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 антивірусного захисту під ОС Windows”

Виконав здобувач вищої освіти
IV курсу, групи КБ-20-3СК
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Махно Я.В.
«____» _____ 2023 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Смірнов С.А.
«____» _____ 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

Махну Ярославу Владиславовичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows*

2. Керівник роботи *Смірнов Сергій Анатолійович, канд. техн. наук, доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 13-02 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 антивірусного захисту під ОС Windows*

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
_____ Смірнов С.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2023 р.

Підпис здобувача
_____ Махно Я.В.
(прізвище та ініціали)

АНОТАЦІЯ

Махно Я.В. Програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows. 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 антивірусного захисту під ОС Windows.

Метою розробки є програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows.

Результат роботи – програмна реалізація системи кібербезпеки антивірусного захисту під ОС Windows.

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, ОС Windows

ABSTRACT

Makhno Ya.V. Software of the cyber security system of antivirus protection for Windows OS. 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 of antivirus protection under the Windows OS.

The purpose of the development is the software of the cyber security system of antivirus protection under the Windows OS.

The result of the work is the software implementation of the cyber security system of antivirus protection under the Windows OS.

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, Windows OS

ЗМІСТ

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. Антивірусний продукт – це програма, призначена для виявлення та видалення вірусів та іншого шкідливого програмного забезпечення з вашого комп'ютера чи ноутбука.

Шкідливе програмне забезпечення , відоме як зловмисне програмне забезпечення , – це код, який може пошкодити ваші комп'ютери та ноутбуки та дані на них. Ваші пристрої можуть заразитися, випадково завантаживши зловмисне програмне забезпечення, яке міститься у вкладенні, пов'язаному з сумнівним електронним листом, або приховано на USB-накопичувачі, або навіть просто відвідавши хитрий веб-сайт.

Потрапивши на ваш комп'ютер або ноутбук, зловмисне програмне забезпечення може викрасти ваші дані, зашифрувати їх, щоб ви не мали до них доступу, або навіть повністю стерти. З цієї причини важливо завжди використовувати антивірусне програмне забезпечення та підтримувати його в актуальному стані, щоб захистити свої дані та пристрої.

Антивірусні продукти працюють шляхом виявлення, ізолювання та/або видалення зловмисного коду, щоб запобігти пошкодженню вашого пристрою зловмисним програмним забезпеченням. Сучасні антивірусні продукти оновлюються автоматично, щоб забезпечити захист від найновіших вірусів та інших типів зловмисного програмного забезпечення.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows.

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

– Огляд існуючих систем антивірусного захисту під ОС Windows.

– Дослідження системи кібербезпеки антивірусного захисту під ОС Windows.

– Програмна реалізація системи кібербезпеки антивірусного захисту під ОС Windows.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі антивірусного захисту під ОС Windows.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows, 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

Антивірусне програмне забезпечення часто безкоштовно входить до складу операційних систем, на яких працюють комп'ютери Windows і Apple. Якщо ви переконаєтеся, що цей вбудований антивірус увімкнено, ви миттєво будете в безпеці.

На нових комп'ютерах часто встановлено пробну версію окремого антивірусного продукту (наприклад, McAfee, Norton і Avast). Слід зауважити, що:

– коли термін дії пробної версії закінчиться, вам доведеться заплатити (або зареєструватися), щоб продовжити її використання;

– окремі антивірусні продукти не завжди працюватимуть разом із вбудованим антивірусним програмним забезпеченням і навіть можуть повністю припинити його роботу;

– з такою кількістю доступних продуктів, ви можете провести власне дослідження, щоб дізнатися, який підходить саме вам.

Як використовувати свій антивірусний продукт?

1. Коли ви вперше встановлюєте (або вмикаєте) свій антивірусний продукт, запустіть повну перевірку , щоб переконатися, що ваш комп'ютер вільний від усіх відомих шкідливих програм.

2. Переконайтеся, що ваше антивірусне програмне забезпечення налаштовано на автоматичне сканування всіх нових файлів, наприклад файлів, завантажених з Інтернету або збережених на USB-накопичувачі, зовнішньому жорсткому диску, SD-карті чи іншому типі знімного носія.

3. Переконайтеся, що ваше антивірусне програмне забезпечення налаштовано на автоматичне отримання оновлень.

Чи потрібні мені антивірусні продукти на моєму смартфоні та планшеті?

Ні, за умови, що ви встановлюєте програми та програмне забезпечення лише з офіційних магазинів, таких як Google Play і Apple App Store. Ви також повинні налаштувати свої програми (і сам планшет/смартфон) на автоматичне оновлення.

1.2 Область застосування

Областю застосування, розробляємого, у ході виконання бакалаврського проектування, програмного забезпечення є система кібербезпеки антивірусного захисту під ОС Windows. Головна якість антивірусного ПЗ – надійність виявлення шкідливого коду. На жаль, у реальних умовах перевірити це досить важко – адже для цього потрібен зразок вірусу, не відомого розроблювачам захисного ПЗ. З відомими сигнатурами, зрозуміло, упорається більшість сучасних антивірусних сканерів.

Антивірусне програмне забезпечення звичайно використовує два відмінних друг від друга метода для виконання своїх завдань:

– Сканування файлів для пошуку відомих вірусів, що відповідають визначенню в антивірусних базах.

– Виявлення підозрілого поводження кожної із програм, схоже на поводження зараженої програми.

По набору функцій і гнучкості налаштувань антивіруси можна розділити на:

1. Продукти для домашніх користувачів:

– Властиво антивіруси.

– Комбіновані продукти (наприклад, до класичного антивірусу доданий антиспам, файрвол, антируткіт і т.д.).

2. Корпоративні продукти:

– Серверні антивіруси.

– Антивіруси на робочих станціях («endpoint»).

– Антивіруси для поштових серверів.

– Антивіруси для шлюзів.

Зараз стало можливо й зараження мобільних телефонів вірусами, але тільки для телефонів на базі операційних систем, таких як Android, Symbian, Windows Mobile, Blackberry, iPhone OS. Усе більше розроблювачів пропонують антивірусні програми для боротьби з вірусами й захисту мобільних телефонів. У мобільних пристроях є наступні види боротьби з вірусами:

– сигнатурний;

– захист від спама по SMS;

– шифрування даних.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows, 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 цьому розділі ми провели огляд останніх версій найбільш популярних антивірусних продуктів класу Internet Security. Рішення дозволяють змінювати як рівень забезпечуваної безпеки, так і методи виявлення шкідливих файлів. Передбачені досить гнучкі налаштування мережного екрана й правил для додатків аналізатора HIPS. Компонент «Анти-Баннер», відключений за замовчуванням, дозволяє вирішити проблему нав'язливої реклами. Досить скачати з офіційного форуму чорний список рекламних адрес і включити даний модуль.

Критеріями нашого тестування стали користувальницькі характеристики додатків: тривалість повної перевірки системи, зручність інтерфейсу, завантаження ЦП і ОЗП при проведенні перевірки ПК. час установки програми. Слід зазначити. що ці показники сильно залежать від конфігурації ПК. Власникам нетбуків і старих комп'ютерів орієнтуватися на них не треба. До речі. деякі компанії, наприклад Trend Micro, пропонують спеціальні версії своїх продуктів для нетбуків.

Outpost Security Suite

Компанія Agnitum продовжує розвивати лінійку своїх продуктів для домашніх користувачів. Антивірус помітно виділяється серед конкурентів можливістю проведення тонкого налаштування кожного компонента захисту, що, безумовно, корисно для фахівців. Новачкам же пропонується готовий набір параметрів, що рекомендується для більшості користувачів.

У головному вікні програми можна побачити стан основних компонентів захисту, версію антивірусного додатка. Убудований у програму брандмауер відразу ж після установки починає працювати в режимі автонавчання, створюючи розв'язні правила для всіх додатку. А після розподілу програм по списках рекомендується вибрати більше безпечний режим роботи: блокування невідомих з'єднань або їхній дозвіл.

Рисунок 2.1 – Outpost Security Suite Pro (Режим автонавчання)

Варто звернути увагу й на компонент «Анти-Реклама», схожий з «Анти-Баннером».. Також передбачене блокування відкриття сторінок, що містять задані користувачем слова, що. безумовно, корисно при організації «Батьківського контролю». Програма надає можливість тонкого налаштування аналізатора HIPS, що дозволить досить точно встановити рівень прав кожного додатка. До достоїнств продукту варто віднести зручність інтерфейсу, гнучкість налаштувань, а також можливість відправлення з головного вікна підозрілого файлу у вірусну лабораторію.

ESET Smart Security

Компанія ESET представила оновлену версію свого продукту Smart Security, що відрізняється високою швидкістю сканування й невеликим завантаженням системних ресурсів.

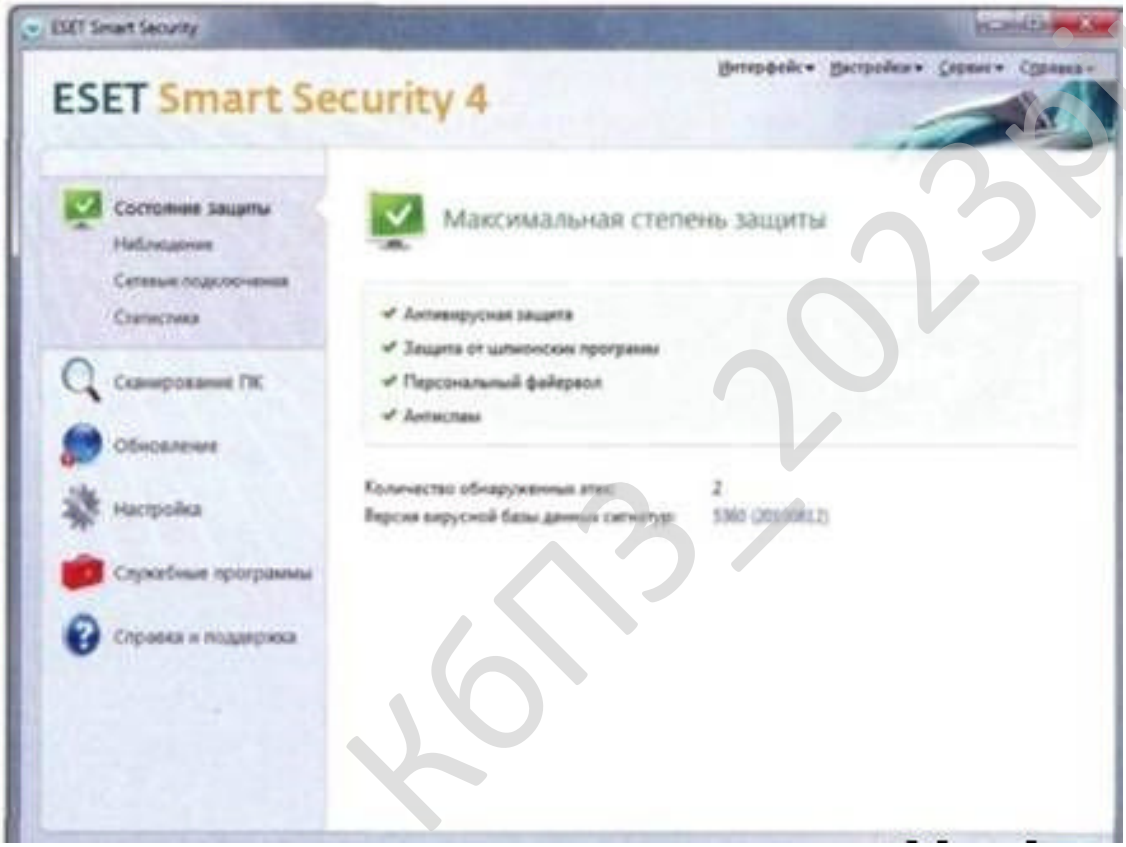

Рисунок 2.2 – Головне вікно ESET Smart Security

Програма передбачає звичайний і розширений режими роботи. У першому випадку користувачеві доступні лише самі необхідні компоненти захисту з можливістю їхнього включення й відключення: сканування ПК на наявність погроз, активування режимів «Блокувати всі», «Пропускати всі» для брандмауера, відновлення антивірусних баз. У другому – надається можливість провести досить тонке налаштування: змінити види виявлених погроз і методи їхнього виявлення, ступінь очищення заражених файлів та ін. Такі функції. як

«Карантин». «Монітор Мережі», «Статистика» і «Звіти», також доступні тільки при використанні розширеного режиму.

За сукупними результатами тестувань незалежних лабораторій VirusBulletin і AV-Comparatives, рішення компанії ESET показало дуже гарний результат серед відібраних для дослідження конкурсантів.

Позитивними сторонами даного рішення можна вважати швидкодію й простоту використання. А по гнучкості налаштувань воно перевершує інших учасників тестування. Але по функціональності й надійності воно поступилося лідерові.

Norton Internet Security

Продукти компанії Symantec завжди відрізнялися привабливим для користувача інтерфейсом, високою швидкістю роботи й установки. І дійсно, у нашім випадку інсталяція пройшла за 35 с., з деяким відривом від інших учасників тестування.

