Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2021 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи хмарного антивірусного забезпечення”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-20МЗ
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Смірнов С.А.
«____» _____ 2021 р.

Керівник проекту
кандидат фізико-математичних наук, доцент
_____ Наталія ЯКИМЕНКО
«____» _____ 2021 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Центр *Заочної та дистанційної освіти*

Кафедра *Кібербезпеки та програмного забезпечення*

Рівень вищої освіти *магістр*

Галузь знань ._12 _____*"*Інформаційні технології*"*_____

Спеціальність *123 "*Комп'ютерна інженерія*"*

Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2021 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Смірнову Сергію Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи хмарного антивірусного забезпечення*

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

затверджені наказом вищого навчального закладу № 41-13 від 02.08.2021 року

3. Строк подання студентом роботи до захисту *10.12.2021 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи хмарного антивірусного забезпечення*

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

1. Призначення та область використання.	*7. Економічна ефективність розробленої*
2.Перегляд аналогічних існуючих систем.	*програми.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	
6. Наукова новизна	

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

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Савеленко Г.В.	05.10.2021	14.11.2021
Охорона праці	Оришака О.В.	06.10.2021	16.11.2021

7. Дата видачі завдання « 6 » вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2021 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2021 р.	
3.	Розробка моделі компонента	20.10.2021 р.	
4.	Розробка структур даних	25.10.2021 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2021 р.	
6.	Програмування алгоритмів	10.11.2021 р.	
7.	Розрахунок економічної ефективності	13.11.2021 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2021 р.	
9.	Оформлення ПЗ	17.11.2021 р.	
10.	Попередній захист роботи	10.12.2021 р.	

Дата видачі завдання
« 6 » вересня 2021 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2021 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Смірнов С.А. Дослідження та програмна реалізація системи хмарного антивірусного забезпечення. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2021.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи хмарного антивірусного забезпечення.

Метою розробки є дослідження та програмна реалізація системи хмарного антивірусного забезпечення.

Об'єктом дослідження є процес хмарного антивірусного забезпечення.

Предметом дослідження є методи хмарного антивірусного забезпечення.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи хмарного антивірусного забезпечення.

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 XP/Vista/7/8/10.

Програму розроблено в середовищі RAD Studio Delphi 10.4.

Ключові слова: комп'ютерна інженерія, хмари, антивірусне забезпечення

ABSTRACT

Smirnov S.A. Research and software implementation of cloud antivirus software. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2021

In this final qualification work on the second (master's) level of higher education the software which is intended for system of cloud antivirus providing is developed.

The purpose of development is research and software implementation of the cloud antivirus system.

The object of research is the process of cloud antivirus software.

The subject of research is the methods of cloud antivirus software.

Research methods are based on methods of information security theory, methods of mathematical statistics, methods of software development.

The result is a software implementation of the cloud antivirus software.

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.

Developed user-friendly interface. Instructions for working with software are given.

The program can be used on an IBM PC with Windows XP / Vista / 7/8/10.

The program is developed in the environment of RAD Studio Delphi 10.4.

Keywords: computer engineering, clouds, antivirus software

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 3
ВСТУП 4
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 6
1.1 Призначення системи 6
1.2 Область застосування 7
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 9
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти 9
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування 17
2.3 Розгорнута постановка завдання 22
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 24
3.1 Опис функціонування системи 24
3.2 Розробка структурної схеми 30
3.3 Розробка функціональної схеми 38
3.4 Розробка діаграми процесів 40
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ 42
4.1 Розробка блок-схем та опис алгоритмів функціонування системи 42
4.2 Захист розробленого програмного забезпечення 62
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ..... 64
6 НАУКОВА НОВИЗНА 71

7 ЕКОНОМІЧНА ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ 72
7.1 Техніко економічне обґрунтування теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. 72
7.2 Розрахунок трудомісткості розробки програмної продукції 74
7.3 Визначення чисельності виконавців і планового фонду зарплати 76
7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника 81
7.5 Визначення собівартості розробки та ціни програмної продукції. 83
7.6 Визначення об'єму капітальних вкладень та експлуатаційних витрат у споживача програмної продукції..... 87
7.7 Визначення експлуатаційних витрат..... 87
7.8 Визначення економічної ефективності програмної продукції..... 89
7.9 Висновок. 91
8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ 92
8.1 Шкідливі і небезпечні фактори при роботі з комп'ютером 92
8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста ... 94
8.3 Розробка заходів з умов поліпшення охорони праці..... 97
8.4 Розрахункова частина 97
8.5 Висновки до розділу..... 99
9 ОСНОВНІ ВИСНОВКИ..... 100
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 102

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. Масове застосування персональних комп'ютерів, на жаль, виявилося пов'язаним з появою програм-вірусів, що самовідтворюються, які перешкоджають нормальній роботі комп'ютера та руйнують файлову структуру дисків і наносять збиток збереженої в комп'ютері інформації. Це ж й відноситься до комп’ютерів приєднаних до Інтернету, або інших пристроїв, які під’єднані до Інтернету.

Щоб ефективно боротися з вірусами, необхідно мати подання про структуру алгоритмів вірусів і орієнтуватися в методах протидії вірусам. Вірусом називається спеціально створена програма, здатна самостійно поширюватися в комп'ютерному середовищі. Якщо вірус потрапив у комп'ютер разом з однією із програм або з файлом документа, то через якийсь час інші програми або файли на цьому комп'ютері будуть заражені. Якщо комп'ютер підключений до локальної або глобальної мережі, то вірус може поширитися й далі, на інші комп'ютери. Автори вірусних програм створюють їх з різних спонукань, однак результати роботи вірусів виявляються, як правило, схожими: інфекції псують програми й документи, які знаходяться на комп'ютері, що часто приводить до їхньої втрати. Деякі віруси здатні знищувати взагалі всю інформацію на дисках комп'ютерів, вартість якої може в десятки й сотні разів перевищувати вартість самого комп'ютера.

Існують три рубежі захисту від комп'ютерних вірусів:

- запобігання надходження вірусів;
- запобігання вірусної атаки, якщо вірус все-таки надійшов на ПК;
- запобігання руйнівних наслідків, якщо атака все-таки відбулася.

Існують три методи реалізації захисту:

- Програмні методи захисту.
- Апаратні методи захисту.
- Організаційні методи захисту.

Хмарні антивіруси відносяться до програмних методів захисту, хоча, якщо будувати комплексну систему захисту від вірусів, то потрібно використовувати

усі вище перераховані методи.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи хмарного антивірусного забезпечення.

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

– Огляд існуючих систем хмарного антивірусного забезпечення.

– Дослідження системи хмарного антивірусного забезпечення.

– Програмна реалізація системи хмарного антивірусного забезпечення.

Об'єктом дослідження є процес хмарного антивірусного забезпечення.

Предметом дослідження є методи хмарного антивірусного забезпечення.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод хмарного антивірусного забезпечення.

– Розроблено вітчизняний продукт хмарного антивірусного забезпечення, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі хмарного антивірусного забезпечення.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LV Науково-технічна конференція здобувачів вищої освіти «Наука – виробництву», 2021, основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №12.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи хмарного антивірусного забезпечення, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система хмарного антивірусного забезпечення – складна розподілена інфраструктура, призначена для інтелектуальної обробки потоків даних, пов'язаних з кіберзагрозами й надаваних добровільно мільйонами користувачів по усьому світу. Автоматичний аналіз цих даних у хмарі дозволяє системі максимально швидко реагувати на нові й ще невідомі погрози. Це один з найбільш важливих компонентів багаторівневого підходу системи хмарного антивірусного забезпечення до захисту нового покоління. Ключовою частиною цього підходу є концепція, яка поєднує в собі експертний аналіз, алгоритми машинного навчання й обробку великих даних, що дозволяє оперативно виявляти нові паттерни погроз, що виникають у кіберпросторі.

Основні принципи роботи системи хмарного антивірусного забезпечення.

1. Продукти системи хмарного антивірусного забезпечення відправляють у хмарну інфраструктуру статистичні дані по виявлених погрозах і підозрілої активності.

2. Отримані масиви великих даних обробляються автоматичною системою аналізу, що розпізнає самі нові кіберзагрози. Ця система використовує потужності системи хмарного антивірусного забезпечення, не завантажуючи ресурси користувачів.

3. Якщо код або Url -Адреса зізнається шкідливим, цей вердикт у лічені хвилини стає доступним для всіх користувачів. Паралельно в базу даних додаються записи про дозволені додатки.

4. Продукти негайно одержують відповіді на будь-які нові запити про репутацію об'єктів.

Цей підхід забезпечує наступні переваги для сукупного рівня безпеки клієнтських систем і даних:

– виявлення просунутих і раніше невідомих шкідливих програм;

– зниження кількості помилок виявлення (неправильних спрацьовувань);

– істотне скорочення часу реагування на нові погрози – у порівнянні із традиційним реагуванням на основі сигнатур цей час виміряється не годинником, а хвилинами або навіть секундами.

Основні принципи роботи системи хмарного антивірусного забезпечення

– Обробляється тільки та інформація, яка необхідна для вдосконалення алгоритмів виявлення, підвищення точності роботи продуктів і розробки нових поліпшених застосунків.

– Система одержує інформацію від клієнтів, що прийняли умови Ліцензійної угоди з кінцевим користувачем (EULA) і угоди система хмарного антивірусного забезпечення, у якому докладно описані види інформації, що збирається.

– Угода система хмарного антивірусного забезпечення можна прийняти або відхилити в будь-який час за допомогою настроювань застосунку.

– Одержувані система хмарного антивірусного забезпечення дані не співвідносяться з конкретними користувачами. Інформація використовується у формі зведеної статистики на окремих серверах зі строгими політиками відносно прав доступу.

– Надавана інформація захищається навіть у процесі її передачі відповідно до вимог законодавства й найсуворішими галузевими стандартами, у тому числі за допомогою шифрування, цифрових сертифікатів, мережних екранів і т.д.

1.2 Область застосування

В основі роботи будь-якої антивірусної програми лежать ті самі методи розпізнавання шкідливих об'єктів. У їхньому числі різні варіанти сканування й моніторингу. Розглянемо найпоширеніші з них.

Пряме сканування

Цей метод припускає прямий пошук в оперативній пам'яті й на вінчестері відомих послідовностей коду вірусу (сигнатур). Для цього розроблювачі ретельно стежать за появою нових вірусів і вносять їхньої сигнатури в ко ристувальницькі

програми шляхом регулярних відновлень. Метод дуже надійний і практично не дає помилкових спрацьовувань. Основний його недолік – неможливість відшукати віруси, що ще не потрапили в пакети відновлень, і поліморфні об'єкти, код яких змінюється при кожному запуску.

Евристичне сканування

Достоїнство цього методу – використання при перевірці файлів не принципу пошуку сигнатур, а комплексного аналізу, включаючи оцінку можливого поводження підозрілого об'єкта. Однак через те, що при евристичному скануванні шукаються не шкідливі об'єкти як такі, а об'єкти, схожі на них, можливі помилкові спрацьовування. Зате таким шляхом можна визначити віруси, які легко видозмінюються й невідомі.

Моніторинг змін

Один із самих заслужених методів, що відслідковує зміну параметрів (розміру, дати та ін.) об'єктів на вінчестері. Вимагає попереднього збору інформації про “здорову систему”. Знижує швидкодію комп'ютера. Помилкові спрацьовування неминучі.

Моніторинг поводження

Цей метод здатний “піймати” невідомий або поліморфний вірус “на льоту”, визначивши його по шкідливих діях. Негативні риси: помилкові спрацьовування й підвищені вимоги до ресурсів комп'ютера.

Після виявлення зараженого файлу антивірусне ПЗ, як правило, пропонує користувачеві “вилікувати” його, перейменувати, перемістити в спеціальну карантинну папку або видалити. Всі ці дії можна робити автоматично, але в цьому випадку збиток від помилкових спрацьовувань може перевищити втрати від вірусної атаки.

Сучасні комплексні програми використовують сполучення різних методів виявлення й захисту. Часто в дистрибутив антивірусного пакета сканери й монітори входять окремими утилітами.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи хмарного антивірусного забезпечення, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Розглянемо існуючі хмарні антивірусні комплекси.

McAfee

McAfee пропонує програмне забезпечення з показниками виявлення вірусів вище за середнє. Пакети McAfee включають вбудовані опції по оптимізації ПК, батьківського контролю, блокування спама по електронній пошті, захисту від хакерів і зломщиків і захисту посилань у соціальних мережах для необмеженого числа пристроїв.

Антивірус McAfee включає повний антивірусний захист із безкоштовною підтримкою 24/7 і вбудованими інструментами оптимізації ПК для одного пристрою.

McAfee Internet Security включає повний антивірусний захист із безкоштовною підтримкою 24/7, вбудовані інструменти оптимізації ПК, батьківський контроль і блокування спама по електронній пошті для необмеженого числа пристроїв.

McAfee Total Protection включає повний антивірусний захист із безкоштовною підтримкою 24/7, вбудовані інструменти оптимізації ПК, батьківський контроль, блокування спама по електронній пошті, захист від хакерів і зломщиків і захист посилань у соціальних мережах для необмеженого числа пристроїв. Захистите особисті файли за допомогою програмного забезпечення шифрування й безпечно зберігаєте свої паролі за допомогою додатка True Key.

Програмне забезпечення McAfee, імовірно, найпростіший антивірусний пакет на ринку. Основна панель керування дозволяє користувачам довідуватися

про поточний стан захисту операційної системи. Швидке й повне сканування виконується всього в кілька кліків, і автоматичне сканування легко налаштувати.

Цей антивірус має всі функції, які включені й попередньо налаштовані відразу після установки, без необхідності перезавантаження. Хоча в минулому програма мала трохи сумнівну якість, нинішнє покоління програмного забезпечення приємно дивує користувачів.

Користувачі також можуть легко налаштувати батьківський контроль, брандмауер, параметри захисту від спама, планові перевірки на віруси й параметри відновлення безпосередньо з користувацького інтерфейсу. Уся навігація інтуїтивно зрозуміла, і користувачі можуть установлювати параметри, які щонайкраще відповідають їхнім потребам.

При порівнянні показників надійності більшість антивірусних пакетів повинне мати свої власні показники у двох основних областях антивірусного захисту: виявлення вірусів за допомогою традиційного сканування вірусів і захисту в режимі реального часу, коли файли зберігаються в системі.

Згідно AV-Comparatives, відомої сторонньої компанії по тестуванню антивірусних програм, усі антивірусні пакети McAfee перебувають на верхньому рівні показників у відношенні виявленні розповсюджених вірусів. І хоча McAfee не так сильний для захисту при скануванні в реальному часі, він однаково одержав середній показник.

Norton

Norton від компанії Symantec забезпечує повний захист до 10 пристроїв з першокласними характеристиками, від захисту від шкідливих програм до автоматичного резервного копіювання світлин.

Norton забезпечує повний захист до 10 пристроїв і має першокласні характеристики, від захисту від шкідливих програм до автоматичного резервного копіювання світлин.

Norton є одним з найвідоміших брендів в області антивірусного захисту. Це краще програмне забезпечення для захисту споживачів з погляду продуктивності й загального виявлення вірусів, згідно з показниками швидкості виявлення вірусів і швидкості роботи програми.

Кожний із трьох пакетів Norton гарантовано забезпечує найкращий захист персональних даних і працює з кожної операційною системою: Windows, Mac, iOS або Android. Він простий у настроюванні й установці, включає прості у використанні програми й забезпечує дружню й корисну підтримку клієнтів при виникненні проблем.

Norton Antivirus Plus: За 39,99 дол. США протягом першого року, можна захистити 1 ПК або 1 mac. Даний антивірус забезпечує потужний захист від шпигунських програм, вірусів, шкідливих програм і здирників. Пакет також включає інтелектуальний брандмауера, онлайн-захист від погроз, 2 ГБ хмарного резервного копіювання й менеджер паролів.

Пакет Norton 360 Standard: За 49,99 дол. США протягом першого року стандартний пакет Norton 360 Standard включає всі те ж, що й Norton Antivirus Plus, але також пропонує 10 ГБ хмарного резервного копіювання, безпечний VPN, Safecam і Dark Web Monitoring (живлення від Lifelock) для більш безпечного перегляду в інтернеті.

Пакет Norton 360 Deluxe: За 59,99 дол. США протягом першого року цей пакет захищає до 5 ПК, Mac, смартфонів або планшетів. На додаток до стандартного пакета Deluxe, він пропонує хмарне резервне копіювання обсягом 50 ГБ і батьківський контроль.

Пакет Norton 360 з Lifelock Select: За 99,99 дол. США протягом першого року цей пакет включає все, що й пакет Deluxe, а також 100 ГБ хмарного резервного копіювання, систему оповіщення Lifelock ID, кредитний моніторинг і пакет захисту на мільйон доларів.

Norton Security може конкурувати з іншими провідними постачальниками, такими як McAfee як один з найпростіших у використанні антивірусних пакетів. Користувачі можуть визначити безпеку й уразливість свого пристрою з першого погляду. Усього в кілька кліків користувач може виконувати швидке сканування, повне сканування, резервне копіювання або оптимізацію продуктивності системи.

Norton Security установлюється дуже швидко, і одночасно користувач знайомиться з функціями програмного забезпечення. Швидка установка й

менший розмір додатка, однак, не компенсують середню продуктивність і швидкість виявлення вірусів.

Панель керування дозволяє користувачам легко запускати антивірусні перевірки, відновлення, настроювання захисту персональних даних, резервне копіювання онлайн і настроювання ПК. Панель керування чітка й інтуїтивно зрозуміла. Залежно від пакета, програма оптимізує захист усіх ваших пристроїв з одного місця й при одній підписці.

Батьківський контроль і функції безпеки родини є, мабуть, кращими аспектами даного програмного забезпечення. Користувачі можуть налаштовувати сайти з панелі керування з різними обмеженнями, і ці обмеження автоматично застосовуються на всіх пристроях користувача. Крім того, антивірус надає розгорнутий звіт і відкликання для батьків, що опікуються про захист своїх дітей. Потім батьки можуть уводити нові обмеження на основі поведінки своєї дитину.

Norton також надає різні додаткові послуги кібер-безпеки.

На ринку антивірусного ПЗ Norton пропонує середній ступінь захисту по дуже конкурентоспроможних цінах. Програми Norton є відмінним вибором для користувачів, зацікавлених у простому програмному забезпеченні із середньою частотою виявлення вірусів і надзвичайно потужними функціями батьківського контролю.

Одним з переваг Norton Security є 60 -денна гарантія повернення грошей. На додаток до цього, Norton також пропонує гарантію захисту від вірусів. Якщо фахівець із безпеки не може вилучити вірус за допомогою програми, ви одержите 100% повернення ваших грошей.

Як і аналогічні компанії, Norton надає клієнтам безліч варіантів підтримки. Центр підтримки на сайті включає базу з актуальними статтями, відео, завантаженнями, що й часто задаються питаннями. Користувачі можуть скористатися онлайн-чатом Norton, електронною поштою або придбати різні пакети підтримки. Офіси компанії є в багатьох країнах миру, тому неважливо, де ви перебуваєте, ви знайдете когось, хто говорить на вашій мові.

Якщо ви прагнете довідатися, як використовувати програми Norton більш ефективно, різні освітні послуги доступні на офіційному сайті Norton.

Користувачі можуть запросити консультацію експерта, записатися на заняття під керівництвом інструктора або частки тренінги, а також одержати доступ до поглибленої бібліотеки.

Norton надає підтримку в будь-який час!

AV-Test, один із самих шановних незалежних тестувальників даної галузі, постійно повідомляє, що Norton є одним з найпростіших у використанні антивірусних програм, як для початківців, так і для професіоналів. Вони прийшли на цей ринок одними з перших, і їх інтерфейс був запозичено багатьма антивірусними продуктами, які прийшли на ринок пізніше, тому Norton може здатися дуже знайомим.

