
    end;
end;

Procedure PauseProcessControl;
begin
  MonPaused := True;
  MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.PCPause);
end;

Procedure ResumeProcessControl;
begin
  MonPaused := False;
  MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.PCRestore);
end;

Procedure ExitProcessControl;
begin
  isMonRun := False;
  ProcList.Free;
  MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.PCStop);
end;

//***Функція старту моніторингу змін у системі***//

{$R *.dfm}
Procedure TMonitorForm.StartMonitor;
begin
  StartProcessControl;
  PausePC.Enabled := True;
  StopPC.Enabled := True;
  StartPC.Enabled := False;
end;

procedure TMonitorForm.StartPCClick(Sender: TObject);
begin
  StartMonitor;
end;

procedure TMonitorForm.PausePCClick(Sender: TObject);
begin
  PauseProcessControl;
  PausePC.Enabled := False;
  StopPC.Enabled := True;
  StartPC.Enabled := True;
end;

procedure TMonitorForm.ClosePCClick(Sender: TObject);
begin
  Close;
end;
procedure TMonitorForm.Timer1Timer(Sender: TObject);
  var
  ss,mm,hh:String;
begin
  if isMonRun then
    if Not MonPaused then
      Label7.Caption := MainForm.PCActive
    else
      Label7.Caption := MainForm.PCPaused;
  if not isMonRun then
    Label7.Caption := MainForm.PCStoped;
  if isMonRun then
  if Not MonPaused then
  begin
    s:=s+1;
    if s = 59 then
```

```
    begin
      s:=0;
      m:=m+1;
    end;
    if m = 59 then
    begin
      m:=0;
      h:=h+1;
    end;
    ss:=inttostr(s);
    mm:=inttostr(m);
    hh:=inttostr(h);
    if length(ss) = 1 then ss:='0'+ss;
    if length(mm) = 1 then mm:='0'+mm;
    if length(hh) = 1 then hh:='0'+hh;
    Label8.Caption := hh+':'+mm+':'+ss;
  end;
end;
procedure TMonitorForm.StopPCClick(Sender: TObject);
begin
  ExitProcessControl;
  PausePC.Enabled := False;
  StopPC.Enabled := False;
  StartPC.Enabled := True;
end;
procedure TMonitorForm.Timer2Timer(Sender: TObject);
  var
  ID: integer;
begin
  if isMonRun = False then Exit;
  if MonPaused = False then
  begin
    ProcList.Clear;
    GetProcessList(ProcList);
    if ProcList.Count-1 <> FileLastID then
    if ProcList[ProcList.count-1] <> FileLast then
    Begin
      MainForm.MonFileCN := MainForm.MonFileCN + 1;
      MonitorForm.Label4.Caption := inttostr(MainForm.MonFileCN);
      MonitorForm.Edit3.Text := ProcList[ProcList.count-1];
      ID := _ScanFileEx(ProcList[ProcList.count-1]);
      if ID <> -1 then
      begin
        MainForm.ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+MainForm.ProcControlSt+ ' ' + '['+MainForm.INFECTED+' - '+GetVirusName(ID)+']
'+ProcList[ProcList.count-1]);
        MainForm.MonFileInfected := MainForm.MonFileInfected + 1;
        MonitorForm.Label5.Caption := inttostr(MainForm.MonFileInfected);
        MonitorForm.Edit2.Text := GetVirusName(ID);
        MonitorForm.Edit1.Text := ProcList[ProcList.count-1];
        MainForm.BalloonTrayIcon(MainForm.Handle ,1,10, ProcList[ProcList.count-
1] ,'['+MainForm.INFECTED+' - '+GetVirusName(ID)+' ]',bitError);
        if OptionsForm.PCAutoKill.Checked then
        if Not KillProcess(ExtractFileName(ProcList[ProcList.count-1])) then
Showmessage(MainForm.ErrorKillProc);
        ShowAlarmForm(ProcList[ProcList.count-1],'['+MainForm.INFECTED+' -
'+GetVirusName(ID)+' ]');
      end;
      FileLast   := ProcList[ProcList.count-2];
      FileLastID := ProcList.count-1;
    end else begin
      FileLast   := ProcList[ProcList.count-1];
      FileLastID := ProcList.count-2;
    end;
  end;
end;