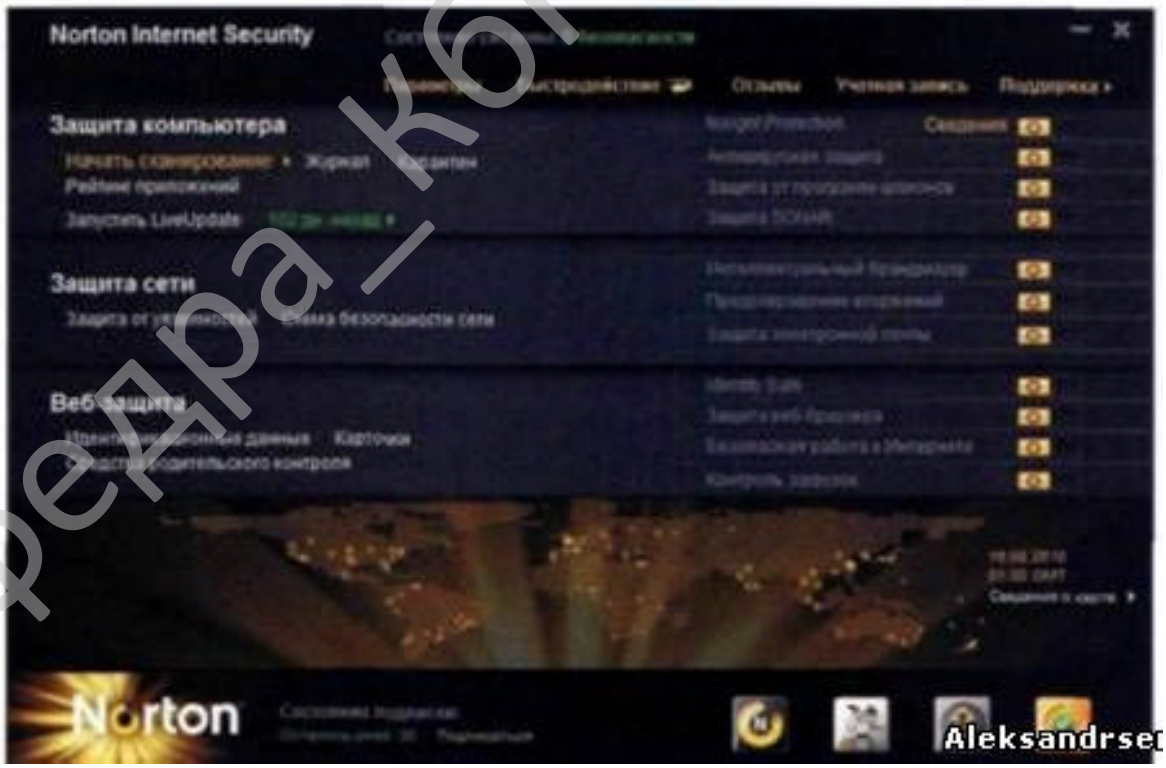

Рисунок 2.3 – Головне вікно Norton Internet Security

Програма вимагає мінімальної уваги до себе з боку користувача. Включена за замовчуванням «Автоматичний захист» перевіряє на льоту файли, що запускаються, й виявивши вредоносний, відразу ж видаляє його, поміщаючи копію на карантин. Ручний режим роботи не передбачений.

У головному вікні представлені вже стандартними функції, що стали: запуск завдань перевірки, включення, відключення й налаштування компонентів захисту. Поряд із цим слід зазначити й компонент Norton System Insight. Він дозволяє стежити за розподілом ресурсів ЦП між запущеними додатками, у тому числі й за самим Norton Internet Security.

Уникнути постійного введення ідентифікаційних даних дозволяє корисна функція Identify Safe, що зберігає їх у спеціальному захищеному сховищі.

Переваги Norton Internet Security полягають у простоті й зручності інтерфейсу, стійкості до збоїв і незначному впливі на продуктивність ПК.

F-Secure Internet Security

Антивірусний пакет F-Secure Internet Security містить антивірус, брандмауера, антиспам і контроль додатків. Простота інтерфейсу – важливе достоїнство продукту.

Рисунок 2.4 – Головне вікно F-Secure Internet Security

У головному вікні пропонується на вибір одне із трьох завдань: «Перегляд стану компонентів захисту», «Перегляд списку завдань» і «Статистика», і лише потім надається можливість перегляду їхньої інформативної частини.

У вікні налаштування антивірусу дозволено змінювати тільки основні параметри програми. Інтерфейс брандмауера реалізований краще: тут можна задати мережні правила, відкрити необхідний порт, побачити розподіл мережних ресурсів. Налаштування антивірусного компонента зводиться до вибору дій над зараженими об'єктами, включення проактивного захисту й вибору типів файлів, що перевіряються. Аналізатор HIPS також є присутнім у даному продукті, однак і його налаштування обмежуються двома параметрами – дозволити й заборонити запуск.

Достоїнствами продукту можна вважати якісний брандмауер, що дозволяє як запобігти, так і швидко виявити активне зараження завдяки можливості відслідковувати доступ до мережі кожної програми.

На жаль, недоліки цього ПЗ не можна назвати незначними: його відновлення займає досить тривалий час і нерідко завершується збоєм.

C-Data Internet Security

Антивірус G-Data Internet Security варто віднести до «легких» пакетів захисту ПК, незважаючи на великий розмір файлу дистрибутива. Інтерфейс розглянутого рішення представлений на вищому рівні. Також тут відбито у вигляді діаграми розподіл ресурсів ЦП, щоправда, лише між програмою й системою. У ході установки антивірус неодноразово запитує згоду користувача на щогодинне відновлення баз. Вікно налаштування передбачає звичайний і адміністративний режими. У першому можна лише переглядати збережені параметри, а змінювати їх – тільки в другому. Слід зазначити гнучкі налаштування брандмауера: надається можливість створення мережних правил як автоматично, так і вручну. Аналізатор HIPS у рішенні відсутній, але передбачений «Радар додатків», де можна заблокувати доступ певних програм до Інтернету.

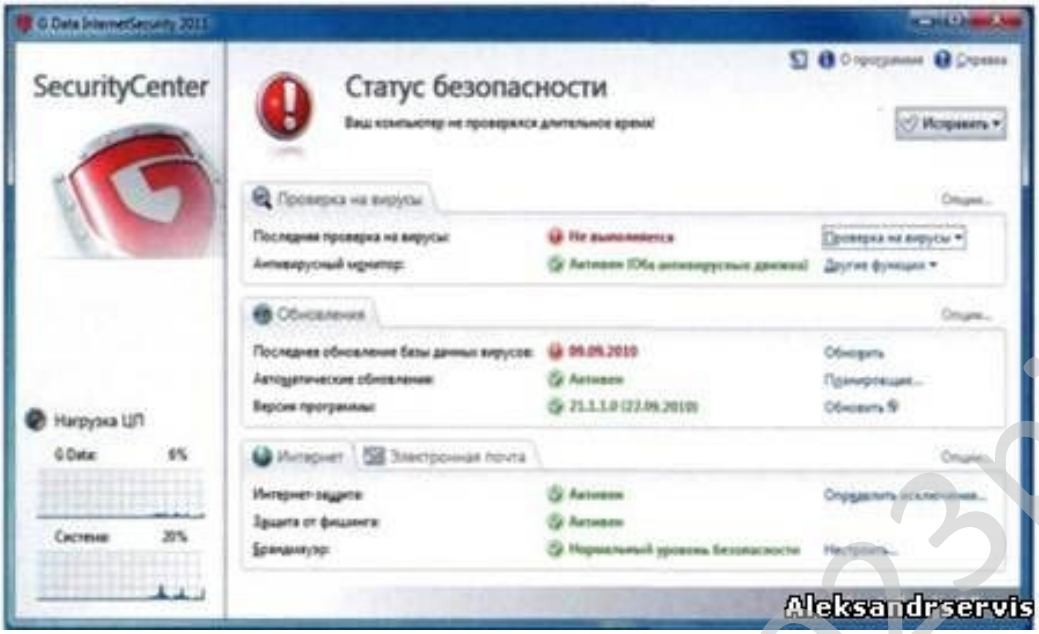

Рисунок 2.5 – Головне вікно C-Data Internet Security

Також користувачеві пропонується можливість вибору одного із двох антивірусних движків. За замовчуванням включені обоє, і настійно рекомендується не міняти це значення. Однак на повільних комп'ютерах для підвищення продуктивності, по заявах розроблювачів, допускається відключення одного з них. Правда. крім назв «Движок А» і «Движок В». ніякої інформації не дано, і не ясно, від чого ж все-таки пропонують відмовитися. Плюсами рішення є широкі можливості брандмауера й один із самих зручних, на наш погляд, інтерфейсів.

Panda Internet Security 2022

Розроблювачі компанії Panda звертають особливу увагу на функцію «Колективний розум», призначену для оперативного реагування вірусної лабораторії на нові погрози. Вступаючи в співтовариство, користувач дозволяє автоматичне відправлення в лабораторію Panda інформації про підозріле поводження програм, а також про можливі віруси. Malware і т.п. Крім стандартного набору функцій (антивірус, брандмауер. «Батьківський контроль»), тут реалізований «пісочниця», подібна до компонента «Безпечний перегляд веб-сайтів» Internet Security.

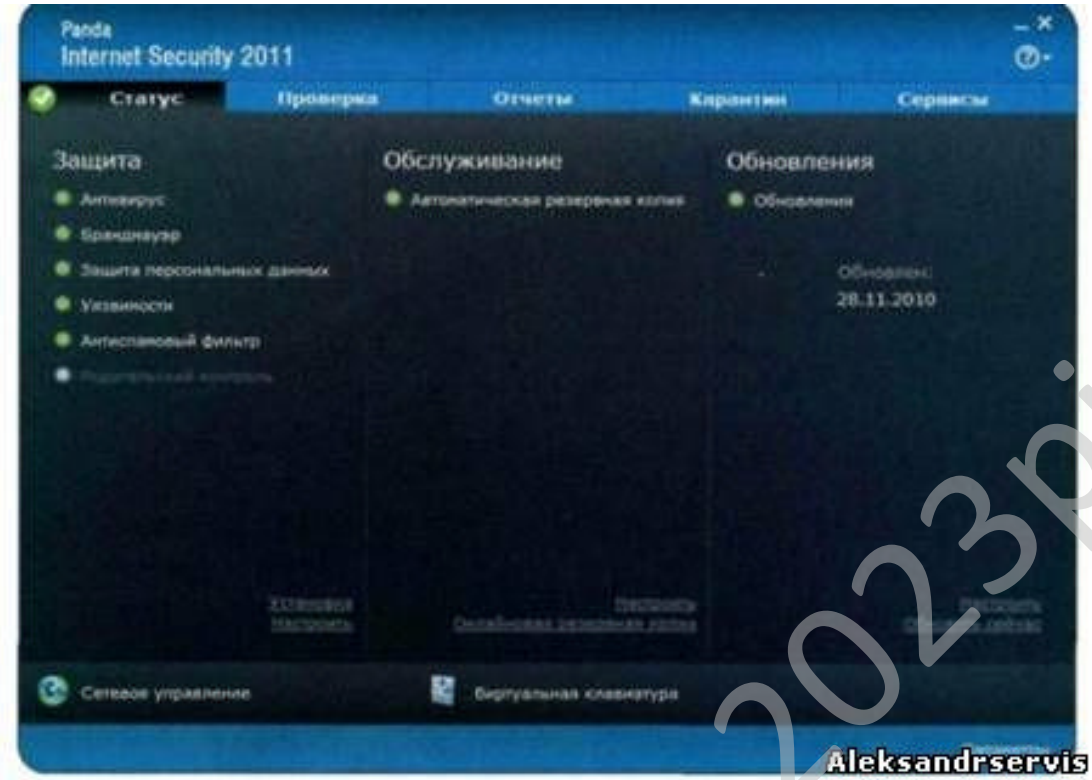

Рисунок 2.6 – Головне вікно Panda Internet Security 2022

Гнучкістю налаштувань рішення не відрізняється: передбачені зміни лише основних параметрів програми. Слід зазначити, що для видалення більшості вірусів потрібна перезавантаження комп'ютера. Незрозуміло, навіщо забирати час користувача у випадку, коли досить просто видалити заражений файл.

Компонент «Мережне керування» дозволяє переглядати статус захисту всіх ПК у мережі, на яких установлений антивірус Panda. Порадувала функція «Перегляд мережного трафіку». Вона надає докладну інформацію про додатки, що використовують мережу, і відкриті порти.

До переваг програми Panda Internet Security ставляться високий рівень детектування шкідливого ПЗ, наявність «пісочниці». а також можливість перегляду використання мережного трафіку. А от відсутність розширеного налаштування. а також примусове перезавантаження при видаленні будь-яких погроз – її недоліки.

Trend Micro Internet Security 2022

Головне вікно програми не занадто інформативно, у ньому відбиті лише статус захисту ПК і кількість виявлених погроз. По гнучкості налаштувань Trend Micro схожий з рішенням Panda, змінювати дозволено тільки самі необхідні параметри: включення й відключення сканування файлів при доступі, блокування потенційно небезпечних вебсайтів, а також спостереження за несанкціонованими змінами системних налаштувань.

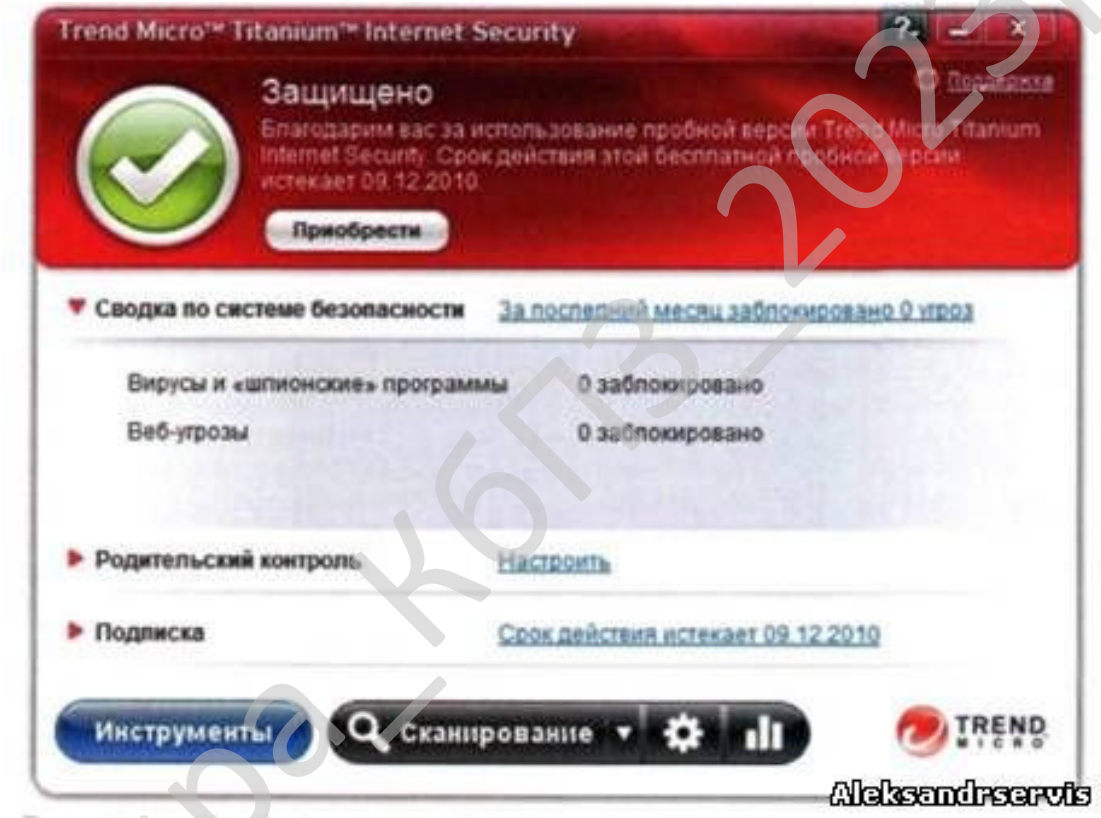

Рисунок 2.7 – Головне вікно Trend Micro Internet Security 2022

Самостійний брандмауер тут не реалізований, але передбачена функція Firewall Booster, призначена для розширення захисту, забезпечуваного брандмауером Windows.