Завдяки інтелектуальному скануванню й низькому споживанню ресурсів, антивірус може працювати на комп'ютерах з низькими характеристиками без переривання роботи користувача й доступний для всіх рівнів кваліфікації. Функції легко включити або виключити, і легко довідатися, як використовувати додаткові функції, наприклад, резервне копіювання файлів і світлин.

Norton надає простий у використанні захист і потужний батьківський контроль. Залежно від пакета, Norton Security включає повний антивірусний захист, захист брандмауера, захист персональних і фінансових даних, батьківський контроль, блокування небажаних листів і спама, менеджер паролів, засобу оптимізації ПК і оперативне резервне копіювання – хмара обсягом до 25 ГБ для 10 комп'ютерів і інших пристроїв.

Коли мова йде про антивірусний захист, надійний батьківський контроль і простота у використанні роблять Norton відмінним вибором для користувачів, яким необхідні середні показники виявлення вірусів у комбінації з ефективним батьківським контролем. При цьому швидкість виявлення вірусів змушує бажати кращого.

Корпорація Norton є одним з першопрохідників в області антивірусного захисту. Компанія позиціонує себе як світовий лідер в області кібербезпеки нового покоління, продовжуючи створювати пакети безпеки, підтримки й антивірусів Norton, у тому числі Norton Security для одного пристрою, Norton Security і Norton Security with Backup.

Корпорація Norton була заснована в 1982 році в Саннивейлі, штат Каліфорнія, і в цей час її штаб-квартира розташована в Маунтин-в'ю, Каліфорнія

Avira

Безкоштовне антивірусне програмне забезпечення Avira може захистити комп'ютери й мобільні пристрої від шкідливих програм, включаючи чирви, віруси, трояни й шпигунські програми.

Безкоштовне антивірусне програмне забезпечення Avira може захистити комп'ютери й мобільні пристрої від шкідливих програм, включаючи чирви, віруси, трояни й шпигунські програми. Для безкоштовного ПЗ це непогано, але платна версія, Avira Pro, пропонує більш повний захист.

Маючи 30-літній досвід роботи в галузі, Avira із самого початку завойовувала нагороди за своє програмне забезпечення. Тільки в 2016 році німецька компанія завоювала більш 20 нагород, а її новітня програма Antivirus Pro була визнана кращою програмою року за версією AV Comparatives.

Avira пропонує як безкоштовну, так і платну версію антивірусного програмного забезпечення, а також окремі програми, такі як Avira System Speed-Up і Avira VPN. Є такі пакети, як Avira Total Security Sui te і Avira Prime. Багато варіантів пакетів дозволяють споживачам створювати свої власні пакети захисту комп'ютера залежно від їхніх потреб.

Безкоштовний антивірус Avira забезпечує захист від шкідливих програм для пристроїв Windows, Mac, Android і iOS. Прог раму можна безкоштовно скачати із сайту компанії. Avira Pro є платною версією, і вона поставляється з додатковими функціями й захистом, такими як анти-здирники й анти-фішинг, керування пристроями, захист персональних даних, захист банківських операцій і покупок, а також конфіденційність даних. Користувач повинен увести ліцензійний ключ для одержання послуг Avira Pro, а клієнти, що оплачують програму, мають безкоштовний доступ до служби підтримки Avira по телефону й електронній пошті. Вартість становить приблизно 45 доларів США в рік.

Що ж стосується ефективності, то AV-Test (незалежна тестова лабораторія) дійшла висновку, що програма Avira забезпечує надійний захист. Avira Pro виявилася на 98,8% ефективної проти погроз нульового дня (атаки

вірусів, які раніше були невідомі) і на 99,9% ефективної проти розповсюджених шкідливих програм, які були виявлені в попередні 4 тижні.

Безкоштовний антивірус Avira простий у використанні - скачайте його й додержуйтеся інструкцій по установці. Панель керування інтуїтивно зрозуміла, і антивірус легко налаштувати. Його також можна налаштувати відповідно до частоти сканування комп'ютера.

Хоча безкоштовне антивірусне програмне забезпечення є простим у використанні, деякі користувачі повідомляють, що не можуть одержати доступ до певних веб-сайтів, якщо вони не підписалися на платну версію Avira. Хоча це не є нормою, і більшість користувачів може одержати доступ до всіх сайтів, вони як і раніше будуть час від часу одержувати пропозиції про придбання платної версії Avira.

Avira Pro, платне антивірусне програмне забезпечення, також просте у використанні. Покупці одержують безкоштовну технічну підтримку й безпечний веб-браузер. Pro сканує вкладення електронної пошти, постійно обновляє свою базу даних і блокує відомі шкідливі веб-сайти. Усе це робиться автоматично, але користувачі також можуть налаштувати параметри у відповідності зі своїми комп’ютерної мережіми - наприклад, включення або вимикання захисту в реальному часі, брандмауера, веб-захисту, захисту пошти й режиму гри.

З появою проблеми, на панелі керування з'явиться червоний знак оклику, який попереджає користувачів про те, що вони повинні запустити сканування або налаштувати параметри. Програмне забезпечення підкаже користувачеві правильний порядок дій.

Основна проблема антивірусу Avira полягає в тому, що він може сповільнювати нормальну роботу комп'ютера, особливо при запуску ретельного сканування. Це може бути єдиним мінусом, особливо для тих, кому потрібна швидко працююча операційна система.

Що стосується безкоштовної версії, то Avira, мабуть, є якісним програмним продуктом. Її ефективність проти шкідливих програм постійно показує високі результати в тестах, проведених незалежними компаніями. Проте, повна версія програми забезпечує більш високий ступінь захисту. Це означає, що

в безкоштовній версії відсутні всі функції, які ви одержуєте з Avira Pro – анти-здирники й анти-фішинг, контроль пристроїв, захист персональних даних, безпека банківських і торговельних операцій і конфіденційність даних.

Маючи вартість 45 доларів США в рі к, Avira Pro небагато дорожче аналогічних програм, але захист, який ви одержуєте, дуже надійна. Avira Pro стала кращою програмою 2016 року й незмінно показує високі бали в тестах, проведених AV-Test. Коротше кажучи, за кілька додаткових доларів, ви одержуєте відмінну програму.

Користувачі, які завантажують безкоштовну версію антивірусу Avira, мають доступ до сторінок довідки Avira, форуму клієнтів, інформації про актуальні кіберзагрозах і словнику безпеки. Також надаються безкоштовні консультації технічних фахівців. Однак, для одержання преміум-підтримки по телефону й електронній пошті користувачі повинні надати номер ліцензії, який видається тільки клієнтам платної версії програми.

Безкоштовна версія антивірусу Avira проста в установці. Після установки Вам також буде запропонований ряд інших безкоштовних програм Avira, таких як VPN, прискорення роботи системи, захист для мобільних пристроїв, менеджер паролів, відновлення програмного забезпечення й багато чого іншого. Установка займає кілька хвилин, і після установки значок Avira з'явиться в правому нижньому куті екрана. Потім можна виконувати сканування системи безпеки й одержувати оповіщення при блокуванні підозрілого веб-сайту.

Avira Pro, платна версія, має аналогічну просту установку. Хоча деякі користувачі повідомляють, що вона сповільнює роботу комп'ютера, незалежне тестування показало, що ПЗ Avira сканує інтернет-файли досить швидко й перебуває на одному рівні з такими програмами як Norton і Mcaffee.

Безкоштовний антивірус Avira пропонує гарний базовий захист від шкідливих програм, але для більш комплексного захисту краще підходить Avira Pro. Обоє програмних продукту можуть працювати на пристроях Windows, Mac, Android і iOS, але тільки Pro включає програмне забезпечення для захисту від здирників і фішингу, керування пристроями, захист персональних даних, безпека банківських і торговельних операцій і конфіденційність даних. Хоча Pro небагато

дорожче аналогічних програм, різниця в ціні не є значної, якщо ви шукаєте якісний захист і доступну підтримку клієнтів.

2.2 Обґрунтування вибору засобів для побудови системи 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 одн ому 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 другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи хмарного антивірусного забезпечення.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) 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 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 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

Хмарний антивірус знімає з окремого комп'ютера робочі навантаження, пов'язані з роботою антивірусу, і переносить їх на хмарний сервер, на якому встановлений всебічний пакет антивірусного захисту. Використання хмарних антивірусів рятує від необхідності встановлювати на окремі комп'ютери антивірусні продукти, що вимагають значних ресурсів і сповільнювані роботу.

Переваги хмарних застосунків

Поліпшене виявлення вірусів у реальному часі

Наявність на пристрої непомічених вірусів або незахищене підключення може стати причиною зараження робочої мережі компанії й інших кінцевих точок. Традиційні застосунки для кінцевих точок вимагають від користувача завантажувати програмне забезпечення безпосередньо на пристрої. Однак хмарні застосунки можуть працювати віддалено, зупиняючи шкідливі програми (наприклад, програми-здирники) і віруси до того, як вони доберуться до робочої мережі. Сканування й відновлення вірусних сигнатур відбуваються автоматично в реальному часі, що не залишає проломів у захисті.

Застосунки з низьким споживанням ресурсів

Традиційний антивірус, установлений на пристрої, може сильно сповільнювати його роботу, особливо під час установки відновлень. Крім того, користувач сам відповідає за актуальність програмного забезпечення. Це забирає час і може заважати роботі, тому багато зневажають установкою відновлень. Але застарілі антивірусні програми піддають ризику всю компанію. Хмарні антивірусні застосунки рятують від цих проблем, тому що розташовані в хмарі.

Захист декількох пристроїв з єдиного центрального сервера

Хмарне антивірусне програмне забезпечення підвищує продуктивність, зміцнює безпеку й підтримує централізоване керування. Платформи хаб системи

хмарного антивірусного забезпечення і ядро системи хмарного антивірусного забезпечення дозволяють додавати нові пристрої, видаляти непотрібні й блокувати доступ для чужих або підозрілих пристроїв. Завдяки використанню централізованої платформи компанії можуть забезпечувати постійне відновлення антивірусних програм. IT-відділам не потрібно витрачати час на завантаження останніх вірусних сигнатур для сканування – хмарний антивірус обновляється автоматично.

Ми використовуємо різні платформи керування

Система хмарного антивірусного забезпечення пропонує на вибір дві платформи керованого захисту з необхідними інструментами й інформацією для повноцінного контролю й огляду всієї вашої екосистеми безпеки. Об'єднання потрібних рівнів на інтуїтивній платформі забезпечує бездоганний захист, не жертвуючи простотою й зручністю. Ознайомтеся з нашими платформами безпеки, щоб вибрати підходящу саме вам.

Хаб системи хмарного антивірусного забезпечення

Платформа хаб системи хмарного антивірусного забезпечення ідеально підходить для малих підприємств, яким необхідна проста у використанні хмарна платформа для швидкого розгортання антивірусу на декількох кінцевих точках, безперервного моніторингу й одержання інформації, необхідної для забезпечення бездоганного захисту:

– Просте керування на основі хмарної технології.

– Керування всіма пристроями з єдиної інтуїтивно зрозумілої панелі керування.

– Швидке розгортання й сканування.

– Моніторинг і оповіщення в реальному часі.

Ядро системи хмарного антивірусного забезпечення

Ядро системи хмарного антивірусного забезпечення – це Saas-платформа, яка дозволяє поєднувати послуги захисту для кінцевих точок і мереж, відслідковувати погрози й усувати неполадки за допомогою єдиної платформи.

Послуги захисту включають безпечний веб-шлюз, захист електронної пошти, антивірус, хмарне резервне копіювання й багато чого іншого.

– Централізована панель керування.
– Докладні звіти.
– Віддалена IT-підтримка.
– Інтеграція сторонніх продуктів.

Хмарний антивірус Система хмарного антивірусного забезпечення захищає всі пристрої в мережі компанії, не сповільнюючи їх роботу, на відміну від антивірусів, які розгортаються на окремих пристроях. Систему хмарного антивірусного забезпечення з підтримкою хаб системи хмарного антивірусного забезпечення можна розгорнути за допомогою нашої платформи хаб системи хмарного антивірусного забезпечення. Це допоможе підвищити безпека за рахунок негайних попереджень і безперервного моніторингу.

Хмарний антивірус для бізнесу працює на безлічі пристроїв. Ви зможете управляти своїми пристроями й мережею, де б ви не були. Знадобиться тільки підключення до Інтернету й дані облікового запису в хаб системи хмарного антивірусного забезпечення. Активація нових пристроїв після установки запобігає автоматичному додаванню невідомих пристроїв на вашу платформу. Ви завжди можете вилучити будь-які незнайомі пристрої.

Існує три рівні захисту: Система хмарного антивірусного забезпечення, Система хмарного антивірусного забезпечення Pro і Система хмарного антивірусного забезпечення Pro Plus. Усе хмарне антивірусне програмне забезпечення включає брандмауера, захист електронної пошти, захист від спама й інші функції. Плани Pro і Pro Plus включають захист серверів і функцію знищення даних, яка видаляє файли безповоротно, кілька раз перезаписуючи на їхнім місці випадкові дані. Функції VPN, «Захист паролів», «Захист веб-камери» і «Очищення браузера» доступні тільки в плані Pro Plus.

Щоб розгорнути керований антивірус за допомогою хаб системи хмарного антивірусного забезпечення, просто придбайте його або виберіть один із трьох

антивірусних планів: Система хмарного антивірусного забезпечення, Система хмарного антивірусного забезпечення Pro або Система хмарного антивірусного забезпечення Pro Plus. Ви побачите панель керування хаб системи хмарного антивірусного забезпечення, на якій можна перевірити стан мережі, переглянути настроювання своїх пристроїв, а потім розгорнути на них настановний файл. Останній крок – активація нових пристроїв.

Єдиним керованим антивірусом, сумісним з ядро системи хмарного антивірусного забезпечення, є система хмарного антивірусного забезпечення Pro Plus, що забезпечує повний захист пристрою, облікових і інших даних. Щоб приступитися до роботи, перейдіть на сторінку ядро системи хмарного антивірусного забезпечення й зареєструйтеся для одержання безкоштовної пробної версії. Представник системи хмарного антивірусного забезпечення зв'яжеться з вами, щоб розповісти про платформу, створити обліковий запис і відправити лист із обліковими даними.

Ядро системи хмарного антивірусного забезпечення

Універсальна платформа для забезпечення повної безпеки, повністю реалізована через хмару

Управляйте всіма своїми клієнтами і їх застосунками по забезпеченню безпеки за допомогою однієї простій у використанні платформи

Ядро системи хмарного антивірусного забезпечення забезпечує надійну службу забезпечення безпеки постачальникам керованих служб, яким потрібен найвищий рівень захисту для клієнтів. Цей застосунок легкий впровадити й управляти ім. ядро системи хмарного антивірусного забезпечення дозволяє скоротити накладні витрати й витрати, забезпечуючи при цьому самий повний захист нового покоління для кінцевих точок і хмарний захист мережі.

Усунення слабких місць у системі безпеки

Ваші клієнти є об'єктами кібератак з високим ризиком. Ви певен у їхній поточній системі безпеки? ядро системи хмарного антивірусного забезпечення

включає всі необхідні рівні захисту від погроз, забезпечуючи безпеку користувачів і пристроїв завжди й скрізь.

Легке керування декількома мережами

Мережі ваших клієнтів не управляють собою самі. Коли дані всіх погроз, відновлень і звітів доступні для перегляду на одній панелі, набагато простіше підтримувати зв'язок із клієнтами й забезпечувати їхню безпеку. Ядро системи хмарного антивірусного забезпечення масштабується відповідно до потреб вашого бізнесу, дозволяючи легко додавати служби забезпечення безпеки, які потрібні вашим клієнтам.

Боротьба зі зростаючими витратами на обслуговування клієнтів

Усе, що вам потрібно, від контролю погроз до функцій віддаленого керування, перебуває на одній платформі, що забезпечує вам додатковий час на розвиток свого бізнесу. ядро системи хмарного антивірусного забезпечення дозволяє ефективно використовувати ресурси, скорочуючи кількість відвідувань об'єктів клієнта й тим самим заощаджуючи ваші гроші й час.

Розвиток MSSP

Ми знаємо, наскільки важливо для вас забезпечувати кращу у своєму класі захист, а також побільшати свій доход. ядро системи хмарного антивірусного забезпечення дозволяє постачальникам керованих служб забезпечення безпеки легко масштабувати свої системи, додавати нові служби й віддалено розгортати системи безпеки для нових клієнтів.

Ще ніколи не було так просто відслідковувати погрози в реальному часі

Забезпечте найефективніший захист і прозорість мереж ваших клієнтів за допомогою універсальної централізованої платформи, створеної для постачальників керованих послуг забезпечення безпеки.

Відслідковуйте погрози й надавайте негайну підтримку за допомогою єдиної хмарної платформи керування безпекою

Безпечне підключення до будь-якого керованого пристрою

Використовуйте наш безкоштовний засіб віддаленої підтримки, щоб безпечно підключатися до будь-якого пристрою із установленим агентом ядро

системи хмарного антивірусного забезпечення й віддалено усувати проблеми, виконувати завдання, перезавантажувати комп'ютери, переносити файли, спілкуватися з користувачами й вирішувати інші питання.

Єдина панель керування для контролю погроз

Налаштуйте оповіщення для певних дій пристроїв і відслідковуйте їх із централізованої панелі оповіщень, щоб одержувати всю інформацію про проблеми на окремих пристроях, а також погрозах, які можуть поширюватися через підключені пристрої.

Розгортання уніфікованих служб багаторівневої кіберзахисту

Застосунок ядро системи хмарного антивірусного забезпечення дозволяє легко розгортати, налаштовувати й підтримувати ряд служб забезпечення безпеки для захисту даних, кінцевих точок, застосунків і мереж ваших клієнтів від самих складних кіберзагроз.

Функції ядро системи хмарного антивірусного забезпечення

Інтуїтивно зрозуміла панель керування

Швидко переглядайте всі оповіщення, вирішуйте проблеми й надавайте інформацію, потрібну для прийняття зважених застосунків, додавайте послуги й швидко вживайте заходів, щоб підвищити тривалість роботи, стабільність і безпека.

Керування пристроями й політиками

Управляйте безпекою всіх пристроїв за допомогою агента ядро системи хмарного антивірусного забезпечення. Зміни в політику будуть настроєні автоматично на всіх контрольованих агентом пристроях у режимі реального часу, що знижує необхідність обслуговування й значно спрощує масштабування бізнес-операцій.

Всебічна звітність

Збирайте дані з оглядів служб, зведень оповіщень і фільтрації контенту й створюйте зручні й докладні звіти про активність для себе й клієнтів.

Повідомлення в реальному часі

Налаштуйте оповіщення про важливі проблеми, що вимагають вашої уваги, і відразу ж відправляйте електронні листи й SMS-повідомлення порушеним сторонам, миттєво реагуючи й обмежуючи можливий збиток.

Розширені функції забезпечення безпеки ядро системи хмарного антивірусного забезпечення

За допомогою нашого універсального застосунку ви можете надавати повний спектр послуг для комплексного захисту.

Антивірус нового покоління для декількох ОС

Скористайтеся нашою системою хмарного антивірусного забезпечення Pro Plus, щоб забезпечити безпека пристроїв з ОС Windows і Mac, даних і клієнтів.

Керування виправленнями

Автоматично скануйте пристрою клієнтів, щоб знайти уразливості, і розгортайте виправлення для тисяч застосунків Windows і сторонніх програм.

Хмарне резервне копіювання

Запобігайте дорогим простоям за допомогою цілого ряду застосунків для резервного копіювання й відновлення файли, що захищають, додатки, сервери й багато чого іншого.

Фільтрація контенту

Підвищуйте продуктивність праці й блокуйте доступ до шкідливих сайтів і відволікаючим факторам в Інтернеті, щоб убезпечити співробітників своїх клієнтів і підвищити продуктивність їх роботи.

Безпечний веб-шлюз

Блокуйте доступ до небезпечних сайтів, завантажень і розташуванням, щоб запобігти атакам, здатні нашкодити мережі ваших клієнтів або викрасти які-небудь дані.