end.
```

Файл AntiVir_InfectedAction.pas – вибір дії над інфікованим об'єктом

```
unit AntiVir_InfectedAction;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, avKernel;

type
  TActionForm = class(TForm)
    DeleteVir: TButton;
    SkipVir: TButton;
    ApplyToAll_Check: TCheckBox;
    Bevel1: TBevel;
    InfoInfectedBox: TGroupBox;
    InfoVirusInfo: TGroupBox;
    Edit1: TEdit;
    VirInfo_2: TLabel;
    VirInfo_0: TLabel;
    VirInfo_1: TLabel;
    TopPanel: TPanel;
    BackImage: TImage;
    InformationLabel: TLabel;
    InfoLabel: TLabel;
    Image2: TImage;
    Bevel: TBevel;
    Edit2: TEdit;
    procedure SkipVirClick(Sender: TObject);
    procedure DeleteVirClick(Sender: TObject);
    procedure CreateParams(var Params: TCreateParams); override;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ActionForm: TActionForm;

implementation

uses AntiVir_Main, AntiVir_Options;

{$R *.dfm}
procedure TActionForm.CreateParams(var Params: TCreateParams);
begin
  inherited CreateParams(Params);
  with params do
    ExStyle := ExStyle or WS_EX_APPWINDOW;
end;

procedure TActionForm.SkipVirClick(Sender: TObject);
begin
  if ApplyToAll_Check.Checked then
  begin
    OptionsForm.PCAutoAction.Checked := True;
    OptionsForm.PCSkipInfect.Checked := true;
    OptionsForm.SaveOptions;
  end;
  Close;
end;
//*********функція знищення вірусу**********
procedure TActionForm.DeleteVirClick(Sender: TObject);
begin
  if ApplyToAll_Check.Checked then
  begin
    OptionsForm.PCAutoAction.Checked := True;
```

```
    OptionsForm.PCDelInfect.Checked := true;
    OptionsForm.SaveOptions;
  end;
  if Not DeleteFileBC(Edit1.Text) then ShowMessage(MainForm.DelError)
  else Close;
end;

end.
```

Файл AntiVir_AddPath.pas – додавання шляхів сканування

```
unit AntiVir_AddPath;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, ShellCtrls;

type
  TAddUserPathForm = class(TForm)
    Bevel: TBevel;
    TopPanel: TPanel;
    Image13: TImage;
    InformationLabel: TLabel;
    InfoLabel: TLabel;
    ApplyBTN: TButton;
    CanselBTN: TButton;
    ShellTreeView: TShellTreeView;
    Image1: TImage;
    procedure CanselBTNClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ShellTreeViewClick(Sender: TObject);
    procedure ApplyBTNClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AddUserPathForm: TAddUserPathForm;

implementation

uses AntiVir_Main, AntiVir_Options, AntiVir_SelInfo;

{$R *.dfm}

procedure TAddUserPathForm.CanselBTNClick(Sender: TObject);
begin
  Close;
end;

procedure TAddUserPathForm.FormShow(Sender: TObject);
begin
  ApplyBTN.Enabled := false;
end;

procedure TAddUserPathForm.ShellTreeViewClick(Sender: TObject);
begin
  if DirectoryExists(ShellTreeView.Path+'\') then
    ApplyBTN.Enabled := True else
    ApplyBTN.Enabled := False;
end;
procedure TAddUserPathForm.ApplyBTNClick(Sender: TObject);
begin
  with OptionsForm.PathList.Items.Add do
  begin
    Caption := ShellTreeView.Path+'\';
    if DirectoryExists(Caption) then ImageIndex := 4 else ImageIndex := 5;
  end;
  OptionsForm.SaveOptions;
  MainForm.CreateDrivesList(MainForm.PathList);
  Close;
end;

                                    end.
```

Файл AboutFrm.pas – довідка

```
unit AboutFrm;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Buttons, ShellAPI, ComCtrls, jpeg;

type
  TAboutForm = class(TForm)
    Bevel2: TBevel;
    Panel1: TPanel;
    OkBTN: TBitBtn;
    Bevel1: TBevel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Image1: TImage;
    procedure OkBTNClick(Sender: TObject);
    procedure LinkLabelClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AboutForm: TAboutForm;

implementation

uses AntiVir_Main;

{$R *.dfm}

procedure TAboutForm.OkBTNClick(Sender: TObject);
begin
  Close;
end;

end.
```