Слід зазначити, що це єдиний продукт, де користувачеві доступний вибір способу завантаження драйверів захисту: при запуску, частково при запуску й

після запуску Windows. Правда, викликає подив, навіщо надавати можливість прийняття таких рішень користувачеві, адже якщо новачок або непрофесіонал вибере «Зверхпродуктивність» (запуск драйверів захисту після завантаження системи), комп'ютер буде підданий значному ризику, оскільки більшість вредоносних програм можуть блокувати запуск антивірусних програм.

Плюси рішення – невисока вартість і знижені вимоги до системних ресурсів, мінуси – відсутність власного брандмауера й незначна гнучкість налаштувань.

Dr.Web Security Space

Лінійка продуктів Security Space містить у собі рішення Dr.Web Security Space і Dr.Web Security Space 6.0 Pro. Єдине, але досить значиме їхнє розходження полягає в тому, що Pro-версія містить власного брандмауера.

Мережний фільтр починає роботу відразу після установки антивірусу. У режимі навчання (заданому за замовчуванням) видається запит права на використання мережі кожним процесом, який зробив спробу виходу в Інтернет. Користувачеві пропонується три варіанти: дозволити однократно, заборонити однократно, створити мережне правило для даного додатка.

Головного вікна в Dr.Web Security Space так і не з'явилося. За станом всіх компонентів захисту можна спостерігати при наведенні курсору миші на значок у треї. а якщо клацнути на ньому правою кнопкою миші, то легко настроїти кожний з них. Правда, для зміни параметрів, а також вимикання основних компонентів захисти потрібні права адміністратора. На наш погляд, для заборони зміни налаштувань ПЗ необхідно б використовувати призначуваний для цього пароль. По гнучкості налаштувань програма нічим не виділяється серед конкурентів. Можливості міжмережного екрана досить широкі. Дозволено зміни режиму роботи, параметрів мережних правил для додатків, налаштування пакетного фільтра й загальних правил для мережних інтерфейсів, а також скидання брандмауера. Безсумнівна перевага рішення – висока швидкість реакції вірусної лабораторії на нові погрози. Не обійшлося й без недоліків: перевірка ПК на наявність вірусів забирає тривалий час.

avast! Free Antivirus

Avast! може аналізувати систему на наявність руткитів при кожному старті Windows, уміє сканувати неформатовані дані на жорсткому диску, де можуть ховатися віруси, має модуль самозахисту, що запобігає несанкціоноване видалення або зараження важливих файлів самого антивірусу. Щоб уникнути помилкових спрацьовувань, avast! перевіряє наявність цифрових підписів у підозрілих файлів. Якщо запуск Windows виконується занадто довго, у налаштуваннях програми можна включити опцію її завантаження після інших системних служб.

Рисунок 2.8 – Головне вікно avast! Free Antivirus

Для ігрових ПК придасться функція «Без повідомлень» – при її активації avast! перестає видавати спливаючі вікна з попередженнями щоб уникнути перемикання на Робочий стіл у процесі гри. Якщо ви встановили антивірус на ПК

рідних або близьких, корисною виявиться опція, що сповіщає поштою про виявлення вірусів. Сканер у реальному часі перевіряє не файли, що виконуються тільки, але й допоміжні – наприклад бібліотеки DLL і скрипти, що запускаються. Також скануються документи, що відкриваються, а при підключенні з'ємних накопичувачів – файли, що запускаються автоматично.

Для евристичного аналізу можна набудовувати чутливість, щоб не було зайвих помилкових спрацьовувань. Також сканер перевіряє пошту й інстантмесенджери на предмет заражених файлів, досліджує вебтрафік, уміє відслідковувати завантаження в P2P-програмах, має базовий захист від мережних вторгнень. Як і ряд інших антивірусів, avast! пропонує користувачеві участь у власному співтоваристві, що означає згоду на передачу певної інформації на сервери компанії для більше оперативного виявлення погроз і реагування на них.

В avast! 6 додані 2 нові функції – AutoSandbox, що при виявленні підозрілого коду автоматично пропонує запустити додаток у безпечному середовищі, і WebRep, що представляє собою систему рейтингів для оцінки репутації вебсайтів і пошукових результатів (реалізована у вигляді плагинів для Internet Explorer і Firefox). Також поліпшений захист у реальному часі і є повна підтримка 64-бітових версій Windows.

Comodo Antivirus

При інсталяції вам порекомендують додатково встановити сервіс Comodo GeekBuddy, що надає техпідтримку фахівцями Comodo в обслуговуванні ПК і усуненні виникаючих проблем (із платною підпискою). Також пропонується застосовувати Comodo Secure DNS – за словами розроблювача, це знижує ризик атак з використанням зараження вмісту DNS-кешу, а також прискорює запуск веб-сайтів. Невеликий розмір інсталятора (2 МБ) пояснюється просто – це вебінсталятор, у процесі установки завантажуючий ще близько 150 МБ антивірусу і його баз даних. Як для безкоштовного продукту в Comodo Antivirus досить багата функціональність.

Рисунок 2.9 – Головне вікно Comodo Antivirus

Так, тут є вбудований файрвол і навіть віртуальне середовище для безпечного запуску підозрілих додатків (sandbox), причому з додатковими налаштуваннями, що дозволяють гнучко обмежувати такі програми в правах доступу до системних ресурсів. Для полегшення обслуговування в антивірусі є 3 заданих шаблони конфігурацій – Internet Security (збалансований захист із помірними налаштуваннями як антивірусу, так і файрволу), Proactive Security (максимальний захист) і Firewall Security (вибирається як опція за замовчуванням при установці лише файрволу). Аналогічна система профілів застосовується й для антивірусного сканера – тут можна вказати перевірку всіх локальних дисків, критично важливих папок або ж пошук spyware. Також користувач може створити й свій власний профіль.

Розширені налаштування при цьому досить докладні – дозволяється вибирати рівень евристичного аналізу, задіяти руткітсканер, кумулятивне сканування. Проактивний захист має 4 різних рівні, є опція для роботи на слабких

конфігураціях. При виявленні підозрілих файлів передбачена можливість їхнього відправлення в Comodo на розгляд.

Microsoft Security Essentials

Відносно новий продукт (перша версія випущена наприкінці 2009 р.), що прийшов на зміну відразу двом сервісам – платному Windows Live OneCare і безкоштовному Windows Defender (останній забезпечував захист тільки від adware і spyware). З кінця 2010 р. Microsoft Security Essentials пропонується в системі Windows Update відновлення, що рекомендується як. Також при установці програма перевіряє Windows на ліцензійність.

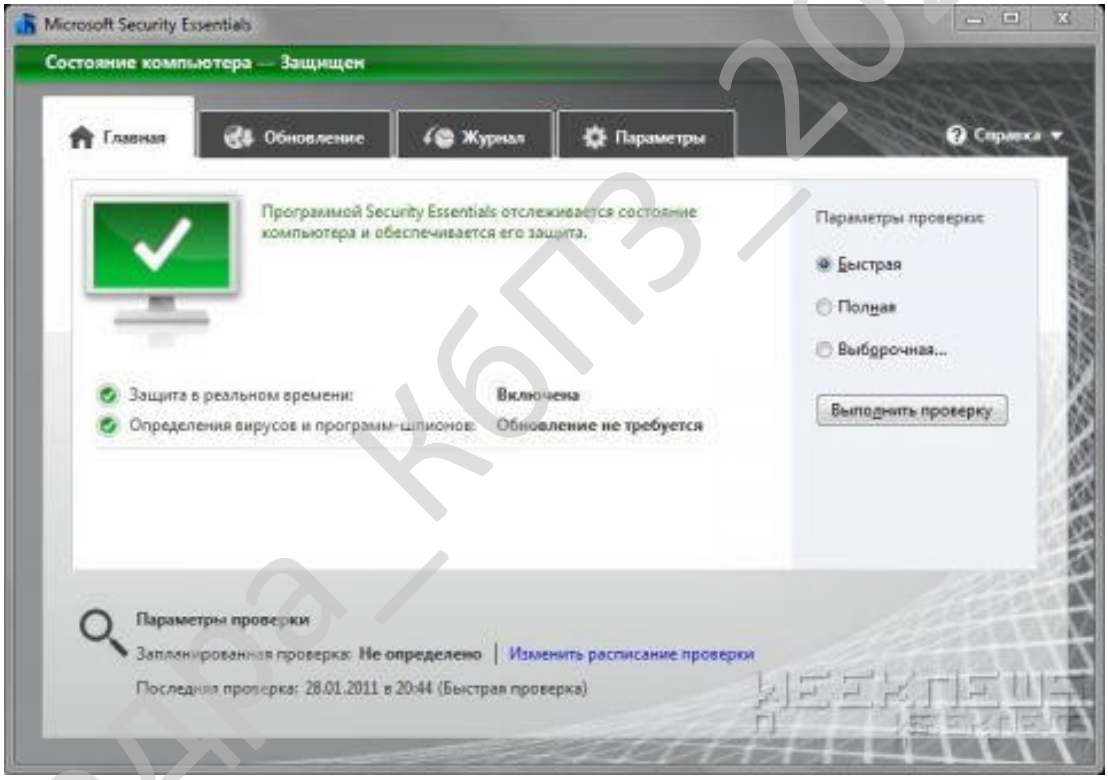

Рисунок 2.10 – Головне вікно Microsoft Security Essentials

Продукт швидко завоював популярність завдяки низькому споживанню системних ресурсів і «ненав'язливості» у роботі – без гострої потреби він про себе не нагадує, що, безсумнівно, по достоїнству оцінять багато користувачів, що не бажають відволікатися на рутинні операції по обслуговуванню ПК.

Налаштувань в Microsoft Security Essentials небагато. Є запуск сканування ОС за розкладом з можливістю обмеження споживання системних ресурсів, вибір дій, що вживаються, при виявленні погрози (залежно від її ступеня доступні варіанти «видалити», «помістити в карантин» і «дозволити» – лікування немає).

Програма може перевіряти архіви, знімні накопичувачі, створювати крапку відновлення системи перед усуненням виявлених вірусів. В Microsoft Security Essentials є захист у реальному часі – користувачеві дозволяється вибирати, які додатки варто перевіряти й чи варто аналізувати файли, що завантажуються з Інтернету.

Отже, підведемо підсумки. Кращим рішенням для захисту ПК був визнаний Internet Security 2022: взаємодія всіх його компонентів забезпечує найвищий ступінь захисту від сучасних погроз. А звання «Краща покупка» одержав антивірус Outpost Security Suite, що забезпечує високий рівень захисту відразу для трьох ПК за помірну плату.

Продукти Norton і G-Data відрізняються простотою використання й високою швидкістю сканування. За результатами тестування незалежної лабораторії рішення G-Data випередило конкурента по якості виявлення погроз.

Програма Outpost Security Suite виділяється серед конкурентів потужним брандмауером. Однак швидкість реакції вірусних лабораторій Agnitum нижче, що підвищує ризик зараження.

Антивірусний пакет Dr.Web Security Space 6.0 Pro практично аналогічний Panda Internet Security, у їхній состав входить досить якісний брандмауер.

Продукти F-Secure і Trend Micro подібні по функціональності й гнучкості налаштувань, однак серйозним недоліком Trend Micro є відсутність власного брандмауера. F-Secure. у свою чергу, відрізняється частими помилками під час відновлення.

Користувачам систем захисту Symantec і F-Secure доступні гаджети для бічної панелі, що також дозволяють стежити за станом захисту ПК.

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 антивірусного захисту під ОС Windows.

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

Кібербезпека – це засіб, за допомогою якого окремі особи та організації зменшують ризик постраждати від кіберзлочинності.

Основною функцією кібербезпеки є захист пристроїв, якими ми всі користуємося (смартфонів, ноутбуків, планшетів і комп'ютерів), а також послуг, до яких ми отримуємо доступ в Інтернеті – як вдома, так і на роботі – від крадіжки чи пошкодження. Це також запобігання несанкціонованому доступу до великої кількості особистої інформації , яку ми зберігаємо на цих пристроях і в Інтернеті.

Кібербезпека важлива, оскільки смартфони, комп'ютери та Інтернет зараз є настільки важливою частиною сучасного життя, що важко уявити, як би ми працювали без них. Від онлайн-банкінгу та покупок до електронної пошти та соціальних мереж – як ніколи важливо вжити заходів, які можуть запобігти кіберзлочинцям отримати доступ до наших облікових записів, даних і пристроїв.

Антивіруси (AV-продукти) традиційно працювали шляхом сканування кожного файлу на пристрої та пошуку зловмисного програмного забезпечення, виявляючи відомі сигнатури, але прогрес у зловмисному програмному забезпеченні та зміни базових платформ обмежили ефективність цього підходу. Незважаючи на те, що зловмисне програмне забезпечення є більш потужним, багато ризиків, від яких захищені традиційні AV-продукти, і багато іншого, тепер пом'якшуються платформою за замовчуванням без додаткових витрат.

Оскільки ці функції тепер звичайні, доцільно запитати, чи потрібно вам усе ще використовувати системи кібербезпеки антивірусного захисту на своїх мобільних пристроях.