Безпечний інтернет-шлюз

Надавайте посилений і надійний захист від інтернет-погроз для всіх клієнтів, їх пристроїв і об'єктів.

3.2 Розробка структурної схеми

Комплексний підхід до забезпечення антивірусної безпеки передбачає погоджене застосування правових, організаційних і програмно-технічних мір, спрямованих на захист від можливих атак зловмисників. Відповідно до цього підходу повинен бути реалізований наступний комплекс мір:

– заходи щодо виявлення й усуненню вразливостей, на основі яких реалізуються вірусні погрози. Це дозволить виключити причини можливого виникнення вірусних атак;

– міри, спрямовані на своєчасне виявлення й блокування вірусних атак;

– міри, що забезпечують виявлення й ліквідацію наслідків вірусних погроз. Даний клас мір захисту спрямований на мінімізацію збитку, нанесеного в результаті реалізації вірусних погроз.

Важливо розуміти, що ефективна реалізація перерахованих вище мір на підприємстві можлива тільки за умови наявності нормативно-методичного, технологічного й кадрового забезпечення антивірусної безпеки.

Технологічне забезпечення повинне бути спрямоване на створення системи хмарного антивірусного забезпечення. Розглянемо підсистеми захисту, які повинні входити до складу системи хмарного антивірусного забезпечення для забезпечення захисту на рівні мережі, робочих станцій і серверів комп'ютерної мережі.

Захист на рівні мережі

Основним компонентом системи хмарного антивірусного забезпечення на рівні мережі є система розмежування доступу, що може реалізовуватися на трьох рівнях моделі OSI – канальному, мережевому й прикладному. На канальному рівні розмежування доступу здійснюється на основі віртуальних локальних мереж VLAN (Virtual Local Area Network), на які розділяється комп'ютерної мережі. Розподіл на такі віртуальні мережі виробляється за допомогою налаштувань комутаторів, у яких кожний фізичний порт включається в певну

віртуальну мережу. Хости можуть вільно обмінюватися даними один з одним у рамках однієї віртуальної мережі, а керування взаємодією між різними віртуальними мережами здійснюється за допомогою списків контролю доступу ACL (Access Control List). У цих списках визначаються правила, відповідно до яких дозволяється або забороняється інформаційний обмін між різними мережами VLAN. Так, наприклад, якщо для роботи комп'ютерної мережі два вузли не повинні обмінюватися між собою інформацією, то вони розділяються на різні віртуальні мережі, між якими забороняється взаємодія. У випадку, якщо комп'ютерний проникне на один з таких вузлів комп'ютерної мережі йому не вдасться одержати доступ до тих ресурсам, які зберігаються на інших серверах, включених в інші віртуальні мережі.

На мережевому й транспортному рівнях моделі OSI для розмежування доступу можуть застосовуватися міжмережеві екрани, призначені для блокування потенційно небезпечних пакетів даних, на основі яких поширюються комп'ютерні віруси. Як правило, міжмережеві екрани встановлюються в точці підключення комп'ютерної мережі до мережі Інтернет і забезпечують фільтрацію пакетів зі шкідливим кодом.

Розмежування доступу на прикладному рівні може реалізовуватися на основі технологій, що забезпечують можливість перевірки рівня безпеки робочих станцій перед наданням їм доступу до ресурсів комп'ютерної мережі. Так, наприклад, якщо на робочій станції буде відсутнє антивірусне ПЗ, або не будуть оновлені сигнатурні бази даних, то в цьому випадку доступ станції до комп'ютерної мережі буде заблокований. Прикладом такої технології є Cisco Network Admission Control.

На прикладному рівні моделі OSI рекомендується використовуватися мережеві засоби виявлення й запобігання атак, призначені для виявлення несанкціонованої вірусної активності за допомогою аналізу пакетів даних, що циркулюють у комп'ютерної мережі. Підсистема доповнює функції міжмережевих екранів (МЕ) за рахунок можливості більш детального

контентного аналізу вмісту переданих пакетів даних. Датчики системи виявлення атак установлюються до й після МЕ, а також у кожному із сегментів, що захищаються.

Крім системи виявлення атак для захисту комп'ютерної мережі також рекомендується використання засобу аналізу захищеності, призначені для виявлення технологічних і експлуатаційних уразливостей комп'ютерної мережі за допомогою проведення мережевого сканування. Як об'єкти сканування можуть виступати робочі станції користувачів, сервери, а також комунікаційне встаткування.

На прикладному рівні також можуть використовуватися шлюзові засоби антивірусного захисту, що дозволяють сканувати файли, передані по мережевих протоколах SMTP, POP3, HTTP, FTP і ін. Даний тип антивірусів підключається до міжмережевого екрана, проксі-серверу або встановлюється в розрив каналу зв'язку на виділеному вузлі. На рівні шлюзу також може забезпечуватися захист від поштових повідомлень, що містять спам.

Засоби захисту від вірусних погроз на рівні мережі перераховані в таблиці 3.1.

Захист на рівні робочих станцій користувачів

Базовим елементом захисту робочих станцій є засоби антивірусного захисту (таблиця 3.2). Основне завдання даних засобів полягає в антивірусній перевірці всіх файлів, які надходять на робочу станцію по мережі або через зовнішні носії інформації. У доповненні до засобів антивірусного захисту на станції рекомендується встановлювати персональні мережеві екрани, які дозволяють контролювати мережеву активність застосунків, а також хостові засоби виявлення атак. У випадку, якщо на станціях користувачів обробляється конфіденційна інформація, то вона повинна підлягати резервному копіюванню.

Таблиця 3.1 – Засоби захисту від вірусів на рівні мережі

№	Рівень моделі OSI	Найменування засобів захисту
1	Прикладний рівень	Шлюзові засоби антивірусного захисту Шлюзові засоби захисту від спаму Мережеві системи виявлення атак Засоби контролю доступу до ресурсів комп'ютерної мережі Засоби аналізу захищеності
2	Транспортний рівень	Міжмережеві екрани
3	Мережевий рівень	
4	Канальний рівень	Засоби розмежування доступу засобами VLAN
5	Фізичний рівень	Фізичне ізолювання певних сегментів комп'ютерної мережі друг від друга

Таблиця 3.2 – Засоби захисту від вірусів на рівні робочих станцій користувачів

№	Рівень моделі вузла комп'ютерної мережі	Найменування засобів захисту
1	Рівень інформаційних ресурсів	Засоби резервного копіювання інформації
2	Рівень прикладного ПЗ	Засоби антивірусного захисту Персональні мережеві екрани Хостові засоби виявлення й запобігання атак
3	Рівень загальносистемного ПЗ	
4	Рівень апаратного забезпечення	–

Захист на рівні серверів

На сервери, також як і на робочі станції повинні встановлюватися засоби антивірусного захисту, що забезпечують виявлення й блокування шкідливого коду. На відміну від робочих станцій, для забезпечення більш високого рівня

захисту на сервери можуть установлюватися багатовендерні хмарні антивіруси, до складу яких одночасно входить кілька скануючих ядер різних виробників.

Для захисту поштових серверів від спаму на них може бути встановлене спеціалізоване ПЗ, що дозволяє виявляти повідомлення рекламного характеру. Крім засобів захисту від вірусів і спаму на серверах доцільно розміщати засобу контролю цілісності інформаційних ресурсів, резервного копіювання, виявлення атак і мережевого екранування.

Таблиця 3.3 – Засоби захисту від вірусів на рівні серверів

№	Рівень моделі вузла комп'ютерної мережі	Найменування засобів захисту
1	Рівень інформаційних ресурсів	Засоби контролю цілісності інформації Засоби резервного копіювання інформації
2	Рівень прикладного ПЗ	Засоби антивірусного захисту Засоби захисту від спаму Персональні мережеві екрани Хостові засоби виявлення й запобігання атак
3	Рівень загальносистемного ПЗ	
4	Рівень апаратного забезпечення	–

Крім розглянутих вище засобів захисту комп'ютерної мережі, що функціонують на рівні мережі, робочих станцій і серверів, до складу системи хмарного антивірусного забезпечення також повинна входити підсистема керування антивірусною безпекою, призначена для виконання наступних функцій:

– хмарної установки й деінсталяції антивірусних засобів на серверах і робочих станціях користувачів;

– віддаленого керування параметрами роботи підсистем захисту, що входять до складу системи хмарного антивірусного забезпечення;

– централізованого збору й аналізу інформації, що надходить від інших підсистем. Дана функція дозволяє автоматизувати процес обробки даних, що

надходять,, а також підвищити оперативність прийняття рішень по реагуванню на виявлені інциденти, пов'язані з порушенням антивірусної безпеки.

Структурна схема системи хмарного антивірусного забезпечення показана на рисунку 3.1.

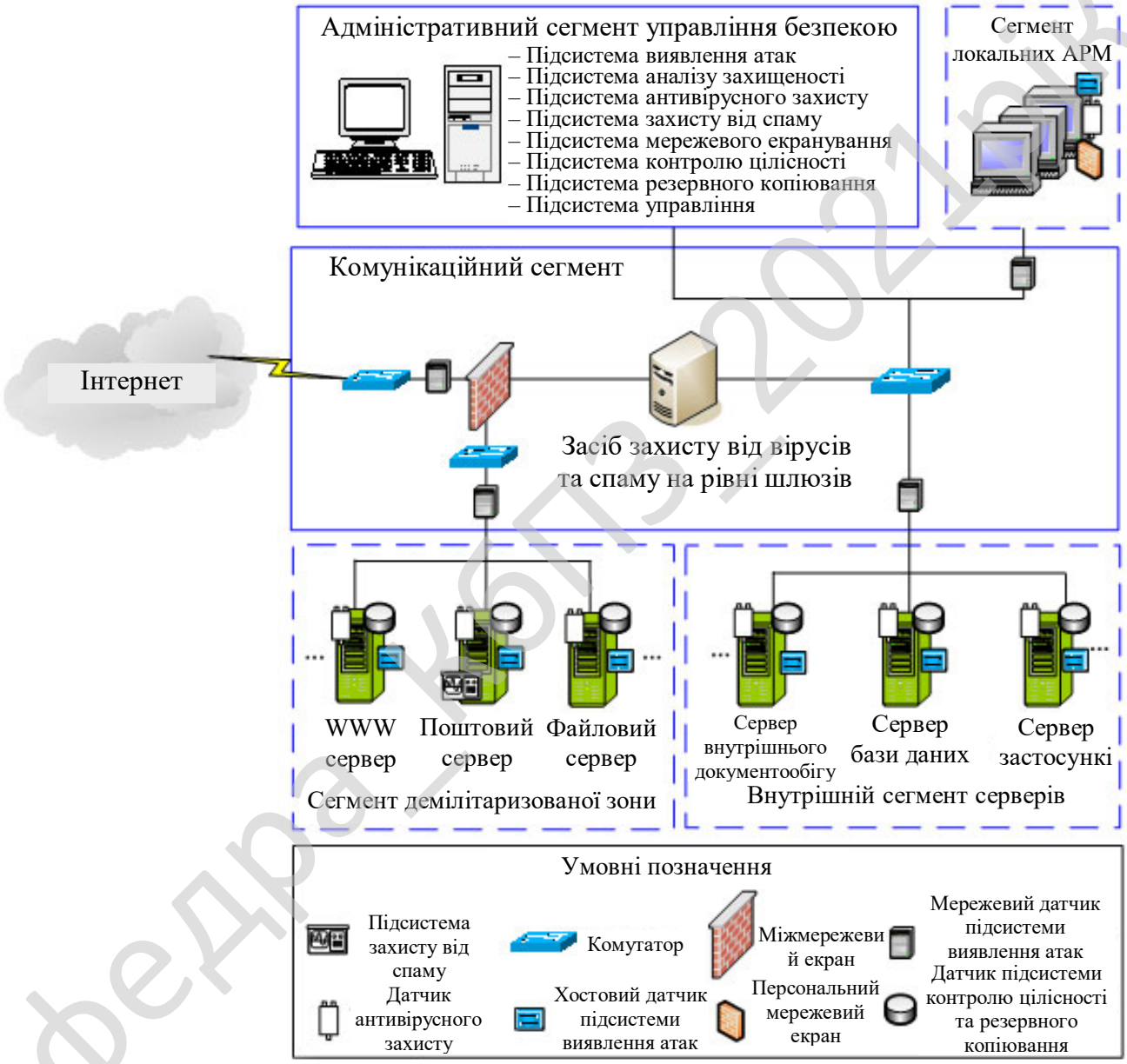

Рисунок 3.1 – Структурна схема системи

Важливо відзначити, що впровадження такої системи хмарного антивірусного забезпечення являє собою складний багатоступінчастий процес, що може містити в собі наступні етапи:

– аудит інформаційної безпеки комп'ютерної мережі, що спрямований на збір вихідної інформації, необхідної для розробки плану впровадження системи хмарного антивірусного забезпечення;

– формування вимог до системи хмарного антивірусного забезпечення, призначеної для захисту комп'ютерної мережі. На даному етапі формується технічне завдання на впровадження системи хмарного антивірусного забезпечення;

– розробка техноробочого проекту по впровадженню системи хмарного антивірусного забезпечення, що містить опис проектних рішень, схем установки, параметрів налаштування системи хмарного антивірусного забезпечення і інших службових даних;

– навчання персоналу організації, відповідального за адміністрування системи хмарного антивірусного забезпечення;

– пусконалагоджувальні роботи, пов'язані з розгортанням системи хмарного антивірусного забезпечення. У рамках даного етапу робіт спочатку створюється пилотна зона, у якій проводиться попереднє тестування впроваджуваної системи хмарного антивірусного забезпечення, після якого реалізується повномасштабне впровадження комплексу захисту в комп'ютерної мережі;

– технічний супровід системи хмарного антивірусного забезпечення, у рамках якого вирішуються питання, пов'язані з обслуговуванням системи в процесі її експлуатації.

Склад етапів, а також їхня тривалість залежить від розмірності комп'ютерної мережі, яка захищається, а також від масштабів впровадження системи хмарного антивірусного забезпечення. Роботи, пов'язані із впровадженням і експлуатацією СОА можуть проводитися як власними силами підприємства, так і із залученням зовнішніх організацій, що спеціалізуються на наданні послуг в області інформаційної безпеки.

3.3 Розробка функціональної схеми

Хмарне антивірусне забезпечення, що розроблений у результаті виконання магістерської роботи – захист вашого комп'ютера від шкідливих програм, що включає базові функції забезпечення безпеки вашого ПК. Хмарне антивірусне забезпечення використовує новітні технології захисту, завдяки якому забезпечується безпека й стабільна роботу комп'ютера.

Основні функції хмарного антивірусного забезпечення, що розроблене:

– 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 хмарного антивірусного забезпечення 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, заснована на аналізі поводження програм.

– Самозахист хмарного антивірусного забезпечення, 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.

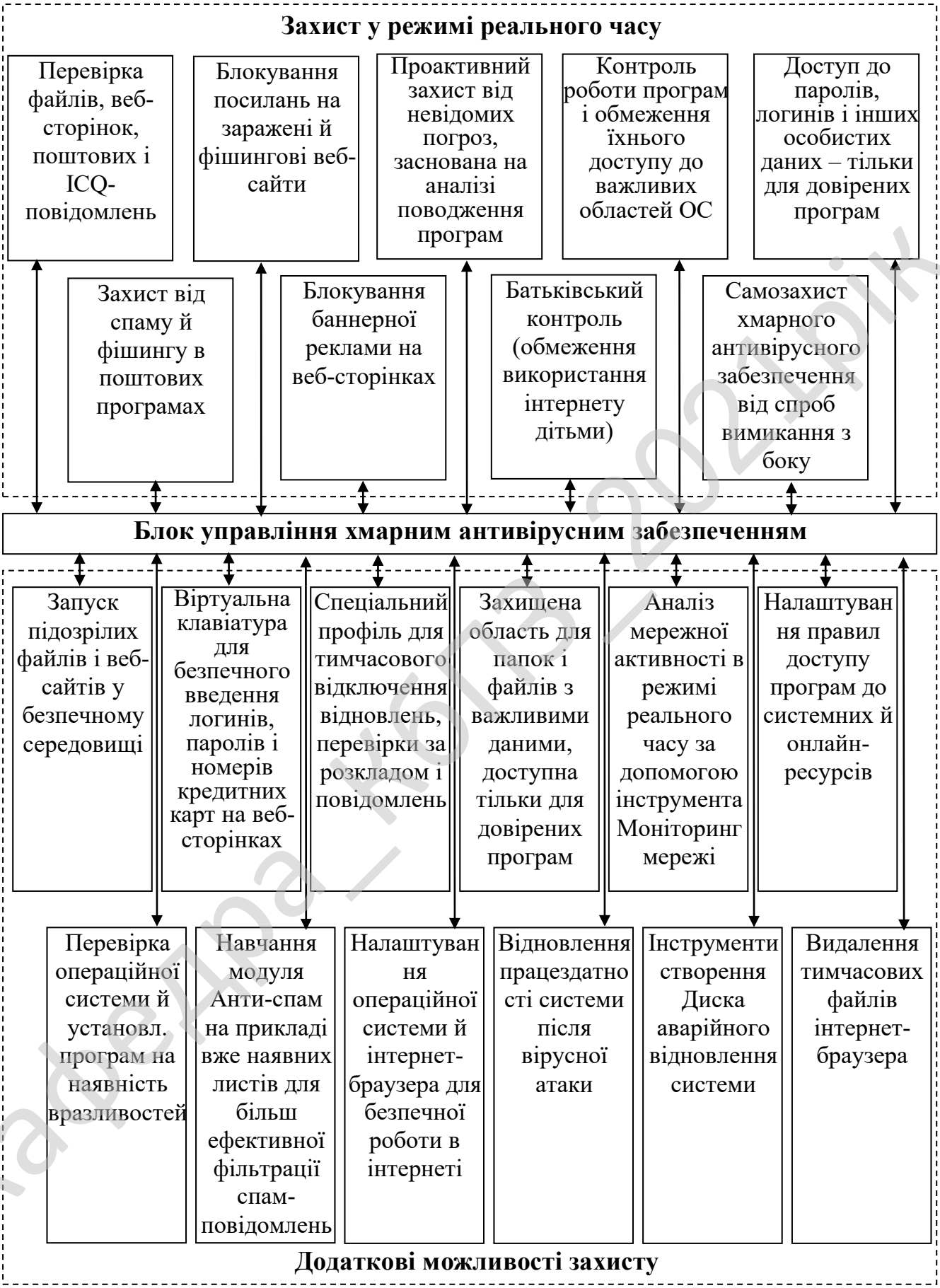

Рисунок 3.2 – Функціональна схема системи

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

– Відновлення працездатності системи після вірусної атаки.

– Видалення тимчасових файлів інтернет-браузера.

На рисунку 3.2 зображена 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 магістерської роботи, наведена на рисунку 3.3.