У інструкціях для мобільних платформ рекомендуємо адміністраторам дозволяти додатки надходити лише через офіційні магазини програм. Це означає, що всі користувачі отримають вигоду від перевірки безпеки, яка виконується для програм, які є частиною цих магазинів програм.

Незважаючи на це, приблизно 0,05% користувачів Android, які завантажують свої програми виключно з Google Play, у певний момент отримують потенційно шкідливу програму (PHA) на своєму пристрої, а також були випадки PHA в iOS App Store. Таким чином, деякі адміністратори та власники ризиків можуть захотіти розглянути дозвіл на перелік (також рекомендовано в наших інструкціях), щоб дозволити встановлення лише схваленого набору програм із магазинів.

Реалізація дозволеного списку означає, що адміністратори можуть перевірити, чи програма, яку запитує користувач, належним чином балансує між потребами бізнесу та ризиком для безпеки, і чи мають розробники програми добру репутацію. Дозволити перелік можна надійно застосувати на сучасних пристроях завдяки використанню підпису коду для ідентифікації програм та їхніх розробників.

Незалежно від того, реалізовано дозволений список чи ні, якщо PHA якимось чином потрапляє в офіційний магазин додатків і згодом виявляється зловмисним, сам магазин додатків може як видалити програму зі своїх магазинів, так і видалити PHA з пристроїв ваших користувачів у точно так само, як і системи кібербезпеки антивірусного захисту.

Коли вам не потрібні системи кібербезпеки антивірусного захисту?

Тоді це означає, що якщо ви дотримуєтеся наших вказівок і розгортаєте сучасні пристрої з дозволеним списком і ізольованим програмним середовищем, і ви отримуєте свої програми через офіційні магазини програм, ви насправді не зменшуєте жодних додаткових використання сторонніх антивірусних програм або продуктів для захисту від шкідливих програм. Цю точку зору поділяє керівник відділу безпеки продуктів Google, який вважає, що 99% користувачів

Android не потребують додаткового програмного забезпечення безпеки на своїх пристроях.

Навіть якщо ваші користувачі входять до групи, яка не включена в список дозволів, і вони можуть завантажувати або встановлювати програми поза офіційними каналами магазину додатків, на Android також є перевірка програм, яка діє як традиційний AV-продукт і регулярно сканує програм, зменшуючи ризик завантаження шкідливих програм на платформу з ненадійного джерела.

В iOS єдиний реальний спосіб завантажувати додатки збоку – це зробити їх джейлбрейком, про запобігання якому ми писали раніше.

Коли можуть знадобитися системи кібербезпеки антивірусного захисту?

Ви можете розглянути можливість використання системи кібербезпеки антивірусного захисту на старіших пристроях, які не можна оновити, щоб включити новітні функції безпеки, як-от перевірку додатків, хоча ми завжди рекомендуємо використовувати оновлені та підтримувані пристрої, щоб справді мінімізувати ризик інфікування.

Деякі AV-продукти мають інші функції безпеки, які вам можуть знадобитися для вашої платформи – по суті використання системи кібербезпеки антивірусного захисту як продукт безпеки кінцевої точки. Наприклад, системи кібербезпеки антивірусного захисту можуть блокувати окремі фотографії, конфіденційні документи або програми за допомогою PIN-коду ; інші функції можуть включати фотографування користувача, коли він намагається та не вдається розблокувати пристрій – щось подібне до фотопастки. Однак враховуйте джерело та репутацію вашого AV-продукту, оскільки раніше з ними виявлялися проблеми. Проведіть деякі дослідження, а також перевірте їхній вплив на час автономної роботи та продуктивність пристрою.

3.2 Розробка структурної схеми

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

Структурна схема системи складається з наступних основних структурних блоків:

– Захист у режимі реального часу.

– Сканування системи.

– Контроль запущених процесів.

– Ядро антивірусної програми.

– Бази даних вірусів.

Розглянемо більш детальніше склад структурних блоків.

Структурний блок захисту у режимі реального часу складається з наступних структурних 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.

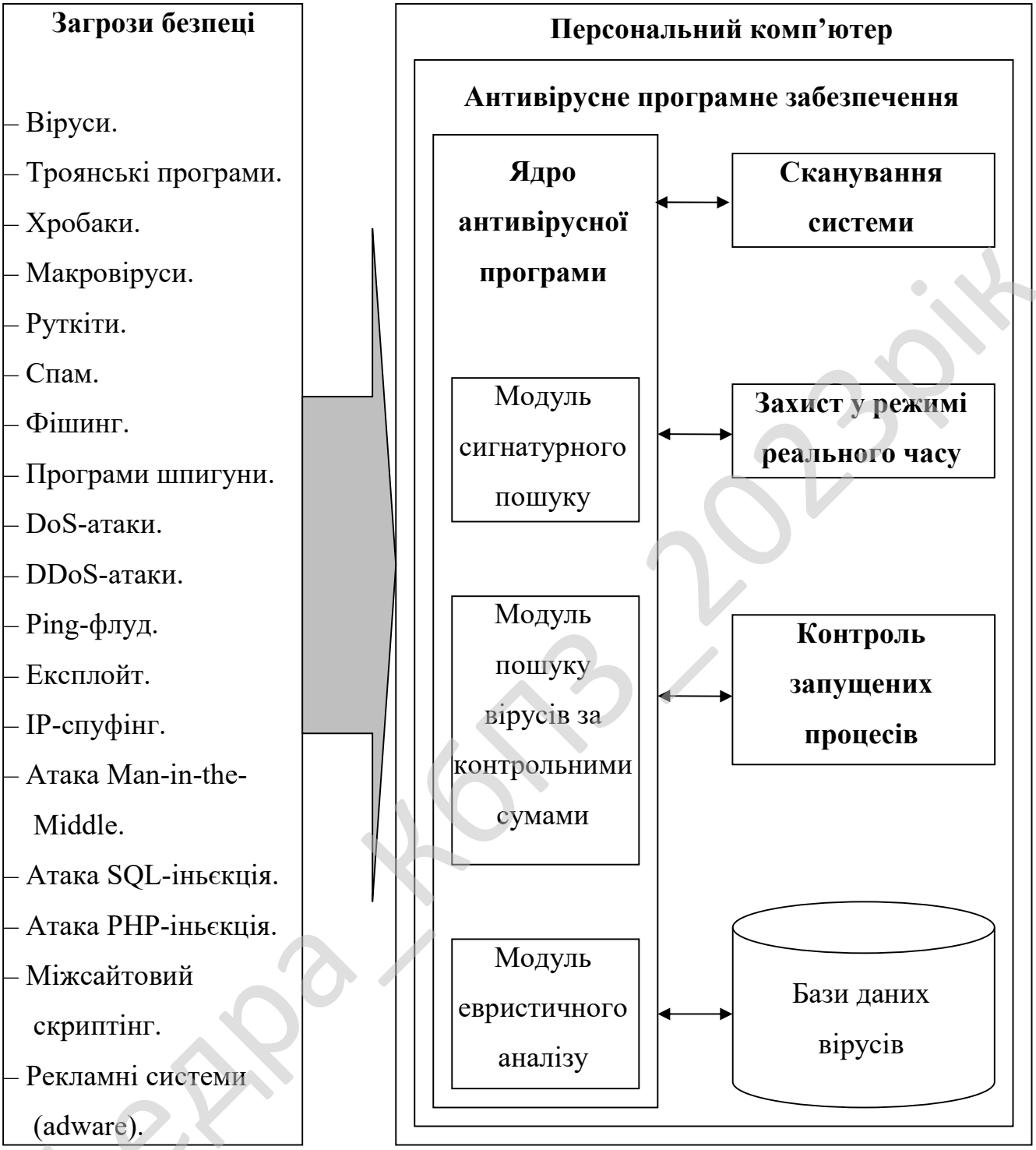

Рисунок 3.1 – Структурна схема системи

Структурний блок сканування системи складається з наступних структурних підсистем:

– Фільтр сканування.

– Швидке сканування дисків.

– Повне сканування дисків.

– Вибіркове сканування файлів.

– Лікування, знищення та переміщення у карантин інфікованих файлів.

– Звіти про результатів сканування.

Структурний блок контролю запущених процесів складається з наступних структурних 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.

Структурний блок ядра антивірусної програми складається з наступних структурних підсистем:

– Модуль евристичного аналізу.

– Модуль сигнатурного пошуку.

– Модуль пошуку вірусів за контрольними сумами.

3.3 Розробка функціональної схеми

На рисунку 3.2 зображена функціональна схема системи. Нижче розглянемо її більш докладно. Антивірус, що розроблений у результаті виконання бакалаврського проектування – захист вашого комп'ютера від шкідливих програм, що включає базові функції забезпечення безпеки вашого ПК. Антивірус використовує новітні технології захисту, завдяки якому забезпечується безпека й стабільна роботу комп'ютера. Основні функції антивірусу, що розроблений:

– Захист у режимі реального часу.

– Базовий захист при роботі в мережі Інтернет і з електронною поштою.

– Мінімальне завантаження комп'ютера.

– Інтуїтивно зрозумілий інтерфейс.

– Для повноцінного захисту комп'ютера крім антивірусу рекомендується використовувати міжмережний екран.

– Перевірка файлів, веб-сторінок, поштових і ICQ-повідомлень.

– Блокування посилань на заражені веб-сайти й сайти, що перехоплюють інформацію.

– Проактивний захист від невідомих погроз, заснований на аналізі поводження програм.

– Самозахист антивірусу, що розроблений попереджає погрозу вимикання з боку шкідливого ПЗ.

– Система миттєвого виявлення погроз, що моментально блокує нові шкідливі коди.

– Реалізовано модуль «Перевірка посилань», що попереджає про заражені або небезпечних веб-сайти.

– Проактивний захист нового покоління від невідомих погроз.

– Віртуальна клавіатура для безпечного введення логінів, паролів і номерів кредитних карт на веб-сторінках.

– Перевірка операційної системи й установлених програм на наявність уразливостей.

– Налаштування операційної системи й інтернет-браузера для безпечної роботи в мережі Інтернет.

– Відновлення працездатності системи після вірусної атаки.

– Видалення тимчасових файлів інтернет-браузера.

Рисунок 3.2 – Функціональна схема системи

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання бакалаврського проектування, наведена на рисунку 3.3. З діаграми взаємодії процесів видно, що спершу запускається процес виведення головного вікна програми. Він взаємодіє з наступними процесами:

– Процес виведення списку ресурсів.

– Процес сканування.

– Процес контролю процесів, які відбуваються у системі.

– Процес завдання параметрів антивірусної системи.

Процес виведення списку ресурсів взаємодіє з процесом вибору ресурсів для сканування.

Рисунок 3.3 – Діаграма взаємодії процесів

Процес сканування взаємодіє з наступними процесами:

– Процес зупинки процесу сканування.

– Процес призупинки процесу сканування.

– Процес запуску процесу сканування.

Останній процес взаємодіє з процесом перевірки всіх вказаних ресурсів, який, у свою чергу, взаємодіє з процесом виведення звіту.

Процес виведення звіту взаємодіє з процесом збереження звіту.

Процес контролю процесів, які відбуваються у системі взаємодіє з наступними процесами:

– Процес зупинки контролю процесів.

– Процес призупинки контролю процесів.

– Процес запуску контролю процесів, який взаємодіє з процесом перевірки запущених процесів, який, у свою чергу, взаємодіє з процесом виведення звіту.

Процес встановлення параметрів антивірусної системи взаємодіє з наступними процесами:

– Процес встановлення параметрів програми.

– Процес встановлення параметрів бази даних вірусів.

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 виводиться список ресурсів.

Наступним кроком, є обирання користувачем дії, яку він бажає здійснити:

– Сканування.

– Запуск контролю процесів.

Якщо користувач обирає сканування, тоді виконуються наступні дії:

– Обирається ресурс для сканування.

– Обирається об’єкт для перевірки.

– Запускається підпрограма сканування та лікування файлів.

Якщо знайдено вірус, тоді виконуються наступні дії:

– Виводиться інформація про вірус.

– Визначається дія, яку потрібно здійснити над інфікованим файлом.

Якщо необхідно вилікувати інфікований файл, тоді відбувається лікування файлу.

Якщо файл неможливо вилікувати, тоді видається запит видаляти файл, або ні.

Якщо видаляти, тоді файл з вірусом видаляється.

Якщо не видаляти, тоді файл з вірусом переміщується у карантин.

Після цього відбувається перевірка, сканування завершено, або ні.

Якщо завершено, тоді виводиться звіт про результати сканування.