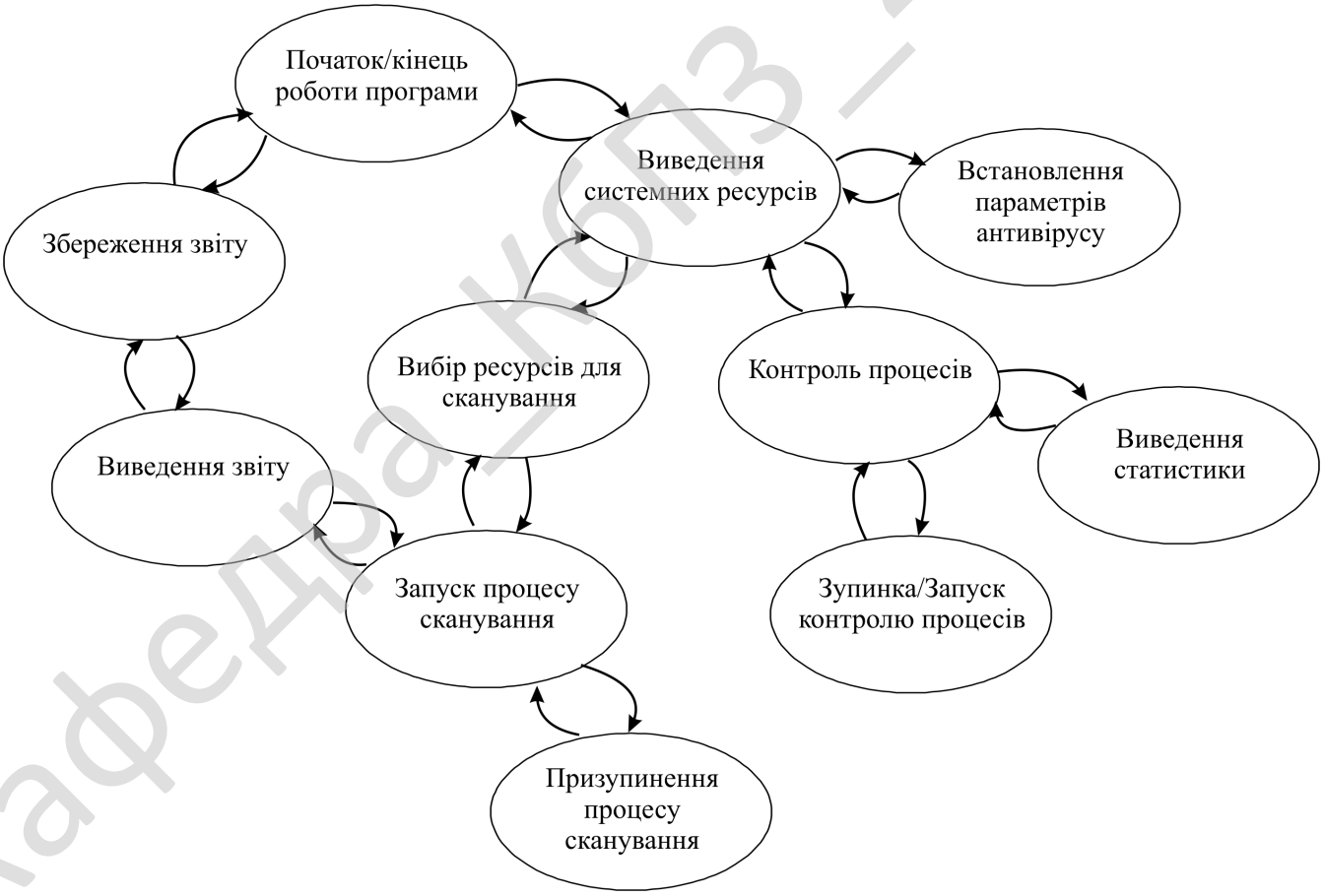

Рисунок 3.3 – Діаграма взаємодії процесів

Першим процесом у розробленій системі являється процес виведення системних ресурсів.

Після нього користувач може перейти до наступних процесів:

– встановлення параметрів хмарного антивірусного забезпечення;

– контроль процесів;

– вибір ресурсів для сканування.

Контроль процесів пов’язаний з наступними процесами:

– виведення статистики;

– зупинки/запуску контролю процесів.

Після вибору ресурсів для сканування слідує процес запуску сканування, що в свою чергу пов’язаний з процесами:

– призупинення сканування;

– виведення звіту;

– збереження звіту.

Таким чином розглянувши структурну схему, функціональну схему та діаграму взаємодії процесів перейдемо до опису блок-схем програмного забезпечення та алгоритмів їх функціонування.

4 Реалізація проекту. Розрахунки і експериментальні дані, що підтверджують правильність проектних рішень

4.1 Блок–схеми та опис алгоритмів функціонування системи

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

Спершу відбувається виведення основного вікна програми.

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

Далі користувач обирає сканувати йому систему, або ні.

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

У іншому випадку, відбувається вибір дисків для сканування.

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

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

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

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

Якщо він вирішує його проводити, то відбувається вивід вікна контролю процесів. У іншому випадку користувач переходить на вікно закінчення роботи з програмним продуктом.

Після виведення вікна контролю процесів, відбувається виведення статистики дій процесів, які функціонують у системі. Після цього користувач обирає працювати йому далі з антивірусом, або завершити роботу з програмою.

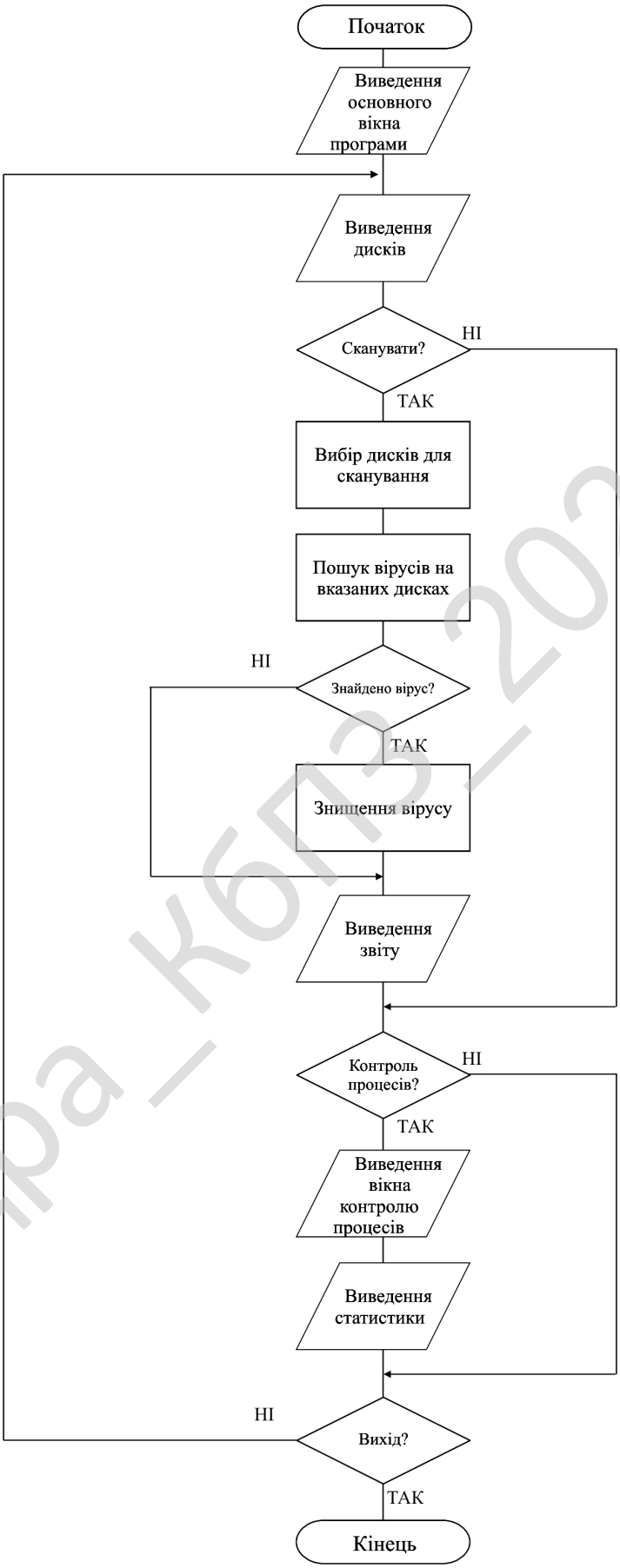

Рисунок 4.1 – Блок-схема роботи основної програми

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

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

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

– Використання скороченої маски.

– Криптоаналіз.

– Статистичний аналіз.

– Евристичний аналіз.

– Емуляція.

Розглянемо кожний із цих методів докладніше.

Пошук за "сигнатурами"

Сигнатура – це унікальний "рядок" байт, що однозначно характеризує ту або іншу шкідливу програму. Сигнатурний пошук, у тій або іншій модифікації, використовується для виявлення вірусів та інших шкідливих програм, п очинаючи з найперших антивірусних програм і дотепер. Незаперечне достоїнство сигнатурного пошуку – швидкість роботи (при використанні спеціально розроблених алгоритмів) і можливості детектування декількох вірусів однією сигнатурою. Недолік – розмір сигнатури для впевненого детектування повинен бути досить великий, як мінімум 8-12 байт (звичайно для точного детектування використовуються набагато більш довгі сигнатури, до 64 байт), отже, розмір антивірусної бази буде досить великим. Крім цього, останнім часом більшу поширеність одержали шкідливі програми, написані на мовах високого рівня

(C++, Delphi, Visual Basic), а в таких програм є окремі частини коду, які практично не змінюються (так звана Run Time Library). Неправильно обрана сигнатура неминуче приведе до помилкового спрацьовування – детектування "чистого", не зараженого файлу як зараженого вірусом. Як рішення цієї проблеми пропонується використовувати або дуже великі сигнатури або використовувати детектування по деяких областях даних, наприклад, таблиці переміщень (relocation table) або текстові рядки, що не завжди добре.

Пошук за контрольними сумами (CRC)

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

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

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

Для детектування таких вірусів застосовуються наступні методи:

використання скороченої маски, криптоаналіз і статистичний аналіз. Розглянемо ці методи докладніше.

Використання скороченої маски

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

$$S = F(T),$$

де T – базовий код вірусу;

S – зашифровані коди вірусу;

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

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

$$S = F(T),$$

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

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

Криптоаналіз

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

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

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

Статистичний аналіз

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

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

Евристичний аналіз

Коли кількість вірусів перевищила кілька сотень, антивірусні експерти задумалися над ідеєю детектування шкідливих програм, про існування яких

антивірусна програма ще не знає (немає відповідних сигнатур). У результаті були створені так звані евристичні аналізатори. Евристичним аналізатором називається набір підпрограм, які аналізують код файлів, що виконуються, макросів, скриптів, пам'яті або завантажувальних секторів для виявлення в ньому різних типів шкідливих комп'ютерних програм. Існують два принципи роботи аналізатора.

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

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

Емуляція

Технологія емуляції коду програм (або Sandboxing) стала відповіддю на появу великої кількості поліморфних вірусів. Ідея цього методу полягає в тому, щоб емулювати виконання програми (як зараженої вірусом, так і "чистої") у спеціальному "оточенні", що називається також буфером емуляції або "пісочницею". Якщо в емулятор попадає заражений поліморфним вірусом файл, то після емуляції в буфері виявляється розшифроване тіло вірусу, готове до детектування стандартними методами (сигнатурний або CRC пошук).

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

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

База даних антивірусного "движка"

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

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

Підтримка "складних", вкладених об'єктів

За останні кілька років антивірусні "движки" сильно змінилися. Якщо першим антивірусам для того, щоб вважатися першокласною програмою, було досить перевіряти системну пам'ять, файли, що виконуються, й завантажувальні сектори, то вже через кілька років у зв'язку з ростом популярності спеціальних утиліт упакування виконавчих модулів перед розроблювачами виникло завдання

розпакувати упакований файл перед тим, як його сканувати.

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

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

Методи детектування

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

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

Практичне застосування способів виявлення поліморфних вірусів (криптоаналіз і статистичний аналіз, застосування скороченої маски й емуляція), зводиться до вибору найбільш оптимального по швидкодії й обсягу необхідної пам'яті методу. Код більшості вірусів, що самошифруються, досить легко відновлюється процедурою емуляції. Якщо використання емулятора не є

оптимальним рішенням, то код вірусу відновлюється за допомогою підпрограми, що реалізує зворотне перетворення – криптоаналіз. Для детектування емуляції вірусів, що не піддаються емуляції і вірусів, для яких не представляється можливим побудувати зворотне перетворення, використовується спосіб побудови скорочених масок.

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

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

Робота з "складними" об'єктами

Антивірусні "движки" підтримують роботу з величезним числом форматів упакування й архівування. Розроблювачі досить рідко публікують повний (або хоча б досить докладний) перелік підтримуваних форматів. Далі представлена офіційно опублікована інформація про підтримку "складних" форматів в Антивірусі Касперского. В інших антивірусних продуктах список підтримуваних форматів повинен бути приблизно таким же. "Движок" Хмарного антивірусного забезпечення Касперского підтримує роботу з більш ніж 400 різними утилітами впакування файлів, що виконуються, інсталяторів і архиваторов (усього більше 900 модифікацій, за станом на травень 2003). Серед них пакувальники файлів, що

виконуються, і системи шифрування. Самі популярні з них: Diet, AVPACK, COMPACK, Epack, ExeLock, ExePack, Expert, HackStop, Jam, LzExe, LzCom, PaquetBuilder, PGMPAK, PkLite, PackWin, Pksmart, Protect, ProtEXE, RelPack, Rerp, Rjcrush, Rucc, Scramb, SCRNCH, Shrink, Six-2-Four, Syspack, Trap, UCEXE, Univac, UPD, UPX (кілька версій), WWPACK, ASPack (кілька версій), ASProtect (кілька версій), Astrum, BitArts, BJFnt, Cexe, Cheaters, Dialect, DXPack, Gleam, CodeSafe, ELFCrypt, JDPack, JDProtect, INFTool, Krypton, Neolite, ExeLock, NFO, NoodleCrypt, OptLink, PCPEC, PEBundle, PECompact (кілька версій), PCShrink, PE-Crypt, PE-Diminisher, PELock, PEncrypt, PE-Pack (кілька версій), PE-Protect, PE-Shield, Petite, Pex, PKLite32, SuperCede, TeLock, VBox, WWPack32, XLok, Yoda.

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

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

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

Слід зазначити, що подібні міркування носять аж ніяк не теоретичний характер. Так, широко відома троянська програма Backdoor.Rbot, що поширювалася упакованою безліччю різних програм (Ezip, Exe32Pack, ExeStealth, PecBundle, PECompact, FSG, UPX, Morphine, ASPack, Petite, PE-Pack, PE-Diminisher, PELock, PESpin, TeLock, Molebox, Yoda, Ezip, Krypton і ін.).

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

Механізм відновлення антивірусних баз і їхній розмір

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

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

Евристичний аналізатор

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

На сьогоднішній день евристичний аналізатор дозволяє виявляти шкідливі коди у файлах, що виконуються, секторах і пам'яті, а також нові скрипт -віруси й шкідливі програми для Microsoft Office (і інших програм, що використовують

VBA), і, нарешті, шкідливий код, написаний на мовах високого рівня, таких як Microsoft Visual Basic.

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

Схема роботи антивірусного "движка"

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

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

На рисунку 4.2 зображена блок-схема роботи підпрограми антивірусного "движка".

Вона працює наступним чином.

Спершу відбувається вибір об’єкту для перевірки.

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

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

– за контрольними сумами;

– сигнатурний пошук;

– криптоаналіз:

– евристичний аналіз.

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

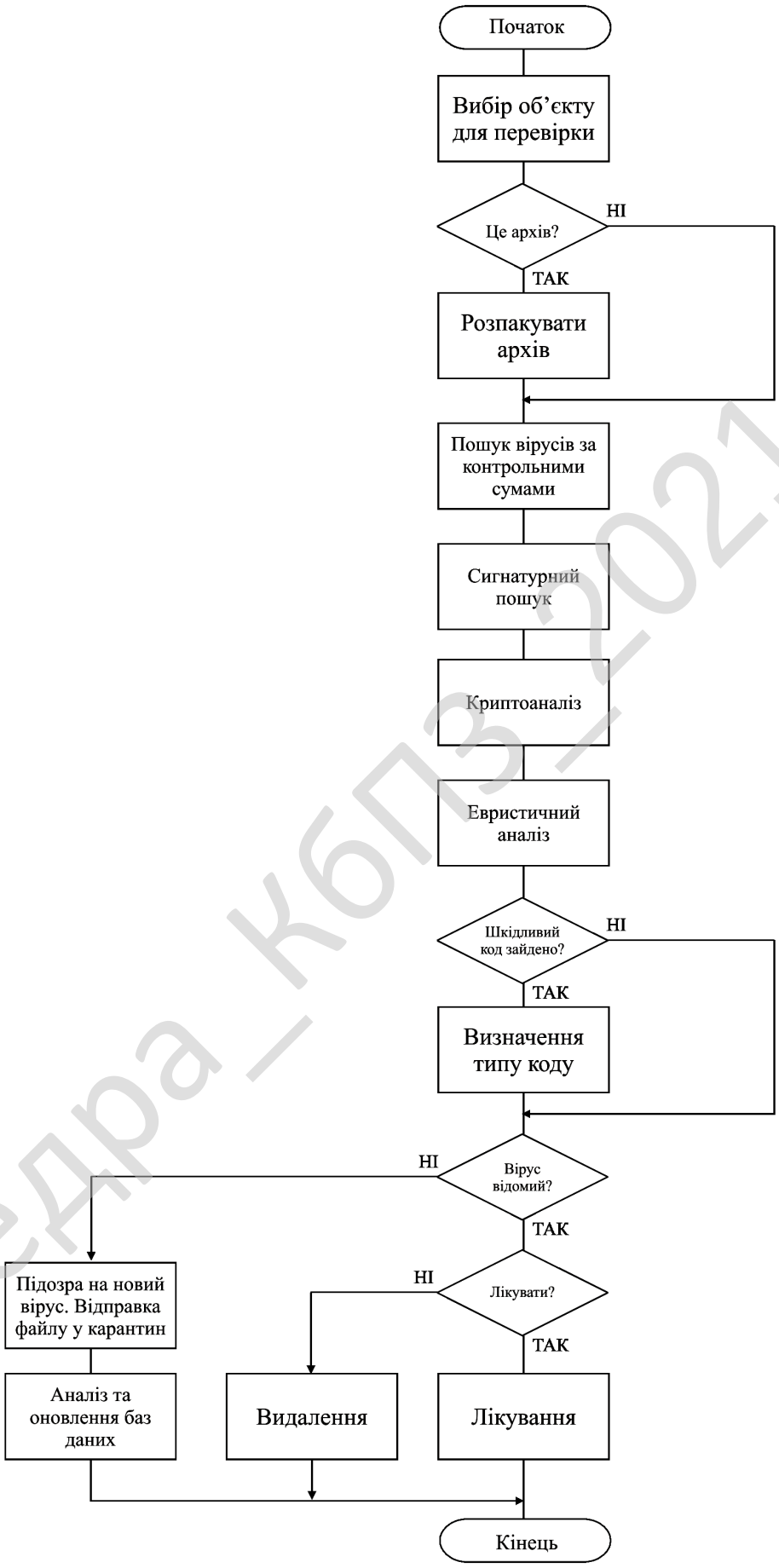

Рисунок 4.2 – Блок-схема роботи підпрограми антивірусного "движка"

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

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

Якщо він обирає лікування, то відбувається спроба вилікувати файл, видаливши з нього шкідливий код. Якщо це неможливо, то файл знищується.

Оригінальні технології в антивірусних "движках"

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

Почати треба з технології iChecker. Ця технологія і її аналоги реалізовані майже в кожному сучасному антивірусі. Слід зазначити, що iChecker – назва, запропонована фахівцями "Лабораторії Касперского". Експерти, наприклад, Panda Software називають її UltraFast. Дана технологія дозволяє домогтися розумного балансу між надійністю захисту робочих станцій (і особливо серверів), і використанням системних ресурсів комп'ютера, що захищається. Завдяки цій технології значно скорочується час завантаження (до 30-40%) операційної системи (у порівнянні із традиційними антивірусними захистами) і час запуску додатків при активному антивірусному захисті. При цьому гарантується, що всі файли на дисках комп'ютера були перевірені й не інфіковані. Основна ідея даної технології – не треба перевіряти те, що не змінювалося, і вже було перевірено. Антивірусний "движок" веде спеціальну базу даних, у якій зберігаються контрольні суми всіх перевірених (і не інфікованих) файлів. Тепер, перш ніж віддати файл на перевірку, "движок" підраховує й порівнює контрольну суму

файлу з даними, що зберігаються в базі даних. Якщо дані збігаються, то це значить, що файл був перевірений і повторна перевірка не потрібно. Варто помітити що час, затрачуваний на підрахунок контрольних сум файлу – значно менше, ніж час антивірусної перевірки.