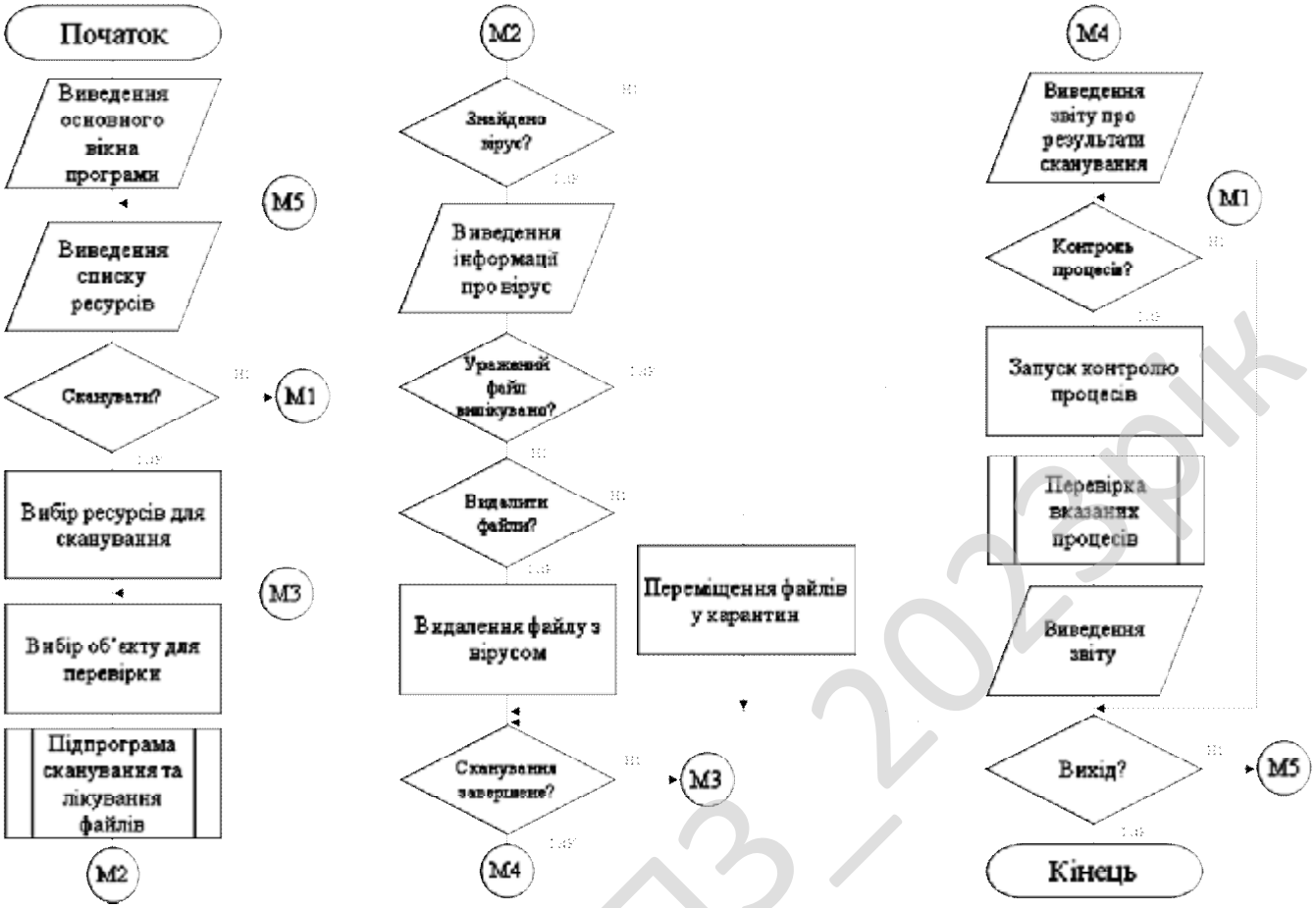

Рисунок 4.1 – Блок-схема основної програми

Якщо сканування не завершено, тоді продовжується сканування.

Якщо користувач обирає контроль процесів, тоді виконуються наступні дії:

– Запускається контроль процесів.

– Запускається підпрограма перевірки вказаних процесів.

– Відбувається виведення звіту.

Після виконання усіх вищеперерахованих дій користувач визначає працювати йому далі з антивірусною системою, або ні.

На рисунку 4.2 наведено блок-схему роботи підпрограми сканування та лікування файлів. Вона працює наступним чином.

Спершу відбувається читання розширення файлу.

Якщо це архів, то відбувається розпакування архіву у тимчасову папку,

яка видаляється після завершення сканування.

Після цього поступово виконуються наступні дії:

– Пошук вірусів за контрольними сумами.

– Сигнатурний пошук.

– Криптоаналіз.

– Евристичний аналіз.

Якщо шкідливий код знайдено, тоді відбувається визначення типу коду.

Якщо вірус відомий, то визначається, чи можливо вилікувати від вірусу.

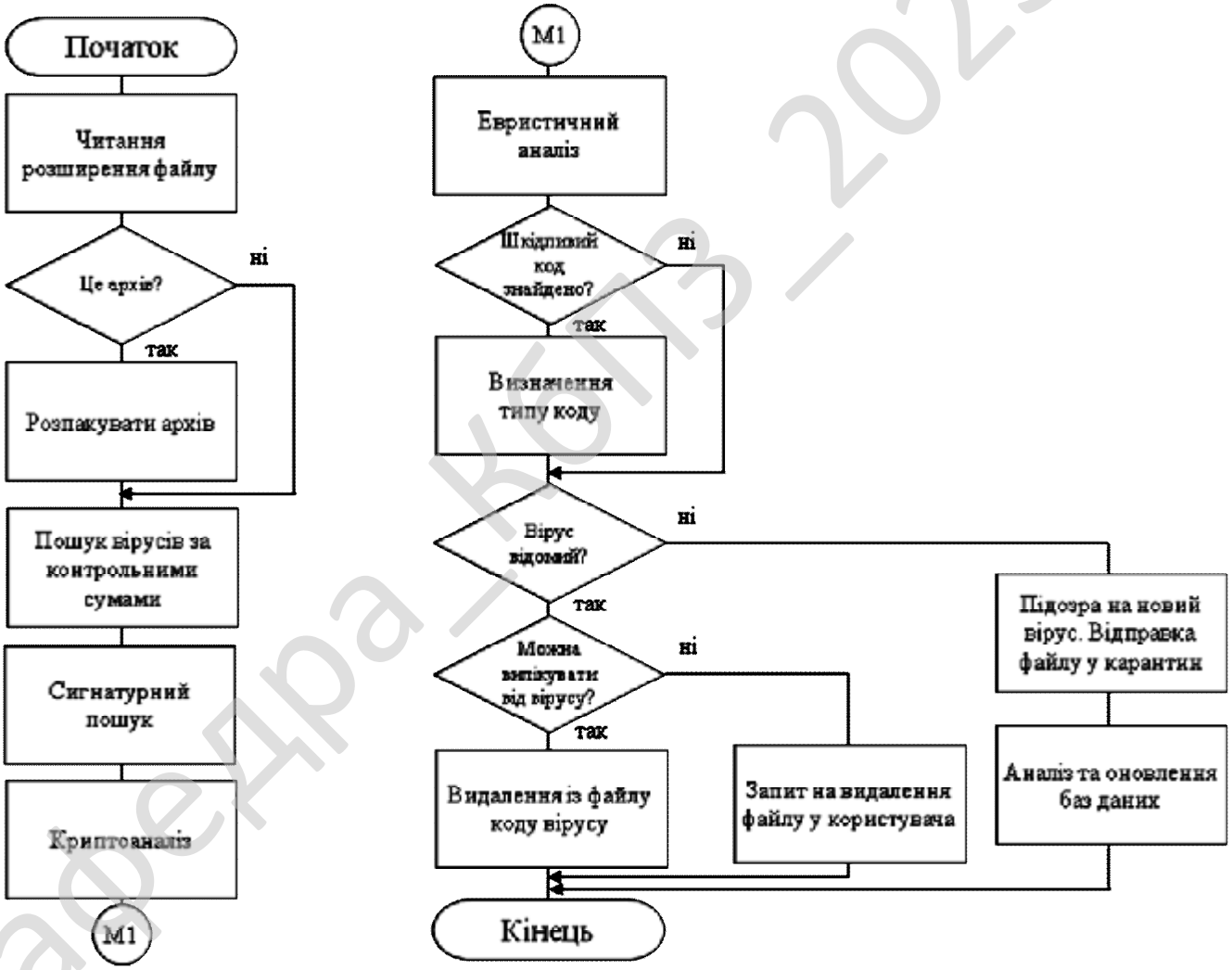

Рисунок 4.2 – Блок-схема роботи підпрограми сканування та лікування файлів

Якщо можливо, тоді відбувається видалення із файлу коду вірусу.

Якщо неможливо, то відбувається запит на видалення файлу у користувача.

Якщо вірус невідомий, тоді відбувається виконання наступних дій:

– Файл відправляється у карантин з підозрою на наявність нового вірусу.

– Відбувається аналіз файлу та оновлення баз даних.

На цьому підпрограма завершує свою роботу.

Розглянемо процес написання антивірусного сканера який буде робити сканування по MD5 хешам вірусів.

Антивірусні сигнатури будуть складатися з 2 файлів: у першому (AVBase.avb) будуть перебувати ті самі хеші, а в другому (AVNames.avb) будуть перебувати їхні імена. Причому ім'я файлу повинне перебувати на тому же рядку у файлі, що й MD5 хеш.

Алгоритм роботи сканера

Алгоритм сканера буде досить простий:

1. Пошук файлу.

2. Завантаження бази.

3. Одержання MD5 хеша файлу.

3. Порівняння хеша із записами з бази.

4. Якщо знайдено вірус – видаляємо.

Інакше – перехід до кроку 1.

Написання антивірусного сканера

Приступаємо до написання антивірусного сканера. Запускаємо Delphi 7, створюємо новий Console Application.

Отже, перше із чого ми почнемо – скачаємо й складемо в папку із проектом Kernel.dll. Цей файл буде нам необхідний для одержання MD5. Після цього напишемо от таку функцію.

Одержання MD5 файлу

```
Function GetFileMd5Hash(FileName:String): PChar;stdcall; external 'Kernel.dll';
```

Наступне: напишемо процедуру перевірки файлу на віруси й видалення знайдених погроз.

Перевірка файлу на віруси

```
procedure FindAndKill(FilePAN: String);
var
  FileMD5: String; // У цій змінній буде MD5 скануємого файлу
begin
      FileMD5 := GetFileMd5Hash(FilePAN); // Одержуємо цей MD5
      AssignFile(BaseFile,  ExtractFilePath(ParamStr(0))  +    'AVBase.avb');  //
      Завантаження бази сигнатур
      AssignFile(NameFile,  ExtractFilePath(ParamStr(0))  +    'AVNames.avb'); //
      Завантаження бази назв вірусів
      reset(BaseFile); // Перехід до читання бази сигнатур
      reset(NameFile); // Перехід до читання бази назв вірусів
      readln(BaseFile, FileBase); // Читаємо назву об'єкта бази
      readln(BaseFile, FileBase); // Читаємо назву об'єкта назв вірусів
      readln(NameFile, FileName); // Читаємо першу MD5 вірусу
      readln(NameFile, FileName); // Читаємо назву 1-го вірусу
      while FileBase <> '}' do
      begin
            if FileMD5 = FileBase then // Якщо MD5 файлу дорівнює MD5 вірусу
            begin
                  Write(' | ' + FilePan + ' | VIRUS ' + FileName); // Тоді
                  сповіщаємо про це користувачеві
                  If  DeleteFile(FilePAN)  then  writeln('  |  DELETED;')  else
                  writeln(' | OMITTED;'); // І видаляємо вірус
            end;
            readln(BaseFile, FileBase); // Читаємо наступну MD5 вірусу
            readln(NameFile, FileName); // Читаємо назву наступного вірусу
      end;
      CloseFile(BaseFile); // Закінчуємо роботу з антивірусними сигнатурами
      CloseFile(NameFile); // Закінчуємо роботу з іменами вірусів
end;
```

Повинні бути підключені наступні модулі й задекларовані наступні змінні.

Модулі й змінні

```
uses
  SysUtils, UffCrt;
var
  BaseFile: TextFile; NameFile: TextFile; // Змінні для читання файлів бази
  FileName: String; FileBase: String; // Змінні для зберігання рядків файлів бази
  Path: String; I: Integer; // И інші змінні
```

Тепер нам потрібна ще одна дуже важлива процедура: вона буде шукати файли на дисках. Пропишемо її.

Пошук файлів на диску

```
procedure Scan(Dir:String);
var
  SR:TSearchRec; FindRes: Integer; EX : String;
begin
      FindRes:=FindFirst(Dir+'*.*',faAnyFile,SR);
      While FindRes=0 do
      begin
            if       ((SR.Attr       and       faDirectory)=faDirectory)     and
            ((SR.Name='.')or(SR.Name='..')) then
            begin
                  FindRes:=FindNext(SR);
                  Continue;
            end;
            if ((SR.Attr and faDirectory)=faDirectory) then
            begin
                  Scan(Dir+SR.Name+'\');
                  FindRes:=FindNext(SR);
                  Continue;
            end;
            Ex := ExtractFileExt(Dir+SR.Name);
            if  (LowerCase(Ex) = LowerCase('.exe')) or (LowerCase(Ex) = '.com') or
            (LowerCase(Ex) = LowerCase('.dll')) then
            begin
                  FindAndKill(Dir + SR.Name);
            end;
            FindRes:=FindNext(SR);
      end;
      FindClose(SR);
end;
```

Підраховуєм кількість антивірусних сигнатур.

Підраховуєм сигнатури

```
function CountBase():Integer;
var
 CB: Integer;
begin
      CB := 0;
      if not FileExists(ExtractFilePath(ParamStr(0)) + 'AVBase.avb') then
      begin
            CountBase := 0;
      end
```

```
      else begin
            AssignFile(BaseFile, ExtractFilePath(ParamStr(0)) +  'AVBase.avb');
            reset(BaseFile);
            readln(BaseFile, FileBase);
            readln(BaseFile, FileBase);
            While FileBase <> '}' do
            begin
                  CB := CB + 1;
                  readln(BaseFile, FileBase);
            end;
            CloseFile(BaseFile);
            CountBase := CB;
      end;
end;
```

Опишемо основну процедуру консолі.

Основна процедура сканера

```
begin
      I := 0;
      WriteLn('                                    Антивірусний сканер');
      Write  (#10 + #13);
      Write  ('Завантажуємо бази...                              ');
      If CountBase = 0 then
      begin
            WriteLn('Не відкриті антивірусні бази.');
      end
      else begin
            WriteLn(IntToStr(CountBase) + ' record(s) loaded.');
            While I = 0 do
            begin
                  Write  (#10 + #13);
                  Write  ('Шлях для сканування: ');
                  Readln(Path);
                  Write  (#10 + #13);
                  if DirectoryExists(Path) = true then
                  begin
                        WriteLn('Сканую...');
                        Write(#10 + #13);
                        WriteLn('Знайдені віруси:');
                        Scan(Path);
                  end
                  else begin
```

```
                        Write  ('Не знайдена директорія');
                        Write  (#10 + #13);
                    end;
                end;
            end;
end.
```

Наведемо опис ряду функцій антивірусу

```
//***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;
//***Функція створення списку процесів***//
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;
//***Функція отримання списку процесів у системі***//
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;
  uProcess      : 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);
  uProcess.dwSize := Sizeof(uProcess);
  try
    if(hSnapShot<>0)then
    begin
      r:=Process32First(hSnapShot, uProcess);
      while r <> false do
      begin
        if  ProgCap = uProcess.szExeFile then
          KillProc:= uProcess.th32ProcessID;
        r:=Process32Next(hSnapShot, uProcess);
      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,'SeAntiVirus_ProjectPrivilege',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 := _AntiVirus_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;
//***Функція постановки процесу на паузу***//
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;
//***Функція виходу з управління процесами***//
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 := _AntiVirus_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;
```

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою Serpent – симетричний блочний алгоритм шифрування, розроблений Россом Андерсоном, Елі Біхамом та Ларсом Кнудсеном. Алгоритм був одним з фіналістів 2-го етапу конкурсу AES. Як і інші алгоритми, які брали участь у конкурсі AES, Serpent має розмір блоку 128 біт і можливі довжини ключа 128, 192 або 256 біт. Алгоритм являє собою 32-раундовий шифр на основі SP-мережі, і працює з блоком з чотирьох 32-бітових слів. Serpent був розроблений так, що всі операції можуть бути виконані паралельно, використовуючи 32-а 1-бітних «потоків».