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

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

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

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

Поліморфні віруси

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

Самошифрування й поліморфічність використовуються практично всіма

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

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

Поліморфні розшифровувачі

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

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

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

і т.д.

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

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

Рівні поліморфізму

Існує розподіл поліморфних вірусів на рівні залежно від складності коду, що зустрічається в розшифровувачах цих вірусів. Такий розподіл уперше запропонував доктор Алан Соломон, через якийсь час Весселин Бончев розширив його:

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

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

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

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

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

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

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

Рівень 1: для детектування вірусу досить мати кілька масок;

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

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

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

Рівень 5: неможливість детектування вірусу по масці.

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

Якщо зробити розподіл на рівні з погляду антивірусів, що використовують системи автоматичної розшифровки коду вірусу (емулятори), то розподіл на рівні буде залежати від складності емуляції коду вірусу. Можливо детектування вірусу

й інших прийомів, наприклад, розшифровка за допомогою елементарних математичних законів і т.д.

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

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

– Використання спеціальних прийомів, що утрудняють емуляцію антивірусами.

– Сталість алгоритму розшифровувача.

– Сталість довжини розшифровувача.

Зміна виконуваного коду

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

Рідше цей спосіб застосовується складними завантажувальними вірусами. Такі віруси впроваджують у завантажувальні сектори лише досить коротку процедуру, що зчитує з диска основний код вірусу й передає на нього керування. Код цієї процедури вибирається з декількох різних варіантів (які також можуть бути розведені "порожніми" командами), команди переставляються між собою й т.д.

Ще рідше цей прийом зустрічається у файлових вірусів – адже їм доводиться повністю міняти свій код, а для цього потрібні досить складні алгоритми. На сьогоднішній день відомі всього два таких віруси, один із яких ("Ply") випадковим чином переміщає свої команди по своєму тілу й заміняє їх на команди JMP або CALL. Інший вірус ("TMC") використовує більш складний спосіб – щоразу при зараженні вірус міняє місцями блоки свого коду й даних, вставляє "сміття", у своїх асемблерних інструкціях установлює нові значення

оффсетів на дані, міняє константи й т.д. У результаті, хоча вірус і не шифрує свій код, він є поліморфним вірусом – у коді не присутній постійного набору команд. Більш того, при створенні своїх нових копій вірус міняє свою довжину.

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм RC5, який являє собою бло ковий шифр із безліччю параметрів: розміром блоку, розміром ключа й числом раундів. В алгоритмі RC5 передбачені три операції: XOR, додавання й циклічні зрушення. На більшості процесорів операції циклічного зрушення виконуються за постійний час, змінні циклічні зрушення являють собою нелінійну функцію. Циклічні зрушення залежать як від ключа, так і від даних.

В RC5 використовується блок змінної довжини, але в приводиться прикладі, що буде розглянутий, 64-бітовий блок даних. Шифрування використовує $2r+2$ залежних від ключа 32-бітових слів – S_0, S_1, S_2,... S_{2r+1} – де r – число раундів. Для шифрування спочатку потрібно розділити блок відкритого тексту на два 32-бітових слова: A и B. (При впакуванні байтів у слова в алгоритмі RC5 дотримується угода про прямий порядок (little-endian) байтів: перший байт займає молодші біти регістра A й т. ін.) Потім:

$$A=A + S_0$$

$$B = B + S_0$$

Для i від 1 до r:

$$A = ((A \oplus \quad B) <<< B) + S_{2i}$$

$$B = ((B \oplus \quad A) <<< A) + S_{2i+1}$$

Вихід перебуває в регістрах A и B.

Розшифрування теж нескладно. Потрібно розбити блок відкритого тексту на два слова, A й B, а потім:

Для i від r до 1 із кроком -1:

$$B = ((B - S_{2i+1}) >>> A) \oplus A$$

$$A = ((A - S_{2i}) >>> B) \oplus B$$

$$B = B - S_i$$

$$A = A - S_0$$

Символом «>>>» позначене циклічне зрушення вправо**.** Звичайно ж**,** всі додавання й вирахування виконуються по модулю 2^{32}.

Створення масиву ключів складніше, але теж прямолінійно. Спочатку байти ключа копіюються в масив L із 32-бітових слів, доповнюючи при необхідності заключне слово нулями. Потім масив S ініціалізується за допомогою лінійного конгруентного генератора по модулю 2^{32}:

$$S_0 = P$$

Для i від 1 до $2(r + 1) - 1$:

$$S_i = (S_{i-1} + Q) \bmod 2^{32}$$

де P = 0xb7e15163 і Q = 0x9e3779b9.

Нарешті, потрібно підставити L в S:

$$i = j = 0$$

$$A = B = 0$$

Виконати $3n$ раз (де n – максимум від $2(r + 1)$ і c):

$$A = S_i = (S_i + A + B) <<< 3$$

$$B = L_i = (L_i + A + B) <<< (A + B)$$

$$i = (i + 1) \bmod 2(r + 1)$$

$$i = (j + 1) \bmod c$$

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

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

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

Головне вікно програми зображене на рисунку 5.1.

Рисунок 5.1 – Головне вікно програми

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

1. SpyWare, AdvWare програм.

2. Руткітів та інших шкідливих програм.

3. Мережних і поштових хробаків.

4. Троянських програм.

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

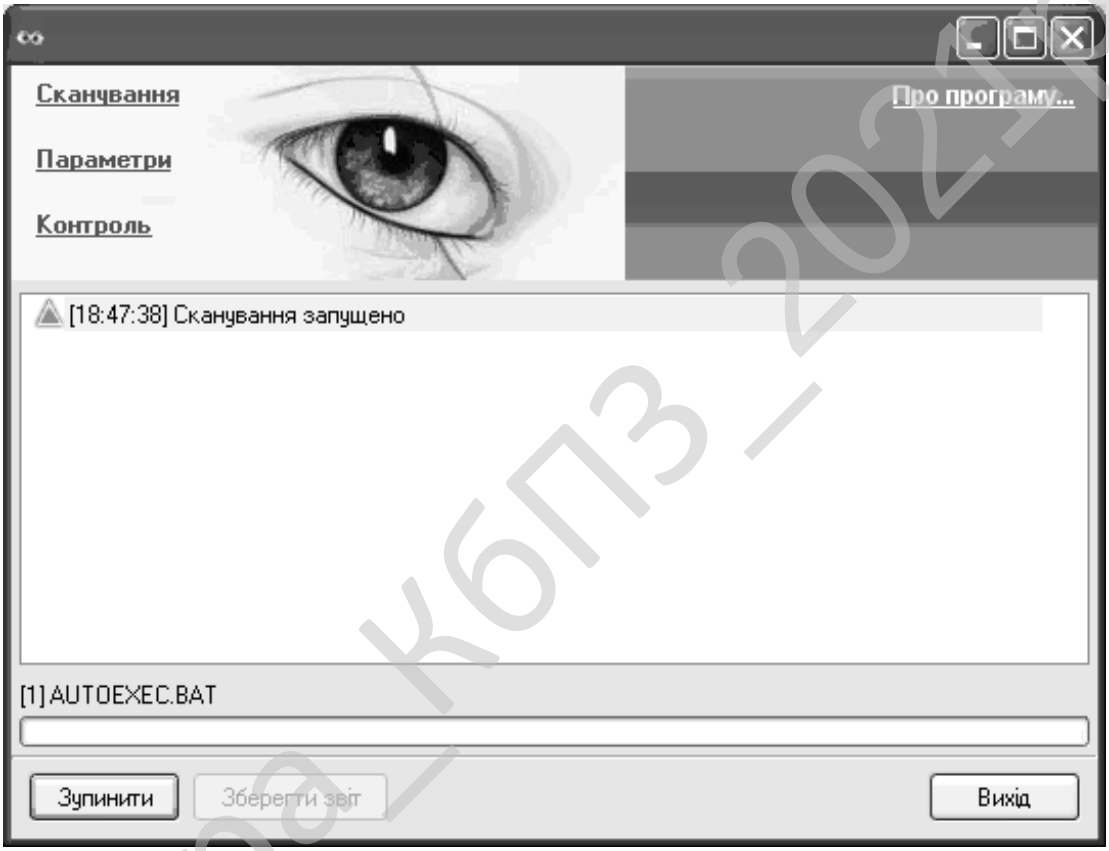

Рисунок 5.2 – Сканування

На ньому розташовані наступні посилання:

- Сканування.
- Параметри.
- Контроль.
- Про програму…

Та кнопки:

– Сканувати.

– Зберегти звіт.

– Вихід.

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

Для запуску процесу сканування слід вибрати диски, що необхідно перевірити та натиснути кнопку «Сканувати», процес сканування зображено на рисунку 5.2. На рисунку 5.3 зображено звіт, що виводиться в кінці сканування.

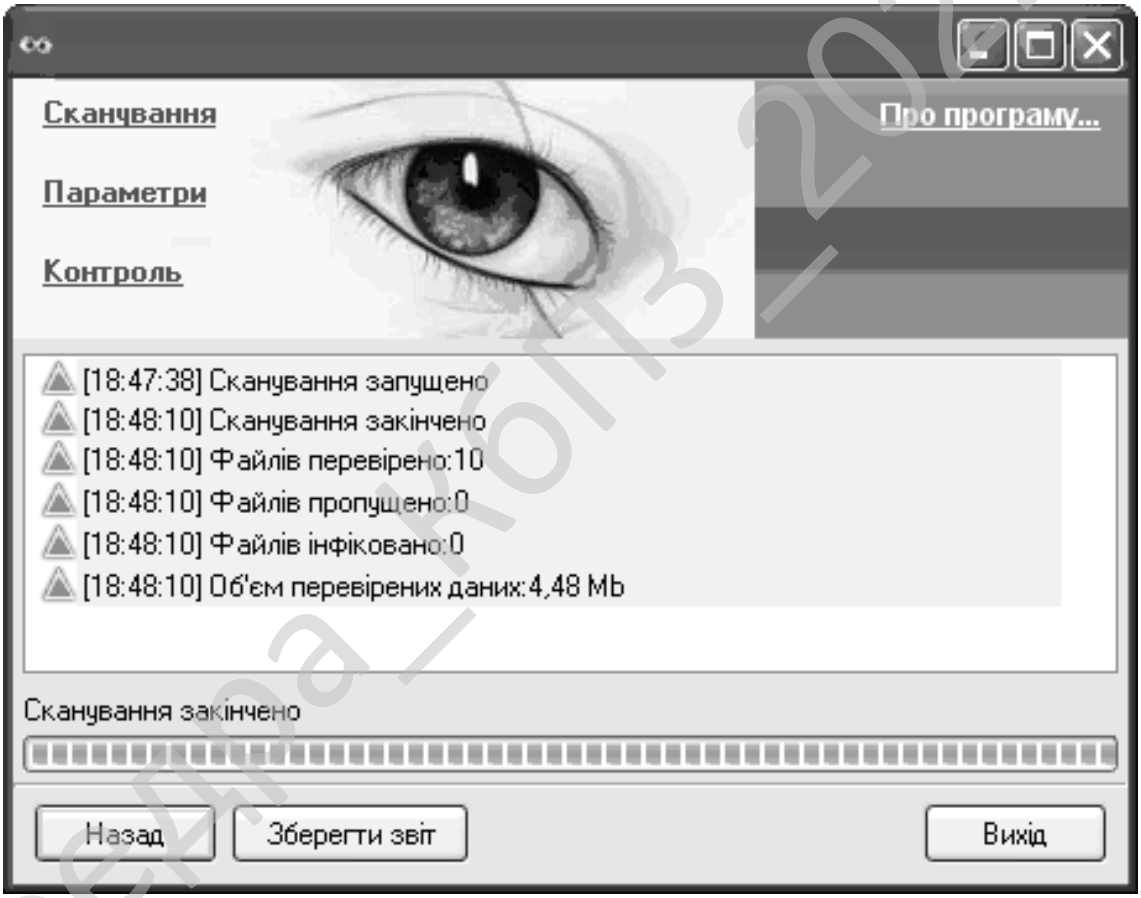

Рисунок 5.3 – Звіт

Для встановлення параметрів програми необхідно натиснути на посилання «Параметри» (рисунки 5.4 – 5.7).

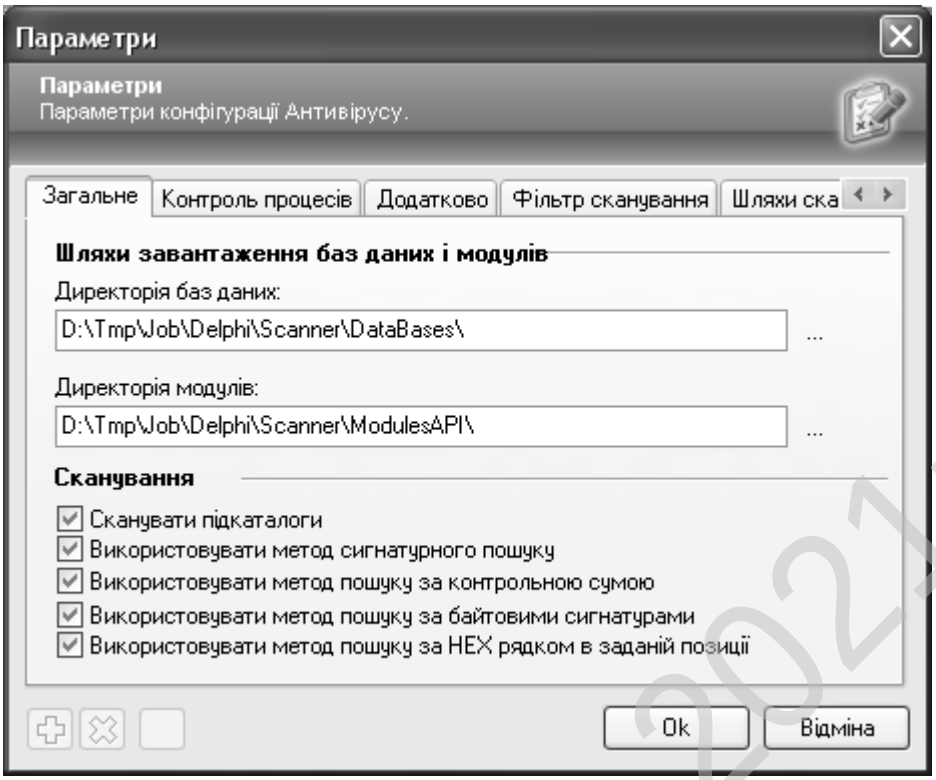

Рисунок 5.4 – Параметри (загальне)

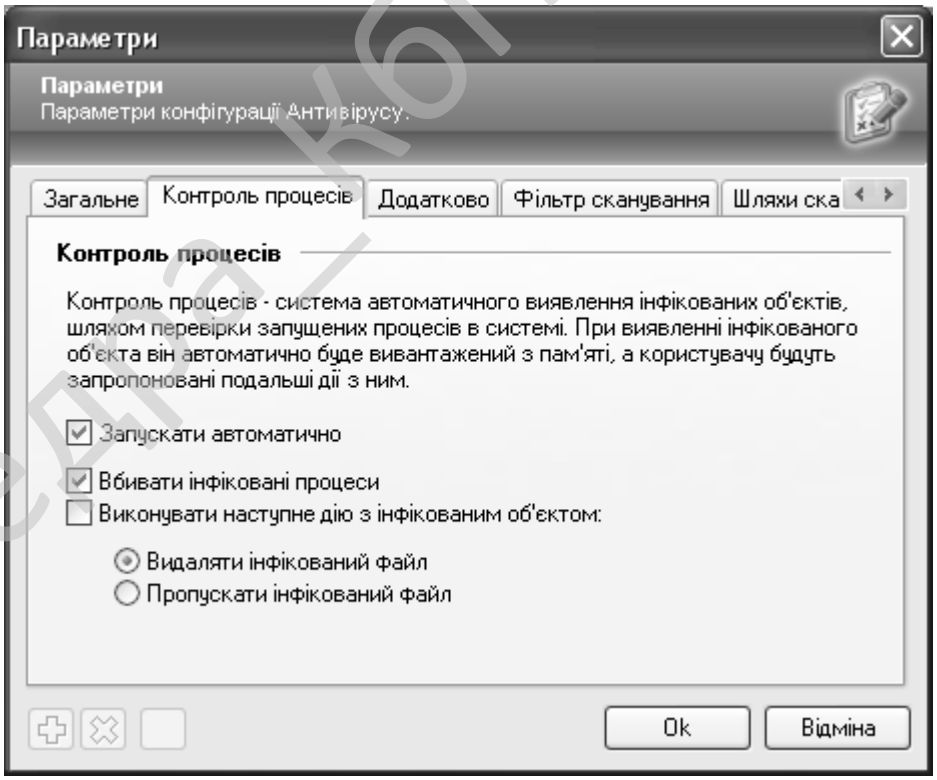

Рисунок 5.5 – Параметри (контроль процесів)

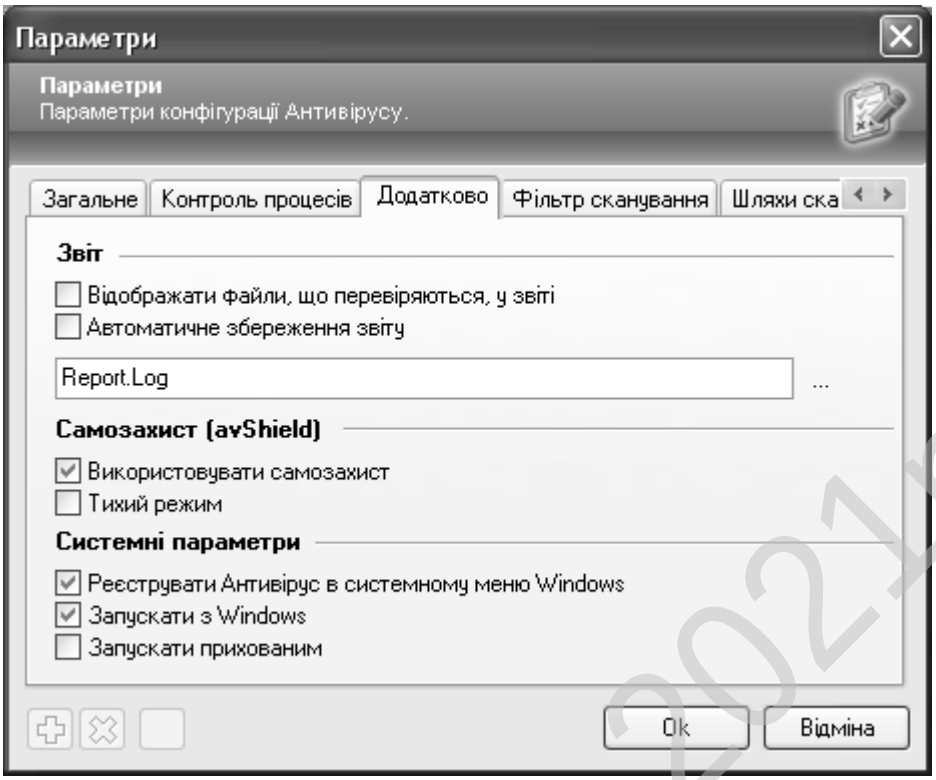

Рисунок 5.6 – Параметри (додатково)

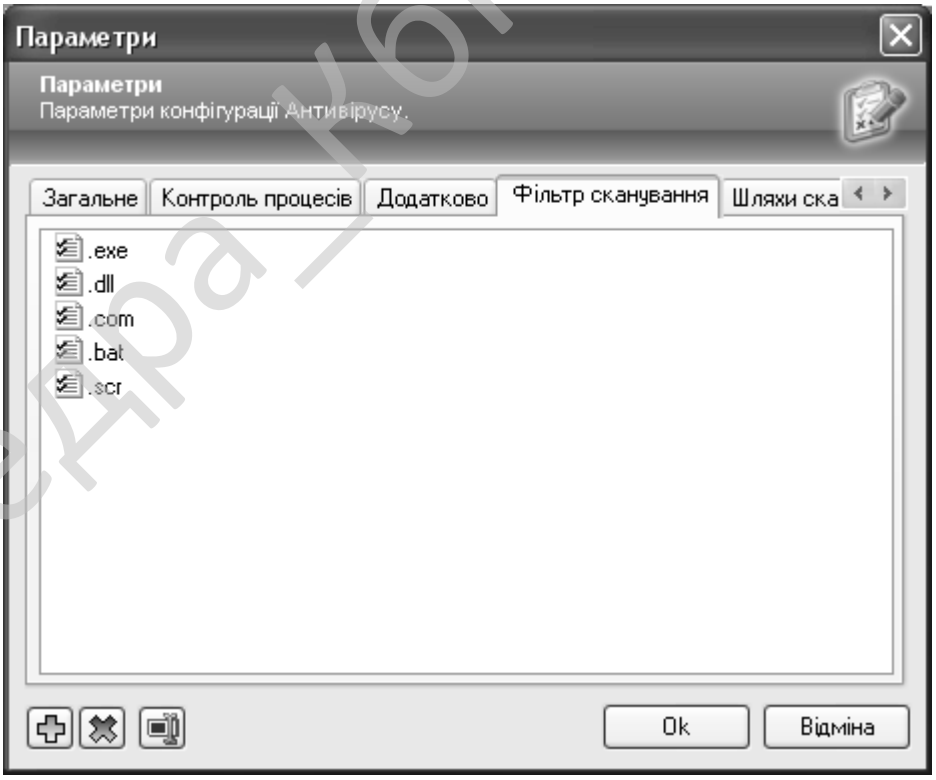

Рисунок 5.7 – Параметри (фільтр сканування)

В розробленій програмі є система Контроль процесів, призначена для автоматичного виявлення інфікованих об'єктів, шляхом перевірки запущених процесів в системі. При виявленні інфікованого об'єкта він автоматично буде вивантажений з пам'яті, а користувачу будуть запропоновані подальші дії з ним. Для перегляду роботи даної системи слід натиснути посилання «Контроль», після чого відкриється вікно, зображене на рисунку 5.8.