При розробці Serpent використовувався консервативніший підхід до безпеки, ніж у інших фіналістів AES, проектувальники шифру вважали, що 16 раундів достатньо, щоб протистояти відомим видам криптоаналізу, але збільшили число раундів до 32, щоб алгоритм міг краще протистояти ще не відомим методам криптоаналізу.

Шифр Serpent не запатентований і є громадським надбанням.

Алгоритм створювався під гаслом «криптографічний алгоритм 21 століття» для участі в конкурсі AES. При створенні нового алгоритму Serpent його автори дотримувалися консервативних поглядів на проектування, що підтверджується первісним рішенням про використання таблиць підстановки з відомого багато років раніше алгоритму шифрування DES, який протягом довгого часу вивчався провідними фахівцями в області криптографії та захисту інформації і чиї властивості і особливості були добре відомі науковому світу. Одночасно з цим до нового алгоритму міг бути застосований вичерпний аналіз, вже розроблений для DES. Не використовувалися нові, неперевірені і невипробувані технології при створенні шифру, який у разі прийняття був би використаний для захисту величезних масивів фінансових транзакцій та урядової інформації. Основною вимогою до учасників конкурсу AES було те, що

алгоритм-претендент повинен бути швидшим, ніж 3DES, і надавати як мінімум такий же рівень безпеки: він повинен мати блок даних довжиною 128 біт і ключ завдовжки 256 біт. 16-раундовий Serpent був би таким же надійним, як 3DES, але в два рази швидшим. Однак, автори вирішили, що для більшої надійності варто збільшити кількість раундів до 32. Це зробило шифр таким же швидким, як DES, і набагато надійнішим, ніж 3DES.

Структура алгоритму

Алгоритм Serpent є SP-мережею, у котрій весь блок даних довжиною 128 біт на кожному раунді розбивається на 4 слова довжиною 32 біти. Всі значення, що використовуються при шифруванні є бітовими потоками. Бітові індекси змінюють значення від 0 до 31 для 32-бітових слів, від 0 до 127 – для 128-бітових блоків та від 0 до 255 для 256-бітових ключів тощо. Для внутрішніх обчислень всі біти величин представлені в прямому порядку (little-endian).

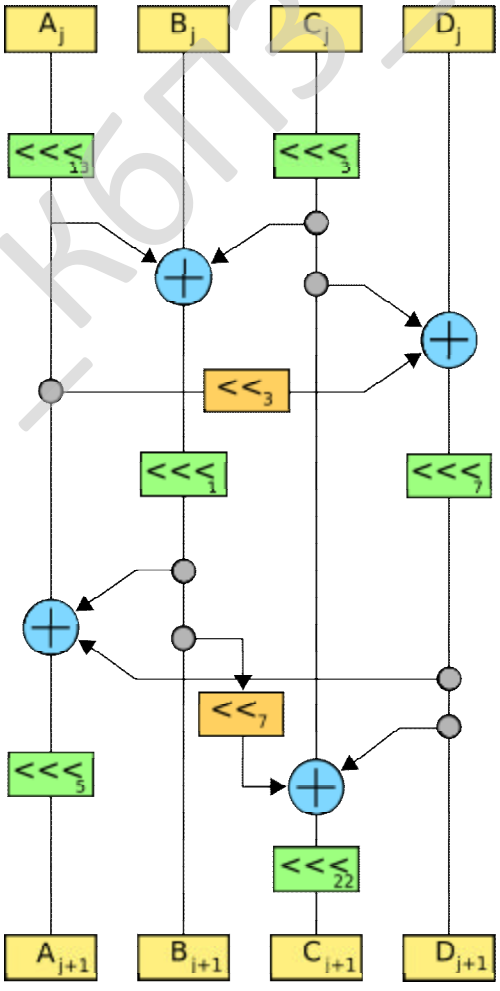

Рисунок 4.3 – Структура алгоритму Serpent

Serpent шифрує відкритий текст P довжиною 128 біт в шифротекст C довжиною таких же 128 біт за 32 раунд за допомогою 33 подключів {\displaystyle K_{0},...,K_{32}\,}{\displaystyle K_{0},...,K_{32}\,} довжиною 128 біт. Довжина використовуваного ключа може приймати різні значення, але для конкретики зафіксуємо їх довжину в 128, 192 або 256 біт. Короткі ключі довжиною менше 256 біт доповнюються до повної довжини в 256 біт.

Шифрування складається з наступних основних кроків:

– Початкова перестановка.

– 32 раунд, кожен з яких складається з операції змішування з 128-бітовим ключем (побітове логічне виключаюче «або»), таблична заміна (S-box) і лінійне перетворення. В останньому раунді лінійне перетворення замінюється додатковим накладанням ключа.

– Кінцева перестановка.

Початкова і кінцева перестановки не мають будь-якої криптографічного значущості. Вони використовуються для спрощення оптимізованої реалізації алгоритму і підвищення обчислювальної ефективності.

Розширення ключа

Як і інші алгоритми, що брали участь в конкурсі AES, Serpent має можливі довжини ключа 128, 192 або 256 біт. «Неповний» ключ довжиною менше 256 біт доповнюється за наступним правилом: додається одиничний біт справа, за ним слід стільки нульових бітів, щоб довжина ключа стала дорівнює 256 бітам.

Початкова перестановка IP

Дана перестановка IP задається таблицею, де вказується позиція, на яку перейде відповідний біт (наприклад, біт 1 перейде на 32 позицію):

S-бокси (таблиці замін)

В алгоритмі Serpent таблиці замін є 4-бітовими перестановками з властивостями стійкості до диференціального криптоаналізу, до лінійного криптоаналізу і такою властивістю, що порядок вихідних біт, як функції вхідних повинен бути максимальний, тобто бути рівним 3.

Таблиця підстановки генерується з відомих і добре вивчених таблиць для алгоритму DES в ітераційному процесі, поки не будуть отримані бажані диференціальні й лінійні властивості. Таким чином, створюється 8 таблиць підстановки.

Лінійне перетворення LT

Лінійне перетворення LTзадається таблицею, де біти перераховані від 0 до 127 (наприклад, вихідний 2 біт утворений 2, 9, 15, 30, 76, 84, 126 битами, складеними за модулем 2) . В кожному рядку описується 4 вихідних біти, які разом складають вхідні дані на одну таблицю замін в наступному раунді. Варто зазначити, що даний набір являє собою таблицю IP(LT(FP(x))), де LT і є те лінійне перетворення.

Таблиця зворотного лінійного перетворення, яке використовується при розшифровці ILT.

Кінцева перестановка FP

Дана перестановка є зворотною до початкової, тобто $FP=IP^{-1}$ і задається наступною таблицею.

Ефективна реалізація алгоритму

Бажання авторів зробити алгоритм саме таким, яким він є стає зрозумілим при розгляді його ефективної низькорівневої реалізації.

Serpent був створений таким чином, щоб всі операції в процесі шифрування і розшифрування одного блоку могли бути виконані паралельно в 32 потоках. До того ж низькорівневий опис алгоритму набагато простішій, ніж стандартний опис. Ніяких початкових і кінцевих перестановок не потрібно.

Шифрування складається з 32 раундів. Відкритий текст є першими проміжними даними $B_0 = P$. Потім виконується 32 раунди, кожен і-й раунд складається з:

– Змішування з ключем. Проводиться побітове виключаюче «або» проміжних даних B_i з ключем довжиною 128 біт.

– Застосування таблиць підстановки. Вхідні дані довжиною 128 біт поділяються на 4 слова по 32 біта. Таблиця підстановки, реалізована послідовністю логічних операцій (як якщо це було б реалізовано апаратно), застосовується до цих 4 словам. В результаті виходить 4 вихідних слова. Таким чином, центральний процесор виконує підстановку по 32 копій таблиці одночасно.

– Лінійне перетворення. 32-бітові слова перетворюються заданим порядком.

Першою причиною вибору такого лінійного перетворення є максимізація лавинного ефекту. Такі таблиці підстановки мають властивість, що зміна кожного вхідного біта призведе до зміни 2 вихідних бітів. Таким чином, кожен вхідний біт відкритого тексту вже через 3 раунди впливає на всі вихідні біти. Аналогічно кожен біт ключа впливає на результат шифрування.

Друга причина полягає в простоті перетворення. Воно може бути реалізоване на будь-якому сучасному процесорі з мінімальними витратами.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено головне вікно програми. Воно складається з наступних блоків:

– Блок меню.

– Блок кнопок швидкого доступу до елементів програми.

– Вікно вибору об’єктів сканування.

Блок меню складається з наступних елементів.

– Файл.

– Сканування.

– Контроль процесів.

– Параметри.

– Довідка.

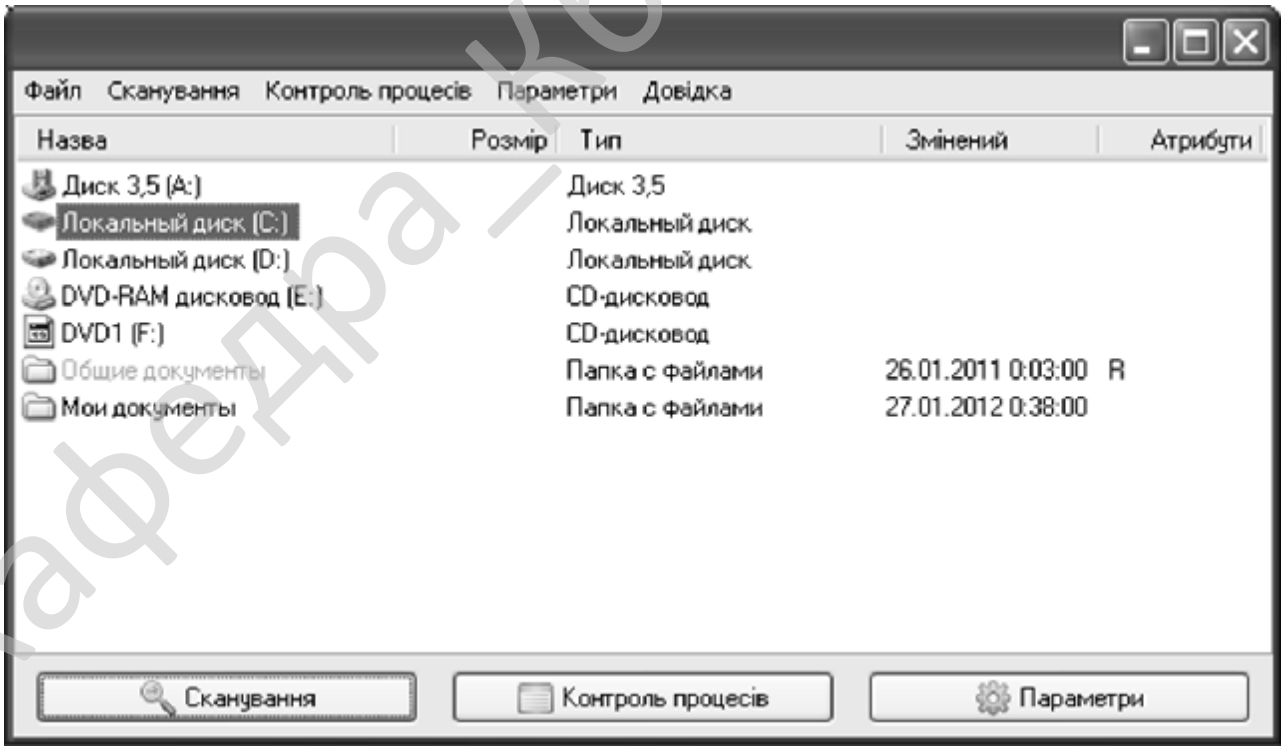

Рисунок 5.1 – Головне вікно програми

Блок кнопок швидкого доступу до елементів програми складається з наступних елементів:

– Сканування.

– Контроль процесів.

Параметри.

На рисунку 5.2 зображена довідка, яка надає наступну інформацію:

– Тема бакалаврського проекту.

– Розробник бакалаврського проекту.

– Керівник бакалаврського проекту.

– Місце виконання бакалаврського проекту.

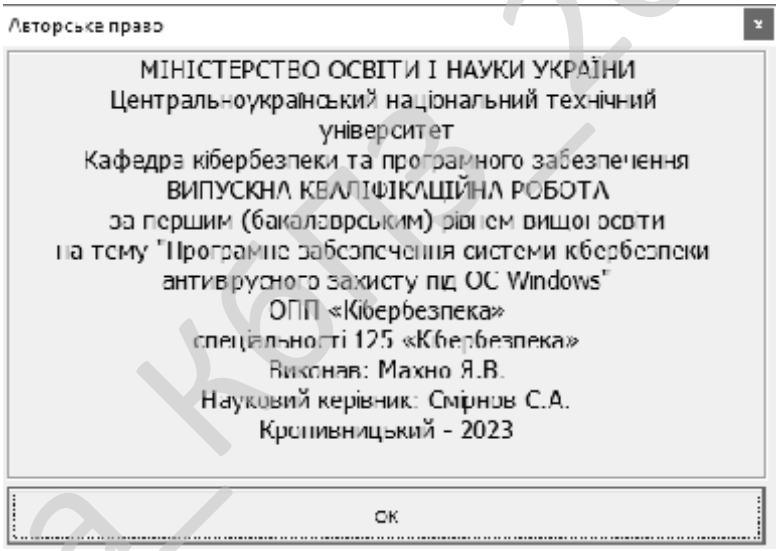

Рисунок 5.2 – Довідка

6 ОСНОВНІ ВИСНОВКИ

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

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем антивірусного захисту під ОС Windows.