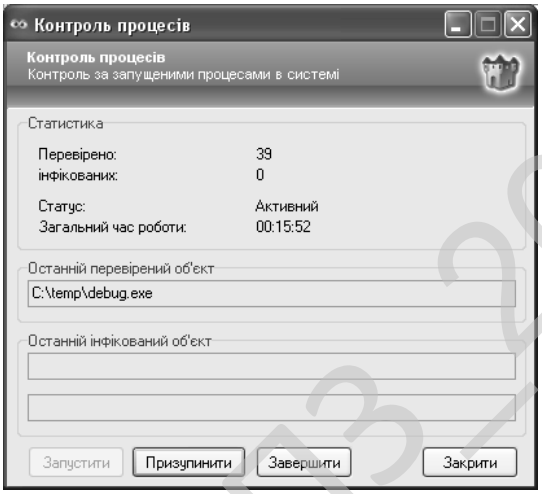

Рисунок 5.8 – Контроль процесів

Коротку довідку про розроблену програму можна переглянути на рисунку 5.9.

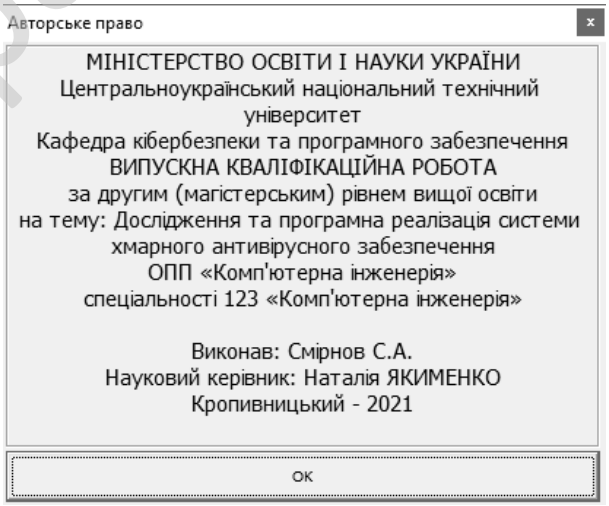

Рисунок 5.9 – Вікно «Про програму…»

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи хмарного антивірусного забезпечення.

Метою розробки є дослідження та програмна реалізація системи хмарного антивірусного забезпечення.

Об'єктом дослідження є процес хмарного антивірусного забезпечення.

Предметом дослідження є методи хмарного антивірусного забезпечення.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод хмарного антивірусного забезпечення.

– Розроблено вітчизняний продукт хмарного антивірусного забезпечення, який має більш широкі можливості, на відміну від існуючих аналогів.

7 ДАНІ ПРО ЕКОНОМІЧНУ ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ

7.1 Техніко-економічне обґрунтування теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Після ознайомлення з підприємством та засобами розробки програмної продукції був розроблений план розробки програми. Був підрахований необхідний час для розробки та впровадження програми. Цей час склав 60 днів (три місяці).

В магістерській роботі було проведене дослідження та виконана програмна реалізація системи хмарного антивірусного забезпечення.

Розроблене програмне забезпечення має достатню надійність і задовольняє усім поставленим умовам, а саме:

а) невеликий розмір;

б) невеликі системні потреби;

в) незалежність від встановлених на комп'ютері баз даних;

г) зручність у користуванні та надійність

Таблиця 7.1 – Початкові дані

Показники	Позначення	Характеристика або величина
1	2	3
1. Кількість розроблених програм період, шт	N	1
2. Кількість екземплярів програм, шт	Ne	20 (2 ост. цифри № зал*10^1)
3. Запланований термін розробки, днів	Fpq	60 (3 місяці)
4. Група задачі підсистеми управління (1-6)	–	1
5. Ступінь новизни задачі (А, Б, В, Г)	–	Б
6. Складність алгоритму (1, 2, 3)	–	2
7. Кількість макетів вхідної інформації	–	3

Продовження табл. 7.1

1	2	3
8. Кількість форм вихідної інформації.	–	4
9. Мова програмування (1-6)	–	2
10. Попередній досвід (1-6)	–	3
11. Гнучкість проекту ПП (1-6)	–	3
12. Детальність проекту ПП (1-6)	_	2
13. Рівень спрацьованості колективу (1-6)	_	2
14. Ступінь вимірності процесів (1-6)	_	3
15. Необхідна надійність програмного забезпечення (1-6)	_	2
16. Розмір бази даних (порівняно з розміром програми) (1-6)	_	2
17. Складність кінцевого програмного продукту (1-6)	_	2
18. Необхідний рівень забезпечення повторного використання (1-6)	_	2
19. Документованість відповідно до планованого життєвого циклу (1-6)	_	2
20. Вимоги до швидкодії ПП (1-6)	_	2
21. Обмеження на розміри основного сховища даних (1-6)	_	2
22. Різноманітність використовуваних обчислювальних платформ (1-6)	_	2
23. Професійний рівень аналітиків (1-6)	_	2
24. Професійний рівень програмістів (1-6)	_	2
25. Постійність складу команди розробників (1-6)	_	2
26. Досвід розробки додатків (1-6)	_	2
27. Досвід роботи з обчислювальною платформою (1-6)	_	2

Продовження табл. 7.1

1	2	3
28. Досвід роботи з мовою і інструментами середовища розробки (1-6)	–	2
29. Досвід роботи з програмними інструментами розробки (1-6)	–	3
30. Розробка ПЗ для декількох серверів одночасно (1-6)	–	2
31. Вимоги до дотримання встановленого графіка робіт (1-6)	–	2
32. Вартість ПЗ у розробника (НМА), грн	–	20000 (2 ост. цифри № зал*10^4)
33. Норматив додаткової зарплати, % :	$Н_д$	10
34. Норматив відрахувань у соціальні фонди, %	Нс	37
35. Норматив загальногосподарських витрат, %	Нг	15
36. Норматив витрат на освоєння нових мов програмування, %	Нп	15
37. Рівень рентабельності програмної продукції, %	Ре	55
38. Ставка податку на додану вартість, %	Ндв	20

7.2 Розрахунок трудомісткості розробки програмної продукції

Значення трудомісткості розробки програмного забезпечення для стадій ТЗ, ЕК, ТП та ВП визначаємо по типовим нормам часу приведеним в додатках МВ. Стадія РП є найбільш тривалою і трудомісткою, що робить значний вплив на інші стадії проекту.

Визначимо трудомісткість розробки ПЗ для стадії РП.

Обчислюємо номінальні трудовитрати, люд-міс.:

$$T_{ном} = A \, \text{Size}^B , \qquad (7.1)$$

де A – коефіцієнт Боема, A=2,45; Size – загальний об'єм відлагодженого програмного коду, тис. рядків; B – показник ступеня, що визначається співвідношенням

$$B = 1{,}01 + 0{,}001\sum W_i \qquad (7.2)$$

де Wi – сумарне значення п'яти показників (МВ, додаток 2), що відображають особливості розробки проекту програмного продукту (ПП) і колективу розробників.

$$B=1{,}01+0{,}001(2{,}43+3{,}64+3{,}38+3{,}95+2{,}73) = 1{,}026$$

$$T_{ном} = 2{,}45 \cdot 2{,}7^{1{,}026} = 6{,}78 \text{ люд-міс.}$$

Визначаємо уточнені (з урахуванням приведених в МВ додатку 3 сімнадцяти додаткових коефіцієнтів) трудовитрати, люд-міс.:

$$T_{уточн}=T_{ном}\, ПВ_j, \qquad (7.3)$$

де $ПВj$ – добуток сімнадцяти додаткових коефіцієнтів, приведених в МВ додатку 3.

$$T_{уточн}= 6{,}78\cdot(0{,}88\cdot0{,}93\cdot0{,}88\cdot0{,}91\cdot0{,}95\cdot1\cdot1\cdot0{,}87\cdot1{,}22\cdot1{,}16\cdot1{,}1\cdot1{,}1\cdot$$

$$\cdot1{,}12\cdot1{,}1\cdot1\cdot1{,}1\cdot1{,}1)=9{,}37 \text{ люд-міс.}$$

Ці коефіцієнти дозволяють диференційовано оцінювати результати роботи програмістів, беручи до уваги швидкодію програми, використання різноманітних обчислювальних платформ і інструментів розробки, взаємодію декількох серверів, вимоги до об'ємів баз даних і ін.

Визначаємо підсумкові трудовитрати по стадії робочий проект, люд-дні:

$$T_{РП} = 0{,}3CT_{уточн}^{0{,}33+0{,}2(B-1{,}01)}S, \qquad (7.4)$$

де C – визначений емпірично коефіцієнт, запропонований авторами методики, (МВ, додаток 4); S – коефіцієнт стиснення (або подовження) графіка робіт %, що дозволяє коректувати терміни розробки ПЗ згідно встановленим вимогам. Вибираємо в межах (25…350)%

$$T_{РП}=0{,}3\cdot2{,}66\cdot9{,}37^{0{,}33+0{,}2(1{,}026-1{,}01)}\cdot100=168 \text{ люд/день}$$

Для зручності визначення загальної трудомісткості на розробку програмного забезпечення результати розрахунків по стадіям зводимо до таблиці 7.2.

Таблиця 7.2 – Визначення трудомісткості розробки програмного забезпечення

Стадії розробки	Трудомісткість за типовими нормами та розрахунками	
	Величина, люд/дні	Підстава
Технічне завдання	9	Д5
Ескізний проект	10	Д6
Технічний проект	9	Д7
Робочий проект	168	Ф 7.1-7.4
Впровадження	13	Д13
Всього	209	–

7.3 Визначення чисельності виконавців і планового фонду зарплати

Чисельність ставок інженерів-програмістів для розробки програмного забезпечення визначається за формулою

$$Ч = \frac{T_{пз} N}{F_{рq} - H_{ев}}, \quad (7.5)$$

де Fpq – плановий фонд робочого часу одного спеціаліста, днів, Тпз – трудомісткість розробки програмного забезпечення люд-дні,

$$Ч = \frac{209 \cdot 1}{60 - 5} = 3{,}8 \text{ ставки}$$

Чисельність інженерів-електронщиків для проведення технічного обслуговування та ремонту комп'ютерних мереж визначається в залежності від наявності технічних засобів і норм витрат часу на виконання профілактичних робіт на протязі року.

Визначаємо затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за період розробки. Результати розрахунку зводимо до таблиці 7.3

Таблиця 7.3 – Затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за розрахунковий період

Найменування обладнання	Профілактичне обслуговування			
	Кількість хв. на один. обл.	Кількість обладнання	Затрати часу в хв.	Затрати часу в год.
Системний блок ПК	385	12	4620	77
Монітор	160	12	1920	32
Клавіатура	140	12	1680	28
Маніпулятор «мишка»	30	12	360	6
Принтер матричний	185	1	185	3
Принтер лазерний	355	2	710	12
Принтер струминний	300	1	300	5
Сканер	155	2	310	5
Концентратор-маршрутизатор	155	2	310	5
Кабельні господарства ЛОМ на 1 м. п.	2,5	100	250	4
Кабельне господарство електромережі	48	50	2400	40
Копіювальний апарат	285	2	570	10
Усього за рік:			$З_ч$	227

Час на профілактику обладнання в загальному балансі робочого часу інженерів-електронщиків неповинен складати більше 10%

Виходячи з цього фонд робочого часу інженерів-електронщиків складає:

$$Ф_{др}^{с} = \frac{З_ч \cdot n_{тіс}}{1,2} \quad (7.6)$$

$$Ф_{др}^{с} = \frac{227 \cdot 3}{1,2} = 567,5 \text{ год}$$

Визначаємо необхідну кількість ставок штатного персоналу сектора ТО:

$$Ч_{ел} = \frac{Ф^{с}_{др}}{F_{др} \cdot T_{зм}} \quad (7.7)$$

$Ч_{ел}$=567,5/(60·8)=1,2 ставки

Для забезпечення нормального технічного обслуговування засобів ТО та мереж, необхідно прийняти найбільше ціле значення розрахункової чисельності інженерів – електронщиків.

Чисельність інженерів-системотехніків, адміністраторів мережі, дизайнерів WEB вузлів, системних програмістів (аналітиків), бухгалтерів-економістів визначається за потребою в залежності від функціональних обов'язків. Після визначення чисельності персоналу складається штатний розклад.

Таблиця 7.4 – Розрахунок чисельності штатного персоналу сектору системного та адміністративного обслуговування засобів ОТ та комп'ютерних мереж

Посада	Вид роботи	Час	Кількість штатних одиниць
Адміністратор загальної мережі, аналітик	Адміністрування локальної мережі, поштового та серверу DNS (ОС FreeBSD), маршрутизатора Cisco, серверу доступу АДСЛ (ОС Linux), Wi-Fi налаштування ADSL, VPN, PPPoE, Frame Relay	2	0,5
	Налаштування і конфігурування базової станції безпровідного зв'язку (CMTS)	0,5	
	Розробка та впровадження проектів з організації зв'язку між віддаленими об'єктами, ЛОМ	0,5	
	Забезпечення цілодобової роботи зв'язку клієнтів до мережі Інтернет	1	
Всього		4	

Продовження таблиці 7.4

Посада	Вид роботи	Час	Кількість штатних одиниць
Продакт-менеджер	Презентації нової продукції, пошук каналів збуту	1	0,25
	Підтримка постійних клієнтів	0,5	
	Оформлення договорів, ведення тендерів	0,25	
	Контроль взаєморозрахунків з постачальниками	0,25	
Всього		2	
Дизайнер WEB	Розробка концепції оформлення та інтерфейсу сайту, оптимізація дизайну існуючих, проектує їх структуру та навігацію	1	0,25
	Створення графічних і стилістичних елементів сайту	0,5	
	Оформлення банерів і промо-сторінок	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	
Інженер верстальник	Розробка та верстка макетів рекламної продукції та технічної документації	1	0,25
	Верстка друкованих видань	0,5	
	Додрукова підготовка макетів	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	

Складемо штатний розклад виконавців:

Таблиця 7.5 – Штатний розклад виконавців

Посада	Кількість ставок	Середньо-місячний оклад, грн.	Всього за період роз-робки, грн.
Керівник (ІТ-менеджер)	1	12000	36000
Продакт-менеджер	0,25	8000	6000
Інженер-програміст	3,8	8000	91200
Інженер-електронщик	1,2	6000	21600
Інженер-системотехнік	0,25	6000	4500
Адміністратор мережі	0,5	6000	9000
Системний програміст	0,25	6000	4500
Дизайнер WEB	0,25	8000	6000
Інженер-верстальник	0,25	6000	4500
Бухгалтер-економіст	0,5	10000	15000
Всього за період розробки	$R_{сп}$=8,25	-	$Ф_{роб}$=198300

Розрахуємо середньоденну зарплату одного виконавця:

$$З_{сд} = \frac{Ф_{роб}}{R_{сп} F_{рд}}, \quad (7.8)$$

де Фроб – загальна сума зарплати за плановий період, грн.

$$З_{сд} = \frac{198300}{8,25 \cdot 60} = 400 \; грн.$$

7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника

Балансова вартість будівель визначається з урахуванням кількості робочих місць виконавців, питомої площі на одне робоче місце, та вартості одного квадратного метра виробничої площі

$$Б_{уд} = R^1_{сп} S_у Ц_{пл}, \quad (7.9)$$

де $R^1_{сп}$ – кількість робочих місць виконавців, шт. Приймаємо 8 робочих місць. Sу – питома площа на одне робоче місце, $м^2$; Цпл – вартість одного квадратного метра площі, грн.

Згідно даних ТОВ науково-дослідницького консалтингового підприємства «Пекторaль» ціна одного квадратного метра площі новобудови, вік якої не перевищує 25 років, по місту складає 800…1600 $у.о./м^2$. Враховуючи, що курс складає 1 у.о. = 25 грн. приймаємо для розрахунку вартість одного метра квадратного рівною 20000 грн./м^2. На кожне робоче місце у середньому потрібно 8 $м^2$. З урахуванням цього:

$$Б_{уд} = 8 \cdot 8 \cdot 20000 = 1280000 \text{ грн.}$$

Вартість передавальних пристроїв складає 10% від вартості будівель, і у даному випадку вона складе: 128000 грн.

Балансова вартість інвентарю розраховується за нормою 3500 грн на одне робоче місце. Тобто

$$І_{нв} = R^1_{сп} \cdot Ц_м, \quad (7.10)$$

де Цм – ціна меблів для одного робочого місця, грн.

$$І_{нв} = 8 \cdot 3500 = 28000 \text{ грн}$$

Балансова вартість обчислювальної техніки визначається по оптовим цінам постачальника з врахуванням витрат на транспортування.

Специфікація на обчислювальну техніку наведена в таблиці 7.7. Дані по оптовій ціні на обладнання та комплектуючі вибирались по прайсу Інтернет магазину Компбест за 30.10.20 – джерело https://compbest.com.ua.