– Досліджена система антивірусного захисту під ОС Windows.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки антивірусного захисту під ОС Windows.

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

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 антивірусного захисту під ОС Windows. 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 Serpent.

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. Моделирование защищенного канала связи с использованием экспоненциальной GERT-сети / С.Г. Семенов, А.А. Можаев // Информатика, математическое моделирование, экономика. – Смоленськ.: Смоленский филиал АНО ВПО ЦС РФ "Российский университет кооперации". – 2012. – Том.1. – С. 152-160.

2. 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.

3. 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.

4. 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.

5. 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, 2022.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. 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.

12. 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.

13. 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.

14. 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.

15. 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.

16. 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.

17. 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.

18. 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.

19. 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.

20. 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.

21. 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.

22. 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.

23. 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.

24. 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.

25. 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.

26. 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.

27. 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.

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

35. 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.

36. 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.

37. 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.

38. 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.

39. 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.

40. 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.

41. 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.

42. 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.

43. 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.

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

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

46. 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.

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

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

49. 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.

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

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки антивірусного захисту під ОС Windows.

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. № 13-02 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 антивірусного захисту під ОС Windows.

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 антивірусного захисту під ОС Windows;

– 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 – 71 аркуш.

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

_____ Смірнов С.А.

Програмне забезпечення системи кібербезпеки антивірусного захисту під ОС Windows

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 42

Літера: РП

Кропивницький – 2023 року

Файл AntiVirus_Monitor.pas – монітор (контроль процесів)

```
unit AntiVirus_Monitor;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, ExtCtrls, AntiVirus_Kernel, AntiVirus_Types,
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;
    PCAntiVirus_Scaned: 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 AntiVirus_Main, AntiVirus_InfectedAction, AntiVirus_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;
  uProcess      : 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);
  uProcess.dwSize := Sizeof(uProcess);

  try
    if(hSnapShot<>0)then
    begin
      r:=Process32First(hSnapShot, uProcess);
      while r <> false do
      begin
        if  ProgCap = uProcess.szExeFile then
          KillProc:= uProcess.th32ProcessID;
        r:=Process32Next(hSnapShot, uProcess);
      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,'SeAntiVirus_ProjectPrivilege',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 := _AntiVirus_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 := _AntiVirus_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 AntiVirus_Project.dpr – головний файл проекту

```
program AntiVirus_Project;
// Список підключаємих модулів
uses
  Forms,
  SysUtils,
  AntiVirus_Kernel in '..\AntiVirus Virus_AntiVirus_Scanner
Modues\AntiVirus_Kernel.pas',
  AntiVirus_Types in '..\AntiVirus Virus_AntiVirus_Scanner
Modues\AntiVirus_Types.pas',
  avMonitor in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avMonitor.pas',
  avVirus_AntiVirus_Scanner in '..\AntiVirus Virus_AntiVirus_Scanner
Modues\avVirus_AntiVirus_Scanner.pas',
  avHex in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avHex.pas',
  avDataBase in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avDataBase.pas',
  avHash in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avHash.pas',
  avExt in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avExt.pas',
  avAPI in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avAPI.pas',
  avConfig in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avConfig.pas',
  avShield in '..\AntiVirus Virus_AntiVirus_Scanner Modues\avShield.pas',
  langs in 'langs.pas',
  AntiVirus_Main in 'AntiVirus_Main.pas' {MainForm},
  uSelInfo in 'uSelInfo.pas' {InformationForm},
  AntiVirus_Options in 'AntiVirus_Options.pas' {OptionsForm},
  uPluginInfo in 'uPluginInfo.pas' {PluginAPIForm},
  AntiVirus_AddPath in 'AntiVirus_AddPath.pas' {AddUserPathForm},
  AntiVirus_About in 'AntiVirus_About.pas' {AboutForm},
  uSelDir in 'uSelDir.pas' {SelDirFrm},
  uMessage in 'uMessage.pas' {MessageFrm},
  uHideForm in 'uHideForm.pas' {HideForm},
  AntiVirus_Monitor in 'AntiVirus_Monitor.pas' {MonitorForm},
  AntiVirus_InfectedAction in 'AntiVirus_InfectedAction.pas' {ActionForm},
  uSplash in 'uSplash.pas' {SplashForm};

{$R *.res}

begin
  Application.Initialize;
  Application.Title := 'Virus_AntiVirus_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.InitVirus_AntiVirus_ScannerKernel;
  {Hide Splash Form}
  SplashForm.Hide;
  Sleep(200);
  {Create Tray Icon}
  MainForm.CreateTray;
  {}
  if OptionsForm.AUTORUN.Checked then begin
```

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

Файл AntiVirus_Main.pas – основна програма

```
unit AntiVirus_Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, Menus, ImgList, XPMan,
AntiVirus_Kernel, AntiVirus_Types, 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;
   uID: UINT;
   uFlags: UINT;
   uCallbackMessage: UINT;
   hIcon: HICON;
   szTip: array[0..MAXCHAR] of AnsiChar;
   dwState: DWORD;
   dwStateMask: DWORD;
   szInfo: array[0..MAXBYTE] of AnsiChar;
   uTimeout: UINT; // union with uVersion: 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;
    AntiVirus_ScanPathesTab: TTabSheet;
    AntiVirus_ScanningTab: TTabSheet;
    ReportTab: TTabSheet;
    BottomPanel: TPanel;
    AntiVirus_ScanBTN: TButton;
    SaveBTN: TButton;
    PathList: TListView;
    Bevel1: TBevel;
    AntiVirus_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;
mnuShowAntiVirusVirus_AntiVirus_Scanner: TMenuItem;
mnuHideAntiVirusVirus_AntiVirus_Scanner: TMenuItem;
N2: TMenuItem;
mnAntiVirus_Options: 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;
AntiVirus_ScanTopBtn: TLabel;
AntiVirus_ScanMenu: TPopupMenu;
mnuSelAntiVirus_ScanPath: TMenuItem;
mnuShowReport: TMenuItem;
N12: TMenuItem;
OptionTopBtn: TLabel;
PCTopBtn: TLabel;
mnuAntiVirusProcessControl: TMenuItem;
N19: TMenuItem;
mnuPCShow: TMenuItem;
N21: TMenuItem;
mnuPCRun: TMenuItem;
mnuPCPause: TMenuItem;
mnuPCStop: TMenuItem;
mnuAntiVirus_ScanStart: TMenuItem;
mnuStopAntiVirus_Scan: TMenuItem;
N13: TMenuItem;
mnuSaveReport: TMenuItem;
N26: TMenuItem;
mnuGoToTray: TMenuItem;
SOURCESTRING: TListBox;
LabelPanel: TPanel;
AntiVirus_ScanFile: TLabel;
procedure DelAllClick(Sender: TObject);
procedure FormResize(Sender: TObject);
procedure ExitBTNClick(Sender: TObject);
procedure AntiVirus_ScanListDblClick(Sender: TObject);
procedure AntiVirus_ScanBTNClick(Sender: TObject);
procedure InitVirus_AntiVirus_ScannerKernel;
Procedure StartAntiVirus_Scan(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 mnuHideAntiVirusVirus_AntiVirus_ScannerClick(Sender: TObject);
    procedure mnuShowAntiVirusVirus_AntiVirus_ScannerClick(Sender: TObject);
    procedure mnuExitClick(Sender: TObject);
    procedure mnAntiVirus_OptionsClick(Sender: TObject);
    procedure mnuHelpClick(Sender: TObject);
    procedure mnuAboutClick(Sender: TObject);
    procedure ApplicationEventsMinimize(Sender: TObject);
    procedure AppMinimize(Sender: TObject);
    procedure FormPaint(Sender: TObject);
    procedure AntiVirus_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 AntiVirus_ScanTopBtnClick(Sender: TObject);
    procedure mnuShowReportClick(Sender: TObject);
    procedure mnuSelAntiVirus_ScanPathClick(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 AntiVirus_ScanMenuPopup(Sender: TObject);
    procedure mnuAntiVirus_ScanStartClick(Sender: TObject);
    procedure mnuStopAntiVirus_ScanClick(Sender: TObject);
    procedure mnuSaveReportClick(Sender: TObject);
    procedure CopyRightLabelClick(Sender: TObject);
    Procedure CreateTray;
  protected
    procedure MineRestore(var Msg: TMessage); message WM_MINERESTORE;
    procedure SendAntiVirus_Scanning(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;

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

        AntiVirusMonitor      : String;
        AntiVirusInit         : String;
        LoadAPI          : String;
        LoadDB           : String;
        CreateDrvList    : String;
        OptFileNotFnd    : String;
        LoadOptFile      : String;
        InitProcedures   : String;
        initShield       : String;
        ErrorInit        : String;
        LogBevel         : String;
        DBKnowledge      : String;
```

```
        SCNOBJ          : String;
        AntiVirus_ScanExecute     : String;
        AntiVirus_ScanEnd         : String;
        PrepareToAntiVirus_Scan   : String;
        FileIgnor       : String;
        FileIfect       : String;
        FileAntiVirus_Scanned     : String;
        DataAntiVirus_Scanned     : String;
        IGNORED         : String;
        SKIPBYSIZE      : String;
        INFECTED        : String;
        STOPB           : String;
        RETURNB         : String;
        ANTIVIRUS_SCANB           : String;
        SCNFILE         : String;
        FileDel         : String;
        FileNotDel      : String;
        PATHNOSEL       : String;
        SysMenu         : String;
        NfoAntiVirusVirus_AntiVirus_Scanner   : String;
        NfoAntiVirusKernel    : String;
        NfoAntiVirusBuild     : String;
        DelDialog       : String;
        DelAllDialog    : String;
        DelError        : String;
        HelpNOFound     : String;
        avShieldMes     : String;
        avError         : String;
        DelResult       : String;
        AllInfected     : String;
        DeleteInfected  : String;
        SkippedInfected : String;
        AntiVirusCloseDlg     : String;
        AllreadyInAntiVirus_Scan  : 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;
    AntiVirusVirus_AntiVirus_ScannerCapt  = 'Антивірусний захист операційної
системи від шкідливих програм';
    AntiVirusVirus_AntiVirus_ScannerVS    = '';
var
    MainForm      : TMainForm;
    inAntiVirus_Scan        : Boolean = False;
    NeedToReturn  : Boolean = False;
    FirstRun      : Boolean = True;
    P             : TPoint;
    MayClose      : boolean=false;
```

```
implementation

uses uSelInfo, AntiVirus_Options, AntiVirus_AddPath, AntiVirus_About, Math,
uMessage, uHideForm,
  AntiVirus_Monitor, AntiVirus_InfectedAction, uPluginInfo;
{$R *.dfm}

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

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

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

procedure TMainForm.SendAntiVirus_Scanning;
var
  pcd: PCopyDataStruct;
begin
  pcd := PCopyDataStruct(Msg.LParam);
  if not inAntiVirus_Scan then
  begin
    StartAntiVirus_Scan(PChar(pcd.lpData));
  end
  else begin
    MessageDlg(AllreadyInAntiVirus_Scan,mtError,[mbOK],0);
  end;
end;

procedure TMainForm.MineRestore(var Msg: TMessage);
begin
  if (Msg.Msg = WM_MINERESTORE) then
  begin
    mnuShowAntiVirusVirus_AntiVirus_Scanner.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;
    uID := IconID;
    uFlags := NIF_INFO;
    StrPCopy(szInfo, BalloonText);
    uTimeout := 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;
    uID := 1;
    uFlags := NIF_ICON or NIF_MESSAGE or NIF_TIP;
    uCallBackMessage := WM_NOTIFYTRAYICON;
    hIcon :=  Application.Icon.Handle;
    szTip := 'AntiVirus Virus_AntiVirus_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;
    uID := 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_ANTIVIRUS_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_ANTIVIRUS_SCAN_FILE;

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

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

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

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

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

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

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

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

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

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

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

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

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

      if GETParamName(AntiVirusConfig[i]) = 'AUTOSAVEREPORTTO' then
OptionsForm.ReportSavePath.Text := GETParam(AntiVirusConfig[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.AntiVirus_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.AntiVirus_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 OnAntiVirus_ScanComplete;
  var
  AntiVirus_ScanEndBalloonText: String;
  i: integer;
begin
  MainForm.ProgressBar.Max := 1;
  MainForm.ProgressBar.Position := MainForm.ProgressBar.Max;
  MainForm.AntiVirus_ScanBTN.Caption := MainForm.RETURNB;
  NeedToReturn := True;
  inAntiVirus_Scan := 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.AntiVirus_ScanFile.caption  := MainForm.AntiVirus_ScanEnd;
  OnAddToLogStr('',-1);
  OnAddToLogStr(MainForm.AntiVirus_ScanEnd,0);
  OnAddToLogStr('',-1);
  OnAddToLogStr(MainForm.FileAntiVirus_Scanned+inttostr(MainForm.FileCN),0);
  OnAddToLogStr(MainForm.FileIgnor+inttostr(MainForm.FileIgnored),0);
  OnAddToLogStr(MainForm.FileIfect+inttostr(MainForm.FileInfected),0);

OnAddToLogStr(MainForm.DataAntiVirus_Scanned+Format('%.2f',[AntiVirus_ScannedDat
aSize / 1024 / 1024])+' Mb',0);
  MainForm.ReportMemo.Lines.Add(MainForm.LogBevel);
  if OptionsForm.AutoSaveReport.Checked then begin
    MainForm.ReportMemo.Lines.SaveToFile(OptionsForm.ReportSavePath.Text);
  end;

  AntiVirus_ScanEndBalloonText := MainForm.AntiVirus_ScanEnd + ':' + Return +
Return
                        +' >>
'+MainForm.FileAntiVirus_Scanned+inttostr(MainForm.FileCN) + Return
                        +' >> '+MainForm.FileIgnor+inttostr(MainForm.FileIgnored)
+ Return
                        +' >>
'+MainForm.FileIfect+inttostr(MainForm.FileInfected) + Return
                        +' >>
'+MainForm.DataAntiVirus_Scanned+Format('%.2f',[AntiVirus_ScannedDataSize / 1024
/ 1024])+' Mb';