Таблиця 7.6 – Специфікація

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Персональний комп’ютер		10947
Системний блок		7347
Процесор	Socket FM2+ AMD Athlon X4 845 3,5-3,8 GHz 4C 2 MB 65 Вт BOX	1350
Системна плата	ASRock FM2+ FM2A68M-DG3+ A68H/DDR3 1600/DVI+D-Sub+PCI-E16x/4 SATA/2xUSB 3.0/5.1/1G	1200
Відеокарта	GIGABYTE GV-N710D3-1GL GeForce GT710 1 GB DDR3 954/1800 MHz 64-bit D-Sub+HDMI+DVI	1150
Жорсткий диск	HDD Seagate Barracuda 750 Gb 7200 32Mb SATAII ST3750528AS (ST3750528AS)	1200
Оперативна пам’ять	DIMM 1024Mb DDR3 PC3-10600 CL9 Transcend JetRam 1333Mhz, 128M x 64, non-Reg., no-ECC , CL 9 (2 модулі)	900
DVD-привод	DVD -RW/+RW , LG SATA SuperMulti Bulk 22x, SecurDisc, black	416
Корпус	GRESSO GE-7525, 500W (120mm big fan), 2xIDE, full-ATX,БЖ 2xSATA, 1xFDD, Air Duct, 2xUSB 2.0, Mic+Audio, silver/black	911
Кардрідер внутрішній	USB 2.0 Card reader STORM CR -35U1A4-B, int. 3.5", 1*USB2.0+AUDIO+1394, multi: All Type Cards, black	220
інше	Клавіатура, мишка	-

Продовження таблиці 7.6

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Монітор	LG W2363V-WF Wide LCD 2ms, 70 000:1, 300кд/м2, 170/160, D-Sub / Glossy White	3600
Принтер лазерний	Canon i-SENSYS LBP6030W	2700
Принтер струменевий	Epson Stylus Photo P50 (C11CA45341) + USB cable	5500
Копіювальний апарат	Canon i-SENSYS MF217W with Wi-Fi	5965

Витрати на транспорт, монтаж та випробування можуть бути прийняті в межах до 10% від оптової ціни.

Для визначення необхідної кількості капітальних вкладень складемо таблицю 7.8.

Таблиця 7.7 – Балансова вартість обчислювальної техніки

Найменування обчислювальної техніки	Кількість, шт.	Ціна за одиницю, грн.	Витрати на транспортування, монтаж та випробовування.	Загальна вартість, грн.
Персональні комп’ютери	8	10947	8757,6	96333,6
Принтер лаз.	2	2700	540	5940
Принтер струм.	1	5500	550	6050
Копіюв. апарат	1	5965	596,5	6561,5
Всього	–	–	–	114885,1

Таблиця 7.8 – Вартість основних фондів та амортизаційні відрахування розробника

Групи та види основних фондів	Балансова вартість, грн.	Амортизація	
		Норма, %	Відрахування, грн.
1	2	3	4
Група 3			
1. Будівлі	1280000	-	-
2. Передавальні пристрої	128000	-	-
Всього по групі	1408000	5	70400
Група 4			
3. Обчислювальна техніка	114885	-	-
Всього по групі	114885	50	57442,5
Група 5			
4. Вимірювальні пристрої	5190	-	-
5. Господарський інвентар	28000	-	-
Всього по групі	33190	25	8297,5
Нематеріальні активи			
6. Нематеріальні активи	40000	10	4000
Разом	$К_р$= 1596075		$А_р$= 140140

7.5 Визначення собівартості розробки та ціни програмної продукції

Визначимо основну зарплату виконавців

$$З_о = \frac{З_{сд} \cdot T_{пз}}{N_e}, \quad (7.11)$$

де Ne – Кількість екземплярів програм, шт.

$$З_о = 400 \cdot 209/40 = 2090 \text{ грн}$$

Визначимо додаткову зарплату (оплата відпусток, виконання державних та суспільних обов'язків) на рівні 10%

$$З_д = З_о \cdot Н_q \cdot 0{,}01, \quad (7.12)$$

де $Н_q$ – норматив додаткової зарплати, %

$$З_д = 2090 \cdot 10 \cdot 0{,}01 = 209 \text{ грн}$$

Відрахування на соціальні потреби за нормативом Нс=37% від суми основної та додаткової зарплати

$$С_{оц} = 0{,}01 \cdot Н_с (З_о + З_д), \quad (7.13)$$

де $Н_с$ – відрахування на соціальні потреби, %

$$С_{оц} = 0{,}01 \cdot 37(2090+209) = 851 \text{ грн}$$

Визначимо загальногосподарські витрати (електроенергію, ремонт і утримання приміщень і т.д) за нормативом Нг=15% від основної зарплати

$$Г_{осп} = З_о \cdot Н_г \cdot 0{,}01, \quad (7.14)$$

де $Н_г$ – загальногосподарські витрати, %

$$Г_{осп} = 2090 \cdot 15 \cdot 0{,}01 = 314 \text{ грн}$$

Визначимо витрати на матеріали для розробки програмної продукції за нормами споживання та діючими цінами за одиницю виміру:

$$З_М = (З_{М1} + З_{М2} + З_{М3})/N_е, \quad (7.15)$$

де $З_{М1}$ – вартість паперу, грн., $З_{М2}$ – вартість запам'ятовуючих пристроїв, грн., $З_{М3}$ – вартість фарби, картріджей, тонеру, грн., $N_е$ – кількість екземплярів програм, шт.

Згідно виданих викладачем норм $n_{міс}$ приймаємо 0,33 пачки паперу на місяць розробки. Тоді, враховуючи, що вартість пачки паперу складає $Ц_п$=105 грн., визначаємо вартість паперу за період розробки $N_м$ =3 міс:

$$З_{М1} = Ц_п \cdot N_м \, n_{міс}. \quad (7.16)$$

$$З_{М1} = 105 \cdot 3 \cdot 0{,}33 = 105 \text{ грн}.$$

Згідно виданих викладачем норм до вартості запам'ятовуючих пристроїв входить вартість CD дисків в кількості, що дорівнює кількості екземплярів програм та одного DVD диска для збереження резервної копії програми:

$$З_{М2} = \sum Ц_{д}., \quad (7.17)$$

де Цд – вартість дисків CD/DVD: CDR TDK 700Mb, 80Min, 52x Cake box – 2 грн/шт., DVD-R LG 4,7Gb, 16x speed Cake box – 2 грн/шт.

$$З_{М2} = 41 \cdot 12 = 492 \text{ грн.}$$

Згідно виданих викладачем норм одноразовій заправці підлягають усі друкуючі пристрої і становить:

$$З_{М3} = \sum Ц_{з}., \quad (7.18)$$

де: $Ц_з$ – вартість розхідних матеріалів друкуючих пристроїв: відновлення та заправка картриджу для Canon i-SENSYS LBP6030W – 574 грн.; картридж для Epson Stylus Photo P50 – 558 грн.; відновлення картриджу для MF217W – 570 грн.

$$З_{М3} = 574+558+570=1702 \text{ грн.}$$

$$З_{М} = (105 + 492 + 1702)/40=57 \text{ грн.}$$

Визначимо витрати на освоєння нових мов програмування або операційних систем за нормативом (Нп = 15%) від основної зарплати виконавців

$$О_{п} = З_{о} \cdot Н_{п} \cdot 0{,}01, \quad (7.19)$$

де Нп – норматив витрат на освоєння нових мов програмування, %

$$О_{п} = 2090 \cdot 15 \cdot 0{,}01=314 \text{ грн}$$

Визначимо витрати на амортизацію основних фондів з урахуванням загальної річної суми амортизаційних відрахувань та кількості екземплярів програм ($N_е$ = 40 прим.)

$$А_{м} = \frac{А_{р} \cdot N_{міс}}{N_{е} \cdot 12}, \quad (7.20)$$

де Ар – загальна річна сума амортизаційних відрахувань, грн.

$$А_{м} = 140140 \cdot 3/(40 \cdot 12)=876 \text{ грн}$$

Повна собівартість ПЗ визначається як сума витрат за попередніми статтями калькуляції

$$C_п = З_о + З_д + С_{оц} + Г_{осп} + З_м + О_п + А_м. \quad (7.21)$$

$$C_п = 2090+209+851+314+57+314+876 = 4711 \text{ грн.}$$

Визначимо плановий прибуток за рівнем рентабельності (Рп) програмної продукції, яка залежить від складності програми та ступеня новизни задачі.

Для даного програмного забезпечення рівень рентабельності складає 55%

$$П_р = 0{,}01 \cdot Р_п \cdot С_п, \quad (7.22)$$

де Рс – рівень рентабельності, %

$$П_р = 0{,}01 \cdot 55 \cdot 4711 = 2591 \text{ грн.}$$

Величини ціна підприємства, податок на додану вартість, відпускна ціна програмної продукції визначаються за формулами, приведеними в таблиці 7.9

Таблиця 7.9 – Нормативна калькуляція собівартості розробки програмного забезпечення задачі

Найменування статей витрат	Позначення	Величина, грн.
1. Основна зарплата виконавців	$З_о$	2090
2. Додаткова зарплата виконавців	$З_д$	209
3. Відрахування на соціальні потреби	$С_{оц}$	851
4. Загальногосподарські витрати	$Г_{осп}$	314
5. Витрати на матеріали	$З_М$	57
6. Освоєння нових операційних систем, мов програмування	$О_п$	314
7. Амортизація основних фондів	$А_м$	876
8. Повна собівартість програмного забезпечення	$С_п$	4711
9. Плановий прибуток	$П_р$	2591
10.Ціна підприємства $Ц_п=С_п+П_р$	$Ц_п$	7302
11.Податок на додану вартість $ПДВ = 0.01 \cdot Н_{дв} \cdot Ц_п$	$ПДВ$	1460,4
12.Відпускна ціна програмної продукції Ц = Цп + ПДВ	$Ц$	9893

7.6 Визначення об’єму капітальних вкладень у споживача програмної продукції

Об’єм капітальних вкладень у споживача програмної продукції визначаємо на основі балансової вартості основних фондів, яка враховує ціну, транспортно-заготівельні витрати, вартість будівель, монтажних та пусконалагоджувальних робіт, а також витрати на випробування у виробничих умовах. Результати розрахунків зводимо у таблицю 7.10.

Таблиця 7.10 – Розрахунок об’єму капітальних вкладень у споживача програмної продукції

Найменування капітальних вкладень	Сума за варіантами, грн	
	Базовий	Новий
Вартість програмної продукції	–	9893
Всього капітальних витрат	–	9893

7.7 Визначення експлуатаційних витрат

Експлуатаційні витрати у споживача програмної продукції визначаємо при умові роботи підсистеми на протязі року. Результати зводимо до таблиці 7.11.

Таблиця 7.11 – Розрахунок експлуатаційних витрат у споживача програмної продукції

Найменування статей витрат	Позначення	Сума витрат за варіантами, грн.	
		Базовий	Новий
1. Витрати на технічне обслуговування	$З_р$	19290	3617
2. Витрати на електроенергію	$З_{ел}$	2099	1049
3. Витрати на амортизацію	$З_{ам}$	0	2473
Всього витрат за рік	$І$	21389	7139

Витрати на оплату праці:

$$З_р = Т_р \cdot З_г \cdot (1 + 0{,}01 \cdot Н_q) \cdot (1 + 0{,}01 \cdot Н_с), \qquad (7.23)$$

де Тр – кількість годин обслуговування системи за рік, *год.*; Зг – заробітна плата обслуговуючого персоналу, *грн / год*.

Після купівлі нового програмного забезпечення кількість профілактичних годин робіт зменшилася з 800 годин на рік до 150 годин на рік, тому витрати на технічне обслуговування зменшилися з

$$З_{р\,баз} = 800 \cdot 16 \cdot 1{,}1 \cdot 1{,}37 = 19290 \; грн.$$

до

$$З_{р\,нов} = 150 \cdot 16 \cdot 1{,}1 \cdot 1{,}37 = 3617 \; грн.$$

Витрати на електроенергію визначаються з урахуванням споживаємої потужності (Пел) в кіловатах, часу експлуатації технічних засобів (Тр) в годинах та ціни однієї кіловат-години (Цел).

$$З_{ел} = П_{ел} \cdot Т_р \cdot Ц_{ел}. \qquad (7.24)$$

Зел баз=·0,475·2455·1,8 = 2099 грн

Зел нов=·0,475·1227·1,8 = 1049 грн

Витрати по амортизації визначаються на основі норм амортизаційних відрахувань, вартості програмної продукції і основних фондів. Для розрахунку складаємо таблицю 7.12.

Таблиця 7.12 – Розрахунок амортизаційних відрахувань

Групи основних фондів	Норма амортизації %	Балансова вартість, грн., за варіантами		Сума відрахувань, грн., за варіантами	
		Базовий	Новий	Базовий	Новий
Програмна продукція	25	–	9893	–	2473,25
Всього відрахувань	-	–	9893	–	2473,25

7.8 Визначення економічної ефективності програмної продукції

Економічна ефективність програмного забезпечення визначається для виготовлювача і споживача за такими показниками.

Величина економічного ефекту при виготовленні програмної продукції, розраховуємо за формулою

$$E_в = (Ц_п - C_п) \cdot N_е - \sum_{i=1}^{m} E_{p_m} \cdot K_{p_m}, \quad (7.25)$$

де: $K_р$ – балансова вартість основних фондів розробника, грн.; $E_р$ – розрахунковий коефіцієнт капіталовкладень.

$$Ев = (7302 - 4711) \cdot 40 - (0,05 \cdot 1408000 + 0,5 \cdot 114885 + +0,25 \cdot 33190 + 0,1 \cdot 40000) \cdot 3/12 = 68605 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у виробника програмної продукції:

$$T_в = \frac{K_р^*}{(Ц_п - C_п) \cdot N_е}, \quad (7.26)$$

де: $K_р^*$ – балансова вартість основних фондів розробника без врахування вартості ОФ третьої групи, так як їх строк служби на порядок більший ніж період розробки ПЗ.

$$T_в = \frac{188075}{(7302 - 4711) \cdot 40 \cdot 12 / 3} = 0,5 \text{ років}.$$

Визначимо величину економічного ефекту у користувача програмної продукції за формулою:

$$E_{сп} = (I_б - I_н) - E_н(K_н - K_б), \quad (7.27)$$

де $I_б$, $I_н$ – величина експлуатаційних витрат за базовим и новим варіантом відповідно, $K_б$, $K_н$ – об'єм капітальних вкладень за варіантами, що порівнюються

$$E_{сп} = (21389 - 7139) - 0,25 \cdot 9893 = 11776,75 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у споживача програмної продукції за рахунок зниження експлуатаційних витрат

$$T_{сп} = \frac{K_н - K_б}{I_б - I_н} \quad (7.28)$$

$$T_{сп} = \frac{9893}{21389 - 7139} = 0,7 \ року$$

Показники економічної ефективності програмної продукції зводимо до таблиці 7.13.

Таблиця 7.13 – Показники економічної ефективності програмної продукції

Найменування показників	Одиниця виміру	Величина
1. Кількість екземплярів програми	Прим.	20
2. Повна собівартість розробленої програми	Грн	4711
3. Ціна розробленої програми	Грн.	7302
4. Плановий прибуток від реалізації розробленої програми	Грн.	2591
5. Рентабельність програмної продукції	%	55
6. Об'єм додаткових капітальних вкладень у виробника програмної продукції	Грн.	1596075
7. Загальний прибуток від реалізації програмної продукції	Грн.	103640
8. Величина економічного ефекту при виготовлені програмної продукції	Грн.	68605
9. Період окупності додаткових капітальних вкладень у виробника програмної продукції	Років.	0,5
10.Об'єм додаткових капітальних вкладень у споживача програмної продукції	Грн.	9893
11.Величина економічного ефекту у користувача програмної продукції	Грн.	11776,75
12.Період окупності додаткових капітальних вкладень у користувача програмної продукції	Років	0,7

7.9 Висновки

Розроблена програма економічно вигідна. За рахунок впровадження програмного забезпечення досягається скорочення часу обробки інформації, підвищується культура праці, підвищення якості приймаючих управлінських рішень.

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Шкідливі і небезпечні фактори при роботі з комп'ютером

Протягом усієї історії людство приділяє прискіпливу увагу безпеці життя і охорони праці, як її складової частини.

Законом України "Про охорону праці" регламентуються загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби польців та м'язи рук) при роботі з клавіатурою мають теж істотне навантаженням. До шкідливих факторів, які впливають на р обитників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

При розгляді шкідливих чинників роботи програмістів та інших спеціалістів ІТ будемо керуватись наступними нормативно-правовими актами: «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98, та

«Правила охорони праці під час експлуатації електронно-обчислювальних машин» НПАОП 0.00-1.28-10,

Умови праці програміста включають наступні фактори:

– параметри повітряного середовища в приміщенні;

– вентиляція приміщення;

– освітлення приміщення;

– параметри повітряного середовища в приміщенні, тощо.

Щоб запропонувати заходи щодо зменшення впливу комп'ютера на організм програміста визначемо фактори, які можуть викликати професійне захворювання і впливають на працездатність програміста,

Програміст працює з електронно-обчислювальною машиною (ЕОМ) та іншим обладнанням, яке є джерелом небезпеки ураження електричним струмом. Так як робота програміста характеризується істотним зоровим навантаженням, то вимагає належного освітлення. Так як програміст постійно перебуває в приміщенні, тому для комфортних умов праці в цьому приміщенні необхідно створити належний мікроклімат.

При роботі з використанням ЕОМ відзначають наступні небезпечні та шкідливі фактори:

– ризик виникнення надзвичайних ситуацій природного або штучного характеру на об'єкті або території.

– ризик виникнення пожежі;

– негативний вплив на органи зору людини;

– ризики ураження електричним струмом;

– недостатня, або надмірна освітленість робочого місця;

– монотонність праці;

– електромагнітні (у т.ч. високочастотні) електромагнітні випромінювання (коливання);

– несприятливі мікрокліматичні умови;

– нервово-емоційна напруженість праці;

– інтелектуальні навантаження;

– невідповідність ергономічних показників робочого місця діючим вимогам;

– шуми;

– статичні навантаження на кістково-м’язовий апарат;

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	4,4
Довжина	5,1
Висота	2,8

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	7,48
Обсяг, V	$м^3$	не менше 20.0	20,9

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

У зазначеному приміщенні працює 3 людей. За даними, які наведено у табл. 8.1, та табл. 8.2, можна зробити висновок, що площа та об’єм приміщення у розрахунку на одно робоче місце програміста відповідають нормативним вимогам (Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи

з екранними пристроями», ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин».

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-23	45-55	0,1
Тепла	23-25	50-70	0,1	23-24	50-65	0,1

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер HP 1100, електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

Роботу працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5 – 28 – 2006 р можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 лк. Крім того все поле зору повинне бути освітлено достатньо рівномірно – ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.3 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень;

Таким чином можна припустити, що основною причиною можливого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: наочне знайомство персоналу з шляхами для евакуації людей із приміщення відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення та залулення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

8.5 Розрахункова частина

Проведемо розрахунок штучного освітлення за методом коефіцієнта використання світлового потоку для приміщення ширина якого складає 4,4 м, довжина – 5,1 м, висота – 2,8 м.

Для того, щоб визначити потрібну кількість світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F=ESKZ/n,$$

де:

F – світловий потік, що розраховується, Лм;

E – нормована мінімальна освітленість, Лк; E = 300 Лк;

S – площа освітлюваного приміщення (у нашому випадку S=4,4 х 5,1 = 22,44 м^2);

Z – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1.1... 1.2, в нашому випадку Z = 1,1);

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку K = 1,5);

n – коефіцієнт використання світлового потоку, (відношення світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{ст\ ін.}$) і стелі ($\rho_{ст\ елі}$), значення коефіцієнтів дорівнюють $\rho_{ст\ ін}$ = 50% і $\rho_{ст\ елі}$ = 50%.

Обчислимо індекс приміщення за формулою:

$$i=S/(h(A+B)),$$

де:

S – площа приміщення, S = 22,44 м^2;

h – розрахункова висота підвісу, h = 2,8 м;

A – ширина приміщення, A = 4,4 м;

B – довжина приміщення, B = 5,1 м.

Підставимо всі значення у формулу та визначимо індекси приміщення: i=0,84.

Знаючи індекс приміщення, за знаходимо n = 0,37 (з табличних даних коефіцієнтів використання світлового потоку (n) світильників з відповідним типом лампам). Підставимо всі значення у формулу, визначемо світловий потік: F=30021 Лм.

Будемо використовувати лампи TL-F 12 36W LED 6000K IP65, світловий потік яких $F_л$ = 3000 Лм.

Число ламп визначається по формулі:

$$N=F/F_л$$

де:

F – світловий потік,

$F_л$ – світловий потік однієї лампи.

Підставимо всі значення у формулу та визначимо індекси приміщення:

$$N= 30021/ 3000=10 \text{ шт.}$$

Приймаємо необхідну кількість ламп 10 шт.

8.5 Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва в цілому.

З цих міркувань було здійснено аналіз приміщення, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров'я програміста. Розроблено заходи з охорони праці.

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи хмарного антивірусного забезпечення.

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

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів хмарного антивірусного забезпечення.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем хмарного антивірусного забезпечення.

– Досліджена система хмарного антивірусного забезпечення.

– На основі отриманих результатів досліджень створена програмна реалізація системи хмарного антивірусного забезпечення.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання хмарного антивірусного забезпечення.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

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 10.4. Дана мова програмування дозволяє найбільш ефективно обробляти дані. 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 й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows XP/Vista/7/8/10.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм RC5.

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.

Розроблена програма має реальний економічний ефект від її впровадження у виробництво у сумі 11776,75 грн. З урахуванням вартості розробки програми та обладнання, строк окуплення становить 0,7 роки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Смірнов С.А. Дослідження та програмна реалізація системи хмарного антивірусного забезпечення // Збірник праць молодих науковців ЦНТУ. – Вип. 12. – Кропивницький: ЦНТУ, 2022.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. 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.

12. 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.

13. 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.

14. 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.

15. 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.

16. Смирнов С. А. Исследование математических моделей технологии распространения компьютерных виру сов / А. А. Смирнов, Мохамад Абу Таам Гани, С. А. Смирнов // Актуал ьні 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.

17. 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.

18. 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.

19. 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.

20. 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.

21. 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.

22. 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.

23. 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.

24. 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.

25. 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» (ІТ & І): зб. тез ІІ міжнар. наук.-практ. конф., м. Київ, 3-5 листопада 2015 р. – К.: КНУ ім. Тараса Шевченка, 2015. – С. 65-67.

26. 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 ІІ междунар. научно-практ. конф., г. Алматы, Казахстан, 3-4 декабря 2015 г. – Алматы: КазНИТУ им. К.И. Сатпаева, 2015. – С. 309-313.

27. 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.

28. 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 ІІ Міжнар. наук.-практ. конф., м. Київ, 24-27 лютого 2016 р. – К.: Європейський університет, 2016. – С. 140-142.

29. 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.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

35. 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.

36. 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 III міжнар. наук.-практ. конф., м. Харків, 28-30 кві. 2016 р. – Х.: ХННІ ДВНЗ «УБС», 2016. – С. 178-182.

37. 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.

38. 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.

39. 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.

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

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

42. 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.

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

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

45. 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.

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

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

48. Basabi Chakraborty. Simultaneous Search for Multiple Routes using Genetic Algorithm / IEEE lntemational Conference on Computational oltelligencs for Measurement System and Applications Boston. MA, USA, 14-16, July 2004, P. 77-80/

49. Barakat, E. Altman, and W. Dabbous. On TCP performance in a heterogeneous network: a survey // IEEE Communications Magazine. – 2000. – №38(1). – P. 40 - 46.

50. Carpenter G., A., Grossberg S. Pattern Recognition by Self -Organizing Neural Networks, Cambridge, MA, MIT Press, 1991.

51. Cloud security, Deep Dive series, August 2011 [Электронный ресурс]. – Режим доступа к ресурсу: http://www.slideshare.net/kimrenejensen/cloud-security-deep-dive-2011#14375029197881&fbinitialized

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування..... 2
2 Підстава для розробки..... 2
3 Мета та призначення розробки..... 2
4 Джерела розробки..... 2
5 Технічні вимоги..... 2
5.1 Вміст проекту..... 2
5.2 Показники призначення..... 3
5.3 Вимоги до функціональних характеристик..... 3
5.4 Вимоги до архітектури..... 3
5.5 Вимоги до надійності..... 3
5.6 Умови експлуатації..... 4
5.7 Вимоги до складу та параметрів технічних засобів..... 4
5.8 Вимоги до інформаційної і програмної сумісності..... 4
5.8.1 Обладнання..... 4
5.8.2 Мова програмування..... 4
5.8.3 Вхідні дані..... 5
5.8.4 Вихідні дані..... 5
6 Вимоги до програмної документації..... 5
7 Економічні вимоги..... 5
8 Вимоги щодо охорони праці..... 5
9 Перелік документів, що розробляються..... 6
10 Етапи розробки..... 6
11 Порядок контролю та приймання..... 6

					ВКРМ-123.21.0002.00.00.ТЗ		
Вим.	Арк.	№ документа	Підпис	Дата			
Розробив		Смірнов С.А.			Дослідження та програмна реалізація системи хмарного антивірусного забезпечення	Літ.	Аркуш 1 / Аркушів 6
Перевірив		Якименко Н.М.				М	
Н. Контр.		Гермак В.С.				ЦНТУ КІ-20М3	
Затв.		Смірнов О.А.					

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи хмарного антивірусного забезпечення.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 41-13 від 02.08.2021 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи хмарного антивірусного забезпечення.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) 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 з ОС та з користувачем;

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи хмарного антивірусного забезпечення;

– 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 XP/Vista/7/8/10 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 XP/Vista/7/8/10.

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 10.4.

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 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2021 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинен бути розглянутий аналіз санітарно-гігієнічних умов праці на робочому місці програміста.

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 109 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

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

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 10.12.2021 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 22.12.2021 р.

Додаток Б

(обов'язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

другим (магістерським) рівнем вищої освіти

_____ Якименко Н.М.

Дослідження та програмна реалізація
системи хмарного антивірусного забезпечення

Лістинг програми

Код документу 12

Носій: CD/DVD-диск

Загальна кількість аркушів: 42

Літера: РП

Кропивницький – 2021 року

Файл debug.dpr – головний файл проекту

```
program debug;
// Список підключаємих модулів
uses
  Forms,
  SysUtils,
  avKernel in '..\CloudAV Scanner Modues\avKernel.pas',
  avTypes in '..\CloudAV Scanner Modues\avTypes.pas',
  avMonitor in '..\CloudAV Scanner Modues\avMonitor.pas',
  avScanner in '..\CloudAV Scanner Modues\avScanner.pas',
  avHex in '..\CloudAV Scanner Modues\avHex.pas',
  avDataBase in '..\CloudAV Scanner Modues\avDataBase.pas',
  avHash in '..\CloudAV Scanner Modues\avHash.pas',
  avExt in '..\CloudAV Scanner Modues\avExt.pas',
  avAPI in '..\CloudAV Scanner Modues\avAPI.pas',
  avConfig in '..\CloudAV Scanner Modues\avConfig.pas',
  avShield in '..\CloudAV Scanner Modues\avShield.pas',
  langs in 'langs.pas',
  uMain in 'uMain.pas' {MainForm},
  uSelInfo in 'uSelInfo.pas' {InformationForm},
  uOptions in 'uOptions.pas' {OptionsForm},
  uPluginInfo in 'uPluginInfo.pas' {PluginAPIForm},
  uAddPath in 'uAddPath.pas' {AddUserPathForm},
  AboutFrm in 'AboutFrm.pas' {AboutForm},
  uSelDir in 'uSelDir.pas' {SelDirFrm},
  uMessage in 'uMessage.pas' {MessageFrm},
  uHideForm in 'uHideForm.pas' {HideForm},
  uMonitor in 'uMonitor.pas' {MonitorForm},
  uInfectedAction in 'uInfectedAction.pas' {ActionForm},
  uSplash in 'uSplash.pas' {SplashForm};

{$R *.res}

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

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

Файл uMain.pas – основна програма

```
unit uMain;

interface

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

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

  type
  TIconType = (itSmall, itLarge);

  type
  NotifyIconData_50 = record
   cbSize: DWORD;
   Wnd: HWND;
   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;
    ScanPathesTab: TTabSheet;
    ScanningTab: TTabSheet;
    ReportTab: TTabSheet;
    BottomPanel: TPanel;
    ScanBTN: TButton;
    SaveBTN: TButton;
    PathList: TListView;
    Bevel1: TBevel;
    ScanList: TListView;
    ReportMemo: TMemo;
    ImageList: TImageList;
    DrivesImg: TImageList;
    PathMenu: TPopupMenu;
    AddFolder: TMenuItem;
    DeletePath: TMenuItem;
```

```
N1: TMenuItem;
Reftesh: TMenuItem;
SaveDialog: TSaveDialog;
XPManifest: TXPManifest;
Bevel4: TBevel;
DelMenu: TPopupMenu;
Del: TMenuItem;
TrayMenu: TPopupMenu;
mnuShowCloudAVScanner: TMenuItem;
mnuHideCloudAVScanner: TMenuItem;
N2: TMenuItem;
mnuOptions: TMenuItem;
N4: TMenuItem;
mnuHelp: TMenuItem;
mnuAbout: TMenuItem;
N7: TMenuItem;
mnuExit: TMenuItem;
Image1: TImage;
TopPn: TPanel;
Bevel3: TBevel;
Image2: TImage;
RightPanel: TPanel;
ExitBTN: TButton;
TopRightPanel: TPanel;
Image3: TImage;
VersionLabel: TLabel;
AboutBTN: TLabel;
DelAll: TMenuItem;
ApplicationEvents: TApplicationEvents;
ProgressBar: TProgressBar;
ScanTopBtn: TLabel;
ScanMenu: TPopupMenu;
mnuSelScanPath: TMenuItem;
mnuShowReport: TMenuItem;
N12: TMenuItem;
OptionTopBtn: TLabel;
PCTopBtn: TLabel;
mnuCloudAVProcessControl: TMenuItem;
N19: TMenuItem;
mnuPCShow: TMenuItem;
N21: TMenuItem;
mnuPCRun: TMenuItem;
mnuPCPause: TMenuItem;
mnuPCStop: TMenuItem;
mnuScanStart: TMenuItem;
mnuStopScan: TMenuItem;
N13: TMenuItem;
mnuSaveReport: TMenuItem;
N26: TMenuItem;
mnuGoToTray: TMenuItem;
SOURCESTRING: TListBox;
LabelPanel: TPanel;
ScanFile: TLabel;
procedure DelAllClick(Sender: TObject);
procedure FormResize(Sender: TObject);
procedure ExitBTNClick(Sender: TObject);
procedure ScanListDblClick(Sender: TObject);
procedure ScanBTNClick(Sender: TObject);
procedure InitScannerKernel;
Procedure StartScan(Parametr: String);
procedure SaveBTNClick(Sender: TObject);
procedure DeletePathClick(Sender: TObject);
procedure RefteshClick(Sender: TObject);
procedure AddFolderClick(Sender: TObject);
function CreateDrivesList(ListView: TListView): boolean;
procedure AboutBTNClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure HelpBTNClick(Sender: TObject);
```

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

    MonFileCN        : Integer;
    MonFileInfected  : Integer;

    Path          : TStringList;
    DeActiveTray  : Boolean;

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

        CloudAVMonitor      : String;
        CloudAVInit         : String;
        LoadAPI          : String;
        LoadDB           : String;
        CreateDrvList    : String;
        OptFileNotFnd    : String;
        LoadOptFile      : String;
        InitProcedures   : String;
        initShield       : String;
        ErrorInit        : String;
        LogBevel         : String;
        DBKnowledge      : String;
```

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

        MLoad           : String;
        MunLoad         : String;

    end;

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

```
implementation

uses uSelInfo, uOptions, uAddPath, AboutFrm, Math, uMessage, uHideForm,
  uMonitor, uInfectedAction, uPluginInfo;
{$R *.dfm}

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

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

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

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

procedure TMainForm.MineRestore(var Msg: TMessage);
begin
  if (Msg.Msg = WM_MINERESTORE) then
  begin
    mnuShowCloudAVScanner.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 := 'CloudAV 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_SCAN_SUBDIR;
  OptionsForm.SCNHEX.Checked          := OPT_USE_HEX_MODE;
  OptionsForm.SCNCRC.Checked          := OPT_USE_CRC_MODE;
  OptionsForm.SCNBIT.Checked          := OPT_USE_BYTE_MODE;

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

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

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

      if GETParamName(CloudAVConfig[i]) = 'PROCCONTROLAUTOMODE' then
```

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

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

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

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

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

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

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

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

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

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

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

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

function GetHDDSerial(ADisk : char): dword;
  var
  SerialNum : dword;
  a, b : dword;
  VolumeName : array [0..255] of char;
begin
  Result := 0;
```

```
  if GetVolumeInformation(PChar(ADisk + ':\'), VolumeName, SizeOf(VolumeName),
  @SerialNum, a, b, nil, 0) then
    Result := SerialNum;
end;

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

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

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

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

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

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

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

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

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

  MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,ScanEndBalloonText,'CloudAV
Scanner',bitInfo);
end;

//***Функція початку сканування***//

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

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

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

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

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

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

  if MES = MES_NONE then Exit;

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

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

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

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

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

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

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

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

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

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

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

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

  if MES = MES_ONSCANEXECUTE then
    OnScanStart;

  if MES = MES_ONSCANCOMPLETE then
```

```
    OnScanComplete;

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

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

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

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

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

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

end;

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

procedure TMainForm.InitScannerKernel;
  var
  i:integer;
begin

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

```
        ScanEnd          := SOURCESTRING.Items[14];
        PrepareToScan    := SOURCESTRING.Items[15];
        FileIgnor        := SOURCESTRING.Items[16];
        FileIfect        := SOURCESTRING.Items[17];
        FileScanned      := SOURCESTRING.Items[18];
        DataScanned      := SOURCESTRING.Items[19];
        IGNORED          := SOURCESTRING.Items[20];
        SKIPBYSIZE       := SOURCESTRING.Items[21];
        INFECTED         := SOURCESTRING.Items[22];
        STOPB            := SOURCESTRING.Items[23];
        RETURNB          := SOURCESTRING.Items[24];
        SCANB            := SOURCESTRING.Items[25];
        SCNFILE          := SOURCESTRING.Items[26];
        FileDel          := SOURCESTRING.Items[27];
        FileNotDel       := SOURCESTRING.Items[28];
        PATHNOSEL        := SOURCESTRING.Items[29];
        SysMenu          := SOURCESTRING.Items[30];
        NfoCloudAVScanner   := SOURCESTRING.Items[31];
        NfoCloudAVKernel    := SOURCESTRING.Items[32];
        NfoCloudAVBuild     := 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];
        CloudAVCloseDlg     := SOURCESTRING.Items[44];
        AllreadyInScan   := SOURCESTRING.Items[45];
        ProcControlSt    := SOURCESTRING.Items[46];
        ErrorKillProc    := SOURCESTRING.Items[47];
        PCActive         := SOURCESTRING.Items[48];
        PCPaused         := SOURCESTRING.Items[49];
        PCStoped         := SOURCESTRING.Items[50];
        PCInit           := SOURCESTRING.Items[51];
        PCPause          := SOURCESTRING.Items[52];
        PCStop           := SOURCESTRING.Items[53];
        PCRestore        := SOURCESTRING.Items[54];
        LASTDBDATA       := SOURCESTRING.Items[55];
        DATABASEdate     := SOURCESTRING.Items[56];
        BASELOADED       := SOURCESTRING.Items[57];
        DBerrorI1        := SOURCESTRING.Items[58];
        DBerrorI2        := SOURCESTRING.Items[59];
        DBerrorI3        := SOURCESTRING.Items[60];

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

        InitKernel(CloudAVKernelMessageAPI);
        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)+' '+NfoCloudAVScanner
+CloudAVScannerVS);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoCloudAVKernel
+GetKernelVersion);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+' '+NfoCloudAVBuild
+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('*','CloudAV.Scan',SysMenu,ParamStr(0)+' %1');
    OptionsForm.FileTAddAction('Directory','CloudAV.Scan',SysMenu,ParamStr(0)+'
%1');
    OptionsForm.FileTAddAction('Drive','CloudAV.Scan',SysMenu,ParamStr(0)+'
%1');
  end else
  begin
    OptionsForm.FileTDelAction('Drive','CloudAV.Scan');
    OptionsForm.FileTDelAction('Directory','CloudAV.Scan');
    OptionsForm.FileTDelAction('*','CloudAV.Scan');
  end;
end;

//***Функція початку сканування***//

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

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

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

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

end;

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
  end;
end;

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

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

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

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

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


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

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

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

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

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

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

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

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

  try
    AboutForm.ShowModal;
  except
  end;
end;

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

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

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

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

procedure TMainForm.ScanTopBtnClick(Sender: TObject);
begin

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

end.
```

Файл uMonitor.pas – монітор (контроль процесів)

```
unit uMonitor;

interface

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

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

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

uses uMain, uInfectedAction, uOptions;
//***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,'SeDebugPrivilege',Priv.Privileges[0].Luid);
```

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

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

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

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

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

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

```
    end;
end;

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

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

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

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

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

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

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

procedure TMonitorForm.ClosePCClick(Sender: TObject);
begin
  Close;
end;

procedure TMonitorForm.Timer1Timer(Sender: TObject);
  var
  ss,mm,hh:String;
begin
  if isMonRun then
    if Not MonPaused then
      Label7.Caption := MainForm.PCActive
    else
      Label7.Caption := MainForm.PCPaused;

  if not isMonRun then
    Label7.Caption := MainForm.PCStoped;

  if isMonRun then
  if Not MonPaused then
```

```
  begin
    s:=s+1;
    if s = 59 then
    begin
      s:=0;
      m:=m+1;
    end;
    if m = 59 then
    begin
      m:=0;
      h:=h+1;
    end;
    ss:=inttostr(s);
    mm:=inttostr(m);
    hh:=inttostr(h);
    if length(ss) = 1 then ss:='0'+ss;
    if length(mm) = 1 then mm:='0'+mm;
    if length(hh) = 1 then hh:='0'+hh;
    Label8.Caption := hh+':'+mm+':'+ss;
  end;
end;

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

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

```
  end;
end;

end.
```

Файл uAddPath.pas – додавання шляхів сканування

```
unit uAddPath;

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 uMain, uOptions, 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 uInfectedAction.pas – вибір дії над інфікованим об'єктом

```
unit uInfectedAction;

interface

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

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

var
  ActionForm: TActionForm;

implementation

uses uMain, uOptions;

{$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 uOptions.pas – параметри антивірусу

```
unit uOptions;

interface

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

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

```
    Bevel4: TBevel;
    optTabPC: TTabSheet;
    optPCLabel: TLabel;
    Bevel8: TBevel;
    PCAutoLoad: TCheckBox;
    PCAutoKill: TCheckBox;
    PCAutoAction: TCheckBox;
    PCDelInfect: TRadioButton;
    PCSkipInfect: TRadioButton;
    optPCInfoLabel: TLabel;
    SHOWBALOONHINT: TCheckBox;
    procedure ApplyBTNClick(Sender: TObject);
    procedure optTabOtherShow(Sender: TObject);
    procedure optTabFilterShow(Sender: TObject);
    procedure optTabPathesShow(Sender: TObject);
    procedure optTabModulesShow(Sender: TObject);
    Procedure SaveOptions;
    procedure CanselBTNClick(Sender: TObject);
    procedure APIListDblClick(Sender: TObject);
    procedure AddBTNClick(Sender: TObject);
    procedure DelBTNClick(Sender: TObject);
    procedure EditBTNClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure EditSaveReportBTNClick(Sender: TObject);
    procedure FileTAddAction(key, name, display, action: String);
    procedure FileTDelAction(key, name: String);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure optTabMainShow(Sender: TObject);
    procedure ChangeReg(StrName: ShortString; delete: boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  OptionsForm: TOptionsForm;

implementation

uses uMain, uPluginInfo, uAddPath, 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('Scanner',False);
  end else
  begin
    ChangeReg('Scanner',True);
  end;

 //******//

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
unit AboutFrm;

interface

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

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

var
  AboutForm: TAboutForm;

implementation

uses uMain;

{$R *.dfm}

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

end.
```