  MainForm.BalloonTrayIcon(MainForm.Handle
,1,10,AntiVirus_ScanEndBalloonText,'AntiVirus Virus_AntiVirus_Scanner',bitInfo);
end;

//***Функція початку сканування***//

Procedure OnAntiVirus_ScanStart;
  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.AntiVirus_ScanBTN.Caption := MainForm.STOPB;
  MainForm.SaveBTN.Enabled := False;
```

```
  MainForm.AntiVirus_ScanList.Clear;
  MainForm.AntiVirus_ScanningTab.Show;
  MainForm.FileCN       := 0;
  MainForm.FileInfected := 0;
  MainForm.FileIgnored  := 0;
  inAntiVirus_Scan := True;
  NeedToReturn := False;
  OnAddToLogStr(MainForm.AntiVirus_ScanExecute,0);
  if AntiVirusVirus_AntiVirus_Scanner.AvAction = TAntiVirus_ScanDir then
  else
  OnAddToLogStr(MainForm.SCNOBJ+AntiVirusVirus_AntiVirus_Scanner.FileName,0);
  OnAddToLogStr('',-1);
  MainForm.BalloonTrayIcon(MainForm.Handle
,1,10,MainForm.AntiVirus_ScanExecute,'AntiVirus
Virus_AntiVirus_Scanner',bitInfo);
  AntiVirusVirus_AntiVirus_Scanner.Resume;
end;

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

Procedure AntiVirusKernelMessageAPI(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.AntiVirus_ScanBTN.Enabled := False;
  end;

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

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

  if MES = MES_PREPARINGTOANTIVIRUS_SCAN then
MainForm.AntiVirus_ScanFile.Caption := MainForm.PrepareToAntiVirus_Scan;

  if mes = MES_INITKERNEL then OnAddToLogStr(MainForm.AntiVirusInit,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_ONANTIVIRUS_SCANEXECUTE then
    OnAntiVirus_ScanStart;

  if MES = MES_ONANTIVIRUS_SCANCOMPLETE then
    OnAntiVirus_ScanComplete;

  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.AntiVirus_ScanFile.caption  := '['+inttostr(MainForm.FileCN)+']
'+ExtractFileName(Pr_1);
    end
    else
      MainForm.AntiVirus_ScanFile.caption  := ExtractFileName(Pr_1);
    if OPT_SEND_ANTIVIRUS_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.InitVirus_AntiVirus_ScannerKernel;
  var
  i:integer;
begin

//******//
        AntiVirusMonitor      := SOURCESTRING.Items[0];
        AntiVirusInit         := 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];
        AntiVirus_ScanExecute     := SOURCESTRING.Items[13];
        AntiVirus_ScanEnd         := SOURCESTRING.Items[14];
        PrepareToAntiVirus_Scan   := SOURCESTRING.Items[15];
        FileIgnor       := SOURCESTRING.Items[16];
        FileIfect       := SOURCESTRING.Items[17];
        FileAntiVirus_Scanned     := SOURCESTRING.Items[18];
        DataAntiVirus_Scanned     := SOURCESTRING.Items[19];
        IGNORED         := SOURCESTRING.Items[20];
        SKIPBYSIZE      := SOURCESTRING.Items[21];
        INFECTED        := SOURCESTRING.Items[22];
        STOPB           := SOURCESTRING.Items[23];
        RETURNB         := SOURCESTRING.Items[24];
        ANTIVIRUS_SCANB           := SOURCESTRING.Items[25];
        SCNFILE         := SOURCESTRING.Items[26];
        FileDel         := SOURCESTRING.Items[27];
        FileNotDel      := SOURCESTRING.Items[28];
        PATHNOSEL       := SOURCESTRING.Items[29];
        SysMenu         := SOURCESTRING.Items[30];
        NfoAntiVirusVirus_AntiVirus_Scanner   := SOURCESTRING.Items[31];
        NfoAntiVirusKernel    := SOURCESTRING.Items[32];
        NfoAntiVirusBuild     := 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];
        AntiVirusCloseDlg     := SOURCESTRING.Items[44];
        AllreadyInAntiVirus_Scan  := 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(AntiVirusKernelMessageAPI);
        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)+'
'+NfoAntiVirusVirus_AntiVirus_Scanner +AntiVirusVirus_AntiVirus_ScannerVS);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoAntiVirusKernel
+GetKernelVersion);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoAntiVirusBuild
+GetKernelBuild);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+DBKnowledge+IntToStr(GetDBRecCount));

  if GetDBVersionDate = '01.01.1880' 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('*','AntiVirus.AntiVirus_Scan',SysMenu,ParamStr(0)+'
%1');

OptionsForm.FileTAddAction('Directory','AntiVirus.AntiVirus_Scan',SysMenu,ParamS
tr(0)+' %1');

OptionsForm.FileTAddAction('Drive','AntiVirus.AntiVirus_Scan',SysMenu,ParamStr(0
)+' %1');
  end else
  begin
    OptionsForm.FileTDelAction('Drive','AntiVirus.AntiVirus_Scan');
    OptionsForm.FileTDelAction('Directory','AntiVirus.AntiVirus_Scan');
    OptionsForm.FileTDelAction('*','AntiVirus.AntiVirus_Scan');
  end;
end;

//***Функція початку сканування***//

Procedure TMainForm.StartAntiVirus_Scan(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
    AntiVirusVirus_AntiVirus_Scanner := TAvVirus_AntiVirus_Scanner.Create(true);
    AntiVirusVirus_AntiVirus_Scanner.NeedForAPI := TRUE;
    AntiVirusVirus_AntiVirus_Scanner.AvAction   := TAntiVirus_ScanDir;
    Path.Add(ExtractFileDrive(Paramstr(0))+'\');
    AntiVirusVirus_AntiVirus_Scanner.Dirs       := Path;
    OnAntiVirus_ScanStart;
    exit;
  end;

  if DirectoryExists(Parametr+'\') then
```

```
  begin
    AntiVirusVirus_AntiVirus_Scanner := TAvVirus_AntiVirus_Scanner.Create(true);
    AntiVirusVirus_AntiVirus_Scanner.NeedForAPI := TRUE;
    AntiVirusVirus_AntiVirus_Scanner.AvAction   := TAntiVirus_ScanDir;
    Path.Add(Parametr+'\');
    AntiVirusVirus_AntiVirus_Scanner.Dirs       := Path;
    OnAntiVirus_ScanStart;
    exit;
  end;

  if FileExists(Parametr) then
  begin
    AntiVirusVirus_AntiVirus_Scanner := TAvVirus_AntiVirus_Scanner.Create(true);
    AntiVirusVirus_AntiVirus_Scanner.NeedForAPI := false;
    AntiVirusVirus_AntiVirus_Scanner.AvAction := TAntiVirus_ScanFile;
    AntiVirusVirus_AntiVirus_Scanner.FileName := Parametr;
    OnAntiVirus_ScanStart;
    exit;
  end;

end;

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

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

procedure TMainForm.AntiVirus_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 inAntiVirus_Scan = False then
```

```
    begin
      if PATH.Count-1 <> -1 then
      begin
        AntiVirusVirus_AntiVirus_Scanner :=
TAvVirus_AntiVirus_Scanner.Create(true);
        AntiVirusVirus_AntiVirus_Scanner.FreeOnTerminate := True;
        AntiVirusVirus_AntiVirus_Scanner.NeedForAPI      := true;
        AntiVirusVirus_AntiVirus_Scanner.AvAction        := TAntiVirus_ScanDir;
        AntiVirusVirus_AntiVirus_Scanner.Dirs            := MainForm.Path;
        OnAntiVirus_ScanStart;
      end
      else begin
        MessageDlg(PATHNOSEL,mtError,[mbOk],0);
      end;
    end
    else begin
      CloseAntiVirus_ScanThread;
    end;
  end else
  begin
    AntiVirus_ScanBTN.Caption := AntiVirus_ScanB;
    MainForm.SaveBTN.Enabled := False;
    NeedToReturn := False;
    AntiVirus_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 AntiVirus_ScanList.Items.Count-1 do
      Report.Add(AntiVirus_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 := AntiVirusVirus_AntiVirus_ScannerVS;
end;

procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  if MessageDlg(AntiVirusCloseDlg,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 (AntiVirus_ScanList.ItemIndex <> -1) and
(AntiVirus_ScanList.Selected.ImageIndex = 2) and (inAntiVirus_Scan = False) then
  begin
    Del.Visible := true;
  end
  else
    Del.Visible := False;

  if (AntiVirus_ScanList.ItemIndex <> -1) and (inAntiVirus_Scan = 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 AntiVirus_ScanList.Items.Count - 1 do
        if AntiVirus_ScanList.Items.Item[i].ImageIndex = 2 then
      begin
            c:=c+1;
                    try
                       if
DeleteFileBC(AntiVirus_ScanList.Items.Item[i].SubItems[0]) then
          begin
            d:=d+1;
                       AntiVirus_ScanList.Items.Item[i].ImageIndex := 4;

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

      ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileNotDel+AntiVirus
_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(AntiVirus_ScanList.Selected.SubItems[0]) then
    begin
      AntiVirus_ScanList.Selected.ImageIndex := 4;

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

ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+FileNotDel+AntiVirus_ScanL
ist.Selected.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                       := AntiVirusVirus_AntiVirus_ScannerCapt;
  TopPn.DoubleBuffered                  := true;
  TopRightPanel.DoubleBuffered     := true;
  PathList.DoubleBuffered             := true;
  AntiVirus_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;
  AntiVirus_ScanList.Columns.Items[0].Width := AntiVirus_ScanList.Width - 25;
end;

procedure TMainForm.mnuHideAntiVirusVirus_AntiVirus_ScannerClick(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.mnuShowAntiVirusVirus_AntiVirus_ScannerClick(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.mnAntiVirus_OptionsClick(Sender: TObject);
begin
  if not inAntiVirus_Scan 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
  mnuHideAntiVirusVirus_AntiVirus_Scanner.Click;
end;

procedure TMainForm.FormPaint(Sender: TObject);
```

```
begin
  if FirstRun then
    if OptionsForm.AUTOHIDE.Checked then
    begin
      mnuHideAntiVirusVirus_AntiVirus_Scanner.Click;
    end;
  FirstRun := false;
end;

procedure TMainForm.AntiVirus_ScanListCustomDrawItem(Sender: TCustomListView;
  Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
begin
  with AntiVirus_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.AntiVirus_ScanTopBtnClick(Sender: TObject);
begin

AntiVirus_ScanMenu.Popup(MainForm.Left+AntiVirus_ScanTopBtn.Left+3,MainForm.Top+
AntiVirus_ScanTopBtn.Top+38);
end;

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

procedure TMainForm.mnuSelAntiVirus_ScanPathClick(Sender: TObject);
begin
  if not inAntiVirus_Scan then
    AntiVirus_ScanPathesTab.Show;
end;

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

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

procedure TMainForm.mnuGoToTrayClick(Sender: TObject);
begin
  mnuHideAntiVirusVirus_AntiVirus_Scanner.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 inAntiVirus_Scan then mnAntiVirus_Options.Enabled := False else
mnAntiVirus_Options.Enabled := True;
end;

procedure TMainForm.AntiVirus_ScanMenuPopup(Sender: TObject);
begin
  mnuPCRun.Enabled := MonitorForm.StartPC.Enabled;
  mnuPCPause.Enabled := MonitorForm.PausePC.Enabled;
  mnuPCStop.Enabled := MonitorForm.StopPC.Enabled;
  mnuSaveReport.Enabled := SaveBTN.Enabled;
  if inAntiVirus_Scan then mnuAntiVirus_ScanStart.Enabled := False else
mnuAntiVirus_ScanStart.Enabled := True;
  if inAntiVirus_Scan then mnuStopAntiVirus_Scan.Enabled := True else
mnuStopAntiVirus_Scan.Enabled := False;
end;

procedure TMainForm.mnuAntiVirus_ScanStartClick(Sender: TObject);
begin
  AntiVirus_ScanBTN.Click;
end;

procedure TMainForm.mnuStopAntiVirus_ScanClick(Sender: TObject);
begin
  AntiVirus_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 AntiVirus_AddPath.pas – додавання шляхів сканування

```
unit AntiVirus_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 AntiVirus_Main, AntiVirus_Options, uSelInfo;

{$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 AntiVirus_InfectedAction.pas – вибір дії над інфікованим об'єктом

```
unit AntiVirus_InfectedAction;

interface

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

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 AntiVirus_Main, AntiVirus_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 AntiVirus_Options.pas – параметри антивірусу

```
unit AntiVirus_Options;

interface

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

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;
    optAntiVirus_ScanLabel: 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 AntiVirus_Main, uPluginInfo, AntiVirus_AddPath, uSelDir, uHideForm;

{$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('Virus_AntiVirus_Scanner',False);
  end else
  begin
    ChangeReg('Virus_AntiVirus_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_ANTIVIRUS_SCAN_SUBDIR         := SCNSUBDIR.Checked;
        OPT_USE_HEX_MODE        := SCNHEX.Checked;
        OPT_USE_CRC_MODE        := SCNCRC.Checked;
        OPT_USE_HEX_INPOS       := SCNHEXINPOS.Checked;
        OPT_SEND_ANTIVIRUS_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('*','AntiVirus.AntiVirus_Scan',MainForm.SysMenu,ParamStr(0)+'
%1');

FileTAddAction('Directory','AntiVirus.AntiVirus_Scan',MainForm.SysMenu,ParamStr(
0)+' %1');

FileTAddAction('Drive','AntiVirus.AntiVirus_Scan',MainForm.SysMenu,ParamStr(0)+'
%1');
  end else
```

```
  begin
    FileTDelAction('Drive','AntiVirus.AntiVirus_Scan');
    FileTDelAction('Directory','AntiVirus.AntiVirus_Scan');
    FileTDelAction('*','AntiVirus.AntiVirus_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 AntiVirus_About.pas – довідка

```
unit AntiVirus_About;

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 AntiVirus_Main;

{$R *.dfm}

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

end.
```